Sistema d'estereovisió amb una càmera

Carles Royán Salvatella

4 de setembre de 2006

Índex

Índex de figures

Capítol 1

Introducció

1.1 Objectius principals

El grup de visió per computador i robòtica de l'IIIA porta treballant en reconstrucció tridimensional des de 1994 amb tècniques de triangulació, emprant tant la projecció de llum estructurada codificada, passant per la projecció de llum làser fins a tècniques d'estereovisió amb dues o més càmeres.

Els sistemes de visió estèreo es basen en la reconstrucció per triangulació a partir de dues càmeres, permetent la representació d'objectes del món real en tres dimensions. L'objectiu d'aquest projecte consisteix en dissenyar i implementar un sistema estèreo amb una sola càmera amb dos petits vidres d'alta transmissivitat davant de la lent, utilitzant la teoria clàssica desenvolupada a partir de dues càmeres. D'aquesta forma obtindrem un sistema molt més compacte que en el cas de tenir dues càmeres, que serà apte per entorns molt reduïts i per escenes molt properes. Es desenvoluparà amb C++, QT i LTI-Lib un sistema que permeti capturar imatges d'una càmera i, gràcies a la geometria epipolar i la teoria de la triangulació, representar en tres dimensions els objectes del món real capturats. Sempre treballarem sobre imatges en escala de grisos, ja que no estudiarem pas ni la textura ni cap altre paràmetre relacionat amb el color.

El projecte es centrarà en mostrar la possibilitat d'aplicar la mateixa teoria que intervé en sistemes estèreo amb dues càmeres en un que consti només d'una càmera, utilitzant un materials i un maquinari econòmic. Per tant, no perseguiriem la precisió i repetivitat en els resultats obtinguts, però si que sigui un sistema de baix cost amb uns resultats acceptables.

Els sistemes de visió estèreo amb dues càmeres estan molt estudiats, però sistemes estèreo on només intervingui una càmera no són tan usuals i aquest projecte servirà per obtenir uns resultats sobre l'aplicació de la teoria estèreo clàssica en aquests sistemes i els seus possibles avantatges i aplicacions.

1.2 Altres objectius

Entre els altres objectius secundaris, podem trobar aprendre tot el necessari per construir un sistema de visió (calibració de les càmeres, algorismes necessaris, matemàtica i mètodes numèrics), així com entendre la visió estèreo clàssica en dues càmeres, ja que la seva teoria és la base per el sistema amb una sola càmera.

També hi ha l'objectiu d'aplicar el coneixement de moltes assignatures de la carrera d'informàtica, així com coneixements de l'anterior carrera d'informàtica tècnica, com ara: "Estructura de dades i algorismes", "Mètodes numèrics", "Visió per computador", ...

Finalment, cal esmentar com a objectius aprendre a extreure informació necessària de llibres, manuals molt extens, *papers*, així com a expressar de manera escrita, organitzada i estructurada (la memòria) tot el treball realitzat durant la durada del projecte.

1.3 Estructura de la memòria

La memòria està estructurada en tres parts: **Fonaments teòrics**, **Aplicació pràctica** i **Conclusions finals**. En la primera s'explicarà la teoria que hi ha darrera el desenvolupament del present projecte. En la segona part es detallarà què s'ha desenvolupat, tot relatant els problemes que hagin pogut sorgir i com s'han solventat. Finalment, s'exposaran les conclusions a les quals s'ha pogut arribar amb el treball realitzat.

El nostre sistema estèreo està composat per diferents etapes, cadascuna d'elles explicades en el seu corresponent capítol. Primerament, adquirim un parell d'imatges d'una càmera. Per poder fer aquesta adquisició correctament necessitem que la càmera estigui calibrada, com s'exposa en el capítol 2. Un cop tenim les imatges, hem d'extreure els punts característics, en el nostre cas les cantonades, utilitzant el detector de cantonades Harris del capítol 3. Llavors, hem d'afrontar el principal problema dels sistemes estèreo: les correspondències. Hem d'obtenir les correspondències entre les cantonades de la primera imatge amb les de la segona. Per aquesta tasca, farem ús de la geometria epipolar detallada en el capítol 4. El pas final, és triangular aquestes correspondències per a trobar les coordenades del punt 3D a l'espai. En el capítol 5 trobarem les tècniques de triangulació apropiades.

Un cop s'ha vist els fonaments teòrics necessaris per el sistema estèreo, a la segona part abordarem el desenvolupament del prototip. Degut a la complexitat intrínseca de l'estereovisió, primerament hem desenvolupat un sistema estèreo *clàssic*, format per dues càmeres, com podem veure en el capítol 7. Degut a que la base matemàtica no varia, els algorismes desenvolupats ens serveixen igualment per el sistema estèreo amb una única càmera, que presentem en el capítol 8. En aquest capítol presentarem les peculiaritats i els problemes trobats.

Finalment, en l'última part mostrarem les conclusions a les quals hem pogut arribar, treballs futurs per a millorar el prototip, aplicacions possibles, entre d'altres. També s'inclou dos apèndixs en els quals s'explica la notació matemàtica utilitzada àmpliament al llarg de la memòria, així com mètodes numèrics i conceptes matemàtics que hagin sorgit en el desenvolupament de la memòria. La bibliografia resta com a cloenda, en la qual apareixen les fonts de la qual s'ha tret la informació i s'han realitzat cites en el present document.

Part I

Fonaments teòrics

Capítol 2

Calibració de càmeres

2.1 Introducció

En aquest capítol s'explicarà, el procés de formació de la imatge des del punt de vista geomètric que aborda la qüestió d'on es projecta en la imatge cada punt del món real. Veurem també com calibrar la càmera per establir una relació entre els punts reals i els punts que veiem en la imatge, així com eliminar la distorsió provocada per la lent.

2.2 El model de la càmera del forat d'agulla o pin-hole

Aquest és el model més simple de la lent, en el qual cada punt d'un objecte de l'espai es projecta en un punt d'un pla anomenat pla imatge a on es forma la imatge de la escena 3D, com es pot observar en la figura 2.1. Com que el forat és d'un diàmetre infinitesimal, dels raigs de llum que procedeixen del punt 3D només un passarà per aquest, per la qual cosa sobre cada punt del pla imatge incideix només un raig de llum. D'acord amb aquest procés, tots els punts de l'espai estaran sempre perfectament enfocats.

Per la seva simplicitat, aquest model és molt utilitzat en un gran nombre d'aplicacions i processos de visió per computador, i a més serà el que també s'aplicarà en el procés de calibració de la càmera del qual parlarem més endavant.

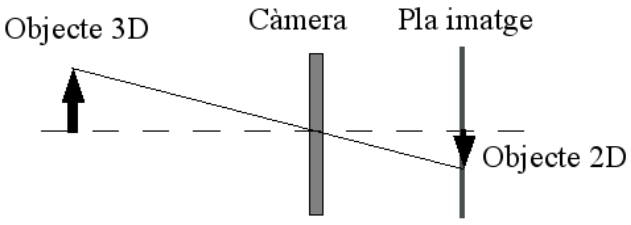


Figura 2.1: Projecció d'un punt d'un objecte 3D en el pla imatge utilitzant el model "pin-hole".

No obstant, des d'un punt de vista radiomètric, és incapaç d'explicar la formació d'imatges, ja que un únic raig procedent de l'espai no poseeix l'energia suficient per excitar el sensor que capta la imatge. Si el forat fos més gran permetria una quantitat de rajos suficients, però la imatge restaria desenfocada.

A més, l'inconvenient fonamental d'aquest model és que no contempla la majoria de característiques i paràmetres típics d'una òptica com: el desenfoc dels objectes, el control de la quantitat de llum incident en el sensor, etc. En realitat, l'únic paràmetre que modela és la distància focal, que controla els augments o "zoom".

Existeixen altres models físics més complets com ara el model de la lent prima que permet especificar la majoria dels paràmetres de les òptiques reals. No obstant, no està tant estès degut a que el model del forat d'agulla és molt més simple i els resultats que proporciona són prou acurats.

2.3 Paràmetres d'interès en la captura d'imatges

En el processament d'imatges capturades per càmeres hi ha uns paràmetres que ens resulten de vital interès a l'hora de tractar la informació rebuda. A continuació explicarem els principals.

- Distància focal: és la distància "f" entre el centre òptic de la lent i el punt en que la imatge queda projectada. Determina la mida de la imatge formada en el sensor (efecte "zoom"). Com que la mida del sensor és fixe, també determina l'angle de visió, és a dir, a major distància focal tindrem menor angle de visió.

- Distància d'enfocament: És la distància, mesurada des del pla de la lent, a la que es troba el pla del espai projectat en el pla imatge.

- Profunditat de camp: Determina la amplada de la zona enfocada, és a dir, és el rang de distància davant i darrera de l'objecte que sembla que estigui enfocat en el pla imatge.

- Obertura: Les òptiques venen proveïdes d'un sistema que regula el pas dels rajos de llum a través de les lents anomenat diafragma o iris. L'obertura especifica com d'obert o tancat està el diafragma, i per tant, la quantitat de llum que hi ha a la lent. Normalment es mesura en "F-Números"(F-number o f-stop), que es calcula dividint la distància focal per el diàmetre d'obertura de la lent, i el seu valor s'expressat per la lletra "F". Un exemple seria F3, que podria correspondre a una lent amb distància focal 60 mm i 20 mm.

- Distància de treball: És la distància, mesurada des de la òptica, a la que es troba l'objecte o els objectes que es desitgen capturar amb la càmera.

- Camp de visió: Es defineix com l'àrea observada per una càmera a la distància d'enfocament.

- Angle de camp: És l'angle d'observació de l'objecte que està enfocat.

- Factor d'augment: És la mida relativa dels objectes de la imatge en relació a la mida real en l'escena.
- Tipus de muntura: La muntura d'una òptica defineix la mida i el mecanisme d'acoblament a la càmera.

2.4 Geometria de la formació d'imatges

En aquesta secció s'explicarà la projecció dels objectes de la escena, és a dir, els objectes reals al pla de la imatge. Aquesta projecció es realitza mitjançant una transformació perspectiva, tenint en compte els diferents sistemes de referència involucrats.

2.4.1 Coordenades homogènies

Primerament, cal tenir clar el concepte de coordenades homogènies. Les coordenades homogènies d'un punt cartesià (X,Y,Z) es defineixen com (kX,kY,kZ,k), on k és una constant arbitrària diferent de 0. En el cas que k=0, les coordenades determinen un vector al infinit, i per tant, defineixen una direcció.

La conversió de coordenades homogènies a coordenades cartesianes es realitza fàcilment dividint les tres primeres coordenades homogènies per la quarta.

Si per exemple, tenim el punt x, definit, com:

$$\mathrm{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

En coordenades homogènies seria:

$$\mathrm{x}_h = \begin{bmatrix} kx \\ ky \\ kz \\ k \end{bmatrix}$$

2.4.2 Model de la càmera

L'objectiu d'establir un model per a les càmeres és poder obtenir la relació que hi ha entre els punts reals i els punts de la imatge i viceversa, per la qual cosa s'aproxima geomètricament les transformacions en posició i orientació de l'escena a la projecció del pla imatge. La forma més senzilla és modelar-ho tenint en compte transformacions lineals, obviant la distorsió provocada per les lents. El model proposat per en Hall ens serveix com a base per establir aquesta relació.

Com ja s'ha comentat, l'objectiu és establir una relació entre els punts 3D de l'escena i els seus corresponents punts projectats en el pla imatge. Aquesta relació la podem aproximar amb una matriu de transformació com la que podem veure en l'equació 2.1.

$$\begin{pmatrix} s^I x_d \\ s^I y_d \\ s \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{pmatrix} \begin{pmatrix} {}^W x_w \\ {}^W y_w \\ {}^W z_w \\ 1 \end{pmatrix} \tag{2.1}$$

Llavors, podem veure que un punt qualsevol X_w, amb les coordenades (x_w, y_w, z_w), expressat amb el sistema referència de l'escena i sistema mètric (també anomenat sistema referència món) quedaria expressat com a ${}^{W}X_w$. Aquest punt, aplicant la matriu de transformació proposada per Hall, esdevindria al punt ${}^{I}X_d$, amb coordenades (x_d, y_d), que seria expressat amb el sistema de coordenades de la imatge i en píxels.

Però trobar aquesta matriu és una tasca complexa, que es sol dividir en quatre passes, com es pot veure en la figura 2.2.

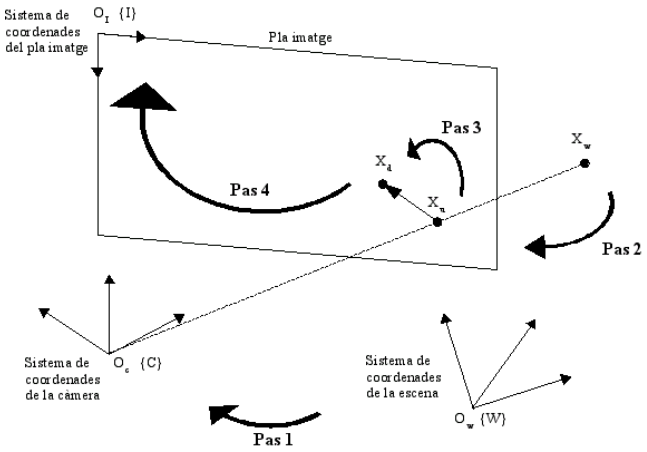


Figura 2.2: Relació geomètrica entre un punt 3D del món i la seva respectiva projecció 2D.

Expliquem amb continuació amb més detall els passos que es poden veure en l'anterior figura 2.2:

1. El primer pas consisteix en expressar el punt ${}^{W}X_w$ del sistema referència de la escena en el sistema de referència de la càmera, tot obtenint el punt ${}^{C}X_w$. Aquesta transformació es realitza mitjançant una matriu de rotació i un vector de translació.

2. A continuació, és necessari obtenir la projecció del punt ${}^{C}X_w$ al pla imatge, obtenint així el punt ${}^{C}X_u$, tot utilitzant una transformació projectiva.

3. El tercer pas té a veure amb la distorsió de la lent. S'obté el punt ${}^{C}X_d$ tot aplicant els paràmetres que modelen la distorsió de la lent sobre el punt ${}^{C}X_u$.

4. Finalment, cal transformar el punt des del sistema de coordenades mètric de la càmera al sistema de coordenades en píxels del la imatge, obtenint el punt ${}^{I}X_d$

Per poder obtenir aquesta matriu, així com altres paràmetres, com els de la distorsió de la lent, cal calibrar la càmera, tot seguint un procés de calibració.

2.5 Calibració de la càmera

La calibració de la càmera fa referència a la obtenció de tots aquells paràmetres que incideixen en la formació de la imatge, tant els anomenats geomètrics, que són els involucrats en el procés geomètric de generació de la imatge, com ens els radiomètrics, que tenen a veure amb la brillantor dels objectes projectats. D'aquest segon tipus són, per exemple, l'obertura de la òptica o el temps d'exposició i el guany de l'amplificador de la càmera. També, ens ajuda a corregir la distorsió ocasionada per l'òptica. Per a calibrar una càmera, fem servir un model específic d'aquesta, com s'ha comentat en l'anterior secció.

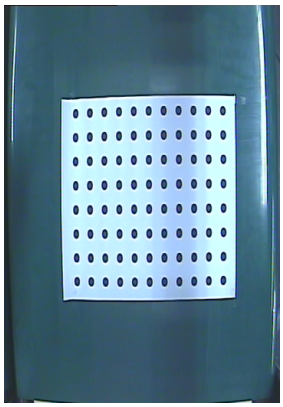


Figura 2.3: Exemple de distorsió provocada per la lent.

Nosaltres només tindrem en compte la calibració geomètrica, que és amb diferència la que té més interès pràctic per els problemes de visió generals i de la que en farem ús al llarg del present projecte.

2.5.1 Definició

La calibració de la càmera consisteix en determinar els paràmetres de la transformació entre punts 3D de l'escena i punts 2D de la imatge. Existeixen dos tipus de paràmetres que defineixen aquesta transformació:

1. Paràmetres intrínsecs: Són aquells que caracteritzen les propietats inherents de la càmera i de la òptica. Es solen considerar 4 paràmetres:

   - Distància focal: és la distància "f"entre el centre òptic de la lent i el punt en que la imatge queda projectada.
   - Desplaçament del centre de la imatge: u_0, v_0 són les components de la projecció del centre òptic O_c en el pla imatge.
   - Coeficient de distorsió: k_1 és el coeficient que modela la distorsió en la imatge ocasionada per la lent. Més endavant es veurà amb més detall aquest paràmetre.

   D'acord amb el model de "pin-hole", en un primer pas la imatge es forma sense cap mena de distorsió, tot projectant l'escena en el pla imatge. En segona instància, es desplaça a les coordenades $(u0, v0)$ de la projecció

del centre òptic i es distorsiona segons un model radial, definit per les expressions:

$$\begin{aligned} x_d + d_x &= x_u \\ y_d + d_y &= y_u \end{aligned} \tag{2.2}$$

on (x_d,y_d) són les coordenades del píxel distorsionat i (x_u,y_u) del píxel no distorsionat. d_x i d_y són els coeficients de desplaçament calculats en la calibració. És important recalcar que en aquestes equacions les coordenades distorsionades són les conegudes i no pas les altres. Quan abordem el mètode de calibració revisarem com es modela aquesta distorsió.

2. Paràmetres extrínsecs: Són els sis paràmetres que defineixen la posició i orientació de la càmera respecte al sistema de referència absolut:

   - Translació: $\mathbf{T}_x$,$\mathbf{T}_y$,$\mathbf{T}_z$
   - Rotació: angles α, β, γ

Només si la càmera està calibrada es possible establir una relació entre les coordenades 3D dels objectes i les seves corresponents projeccions 2D, i al revés. Per tant, en el nostre sistema estèreo serà imprescindible aquest pas.

2.6 Procediment general de calibració

Tot i que existeixen diferents mètodes de calibració d'una càmera, el procediment bàsic és el mateix en cadascun d'ells:

1. Determinar amb precisió un conjunt de punts tridimensionals.
2. Determinar les seves correspondents projeccions en la imatge.
3. Obtenir els paràmetres que millor resolen la correspondència entre els primers i els segons.

Els dos primers passos requereixen conèixer una sèrie de punts 3D i les seves corresponents projeccions en la imatge; són el que s'anomenen "punts de calibració". Depenent del tipus de mètode, aquests punts poden ser o no coplanaris. En el primer cas, els punts de calibració es prenen sobre una plantilla que permet el posicionament exacte d'aquests en l'espai, i la seva extracció automàtica i precisa en la imatge. Per facilitar el posicionament dels punts, i sempre que sigui possible, la plantilla es col·loca paral·lela al pla $x_w y_w$, aconseguint que tots els punts tinguin la mateixa z_w. En la 2.4 es mostren dues de les plantilles més utilitzades.

Per el cas de punts no-coplanaris el sistema és similar, tot i que s'utilitza un sistema on es posiciona la plantilla en diferents plans (vegeu 2.4).

Les característiques que s'exigeixen a un mètode de calibració són, fonamentalment, les següents:

- Autonomia: El procediment de calibració no tindria que necessitar de la intervenció de cap operador extern, com ara estimacions inicials o la selecció de certs paràmetres de manera manual.

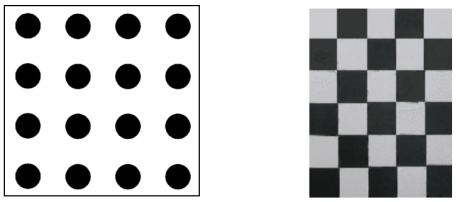


Figura 2.4: Algunes plantilles típiques de calibració.

- Precisió: Moltes aplicacions requereixen una gran precisió, i el mètode de calibració ha de proporcionar els resultats d'una forma prou acurada, depenent de l'aplicació corresponent del sistema.
- Eficiència: El procediment complet de calibració no tindria que incloure procediments de cost computacional elevat.
- Versatilitat: La tècnica de calibració tindria que operar de forma uniforme i autònoma en un ampli rang d'aplicacions, òptiques, nivells de precisió, etc.

Existeixen diversos mètodes de calibració, com ara el mètode de Tsai, el mètode de Weng o el mètode de Faugeras. Aquests mètodes es diferencien per les distorsions que tenen en compte, i per tant que modelen. El model de càmera proposat per Olivier Faugeras [Faugeras 1993] modela la calibració bàsica: posició de la càmera, la seva orientació i la projecció de la perspectiva. No modela la distorsió de la lent, però pot ésser incorporada la distorsió radial, com s'ha demostrat en treballs posteriors. El model de calibració proposat per Tsai [Tsai 1997] sí que incorpora la distorsió de la lent, la de forma radial, essent més complet que el proposat per Faugeras. Finalment, el model de Weng [Weng 1992] és el que proposa una calibració més acurada, tot modelant tres tipus de distorsió: distorsió radial, distorsió descentrada (que és dóna per el fet que el centre òptic de les lents no està aliniat correctament amb el centre de la càmera) i distorsió de la "lent prima"(que apareix per les imperfeccions en el disseny i manufactura de les lents així com en acoblament de les peces de la càmera).

2.7 Calibració de càmeres a partir del mètode de Faugeras amb distorsió radial

Passem a detallar el model de Faugeras. Aquest model de càmera calibra: posició de la càmera, la seva orientació i la projecció de la perspectiva, no tenint en compte la distorsió de la lent. No obstant, com es pot veure en [Salvi 1998], es pot introduir la distorsió radial de forma relativament simple. Nosaltres farem servir el model de Faugeras modificat per a tractar la distorsió radial. Vegem pas a pas com modela els diferents paràmetres. La majoria de les passes que es veuran són comuns tant en Faugeras, com en altres mètodes de calibració, com Hall, Tsai, etc.

2.7.1 Orientació i posició de la càmera

La transformació de les coordenades món al sistema de coordenades de la càmera es realitza mitjançant un vector translació i una matriu de rotació, com es pot veure en la següent equació 2.3.

$$\begin{pmatrix} {}^{C}x_w \\ {}^{C}y_w \\ {}^{C}z_w \end{pmatrix} = {}^{C}R_W \begin{pmatrix} {}^{W}x_w \\ {}^{W}y_w \\ {}^{W}z_w \end{pmatrix} + {}^{C}T_w \tag{2.3}$$

Llavors, donat un punt ${}^{W}X_w$ amb coordenades $\left({}^{W}x_w, {}^{W}y_w, {}^{W}z_w\right)$ expressat en el sistema de coordenades del món, i aplicant l'equació 2.3 obtenim el punt ${}^{C}X_w$ expressat en el sistema de coordenades de la càmera. El paràmetre ${}^{C}R_W$ expressa la orientació del sistema de coordenades del món W respecte del de la càmera C, i el paràmetre ${}^{C}T_w$ expressa la posició de l'origen del sistema de coordenades món respecte del de la càmera.

2.7.2 Projecció perspectiva

Seguint el model de la càmera com a forat d'agulla o "pin-hole", el pla imatge està a una distància f del centre òptic O_C, i és paral·lel al pla definit per els eixos de coordenades x_C i y_C. Llavors, donat un punt real de l'objecte ${}^{C}X_w$, expressat en el sistema de coordenades de la càmera, si es projecta a través del punt focal O_C, el raig òptic intercepta en el pla imatge 2D en un punt que anomenarem ${}^{C}X_u$, és a dir, un punt de la imatge expressat en el sistema de coordenades de la càmera. Aquesta relació la podem veure a la següent equació:

$$\begin{gathered} {}^{C}x_u = f\frac{{}^{C}x_w}{{}^{C}z_w} \qquad {}^{C}y_u = f\frac{{}^{C}y_w}{{}^{C}z_w} \\ \text{Essent f la distància del pla imatge de } O_C \end{gathered} \tag{2.4}$$

2.7.3 Distorsió de la lent

Per modelar la distorsió de la lent, transformem el punt sense distorsió ${}^{C}P_u$ (el subíndex ve de l'anglès *undistorted*) en el punt distorsionat, anomenat ${}^{C}P_d$ (de l'anglès *distorted*), seguint l'equació següent:

$${}^{C}x_u = {}^{C}x_d + \delta_x \qquad {}^{C}y_u = {}^{C}y_d + \delta_y \tag{2.5}$$

El model original proposat per Faugeras no modela la distorsió de la lent, per tant, ${}^{C}X_u$ i ${}^{C}X_d$ són el mateix punt, i conseqüentment δ_x i δ_y són zero. En canvi, com hem comentat anteriorment, podem millorar el model amb la distorsió de la lent, tenint en compte només la distorsió radial. Llavors, a partir del treball de [Slama 1980], ho representaríem amb la següent equació:

$$\delta_x = \delta_{xr} \qquad \delta_y = \delta_{yr} \tag{2.6}$$

Aquest tipus de distorsió es produeix bàsicament per imperfeccions en la curvatura de les lents. Es presenta sobretot en els contorns de la imatge, en els quals objectes que en la realitat presenten línies totalment rectes, en la imatge presenten una certa cobertura. La figura 2.5 representa la distorsió radial.

El desplaçament donat per la distorsió radial *dr* pot ser modelat amb les següents equacions:

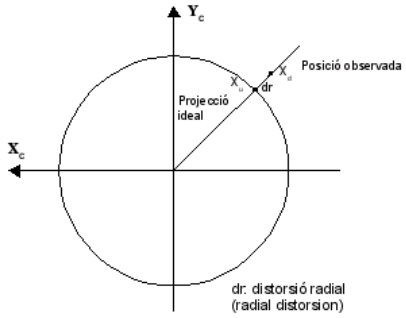


Figura 2.5: Representació de la distorsió radial.

$$\delta_{xr} = -k_1 {}^C x_d \left({}^C x_d^2 + {}^C y_d^2 \right) \qquad \delta_{yr} = -k_1 {}^C y_d \left({}^C x_d^2 + {}^C y_d^2 \right) \tag{2.7}$$

Com podem veure, l'únic paràmetre que considerem és k_1, que és el primer terme de la sèrie que descriu la distorsió radial. Està comprovat que el primer terme de les sèries és suficient per modelar la distorsió radial en la majoria de les aplicacions.

2.7.4 Expressar els punts del pla imatge en píxels

El pas final és expressar el punt ${}^C X_d$ en el pla imatge utilitzant el sistema de coordenades de les imatges de l'ordinador, és a dir, els píxels, amb el que ens referirem com {I}. A diferència dels passos anterior, aquest pas és pot afrontar de diferents maneres, depenent del model de calibració que estiguem utilitzant. Per tant, els següents passos són aplicables només per el model de Faugeras. Les equacions que porten a terme la transformació són les següents:

$$ {}^I x_d = -k_u {}^C x_d + u_0 \qquad {}^I y_d = -k_v {}^C y_d + v_0 \tag{2.8}$$

Els paràmetres k_u i k_v són els que transformen les mesures mètriques expressades en el sistema de coordenades de la càmera en píxels del sistema de coordenades imatge. Els paràmetres u_0 i v_0 són els components que defineixen la projecció del punt focal en el pla imatge en píxels i s'utilitzen per determinar la translació entre els dos sistemes de coordenades.

2.7.5 El mètode de Faugeras amb distorsió radial

Quan es necessita certa precisió, el mètode de Faugeras, que és lineal, no serveix ja que no modela la distorsió radial provocada per lents òptiques de la càmera. No obstant, és possible modificar el mètode de Faugeras original incloent la distorsió radial, vista en la secció 2.7.3. Llavors, el mètode de Faugeras amb distorsió radial esdevé no lineal i es resol mitjançant un algorisme iteratiu (recordem que és un algorisme que parteix d'una solució candidata inicial i que la va refinant fins que l'error de la solució està sota un llindar).

Combinant les anteriors equacions (2.3), (2.4), (2.5), (2.6), (2.7), s'obtenen les equacions (2.9).

$$
\begin{aligned}
{}^{C}x_d + {}^{C}x_d k_1 r^2 &= f\frac{r_{11}{}^{W}x_w + r_{12}{}^{W}y_w + r_{13}{}^{W}z_w + t_x}{r_{31}{}^{W}x_w + r_{32}{}^{W}y_w + r_{33}{}^{W}z_w + t_z} \\
{}^{C}y_d + {}^{C}y_d k_1 r^2 &= f\frac{r_{21}{}^{W}x_w + r_{22}{}^{W}y_w + r_{23}{}^{W}z_w + t_y}{r_{31}{}^{W}x_w + r_{32}{}^{W}y_w + r_{33}{}^{W}z_w + t_z} \\
r &= \sqrt{{}^{C}x_d^2 + {}^{C}y_d^2}
\end{aligned}
\tag{2.9}
$$

A més a més, cal combinar també les equacions (2.8) per transformar de les coordenades mètriques a píxels. Llavors, obtenim l'equació (2.10), que defineix un vector d'incògnites les quals han de ser computades per un mètode iteratiu, com per exemple el Newton-Raphson, Levenberg-Marquardt, etc.

$$
f = (\alpha, \beta, \gamma, t_x, t_y, t_z, k_u, k_v, u_0, v_0, k_1)^T \tag{2.10}
$$

Per exemple, el mètode general de Newton-Raphson minimitza la següent equació,

$$
G(f_k) \approx G(f_{k-1}) + J\Delta f_k \tag{2.11}
$$

en la qual f és el vector d'incògnites, $G(f)$ és la minimització de la funció, $G(f_k)$ és una valor proper a la solució, i J representa la matriu jacobiana de la funció $G(f)$. Per a trobar una solució de Δf_k, és necessari igualar $G(f_k)$ a zero.

$$
G(f_k) = 0 \tag{2.12}
$$

Cal dir que un dels problemes de convergència dels algorismes iteratius és la solució candidata inicial. No obstant, aquesta pot ésser obtinguda calibrant el mètode lineal de Faugeras obviant la distorsió de la lent i assumint que $k_1 = 0$. A més a més, la diferència entre el valor inicial i els paràmetres estimats esdevindrà l'error de la funció. Per cada iteració és necessari calcular el valor de Δf_k per obtenir el nou valor de f.

$$
J\Delta f_k = -G(f_{k-1}) \tag{2.13}
$$

Llavors, combinant les equacions (2.9) i (2.8), s'obtenen les funcions $U(f)$ i $V(f)$.

$$
\begin{aligned}
U(f) &= f\frac{r_{11}{}^{W}x_w + r_{12}{}^{W}y_w + r_{13}{}^{W}z_w + t_x}{r_{31}{}^{W}x_w + r_{32}{}^{W}y_w + r_{33}{}^{W}z_w + t_z} - \frac{\left({}^{I}x_d - u_0\right)}{-k_u} \\
&-k_1\left(\left(\frac{\left({}^{I}x_d - u_0\right)}{-k_u}\right)^2 + \left(\frac{\left({}^{I}y_d - v_0\right)}{-k_v}\right)^2\right)\frac{\left({}^{I}x_d - u_0\right)}{-k_u} \\
V(f) &= f\frac{r_{21}{}^{W}x_w + r_{22}{}^{W}y_w + r_{23}{}^{W}z_w + t_y}{r_{31}{}^{W}x_w + r_{32}{}^{W}y_w + r_{33}{}^{W}z_w + t_z} - \frac{\left({}^{I}y_d - v_0\right)}{-k_v} \\
&-k_1\left(\left(\frac{\left({}^{I}x_d - u_0\right)}{-k_u}\right)^2 + \left(\frac{\left({}^{I}y_d - v_0\right)}{-k_v}\right)^2\right)\frac{\left({}^{I}y_d - v_0\right)}{-k_v}
\end{aligned}
\tag{2.14}
$$

Llavors, per solucionar el sistema, cal aplicar les anteriors equacions als n punts de calibració. No obstant, a l'hora de calcular el valor Δf_k, com es mostra en l'equació (2.13), és necessari obtenir la funció $G(f)$ i la seva matriu derivativa parcial J, com es mostra en les següents equacions,

$$G(f_{k-1}) = \begin{pmatrix} U_1(f_{k-1}) \\ V_1(f_{k-1}) \\ \vdots \\ V_n(f_{k-1}) \end{pmatrix} \tag{2.15}$$

$$J = \begin{pmatrix} \frac{\partial U_1(f_{k-1})}{\partial \alpha} & \frac{\partial U_1(f_{k-1})}{\partial \beta} & \dots & \frac{\partial U_1(f_{k-1})}{\partial k_1} \\ \frac{\partial V_1(f_{k-1})}{\partial \alpha} & \frac{\partial V_1(f_{k-1})}{\partial \beta} & \dots & \frac{\partial V_1(f_{k-1})}{\partial k_1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial V_n(f_{k-1})}{\partial \alpha} & \frac{\partial V_n(f_{k-1})}{\partial \beta} & \dots & \frac{\partial V_n(f_{k-1})}{\partial k_1} \end{pmatrix} \tag{2.16}$$

Finalment, els paràmetres del model s'obtenen aplicant les pseudo-inverses de les equacions (2.17) en cada iteració. A major nombre d'iteracions, s'obtenen valors més precisos i propers a la convergència.

$$\begin{gathered} \triangle f_k = -(J^T J)^{-1} J^T G(f_{k-1}) \\ f_k = f_{k-1} + \triangle f_k \end{gathered} \tag{2.17}$$

2.7.6 Matriu de projecció de Faugeras

Un cop hem obtingut els paràmetres de la càmera, podem generar la matriu de projecció. Aquesta matriu és la que ens permet passar dels píxels als punts en mètric de l'escena, i és l'aplicació directa dels paràmetres trobats en la calibració de la càmera. Aquesta matriu de projecció varia depenent del mètode de calibració utilitzat, per tant mostrarem la que s'aplica al Faugeras amb distorsió radial.

Primerament, posem els paràmetres intrínsecs en forma matricial com es pot veure en l'equació 2.18.

$$\begin{gathered} P1 = \begin{pmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \\ \text{On } \alpha_u = -fk_u; \alpha_v = -fk_v \end{gathered} \tag{2.18}$$

Segonament, farem el mateix amb els paràmetres extrínsecs (matriu de rotació i vector de translació) formant l'equació 2.19.

$$P2 = \begin{pmatrix} R & T \\ 0_{1\times 3} & 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.19}$$

Finalment, combinant les dues equacions trobem la matriu de projecció P, que relaciona els punts del món real, amb la seva projecció.

$$P = \begin{pmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.20}$$

Finalment, la matriu de projecció podem relacionar els punts reals amb les seves projeccions en el pla imatge i la seva aplicació es pot observar en l'equació 2.21.

$$\begin{pmatrix} {}^{I}x_u \\ {}^{I}y_u \\ s \end{pmatrix} = \begin{pmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} {}^{W}x_w \\ {}^{W}y_w \\ {}^{W}z_w \\ 1 \end{pmatrix} \tag{2.21}$$

Capítol 3

Detecció de cantonades amb el filtre de Harris

3.1 Introducció

Per estudiar les imatges hem d'extreure característiques que identifiquin el que volem estudiar o analitzar. Altrament, per representar el món real en models en tres dimensions utilitzant les bases de la visió estèreo, hem de partir de la informació de les imatges. Aquestes característiques han de ser discretes i no pas continues. Per exemple, la textura dels objectes representats no serviria. Per el nostre sistema estèreo, una característica interessant són les cantonades, ja que defineixen molt bé la forma dels objectes presents en la imatge. Existeixen diferents detectors de cantonades, els més coneguts són el detector de Harris i el detector de Kanade-Lucas-Tomasi (KLT). Tots dos donen uns resultats semblants, així que es pot utilitzar qualsevol dels dos indistintivament. Optarem per el detector de Harris per què és el més utilitzat en l'àmbit europeu.

3.2 Descripció d'una cantonada

Primerament, explicarem la diferència entre cantonada (en anglès *corner*) i vora (en anglès *edge*). Les cantonades són característiques locals de la imatge composades per localitzacions on la variació de l'intensitat del color del píxel és alta tant en la direcció de X, com de Y. Les derivades parcials de f_x i de f_y són altes.

En canvi, les vores són localitzacions on la variació de l'intensitat només és alta en una certa direcció, mentre que la variació en la direcció ortogonal és baixa. Per exemple, en una vora orientada al llarg de l'eix Y, f_x és alta, mentre que f_y és baixa. En la figura 3.1 hi ha un exemple d'una cantonada i una vora.

Per tant, una cantonada és el punt on intersecccionen dues vores que segueixin direccions diferents. Encara que la manca de connectivitat entre les cantonades (són solsament punts) pugui resultar en una limitació a l'hora d'obtenir descripcions riques del món en tres dimensions, com ara superfícies, per a objectes que no siguin llisos resulta molt apropiat per obtenir la seva estructura.

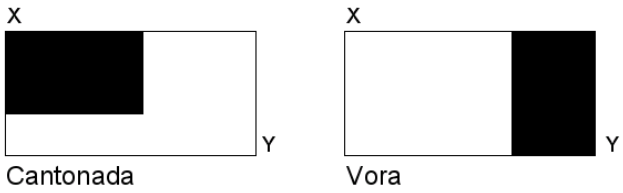


Figura 3.1: Exemple d'una cantonada i una vora

3.3 El detector de cantonades

Per a jutjar la qualitat d'un detector de cantonades normalment ens fixem en l'habilitat de detectar la mateixa cantonada en múltiples imatges. Aquestes imatges seran similars però no idèntiques, és a dir, presentaran diferents condicions d'il·luminació, translació, rotació i altres possibles transformacions. Naturalment, per el nostre sistema estèreo això és molt important, ja que el detector de cantonades ens ha de detectar les mateixes cantonades en imatges que presenten una translació i una possible petita rotació.

Una tècnica senzilla per detectar cantonades en imatges seria utilitzar la correlació amb el color dels píxels, però aquesta resulta ser molt ineficient i amb resultats no gaire òptims. L'alternativa és utilitzar tècniques com les implementades en el detector de cantonades de Harris [Harris 1988], el qual és una millora del mètode introduït inicialment per Moravec [Moravec 1977]. Aquest algorisme és considera obsolet, ja que no és invariant a la rotació de la imatge i és sensible al soroll. No obstant, és computacionalment molt eficient respecte, la qual cosa era molt important en el moment en que es va presentar.

Per tant, estudiarem el detector de cantonades de Harris, que millora les deficiències de variabilitat en rotació i sensibilitat al soroll que presenta el de Moravec.

3.3.1 El detector de cantonades de Harris

El detector de cantonades de Harris és un filtre que rep com entrada una imatge i extreu les cantonades d'aquesta. Nosaltres ens centrarem en les imatges amb nivells de grisos, tot obviant les imatges de color. Assumirem que tenim una imatge I de dues dimensions, i als píxels d'aquesta ens referirem com $I(x, y)$.

$\nabla I(x, y)$ és el pendent local de la imatge i $\nabla I(x, y) \cdot n$ és el grandient al llarg de la direcció n. Per a un corner, rotem n al llarg de tots els possibles valors (agafar totes les direccions), i hauriem de trobar dues direccions en les qual el pendent és màxim (màxim local).

Definim doncs:

$$\begin{aligned} C_n &= \frac{|\nabla I(x,y) \cdot n|}{\|n\|} \\ {C_n}^2 &= \frac{n^T \nabla I \nabla I^T n}{n^T n} \end{aligned} \tag{3.1}$$

on,

$$\nabla I \nabla I^T = Q = \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) \\ \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \left(\frac{\partial I}{\partial x}\right)^2 \end{bmatrix} \tag{3.2}$$

Com en qualsevol detector de cantonades, cal que eliminarem abans el soroll de les imatges. Un mètode apropiat és utilitzar una convolució amb kernel de Gauss. Aquest operador està reflectit en la matriu A següent (utilitzant la notació de les derivades parcials):

$$A = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix} \tag{3.3}$$

La matriu A de l'equació 3.3 la podem substituir en 3.1, tot obtenint

$$C_n^2 = \frac{n^T A n}{n^T n} \tag{3.4}$$

L'anterior equació és un coeficient de Rayleigh, la qual cosa significa que el podem establir entre dos límits

$$\lambda_1 \leq \frac{n^T A n}{n^T n} \leq \lambda_2$$

on λ_1 i λ_2 són els valors més petit i més gran dels valors propis de la matriu A respectivament. Per tant, si n varia entre tots els possibles valors (per a cobrir totes les direccions), C_n^2 estarà retringida entre aquests dos límits. Si ens basem en els valors propis, llavors podem observar que es produeixen els següents casos:

1. Si $\lambda_1 \approx 0$ i $\lambda_2 \approx 0$, llavors no hi ha cap característica d'interès en el píxel (x, y).
2. $\lambda_1 \approx 0$ i λ_2 té valors positius elevats, llavors s'ha trobat una vora.
3. λ_1 i λ_2 tenint valors alts positiu i són diferents, llavors s'ha trobat una cantonada.

En la pràctica, quan s'implementa l'algorisme normalment es cerca fent servir un llindarα en la funció M_c que pot ésser canviat com a paràmetre d'entrada. Aquest llindar determina la mesura en que s'han d'evitar les vores.

$$\begin{aligned} M_c &= \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2 \\ M_c &= \det(A) - \alpha \text{trace}^2(A) \end{aligned} \tag{3.5}$$

Com podem veure, l'algorisme no cal que calculi la descomposició en valors propis de la matriu A, sinó que és suficient amb avaluar el determinant i fer una traça de A per trobar les cantonades. El valor de α s'ha de determinar empíricament, encara que un valor entre el rang de $0.04 - 0.15$ sol donar resultats prou bons.

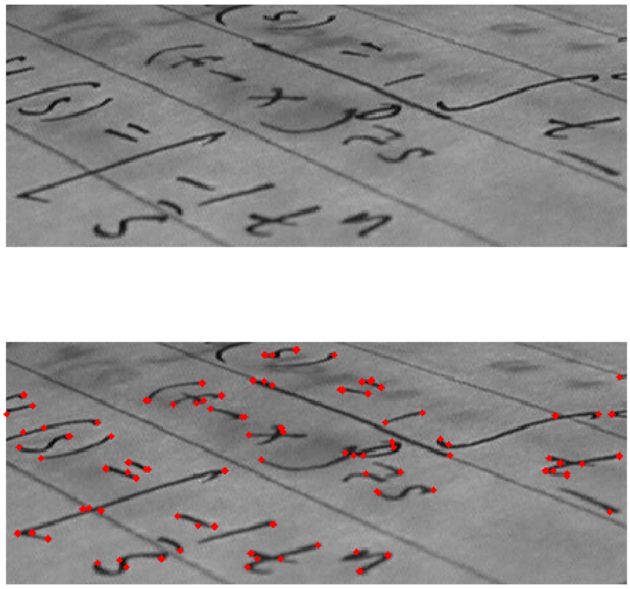


Figura 3.2: Exemple de corners trobats per l'algorisme de Harris

Capítol 4

Geometria epipolar

4.1 Introducció

La visió estèreo és una forma d'obtenir informació tridimensional a través d'un sistema de visió artificial. La visió estèreo o estereoscòpica es fonamenta en com funciona la vista biològicament: la profunditat de qualsevol punt de l'espai es calcula en base a la diferència de posició de la seva projecció en un parell d'imatges.

En 1971, Julesz [Julesz 1971] va dur a terme un experiment on demostrava que el sistema de visió humà no necessita del reconeixement de característiques o de la interpretació a alt nivell de les imatges per a percebre formes tridimensionals. Experiments posterior s'ha vist que, per efectuar la visió estèreo, les persones també fem ús de característiques i processos d'alt nivell (informació de textures, grups perceptuals, reconeixement d'objectes etc.), tot i que no és imprescindible.

Els sistemes estèreo artificials pretenen emular el funcionament del nostre sistema de visió. La metodologia general per abordar el problema passa per un primer pas on s'ha d'establir la correspondència dels punts individuals en el nostre parell d'imatges. Seguidament, utilitzant la diferència de posició d'aquests, calcular la profunditat de cadascun dels punts. Això comporta, a priori, dos problemes fonamentals:

- La selecció dels punts en la primera imatge i la cerca de la seva corresponent correspondència en la segona imatge. Aquest problema és conegut com *correspondència* i és crític en la visió estèreo.
- L'obtenció de la profunditat, un cop tenim el conjunt de parelles de punts o correspondències entre les dues imatges. El procediment sota condicions ideals resulta bastant senzill, però a la pràctica es veu complicat. És imprescindible realitzar una precisa calibració de les càmeres utilitzades.

4.2 Geometria epipolar

La geometria epipolar és la geometria projectiva intrínseca entre dues vistes o imatges de la mateixa escena. És independent de l'estructura de l'escena, i només

depèn dels paràmetres intrínsecs de la càmera (que s'extreuen de la calibració, veure capítol Calibració de càmeres) i la seva posició relativa.

La matriu fonamental F és una matriu de 3x3 i de rang 2 encapsula la geometria intrínseca. Si tenim que un punt X a l'espai es projecta com x en la imatge de la primera càmera, i com x' en la imatge de la segona, llavors els punts imatge satisfan la relació ${x'}^T F x = 0$. La matriu fonamental és independent de l'estructura de l'escena.

La geometria epipolar es basa en la geometria de la intersecció dels plans imatge amb el raigs dels plans tenint la *línia de fons* com a eix (la línia de fons és la línia que uneix els centres òptics de cada càmera). Com es pot veure a la figura 4.1 els punts x i x' són coplanars, i a partir d'ara ens referirem a aquest pla com π.

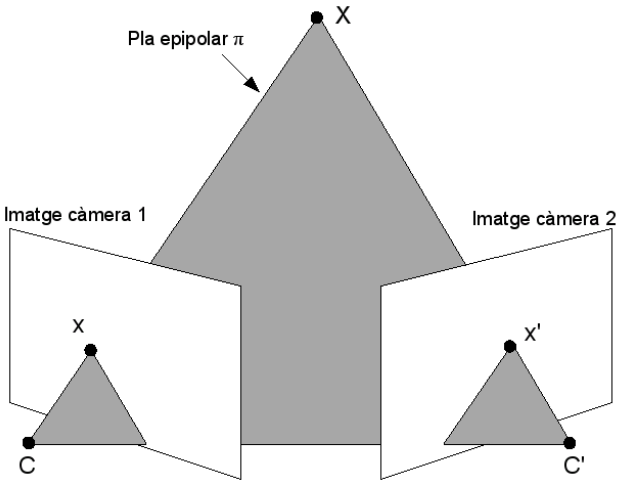


Figura 4.1: Geometria epipolar per correspondència de punts.

Com es pot observar, els rajos que projecten els punts x i x' interseccionen en X, i a més aquests rajos són coplanars, pertanyent al pla π.

Suposem ara que només coneixem el punt x i que a partir d'aquest volem trobar la seva correspondència x'. El pla π és determinat per la línia de fons i el raig definit per x. Per tant, sabem que el raig corresponent per el punt x', que per ara és desconegut, pertany al pla π, i que per tant el punt x' es troba en la línia l' que conforma la intersecció de π amb el segon pla imatge. Aquesta línia l' és la projecció en el segon pla imatge del raig del punt x. Això comporta que l'algorisme estèreo de cerca de correspondències no ha de tractar amb tota la imatge, sinó que es pot restringir a la línia l'. La qual cosa ens beneficia reduint el temps de cerca i reduint la llista de candidats a correspondència, la qual cosa també ens redueix la probabilitat d'error (d'agafar un punt correspondència equivocat o un outlier).

Veiem a continuació la terminologia que utilitzarem (es pot observar en la figura 4.2):

- L'epipol: és el punt on intersecciona la línia que uneix els centres òptics de les càmeres (línia de fons) amb el pla imatge.

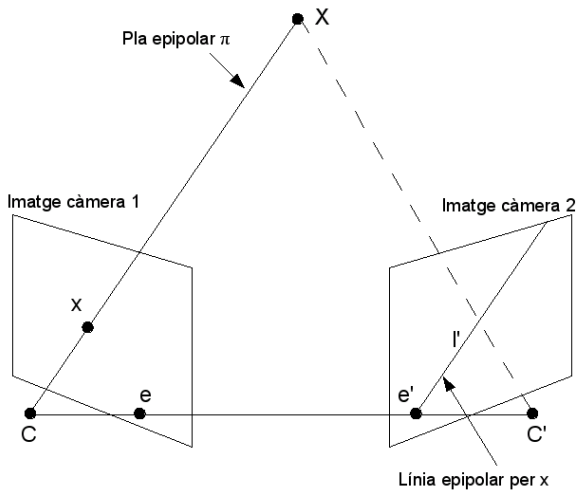


Figura 4.2: Conceptes bàsics de geometria epipolar.

- Un pla epipolar és un pla que conté la línia de fons.
- Una línia epipolar és la línia que es forma a partir de la intersecció d'un pla epipolar amb un pla imatge. Totes les línies epipolars s'uneixen en l'epipol. Un pla epipolar intersecciona amb els dos plans imatge formant les línies epipolars, i defineix la correspondència entre les línies.

4.3 La matriu fonamental

La matriu fonamental és la representació algebraica de la geometria epipolar. Com hem comentat anteriorment, és una matriu de 3 x 3 de rang 2. I a més, per tots els parells de punts x i x' que conformen una correspondència, es compleix la relació ${x'}^T F x = 0$.

Donat un parell d'imatges, cada punt x de la primera imatge té una correspondència amb una línia epipolar l' en la segona imatge. I cada punt x' de la segona imatge que tingui correspondència amb el punt x ha de pertànyer a la línia l'. La línia epipolar és la projecció en la segona imatge del raig de llum que passa per el punt x fins el punt focal C de la primera càmera. Per tant, podem definir la relació:

$$x \mapsto l' \tag{4.1}$$

De la qual, a cada punt x de la primera imatge li correspon una línia epipolar en la segona imatge. Aquesta relació és una correlació, que mapeja de punts a línies de forma projectiva, i queda representada per la matriu fonamental F.

4.3.1 Explicació geomètrica

Veiem com es deriva la matriu fonamental geomètricament. El mapeig d'un punt en la primera imatge a una línia epipolar en la segona imatge es pot descomposar en dues passes.

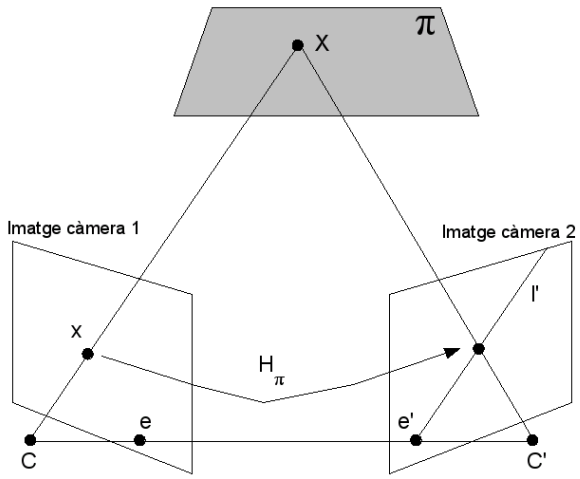


Figura 4.3: Un punt x en una imatge es transfereix via el pla π cap al punt x' en la segona imatge.

Primerament, el punt x és mapeja a un punt qualsevol x' que pertanyi a la recta l' en la segona imatge. Aquest punt x' és un candidat a ser la correspondència de x. En el segon pas, la línia epipolar l' s'obté unint el punt x' amb l'epipol e'. Veiem amb més atenció aquests passos.

Per el primer pas, transferir un punt via un pla, considerarem un pla π en l'espai que no travessa cap dels dos punts focals de les càmeres (veure la figura 4.3). El raig que passa per el punt focal de la primera càmera i el punt x, intersecciona el pla π en el punt X. Aquest punt X és llavors projectat a al segona imatge, tot formant el punt x'. Aquest procediment és la transferència del punt x via el pla π. Com que X està situat en el raig corresponent a x, el punt x' ha de pertànyer en la línia epipolar l', que correspon a la projecció en la segona imatge d'aquest raig. Els punts x i x' són tots dos projeccions del punt X de l'espai. El conjunt de punts x_i de la primera imatge i les seves correspondències ${x'}_i$ són projectivaments equivalents, ja que cadascun d'ells són projetivament equivalents al conjunt de punts X_i del pla. Per tant, hi ha una homografia H_π que mapeja cada punt x_i en ${x'}_i$.

Passem a veure el segon pas: construir la línia epipolar. Donat el punt x', la línia epipolar l' que uneix x' amb l'epipol e' pot ésser escrita com (la notació $[e']_x$ està definida en (B.4 - pàg. 82)):

$$l' = e' \times x' = [e']_x x' \tag{4.2}$$

Com que x' es pot escriure com $x' = H_\pi x$, llavors obtenim:

$$l' = [e']_x H_\pi x = Fx \tag{4.3}$$

A partir de la qual, definirem la matriu fonamental com:

$$F = [e']_x H_\pi \tag{4.4}$$

On H_π és el mapeig de transferència d'una imatge a l'altre passant per el pla π, com ja hem vist anteriorment. A més, ja que $[e']_x$ té rang 2 i H_π té rang 3, llavors podem deduir que la matriu fonamental F és una matriu de rang 2.

Geomètricament, F representa un mapeig d'un pla projectiu de dos dimensions de la primera imatge a un conjunt de línies epipolars a partir de l'epipol e'. Per tant, representa un mapeig des de dues dimensions cap a una dimensió en l'espai projectiu, i per això ha de tenir rang 2.

Cal notar que el pla π que s'ha utilitzat realment no és requerit per F. Aquest pla només s'ha utilitzat per definir el mapeig dels punts d'una imatge a l'altre. Més endavant, en el càlcul de F es veurà que no cal aquest pla, així com en l'aplicació d'aquesta sobre els punts d'una imatge.

4.3.2 Propietats de la matriu fonamental

La propietat més important és que partint de dues càmeres amb punts focals no coincidents, la matriu fonamental F és la única matriu homogènia de 3 x 3 i de rang 2 que satisfà:

$$x'^T F x = 0 \tag{4.5}$$

Per totes les correspondències $x \leftrightarrow x'$.
A continuació, hi ha altres propietats que compleix la matriu fonamental:

- Transposada: Si tenim la matriu fonamental F d'un parell de càmeres (P,P'), llavors F^T és la matriu fonamental del parell de càmeres en ordre oposat (P',P).

- Línies epipolars: Per qualsevol punt x en la primera imatge, la corresponent línia epipolar és $l' = Fx$. De forma anàloga tenim que $l = F^T x'$ representa la línia epipolar corresponent al punt x' de la segona imatge.

- L'epipol: per qualsevol punt x, que sigui diferent de l'epipol e, la línia epipolar $l' = Fx$ conté l'epipol e'. Per tant, e' satisfà $e'^T(Fx) = (e'^T F)x = 0$ per tota x.

- La matriu fonamental té set graus de llibertat: una matriu homogènia de 3 x 3 té vuit graus de llibertat (hi ha 9 elements, però l'últim és el factor d'escala, que no és significatiu). No obstant, la matriu fonamental satisfà la restricció $det(F) = 0$, la qual li treu un grau de llibertat.

- La matriu fonamental és una correlació, un mapeig projectiu des d'un punt a una línia. El punt x de la primera imatge defineix en la segona una línia $l' = Fx$, la qual és la línia epipolar de x. Si l i l' són línies epipolars llavors qualsevol punt x que pertanyi a l serà mapejat sobre la mateixa l'. Això significa que no hi ha mapeig invers, i que per tant F no té un rang complet, és a dir, que no és de rang 3 (recordem que és de rang 2). Per aquesta raó, F no és pròpiament una correlació, ja que li falta la propietat de que sigui invertible.

4.3.3 Càlcul de la matriu fonamental

La matriu fonamental és definida per la següent equació:

$$\mathrm{x}'^T F \mathrm{x} = 0 \tag{4.6}$$

Que s'aplica per qualsevol correspondència $\mathrm{x} \leftrightarrow \mathrm{x}'$ en el nostre parell d'imatges. Donat un conjunt suficient de correspondències $\mathrm{x}_i \leftrightarrow \mathrm{x}'_i$ (com a mínim 7), l'equació 4.6 pot ésser usada per calcular la matriu desconeguda F. Si definim $\mathrm{x} = (x, y, 1)^T$ i $\mathrm{x}' = (x', y', 1)^T$, cada correspondència determina una equació lineal amb les incògnites de F. Els coeficients d'aquesta equació poden ésser escrits de forma senzilla en termes de les conegudes coordenades de x i x′. Seguint amb l'exemple anteriorment, i partint de l'equació 4.6, obtindríem la següent equació:

$$\begin{pmatrix} x' & y' & 1 \end{pmatrix} \begin{pmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.7}$$

Que operant:

$$\begin{pmatrix} x'f_{11} + y'f_{21} + f_{31} & x'f_{12} + y'f_{22} + f_{32} & x'f_{13} + y'f_{23} + f_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Ens resulta:

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0 \tag{4.8}$$

Anomenarem com f el vector de 9 coeficients els quals són els valors de F ordenats per ascendentment per files. Llavors, l'equació 4.8 pot ésser expressada per el producte:

$$(x'x, x'y, x', y'x, y'y, y', x, y, 1)\, f = 0 \tag{4.9}$$

Llavors, per a un conjunt de n correspondències, obtenim un conjunt d'equacions lineals de la mateixa forma que l'anterior trobada:

$$Af = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} f = 0 \tag{4.10}$$

El conjunt d'equacions 4.10 és homogeni. Per a que una solució existeixi, la matriu A ha detenir com a mínim rang 8, i si el rang és exactament 8, llavors la solució és única, i pot ésser trobada per mètodes lineals.

No obstant, és probable que les dades no siguin del tot exactes, degut a soroll en les coordenades dels punts. Això comportarà que el rang sigui superior a 8, per a ser més exactes 9, ja que la matriu A té 9 columnes. En aquest cas, es pot trobar la solució utilitzant mínims quadrats. Un algorisme idoni és el SVD (Singular Value Descomposition o Descomposició en Valors Singulars).

Nosaltres hem utilitzat el SVD per trobar la matriu fonamental, ja que s'aplica bé a tots els casos. Com es pot observar, per calcular la matriu fonamental ens cal un conjunts de correspondències conegudes. Per formar aquest conjunt pot ésser ideal utilitzar el patró de calibració, ja que són coneguts i podem realitzar l'associació de correspondències de forma senzilla.

4.4 Cercar la correspondència

Fins ara, hem vist què és la matriu fonamental, com a aplicació de la geometria epipolar, i com és calcula. Ara analitzarem com s'aplica per a trobar les correspondències.

Partirem de dos conjunts de punts, x per els punts de la imatge de la primera càmera i x' per els de la segona; i de la matriu fonamental ja calculada. El principal problema és establir les correspondències, és a dir, la relació $\mathrm{x} \leftrightarrow \mathrm{x}'$. Utilitzarem els punts x de la primera imatge, com a punts coneguts, i reduirem el problema a trobar la parella en el conjunt x'.

Com hem vist, el primer pas és multiplicar la matriu fonamental per el punt, per trobar la línia epipolar en la segona imatge: $l' = Fx$. Però com passem d'aquesta línia, a punt concret x'? La idea és explorar els píxels d'aquesta línia per trobar el x' que més semblança tingui amb el nostre. per exemple, comparant el color dels píxels. No obstant, degut al soroll en les coordenades dels punts, pot ser que el punt x' que busquem no estigui ven bé sobre la línia l'. Per tant, la millor solució és obrir una finestra centrada en la línia l' i cercar dintre aquesta finestra punt x'.

Si s'utilitza el valor de color com a mesura de comparació entre les correspondències, pot ésser convenient alguna mena d'ajust si varien les condicions d'il·luminació entre cada imatge. Aquests aspectes varien en l'aplicació de cada cas i s'ha d'estudiar per cada cas de sistema estèreo.

4.5 El filtre de RANSAC

Després de la secció anterior, som capaços d'obtenir un conjunt de correspondències, en les qual l'única font d'error és la mesura de les coordenades dels punts, la qual segueix una distribució Gausiana. Però aquesta premissa no és vàlida en la majoria de situacions, ja que hi ha punts que realment no són correspondència, ja que pertanyen a cantonades diferents de l'objecte. Aquests punts són els que anomenem *outliers*. Aquests outliers poden ocasionar soroll en la homografia que acabem d'estimar. Per això, aplicarem un *estimador robust*, un algorisme que estima una homografia de forma robusta (tolerant als outliers). Un d'aquests estimadors robusts és el RANSAC.

L'algorisme RANSAC (de l'anglès RANdom SAmple Consensus) és comporta molt bé quan hi ha una alta presència de outliers. La idea és simple: es selecciona de forma aleatòria un parell de punts; els quals defineixen una línia. El *suport* d'aquesta línia es mesura a partir del nombre de punts la distància dels quals a la línia es troba sota un llindar. Aquesta selecció aleatòria es repeteix un número de vegades i la línia que obtingui el major suport serà la que s'ajusta de forma més robusta. Els punts que trobem en la distància del llindar són els inliers. Per tant, si un dels dos punts és un outlier, la línia no rebrà gaire suport. A més a més, puntuar una línia tot calculant el seu suport té l'avantatge d'afavorir aquelles que millor s'adapten.

En general, el que volem és adequar un model, en aquest una línia, a un conjunt de dades. I la mostra aleatòria consisteix en un subconjunt de les dades, en aquest cas dos punts, suficient per determinar un model. Si el model és una homografia planar, i les dades són un conjunt de correspondències 2D, llavors el conjunt mínim estarà format per quatre correspondències.

Com es defineix en [Fischler i Bolles 1981], l'algorisme de RANSAC funciona de forma diferent als algorismes de suavitzat convencionals: em comptes d'utilitzar el màxim de dades possibles per a obtenir una solució candidata inicial bona i eliminar després els punts invàlids, RANSAC utilitza un conjunt de dades inicial tant petit com sigui possible i el va ampliant amb dades consistens quan sigui possible. L'algorisme 4.1 conté l'algorisme de RANSAC.

Algorisme 4.1 Algorisme per l'estimador robust RANSAC.

i Seleccionar de forma aleatòria una mostra de punts s de S i instanciar el model en aquest subconjunt.

ii Determinar el conjunt de punts S_i que es troben dintre la distància llindar t del model. El conjunt S_i és el conjunt de consens de la mostra i defineix els inliers de S.

iii Si la mida de S_i (el nombre de inliers) és més gran que el llindar T, tornar a estimar el model utilitzant tots els punts de S_i i acabar.

iv Si la mida de S_i és més petita que T, selecciona un nou conjunt i torna a repetir els passos anteriors.

v Després de N intents, escollirem el conjunt consens més gran S_i, i el model es torna a estimar utilitzant tots els punts del subconjunt S_i.

L'algorisme de RANSAC presenta tres paràmetres: llindar de la distància, número de mostres i com de gran és un conjunt acceptable. Vegem a continuació aquests paràmetres amb més de detall.

4.5.1 Llindar de distància

Volem escollir un llindar de distància t amb la probabilitat α que un punt sigui inlier. Aquest càlcul requereix la distribució de la probabilitat de la distància d'un inlier al model. A la pràctica, la distància s'escull de forma empírica. No obstant, si assumim que l'error messurat és una Gaussiana amb una mitjana de zero i la desviació estàndard σ, llavors podem calcular un valor per a t. En aquest cas, el quadrat de la distància al punt, $d_\perp^2$, és la suma dels quadrats de les variables Gaussianes i segueixen una distribució χ_m^2 amb m graus de llibertat, on m és la codimensió del model. Per a una línia la codimensió és 1, ja que només es mesura la distància perpendicular a la línia. Si el model és un punt, la codimensió tindria valor de 2, i el quadrat de la distància seria la suma dels quadrats de les mesures d'error de x i de y. La probabilitat que el valor d'una variable aleatòria de χ_m^2 sigui més petita que la k^2 és donada per la distribució acumulada de χ_m^2: $F_m(k^2) = \int_0^{k^2} \chi_m^2(\xi)d\xi$. Per la distribució acumulada, trobem la següent equació:

$$\begin{cases} inlier & d_\perp^2 < t^2 \\ outlier & d_\perp^2 \geq t^2 \end{cases} \quad \text{amb } t^2 = F_m^{-1}(\alpha)\sigma^2 \tag{4.11}$$

Normalment α té un valor de 0.95, és a dir, que hi hagi un 95% de probabilitat que el punt sigui un inlier. Això significa que un inlier serà incorrectament

rebutjat només un 5% del temps.

Codimensió (m)	**Model**	t^2
1	Línia, matriu fonamental	$3.84\sigma^2$
2	Homografia, matriu de la càmera	$5.99\sigma^2$
3	tensor trifocal	$7.82\sigma^2$

Taula 4.1: La distància llindar per a una probabilitat $\alpha = 0.95$ que el punt sigui inlier.

4.5.2 Número de mostres

És massa costós computacionalment i innecessari que es provin totes les mostres possibles. Per això s'escull un número de mostres N suficientment elevat per garantir amb una probabilitat p, que almenys hi ha una mostra de vuit punts lliure d'outliers. Normalment p té un valor de 0.99. Si suposem que w és la probabilitat que qualsevol punt escollit sigui un inlier, llavors $\epsilon = 1 - w$ és la probabilitat que sigui un outlier. Per tant, almenys N conjunts (cadascun de s punts) calen ser agafades, on $(1 - w^s)^N = 1 - p$, i aplicant logaritmes ens queda l'equació 4.12.

$$N = log(1 - p)/log(1 - (1 - \epsilon)^S) \tag{4.12}$$

4.5.3 Grandària d'un conjunt de consens acceptable

La manera més simple és acabar quan la mida del conjunt de consens és similar al número d'inliers que es creu que podrien haver-hi en el conjunt de dades, tenint en compte la proporció d'outliers.

4.5.4 Normalització

El resultat d'un algorisme DLT per computar homografies 2D depèn del sistema de coordenades en el qual estan expressats els punts. De fet, el resultat varia si se li aplica transformacions de similaritat en la imatge. Però que és una transformació de similaritat?

Hi ha diversos tipus de transformacions que es poden aplicar a un punt o conjunt de punts, com ara: rotacions, translacions, escalats,... Una transformació de similaritat és composa d'una isometria i d'un escalat isotròpic, la qual cosa significa que s'escala cada coordenada amb el mateix factor. Quan es treballa amb l'espai euclidià aquesta transformació es composa d'una translació i un escalat que es representa amb la següent matriu:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} scos\Theta & -ssin\Theta & t_x \\ ssin\Theta & scos\Theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

La normalització de les dades consisteix en una translació i un escalat de les coordenades del pla imatge. Aquesta normalització s'ha de realitzar abans d'aplicar cal algorisme de DLT i ens proporciona uns resultats més precisos. A més, com que incorpora aquesta normalització fa que l'algorisme DLT sigui invariant

respecte a decisions de l'escala i l'origen del sistema de coordenades. Això passa a causa que el pas de normalització desfà l'efecte del canvi de coordenades, escollint un sistema de coordenades canònic per a mesurar les dades.

Com a primer pas de la normalització, les coordenades de cada imatge són traslladades (per a una diferent translació per a cada imatge) per a portar el centre de masses del conjunt de punts a l'origen. Les coordenades són també escalades, de forma que cada punt x és de la forma $\mathrm{x} = (x, y, w)^T$, en les que cadascuna de les x, y, i w tenen la mateixa magnitud mitjana. En comptes d'escollir diferent factors d'escala per cada direcció del sistema de coordenades, un factor isotròpic d'escala és escollit per tal que les coordenades de x i y d'un punt s'escalin igual. Finalment, hem escollit d'escalar la distància mitjana, de manera que la distància mitjana de cada punt x al origen és $\sqrt{2}$. Això significa que el punt mig és igual a $(1, 1, 1)^T$.

Podem resumir la transformació en aquests passos:

i Els punts són traslladats per tal que el seu centre de masses es situï a l'origen.

ii Els punts són escalats per tal que la distància en mitjana a l'origen sigui $\sqrt{2}$.

iii La transformació es aplicada a cada imatge independentment.

I també podem veure l'algorisme en pseudocodi en 4.2.

Algorisme 4.2 Algorisme DLT amb normalització per homografies 2D.

i **Normalització de x:** Calcular la transformació T de similaritat, que consisteix em una translació i un escalat, que agafa els punts x_i per convertirlos en un nou conjunt de punts $\tilde{\mathrm{x}}_i$, el centre de masses del qual està situat en l'origen $(0, 0)^T$, i la seva distància mitjana a l'origen és $\sqrt{2}$.

ii **Normalització de x′:** Calcular la transformació T' de similaritat per els punts de la segons imatge, transformant els punts de x'_i a $\tilde{\mathrm{x}}'_i$.

iii **DLT**: Aplicaríem com a algorisme DLT, RANSAC, sobre les correspondències $\tilde{\mathrm{x}}_i \leftrightarrow \tilde{\mathrm{x}}'_i$, per a obtenir una homografia $\tilde{H}$ que elimini els outliers.

Per tant, la normalització del conjunt de dades és un pas essencial i mai s'ha de considerar com opcional. A l'hora d'aplicar RANSAC, haurem primer de normalitzar les correspondències candidates.

Un cop aplicat RANSAC, ja tenim les correspondències finals, les quals les farem servir en la triangulació per obtenir les coordenades 3D dels punts, com veurem en el capítol següent.

Capítol 5

Triangulació per a reconstruir el punt en l'espai

5.1 Introducció

En aquest capítol descriurem com calcular la posició d'un punt 3D a l'espai donat les seves projeccions en el pla imatge de dues càmeres, i la matriu de calibració de cada càmera. Assumirem que hi ha errors només en les coordenades mesurades en el pla imatge, però no pas en les matrius projectives P i P' que contenen la calibració de cada càmera.

La idea bàsica és triangular tot projectant els dos rajos que passin per la projecció del punt i el punt focal de cada càmera. En la teoria, els dos rajos haurien de convergir en un punt, que seria el punt X en l'espai. Però degut a diferents errors de mesura, els rajos normalment no interseccionen, sinó que es creuen en l'espai. Veurem com solucionar aquest problema.

5.2 Descripció geomètrica

Suposarem que hem obtingut tant les matrius de calibració de les càmeres, com la matriu fonamental. Per tant, es poden trobar les correspondències x i x$'$. No obstant, no podem evitar cometre errors en la mesura de les coordenades de x i x$'$, la qual cosa farà que els raigs que uneixen cada punt de la correspondència amb el seu respectiu punt focal de la càmera no interseccionin mai en l'espai, com es pot veure en la figura 5.1. Això significa que no podem trobar cap punt X que satisfaci $\mathrm{x} = PX$, $\mathrm{x}' = P'X$; i també que les punts del pla imatge no satisfan la restricció epipolar $\mathrm{x}'^T F\mathrm{x} = 0$. Les dues regles anteriors són equivalents, ja que el parell de punts de la correspondència convergiran en l'espai si i només si els punts satisfan la restricció epipolar. En la figura 5.2 es pot veure com la restricció epipolar no existeix. La línia epipolar $l' = F\mathrm{x}$ és la projecció del raig de x i $l = F^T\mathrm{x}'$ és la projecció de x$'$. Com que els raigs no interseccionen, x$'$ no pertany a l' i x no pertany a l.

Una característica que hauria de tenir el mètode de triangulació a utilitzar és

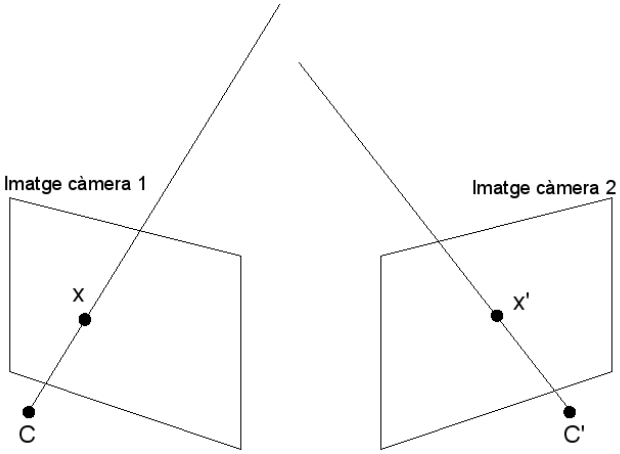


Figura 5.1: Degudes a les imprecisions en x i x′, els rajos no convergeixen.

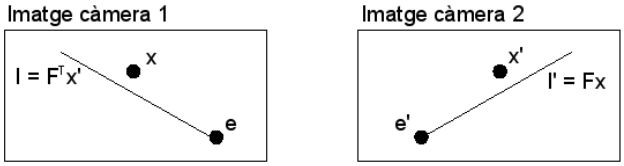


Figura 5.2: Si les rectes no convergeixen, llavors tampoc compleixen la restricció epipolar.

que hauria de ser invariable sota transformacions d'una apropiada classe per a la reconstrucció. És a dir, si les matrius de calibració de les càmeres són conegudes només per a una transformació afí (o projectiva), llavors és desitjable utilitzar un mètode de triangulació afí invariable per a calcular els punts 3D de l'espai. Si anomenem τ al mètode de triangulació utilitzat per a calcular el punt 3D X a partir de la correspondència $\mathrm{x} \leftrightarrow \mathrm{x}'$ i les matrius de les càmeres P i P', podem escriure:

$$X = \tau(\mathrm{x}, \mathrm{x}', P, P') \tag{5.1}$$

La triangulació serà invariable sota una transformació H si compleix:

$$\tau(\mathrm{x}, \mathrm{x}', P, P') = H^{-1}\tau(\mathrm{x}, \mathrm{x}', PH^{-1}, P'H^{-1}) \tag{5.2}$$

Això significa que la triangulació utilitzant les càmeres transformades té com a resultat el punt transformat.

En reconstruccions projectives és inapropiat minimitzar els errors en l'espai projectiu 3D. Per això, el mètode que troba el punt mig de la perpendicular comuna de dos raigs en l'espai no és apropiat per la reconstrucció projectiva, ja que els conceptes de distància i perpendicularitat no són vàlids en la geometria projectiva. De fet, en la reconstrucció projectiva, aquest mètode donarà diferents resultats depenent de quina particular reconstrucció projectiva considerem. Per tant, el mètode no és projectivament invariable.

A continuació veurem mètodes de triangulació lineals simples. Encara que no siguin projectivament invariables, sovint donen resultats acceptables, i per tant s'adequen perfectament a l'abast del projecte.

5.3 Mètodes de triangulació lineals

En aquesta secció explicarem dos mètodes de triangulació singulars simples i discutirem quin és l'apropiat per el nostre projecte, i per tant, el que ha estat implementat. Continuarem amb l'afirmació de que tenim errors de mesura en les coordenades del punts imatge, i per tant, que els raigs descrits per a triangular no convergeixen.

En cada imatge, tenim els punts imatge definits com $x = PX$ i $x' = P'X$, i aquestes equacions és combinen tot formant el sistema d'equacions de la forma $AX = 0$, que és lineal.

Primerament, el primer factor d'escala és eliminat per a un producte vectorial que dóna tres equacions per cada punt imatge, de les quals dues són independents linealment. Per exemple, per el punt de la primera imatge tindríem $x \times (PX) = 0$ i que descomposant obtenim:

$$\begin{aligned} x(p^{3T}X) - (p^{1T}X) &= 0 \\ y(p^{3T}X) - (p^{2T}X) &= 0 \\ x(p^{2T}X) - y(p^{1T}X) &= 0 \end{aligned} \tag{5.3}$$

Cal notar, que en l'equació 5.3, p^{iT} és la fila i de P. Aquestes equacions són lineals en els components de X. L'última equació la podem eliminar, ja que és una composició de les dues primeres. Unint les equacions per als dos punts, obtenim un sistema d'equacions de la forma $AX = 0$, com el següent:

$$A = \begin{bmatrix} xp^{3T} - p^{1T} \\ xp^{3T} - p^{2T} \\ x'p'^{3T} - p'^{1T} \\ y'p'^{3T} - p'^{2T} \end{bmatrix} \tag{5.4}$$

El sistema d'equacions anterior és de quatre equacions amb quatre incògnites homogènies. És un conjunt redundant d'equacions, ja que la solució són tres incògnites més un factor d'escala. Hi ha dues maneres de resoldre el sistema d'equacions que seran discutides a continuació.

5.3.1 Mètode homogeni (DLT)

El mètode anomenat *Direct Linear Transformation* (DLT) o Transformació Lineal Directa es basa en aplicar l'algorisme SVD (Single Value Descomposition) per a trobar la solució de l'equació. Com que tenim quatre equacions, el resultat serà quatre quoficients. Els tres primers, corresponent a les coordenades del punt X, mentre que el quart és un factor d'escala que s'ha d'aplicar a cada component.

5.3.2 Mètode no homogeni

El mètode no homogeni es basa en tractar el sistema com equacions no homogènies. Reescrivim X com $X = (x, y, z, 1)^T$, i el sistema $AX = 0$ queda reduït a un conjunt de quatre equacions no homogènies de tres incògnites. Llavors, es pot aplicar un mètode de resolució per mínims quadrats a aquest conjunt d'equacions no homogènies. No obstant, si la solució real de X té la coordenada z igual o propera a zero, poden ocórrer inestabilitats i donar solucions no vàlides.

5.3.3 Discussió del mètode apropiat

Els dos mètodes anteriors són molt similars, encara que tenen propietats bastant diferents amb la presència de soroll. El mètode no homogeni assumeix que el punt solució X no cau en l'infinit, ja que llavors no es podria assumir que $X = (x, y, z, 1)^T$. Això és un desavantatge per aquest mètode quan cerquem una reconstrucció projectiva en la qual alguns punts podrien caure en un pla a l'infinit.

Cal afegir, que cap d'aquests dos mètodes és projectivament invariable. Per comprovar això, podem suposar que les matrius de les càmeres P i P' són substituïdes per PH^{-1} i per $P'H^{-1}$. Podem veure que en aquest cas, la matriu d'equacions de A queda com AH^{-1}. Un punt X que sigui solució, $AX = \epsilon$ (ϵ és l'error), per les equacions originals, correspon al punt HX que satisfà $(AH^{-1})(HX) = \epsilon$ per el problema transformat. Llavors, hi ha una correspondència ú a ú entre els punts X i els punts HX que donen el mateix error. No obstant, ni la condició $\|X\| = 1$, per el mètode homogeni, ni la condició $X = (x, y, z, 1)^T$ per el mètode no homogeni, són invariants sota l'aplicació de la transformació projectiva H. Per tant, el punt X que és solució del problema original no es correspondrà al punt HX del problema transformat.

D'altra banda, per a les transformacions afins la situació és diferent. De fet, la condició $\|X\| = 1$ no és manté sota una transformació afí, però sí que ho fa $X =$

$(x, y, z, 1)^T$, ja que en una transformació afí es compleix que $H(x, y, z, 1)^T = (x', y', z', 1)^T$. Això significa hi ha una correspondència ú a ú entre un vector $X = (x, y, z, 1)^T$ que se li apliqui $A(x, y, z, 1)^T = \epsilon$ i el vector $HX = (x', y', z', 1)^T$ que se li apliqui $(AH^{-1})(x', y', z', 1)^T = \epsilon$. L'error és el mateix per els punts de la correspondència. Així, els punts que minimitzen l'error $\|\epsilon\|$ es corresponen de forma correcta. Llavors, el mètode no homogeni és afí invariant, mentre que l'homogeni no ho és.

En [Hartley i Zisserman, 2000] es proposa un mètode de triangulació que és projectivament invariant i minimitza l'error geomètric de la imatge. Es desenvolupa un MLE (Maximum Likelihood Estimate) o Estimador de Màxima Semblança i es troba una solució òptima utilitzant un algorisme no iteratiu, sense caldre cap mètode de minimització numèrica lineal.

No obstant el mètode [Hartley i Zisserman, 2000] és el recomanat per les seves propietats i robustesa, el mètode de triangulació lineal homogeni normalment dóna uns resultats acceptables, i es sol utilitzar molt. A més, té la virtut de que és fàcilment generalitzable per triangulacions on participin més de dues vistes. Per tant, utilitzarem el mètode homogeni de triangulació degut a que la relació complexitat/precisió és l'idònia per el present projecte.

5.4 Aproximació de Sampson (correcció geomètrica de primer ordre)

5.4.1 Error de Sampson

L'error geomètric és complex i necessita estimar simultàniament tant la matriu homografia com els punts $\hat{x}_i$, $\hat{x}'_i$. La seva complexitat és oposada a la simplicitat de minimitzar l'error algebraic. Utilitzarem la tècnica introduïda per Sampson [Sampson 1982].

Si X és el punt real i $\hat{X}$ la nostra estimació, $\|X - \hat{X}\|^2$ és l'error que volem minimitzar en la variabilitat ν_H respecte el mesurament de X. ν_H representa la variabilitat d'una transformació H. Aquest punt no pot ésser estimat directament via iteració degut a la naturalesa no lineal de ν_H. La idea de la funció d'error de Sampson és estimar una aproximació de primer ordre al punt $\hat{X}$.

Per a una homografia H, qualsevol punt $X = (x, y, x', y')^T$ que pertanyi a ν_H satisfarà l'equació 5.5, en la que els vectors són homogenis, per tant w_i és la tercera component homogènia i h^j és la fila j de la matriu H.

$$\begin{bmatrix} 0^T & -w'_i \mathrm{x}_i^T & y'_i \mathrm{x}_i^T \\ w'_i \mathrm{x}_i^T & 0^T & -x'_i \mathrm{x}_i^T \end{bmatrix} \begin{pmatrix} h^1 \\ h^2 \\ h^3 \end{pmatrix} = 0 \tag{5.5}$$

L'anterior equació pot ser escrita com

$$A_i h = 0 \tag{5.6}$$

Per recalcar la dependència en X, escriurem l'anterior equació com $C_H(X) = 0$, on $C_H(X)$ és un vector de dos elements (en les equacions 5.5 només dues són linealment independents). Per el primer ordre, aquesta funció de cost pot ésser aproximada utilitzant Taylor donant el resultat de l'equació 5.7

$$C_H(X + \delta_x) = C_H(X) + \frac{\partial C_H}{\partial X}\delta_X \tag{5.7}$$

Si $\delta_X = \hat{X} - X$ i volem que $\hat{X}$ pertanyi a ν_H per que compleixi $C_H(X) = 0$, llavors el resultat és $C_H(X) + (\partial C_H/\partial X)\delta_X = 0$. L'anterior expressió pot ésser escrita com $J\delta_x = -\epsilon$, on J és la matriu parcial derivativa, i ϵ és el cost de $C_H(X)$ associat a X. El problema de minimització esdevé a trobar el valor més petit de δ_X que satisfaci l'equació. Per tant, hem de trobar el vector δ_X que minimitza $\|\delta_X\|$ respecte l'equació $J\delta_X = -\epsilon$.

La tècnica més utilitzada per resoldre aquest tipus de problemes és utilitzar multiplicadors de Lagrange. Introduïm un vector multiplicador de Lagrange λ i el problema passa a minimitzar $\delta_X^T\delta_X - 2\lambda^T(J\delta_X + \epsilon)$, on el factor 2 és introduït per conveniència (no afecta al resultat final). Operant agafant les derivades respecte δ_X i igualant a zero, ens resulta

$$2\delta_X^T - 2\lambda^T J = 0^T$$

A partir de l'equació anterior, simplificant i reordenant obtenim $\delta_X = J^T\lambda$. La derivada respecte λ dóna $J\delta_X + \epsilon = 0$, que és la restricció original. Substituint δ_X, ens dóna la següent equació

$$JJ^T\lambda = -\epsilon$$

que pot ser resolta per λ, resultant $\lambda = -(JJ^T)^{-1}\epsilon$. Finalment, substituint, arribem a l'equació

$$\delta_X = -J^T(JJ^T)^{-1}\epsilon \tag{5.8}$$

Resumint, tenim que $\hat{X} = X + \delta_X$, i el valor de δ_X està expressat en l'equació 5.8. La norma $\|\delta_X\|^2$ és l'error de Sampson:

$$\|\delta_X\|^2 = \delta_X^T\delta_X = \epsilon^T(JJ^T)^{-1}\epsilon \tag{5.9}$$

5.4.2 Correcció geomètrica aplicant Sampson

Quan els errors en la mesura dels punts del pla imatge és petita comparada amb la mesura en sí, es pot aplicar l'aproximació de Sampson per obtenir uns resultats més acurats. El que farem és calcular una correcció als punts mesurats.

La correcció Sampson δ_{x} per el punt mesurat $X = (x, y, x', y')^T$ (cal notar que en aquesta secció X no s'associa a punt homogeni en l'espai 3D) és mostra com hem vist en l'equació 5.8:

$$\delta_{\mathrm{x}} = -J^T(JJ^T)^{-1}\epsilon$$

I el punt corregit és:

$$\hat{X} = X + \delta_{\mathrm{x}} = X - J^T(JJ^T)^{-1}\epsilon$$

Com que volem que es compleixi la restricció epipolar $\mathrm{x}'^T F\mathrm{x} = 0$, podem concloure que l'error a minimitzar és $\epsilon = \mathrm{x}'^T F\mathrm{x}$, i la matriu Jacobiana és

$$J = \partial\epsilon/\partial x = [(F^T\mathrm{x}')_1, (F^T\mathrm{x}')_2, (F\mathrm{x})_1, (F\mathrm{x})_2] \tag{5.10}$$

En l'equació 5.10, tenim que, per exemple, el terme $(F^T \mathrm{x}')_1 = f_{11}x' + f_{21}y' + f_{31}$, i en els altres termes apliquem aquest patró respectivament. Llavors, obtenim que l'aproximació de primer ordre per el punt corregit és simplement

$$\begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{x'} \\ \hat{y'} \end{pmatrix} = \begin{pmatrix} x \\ y \\ x' \\ y' \end{pmatrix} - \frac{\mathrm{x}'^T F \mathrm{x}}{(F\mathrm{x})_1^2 + (F\mathrm{x})_2^2 + (F^T\mathrm{x}')_1^2 + (F^T\mathrm{x}')_2^2} \begin{pmatrix} (F^T\mathrm{x}')_1 \\ (F^T\mathrm{x}')_2 \\ (F\mathrm{x})_1 \\ (F\mathrm{x})_2 \end{pmatrix} \tag{5.11}$$

L'aproximació és precisa si la correcció en cada imatge és petita (menys d'un píxel), i computacionalment senzilla. No obstant, és important senyalar que els punts corregits no satisfan la relació epipolar $\mathrm{x}'^T F \mathrm{x} = 0$ exactament, sinó que sempre tindrem un error ϵ.

Part II

Aplicació pràctica

Capítol 6

Entorn de treball

En aquest capítol descriurem breument quines eines farem servir per a la realització pràctica del projecte.

6.1 Sistema operatiu

El sistema operatiu és Linux, degut a que la majoria de la recerca en visió per computador es fa sobre aquest sistema, i per tant, serà més fàcil integrar els resultats d'aquest projecte en les línies de treball del departament.

D'altra banda, és un sistema operatiu molt eficient a l'hora de gestionar els recursos de la màquina, i presenta moltes facilitats per els desenvolupador.

6.2 Llibreries a utilitzar

Primerament, el llenguatge de programació escollit és el C++, ja que és un llenguatge orientat a objectes molt eficient i versàtil, i les llibreries que utilitzarem estan escrites en aquest llenguatge. Utilitzarem el compilador GNU C++, del paquet GCC.

Les llibreries que utilitzarem seran les QT per a la interfície gràfica i les LTI-Lib com a llibreria matemàtica i de tractament d'imatge.

D'altra banda, també ens caldrà les llibreries LAPACK, per a poder utilitzar les implementacions més eficients de diversos algorismes numèrics, com ara el SVD.

Si es compleixen els requisits de dependències de llibreries, no hi hauria d'haver cap problema per compilar i executar el codi que presentarem d'ara en endavant.

Capítol 7

Estereovisió amb dues càmeres

7.1 Introducció

Encara que l'objectiu del projecte és arribar finalment a un sistema estèreo format per una sola càmera, degut a la complexitat intrínseca en aquests sistemes ens hem proposat un objectiu intermedi: construir un sistema estèreo *clàssic* amb dues càmeres. Els algorismes implementats en aquesta fase serviran sense problemes per la posterior amb una sola càmera.

7.2 Funcionament del sistema estèreo

Com podem veure en la figura 7.1, podem dividir el funcionament del sistema en una sèrie de passos que s'executen de forma seqüencial:

i Adquirir una imatge de cada càmera. A més, farem el pre-processat de les imatges, que consisteix passar-les de color a escala de grisos i treure la distorsió gràcies a la calibració mitjançant Faugeras.

ii Extreure els punts característics de cada imatge. Aplicarem el filtre de Harris per extreure les cantonades dels objectes, les quals són els nostres punts característics.

iii Crear un conjunt de correspondències candidates. Tot utilitzant la geometria epipolar, crearem les correspondències entre els punts de la primera imatge amb els de la segona.

iv Eliminar *outliers*. Eliminarem els outliers aplicant l'algorisme de RANSAC, que ens retornarà una homografia que relaciona els punts de la primera imatge amb els de la segona.

v Triangulació. Per totes les correspondències, agafem els dos punts que la formen i construïm la matriu de triangulació, solucionant-la amb SVD, com hem vist en el capítol 5.

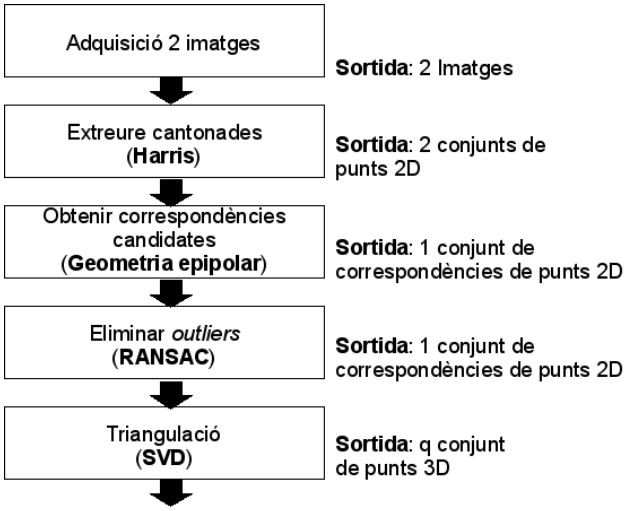


Figura 7.1: Etapes del sistema estèreo.

A continuació entrarem amb més detall en el desenvolupament del sistema estèreo. Primerament, abordarem el maquinari, és a dir, el muntatge manual. Després, mostrarem el programari i algorismes implementats.

7.3 Construcció del sistema estèreo

En aquesta secció veurem l'estructura de suport i les càmeres utilitzades en el sistema estèreo amb dues càmeres.

7.3.1 Model de càmera utilitzada

Les càmeres utilitzades són unes webcam NGS Show Cam Plus, de color amb una resolució de 640x480 píxels, com la de la figura 7.2. Naturalment, és una càmera de gama baixa, i per tant la qualitat d'imatge és bastant pobre. A més, té el problema que la pròpia càmera envia les imatges en format JPEG. Per a l'ús que ha estat dissenyada (webcam) va molt bé, ja que s'envia les imatges de forma més ràpida per què ja estan comprimides, però per a la nostra aplicació suposa un desavantatge, ja que el JPEG és un algorisme de compressió d'imatges amb pèrdua de qualitat.

El driver de Linux per controlar la càmera ja realitza la conversió de JPEG a bitmap, que és l'ideal per a les aplicacions de visió per computador. No obstant això sigui transparent, repercuteix en feina extra per a la CPU, la qual cosa farà disminuir les imatges per segon que puguem obtenir.

No obstant, el nostre objectiu no és la precisió, sinó demostrar que es pot construir un sistema estèreo amb una càmera, per tant, aquestes càmeres s'adequen a les necessitats del projecte.

Figura 7.2: Model de la càmera utilitzada durant el projecte.

7.3.2 Suport per a les càmeres

Tenir dues càmeres implica que ens cal un suport robust, ja que després de la calibració, aquestes no es poden moure. De fet, el que hem d'evitar és que es mogui una càmera vers l'altre, però sí que es poden moure les dues càmeres a l'hora. Per tant, la nostra solució esdevé un sistema *portàtil*, com podem veure en la figura 7.3. Tot el suport es pot moure, però en canvi les càmeres resten inamovibles dintre l'estructura. A les càmeres se li ha desmontat el suport de plàstic per tal de poder acoblar-les de forma correcta a la plataforma.

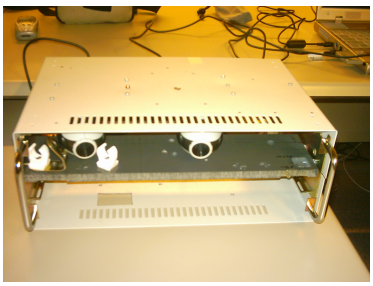


Figura 7.3: Suport per a l'escànner amb dues càmeres.

És important que el suport de les càmeres en la plataforma pugui ésser ajustat, ja que un dels principals de la construcció del suport és que les dues càmeres mirin la *mateixa* zona de l'escena. Això fa que no podem reaprofitar la rosca de la càmera que la unia a la seva pròpia base, ja que és fàcil que amb una manipulació constant es passi de rosca.

Per tant, hem modificat el suport de la figura 7.3 afegint un suport especial per a cada càmera. Aquest petit suport consta: d'una abraçadora que la subjecta, i que permet fer-la rotar en l'eix y, és a dir, amunt i avall, a més de rotar sobre ella mateixa. Aquesta abraçadora està collada a un suport de plàstic, que és el que permet el moviment per orientar les dues càmeres a la mateixa zona. En la figura 7.4 podem veure el resultat final del suport de la càmera.

Finalment, ja tenim a punt el sistema estèreo. Passem a veure el programari que controla el sistema i ens permetrà escanejar els objectes.

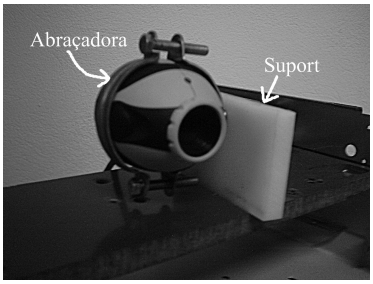


Figura 7.4: Suport especial per a la càmera que permet manipular-la fàcilment.

7.4 Programari desenvolupat

A continuació mostrarem els algorismes i classes programades per tal de desenvolupar el sistema estèreo. Ja que el sistema estèreo està dividit en diverses etapes, hem dissenyat el programa de forma modular, tal que permet realitzar proves en cada fase, de forma independent a les altres. També s'ha tingut en compte que fos prou versàtil per ésser utilitzat tant en quan tenim dues càmeres, com posteriorment quan només en tinguem una.

En el diagrama UML de la figura 7.5 podem veure les diferents classes que composen el sistema de visió estèreo. Cal dir que és una versió resumida, es mostren totes les classes però no pas tots els seus mètodes i atributs, per tal de millorar la llegibilitat.

Les classes s'han agrupat en funció de la seva funcionalitat, ja que, com podem veure, es poden dividir en filtres, en auxiliars per entrada/sortida, en les pròpies del sistema estèreo, entre d'altres.

7.4.1 El detector de cantonades Harris

Utilitzem el detector de cantonades de Harris que ve inclòs en les llibreries LTI-LIB, anomenat **harrisCorners**. Aquest treballa utilitzant els conceptes explicats en el capítol 3. En la figura 7.6 es pot veure el diagrama d'herència de la classe.

En el treball original del detector de Harris [Harris 1988], es va utilitzar un kernel del tipus [-2 -1 0 1 2] per a detectar el pendent tant en la direcció de x com la de y. No obstant, treballs posteriors, com el de [Schmid et al. 2000] han trobat que altres kernels derivats dels Gaussians poden donar resultats superiors. La funció del pendent utilitzada per la classe harrisCorners pot ésser escollida per millora l'efectivitat, tot i que per defecte utilitza la tradicional proposada per Harris.

Aquesta classe té diferents paràmetres amb els quals podem adaptar el seu funcionament, vegem-los.

- *Variance*: La variància utilitzada per Gauss. Si el valor és negatiu, llavors la variància es calcula de forma que el valor a $floor$(mida/2) sigui $1/(1 + floor$(mida/2)) vegades el valor a l'índex 0. Per exemple, per a una mida de 3, el valor a l'índex 1 seria 1/2 el valor del 0. Per defecte té el valor de -1.

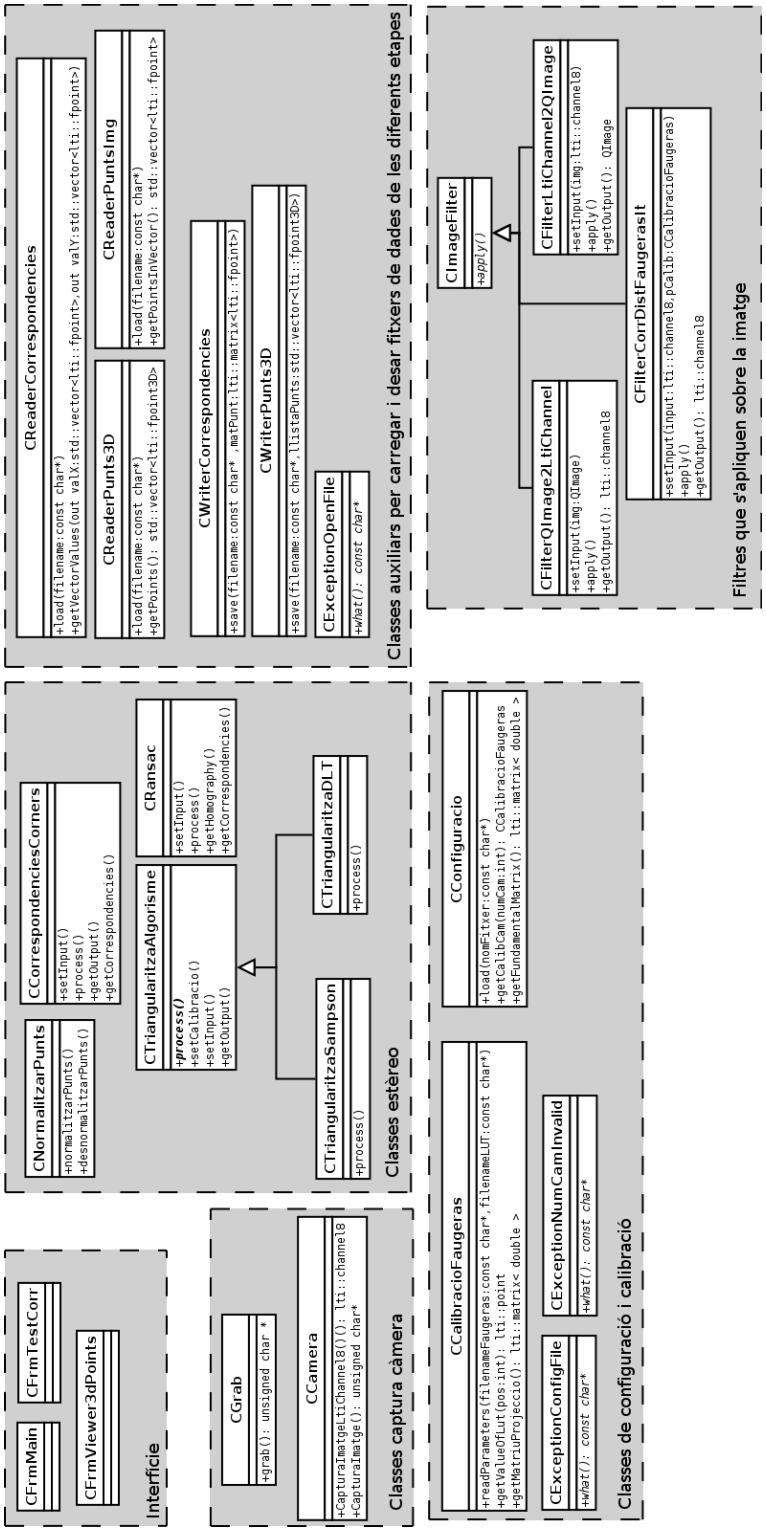


Figura 7.5: Diagrama de classes.

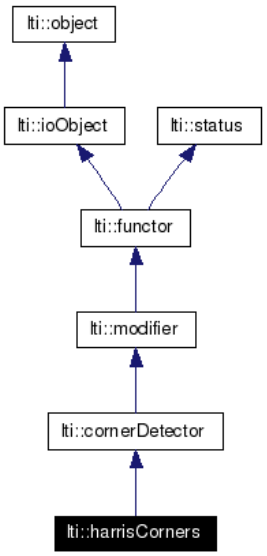
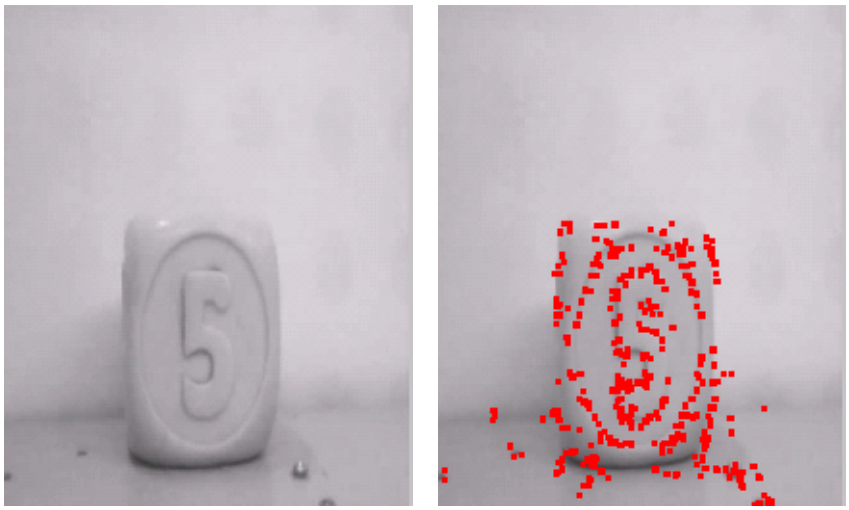


Figura 7.6: Diagrama d'herència de la classe harrisCorners de LTI-LIB.

- *kernelSize*: La mida del kernel de Gauss que s'utilitzarà per realitzar l'autocorrelació. Per defecte té el valor de 7.
- *maximumCorners*: El nombre màxim de corners que cercarem. En cas que vulguem un millor detall, es pot augmentar. El valor per defecte és 300.
- *scale*: És el valor d'escala, corresponent al paràmetre α que varem veure en el capítol 3.
- *localMaximaParameters*: És la funció utilitzada per detectar les cantonades, encarregada de trobar el màxim local d'una regió. Accedint a aquesta propietat es poden canviar els paràmetres de la funció de detecció.
- *gradientFunctorParameters*: La funció de pendent, que com hem comentat anteriorment pot ésser canviada. Per defecte utilitza el kernel [-2 -1 0 1 2] del treball original de Harris.

El mètode més important de la classe és el que porta per nom **apply**, que està sobrecarregat per diferents paràmetres d'entrada (imatges de color i escala de grissos, entre d'altres) i de sortida (imatge amb les cantonades marcades, llista de cantonades, etc.). Aquest mètode s'aplica sobre una imatge i retorna les cantonades d'aquesta.

En la figura 7.7 podem veure un exemple de detecció de cantonades en una imatge. Els paràmetres que hem utilitzat són els paràmetres per defecte, ja que realitzant proves amb els diferents valors dels paràmetres no hem trobat millores apreciables. Com que els objectes que escanejem són petits i amb no masses arestes, el número màxim de cantonades de 300 ja ens resulta correcte.



(a) Imatge original.

(b) Cantonades detectades en la imatge original.

Figura 7.7: Exemple de detecció de cantonades.

7.4.2 Correspondències per geometria epipolar

Per a trobar les correspondències candidates entre els punts característics de cada imatge, s'ha desenvolupat la classe **CCorrespondenciesCorners**. Aquesta utilitza la matriu fonamental per a cercar les cantonades sobre la segona imatge, com vàrem veure en el capítol 4. Veiem de quina manera treballa aquesta classe.

El mètode públic **setInput** especifica les dades d'entrada de la classe, les quals són:

- *llistaCorners*: La llista de punts característics de la primera imatge, els quals hem de trobar la seva parella en la segona per a formar la correspondència.
- *matriuFonamental*: La matriu fonamental, que ens servirà per a trobar la línia epipolar en la segona imatge, al llarg de la qual cercarem la correspondència.
- *img1*: Primera imatge de la qual s'ha tret els punts característics. La necessitem si volem comprovar els nivells de gris dels píxels si tenim més d'un punt com a correspondència candidata.
- *img2*: Segona imatge sobre la qual s'ha tret els punts característics. La necessitem si volem comparar els nivells de gris amb els de la primera imatge.
- *imgCorners*: Imatge on estan representades les cantonades trobades en la imatge. Quan s'obté la recta epipolar, es busca utilitzant les coordenades de la recta, les cantonades d'aquesta imatge.

El mètode **process** realitza el procés de trobar les correspondències candidates amb els paràmetres d'entrada donats, mentre que el mètode **getCorrespondencies** retorna com a paràmetre de sortida les correspondències candidates trobades.

La cerca de les correspondències candidates no és una tasca senzilla, just al contrari, és una de les més complexes que intervenen en la visió estèreo. Per això, aquesta classe té diferents paràmetres que permet ajustar-la per a trobar el conjunt de correspondències candidates millor possible. Aquests paràmetres es poden especificar amb els mètodes públics que comencen amb *setParameter* + el nom del paràmetre. Vegem a continuació els paràmetres disponibles.

- *MidaFinestra*: la mida de la finestra de cerca en la qual cercarem la cantonada. Degut a errors en la mesura, és poc probable que el punt característic que busquem caigui just en la recta epipolar trobada. Per tant, per cada punt de la recta epipolar obrirem una finestra centrada en aquest punt, dintre la qual buscarem el punt característic en qüestió. El valor per defecte és 5 píxels.

- *RestriccioLiniaEpipolar*: Si està cert, no busca sobre tota la línia epipolar. El que fa, és buscar a partir de la mateixa coordenada x de la primera imatge. Si coneixem la translació que hi ha hagut entre les dues imatges, podem utilitzar aquest paràmetre per reduir al màxim possible l'àrea de cerca. Per defecte està a fals.

- *PixelsRestriccioLiniaEpipolar*: Especifica els píxels sobre la restricció epipolar. Hem d'anar en compte, per que si posem un valor massa baix, és possible que no trobi cap correspondència, o que la trobada sigui un outlier. El valor per defecte és zero, ja que el paràmetre RestriccioLiniaEpipolar està desactivat. Un valor recomanat és 50-80 píxels.

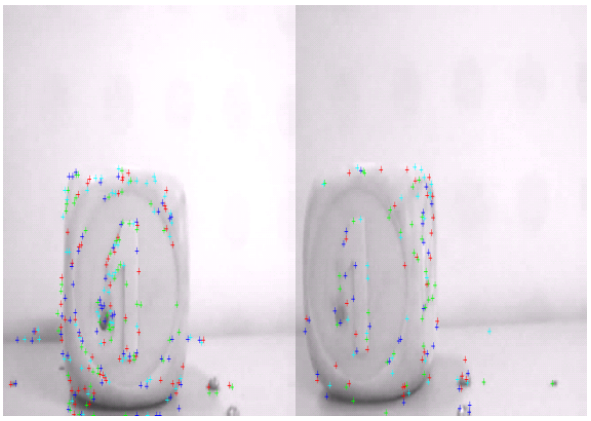
- *AssigCorrespColorSemblant*: Especifica si assignem només un punt característic per cada punt de la primera imatge. En cas que estigui a cert, entre tots els punts característics trobats al llarg de la línia epipolar, es queda amb el que tingui un nivell de gris similar amb el de la primera imatge, tot calculant la mitjana dels píxels del voltant. Si està a fals, assigna tots els punts característics com a correspondència, en espera que RANSAC elimini els outliers. No obstant, si posem el paràmetre a cert, filtrem molt els outliers i el RANSAC es comporta de forma més òptima. Per defecte està a cert.

- *FinestraColorSemblant*: Especifica els píxels de la finestra sobre la qual es calcularà la mitjana del nivell de gris dels píxels al voltant del punt característic candidat de la correspondència. El valor per defecte és 5.

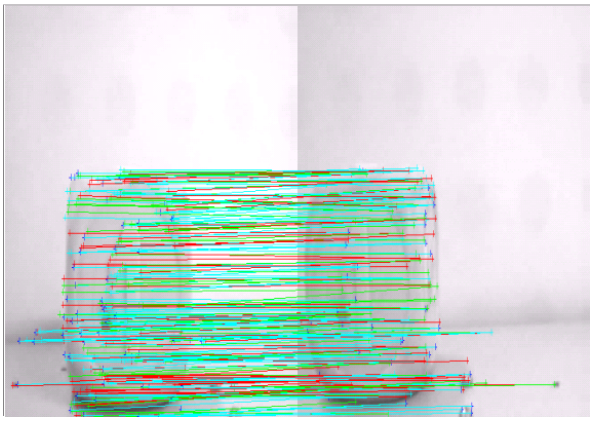
- *RepetirCorner2*: Especifica si es pot repetir el mateix punt característic per a dues correspondències diferents. Si els punts característics estan molt junts, pot ser que s'esculli com a candidat el mateix, per a vàries correspondències. Amb aquest paràmetre, si està a fals, un cop assignat el punt a una correspondència, el descarta per a les següents. A més, millora l'eficiència ja que el conjunt de punts candidats disminueix, la qual cosa fa que l'algorisme cada vegada tingui que comprovar menys punts. El valor per defecte és cert.

La majoria d'aquests valors estan representats en un fitxer XML de configuració que es carrega al inici, i permet canviar-los sense haver de variar el programa. Més endavant, veurem com és el fitxer de configuració.

En la figura 7.8 veiem un exemple de cerca de correspondències candidates. En la figura 7.8(a) mostrem només els punts característics que formen part d'una correspondència, mentre que en la figura 7.8(b) unim amb línies els punts de cada correspondència. Cal fer nota, que aquestes línies haurien de ser tant paral·leles com sigui possible, ja que les línies que no són paral·leles corresponen a outliers.



(a) Imatge amb les correspondències marcades.



(b) Imatge amb línies que uneixen les correspondències candidates.

Figura 7.8: Exemple d'assignació de correspondències candidates.

7.4.3 Estimador robust RANSAC

Després de trobar les correspondències candidates, hem d'eliminar els outliers tot utilitzant un estimador robust com el RANSAC. Aquest ajusta una homografia que relaciona els punts característics de la primera imatge amb els de la segona, per a poder formar les correspondències, com varem veure en 4.5.

L'algorisme RANSAC que utilitzarem és el que ens proporciona la classe **ransacEstimator** de les LTI-Lib, tot i que no s'utilitza directament, sinó a través d'una classe embolcall desenvolupada, anomenada **CRansac**. Passem primer a veure el funcionament de la classe ransacEstimator.

En la figura 7.9 es pot veure el diagrama d'herència de la classe. La classe ransacEstimator implementa el mateix algorisme vist en la secció 4.5. Per executar l'algorisme, cal cridar el mètode sobrecarregat **apply**, tot donant d'entrada les correspondències candidates trobades i obtenint de sortida la homografia.

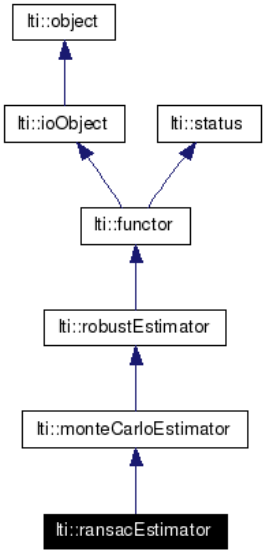


Figura 7.9: Diagrama d'herència de la classe ransacEstimator de LTI-LIB.

Aquesta classe té diferents paràmetres amb els quals podem adaptar el seu funcionament, vegem-los.

- *adaptiveContamination*: Ajusta el grau de *contaminació* després de cada conjunt candidat correcte. La contaminació sempre disminueix, mai augmenta. Aquesta paràmetre afecta al nombre d'iteracions que es realitzaran. No obstant, sempre acabarem si arribem al màxim d'iteracions especificades per paràmetre. Quan aquest paràmetre està activat, el mètode apply retorna cert encara que que els inliers suggereixen una contaminació major que la del paràmetre *contamination*. Per defecte és fals.
- *numCorrespondencesPerTrial*: Si és cert, utilitzarem el nombre de correspondències mínim per a cada iteració per estimar la transformació. En general, es recomana que s'utilitzi el nombre mínim de correspondències que calguin, el qual és trobat utilitzant estadística. Per defecte té el valor 10.

- *checkStop*: Si és cert l'algorisme para tant bon punt troba una transformació tal que el nombre d'outliers sigui inferior a la contaminació esperada (paràmetre contamination). Per defecte és cert.
- *contamination*: El grau de contaminació esperada. Per defecte és 0.5.
- *confidence*: El nombre d'iteracions en el mode adaptatiu depèn de la contaminació estimada i de la confidència, sota la qual el resultat és correcte. Per defecte té el valor de 0.9.
- *useMinCorrespondences*: Si és cert, s'utilitza el nombre requerit de punts, ignorant el valor del paràmetre numCorrespondencesPerTrial. Per defecte està a cert.
- *maxError*: L'error màxim per a una correspondència o la mida mitjana del residu. Per defecte és 0.8.

Segons les proves que hem realitzat, hem notat poca diferència variant algun dels paràmetres anteriors, així que, com que els resultats eren ja prou bons, hem fet servir els valors per defecte dels paràmetres, els quals ja es solen adaptar molt bé a tot tipus d'escenaris.

Com anteriorment hem vist, abans d'aplicar l'algorisme de RANSAC, hem de normalitzar el conjunt de punts d'entrada, i naturalment, després de trobar la homografia s'ha de desnormalitzar. La classe CRansac s'encarrega d'adequar les dades fent la citada normalització d'aquestes. En realitat, la classe CRansac fa ús de la classe **CNormalitzarPunts** per a la normalització del conjunt de punts.

La classe CRansac, té el mètode **setInput** en el qual rep com a paràmetre d'entrada les correspondències candidates. Llavors, el mètode **process** és qui normalitza els punts, aplica el ransacEstimator de les LTI-Lib i posteriorment desnormalitza la matriu homografia. A més, al fer de classe embolcall, la sortida no és pas la homografia, sinó que la funció **getCorrespondencies** retorna el conjunt de correspondències en els quals ja se li ha aplicat la homografia tot eliminant els outliers.

La classe CNormalitzarPunts aplica la normalització vista en (4.5.4 - pàg. 33). La creació d'una classe independent permet reutilitzar-la per altres algorismes que també necessitin de la normalització. Per tant, en aquesta classe trobem els mètodes privats **calcularTransformacioSimilitud**, **calcularFactorEscala** i **calcularCentroide** que s'encarreguen de realitzar els càlculs de la normalització. D'altra banda, tenim el mètode públic sobrecarregat **normalitzarPunts**, que com el seu propi nom indica normalitza el conjunt de punts. I els mètodes **desnormalitzarHomografia** i **desnormalitzarPunts** que fan l'acció contrària.

7.4.4 Mètode de triangulació

La triangulació és l'etapa que a partir dels dos punts de cada correspondència troba les coordenades del punt 3D a l'espai. Hem desenvolupat els dos mètodes que hem vist en el capítol 5.

La classe **CTriangularitzaAlgoritme** és la classe base dels mètodes de triangulació. És una classe abstracte i per tant no pot ésser instanciada directament. El mètode públic **setCalibracio** serveix per especificar la calibració

de les dues càmeres, ja que necessitarem la matriu de projecció d'aquestes per a triangular. El mètode públic **setInput** especifica l'entrada de dades, que en aquest cas són les pròpies correspondències. El mètode públic **process** és un mètode abstracte, i les classes derivades l'han de reimplementar. És en aquest mètode on es realitzen els càlculs per obtenir les coordenades 3D. Finalment, el mètode **getOutput** retorna un vector amb els punts 3D reconstruïts.

La classe **CTriangularitzaDLT** implementa el mètode homogeni DLT. Aquesta és subclasse de CTriangularitzaAlgoritme, i per tant reimplementa el mètode **process**. En aquest mètode es generen les equacions 5.4 de la pàgina 38, que es resolen utilitzant l'algorisme SVD.

D'altra banda, la classe **CTriangularitzaSampson** implementa l'aproximació de Sampson vista en (5.4 - pàg. 39), tot reimplementant el mètode **process**, ja que és una subclasse de CTriangularitzaAlgoritme. Aquest mètode primerament troba els punts corregits per tal que compleixin la restricció epipolar per després cridar al mètode process de la classe CTriangularitzaDLT, per a que els calculi resolent les equacions amb SVD.

La classe CTriangularitzaDLT fa ús de la classe **unifiedSVD** de les LTI-Lib. Aquesta classe en realitat és un embolcall per a dues classes: **fastSVD** i **singularValueDecomp**. Aquestes dues classes implementen el SVD de manera diferent. La classe fastSVD utilitza la llibreria LAPACK per a solucionar el SVD de forma molt eficient i ràpida, i per tant és el mètode desitjat. Si en el sistema on s'executa no hi ha la llibreria LAPACK instal·lada, llavors s'utilitza la classe singularValueDecomp que és menys eficient, implementada en C++ estàndard. Normalment, la classe unifiedSVD s'encarrega d'intentar utilitzar sempre la opció de fastSVD.

La classe unifiedSVD té els següents paràmetres:

- *sort*: Si és cert, els valors i vectors propis són ordenats descendentment. El valor per defecte és cert.
- *transposeU*: Si és cert, en comptes de retornar U, retorna la seva transposada. El valor per defecte és fals.
- *transposeV*: Si és cert, en comptes de retornar V, retorna la seva transposada. Per defecte és fals.
- *useDC*: Aquest paràmetre només té sentit quan s'executa el fastSVD. Determina si s'utilitza l'estratègia de divideix i venceràs (de l'anglès *divide and conquer*). Si és cert, s'utilitza el divideix i venceràs, que sol ser més ràpid per grans volums de dades, tot i que necessita més memòria. Si la reserva de memòria dinàmica falla, llavors utilitza el mètode tradicional. El valor per defecte és cert.
- *useMinDimensions*: Aquest paràmetre només té sentit quan s'executa el fastSVD. Si és cert, només es calcula $min(\text{files}, \text{columnes})$ de la matriu de vectors singulars. El valor per defecte és cert.
- *svdType*: Determina si ha d'utilitzar la classe fastSVD o singularValueDecomp. No es pot utilitzar fastSVD si no hi ha la llibreria LAPACK instal·lada en el sistema. Per tant, el valor per defecte varia depenent del sistema, tot i que utilitzarà fastSVD sempre que pugui.

A l'hora d'aplicar DLT o l'aproximació de Sampson, els resultats són molt semblants. La millora obtinguda no és apreciada a simple vista, i com que el nostre objectiu no és la precisió, podem utilitzar l'algorisme DLT o l'aproximació de Sampson indistintament.

7.4.5 Classes de configuració

Com hem vist, moltes de les classes tenen diferents paràmetres que poden ser ajustats. L'especificació d'aquests paràmetres en el codi dels algorismes faria la tasca de prova d'aquests molt pesada, ja que qualsevol canvi implicaria una recompilació. Per això, s'ha optat en traslladar aquests paràmetres en un fitxer XML de configuració, que es llegeix al inici de l'aplicació.

La classe encarregada és **CConfiguracio**, que poseeix el mètode **load**, que llegeix el fitxer xml passat per paràmetre. En cas que hi hagi algun error, aquest mètode llança una excepció del tipus **CExceptionConfigFile**. Un cop llegit, aquesta classe conté tota la configuració dels diferents paràmetres, a més de la calibració de les dues càmeres i la matriu fonamental del sistema de visió estèreo. Per accedir a la calibració d'una càmera s'utilitza el mètode **getCalibCam**, tot passant-li el número de la càmera que vulguem (com que utilitzem dues càmeres, ha der ser 1 ó 2). En cas que el número no sigui vàlid llançarà una excepció del tipus **CExceptionNumCamInvalid**. Aquest mètode retorna una classe del tipus **CCalibracioFaugeras**, que conté els paràmetres extrets per Faugeras, com vam veure en (2.7 - pàg. 15):

- Vector de translació del pla imatge respecte el món.
- Matriu de rotació del pla imatge respecte el món.
- Els paràmetres intrínsecs $u0$ i $v0$.
- Els paràmetres intrínsecs α_u i α_v.
- La matriu de projecció.
- Una LUT que permet de forma ràpida treure la distorsió radial de la imatge.

A continuació, mostrarem el format del fitxer XML, amb els paràmetres pertinents.

Fitxer de configuració XML

Veiem un exemple de fitxer XML de configuració, per a després explicar el significat de cadascuna de les etiquetes.

```
<Configuracio>
<FitxerCalibStereo>./data/stereo_2006_05_20.txt</FitxerCalibStereo>
<Calibracio numCam="1">
<FitxerCalibCam>
./data/CamEsquerra/dues_cam/CalFaugerasIter.txt
</FitxerCalibCam>
<FitxerLUTCam>
```

```
./data/CamEsquerra/dues_cam/EraseDistortionLUT.txt
</FitxerLUTCam>
</Calibracio>
<Calibracio numCam="2">
<FitxerCalibCam>
./data/CamDreta/dues_cam/CalFaugerasIter.txt
</FitxerCalibCam>
<FitxerLUTCam>
./data/CamDreta/dues_cam/EraseDistortionLUT.txt
</FitxerLUTCam>
</Calibracio>
<CercadorCorrespondencies>
<MidaFinestra>10</MidaFinestra>
<LiniaEpipolarRestringida
actiu="1"
pixels="80">
</LiniaEpipolarRestringida>
<AssigCornerColorSemblant
actiu="1"
midaFinestra="5">
</AssigCornerColorSemblant>
</CercadorCorrespondencies>
<!-- L'atribut de triangulacio pot ser:
"1": DLT.
"2": DLT + Sampson.
-->
<Triangulacio algorisme="1">
</Triangulacio>
<VideoForLinux actiu="0"></VideoForLinux>
</Configuracio>
```

La etiqueta **Configuracio** és el node arrel, el qual conté els diferents paràmetres, els quals els detallem a continuació.

- **FitxerCalibStereo**: Conté la ruta del fitxer en el qual hi ha desada la matriu fonamental del sistema de visió estèreo.
- **Calibracio**: Conté la calibració d'una càmera. L'atribut **numCam** indica de quina càmera es tracta, i és també el número amb el qual s'accedirà a través de l'aplicació. Aquest node conté diferents subnodes:
  - **FitxerCalibCam**: Conté els paràmetres extrets per la calibració de Faugeras Iteratiu, com ara la matriu de rotació, vector de translació, etc.
  - **FitxerLUTCam**: Conté la LUT amb la qual es pot treure de forma ràpida la distorsió radial de la imatge de la càmera.
- **CercadorCorrespondencies**: Conté els paràmetres de l'algorisme de l'assignador de correspondències candidates (més informació en (7.4.2 - pàg. 50), els quals són:

  - **MidaFinestra**: La mida de la finestra sobre la línia epipolar en la que cercarem els punts característics.
  - **LiniaEpipolarRestringida**: Conté el paràmetre de l'ús de la línia epipolar restringida. L'atribut **actiu** indica si s'ha d'utilitzar o no aquest paràmetre. L'atribut **pixels** indica els píxels de la línia epipolar en els quals cercarem els punts característics.
  - **AssigCornerColorSemblant**: Conté el paràmetre de l'ús de l'assignació de correspondències per color semblant. L'atribut **actiu** indica si s'ha d'utilitzar o no aquest paràmetre. L'atribut **midaFinestra** indica els píxels de la finestra sobre la qual calcularem la mitjana de nivell de gris.

- **Triangulacio**: Indica l'algorisme de triangulació a fer servir, mitjançant l'atribut **algorisme** que pot ser 1 per a DLT i 2 per l'aproximació de Sampson.

- **VideoForLinux**: Indica si s'ha d'inicialitzar el VideoForLinux per capturar directament des de la càmera, mitjançant l'atribut **actiu**. En cas que actiu estigui a zero (desactivat) es poden carregar imatges obtingudes prèviament.

7.4.6 Classes de filtre sobre les imatges

La classe **CImageFilter** és la classe base per els filtres sobre les imatges. Aquesta classe té el mètode virtual **apply**, que s'ha de reimplementar en les seves subclasses.

Primerament, trobem dues classes molt útils: **CFilterQImage2LtiChannel** i **CFilterLtiChannel2QImage**, que com els seus noms indiquen s'encarreguen de transformar de QImage a lti::channel8, i de lti::channel8 a QImage respectivament. Aquestes dues classes faciliten la integració de les classes de la llibreria LTI-Lib amb les QT.

La classe CFilterQImage2LtiChannel té el mètode públic **setInput** per especificar la imatge QImage que volem transformar, mentre que el mètode **apply** realitza la transformació a una imatge lti::channel8, que pot ser obtinguda amb el mètode **getOutput**. D'altra banda, la classe CFilterLtiChannel2QImage conté el mètode públic **setInput** per especificar la imatge d'entrada lti::channel8, mentre que el mètode **apply** realitza la transformació a una QImage, que pot ser obtinguda amb el mètode **getOutput**.

Finalment, la classe **CFilterCorrDistFaugerasIt** s'encarrega de treure la distorsió radial de la imatge aplicant la LUT trobada durant l'etapa de calibració. El mètode **setInput** rep com entrada la imatge de tipus lti::channel8 i una instància de CCalibracioFaugeras amb la calibració de la càmera. El mètode **apply** és l'encarregat de treure la distorsió, tot accedint a la LUT per saber a quina posició ha d'anar cada píxel. El que fa és generar una nova imatge. I per cada píxel sense distorsió de la nova imatge busca quin píxel de la imatge amb distorsió li toca, tot fent ús de la LUT. El mètode **getOutput** permet obtenir la imatge sense distorsió.

7.4.7 Classes auxiliars per a la captura d'imatges

Per a realitzar la captura d'imatges directament des de la càmera s'utilitza la API VideoForLinux 1. La classe **CGrab** és una classe embolcall que s'encarrega d'inicialitzar el dispositiu, i realitzar les captures utilitzant la API VideoForLinux. El mètode **grab** és l'encarregat de fer la captura. S'utilitza una àrea de memòria compartida fent servir la funció POSIX *mmap*, per tal d'agilitzar el màxim la captura i no perdre temps realitzant còpies en diferents *buffers*.

D'altra banda, la classe **CCamera** és la que fa de pont entre l'aplicació i la classe CGrab. El mètode **CapturaImatge** realitza una captura d'imatge, tot retornant un punter on hi ha desada aquesta. El mètode **CapturaImatgeLtiChannel8** fa el mateix, però retornant un lti::channel8, mentre que el mètode **GuardarImatgeBmp** guarda directament la imatge a disc en format BMP.

7.4.8 Classes auxiliars del sistema estèreo

Hem desenvolupat també una sèrie de classes auxiliars que tenen com a objectiu carregar i desar dades obtingudes en les diferents etapes del sistema estèreo. Això permet provar cada etapa de forma independent.

Primerament, la classe **CReaderPuntsImg** permet llegir amb el mètode **load** els punts característics o cantonades d'un fitxer, els quals poden ser obtinguts mitjançant **getPointsInVector**. Segonament, la classe **CReaderCorrespondencies** permet llegir amb el mètode **load** les correspondències desades en un fitxer, i obtenir-les amb el mètode **getVectorValues**. Finalment, la classe **CReaderPunts** permet llegir les coordenades de punts 3D reconstruïts per el sistema mitjançant el mètode **load** i obtenir-los amb el mètode **getPoints**. Cal dir, que si alguna de les classes anteriors troba algun problema al llegir al fitxer, llencen una excepció de tipus **CExceptionOpenFile**.

D'altra banda, tenim les classes **CWriterCorrespondencies** i **CWriterPunts3D** que permet desar correspondències i punts 3D en fitxers respectivament, mitjançant la funció **save** i els paràmetres adequats.

7.4.9 Classes que implementen les finestres

Les classes **CFrmMain**, **CFrmTestCorr** i **CFrmViewer3dPoints** són les que implementen les diferents finestres de l'aplicació. No entrarem gaire en detall en aquest aspecte, ja que la interfície és un aspecte secundari del projecte.

En termes generals, cal dir que en comptes d'utilitzar directament les classes generades per el programa QTDesigner, s'ha preferit crear noves classes que heredin de les primeres. Això fa que el codi sigui més net i disminueixi el temps de compilació. Per tant, la connexió dels *signals* es fa de manera manual, com es pot veure en els constructors de les classes. Els signals són la manera d'associar els events dels diferents controls a una funció determinada, anomenada *slot*.

7.5 Aplicacions auxiliars desenvolupades

En aquesta secció explicarem les aplicacions auxiliars que s'han desenvolupat a part de l'escànner de visió estèreo: el calibrador del sistema estèreo, el qual extreu la matriu fonamental, i el generador de punts, que genera fitxers amb coordenades de punts 3D útil per el procés de calibració.

7.5.1 Aplicació per obtenir la matriu fonamental

Fins ara, sempre havíem donat per assumit que teníem la matriu fonamental. Però ens cal implementar una aplicació per poder trobar-la. Aquesta aplicació es diu **CalibStereo**.

Conté dos finestres, implementades de la mateixa forma que s'ha fet amb el sistema estèreo, consulteu 7.4.9. Aquestes són **CFrmMain**, que és la finestra principal, i **CFrmCheckCalib** que permet comprovar la matriu fonamental, tot obtenint línies epipolars a partir d'un punt. En la finestra CFrmCheckCalib no entrarem en més detall, ja que funciona de forma similar a com ho fa el cercador de correspondències candidates del sistema estèreo. Vegem, en canvi, la finestra principal, ja que conté el codi per a trobar la matriu fonamental.

La finestra principal té per una banda la funció de cercar els centres de masses d'una regió marcada per l'usuari mitjançant el ratolí. La cerca d'aquests centres de masses la realitza la classe **CProcess**. Bàsicament, pregunta a l'usuari quants centres de masses hi ha, i a partir d'aquí divideix l'espai entre els centres de cada extrem de la regió en tantes parts iguals com centres de masses, sembra una llavor al mig de cada part, i busca el centre de masses en aquella àrea, fent créixer la llavor en cada direcció de x i de y.

Aquesta cerca de centres s'ha de fer per una imatge de cada càmera. És a dir, s'ha de fer per dues imatges pertanyent a la mateixa regió del món, vista per les dues càmeres. D'aquesta manera, obtindrem les coordenades dels mateixos centres de masses en el pla imatge. Aquestes coordenades són correspondències sabudes, i són les que utilitzarem per trobar la matriu fonamental, com vàrem veure en (4.3.3 - pàg. 30).

Per això, utilitzarem la classe **fundamentalMatrixSolverLMS** de les LTI-Lib. Aquesta classe utilitza un algorisme de mínims quadrats amb un RANSAC, per tal d'obtenir la matriu fonamental. El mètode **apply** s'executa sobre dos conjunts de punts, que seran els nostres centres de masses de cada imatge, i té com a sortida la matriu fonamental. L'algorisme 7.1 mostra el funcionament de fundamentalMatrixSolverLMS.

Algorisme 7.1 Algorisme de cerca de la matriu fonamental aplicat en fundamentalMatrixSolverLMS.

i Escollir un subconjunt de 8 punts aleatoris (l'algorisme també pot treballar amb més de 8 punts).

ii Estimar la matriu fonamental per aquest subconjunt utilitzant la descomposició en valors singulars.

iii Calcular l'error residual per cadascuna de les mostres. L'error serà la distància del punt amb la seva línia epipolar.

iv Agafar la mediana de les distàncies resultats i comparar-la amb la millor mediana fins ara (i canviar-la si és millor). També és posible aplicar RANSAC, agafant el màxim de les mostres obtingudes que siguin inliers.

Repetir fins arribar al llindar de la mediana de l'error residual o al màxim d'iteracions definides.

La funció **calcResidual** permet obtenir aquest error residual, tant sumat,

com desglossat per a cada centre de masses. La funció **buildPointMatrix** és una funció auxiliar, que crea una matriu amb els centres de masses a partir de les dues llistes d'aquests, per tal poder ésser passats a la funció calcResidual, que espera una matriu en comptes de llistes.

Aquesta classe conté els paràmetres que hem pogut veure en el seu algorisme:

- *numTrials*: El màxim d'iteracions que volem que faci l'algorisme.
- *threshold*: El llindar de la mediana mínima de la distància dels punts a les seves línies epipolars.

En la figura 7.10 es pot veure un exemple d'execució de l'aplicació. En la figura 7.10(a) es mostra com es marquen els centres de masses, mentre que en la figura 7.10(b) es pot veure la línia epipolar d'un punt, tot fent ús de la matriu fonamental trobada.

7.5.2 Aplicació genPunts3d per generar fitxers de patró

A l'hora de calibrar, s'ha d'especificar les coordenades 2D de la projecció dels patrons de calibració en el pla imatge, la qual cosa ho fa automàticament el programa de calibració utilitzat, desenvolupat en [Armangué 03]. No obstant, les coordenades dels punts 3D dels patrons de calibració en el sistema de coordenades món s'han d'introduir a mà, la qual cosa resulta una tasca molt feixuga i propensa a errors si el nombre de punts és elevat.

Per ajudar-nos en la tasca de la calibració, hem desenvolupat l'aplicació **genPunts3d**, que és una aplicació de consola que realitza preguntes a l'usuari del tipus: coordenades del primer punt de calibració, número de punts, separació en mil·límetres entre aquests, etc. Un cop ha recollit les dades, genera un fitxer amb les coordenades dels punts, que pot ésser introduït directament en el programa de calibració, simplificant molt el procés.

Aquest programa consta de la funció **main**, que realitza les preguntes a l'usuari tot recollint les dades i de la classe **CGenPunts**, que a partir de les dades genera el fitxer amb els punts 3D.

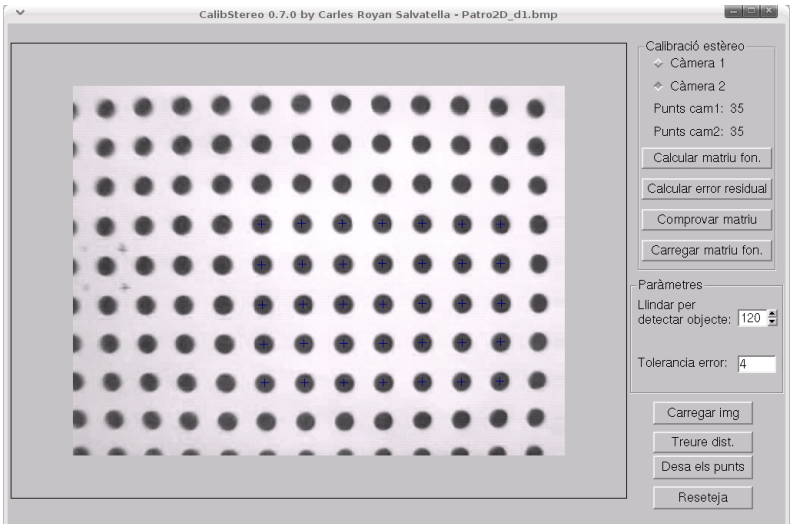
És un programa molt petit, senzill i simple però alhora altament útil que ha disminuït molt el temps que cal per a la calibració del sistema.

7.6 Configuració i calibració inicial del sistema

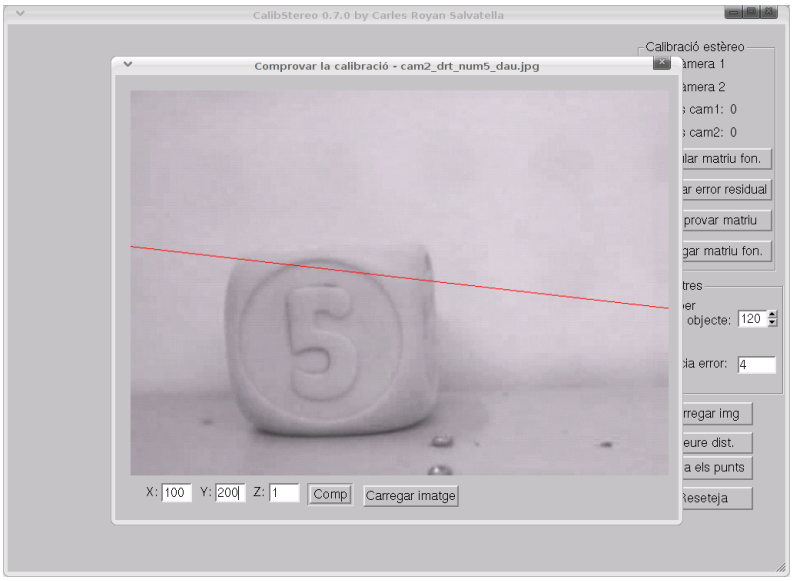
Els passos per calibrar el sistema passen per una primera etapa de calibrar les càmeres, per a després calibrar l'estèreo tot buscant la matriu fonamental que relacioni les dues càmeres.

Per a calibrar les càmeres farem ús del mètode de Faugeras amb distorsió radial que vàrem veure en (2.7-pàg.15). Cada càmera s'ha de calibrar per separat, per tant explicarem com fer-ho per una càmera, assumint que s'ha de fer de forma anàloga per l'altra. Utilitzarem un patró de punts, els quals sapiguem la distància entre centres. Aquest patró el posarem davant de les càmeres a 20 cm, que serà a la mínima distància a la qual treballarà l'escànner, de forma que quedi el més paral·lel possible a l'estructura de suport del sistema.

Llavors, caldrà ajustar la posició de les dues càmeres per a que observin la mateixa àrea del patró. El més senzill és realitzar una petita marca per



(a) Centres de masses marcats en el patró.



(b) Finestra de prova de la matriu fonamental trobada.

Figura 7.10: Exemple d'execució de l'aplicació CalibStereo.

orientar-nos i anar movent el suport de les càmeres fins a la posició desitjada. És recomanable que les dues càmeres quedin girades amb un angle semblant vers l'estructura, ja que així tindran la mateixa coordenada z de profunditat. Un cop hem trobat la posició ideal de cada càmera, s'han de collar les dues de

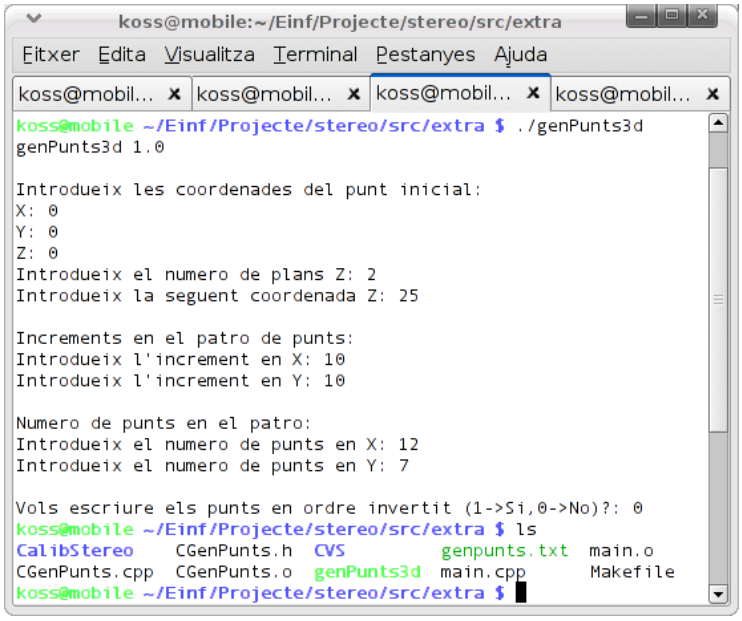


Figura 7.11: Exemple d'execució de l'aplicació genPunts3d.

forma permanent, ja que si varia la seva posició, s'hauria de tornar a realitzar el procés de calibració.

A partir d'aquí, realitzarem la calibració tot capturant la imatge del patró a 20 cm, i després capturant una segona imatge a 25 cm. Amb el programari realitzat per [Armangué 03] trobarem els centres de masses de les dues imatges i extreurem els paràmetres definits en la calibració per Faugeras (consultar en l'annex per veure el funcionament del programa).

Un cop hem calibrat les dues càmeres, passarem a fer el mateix amb el sistema estèreo. La calibració de l'estèreo consisteix a trobar la matriu fonamental que relacioni els punts de la primera imatge amb els de la segona utilitzant la geometria epipolar vista en el capítol 4. Utilitzarem el programa que hem desenvolupat per trobar la matriu fonamental (consultar en l'annex per veure el funcionament del programa). Aquest programa cerca els centres de masses de les dues imatges que carreguem, i realitza la correspondència d'aquests punts coneguts. A partir, de la correspondència i utilitzant SVD troba la matriu fonamental. Consulteu (4.3.3-pàg.30) per a més informació.

7.7 Resultats

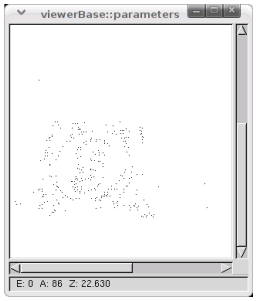
Degut a les limitacions de cost, no podem esperar uns resultats molt precisos. No obstant, els resultats són prou bons per el material que tenim.

En la figura 7.12 podem veure una captura de l'objecte que volem escanejar des de la perspectiva de cada càmera, mentre que a la figura 7.13 podem veure el resultat que ens proporciona el nostre sistema. En la figura 7.13(a) es mostra amb el visualitzador intern de la nostre sistema estèreo, mentre que en la figura 7.13(b) es mostra amb el programa Matlab. La forma i la proporció es mantenen

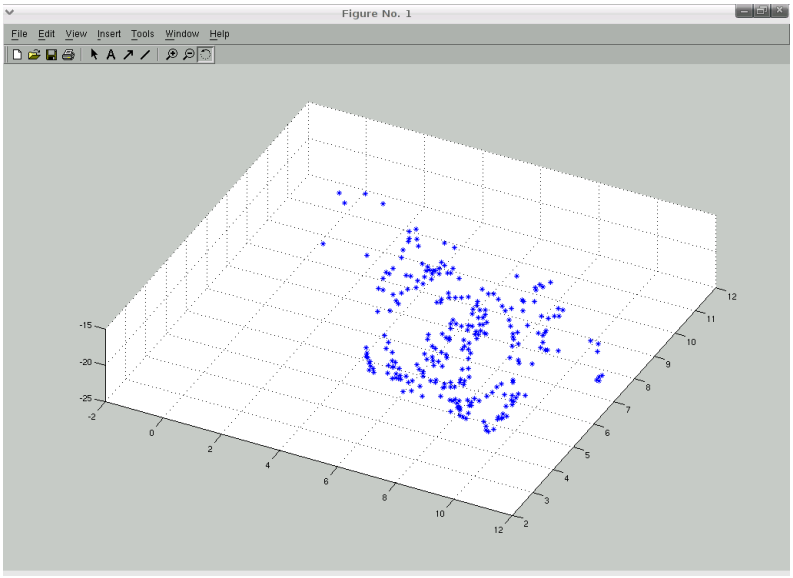
de forma correcta, encara que en certes zones hi ha una manca de punts. Això fa notar la importància del detector de cantonades per tal que trobi el màxim nombre de punts d'interès. Sobre això cal distingir dos casos: o bé no troba cantonades per què el contrast és fluix o la superfície massa plana, o bé no totes les cantonades tenen correspondència, ja que les condicions d'il·luminació poden variar entre les dues imatges donant un conjunt de cantonades diferent per a cada una. Per tant, els punts 3D finals es composaran per la intersecció dels dos conjunts de punts formant les correspondències.

També cal comentar que a part dels punts de l'objecte, tenim altres punts que són cantonades inherents a l'escena. En cas que vulguem estudiar l'objecte i prou, es poden eliminar aquells punts que cauen més lluny del centre de masses.

Figura 7.12: Captura de l'objecte amb la càmera esquerra i dreta respectivament.



(a) Reconstrucció visualitzada per l'aplicació.



(b) Reconstrucció visualitzada en Matlab.

Figura 7.13: Punts 3D obtinguts amb el nostre sistema estèreo.

Capítol 8

Estereovisió amb una càmera

8.1 Introducció

Després d'haver desenvolupat un sistema estèreo complet amb dues càmeres, passarem a construir l'escànner amb una sola càmera. El programari i algorismes desenvolupats en el capítol 7 són aplicables directament aquí, ja que la teoria de visió estèreo és la mateixa.

8.2 El truc òptic per simular dues càmeres

Per tal que el nostre sistema sigui estèreo, necessitem treballar amb dues vistes de la mateixa escena. Però com aconseguir-les si només tenim una càmera? Les hem de simular posant una òptica davant la càmera. Ens cal que aquesta òptica tingui un índex de refracció molt elevat, per què el que volem aconseguir és que la imatge que capturi la càmera estigui dividida en dues parts, i que cadascuna d'aquestes constitueixi una vista de l'escena. En la figura 8.1 es mostra la òptica ideal, mentre que en la figura 8.2 es mostra el resultat esperat per aquesta òptica.

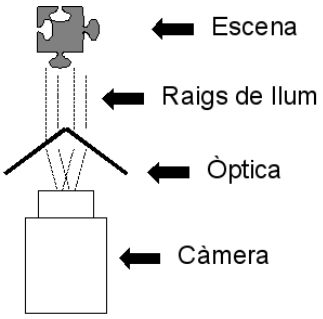


Figura 8.1: Òptica ideal amb un alt índex de refracció per simular dues càmeres.

Per a poder simular aquestes òptiques hem optat per el metacrilat. Agafant un tall de metacrilat de 10 mm de gruix i posant-lo davant la càmera, veiem

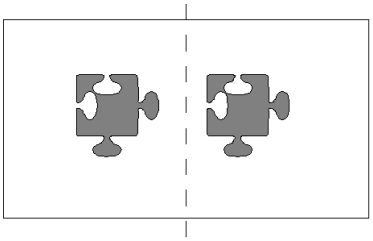


Figura 8.2: Resultat esperat de l'òptica ideal.

que té un índex de refracció important, tot traslladant la imatge. No obstant, al posar els dos talls de la mateixa manera que en la figura 8.1 veiem que el resultat difereix del què esperàvem. L'índex de refracció no és prou gran, donant com a resultat que l'objecte queda traslladat lleugerament en la zona central de la imatge, tant cap a l'esquerre com a la dreta. A més, en el centre de la imatge es veu la zona on s'ajunten els dos talls de metacrilat. Aquest efecte el podem observar en la figura 8.3.

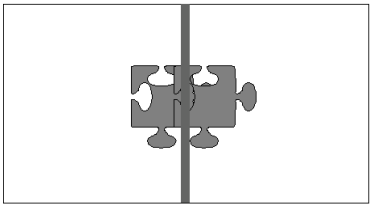


Figura 8.3: Resultat obtingut amb el metacrilat de 10 mm.

Això significa que necessitem un metacrilat més gruixut per tal d'augmentar l'índex de refracció. Però això comporta dos problemes: d'una banda el cost, un metacrilat més gruixut de 10 mm pot costar trobar-lo i és car. D'altra banda, per poder fer desaparèixer la molesta juntura dels dos talls de metacrilat, s'hauria de fer tallar tots dos en biaix de 45 graus, per tal que s'acoblessin i minimitzin la zona de la juntura. Realitzar aquest tall també comporta un cost elevat que no ens podem permetre. Veiem com hem resolt els impediments trobats.

8.3 Obtenció de dues vistes amb un tall de metacrilat

Mentre més gruixut és un tall de metacrilat, major índex de refracció té, i per tant la imatge queda més traslladada. Utilitzant un tall de metacrilat de 10x10 mm i 20 mm de gruix s'aprecia el desplaçament.

Llavors, en comptes de partir la imatge en dos, i simular que cada part és una captura d'una càmera, el que farem és agafar dues imatges, en les quals el metacrilat es trobi en una posició diferent, per tal de simular que tenim dues càmeres, en la figura 8.4 tenim un exemple.

D'aquesta manera, aconseguim tenir les dues vistes amb una sola càmera. No obstant, ens apareix un nou problema: les reflexions. Al ser tan gruixut, el

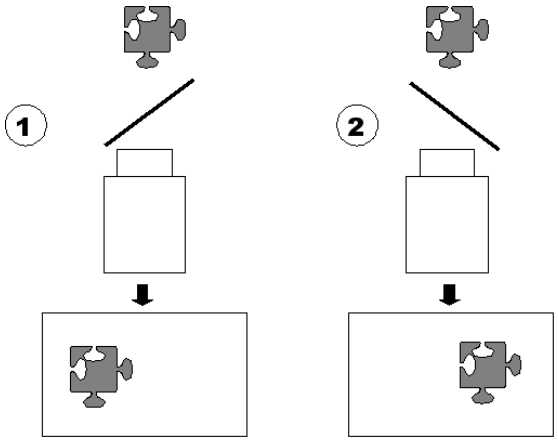


Figura 8.4: Fem la captura en dos passos, posant el metacrilat en posicions diferents.

metacrilat ha agafat una mica de propietats de mirall, reflexant fonts de llum i objectes que tingui a darrera. Naturalment, aquests reflexos ens apareixen en les imatges capturades, tot introduint soroll.

En qüestió a la reflexió de fonts de llum ho podem solucionar movent l'estructura, assegurant-nos que no hi hagi cap llum al costat. Però per el problema del reflex dels objectes ja no és tan senzill. Finalment, s'ha optat per construir una nova estructura per el sistema amb una càmera. Aquesta estructura té dos particularitats:

- Uns rails per situar el metacrilat en les dues posicions. Això és molt important ja que un cop es calibri la càmera amb el metacrilat en una posició, ens hem d'assegurar que les següents captures estigui allà mateix. En cas contrari la calibració perdria la validesa.
- A cada costat de la càmera hi ha uns pannells de fusta, de tal manera que és impossible que aparegui reflex de fonts de llum ni d'objectes del voltant.

A la figura 8.5, veiem el nou suport junt al metacrilat i el patró de calibració.

8.4 Configuració i calibració inicial del sistema

La calibració i configuració és molt semblant a la trobada quan teníem dues càmeres. No obstant, hem de pensar que les dues càmeres les simulem amb el metacrilat.

Primerament, calibrarem les dues càmeres simulades, posant el metacrilat en les dues posicions. Per tal de reduir l'error en la calibració, el que farem serà:

i Posicionar el metacrilat en la posició 1, que simularia la primera càmera.

ii Realitzar la captura del primer pla del patró de calibració.

iii **Sense moure el metacrilat**, desplaçarem el patró de calibració a la segona posició i realitzarem una segona captura.

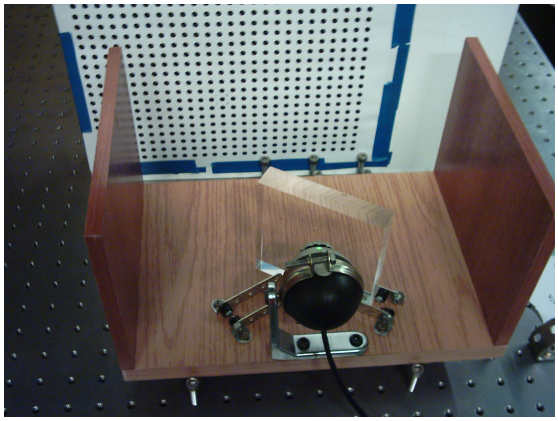


Figura 8.5: Estructura de suport final per el sistema estèreo amb una càmera.

iv Amb les dues imatges calibrarem els paràmetres de la càmera.

v Moure el metacrilat en la posició 2, i repetir els passos anteriors.

És important no moure el metacrilat entre la captura dels dos plans d'una mateixa càmera, ja que encara que tinguem els rails, sempre hi haurà un error de posicionament. En aquest cas, la càmera s'ha calibrat a 18 i 20.5 cm, més propera que en el cas anterior, i amb uns punts de calibració més petits, per tal de tenir més centres de masses i obtenir un resultat més acurat.

Per a trobar la matriu fonamental, farem igual que el cas anterior, utilitzant dos imatges del patró de calibració a la mateixa distància. Extraurem els centres de masses i calcularem la matriu a partir d'aquests.

8.5 Resultats

Els resultats no han estat tant bons com el sistema estèreo amb dues càmeres. Les reconstruccions obtingudes són molt poc precises. Encara que mantenen l'estructura a grans trets, aquesta es deforma, provocant molt d'error.

Hem realitzat moltes calibracions i provatures, i no hi hem pogut reduir l'error obtingut. Finalment, hem arribat a la conclusió que aquest error és degut al posicionament del metacrilat. Encara que tinguem els rails, i anem amb molta cura, és inevitable un error de posicionament d'aquest ens produeix aquest error en la reconstrucció.

Aquest error de posicionament és acumulatiu, ja que ens produeix error en la calibració de les dues càmeres, és a dir, de les dues posicions del metacrilat, i de la matriu fonamental. No obstant, si utilitzéssim dos metacrilats estàtics, del tipus *car* descrit anteriorment, ens desapereixeria aquest error i obtindríem uns resultats propers al de les dues càmeres.

En la figura 8.6(a) i 8.6(b) podem veure la imatge esquerre i dreta respectivament de l'objecte que volem reconstruir. I en la figura 8.7 podem veure el resultat de la reconstrucció, el qual poseeix un alt grau d'error i soroll, introduït per els desplaçaments del metacrilat necessaris per a cada captura.

(a) Vista esquerra del metacrilat.

(b) Vista dreta del metacrilat.

Figura 8.6: Imatge original amb les dues vistes del metacrilat.

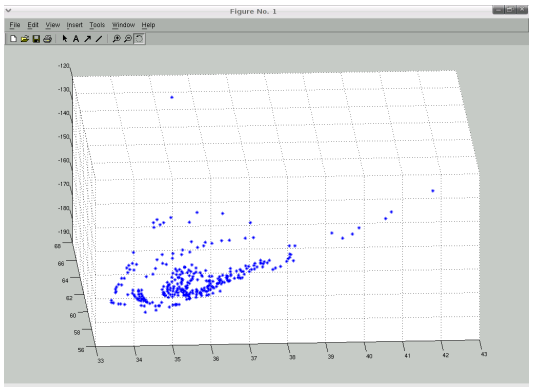
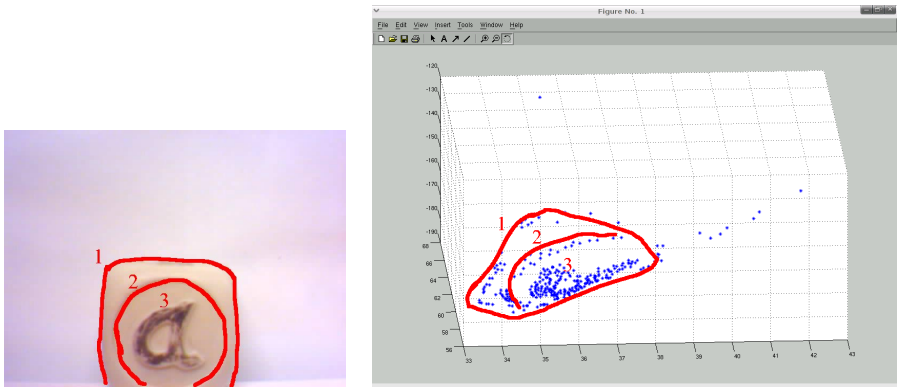


Figura 8.7: Resultat de la reconstrucció amb una càmera.

En la figura 8.8 podem veure marcades les parts reconeixibles de la imatge original amb la seva reconstrucció. En la figura 8.8(a) podem veure que el número 1 correspon al contorn del dau, mentre que el 2 és el cercle interior i finalment el 3 correspon a la lletra a. Aquestes marques es poden també veure en la figura 8.8(b), però amb una pèrdua notable de la forma i estructures originals.

Finalment, per a mostrar l'error en el posicionament del metacrilat, reconstruirem el patró de calibració. Ens saltarem els passos de detecció de cantonades, ja que els punts característics seran els centres de masses. Tampoc realitzarem cerca de correspondències, ni RANSAC, ja que les correspondències les podem assignar a mà, ja que els centres de masses són punts coneguts i tots inliers. Per tant, si reconstruïm directament els centres de masses, hauríem de tenir un pla amb tots els punts equidistants a la mateixa z. No obstant, el resultat de la figura 8.9 ho desmenteix, mostrant realment l'error introduït per el desplaçament del metacrilat que s'ha de realitzar per a cada captura.

Podem concloure, que la visió estèreo amb una sola càmera és possible utilitzant l'enginy per a simular-ne dues. No obstant, la solució escollida en el present projecte presenta massa error, i no permet unes obtenir reconstruccions massa



(a) Captura del dau amb les marques de les parts reconstruïdes.

(b) Captura de l'objecte reconstruït amb parts reconeixibles marcades.

Figura 8.8: Resultat de la reconstrucció amb una càmera amb marques sobre l'original.

bones. Ambtot, esperem que sigui una base sòlida per a futures millores, ja que els resultats resulten prometedors.

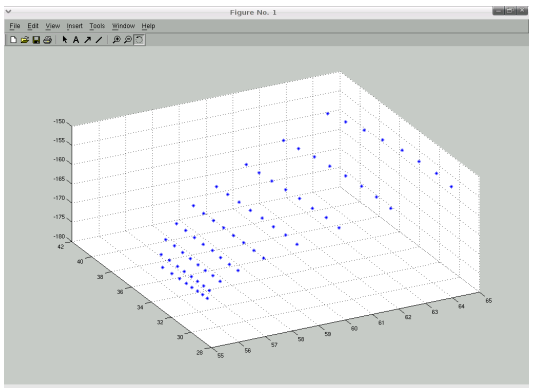


Figura 8.9: Resultat de la reconstrucció dels centres de masses d'un patró de calibració.

Part III

Conclusions finals

Capítol 9

Recull de conclusions

9.1 Cost i hores trigades

A continuació es mostrarà una temporització orientativa del que ha costat realitzar el projecte. S'ha de tenir en compte que aquesta temporització és aproximada, i només serveix per tenir una idea del cost de la seva implementació.

Tot seguit, es mostrarà un gràfic en el que es detalla el temps dedicat a cada tasca:

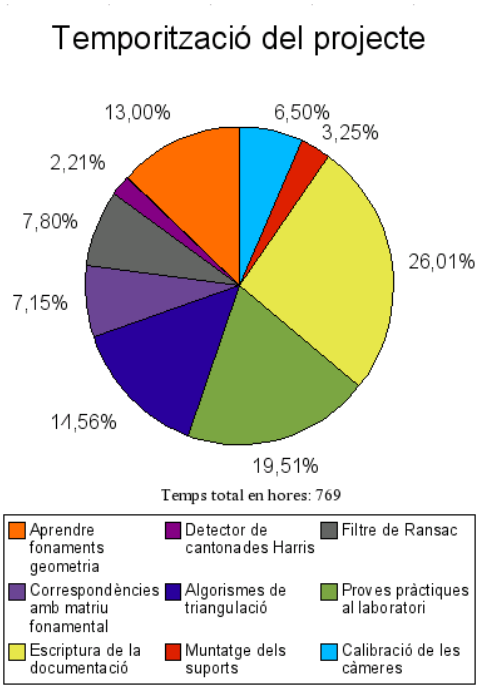


Figura 9.1: Temporització orientativa del projecte.

Com podem observar, la tasca més llarga és la de documentació, seguida per les proves de laboratori. Les proves de laboratori són molt importants ja que

han permès anar evolucionant el projecte fins al seu estat final. Seguidament, trobem les tasques relacionades amb les diferents etapes de l'escànner estèreo, sent les més complexes (i per tant les que més hores s'ha trigat) la triangulació i la geometria epipolar, els dos pilars bàsics del sistema estèreo.

També podem fer un petit anàlisi del cost material que costa l'escànner 3D, sense tenir en compte les hores de treball. A la taula 9.1 es mostra els components més importants utilitzats durant el projecte. La suma de tot ascendeix a 86 €, la qual cosa és un pressupost ínfim per a un escànner 3D. Encara que calgui sumar encara les hores de feina, hem aconseguit la fita de desenvolupar un escànner 3D amb visió estèreo de baix cost.

Descripció	**Cost unitari**	**Unitats**	**Total**
WebCam NGS Show Cam Plus	35 €	2	70 €
Metacrilat de 10x10mm i 20mm de gruix	13 €	1	13 €
Fustes i altres per el suport final	3 €	1	3 €

Taula 9.1: Cost material del projecte.

9.2 Aplicació de l'estereovisió amb una càmera

En la visió estèreo normalment s'utilitza sistemes basats en dues càmeres o basats en una càmera i llum estructura (un làser). Per el primer sistema, ens cal un suport on subjectar les dues càmeres de forma robusta, com hem pogut comprovar en el present projecte. Per el segon, cal un suport per la càmera i un sistema motriu que mogui o bé el làser o bé l'objecte a escanejar. Això és degut a que els sistemes amb càmera i làser captura seccions de l'objecte, les que són il·luminades per el làser. Per tant, necessiten un sistema motriu que faci fer un recorregut al làser per capturar n seccions d'un objecte i reconstruir-lo. Ambdós sistemes necessiten construir una estructura considerable que fa difícil la seva miniaturització.

En canvi, la proposta de desenvolupar un sistema estereovisió amb una càmera obre noves portes per a la miniaturització dels escàners 3D. Com hem vist, aquest sistema només necessita una càmera i una òptica adequada. Aquest fet possibilita la seva aplicació en àrees de medicina, com per exemple les sondes introduïdes en el cos humà per descobrir càncer i altres malalties. L'obtenció de representacions 3D pot ser de gran ajuda per a la diagnosi dels metges. D'altra banda, també podem pensar en aplicacions on s'hagi d'extreure dades de terrenys desconeguts amb un accés difícil per a sistemes que no siguin molt petits. És a dir, àrees on sigui important la miniaturització i es pugui aplicar visió per computador tridimensional es poden beneficiar del desenvolupament de sistemes estèreo amb una sola càmera.

Un altre avantatge és la mobilitat. Els sistemes de dues càmeres o càmera-làser solen ser més pesats i estàtics. En canvi, un sistema que consti només d'una càmera permet ser més lleuger i transportable, la qual cosa el converteix en un sistema que es pot utilitzar en aplicacions que requereixen un cert desplaçament.

9.3 Conclusions

A continuació exposarem les conclusions a les que podem arribar després d'haver desenvolupat aquest projecte.

La visió estèreo és complexa com ja s'ha demostrat al llarg del projecte. Per aquesta raó s'ha dividit el problema en dues parts: primerament desenvolupar un sistema estèreo *clàssic* amb dues càmeres per després passar al sistema estèreo amb una sola càmera, tot utilitzant els algorismes programats en l'etapa anterior. Això ha permès suavitzar la corba de dificultat, ja que sinó hagués estat excessiva.

La calibració de la càmera és un pas que a vegades s'ignora en aplicacions de visió per computador, encara que és molt important. La calibració ens permet d'una banda eliminar la distorsió introduïda per les lents de la càmera, i de l'altre obtenir la relació de les coordenades dels punts 3D de l'escena amb els punts projectats en el pla imatge. En els sistemes estèreo és molt important, ja que necessitem aquesta relació per triangular les correspondències i reconstruir així el punt 3D a partir de dues imatges.

La idea original era aconseguir un sistema estèreo amb una sola càmera i un parell de vidres davant d'aquesta que fessin l'efecte òptic de dues càmeres. Capturant una imatge, obtindríem dues vistes diferents de l'escena, d'igual com passa amb un sistema dual. No obstant, hem hagut de variar la implementació, ja que aconseguir la idea original, tot i ser possible, hagués resultat molt car, i per tant no complírem l'objectiu que fos de baix cost. Aquest canvi ha consistit en posar únicament un vidre amb un índex de refracció elevat, un tall de metacrilat de 20 mm de gruix, el qual posat en diferents posicions realitzava desplaçaments en la imatge. Per tant, realitzant dues captures amb el vidre posicionat de forma diferent hem simulat les dues càmeres.

Així doncs, hem demostrat que podem construir sistemes estèreo amb una sola càmera, tot fent ús de tècniques òptiques per simular la carència d'una segona. Però els resultats són molt millorables, ja que degut a la solució escollida: un metacrilat situat en dues posicions diferents, tenim molt d'error i soroll en la reconstrucció. No obstant això, aquests resultats són esperançadors i esperem que sigui una sòlida base per a futures millores.

Per tant, podem concloure que els sistemes estèreo no necessiten dues càmeres, sinó que necessiten dues vistes, és a dir, dues imatges diferents de la mateixa escena. Quan diem diferents, ens referim a que se li hagi aplicat a la imatge una transformació de translació i rotació. La forma més fàcil d'obtenir les dues vistes és utilitzar dues càmeres i tenir una vista per càmera, però no és l'única. Hem comprovat que amb una sola càmera i utilitzant l'enginy podem obtenir també dues vistes diferents, eliminant la dependència de la segona càmera i repercutint en un sistema més compacte. Així, la teoria dels sistemes estèreo (geometria epipolar, triangulació, etc.) no està basada en partir d'imatges de dues càmeres, sinó en partir d'imatges de dues vistes.

Cal afegir també que hem comprovat que la il·luminació de l'escena és un aspecte molt important a tenir en compte en qualsevol aplicació de visió. Hem hagut d'adequar el suport del sistema per tal d'obtenir una il·luminació adequada, ja que sinó ens distorsionava les imatges i no podíem obtenir resultats.

Finalment, cal assenyalar que el sistema de visió amb una càmera pateix els inconvenients associats als sistemes de visió estèreo: els problemes d'oclusió. Aquests problemes es resumeixen en que poden haver-hi objectes visibles per a

una vista, que quedin ocults per a l'altre, per què hi ha objectes davant seu que el tapen.

9.4 Treballs futurs

En aquesta secció exposarem diverses millores que es poden realitzar en el projecte presentat, que algunes d'elles no s'han pogut realitzar, bé per que no constaven en els nostres objectius o bé per la seva complexitat que les posicionaven fora del nostre abast.

Primerament, com ja hem comentat, es pot millorar la òptica per a simular les dues càmeres. En comptes d'un vidre que cal moure per obtenir cada imatge, es poden unir dos vidres tallats amb biaix. Això comportaria que no s'hauria de motoritzar el moviment del vidre i a més permetria obtenir les dues vistes a partir d'una sola captura d'imatge (en el prototip desenvolupat calen dues captures). D'aquesta manera, faríem desaparèixer l'error que tenim per culpa del desplaçament que hem d'aplicar al metacrilat per a realitzar les captures de les dues vistes.

A partir de la millora anterior, es pot treballar per a millorar la precisió i la repetitivitat del sistema, dues propietats que són molt importants si volem veure aplicacions reals, fora del laboratori, d'aquest tipus de sistemes. Amb la millora del material (càmera, vidres, etc.) s'hauria de percebre també una millora en la precisió dels resultats.

Un altre punt a millorar és que s'ha ignorat els problemes d'oclusió que tenen normalment els sistemes estèreo. Aquests problemes es presenten en dues càmeres, per punts que es veuen en una però que queden ocults en l'altre per a la superposició d'objectes. S'hauria de fer un estudi per comprovar si els problemes d'oclusió també afecten als sistemes estèreo amb una càmera, i en quina mesura.

Per a l'etapa de triangulació s'han implementat dos algorismes simples: el DLT utilitzant SVD i el l'aproximació de Sampson. Com a treball futur quedaria implementar nous algorismes de triangulació que fossin més precisos i eficients als aportats actualment.

Com es pot veure, els resultats presentats es basen en núvols de punts, els quals s'han reconstruït gràcies a la triangulació. Seria una millora visual important unir aquests punts mitjançant polígons, per a crear una trama mallada, la qual permet reconèixer molt millor els objectes i afegir el concepte de superfície i cara.

Apèndix A

Notació matemàtica

A.1 Introducció

Aquest apèndix sintetitza la nomenclatura utilitzada en la matemàtica present en el projecte. No obstant, la notació seguida és la comuna en l'àmbit de visió per computador, s'ha considerat oportú detallar-la per a que no presenti cap barrera a l'hora d'entendre els conceptes exposats.

A.2 Matrius i enumeracions

Les matrius es designen amb una lletra majúscula per anomenar-les. Les matrius estan formades per files i columnes, i per especificar quantes files i columnes la formen escrivim: $m \times n$, on m són les files i n les columnes. Per referenciar un element d'una matriu, s'utilitza dos subíndexs, el primer designa la fila, mentre que el segon ho fa per a la columna. Per exemple, podem escriure la matriu A de 3×4 amb els següents elements:

$$A = \begin{pmatrix} f_{11} & f_{12} & f_{13} & f_{14} \\ f_{21} & f_{22} & f_{23} & f_{24} \\ f_{31} & f_{32} & f_{33} & f_{34} \end{pmatrix}$$

Com podem veure, l'element situat en la primera fila i en la tercera columna és f_{13}, mentre que l'element situat en la segona fila i en l'última columna és f_{24}.

Hi ha dos operacions unitàries molt comunes en les matrius. D'una banda, la inversa (recordem que la matriu inversa és aquella que multiplicada per la pròpia matriu dóna la matriu identitat) que la simbolitzem amb el superíndex -1. D'altra, la matriu transposada (permutar files per columnes) s'indica amb el superíndex T. Així, per indicar la matriu inversa de A, seria A^{-1}, mentre que la matriu transposada seria A^T.

A l'hora de referir-nos a un punt, ho farem també amb una lletra, però normalment en minúscula, per exemple x. Si ens referim a un punt 3D de l'escena, llavors l'escriurem en majúscula. Si el punt forma part d'una matriu, o d'una equació en forma matricial, farem referència al punt com a x^T, ja que els punts són sempre matrius d'una sola columna i tantes files com dimensions tingui. Si volem distingir un punt d'un conjunt de punts, utilitzem el subíndex. Així, si x_i és el punt i del conjunt de punts de x.

A.3 Sistemes de coordenades

{H} defineix un sistema de coordenades H, que està composat per un origen O_H i per dos $\{X_H, Y_H\}$ o tres $\{X_H, Y_H, Z_H\}$ eixos, depenent del nombre de dimensions definides.

En el projecte apareixen els següents sistemes de coordenades:

- $\{W\}$=$\{O_W, X_W, Y_W, Z_W\}$ defineix el sistema de coordenades món, és a dir a l'espai de tres dimensions, on tenim l'escena o objecte a escanejar.
- $\{C\}$=$\{O_C, X_C, Y_C, Z_C\}$ defineix el sistema de coordenades de la càmera, amb el punt focal O_C com a origen.
- $\{I\}$=$\{O_I, X_I, Y_I, Z_I\}$ defineix el sistema de coordenades del pla imatge, amb el punt origen situat en la cantonada superior esquerra de la imatge.

Quan volem especificar un punt junt amb el seu sistema de coordenades, fem servir el superíndex davant d'aquest. Per exemple, el punt ${}^H P$ correspon al punt P expressat en el sistema de coordenades $\{H\}$, per tant tindríem que ${}^H P = ({}^H X, {}^H Y, {}^H Z)$. Tot punt està sempre relacionat amb un sistema de coordenades. No obstant, en algunes fórmules es pot prescindir el sistema de coordenades del punt si aquest queda explicitat de forma clara en el text, i naturalment no hi ha cap operació de canvi de sistema de coordenades involucrada.

D'altra banda, quan volem realitzar canvis de sistema de coordenades, apareix la següent notació: ${}^C P_W = ({}^C X_W, {}^C Y_W, {}^C Z_W)$ correspon a un punt 3D del món expressat en el sistema de coordenades $\{C\}$.

Per realitzar una transformació entre dos sistemes de coordenades s'utilitza una matriu de transformació. Per exemple, per transformar del sistema de coordenades $\{H\}$ al $\{J\}$ podem utilitzar la matriu ${}^J K_H$, expressada com:

$$ {}^J K_H = \begin{pmatrix} {}^J R_H & {}^J T_H \\ 0_{1\times 3} & 1 \end{pmatrix} $$

En l'anterior equació, $R = (r_1, r_2, r_3)^T$ és la matriu de rotació que expressa l'orientació de $\{H\}$ respecte el sistema de coordenades de $\{J\}$. R pot ésser expressada en tres angles de rotació, α, β o γ. D'altra banda, $T = (t_x, t_y, t_z)$ expressa la posició de l'origen de $\{H\}$ respecte $\{J\}$.

Apèndix B

Fonaments matemàtics

B.1 Introducció

En aquest apèndix explicarem varis conceptes matemàtics que apareixen en la memòria, però que per evitar confusions són preferibles explicar-los separadament del text principal.

B.2 SVD

B.2.1 Descripció

La descomposició en valors singulars (de l'anglès *Singular Value Descomposition* o SVD) és una de les descomposicions matricials més útil, particularment per a la computació numèrica. Donada una matriu quadrada A, la factorització d'aquesta esdevé com $A = UDV^T$, on U i V són matrius ortogonals, i D és una matriu diagonal amb tots els coeficients positius. Cal notar que normalent s'escriu V^T en comptes de V a l'hora de fer la descomposició. La descomposició pot ésser feta de tal manera que els coeficients de la diagonal de D estiguin ordenats de forma descendent, i a partir d'ara assumirem que és així, ja que només ens interessa l'element de D que sigui més proper a zero.

El SVD també es pot aplicar matrius que no siguin quadrades, normalment s'utilitza en matrius que tenen més files que columnes. Llavors, anomenarem A a una matriu $m \times n$ que compleixi que $m \geq n$. En aquest cas, A queda factoritzada en $A = UDV^T$ on U és una matriu $m \times n$ amb columnes ortogonals, D és una matriu diagonal $n \times n$ i V és una matriu ortogonal de $n \times n$. El fet que U tingui columnes ortogonals significa que $U^T U = I_{n \times n}$.

També es poden definir descomposicions en valors singulars per a matrius que tinguin més columnes que files. No obstant, no ho tractarem ja que rarament s'utilitza i no apareix cap situació que ho requereixi en el present projecte. A més, en aquests casos, és factible estendre la matriu tot afegint files amb zeros per obtenir una matriu quadrada i aplicar el SVD sobre la matriu resultant.

B.2.2 Valors singulars i valors propis

Molts cops es confon el concepte de valors singulars i el de valors propis. Els coeficients de la matriu D del SVD són positius i són coneguts com els valors singulars de la matriu A. I naturalment són diferents dels valors propis. Per veure la connexió existent entre els valors singulars i els propis partirem de la descomposició $A = UDV^T$. Llavors, podem escriure $A^TA = VDU^TUDV^T = VD^2V^T$. Com que V és ortogonal, tenim que $V^T = V^{-1}$, i per tant si substituïm obtenim $A^TA = VD^2V^{-1}$. Aquesta és la definició de l'equació dels valors propis, ja que els coeficients de D^2 són els valors propis de A^TA i les columnes de V són els vectors propis de A^TA. Per tant, els valors singulars de A són l'arrel quadrada dels valors propis de A^TA.

B.2.3 Complexitat computacional del SVD

La complexitat computacional del SVD depèn en quanta informació necessita ser retornada. La solució del SVD és l'última columna de la matriu V, com veurem més endavant en B.1. La matriu U no s'utilitza i normalment no cal ser calculada. Per sistemes d'equacions amb moltes més files que columnes, el càlcul d'U pot reduir el temps de càlcul de forma significativa. Encara que l'algorisme proposat calcula completament la descomposició per SVD per a simplificar la descripció, hi ha formes per resoldre el sistema d'equacions sense calcular U.

La complexitat de calcular D i V és lineal respecte el nombre de files. Si m és el nombre de files, calcular D i V és de complexitat O(m), i per tant lineal. Mentre que la complexitat de calcular U és m^2, i per tant tindríem O(m^2), d'aquí que a la pràctica sigui important intentar evitar calcular U.

B.2.4 Solucions d'equacions lineals genèriques per mínims quadrats

La descomposició en valors singulars (SVD) s'utilitza sobretot per trobar la solució de sistemes d'equacions. Si considerem un sistema d'equacions de la forma $A\mathrm{x} = b$, sent A una matriu de $m \times n$, llavors tenim tres casos:

- Si $m < n$ llavors hi ha més incògnites que equacions. En aquest cas no hi ha una única solució, sinó un vector de solucions.
- Si $m = n$ llavors hi ha el mateix nombre d'equacions que d'incògnites i, per tant, hi ha només una única solució.
- Si $m > n$ llavors hi ha més equacions que incògnites. En general el sistema no tindrà solució excepte que b estigui inclòs en les columnes de A.

Nosaltres ens centrarem en el cas que $m \geq n$, i el rang de A sigui n. Si la solució no existeix, en la majoria de casos és possible trobar un vector x que és proper a ser solució del sistema $A\mathrm{x} = b$. En altres paraules, cercarem el vector x que minimitzi $\|A\mathrm{x} - b\|$. Llavors, denominarem x com la solució per mínims quadrats del sistema. La solució per mínims quadrats es troba utilitzant el SVD.

Volem trobar x que minimitza $\|A\mathrm{x} - b\| = \|UDV^T\mathrm{x} - b\|$. A causa de la propietat de la preservació de la norma en les transformacions ortogonals, $\|UDV^T\mathrm{x} - b\| = \|DV^T\mathrm{x} - U^Tb\|$, sent aquesta la quantitat que volem minimitzar. Anomenant $\mathrm{y} = V^T\mathrm{x}$ i $b' = U^Tb$, llavors podem reescriure el problema,

sent el que volem minimitzar $\|D\mathrm{y} - b'\|$ on D és una matriu $m \times n$. Podem representar les equacions en forma matricial:

$$\left[\begin{array}{cccc} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \\ \hline & & & \\ & & 0 & \\ & & & \end{array}\right] \left(\begin{array}{c} y_1 \\ y_2 \\ \vdots \\ y_n \end{array}\right) = \left(\begin{array}{c} b'_1 \\ b'_2 \\ \vdots \\ b'_n \\ \hline b_{n+1} \\ \vdots \\ b_m \end{array}\right)$$

Podem observar que el valor més proper de Dy que s'acosta a b' és el vector $(b'_1, b'_2, \cdots, b'_n, 0, \cdots, 0)$, i que aquest s'obté especificant $y_i = b'_i/d_i$ per $i = 1, \cdots, n$. El fet d'assumir que el rang d'$A = n$ comporta que $d_i \neq 0$. Finalment, trobem $\mathrm{x} = V\mathrm{y}$. L'algorisme complet el trobem en B.1.

Algorisme B.1 Algorisme de resolució SVD per equacions lineals genèriques.

i Trobar el SVD $A = UDV^T$.

ii Calcular $b' = U^T b$.

iii Trobar el vector y definit com $y_i = b'_i/d_i$, on d_i és la diagonal i-enèsima de D.

iv La solució és $\mathrm{x} = V\mathrm{y}$.

B.2.5 Solucions d'equacions lineals homogènies per mínims quadrats

La solució d'un sistema d'equacions lineals homogènies, de la forma $A\mathrm{x} = 0$ és semblant a la proposada anteriorment. Aquest cas es sol trobar en problemes de reconstrucció, i ens apareix en el capítol 5. Considerarem que tenim més equacions que incògnites. La solució $\mathrm{x} = 0$ la obviem, ja que busquem una solució diferent a zero per el nostre sistema d'equacions. Cal fer notar que si x és solució del sistema d'equacions, també ho serà kx per qualsevol valor escalar de k. Una restricció que podem aplicar és que es compleixi $\|\mathrm{x}\| = 1$.

En general, aquest conjunt d'equacions no convergirà a una solució exacta. Si A té dimensió $m \times n$ llavors hi ha una solució exacta si només si el rang$(A) < n$. Per tant, el rang de la matriu A ha d'ésser més petit que les columnes. En absència de la solució exacta, cercaquerem la solució utilitzant mínims quadrats, de forma anàloga al cas anterior.

Per tant, el problema esdevé en trobar x que minimitza $\|A\mathrm{x}\|$ amb la restricció que $\|\mathrm{x}\| = 1$. Com que $A = UDV^T$, llavors obtenim que hem de minimitzar $\|UDV^T\mathrm{x}\|$. Un altre cop, per les propietats de les transformacions ortogonals, tenim que $\|UDV^T\mathrm{x}\| = \|DV^T\mathrm{x}\|$ (la matriu U desapareix per què $b = 0$) i $\|\mathrm{x}\| = \|V^T\mathrm{x}\|$. Per tant, resulta que hem de minimitzar $\|DV^T\mathrm{x}\|$ amb la restricció $\|V^T\mathrm{x}\| = 1$. Podem escriure $\mathrm{y} = V^T\mathrm{x}$, i el problema es resumeix en

minimitzar $\|D\mathrm{y}\|$ amb la restricció $\|\mathrm{y}\| = 1$. Ara, D és la matriu diagonal amb els coeficients de la diagonal ordenats de forma decreixent. D'aquí, la solució del problema és $\mathrm{y} = (0, 0, \cdots, 0, 1)^T$, tenint un coeficient diferent de zero en l'última posició. Per tant, $\mathrm{x} = V\mathrm{y}$ és resumeix simplement en agafar l'última columna de V, com es mostra en l'algorisme B.2.

Algorisme B.2 Algorisme de resolució SVD per equacions lineals homogènies.

i Trobar el SVD $A = UDV^T$.

ii La solució x és la darrera columna de V (suposant que D està ordenada de forma decreixent).

B.3 Matriu Jacobiana

La matriu Jacobiana s'utilitza molt en el càlcul relacionat amb vectors. Aquesta matriu està formada per totes les derivades parcials de primer ordre d'una funció avaluada com a vector. La seva importància radica en que representa la millor aproximació lineal a una funció diferenciable prop a un punt donat. En aquest cas, la matriu Jacobiana es converteix en una derivada de funció amb múltiples variables.

Suposem que tenim una funció $F : R^n \to R^m$ que relaciona un espai Euclidià de dimensió n amb una altra Euclidià de dimensió m. Aquesta funció dóna m components reals de la forma $y_1(x_1, \cdots, x_n), \cdots, y_m(x_1, \cdots, x_n)$. Les derivades parcials d'aquestes funcions (si existeixen) es poden expressar en una matriu de $m \times n$, el que anomenarem la matriu Jacobiana:

$$J = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

Aquesta matriu també es pot expressar com $J_F(x_1, \cdots, x_n)$ o per $\frac{\partial(y_1, \cdots, y_m)}{\partial(x_1, \cdots, x_n)}$. La fila i de la matriu s'obté transposant el pendent de la funció y_i per $i = 1, \cdots, m$. Si p és un punt en $\Re^n$ i F és diferenciable en p, llavors la seva derivada vindria donada per $J_F(p)$, sent aquest el millor mètode per calcular la derivada. En aquest cas, la relació lineal descrita per $J_F(p)$ és la millor aproximació de F vora el punt p, i per tant podem deduir que:

$$F(x) \approx F(p) + J_F(p) \cdot (x - p)$$
per qualsevol x propera a p.

B.4 Productes vectorials

Les matrius antisimètriques tenen propietats molt útils, particularment les matrius antisimètriques de 3×3. Si tenim un vector $\mathrm{a} = (a_1, a_2, a_3)^T$, podem definir la següent matriu antisimètrica:

$$[\mathrm{a}]_x = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{B.1}$$

Tota matriu antisimètrica es pot escriure de forma abreviada com $[\mathrm{a}]_x$ per a un vector a.

El producte vectorial de dos vectors $\mathrm{a} \times \mathrm{b}$ (també es pot escriure com $\mathrm{a} \wedge \mathrm{b}$) és el vector $(a_2b_3 - a_3b_2, a_3b_1 - a_1b_3, a_1b_2 - a_2b_1)^T$. El producte vectorial està relacionat amb les matrius antisimètriques complint el següent:

$$\mathrm{a} \times \mathrm{b} = [\mathrm{a}]_x \, \mathrm{b} = (\mathrm{a}^T \, [\mathrm{b}]_x)^T \tag{B.2}$$

Bibliografia

[Armangué 03] X. Armangué. "Modelling stereoscopic vision systems for robotic applications". Tesi, 2003.

[Faugeras 1993] O. Faugeras. "Three-Dimensional Computer Vision". MIT Press, 1993.

[Fischler i Bolles 1981] M. A. Fischler i R. C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". Comm. of the ACM 24: 381-395.

[Hall 1982] E. L. Hall, J. B. K. Tio, C. A. McPherson i F. A. Sadjadi. "Measuring Curved Surfaces for Robot Vision". Computer Journal, vol. 15, no. 12, planes 42-54, decembre 1982.

[Harris 1988] C. Harris, M. Stephens. "A Combined Corner and Edge Detector". Proc. 4th Alvey Vision Conferences, planes 147-151, 1988.

[Hartley i Zisserman, 2000] R. Hartley i A. Zisserman. "Multiple View Geometry in Computer Vision". Cambridge, 2000.

[Julesz 1971] B. Julesz. "Foundations of Cyclopean Perception". Chicago: The University of Chicago Press, 1971

[Moravec 1977] H.P. Moravec. "Towards automatic visual obstacle avoidance". Fifth International Joint Conference On Artificial Intelligent, p. 584.

[Paragios 2005] N. Paragios, Y. Chen i O. Faugeras. "Handbook of mathematical models in computer vision". Springer, 2005.

[Salvi 1998] J. Salvi, J. Batlle i E. M. Mouaddib. "A Robust-Coded Pattern Projection for Dynamic 3D Scene Measurement". Pattern Recognition Letters, vol. 19, no. 11, pages 1055-1065, Setembre 1998.

[Sampson 1982] P.D. Sampson. "Fitting conic sections to very scattered data: an iterative refinement of the Bookstein algorithm". Computer Graphics and Image Processing, (18):97–108, 1982.

[Slama 1980] C. C. Slama, C. Theurer i S. W. Henriksen. "Manual of photogrammetry". American Society of Photogrammetry, Falls Church, VA, 4**?** edició, 1980.

[Schmid et al. 2000] Cordelia Schmid, Roger Mohr i Christian Bauckhage. "Evaluation of Interest Point Detectors". International Journal of Computer Vision volum 37, núm. 2 planes 151-172, juny 2000.

[Trucco 1998] E. Trucco i A. Veri. "Introductory Techniques for 3-D Computer Vision". Prentice-Hall 1998.

[Tsai 1997] R. Y. Tsai. "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses". IEEE International Journal on Robotics and Automation, vol. RA-3, núm. 4, planes 323-344, agost 1987.

[Weng 1992] J. Weng, P. Cohen i M. Herniou. "Camera Calibration with Distortion Models and Accuracy Evaluation". IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, núm. 10, planes 965-980, octubre 1992.