Universidad del País Vasco
Euskal Herriko Unibertsitatea

Camera Perspective Distortion in Model-based Visual Localisation

A dissertation submitted for the degree of Doctor in Computer Science

Nagore Barrena Orueechebarria

Advisors
Alejandro García-Alonso
Jairo R. Sánchez

Donostia-San Sebastián, December 2018

Aitite eta Amamari

Summary

This thesis starts with a proposal for a collaborative global visual localisation system. Then, it centres on a specific visual localisation problem: perspective distortion in template matching.

The thesis enriches 3D point cloud models with a surface normal vector associated with each 3D point. These normals are computed using a minimisation algorithm.

Based in this new model, the thesis proposes an algorithm to increase the accuracy of visual localisation. The algorithm improves for template matching processes using surface normals.

The hypothesis, '*Given a 3D point cloud, surface orientation of the 3D points in a template matching process increases the number of inliers points found by the localisation system, that is, perspective compensation.*' is objectively proved using a ground truth model.

The ground truth is achieved through the design of a framework which using computer vision and computer graphics techniques carries out experiments with out the noise of a real system, and prove in an objective way the hypothesis.

Keywords: model-based visual localisation, template matching, perspective distortion, surface normal.

Laburpena

Hemen aurkezten den tesia, ikusmen artifizial bidezko lokalizazio sistema global eta kolaboratiboaren proposamenarekin hasten da. Ondoren, ikusmen artifizial bidezko lokalizazio sistemetan aurkitu daitekeen arazo espezifiko batean jartzen du arreta: perspektiba distortsioa *template matching* izeneko tekniketan.

Tesiak, 3D puntu-hodeiak informazio gehigarriarekin aberasten ditu: puntu bakoitzari dagokion orientazio informazioa (gainazaleko bektore normala) gehitzen zaio. Minimizazio algoritmo bat diseinatu da bektore horiek estimatzeko.

Eredu berri honetan oinarriturik, tesiak algoritmo berri bat proposatzen du. Algoritmo horrek ikusmen artifizial bidezko lokalizazio sistemen doitasuna haunditzen du, *template matching*-eko prozesuak hobetzen dituelarik, aipatutako gainazaleko normal bektoreak erabiliz.

Tesi honek ondorengo hipotesia modu objektiboan balioztatu du: '*3D puntu-hodeia emanik eta puntu bakoitzari dagokion gainazaleko normal bektorea erabiliz template matching prozesuetan, ikusmen artifizial bidezko sistemak topatzen dituen puntu tipikoen kopurua (ingelesez inlier) aregaotzen da (perspektiba konpentsazio sistema)*'. Horretarako eredu birtual bat diseinatu da. Eredu birtual horri esker, aldagai guztien balioak ziurtasunez ezagutu daitezke.

Bestalde, eredu birtuala lortzeko, software tresna multzo bat diseinatu eta garatu da. Software tresna multzo horrek, ikusmen artifizialeko zein konputagailu bidezko grafiko teknikak erabiltzen ditu esperimentuak martxan jartzeko. Esperimentu horiek mundu errealean egiten diren esperimentuetan sortzen den zarata deusestazten dute. Gainera, hipotesia era objektiboan frogatzea ahalbidetzen du.

Hitz gakoak: ereduetan oinarritutako ikusmen artifizial bidezko lokalizazio sistema, *template matching*, perspektiba distortsioa, gainazaleko bektore normala.

Resumen

La presente tesis comienza con la propuesta de un nuevo sistema global y colaborativo de localización mediante visión. Posteriormente, se centra en un problema específico que se encuentra en los sistemas de localización mediante visión: la distorsión de perspectiva en los sistemas denominados *template matching*.

La tesis enriquece las nubes de puntos 3D añadiendoles información sobre la orientación de dichos puntos (vector normal de superficie). Un algoritmo de minimización es diseñado para la estimación de dichos vectores.

Basado en este nuevo modelo, la tesis propone un nuevo algoritmo para incrementar la precisión de los sistemas de localización mediante visión. Dicho algoritmo mejora los procesos de *template matching* haciendo uso de las mencionadas normales de superficie.

La hipótesis, '*Dada una nube de puntos 3D, haciendo uso de la orientación superficial de los puntos que la componen en procesos de template matching incrementa el número de inliers encontrados por un sistema de localización por visión (sistema compensación de perspectiva)*' es objectivamente provada haciendo uso de un modelo *ground truth*.

Por otro lado, el modelo *ground truth* es estimado y obtenido gracias al diseño realizado de un *framework* que haciendo uso de técnicas de visión por computador y gráficos por computador permite ejecutar los experimentos libre del rudio que se encuentra en los modelos reales. Además, de este modo la hipótesis es demostrada de manera objetiva.

Palabras clave: sistema de localización por visión basado en modelos, *template matching*, distorsión de perspectiva, normal de superficie.

Acknowledgment

It's hard to stay mad when there's so much beauty in the world. Sometimes I feel like I'm seeing it all at once, and it's too much; my heart fills up like a balloon that's about to burst. And then I remember to relax and stop trying to hold on to it, and then it flows through me like rain and I can't feel anything but gratitude for every single moment of my stupid little life.

Lester Burnham, American Beauty (1999)

Tesia bukatzerakoan hartzen duzu arnasa. Orduan ohartzen zara egun honetara ekarri zaituen pausu ororen garrantziaz. Pausu horiek osatzen duten bide luzea begirada lasai batean jorratzen duzu. Eta orduan, Lester Burnham bezala, konturatzen zara bizitza edertasunez beteta dagoela: alboan eduki dituzun guzti horietaz, laguntza eskaini dizun pertsona orotaz, hau egitera bultzatu zaituztenak, bizitza alaiagoa egiten dizutenak, eta abar. Horiei guztiei, eskerrak eman nahi dizkiet nire bizitza ederragoa egiteagatik.

En primer lugar quisiera mencionar que le estaré eternamente agradecida a Álex García-Alonso. Por ser el mejor director de tesis que hubiese podido tener. Gracias, por todas las horas dedicadas a la confección y corrección de los artículos científicos, por guiarme, por las horas dedicadas a esta disertación. Pero gracias también, por cuidar de mí, por pensar siempre en la mejor opción, por estar siempre dispuesto a ayudarme para que el desarrollo de esta tesis afectase lo mínimo posible a mi vida personal.

Mención especial se merece Jairo Sánchez, mi mentor. Agradezco enormemente las horas que dedica a responder mis siempre presentes dudas, además del conocimiento que me transmite día a día. Todo lo aprendido de esta nuestra profesión, lo he aprendido de ti. Me gustaría no solo agradecerte las horas que me has dedicado para que esta tesis siga adelante, sino por ser también un fiel compañero.

No podría olvidarme tampoco de todos mis compañeros de trabajo, sin los que esta tesis no podría haber nacido. En primer lugar a David Oyarzun; este proceso empezó gracias a ti. Gracias por la confianza depositada en mí,

y por haberme empujado una y otra vez a realizar la tesis. Tu constante impulso para que realizara la tesis y trabajara en ello ha dado finalmente sus frutos.

Mil gracias, al que siempre será el departamento más molón de Vicomtech, a pesar de su desaparición: gracias a V5 (Forever V5!). Gracias por alegrarme cada mañana, por las mil y unas risas, gracias por haber tenido que trabajar de más alguna que otra vez para que yo me pudiese dedicar a esta tesis, gracias por vuestra ayuda, por vuestros consejos. Gracias por ser los mejores compañeros que una pueda tener. Gracias Sara por ser siempre amable y estar dispuesta a ayudarme cada vez te pido alguna imagen, modelo, consejo con los diseños, etc. Gracias Víctor, Javi, Ramón, Rubén y Hugo por la ayuda en los desarrollos y en las dudas (Hugo, ¡siempre tienes la respuesta a todas mis dudas!). Helen, Sara, Andoni, Andrés y Aitor por vuestra siempre disponible ayuda y alegría.

No quisiera olvidarme del departamento V0. Gracias por darme refugio en vuestro departamento y por estar siempre disponibles para ayudar.

Por supuesto, gracias a mis tiburones. Gracias a vosotros una va al trabajo con una sonrisa en la boca. Siempre deseando que llegue el descanso y la hora de la comida, para hablar y reír junto a todos vosotros. Pero, gracias sobre todo por haber llevado esta amistad más allá de lo laboral. Gracias, por los momentos vividos (y por los que nos sigan quedando por vivir): por las despedidas, por los vídeos, por las barbacoas, por las tarde-noche de cine los miércoles en el Trueba, por las noches de Tamborrada, por los mil y un bailoteos, por las colas en el Victoria Eugenia a las tantas de la madrugada conspirando para poder entrar gratis (¡qué mal conspiramos de madrugada! :-)), y por esta lista que podría continuar sin fin.

Bestalde, aipatu bezala, lan hau atal pertsonalarekin ere oso lotuta dago. Honaino iritsi ahal izatea ere, beraiei esker baita.

A mis amigas. Por todos esos momentos vividos desde que somos unas crías: risas, lloros, farras, vernos crecer, madurar, formar familias, por ser siempre parte del elenco principal de esta película que es nuestras vidas.

Nire familia *´politikoari*'. Harkaitz, Olatz, Edorta, Sheila eta Xabier, familian lekutxoa egiteagatik, beti laguntzeko prest egoteagatik. Nire ilobei: Jare, Inhar, Lia, Ur, Peru eta Intzari, segi horrela, maitagarri.

Mikel, Eba, Maddi eta Juneri, zazpiren artean sortzen dugun '*familia txiki*' honegatik, aholkuengatik, Olakorreko gosariengatik, :-) eta elkarrekin bizitzeko geratzen diren momentu guztiengatik.

Azkeneko hitzak, nire bizitzako pertsona garrantzitsuenei eskaini nahi dizkiot.

Nire familia guztiari. Aitite eta amamei, azken eguneraino zuen maita-

sun beroena eskaintzegatik. Zuen onena emategatik.

A izeko Belén, por haber estado sin descanso presente en mi vida. Por estar siempre disponible, por tu cariño, por todo lo que me has ofrecido. Por ser IZEKO, con mayúsculas.

Xabier eta Anderri, momentu ederrak eskaintzegatik: *¡duerme pueblo!*, *¡Tu alma!*, *¡Tu alma que sale de tu cuerpo!*, *Spyro the dragon*, *Billy Elliot*, eta momentu paregabe gehiago batera bizitzeagatik. Lehengusu paregabeak zarete.

Nire gurasoei. Haiei eskertzeko ez dago nahikoa hitz. Dena zor dizuet. Zuek gabe ez noa inora. Hau dena zuen lana, heziketa, laguntza, babesa eta leialtasunagatik izan da posible. Hau zuen lorpena ere bada. Asko maite zaituztet!

Nire Alaia maiteari. Oraindik txikiegia zara momentu hau gogoan izateko. Zure irribarrea eta laztanak dena errazago eramaten laguntzen dute. Lan apal hau zuretzat adibide izan daitekeela pentsatzeak, aurrera jarraitzeko indarrak eman dizkit. Zure bizitza zoriontasunez betetzeko asmotan, lan egingo dugu aita eta biok etengabe.

Eta bukatzeko, eskerrik beroenak zuretzat, Gorka. Egunero nire alboan egoteagatik, bizitzako arlo guztietan zure laguntza leiala eskaintzeagatik, amets denak egi bihurtu daitezen bidean bikotekide paregabea izateagatik, momentu txarretan ere nire alboan jarraitu izanagatik, hainbeste momentu alai eskaintzeagatik, zure maitasuna eskaintzeagatik, paregabeko aita izategatik. Maite zaitut Gorka!

Eskerrik asko,
Nagore Barrena
December 2018

Contents

Glossary

AR Augmented reality

BA Bundle Adjustment algorithm

BVLS Benchmark Visual Localisation System

DLT Direct Lineal Transformation

DoG Differences of Gaussian

EKF Extended Kalman Filter

FAST Features from accelerated segment test

FastSLAM A Factored Solution to SLAM

FkF First *keyFrame*

FREAK Fast Retina Keypoint

FLANN Fast Approximate Nearest Neighbour With Automatic Algorithm Configuration library

GPS Global Positioning System

Harris Harris Corner detector

KLT Kanade Lucas Tomasi algorithm

Kd-Tree K-dimensional Tree

LM Levenberg-Marquart minimisation algorithm

MonoSLAM Real-time Single Camera SLAM

NCC Normalised cross-correlation

NkF Nearest *keyFrame*

PCS Perspective Compensation system

PTAM Parallel Tracking and Mapping

SIFT Scale-Invariant Feature Transform

SSD Sum of Squared Differences

SfM Structure from Motion

SVD Singular Value Decomposition

vSLAM visual Simultaneous Localisation and Mapping

VR Virtual reality

WaPT Warped Template Patch Tracking algorithm

Nomenclature

Vectors are represented with a hat $\vec{a}$, matrices in boldface capital letter $\mathbf{A}$, scalars in italic face a and sets in uppercase `S`.

$\vec{X}$:	A vector representing a point in real world
X:	A point in real world
$\vec{x}$:	A vector representing a point in image plane
x:	A point in image plane
$\vec{C}$:	A camera center or optical center
I:	An image plane
$(X, Y, Z)^T$:	3D point
$(x, y)^T$:	Image point
$[\mathbf{R}\|\vec{t}]$:	Extrinsic matrix or camera motion matrix
f:	Focal distance
`x`:	A set of 2D points
`Im`:	A set of images
`PC`:	A set of 3D points, 3D point cloud
`PCn`:	A set of surface normal vectors
`KF`:	A set of *keyFrames*
$\mathbf{H}$:	Homography matrix
$\mathbf{E}$:	Essential matrix
$\mathbf{F}$:	Fundamental matrix
$\mathbf{P}$:	A camera pose
$\mathbf{P}^+$:	A Moore-Penrose pseudo-inverse projection matrix
p:	Candidate point or pixel
th:	A threshold
e:	An epipole point
H_0:	Null hypothesis
H_α:	Alternative hypothesis
α:	Significance level
p_v:	p-value

List of Figures

List of Tables

Part I

Introduction

Chapter 1

Introduction

Augmented reality has the potential to change the way humans interact with their daily environments. It enriches the near surroundings with additional information, being a powerful tool in different fields such as education, tourism, industry, marketing or health among others. Good examples are: Vávra et.al [64], present some augmented reality applications used to assist in neurosurgery procedures, overlaying needed information into the patient. In this manner, they present some works which use augmented reality in order to 'precise localisation of individual blood vessels, important neuronal tracts or applications to plan the operation corridor in a removal of superficial tumours, or in neurovascular surgery' [64]. On the other hand, some works ([42],[4], [58] and [63]) present augmented reality applications for remote maintenance or assistance in industry. There can also be found a lot of augmented reality applications oriented to educational purposes [7] [9] and tourism or cultural heritage[8].

Localisation in real-time is an essential task in augmented reality applications. Its use is mandatory to solve above mentioned problems. Section 1.1 exposes the context and motivation for this dissertation.

The localisation problem can be approached in different ways. Those which use computer vision techniques are analysed in this thesis. They are denominated visual localisation systems. More concretely, this thesis research delves in perspective distortions generated by the camera motion and its impact on the quality of the visual localisation algorithms. These distortions can generate negative consequences in the visual localisation task. In that way, Section 1.2 defines the hypothesis to be proved. The steps, which have to be done in order to prove the hypothesis, are also enumerated in that section.

Section 1.3 exposes the research methodology followed by this dissertation. In Section 1.4 the scientific and technological contributions achieved, as a results of the thesis research, are enumerated. Finally, Section 1.5 describes how the dissertation is structured.

1.1 Context and Motivation

Augmented reality (AR from now on) is a concept that can enhance the environment surrounding the user. It is considered an evolution of virtual reality (VR from now on). VR immerses the user in a completely artificial world. While the user is immersed in this virtual world, the user does not have any notion about the real environment.

However, AR technology allows the user to see the real world while virtual images are overlapped; real and virtual coexist in the same space. The work described in [37] expresses this concept using *Reality-Virtuality Continuum* (Figure 1.1), a scale ranging between the completely virtual and completely real. The area between the two extremes, where both the real and the virtual are mixed, is the so-called mixed reality. AR is the area where the virtual augments the real, and augmented virtuality is the area, where the real augments the virtual.

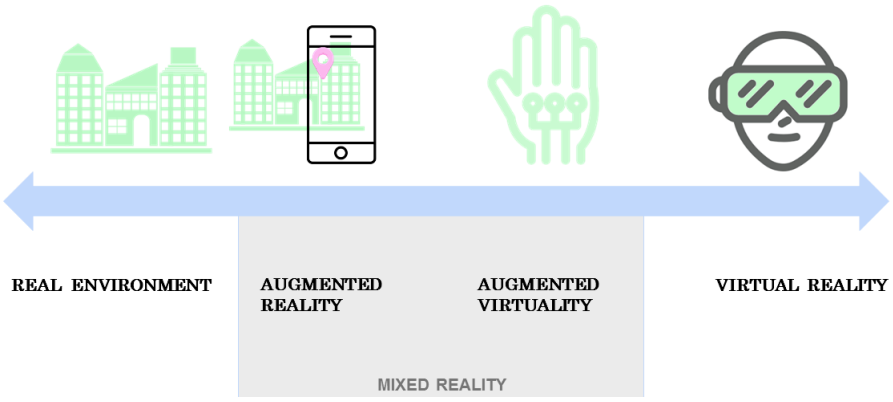


Figure 1.1: The *Reality-Virtuality Continuum* concept. Diagram adapted from [37]

According to [5] an AR application is described by the following features: 'it combines real and virtual elements, it is interactive in real-time and it must register in 3D'. So, in AR applications, the virtual elements are 3D/2D

models that should be put in a particular place of the environment in real-time.

The user of an AR system, will see an image in which the real world and one or more 3D/2D elements are projected. An example is shown in Figure 1.2. An image of a real site is shown. On the right part of the image a virtual sculpture is also seen as an element of the reality. The goal is to place virtual elements at specific position in the world in real-time. To achieve this target, the location of an object/place must be identified continuously, as when the object or the camera moves. This process is named localisation in real-time.

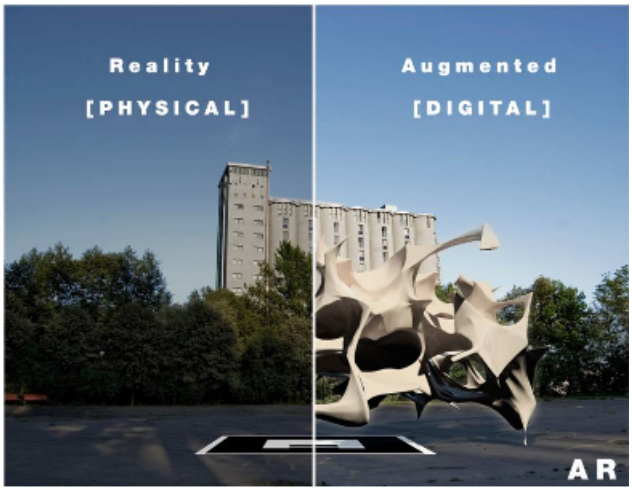


Figure 1.2: Example of an augmented reality application

Given those features, the following components are needed to form an AR system: (i) a camera that captures the real world, (ii) a computer to estimate the localisation in real-time and perform the rendering and (iii) a display where the image with real world and virtual elements is rendered.

In that way, AR for mobile devices became very popular thanks to the emergence of the new generations of smartphones. These first AR applications showed geolocated information overlaid with the images that the camera captures. In general, the information shown consisted of 2D tags taken from public databases. Information was overlaid using the position and orientation information given by the GPS, compass and accelerometers of the device. However, the accuracy of the readings of these sensors is not enough to insert 3D content properly aligned with images.

Nevertheless, computer vision techniques can be used with localisation

purposes (visual localisation). But, these techniques are expensive in computational terms. As mobiles and tablets have evolved, acquiring high computational power, the use of visual localisation systems to AR proposes have settled.

In that way, having some 3D knowledge of the environment allows applying visual localisation algorithms that can accurately estimate the position and the orientation (pose) of the camera. With this regard, those which are denominated model-based visual localisation systems have shown to be very precise localisation techniques. These techniques build (offline and online) a map and the camera localise itself in that map. The main problem of these maps is that they need to be extended and continuously updated. Although this problem has been widely studied in small and controlled scenarios, the process of maintaining and extending a map in large outdoor environments is still very complex. Section 1.1.1 address this problem, then Section 1.1.2 places the specific research of this thesis within this global framework.

1.1.1 Global visual localisation

The initial work for this thesis was discussed in our 2013 poster at Eurographics conference [10]. The proposal was a new visual localisation concept, which is named *collaborative model-based visual localisation system*, and in which users act as data suppliers. That approach proposes a local model-based visual localisation system that uses a low precision but efficient map to perform the localisation in real-time. In parallel this map is used to extend the large-scale one in the Cloud by means of accurate 3D reconstruction techniques.

More concretely, based on the results obtained in [59] the system proposed in [10] is divided into two main parts. On the one hand, model-based visual localisation system is applied locally to small environments.

On the other hand, the map obtained locally is used to extend and update the map in the Cloud using dense 3D reconstruction techniques. This accurate map is recursively used to update the local maps. Figure 1.3 shows the overview of proposed system.

This method opened new research lines in the field of collaborative model-based visual localisation systems where many users can contribute simultaneously in the building process of the huge map. These research lines could be focused on tasks such as, building process of the huge map from different users, selection of the correct local map to improve local localisation, be robust against illumination, scale, or rotations changes among others.

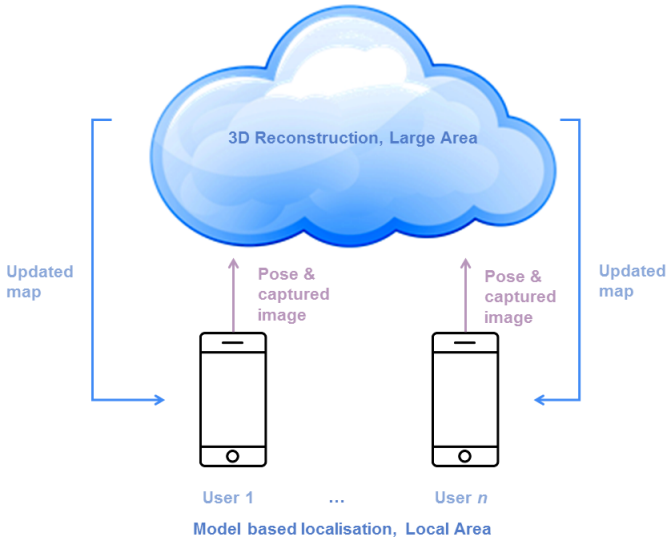


Figure 1.3: Collaborative model-based visual localisation system proposed by [10]

1.1.2 Patch Fitting

The thesis presented here is focused on one of the challenges described above. More concretely it is centred on measuring the similarity between images during the motion in order to improve that model-based visual localisation. The components for that kind of model-based visual localisation systems are the following one:

- The 3D reconstruction of the environment and the input images.
- Projection of each 3D point into the input image.
- Adjustment of the projection, i.e Patch Fitting: comparing the surroundings of each point in the image with the surroundings of the same point in a reference image.
- Estimate the pose of the camera.

Exactly, the thesis presented here is focused on the improvement of the third point.

1.2 Hypothesis and Aims

A lot of applications like AR or autonomous robot navigation apply visual localisation methods in order to localise themselves in the surrounding environment. Real-time localisation task is also named tracking.

The system proposed in the initial work of this thesis (Section 1.1), it is oriented to the use of a 3D reconstruction of the environment (from now on 3D point cloud) as a reference model or map where the user must localise itself.

These kinds of visual localisation systems rely on matching 3D points, previously identified in an environment (3D point cloud), with their corresponding 2D projections measured in each image (or frame) captured by the camera. The inliers are the subset of 3D–2D correspondences that are correctly matched. So, for each frame the system obtains a number of inliers which are used later on to update the global position of the camera.

Hence, localisation methods have to establish a relationship between an internal representation of the real scene and the images captured by the camera. The internal representation can be composed by different information levels, being 3D points the simplest representation.

There are different ways to establish mentioned relationship in in the state of the art, as it will be observed later in this thesis.

Nonetheless, this thesis is focused on that which use template matching methods. These kinds of methods, use the appearance similarity between images in order to establish mentioned relationship:

For each image captured by the camera and each 3D point, a patch around its projection is compared with the appearance that the same point had when it was reconstructed.

However, as the camera gets further the similarity of the image patches decreases. Consequently, the system can fail because of perspective distortions [38]. When the orientation disparity between the patches is high, the matching process tends to fail.

The initial hypothesis of the research of this thesis is that, considering a surface orientation (surface normal vector or normal vector) associated with each 3D point, perspective distortion can be reduced while performing the template matching process. The expectation is that this improvement will contribute to increase the number of good matches. Consequently, the localisation process will become a lightweight and stable process.

So, this work aims at proving the following hypothesis:

Hypothesis *Given a 3D point cloud, using surface orientation of the 3D points in a template matching process increases the number of inlier points*

found by the localisation system (perspective compensation).

In order to validate the hypothesis defined, the following steps are proposed to be performed:

1. Design and implementation of a visual localisation system (model-based localisation system) as a baseline. It will be a visual localisation system which uses Structure from Motion technique in order to obtain a map or 3D reconstruction of the environment. On the other hand, the tracking process will use template matching techniques in order to obtain necessary 3D-2D matches and camera pose. It will be named the **Benchmark visual localisation system**.

2. Design an algorithm to palliate perspective distortions. It will be named the **Warped Template Patch Tracking algorithm**. This algorithm should be an improvement of the template matching technique. The algorithm enriches each 3D point with a normal vector that approximates the orientation of the surface where the 3D point is lying. This normal improves the transfer process of patches providing more precise warped patches, because perspective distortion is taken into account. The visual localisation system designed and implemented, which considers the algorithm designed, is named **Perspective Compensation system**.

3. Perform **preliminary tests as concept validation**. They will measure the similarity of the patches or sub-images obtained by both systems. If the results obtained in these tests are hopeful, an objective validation test should be designed.

4. Design a **test to prove the hypothesis** defined, using a statistical inference theory to make it **in an objective way**.

   (a) Design and implement a way to obtain a ground truth.

   (b) Design the experiments for an objective prove of the hypothesis: define null and alternative hypothesis as well as the statistical methods to prove or reject them.

   (c) Perform the experiments.

Figure 1.4 shows the flowchart to be followed in order to validate the hypothesis.

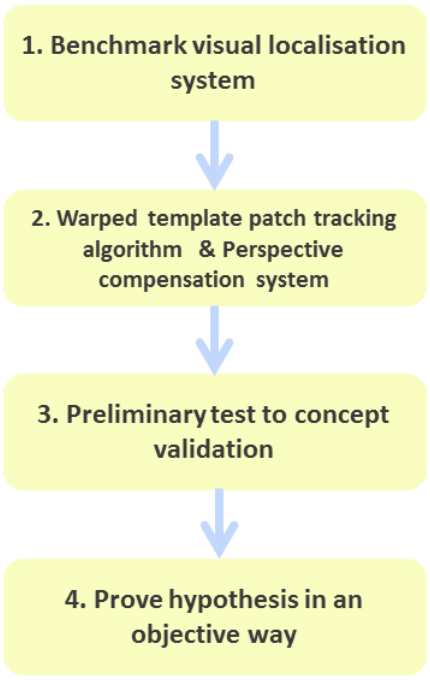


Figure 1.4: The steps which this thesis has to follow in order to validate the hypothesis

1.3 Methodology

This dissertation presents a research work, which has been developed following a specific strategy. The strategy is defined as Figure 1.5 shows.

Deepening in each of the steps:

1. **Ask a question**. All research works start when a question disturbs the scientist or student. What do you want to research? What do you want to check? Define that questions is the first steps of all. The research field must be focused.

2. **Study the background**. Update knowledge by reviewing the state of the art and studying necessary theorems or items.

3. **Define the hypothesis**. The question exposed in the first step, has to be formalised, it has to be defined exactly what has to be proved in the research work. The hypothesis should be the statement that has

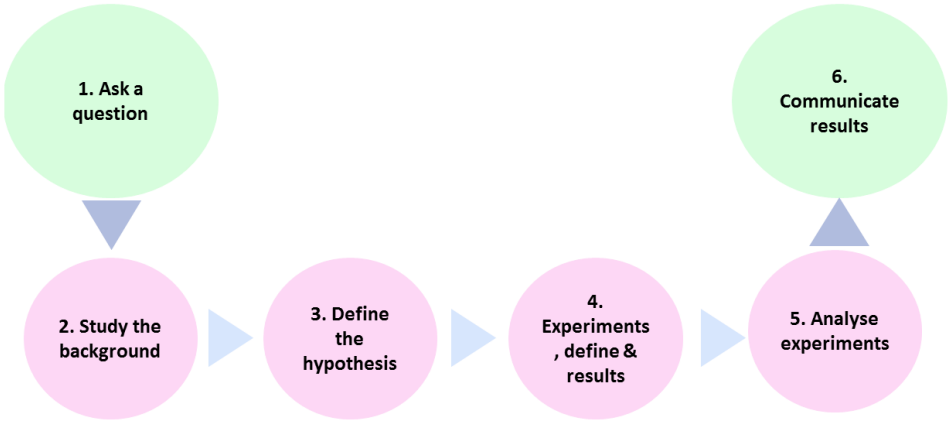


Figure 1.5: The process of a research methodology followed by this thesis

to be proved or rejected according to the results.

4. **Experiments**. In order to prove the hypothesis defined in the previous step, a series of experiments has to be designed. The hypothesis itself will define what should be measured and in what conditions.

5. **Analyse the experiments** The result of the experiments carried out are analysed in order to know if the hypothesis can be verified or rejected. In addition, some of the results can be derived into new experiments in order to discuss more details.

6. **Communicate results**. Finally, all the results must be presented and published.

Focusing on this thesis for each of the steps:

1. **Ask a question**. Section 1.1 explains in what field is located this research and what was the motivation to focused the research on: the use of images similarity during the motion in order to improve model-based visual localisation systems.

2. **Study the background**. Review the state of the art in AR and model-based visual localisation systems. The results of that analysis have been exposed in Chapter 2. In addition, this analysis has been reinforced by attending specialised scientific conferences. Besides, other mathematical and statistics theories have been also studied, in order

to perform the next steps properly. That means the study of statistical inference, divulged in Section 5.1.

3. **Define the hypothesis**. In Section 1.2 can be found the hypothesis formulated for this thesis, as a result of the work done in the first and second step. In addition, Chapter 5 exposes most formal hypothesis formulated for this thesis and the objective way designed in order to prove or reject it, using statistical inference theory.

4. **Experiments**. In order to prove the hypothesis defined in the previous step, model-based visual localisation systems are designed and implemented with that purpose. It involves the design of a new algorithm to be validated. Chapters 3, 4 and 5 are dedicated to describe the experimental set-up designed to be carried out, as well as model-based visual localisation designed and developed algorithms to that purpose.

5. **Analyse the experiments**. The results show conclusions that are discussed and exposed (Chapters 4 and 5). Some of the results derived into new experiments in order to discuss more details. In that thesis the preliminary results obtained in the experiments carried out in Chapter 4 derived into experiments carried out and exposed in Chapter 5.

6. **Communicate results**. Finally, scientific publications in international conferences and journals endorse the research work, Section 6.3. On the other hand, this dissertation it is also part of this communication.

The scheme exposed in Figure 1.6 shows the relationship between chapters of this dissertation and the steps of the explained methodology.

1.4 Contributions

Section 6.1 *briefly* describes the research work and the conclusions of the thesis. Section 6.2 *enumerates* the contributions and Section 6.3 relevant *publications* from this thesis.

The thesis proposes a **global visual localisation paradigm** presented in Section 1.1.1. This is an ambitious challenge whose implementation requires a huge research development. On the other hand, most of the research work described along the thesis and summarised in Chapter 6 is devoted to

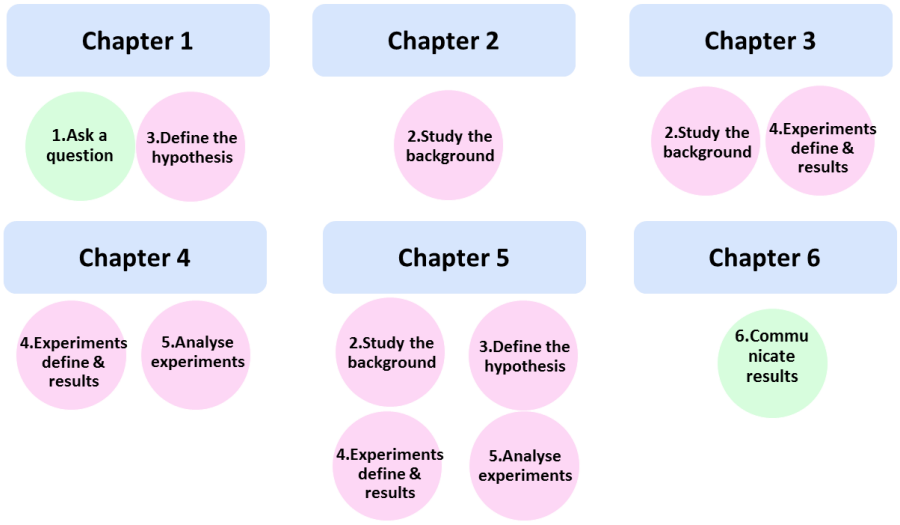


Figure 1.6: The methodology and chapters relationship

a quiet specific problem: solving the **perspective distortion** problem that hinders performance in **template matching** algorithms. Its consequence is a degradation of **visual localisation**.

1.5 Thesis Outline

The thesis is structured in six chapters.

Chapter 1 is the current chapter. It presents the context and motivation of the research, as well as the hypothesis, goals and scope. To achieve those goals and validate the hypothesis, a research methodology is proposed. Finally, the contributions of the dissertation and their location in the document are also shown.

Chapter 2 describes the state of the art relevant to the dissertation. The fundaments of AR, as well as calibration process and model-based visual localisation systems are explained in order to understand better the computer vision techniques used in visual localisation system.

Chapter 3 establishes the basis of the Benchmark Localisation System, which should be designed and implemented in order to be a baseline of the research (it is the first goal to aim in order to prove the hypothesis). In that way, the chapter exposes the theory of Structure from Motion methods and the tracking processes which use template matching. In addition, it describes

the architecture and implementation details of the Benchmark Localisation system.

Chapter 4 describes the algorithm, and the steps which form it, in order to palliate perspective distortions problems. It will be named Warped Template Patch Tracking algorithm (from now on WaPT). The visual localisation system which consider WaPT is also exposed in this chapter. Finally, the preliminary experiments are also exposed and discussed in order to validate the concept.

Chapter 5 explains the last stage of the hypothesis verification. According to the promising results obtained in the Chapter 4 more formal experiments are designed. The goal of these experiments is to prove, in an objective way, that the use of the 3D point orientation, in order to palliate perspective distortions, estimates more accurately the camera pose. In order to verify that it is the use of the orientation, and no other variable, which increase the inliers, the experiments must be carried out in a controlled scenario. So, a ground truth must be generated. On the other hand, in order to prove in an objective way, and verify that the results obtained are not generated by chance, the statistical inference methods are used. Taking into account those specifications, the chapter explains firstly the statistical inference theory in order to know how to set-up the experiments. Then, the experiments set-up is exposed. The generation of the ground truth is also demonstrated. Finally, the results obtained carrying out the experiments are also exposed and discussed.

Finally, Chapter 6 summarises the dissertation, draws the conclusions of this research work and shows the related future lines of work.

Figure 1.7 shows the relationship between the chapters of this dissertation and the steps they are covering. Steps of the methodology followed by this thesis as well as steps to validate the hypothesis.

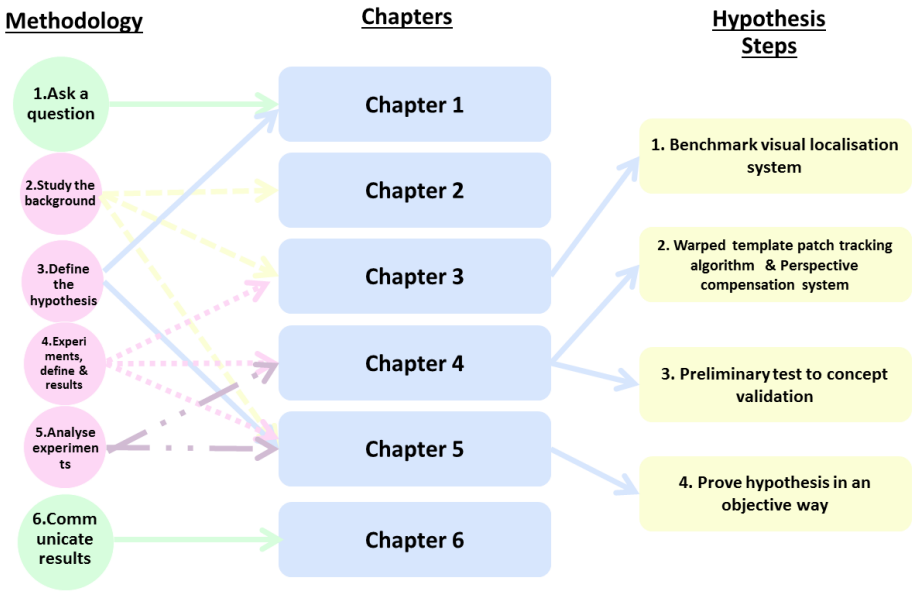


Figure 1.7: The steps covered by chapters

Chapter 2

Related Work

Some AR applications and autonomous navigation systems use visual localisation methods to trace themselves in the environment. The main goal of these kinds of systems is to measure the pose along the motion in real-time. In order to perform properly an AR application, the localisation process is a necessary task as long as the alignment between virtual and real objects depends on that operation.

With that goal, AR applications can use different ways to estimate and get the position in real-time (localisation). According to the work presented in [33], different sensors can be used with this purpose. The authors evaluate some problems in many of those methods. In that way, 'mechanical trackers in spite of being accurate enough, they tether the user to a limited working area. Magnetic trackers are vulnerable to distortions by metal in the environment limiting the range of displacements. Ultrasonic trackers suffer from noise, and they are inaccurate due to the ambient temperature variations. Inertial trackers drift with time' [33].

Nevertheless, the use of a camera to localisation purposes involves a good solution: cameras are cheap and they can be found in the majority of the devices nowadays. For example, mobile phones, tablets, etcetera. Besides, computer vision techniques make possible the accurate and fast estimation of the camera pose. These kinds of systems are denominated visual localisation systems.

The scope of interest of the thesis presented here is centred on visual localisation systems which use a reference model. They are denominated model-based visual localisation systems.

This chapter is divided as follow: first of all, it is important to understand how the AR applications work. For this reason, the first section explains

the basis of an AR system. Then, camera calibration process is exposed. Later, model-based visual localisation systems are defined. Finally, the last section is dedicated to analyse the use of surface normal vector in literature due to the hypothesis defined in this thesis.

2.1 Fundaments of augmented reality

The main goal of AR applications is to render a virtual object in a real environment. The real and virtual objects must coexist in the same space. With that purpose, an AR system is taking the following steps continuously:

1. Capture images of the real surroundings.
2. Define a virtual camera which simulates the real one. That camera generates a new image in which the content obtained from the real camera as well as the virtual object are visualised aligned.
3. Render the image generated by the virtual camera in a display. The virtual object should be integrated between real objects as realist as possible. With that purpose, it has to be taken into account the influence of illumination or occlusions among others.

The second of the mentioned steps is critical for AR applications. The total integration of virtual objects in the reality depends on the correct execution of mentioned step. The quality of the AR system depends on this step, mainly.

For this reason, in order to perform that step properly it is very important to parameterise the virtual camera, to know the parameters which define it. With that purpose, measured images by the real camera are used by AR systems to make that parameterisation.

In that way, it can be define AR applications as the opposite of the VR or computer graphics applications. While VR systems start out from a virtual model and the camera model parameterisation must be done to obtain images, the AR systems depart from real images and must parameterise a virtual model matching those images. Figure 2.1 shows those paths.

Summarising, in AR applications the alignment between real and virtual objects depends on the similarity between the real and the virtual camera. Those images, where real and virtual objects are shown together, are generated by the virtual camera. In order to describe the behaviour of a real camera a physical camera model as well as measures from the real images are used.

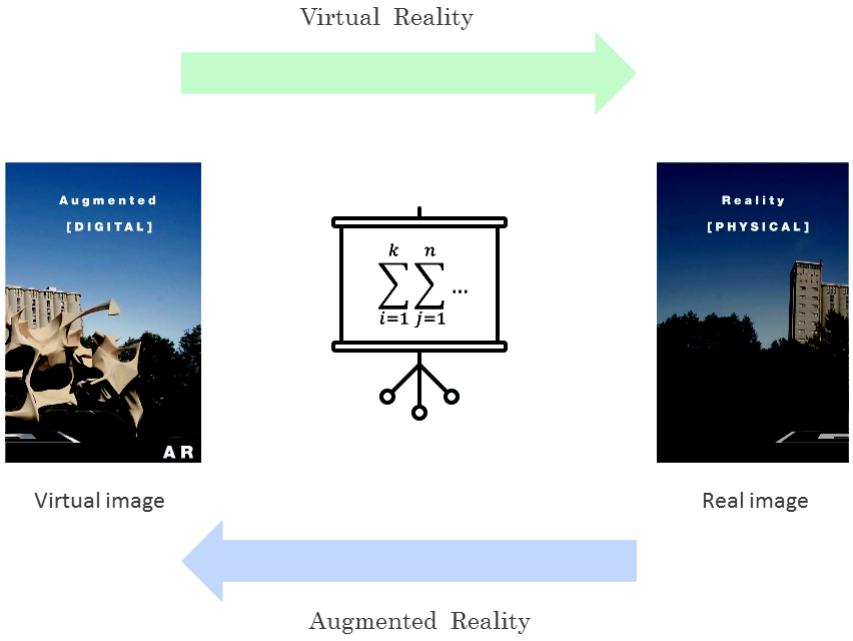


Figure 2.1: AR and VR parameterisation steps

The simplest representation of the perspective projection model is a pinhole camera model. Next section deepens in the description of this model.

2.1.1 Pinhole camera model

The camera model describes the relationship between 3D points, $\vec{X} = (X, Y, Z)^T$, in the real scenario and their projections, $\vec{x} = (x, y)^T$, into the image plane, I. Several camera models have been defined in the literature but the pinhole camera model is the most appropriate to represent digital cameras [54]. The pinhole model describes the projection of a 3D point, $\vec{X}$, in object space to a 2D point, $\vec{x}$, in image space. In Figure 2.2 the relationship between the coordinate systems in a pinhole camera model is described.

The most important elements of a pinhole camera model are enumerated below:

- An optical center: the camera center ($\vec{C}$).
- Image plane or focal plane (I).

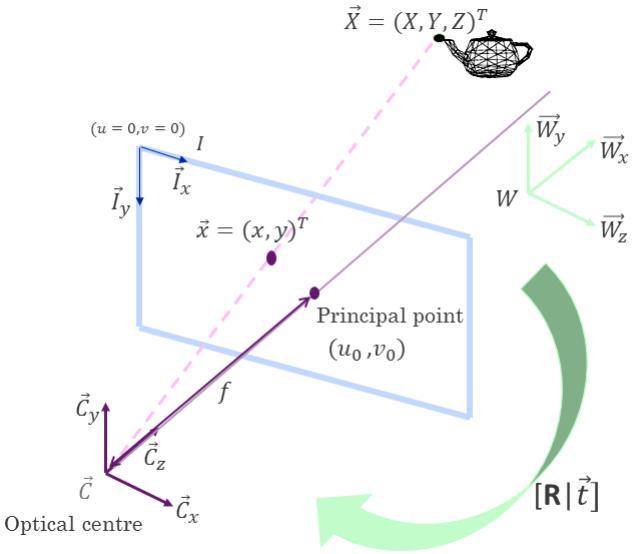


Figure 2.2: The camera model

- Focal distance: distance between the projection center and image plane (f).

The principal axis is the line perpendicular to the image, I, whose origin is in the optical center, $\vec{C}$. On the other hand, the intersection point between the principal axis and the image plane, I, is known as the principal point.

The projection of the 3D point, $\vec{X}$, into the image plane, I, is represented by $\vec{x}$. It is the intersection between image plane, I, and the line joining $\vec{x}$ with $\vec{X}$ through the optical center $\vec{C}$.

Three different coordinate systems are involved in this procedure:

- The world coordinate system (W).
- The camera coordinate system (C).
- The image coordinate system (I).

The camera pose matrix or extrinsic matrix $[\mathbf{R}|\vec{t}]$, represents the relation between world coordinates, W, and camera coordinates, C, i.e the camera pose (Equation 2.1).

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} \alpha_x & 0 & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{R}|\vec{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.1}$$

The first vector of Equation 2.1 represents the position of the point in the image plane (in I). The second element of the equation, the 3×3 matrix, is the intrinsic matrix (denominated **K** matrix), that defines the internal parameters of the camera, which are constant. These parameters are $\alpha_x = fm_x$ and $\alpha_y = fm_y$ where f is the focal distance and m_x, m_y are scalar factors. On the other hand, $(u_0, v_0)^T$ is the principal point (in I).

The $[\mathbf{R}|\vec{t}]$ defines the camera pose and the last vector represents the 3D position of the point in W. Note that points, in image coordinates and in world coordinates, are expressed in augmented notation of homogeneous coordinates. Using this type of coordinates, the pinhole model can be expressed as a linear transformation.

In a resume way, pinhole camera model can be expressed as Equation 2.2.

$$2D = \mathbf{K}[\mathbf{R}|\vec{t}]3D \tag{2.2}$$

2.1.2 Pinhole model parameterisation

AR systems follow a pinhole camera model, as mathematical model to parameterise the virtual camera and make the alignment between real and virtual objects properly.

In order to solve that parameterisation, the value of the variables that intervene in Equation 2.1 must be obtained. That equation is a linear transformation with six degrees of freedom, where at least four 3D-2D pairs are needed to solve $[\mathbf{R}|\vec{t}]$ [25]. With that purpose, two steps are identified: (i) calibration and (ii) tracking or localisation. Figure 2.3 shows those steps.

Calibration means to obtain the internal parameters of the real camera. So, it is the process to measure the **K** matrix of Equation 2.2.

On the other hand, tracking or localisation is to get the correct $[\mathbf{R}|\vec{t}]$ matrix for each image. In order to obtain camera matrix, it is also needed: (i) to measure the 3D-2D points pairs and (ii) to generate the equation system from at least four point matches (3D-2D pairs) to solve $[\mathbf{R}|\vec{t}]$ [25].

Calibration process as well as localisation process should be done from images. It means that computer vision techniques are used in order to make

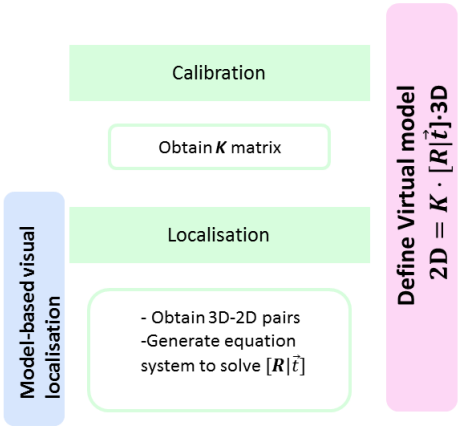


Figure 2.3: Main steps to be followed in order to define the virtual camera

the needed measurements from images and obtain mentioned values for the variables.

Model-based visual localisation systems include all the vision based methods which use a reference model in order to obtain needed points pairs (Figure 2.3). The next two sections of this chapter are dedicated to explain mentioned steps, calibration and model-based visual localisation systems.

2.2 Calibration

Calibration is the process performed in order to obtain the internal parameters of the camera, such as the focal length, the focal plane and the distortion of the lens to be able to subsequently determine the geometry of the objects observed by the camera, i.e **K** matrix (Equation 2.2). Two main ways to perform that process are classified: a pattern based calibration and auto-calibration.

Focusing on pattern based calibration processes, according to [54] the most used algorithms were presented by Tsai [62] and Zhang [69]. These kinds of methods use objects with known geometry to perform the process. These known objects are denominated calibration patterns. Figure 2.4 shows an example of calibration pattern. The most usual calibration pattern is a chessboard or a rectangular grid of dots. In these kinds of calibration patterns the points can be extracted easily, and matching the images with the geometric information is done quickly and accurately in order to find

the parameters of **K** matrix.

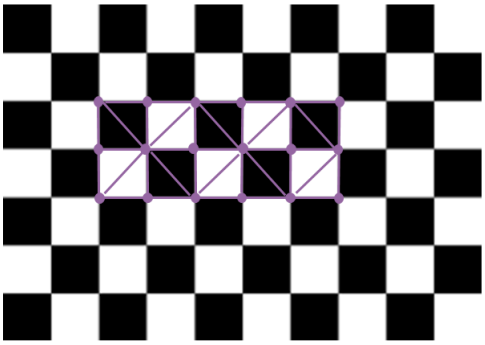


Figure 2.4: Usual calibration pattern

On the other hand, auto-calibration processes find the internal parameters of the camera without following a known geometry into the images. They take advantage from the rigidity constraints presented in the scene and simplifying assumptions about the camera (e.g., rectangular/square pixels, known principal points, constant intrinsic parameters) [2]. They can also set restrintions on the camera motion [27].

Auto-calibration involves 3D reconstruction process. Point correspondences between images, and the fundamental matrix computed from these point correspondences, are enough to recover the internal orientation of the camera (its calibration) [36]. It means, that the calibration is done while 3D reconstruction of the environment is perform to localisation purposes [26][48]. The calibration and localisation are done together. Section 2.3 explains the localisation system which use a 3D reconstruction (3D point cloud) of the environment as reference model. In addition, Chapter 3 will explain 3D reconstruction problem, as well as the fundamental matrix concept.

2.3 Model-based visual localisation systems

Localisation, or tracking, term is used to describe the dynamic estimation of the AR device pose (orientation and position), i.e the $[\mathbf{R}|\vec{t}]$. Due to the fact that AR applications operate in real-time, pose estimation must be updated continuously [54].

The visual localisation systems must identify the camera pose from the images (2D). Model-based visual localisation systems must compare those

images with some reference model. The model can be a marker (or makers), CAD model or the map of the environment (3D point cloud). This thesis is focused on the third group. It should be also noted that this thesis is developed for monocular systems, i.e systems which use a single camera.

Taking the pinhole camera as a mathematical model (see Equation 2.2) and knowing that the goal is to obtain the $[\mathbf{R}|\vec{t}]$ for each image, model-based visual localisation systems rely on matching 3D points, previously identified in an environment (using knowing reference model), with their corresponding 2D projections measured in each image captured by the camera. It is denominated *inliers* to the subset of 3D–2D correspondences or pairs that are correctly matched. So, for each image the system obtains a number of inliers which are used later on to update the global pose of the camera from generated equation system. The matches which are incorrect are denominated outliers.

There are different methods to solve mentioned equation system, depending on the restrictions added to the system. In that way, $[\mathbf{R}|\vec{t}]$ could be solved using Homography matrix (denominated **H** matrix) or Direct Linear Transformation method [25], mainly.

Model-based visual localisation systems which use markers as a model add a new restriction to the pinhole camera model; markers are flat and they find 2D-2D pairs instead of 3D-2D.

Those which use a 3D point cloud as a reference model use Direct Lineal transformation method to obtain the camera matrix. However, those which use markers use **H** matrix to solve $[\mathbf{R}|\vec{t}]$.

For all this, the background study done for this thesis is focused on those which use markers or 3D point cloud. The following sections will deepen in each methods.

2.3.1 Markers

The markers are usually printed papers which are placed in the environment. These marks are known previously. The geometry and appareance of these marks will be captured by the camera of the device and recognised thanks to a previous training. Localisation using markers became popular thanks to ARToolkit [29] and ARToolkitPlus [66]. Figure 2.5 shows an usual ARtoolkit marker.

The main goal of these kinds of localisation methods is to detect the marker and more concretely the four corners of the marker from images. As mentioned previously, model-based visual localisation systems rely on matching 3D points of the environment with their correspondences 2D points

Figure 2.5: Usual marker used for AR applications

in the image, to obtain an equation system which must solve the $[\mathbf{R}|\vec{t}]$ according to pinhole camera model defined by Equation 2.2. Detecting the four corners of the marker is the way to generate mentioned matching. On the other hand, Homography matrix is used to solve $[\mathbf{R}|\vec{t}]$ from obtained pairs.

2.3.1.1 Find four corners

For this reason, first of all it is essential to detect the corners of the marker. In that way, the images are converted into binary image (black and white). Then thresholding is used in order to delimit the marker in the image. It can be done automatically or manually.

Automatic thresholding is done analysing histograms and adapting the threshold based on the gradient of the logarithm of the image intensity [44]. This process is computationally expensive.

On the other hand, perform thresholding manually means determine locally (4×4 subarea) and interpolate it linearly over the image [65].

Once the thresholding is done, and the position of the marker in the image delimited, edges are found. It is considered a edge if there is a transition between white and black pixel. Each edge is candidate for be the marker if it is enough big and fit a quadrilateral [65].

Figure 2.6 shows the process followed to fit a quadrilateral. This process chooses an arbitrary point denominated a as a start point. The point with the maximum distance to a is considered the first corner, p_1. The centroid, m, of the marker is computed. The corner points p_2 and p_3 lie on the other side of the diagonal $dg_{1,m}$ through p_1 and m. The forth corner, p_4 is identified as the farthest point from p_1 on the left of the diagonal $dg_{2,3}$ from p_2 to p_3.

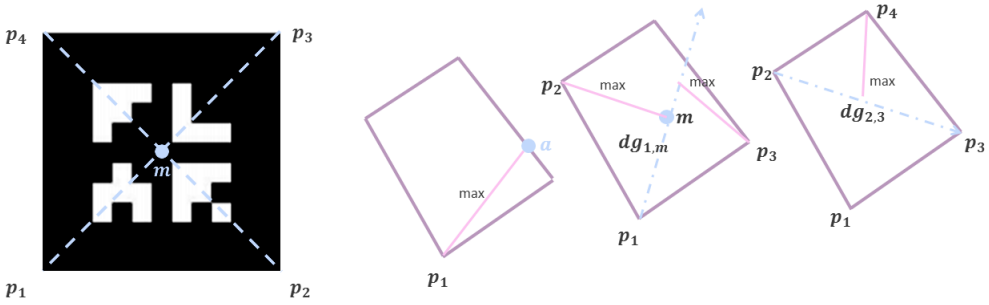


Figure 2.6: Steps for fit the quadrilateral by [54]

2.3.1.2 Pose estimation by Homography

Markers have a flat geometry, i.e the poses of the corners are defined in 2D. Consequently, in this case, 2D information are the only data used in order to estimate camera matrix.

Assuming that Z value of the world coordinates is zero, it is pretend to simulate that the point lies on a plane, it is located in a 2D space (Figure 2.7). **Projective transformation** is defined as transformation that maps points into points, lines into lines, planes into planes and any two incident elements into two incident elements. So, the camera model can be defined by projective transformation, since it is needed to know the camera matrix which became $\vec{x}_i$ point into $\vec{x}_i^{'}$ point.

The mathematical definition of a projective transformation is as follows (Equation 2.3): a planar projective transformation is a linear transformation on homogeneous 3-vectors represented by a non-singular 3×3 matrix [25]:

$$\begin{pmatrix} x_1^{'} \\ x_2^{'} \\ x_3^{'} \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \tag{2.3}$$

Or, in an abbreviate way:

$$\vec{x}^{'} = \mathbf{H}\vec{x} \tag{2.4}$$

where, $\mathbf{H}$ is Homography and it is a 3×3 homogeneous matrix, which given a set of at least four points $\mathtt{x_i}$ in a plane ($\mathbb{P}^2$) and a set of corresponding points $\mathtt{x_i^{'}}$ also in a plane ($\mathbb{P}^2$), computes the projective transformation that takes each $\vec{x}_i$ to $\vec{x}_i^{'}$ [25]. In this case, the points sets $\mathtt{x_i}$ and $\mathtt{x_i^{'}}$ are the corners of the marker.

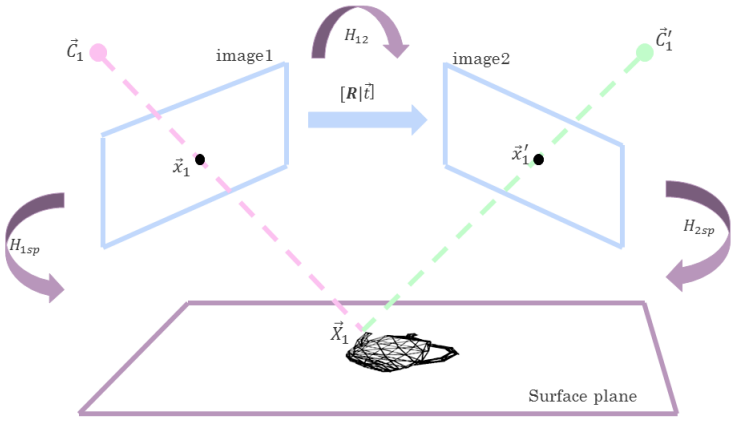


Figure 2.7: Projective transformation between two images. $\vec{X}_1$ is the position of the 3D point. The value of the Z component is zero. Consequently $\vec{X}_1$ lies on a surface plane. On the other hand, $\vec{x}_1$ and $\vec{x}'_1$ are the projection of $\vec{X}_1$ into image 1 and image 2 respectively. With the **H** matrices, the planar transformation between different planes is defined: from image 1 to surface plane, from image 2 to surface plane or from image 1 to image 2.

Hartley et al. [25] define the decomposition process to obtain $[\mathbf{R}|\vec{t}]$ from **H** matrix.

2.3.2 3D Reconstruction

Model-based visual localisation systems which use a 3D point cloud of the environment as a reference model, use different methods to stablish the relationship between 3D points and their correspondences in each image: feature detection, optical flow or template matching are some of these techniques.

Once the pairs are obtained, these methods use Direct Linear transformation to solve $[\mathbf{R}|\vec{t}]$.

These methods involve a process to obtain the 3D point cloud. Nevertheless, this related work is focused on the process to get 3D-2D pairs and to solve $[\mathbf{R}|\vec{t}]$ assuming that 3D point cloud is known. In that way, Chapter 3 explains how can be this reconstruction obtained.

The following sections are dedicated to explain Direct Lineal Transformation method to obtain camera matrix and to explain the different techniques to get the 3D-2D pairs by model-based visual localisation systems, i.e feature detection, optical flow and template matching methods are explained.

2.3.2.1 Pose estimation by Direct Lineal Transformation

Direct Lineal transformation (from now on, DLT) [1] is an algorithm which calculates a transformation matrix and obtains the extrinsic parameters of the camera parameters according to the parameter decomposition [70]. Starting from a set of 3D-2D matches, the system of equations is reordered so that a homogeneous system is generated and the least squares can be applied. In that way, DLT allows to satisfy Equation 2.5 where $\boldsymbol{A}$ is the known parameter, i.e 3D-2D pairs and $\vec{\boldsymbol{X}}$ that which want to be obtained, i.e $[\mathbf{R}|\vec{t}]$.

$$\boldsymbol{A}\vec{\boldsymbol{X}} = 0 \tag{2.5}$$

Hartley et al. [25] define the decomposition process to obtain $[\mathbf{R}|\vec{t}]$ from $\vec{\boldsymbol{X}}$.

2.3.2.2 Feature detection

This method use the most characteristic points of an image in order to get 3D-2D matches or pairs. These characteristics points, also named features must be detected in each image. Then, they have to be compared with the 3D points, to identify the points that are correspondences. With that purpose, descriptors are the way used to define the points.

But, what is a feature and a descriptor? how can be detected a feature? and how can be compared the points?

There is no universal or exact definition of what constitutes a feature. A feature can be defined as an interesting part of the image. So, edges, corners and interest points are the characteristics that can be found in an image.

A corner can be defined as the intersection of two edges. A corner can also be defined as a point for which there are two dominant and different edge directions in a local neighbourhood. On the other hand, an interest point is a point in an image which has a well defined position and can be robustly detected.

The work [56] defines a right interest point as a point whose area around is distinct. It means that it is textured, and in a small area around the point high contrast intensity changes are identified.

A good corners/interest points detector algorithm should have properties in order to help in feature identification: (i) it should detect all the corners of the image, (ii) it should not detect false corners, (iii) they should be well localised and (iv) should be robust and efficient.

For example, is better if the feature is not part of a repetitive structure to avoid the confusion with other points in the same image. Additionally, the same feature should be identified independent to some observation parameters, such as viewpoint or illumination conditions, being also robust to rotation, scale or perspective transformations.

There are in the state of the art several algorithms to detect features. Below, some of the most commons are explained.

Harris Corner detector [24] is a popular corner detector algorithm. A corner is defined as an image region that has intensity changes in different directions. For this reason, this algorithm uses an image auto-correlation to determine corners: a small window is defined around each pixel or point. A matrix (denominated $\mathbf{A}$) is calculated for a window that covers a point p that is located in $(x, y)^t$ position. Mentioned matrix($\mathbf{A}_{(x,y)}$) describes how similar a sub-image $\mathbf{I}_{(x,y)}$ is to a shifted version of itself. The elements of the $\mathbf{A}$ matrix represent illumination changes in the neighbourhood of the pixel (the strength of the image gradients is expressed in the eigenvalues, λ_1 and λ_2 of $\mathbf{A}$). The intensity changes that occur at each pixel will indicate if there is a corner or not. The eigenvalues of the matrix and the value of auto-correlation function (AC) of Equation 2.6 (where r is a constant) applied to these eigenvalues indicates the intensity changes of this pixel.

$$AC = \lambda_1\lambda_2 - r(\lambda_1 + \lambda_2)) \tag{2.6}$$

There are three possibilities depending on the value obtained from the auto-correlation, AC, function. In Figure 2.8 the three options are shown. If there are no changes in either direction (a both eigenvalues are small), it is a flat region. However, if a change in the edge direction is appreciated (if one eigenvalue is large) then this region represents an edge. On the other hand, if there are significant changes in all the directions (both eigenvalues are large), the region represents a corner.

The points detected by this procedure remain invariant to changes in scale, rotation, illumination and image noise. Nevertheless it is not robust against scale changes.

Scale-Invariant feature Transform (SIFT from now on)[34] takes into account scale problem, being also invariant to illumination, noise, and orientation changes. The scale problem is solved, detecting local extrema of the images filtered with Differences of Gaussian (DoG from now on). This method works in the scale space, which is obtained from an image pyramid.

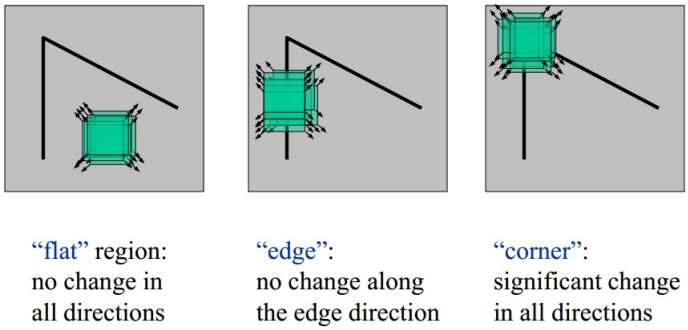


Figure 2.8: Harris Corner Method [24]

So, firstly a scale space must be built. With this purpose an image pyramid is built from the convolution of image with differences of Gaussian filters.

Then, a point detection in scale space is done. This step performs a search on different scales and dimensions of the image to identify possible features, invariant to changes in orientation and scaling. DoG function is used for this purpose. The points of interest are located at the relative maxima and minima.

SIFT is a better or superior detector than Harris, because of his notion of scale, but the Gaussian convolution makes it computationally expensive.

Features from accelerated segment test (FAST from now on) is a detector presented in the work [52].

FAST operates by considering a circle of sixteen pixels, c_i where $i \in [1..16]$, around the corner candidate p as can be seen in Figure 2.9. The detector classifies p as a corner if exists a set of `n` contiguous pixels in the circle which are all brighter than the intensity of the candidate pixel p plus a threshold th. The pixel p is also classified as corner if all the contiguous pixels in the circle are darker than $p - th$.

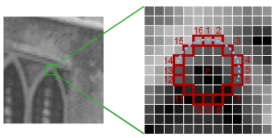


Figure 2.9: FAST Feature Detector [52]

Once the features are detected, it must be created a **descriptor** which identifies the characteristics of the points. A descriptor is a a data structure which contains the characteristics of the feature. A descriptor must be different for each feature, but identical for arbitrary viewpoints and lighting conditions.

An ideal descriptor should take into account the texture of the local area of the feature. In addition, it should be invariant to changes in illumination, scale, rotation and affine transformations.

SIFT creates one of the most popular descriptors [54]. For each feature candidate or point p the orientations are stored in a vector building a descriptor. With this purpose, patch around point p is taken. This patch is also divided into a sub-patches or neighborhoods. A histogram is created for each sub-patch. These histograms are built using known orientations of each pixel of the sub-patch, weighted by the magnitude and the Gaussian of the distance of the pixel to p. Each value of the histogram is stored as a individual value, in a vector. This vector is the descriptor of p. In Figure 2.10 , explained process can be shown easier. In a standard SIFT descriptor, a patch created around of each point p is 16×16 size. Then, this patch is divided into a 4×4 neighborhoods. Histograms are computed for 8 bins. Consequently, there are $4 * 4 = 16$ histograms, and they are composed by 8 bins, i.e the descriptor is a 128 size $(4 * 4 * 8)$. Finally the vector is normalised in order to avoid or minimise the illumination influence. In some works as [30] to reduce the effects of non-linear illumination a threshold of 0.2 is applied and the vector is again normalised.

Fast Retina keyPoint (FREAK, from now on)[3]. It is a feature point descriptor based on the human retina. It is a binary descriptor which use a human retina as pattern.

Binary descriptors are composed by three parts: (i) a sampling pattern, (ii) orientation compensation and (iii) sampling pairs. The pattern is used in order to sampling points which are located in the neighbourhood of the interest point or feature. Orientation compensation is a mechanism which measures the orientation of the feature and rotate it to make it robust against rotation changes. Finally, sampling pairs means to create pairs to be compared in final building process.

Figure 2.11 shows a retina pattern. It is a circular grid with some characteristics.

- The density of the sample points increases as central points is closer.

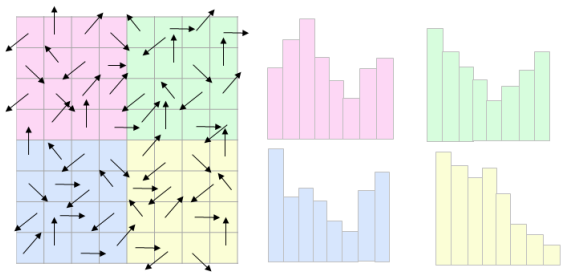


Figure 2.10: Example of the process of creating a SIFT descriptor. It can be seen, how 4 sub-patches are taken for this particular case as an example. Usually, 16 sub-patches are created. Histograms for each sub-patches are also generated, according to orientations of each pixel in sub-patches. Histograms are composed of 8 bins

- The areas that are taking into account for the description of the sample point are overlapped. It means that redundant information is generated improving the performance of the descriptor.

- Each sampling point is smoothed with Gaussian kernel where the radius of the circle illustrates the size of the standard deviation of the kernel.

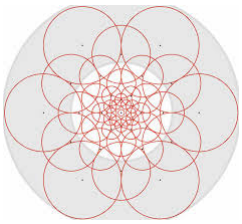


Figure 2.11: 'Illustration of the FREAK sampling pattern similar to the retinal ganglion cells distribution with their corresponding receptive fields. Each circle represents a receptive field where the image is smoothed with its corresponding Gaussian kernel.' [3]

FREAK chooses the sampling pairs maximising variance of the pairs and taking pairs that are not correlated.

According to Alahi et al. 'FREAKs are in general faster to compute with lower memory load and also most robust than SIFT' [3].

Once features are identified in images and their descriptors created, the correspondence between features and 3D points of the reconstruction must be found. Features of the image must be compared with 3D points of the reconstruction identifying the similarities.

The simplest way to make this matching and comparison is to apply Euclidean distance to the descriptor vectors. For a point (and its descriptor) in the reference image the descriptor in the comparison image with the smallest distance represents the best match. In order to guarantee the authenticity of the match, the distance proportion of the best match and the second best one must be bigger than a established threshold (usually 80%).

However, this method can be very expensive in computational terms, because all the features must be compared one by one. In order to improve this cost the use of hierarchical structures as k-dimensional tree (from now on Kd-Tree) helps. This data structure split the space agglutinating features of the same space in the same tree or data structure.

It should be considered that some the matches will be incorrect. They will be denominated outliers.

2.3.2.3 Optical flow

In that kind of model-based visual localisaton system, it can be distinguished two main families of tracking approaches [33]. They can be differentiate depending on the nature of the image features being used to get the 3D-2D matches. In the first family there are located all the algorithms which are based on match the projections of the target object 3D edges to area of high image gradient. They are denominated edge-based. On the other hand, in the second family, there are included all the techniques that rely on use the information provided by pixels inside the projections of the object, to obtain the 3D-2D matches. Optical flow, as well as template matching techniques are grouped in this second family.

Optical flow technique, is the measurement of the apparent motion of image objects in frame sequence, caused by the movement of the camera [33]. So, instead of detect interest points in each image to match correctly with its correspondence 3D points, optical flow estimates the displacement suffers from one frame to another. The displacement is applied to the 3D point projections in the consecutive frames. In this manner, the correspondence 3D-2D matches for each frame are obtained, and consequently the camera pose is estimated.

Optical flow works assuming that: (i) the pixel intensities of an object do not change between consecutive frames and (ii) pixels in the same neigh-

bourhood have similar motion.

So, considering $p = (x, y, t)^T$ a point or pixel in the first frame, (d_x, d_y) the distance that p moves to the next frame (taken d_t time), and taking into account the first assumption, the Equation 2.7 is defined.

$$p(x, y, t)^T = p(x + d_x, y + d_y, t + d_t)^T \tag{2.7}$$

Taking into account the Taylor series approximation (right hand) [46], the Equation 2.8 appears. Focusing on the elements of the equation: f_x and f_y are image gradients. Similarly f_t is the gradient along time. The unknown variables are (u, v). In order to know the value of u and v an equation system has to be created.

$$\begin{aligned} f_x u + f_y v + f_t t = 0 \\ f_x = \frac{\partial_f}{\partial_x}; f_y = \frac{\partial_f}{\partial_f} \\ u = \frac{d_x}{d_t}; v = \frac{d_y}{d_t} \end{aligned} \tag{2.8}$$

Kanade Lucas Tomasi (KLT from now on) [35][60] is a classical model-based tracking method which uses optical flow. Taking into account the second assumption of optical flow (pixels in the same neighbourhood have similar motion), KLT takes a 3×3 size patch around the point p. In this way, all the point of the patch, 9, have the same motion. Now, (f_x, f_y, f_t) can be found from these 9 points. So, now the problem becomes solving 9 equations with two unknown $((u, v))$ variables, which it is over-determined.

Optical flow is a very lightweight algorithm in computational terms. So, it seems to be a very suitable method to real-time model-based tracking systems. However, this technique tends to drift, and the error propagates during the camera motion.

2.3.2.4 Template matching

Template matching techniques are focused on the correction by appearance. It means that this technique is not estimating the motion of the 2D point during the camera sequence. In this way, two components can be identified in a template matching process:

- **Source image**: The image or sub-image in which it is expected to find a match to the template image.

- **Template image**: The patch image which will be compared or find in the source image.

Using template matching, the 3D-2D matches are found taking into account the appearance of the window or patch created around the projections of the 3D points into images. In this way, template matching is used to improve the projections of the 3D points in each frame.

Among the most used model-based visual localisation methods are those that use feature descriptors. They are very robust thanks to their invariance to illumination, scale and orientation changes. Consequently, the probability of getting correct matches is high. However, they present efficiency problems.

For this reason, while feature descriptor-based methods are used in the initial matching, faster techniques like optical flow are used to track the 2D motion throughout video sequences. Some authors use optical flow to estimate the image velocity, or the displacement of a specific point from frame to frame [13] [15].

Although optical flow has lower computational cost, point displacement estimation is less accurate compared with other methods, and tracking is prone to drift.

A third family of algorithms, template matching methods, try to balance the problems mentioned in the previous methods. Template matching methods use different operators to estimate the similarity of images. Good examples are provided by [19] [31].

In these kinds of methods, for each 3D point, a patch around its projection into a denominated reference image is saved as a reference patch. Then, template matching techniques are applied during the tracking. However, the matching of reference patches within a given image can fail because of perspective distortion [38]. When the orientation disparity between the reference image and the camera image is high, the matching process tends to fail.

Summing up, the use of feature descriptors during all the process is the most accurate option, but it is very expensive in computational terms. Optical flow is fast but tends to drift.

On the other hand, template matching is not as expensive as feature descriptors and it does not tend to drift. Even more, it is robust against illumination changes. Nevertheless, it is not robust against perspective distortions created by camera motion. Consequently, the method may fail to obtain 3D–2D correspondences affecting the stability of the localisation process.

This thesis is focused on solve that distortion problem, using a surface normal, in order to become model-based visual localisation systems which use template matching, in fast and robust (against illumination changes as well as rotation) methods.

For this reason, how template matching works will be explained later in Chapter 4 and the architecture specifications of the model-based visual localisation systems which use it in Chapter 3.

2.3.3 vSLAM

Model-based visual localisation systems use a reference model to localisation purposes. Until now, different methods have been described, based on the type or model uses as reference. In that way, the methods described assume that the model is known. However, there is a new family or model-based visual localisation system which does not have a model at hand. They are denominated vSLAM, from visual Simultaneous Localisation and Mapping [18]. They compute the map (3D point cloud) and localise the camera in that map simultaneously and online.

Due to this particular characteristic, vSLAM is known as chicken and egg problem: what should be carried out first? the construction of the map or localisation? Both need each other for their correct execution. In that way, vSLAM is described as a probabilistic problem. It means, that over discrete time steps t, for each observation done by the camera, o_t, it has to be computed and estimated the probability of location, x_t, and of the map, mp_t (Equation 2.9).

$$P(mp_t, x_t|o_t) \tag{2.9}$$

There are found in the state of the art different algorithms to solve mentioned vSLAM problem. Below, some of the most uses are briefly explained.

Extended Kalman Filter to vSLAM (EKF, from now on) [6] is a variant of Bayesian filter [20] which provides a recursive estimation of the camera motion as a non-linear estimation problem. It uses a state vector composed by the location and some map elements. The non-linear observation and transition models are used to estimate each state vector entry recursively. The uncertainty is represented by probability density functions. The optimal solution approaches by the recursive propagation of the mean and covariance of these functions.

The map in vSLAM uses visual landmarks, that are created along the motion of the camera. The complexity of EKF based vSLAM algorithms

increases according to the number of landmarks. For this reason, its use in large maps is very difficult [21].

Davidson [18] presents the first vSLAM algorithm using a single camera named MonoSLAM. This work employs EKF to estimate data. So, it was limited for working in small and indoor spaces.

A Factored Solution to SLAM (FastSLAM, from now on)[39]. In order to solve above mentioned problem with EKF based vSLAM methods, FastSLAM was designed. With that purpose, the problem of estimating landmarks positions can be divided into a n independent estimation problems. In that way the problem is decomposed into $n + 1$ problems: n problems for estimate the n landmarks and one problem to estimate the pose of the camera. Particle filters [50] are used with that purpose.

However this division become this technique very expensive in computational terms. The computational cost of these systems is logarithmic $O(ln)$, where ln is the number of landmarks.

Parallel Tracking and Mapping (PTAM, from now on)[31] is a modern approach of vSLAM concept. It separates the tracking or localisation process and mapping process into two separate tasks which are running on parallel threads. This system is based on the idea that it is not necessary to use every frame or image from 3D point cloud (or map) estimation, due to the fact that the images are very close each other and they have redundant information.

This characteristic made PTAM a very fast method which is also suitable to be carried out in mobile devices.

2.4 The use of surface normal in the literature

In model-based visual localisation system which 3D–2D matching must be done, template matching processes are very suitable in order to compare the appearance during the image sequence [16] [57]. Template matching processes are very robust against some appearance changes, as illumination. Nevertheless, these methods can fail due to perspective distortions and the motion of the object during the image sequence. Taking into account these perspective distortions, warping the patch (using surface normal vector) can palliate the problem [11] [55].

It can be observed in the last decade that the use of surface normal in order to improve vision localisation and 3D reconstruction processes

has increased. For example, for tracking processes, some researchers use a gradient-based image alignment method for matching [31] [38]. In consequence, features are compensated more accurately during camera movement. Although authors describe the use of surface normal vector for localisation, they do not study and analyse the benefits of using such approaches.

In the same way, regarding 3D reconstruction, Wu et al. show an improved 3D reconstruction algorithm due to the use of surface normal vectors in the matching steps [68]. Goesele et al. [23] present an algorithm that performs 3D reconstruction of different buildings and sculptures from a set of images found in the Internet photo-sharing sites. These works suggest that considering surface normals vectors significantly improves the matching results.

Another example of the use of surface normals vectors in 3D reconstruction is presented by Furukawa et al. [22]. They obtain very effective model reconstructions of sculptures and architectural elements. Their work emphasizes that performance is effective due to the use of techniques for enforcing local photometric consistency and global visibility. These enforcers are achieved estimating the surface orientation in the patch matching process.

Creating 3D point clouds data from camera images is an essential task in markerless indoor tracking processes. It provides the 3D information required for camera localisation.

Focusing on this type of tasks, Mostofi et al. present a RGB-D indoor plane-based 3D modeling method [41]. This provides rich information that simplifies the disambiguation of different places. They use the surface normal vector calculation for this purpose.

Another significant example is the recent work by Charmette et al. [17] who present a novel robot localisation process that is improved using surface normal vectors in the matching step. In this work, the descriptors are 3D patches warped to match the current viewpoint. The same philosophy is applied by Wu et al. [68]. In this second example, the work is focused on the 3D reconstruction, while in the first one it is focused on the tracking field.

Summing up, the state of the art shows that there are works in the field of 3D reconstruction and localisation that estimate the surface orientation of 3D points in order to improve their results. As a consequence, in the field of computer vision, it seems interesting to consider normal vectors in template matching techniques. However, there is not any work in the literature that studies that taking into account perspective distortion improves objectively these processes. The work we present proves in an objective way the hypotheses posed in Section 1.2.

Part II

Proposal

Chapter 3

Fundaments for the Benchmark Visual Localisation System

In order to prove the hypothesis defined in the Section 1.2 a visual localisation system is needed. This chapter describes this system and the global modules which form it. Along the thesis, it will be called the Benchmark Visual Localisation System (from now on, BVLS).

AR applications can use different methods in order to localise itself in real-time. This thesis is focused in those which use model-based visual localisation systems. Exactly in those which use a 3D reconstruction of the environment (3D point cloud) as a model, and template matching techniques to relate the 3D model information with image information.

In this way, two main tasks are identified. On the one hand, the reference model acquirement and on the other hand, the tracking process, which should localises the camera in that map. Usually it is mentioned that two main kinds of model-based visual localisation systems (using a single camera) are identified: Structure from Motion (from now on, SfM)[32] and visual Simultaneous Localisation and Mapping (from now on, vSLAM)[18]. They differ basically on the way they obtain the map of the environment (reference model): the first estimates the map in a previous and offline process and the second one is estimating the model and the camera pose (tracking) simultaneously and online.

Nevertheless, rigorously SfM is a method to obtain exclusively the reference model. It does not estimate the camera pose in real-time. vSLAM can be considered an evolution of SfM, which estimates both, the reference

model and camera pose, at the same time. BVLS will use SfM.

In the localisation systems which use the reference model obtained by SfM, as well as in vSLAM, 3D–2D matching must be done in the tracking process. Template matching processes are very suitable to compare image appearance along image sequences [31, 38]. Consequently they are convenient in order to make the matching. BVLS will also use template matching.

In this chapter firstly the SfM method to obtain the reference model is explained. Then, how the tracking process works is defined. Next section describes BVLS.

3.1 Reference model construction

This section explains the way that SfM method works. As mentioned previously, the thesis presented in this dissertation is focused in model-based visual localisation systems which use a 3D point cloud of the environment as a reference model. With this purpose SfM is one of the most common methods used in computer vision field [33]. SfM combines multi-view techniques with Bundle Adjustment process [61]. System Overview 3.1 resumes the goal of the SfM method as well as the input and output data of this process.

System Overview 3.1: SfM

Goal: To obtain the 3D point cloud which will work as a map of the environment

Input:

- `Im`: set of images

Output

- `PC`: 3D point cloud
- `KF`: *keyFrames* camera pose matrices

A 3D reconstruction of the environment is a set of 3D points, denominated 3D point cloud (`PC`). A set of images (`Im`), of the environment from different perspectives, is the input data for this process. These images are denominated *keyFrames* in order to distinguish from images that later are going to be used in the tracking process. The 3D position, $(X,Y,Z)^T$, of a point in real world is acquired from corresponding 2D points, $(x,y)^T$, in the *keyFrames*.

The reconstruction of a real scenario is generated from different camera positions (different images of the same scenario). With this purpose, the depth of each point is found by a triangulation process (later on, the reason why the triangulation process is used is explained in this section). Triangulation is a process which allows to obtain a 3D point from two or more 2D point correspondences. So, in order to achieve the goal of the SfM it is necessary to find the corresponding points in *keyFrames*, as well as the camera pose matrix of each *keyFrame* (`KF`). Finally the 3D position of each point (`PC`) is computed.

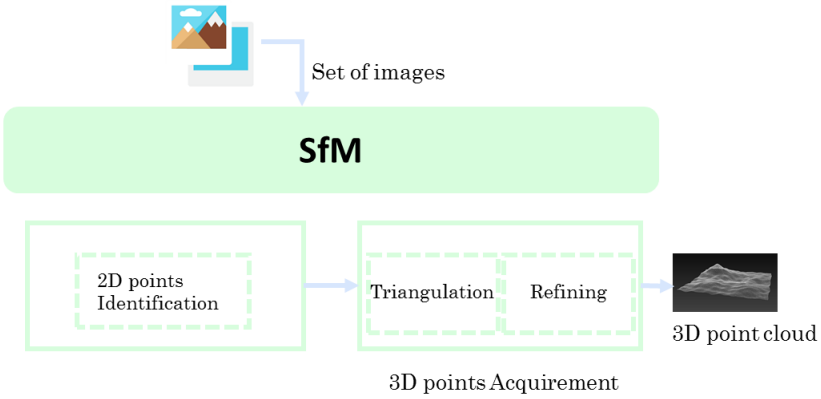


Figure 3.1: The overview of the SfM method architecture

The architecture needed to obtain these measures can be seen in Figure 3.1. Two main processes are identified: 2D points Identification and 3D points Acquirement.

The 3D points Acquirement process must determine points in 3D space. Each point in an image corresponds to a line in 3D space and all points on the line can be projected to the same point in the image. In that way, triangulation method allow to found the pose in the 3D space of one point, from two or more images. So, this process applies a triangulation method, in order to estimate the 3D pose from the 2D points identified in the previous process. As triangulation estimation can be uncertain an optimisation methods should be also use. Besides, the camera pose matrices for each *keyFrame*, are also estimated in this process. The correct performance of these two steps is essential in order to get a valuable 3D point cloud or reference model to use later in the tracking process.

3.1.1 2D points Identification

The first problem that must be solved is identifying 2D points in the *keyFrames*. Then, the correspondence between 2D points of the different *keyFrames* should be found. These steps must be followed:

1. Feature detection
2. Descriptor creator
3. Descriptor matching

1- Feature detection In Section 2.3.2.2 the most popular feature detectors in the state of the art were exposed. Each SfM application uses the detector which best fits the requirements of the system.

2- Description creator Once the features are detected in each *keyFrame*, it must be created a descriptor which identifies the characteristic of the points. In Section 2.3.2.2 the descriptors found in the literature were defined. Each application uses de descriptor which best fit the needs of the system.

3- Descriptor matching Once features are identified in the images and their descriptors created, the correspondence between features in different *keyFrames* must be found. Features of one *keyFrame* must be compared with features of the next image, and identify the similarities.

It should be considered that some matches will be incorrect. They will be denominated outliers.

3.1.2 3D points Acquirement

Once all the 2D points pairs are identified on the *keyFrames*, the corresponding 3D point for each 2D pair can be computed.

When a 2D pair is known, its corresponding 3D point can be computed using a triangulation method [25]. However, by triangulation the mistakes made in the different steps of the reconstruction are accumulated when adding new images and 2D points pairs, i.e drift problem appears. So, in order to minimise the error and stop the drift propagation a refining process is also applied. This refining process is done using Bundle Adjustment algorithm [61].

Triangulation The triangulation problem is to calculate the position of a point in 3D space, $\vec{X} = (X, Y, Z)^T$, from the position of this point in two images, $\vec{x} = (x, y)^T$ and $\vec{x'} = (x', y')^T$, [28]. Hence, two or more images are needed to get the third dimension. Let us show the reason.

Figure 3.2 shows point $\vec{x}_1$ in image. It is a 3D point projected into the image captures by the camera. The camera is placed in $\vec{C}$. With this data it is known that $\vec{X}_1$ point must be placed some where in the line. However, the distance (depth) from $\vec{X}_1$ to the image (projection plane) is not known.

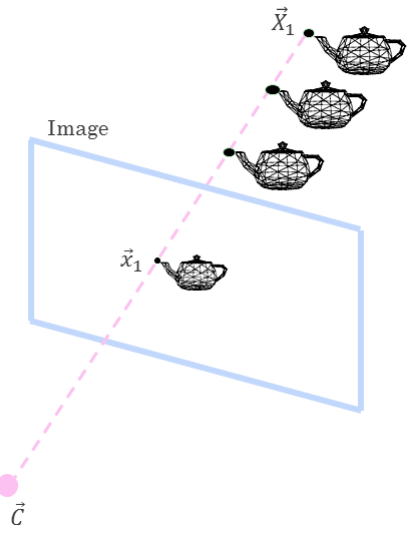


Figure 3.2: Projection of a 3D point in a single image

Nevertheless, as can be seen in Figure 3.3, the depth of $\vec{X}_1$ is clearly defined by the intersection of two rays. Each ray is launched from the center of each camera ($\vec{C}_1$ and $\vec{C}'_1$) through the projection of the point $\vec{X}_1$ in each image plane.

It has to be remarked that the triangulation must be done using two image planes which are not very close each other. Figure 3.4 shows how the closer the images are, the higher uncertainty level estimating the depth of the point.

In order to be able to triangulate (launch the rays from the camera centres through the projections) and obtain the 3D point, it is essential to know the camera matrices.

The camera matrices are computed using Epipolar geometry.

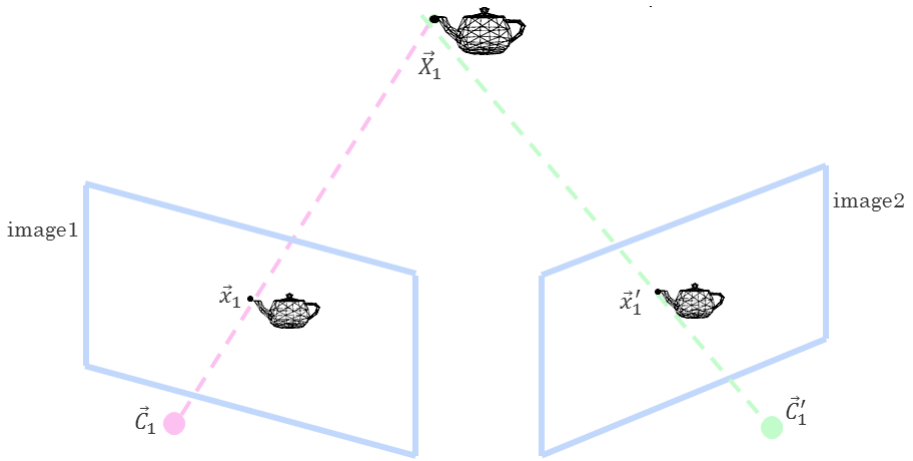


Figure 3.3: Depth of the 3D point from two images

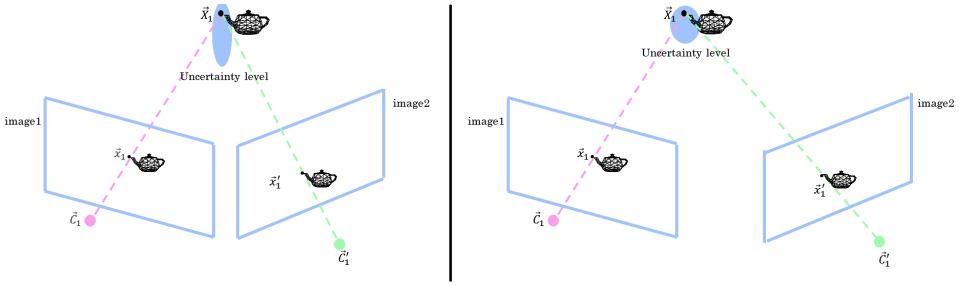
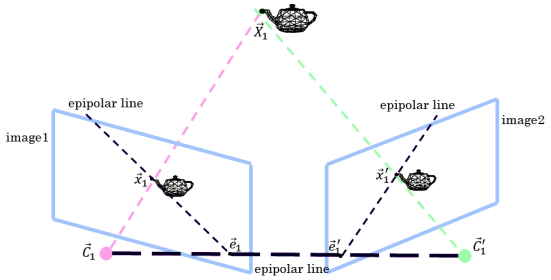


Figure 3.4: Uncertainty levels estimating the depth of the point by triangulation

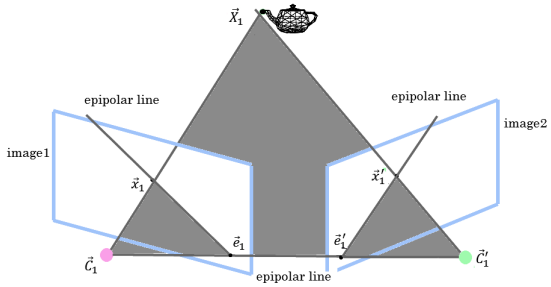
Epipolar Geometry is the geometry which defines a stereo vision system and in which two models of simple cameras (pinhole) and some points of interest denominated epipoles are combined. In Figure 3.5(a) a definition of epipolar geometry is shown and the following elements are identified: a projection center for each camera ($\vec{C}_1$ and $\vec{C}_1'$). There are also two projective planes (image 1 and image 2). Point $\vec{X}_1$ of the object in physical space has a projection on each projective plane and expressed by $\vec{x}_1$ and $\vec{x}_1'$ points respectively.

New interest points appear in this geometry, they are the epipoles (e_1 and e_1'), they lie on each plane (image 1 and image 2) and they are defined as the *picture or image* of the projection center of the other camera. For example, the epipole e_1, represents the projection of the $\vec{C}_1'$ in image 1. The

plane in the space, formed by the point $\vec{X}_1$ and two epipoles is denominated epipolar plane (Figure 3.5(b)).



(a) Global epipolar geometry view



(b) Epipolar plane, defined by the camera centres, epipole lines and 3D point

Figure 3.5: Epipolar geometry

Epipole geometry has some characteristics as: given a feature or point in one image, it can be found in the other image just along the corresponding epipolar line. This is known as a epipolar restriction.

To estimate the camera matrices using epipolar lines, Fundamental and Essential matrices are needed.The Essential matrix (**E**, from now on) defines the rotation and translation which relates the two cameras in the physical space. This matrix describes the purely geometrical relationship, i.e the translation and rotation in physical coordinates (normalised coordinates) to go from $\vec{C}_1$ to $\vec{C}_1'$.

On the other hand, the Fundamental matrix (**F**, from now on) contains the same information as the essential matrix, but also involves the intrinsic parameters of each camera (**K** matrix) and relates the points in image coordinates, i.e pixels.

According to [25], if 3D point $\vec{X}_1$ is imaged as $\vec{x}_i$ and $\vec{x}_i'$ in two image planes, Equation 3.1 and Equation 3.2 are satisfies for **F** and **E**.

$$\vec{x}_1^T \mathbf{F} \vec{x}_1 = 0 \tag{3.1}$$

$$\vec{\tilde{x}}_1'^T \mathbf{E} \vec{\tilde{x}}_1 = 0 \tag{3.2}$$

So, **E** as well as **F** can be computed from 2D point pairs of two *keyFrames*. Consequently the camera matrix can be computed by deduction.

To sum up, at this point, the first approximation of the 3D point cloud is estimated. Two chosen *keyFrames* were used with this purpose. Taking into account the epipolar geometry, **F** and **E** were calculated in order to obtain the camera pose ($[\mathbf{R}|\vec{t}]$) of these two *keyFrames*. Once the camera poses are known, the depth of each 2D point is estimated by triangulation, and consequently the 3D point cloud obtained.

Bundle Adjustment (BA, from now on) is the problem of refining a visual reconstruction to produce jointly optimal structure and viewing parameter estimates. It is really just a large sparse geometric parameter estimation problem, the parameters being the combined 3D point cloud, camera motion and calibrations (**K** matrix).

BA is almost invariably used as the last step of every feature based multiple view reconstruction vision algorithm to obtain optimal 3D point cloud and camera matrix estimation. Provided with initial estimates, BA simultaneously refines motion and 3D point cloud by minimising the reprojection error between the observed and predicted image points, which are expressed as the sum of squares of a large number of non-linear, real-valued functions. Equation 3.3 defined the reprojection error, between observed point $(x, y)^T$ and predicted $(x^{'}, y^{'})^T$.

$$\sum_i d(x_i, x_i^{'})^2 + d(y_i, y_i^{'})^2 \tag{3.3}$$

Usually BA is used in SfM algorithms. Generally SfM is performed offline because BA optimisation is very expensive in computational terms.

3.2 Model-Based tracking process

In this section the tracking method used in model-based visual localisation systems which use template matching is explained. It is assumed that a map of the environment is known. The goal of this section is to localise the camera pose in this map. System Overview 3.2 explains briefly the input

and output data of a tracking process in a model-based visual localisation system.

Taking into account methods of model-based visual localisation systems which use 3D reconstruction as a reference model (all of them exposed in Section 2.3.2), those which use template matching are the core of this thesis.

System Overview 3.2: Tracking Process

Goal: To know the pose in the real world of the user or camera in real-time
Input:

- `Im`: images captured by the camera in real-time
- `PC`: 3D point cloud which describes the scenario (map)
- `KF`: *keyFrames* camera pose matrices

Output

- $[\mathbf{R}|\vec{t}]$: Camera pose for each image

For each image obtained by the camera (`Im`), it is essential to identify the points of the 3D point cloud in the current image. Take a notice that images captured by the camera, are images obtained in real-time and they are not the same as *keyFrames*. As it was explained in Section 2.3 the goal of a model-based visual localisation systems is to identify 3D-2D matches in order to be able to solve $[\mathbf{R}|\vec{t}]$ according to pinhole camera model (Equation 2.2). With that purpose, the points of the 3D point cloud must be matched correctly with the 2D points of the image, in order to obtain the camera pose ($[\mathbf{R}|\vec{t}]$). Template matching techniques are used in this thesis to obtain those 3D-2D matches.

The architecture of a model-based tracking which follows the template matching technique is exposed in Figure 3.6. Two main processes can be identified: Approximate Projection process and Patch Fitting. The objective of the first is to project the points of the 3D point cloud in the current image as first approximation. On the other hand, the aim of the second process is to improve this projection to get a more accurate one.

These projections are used as 3D-2D points match, i.e for each image a 3D-2D match vector is stored. An equation system is formed from a set of matches. Each match satisfies Equation 2.1 where camera matrix is an unknown variable on the whole system. DLT [1] algorithm is applied to

obtain the camera matrix from those matches (Section 2.3.2.1).

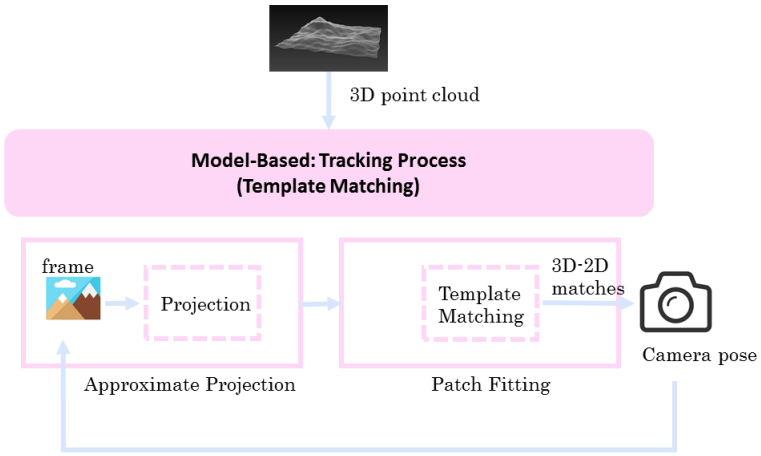


Figure 3.6: The overview of the template matching model-based tracking process

3.2.1 Approximate projection

The projection of the 3D points of the 3D point cloud in the current frame or image is done. It has to be noticed that the camera matrix of the frame is unknown, in fact to get that camera matrix, is the objective of the process explained in this section.

In order to solve this problem, it is assumed that the displacement between consecutive images is small. So, the extrinsic ($[\mathbf{R}|\vec{t}]$) parameters of the previous frame are used to project 3D points into the current frame or image. This provides an initial guess (approximation) about their projections.

Then, the Patch Fitting algorithm improves these projections.

Nevertheless, in the case of the first input frame, there is not previous frame, neither its corresponding camera pose. For this reason, an initialisation process is needed. The goal of the initialisation is to get the camera pose of the first frame. The initialisation process estimates the camera pose applying feature detection (Section 2.3.2.2)and DLT algorithm.

3.2.2 Patch Fitting

The projection of the 3D points in the image and a patch around this projection are used. Then, an adjustment of the projection is done, in order

to obtain more accurate projection match. The template matching technique itself is applied here. Template matching is a technique which allows to identify a position in image which match the given second image [45]. Given a 3D point, it is projected into a *keyFrame* (*keyFrame* is defined in Section 3.1). A patch around its projection is called *template image*. This image is used for improving the projection of a 3D point into the current image.

The approximate projection defines a source patch on the current image: a *source image*. The similarity between *template image* and *source image* is measured. A process is followed to find a patch within the current image that is the most similar to the *template image*.

In order to identify the position in the *source image* where the *template image* fits, it has to be compared the *template image* and *source image* by sliding. Sliding consists on moving the patch or *template image*, one pixel at a time (left to right and up to down) in the *source image*. For each movement a metric is calculated. It represents how 'good' or 'bad' the similarity at this location is (Figure 3.7).

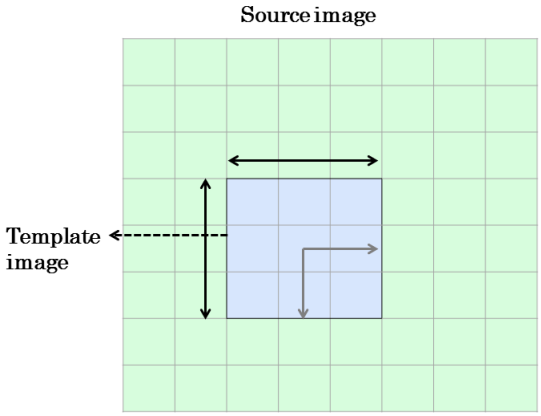


Figure 3.7: Template matching: how the *template image* is moving within the *source image* looking for the best similarity

This process is applied to a specific number of 3D points projections in order to obtain a good number of 3D-2D matches. It is considered a good number of matches when: (i) it is large enough to create an equation system which obtains a precise result and (ii) it is small enough to avoid slow down the system due to the associate computational cost.

Once, the two main processes are performed, the obtained 3D-2D matches

are used to calculate the pose of the current frame. With this purpose, DLT algorithm is applied to the equation system.

3.3 The benchmark visual localisation system

In order to prove the hypothesis exposed in this thesis, it is necessary to define a visual localisation system. A model-based visual localisation system is the base of BVLS. In this way, a SfM tracking system based in template matching techniques is used.

The schematic architecture of BVLS is presented in Figure 3.8.

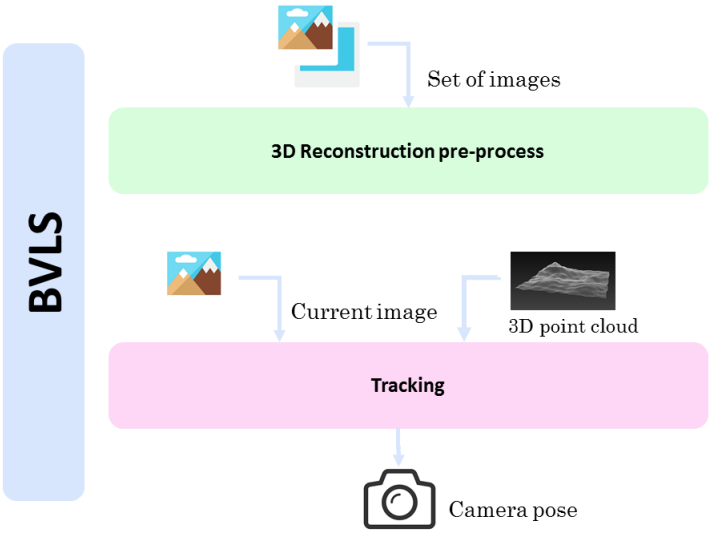


Figure 3.8: The overview of the benchmark visual localisation system

Two main tasks can be identified: *3D reconstruction pre-process* and *tracking*. In the first one, the reference model is obtained in a previous and offline process, i.e SfM is applied. Once, the map of the environment is available (3D point cloud obtained in SfM) the device must be localised in this map (*tracking*) using a model-based tracking processes.

A more detailed architecture of BVLS system is exposed in Figure 3.9. It takes into account how the SfM works and also model-based tracking processes which use template matching (explained in Section 3.1 and Section 3.2 respectively).

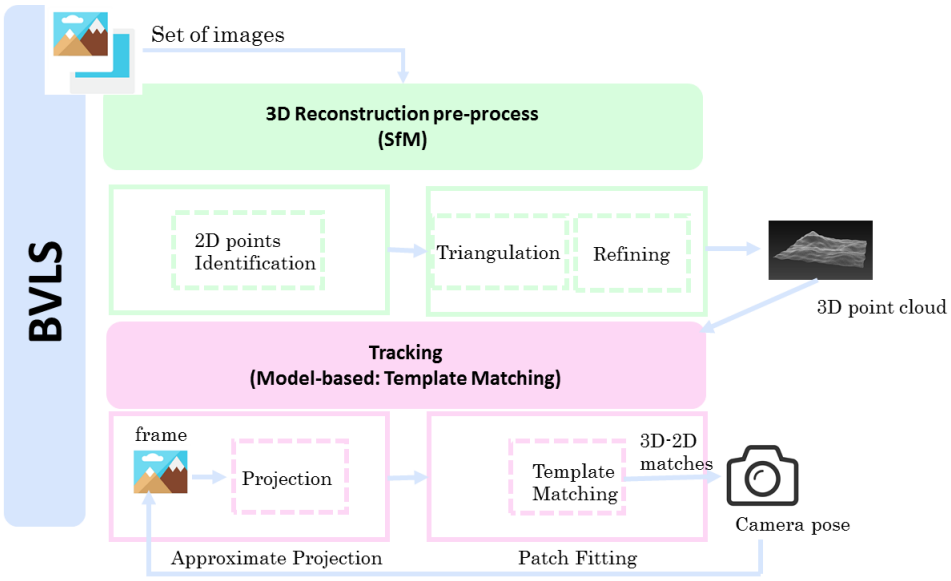


Figure 3.9: The overview of the detailed benchmark visual localisation system

3.3.1 *3D reconstruction pre-process*

In this section the development of *3D reconstruction pre-process* of BVLS is explained, step by step.

2D points Identification In order to identify the features or 2D points in the *keyFrames*, SIFT detector (explained in Section 2.3.2.2) is used . SIFT descriptors are also used to identify the characteristics of the points or features. The 3D point cloud will be reconstructed from these corresponding points. SIFT is computationally expensive, but it is quite robust. As this process is carried out as a pre-process the computational cost is acceptable.

In order to match the identified features, matching must be done. Fast Approximate Nearest Neighbour With Automatic Algorithm Configuration library (FLANN, from now on)[43] is used with this purpose.

FLANN contains a collection of algorithms optimised for fast nearest neighbour search in large datasets and for high dimensional features. It uses Kd-Tree[14] for perform the searching process faster.

Triangulation In order to perform a triangulation process, the first two *keyFrames* are used initially. Epipolar geometry is applied in order to get

the **F** matrix (Section 3.1.2). Then, **E** matrix is estimated with a Singular value Decomposition (from now on SVD) [25]. Finally the camera matrix, $[\mathbf{R}|\vec{t}]$, of these two *keyFrames* are obtained from **E**.

Triangulation is done then, using camera matrices of those *keyframes* and their 2D points correspondences. The first 3D point cloud is obtained.

It is known the 2D pairs from which the 3D point cloud has been estimated. It is also know their corresponding 2D points in the rest of the *keyFrames*. Consequently, it is know the 3D-2D matches for all the *keyFrames*. So, using these matches and applying DLT algorithm the camera matrices of the all *keyFrames*, which have not been used to triangulation, are also obtained.

Finally, to finish with triangulation process, as all the camera matrices are known, the 3D point cloud is extended adding new points. These new points are visible in at least two consecutive *keyFrames* and they were not obtained in the first estimation, because they were not visible in the first two *keyFrames*.

Refining The 3D point cloud has been estimated, and also the camera matrices of all the *keyFrames*. However, this estimations contain errors. Usually a precise estimation is not achieved. For this reason, an adjustment must be done. The *keyFrames* pose and the 3D point cloud must be improved.

With this purpose a BA algorithm is applied. BA is a minimisation algorithm which finds the camera pose and 3D points minimising the reprojection error between the observed and predicted image points.

In BVLS, BA is carried out using Levenberg-Marquart minimisation algorithm (LM from now on) algorithm [40]. This algorithm is applied to the reprojection error function shown in Equation 3.3.

The LM algorithm is an iterative technique that locates a local minimum of a multivariate function that is expressed as the sum of squares of several non-linear, real-valued functions.

3.3.2 *Tracking*

This section is focused on explain step by step the *tracking* task of BVLS.

Approximate Projection The 3D-2D match approximation must be done in this process. This approximation is done by a simply projection of the 3D point cloud in the current frame. However, the camera matrices of the frames or images that coming are unknown.

Assuming that the distance between consecutive images or frames is very small, due to images are got in real-time from a video sequence, the previous camera matrix is used in order to project the 3D point cloud in the current frame. It is done for all the frames except the first one.

Initialisation, estimates the camera matrix of the first frame using feature detector and matching techniques [1].

Patch Fitting In order to improve the 3D point projection accuracy, adjustment is done (see 3.2.2). Template matching techniques are used for this purpose. A patch in the *keyFrame* were this 3D point was firstly saw is used as *template image*. In order to make the comparison between *template image* and *source image*, normalised cross-correlation is used (it will be explained in Chapter 4).

Applying template matching to all the projections, a 3D-2D matches are obtained, creating an equation system which will be used in DLT algorithm to estimate the current camera pose.

[1] FAST[43] detector and FREAK[3] descriptor are used in the Initialisation process. Then, matching is done, using FLANN, with all the *keyFrames* used in the *3D reconstruction pre-process*. The *keyFrame* with more matches is chosen, and those matches are used to apply DLT and get the matrix pose. As a consequence, the first image will not need the next step (Patch Fitting).

Chapter 4

Warped Template Patch Tracking algorithm (WaPT)

This chapter describes the algorithm proposed in this thesis. First section explains the distortion problem. The second section explains the algorithm to palliate this problem. The algorithm is named WaPT (Warped Template Patch Tracking).

The third section defines the architecture of the perspective compensation system (PCS from now on). It integrates the proposed algorithm (WaPT). PCS will be an extension of BVLS, which was explained in the previous chapter (Chapter 3).

Finally, the last section exposes launched concept validation process, and the results obtained, in order to get an approximation of the algorithm and PCS viability.

4.1 Perspective distortion problem

BVLS is a SfM based visual localisation system which uses template matching technique. This system does not take into account the perspective distortion.

Template matching uses patches similarity in order to find 3D point projection into images, and get accurate 3D-2D matches. Nevertheless, rotation of the camera during the user movement affects this similarity measurement, and consequently the accuracy of the 3D-2D matches. This problem is denominated perspective distortion.

In order to understand better how the template matching works the following section explains how the similarity is measured. Then, the per-

spective distortion problem is exposed.

4.1.1 Similarity measurement

Template matching is the technique that finds the location of a patch within an image. In this case a *template image* position with the highest similarity is found within *source image*. In order to find that location, the template image is moved within the source image pixel by pixel (from left to right and from up to down) estimating the similarity as it is explained in Figure 4.1.

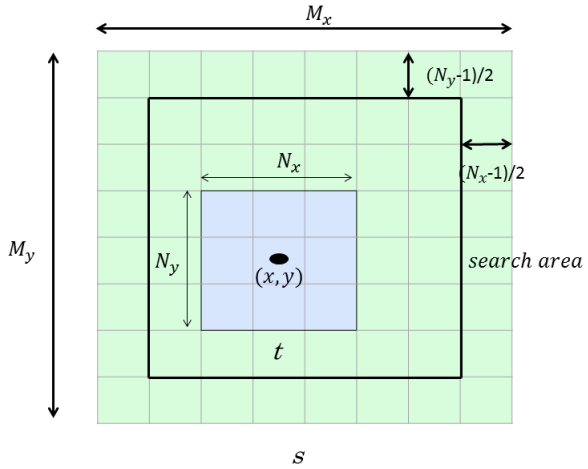


Figure 4.1: Source image s of size $M_x \times M_y$ and template image t of size $N_x \times N_y$ and how the last one is moving around the first one measuring the similarity.

But, how is estimated this similarity? How can be computed? Two main methods are distinguished with that purpose, cross-correlation and normalised cross-correlation.

4.1.1.1 Cross-correlation

Cross-correlation is a measure of similarity or relationship between two signals. Focusing in the field of computer vision or image processing, the goal is to compare or measure the similarity of two vectors (images).

This measure is the Sum of Squared Differences (SSD from now on) defined by Equation 4.1, where: (i) s is the source image and t the template one, (ii) $p(i)$ is the pixel $i = (x, y)$ and (iii) $u = (u, v)$ the displacement done

in the image or the position of the template in the source image. Hence, the goal is to find the u that minimise the value of $E(u)$.

$$E(u) = \sum_i (s(p_i + u) - t(p_i))^2 \tag{4.1}$$

Expanding mentioned equation (see Equation 4.2) the term of cross-correlation appears ($2\sum_i s(p_i + u)t(p_i)$). When template image, t, matches with source image, s, the value of this term is high.

$$\begin{aligned} (a-b)^2 &= a^2 - 2ab + b^2 \\ E(u) &= \sum_i s(p_i + u)^2 - 2\sum_i s(p_i + u)t(p_i) + \sum_i t(p_i)^2 \end{aligned} \tag{4.2}$$

If defined cross-correlation is expressed in a vector representation, it can be defined as a dot product of vectors s and t (see Equation 4.3), where $c(x, y)$ is the correlation score.

$$\begin{aligned} c(x,y) &= \sum_{u=-N_x/2}^{N_x} \sum_{v=-N_y/2}^{N_y} t(u,v)s(x+u, y+v) \\ &\quad t(x,y) \otimes s(x,y) \end{aligned} \tag{4.3}$$

However, doing all these calculations from the pixel intensity values has a specific problem: it is biased by changes in global brightness of the images. Brightening of an image may shoot up its cross-correlation with another image, even if the second image is not similar at all [45].

4.1.1.2 Normalised cross-correlation

Normalised cross-correlation (from now on NCC) is an enhanced version of the cross-correlation method which introduces two improvements. First, as was mentioned previously cross-correlation does not take into account the illumination changes. However, with NCC this weakness is eliminated. The results obtained using NCC are invariant to the global brightness changes, i.e. consistent brightening or darkening of either image has no effect on the result. Second, the final correlation value is scaled to $[-1, 1]$ range, so 1.0 value indicates perfect match, and -1.0 indicates perfectly opposite.

NCC eliminates the illumination problem by subtracting the mean image brightness from each pixel value. Equation 4.4 shows how to eliminate the brightness mean value.

$$c(x,y) = \frac{\sum_{u,v}(t(u,v) - \overline{t})(s(x+u, y+v) - \overline{s})}{\sqrt{\sum_{u,v}(t(u,v) - \overline{t})^2 (s(x+u, y+v) - \overline{s})^2}} \tag{4.4}$$

In Equation 4.4 $\overline{s}$ denotes the mean value of $s(x,y)$ within the area of the template image t shifted by (u,v) which is calculated as in Equation 4.5.

$$\overline{s} = \frac{1}{N_x N_y} \sum_{u}^{u+N_x-1} \sum_{v}^{v+N_y-1} s(x,y) \tag{4.5}$$

4.1.2 Perspective distortion

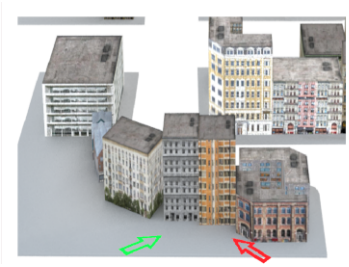
Section 3.2.2 describes the Patch Fitting algorithm briefly. This algorithm is used to increase the precision of 3D point projections within an image. This algorithm knows an approximate projection of a 3D point into the image. This point has associated an small patch. This patch was computed in pre-process. The patch was obtained projecting the 3D point into one of the *keyFrames*. A *keyFrame* is the image of a camera whose pose is known. When the current image and the *keyFrame* have different orientations, generated patches are quite different. So, the Patch Fitting algorithm behaves poorly. Figure 4.2 shows that behaviour.

In Figure 4.2(a) a city can be seen, as well as two camera poses. It is assumed that the camera motion is done from the first camera pose (green one) to the second camera pose (red one). The images generated by those cameras are seen in Figure 4.2(b). In the same figure can be observed how the same point in the environment, and a patch around it, looks like from each camera. The appearance of generated patches is different (Figure 4.2(c)). There are enough different for Patch Fitting to fail in the search for similarity.

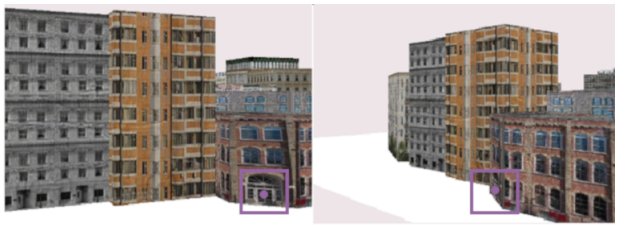
4.2 WaPT algorithm

In order to palliate perspective distortions, this thesis presents an algorithm denominated WaPT (Warped Template Patch Tracking). It generates an accurate warped patch improving the similarity measurement between patches. Consequently, the template matching process is more precise.

WaPT extends the internal representation of the real scene: each 3D point will have a normal vector that approximates the orientation of the real surface where it is placed on. WaPT uses this orientation in order to



(a) Pose of cameras in the environment



(b) Image generated by the first camera pose (green one) and the second camera pose (red one)

(c) Patches around the same point obtained from the images of the first and second camera

Figure 4.2: Distortion problem

warp the patch and improve the template matching process. From onwards, the 3D point orientation means the normal to its surface in the model.

In order to understand better the algorithm, it is important to assume the following definitions:

- Reference *keyFrame*: a *keyFrame* where the 3D point was seen firstly.
- Transferred patch: a square patch defined around the projected 3D point into a frame (*source image*).
- Reference patch: the patch obtained when a given Transferred patch is transferred to the Reference *keyFrame*, the warped patch. It will be used to define the *template image*.

The process to obtain a Reference patch is the following one: given any 3D point, its projection onto the current frame is used to define a Transferred patch. Then, this Transferred patch is back-projected onto the

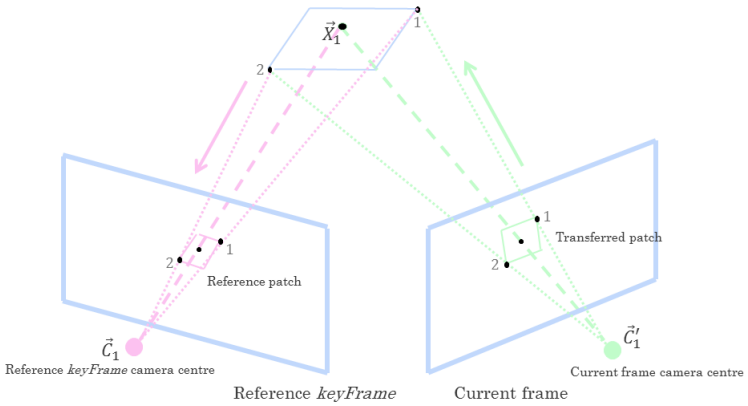


Figure 4.3: Transfer process; back-projection of two points from the Transferred patch onto the Reference *keyFrame* using a plane associated with a 3D point

plane associated to the 3D point. This new polygon is projected onto the Reference *keyFrame* generating the Reference patch (see Figure 4.3).

In order to make it possible, two steps are identified: first the associate plane must be got, and then the patch transferring. WaPT also integrates a searching process when template matching is applied.

To sum up, WaPT algorithm is defined by three steps: feature plane, transfer the patch and find adjusted point. The following sections will explain each step.

4.2.1 Feature Plane

In the projective space a plane is defined as shown in Equation 4.6 [25]. In order to get it, three points $(\vec{X}_1, \vec{X}_2, \vec{X}_3)$ lying on the plane are needed. $\vec{X}_1$ is used to represent the given 3D point. The other two points ($\vec{X}_2$ and $\vec{X}_3$) are calculated using the normal vector and the restriction imposed by Equation 4.7, where $(a, b, c)^T$ is the normal and $(x, y, z)^T$ the 3D point in the euclidean space. Take into account that d is defined by Equation 4.8. This equation presents singularities when the normal is aligned with one of the main axis.

$$\vec{\pi} = \begin{pmatrix} (\vec{\hat{X}}_1 - \vec{\hat{X}}_2) \times (\vec{\hat{X}}_2 - \vec{\hat{X}}_3) \\ -\vec{\hat{X}}_3^T (\vec{\hat{X}}_1 \times \vec{\hat{X}}_2) \end{pmatrix} \tag{4.6}$$

$$ax + by + cz + d = 0 \tag{4.7}$$

$$d = -(ax_1 + by_1 + cz_1) \tag{4.8}$$

4.2.2 Transfer the patch

Each pixel of the Transferred patch is back-projected onto the plane,i.e. each pixel of the Transferred patch defines a point as the intersection of its back-projected ray and the plane. Then, these points are projected to the Reference *keyFrame*. The process can be seen in Figure 4.3.

The back-projection of the point in the image is defined by Equation 4.9, where P^+ is the Moore-Penrose pseudo-inverse projection matrix of the frame and $\vec{C}$ is the center of the camera in the global reference system. In addition, a point which lies in a plane must fulfil Equation 4.10. Solving this equation system the points are obtained. Then, the projections of these points in the Reference *keyFrame* are done, in order to obtain the Reference patch.

$$\vec{X} = P^+ * \vec{x} + \lambda + \vec{C} \tag{4.9}$$

$$\pi^T + \vec{X} = 0 \tag{4.10}$$

The steps defined are necessary in order to warp the patch and take into account the perspective deformation generated by the camera motion.

However, it must be remembered that for BVLS warped process is not done. The Reference patch in that case is established as follows: the projection of the 3D point is done also in its Reference *keyFrame*, and a window around that projection is defined. That window has the same size, as the Transferred patch.

So, once the 3D points are projected in the current frame, the Reference patch is obtained. This patch is used in the template matching process: finding the patch in the current image with the highest similarity to the Reference patch. The aim of this search process is to adjust the projection of the 3D point and obtain an improved 3D-2D match (see Section 3.3.2).

4.2.3 Find Adjusted Point

With mentioned purpose, the NCC coefficient is applied into an established search area. The search area is a fixed size window W within the current image. The projection of the 3D point is its center.

In order to achieve more accuracy in the search process a three level pyramidal reduction is applied. In this way:

- A three level reduction of W is calculated obtaining W_{L0},W_{L1} and W_{L2} where $W = W_{L0}$.

- I is the original image, i.e the current image.

- A three level pyramidal reduction is also calculated for the Reference patch R obtaining R_{L0}, R_{L1} and R_{L2} where $R = R_{L0}$.

Notice that R_{L0}, R_{L1} and R_{L2} are the same image with different resolutions. In order to get robustness against noise the patches that will be used in the search process are fixed size regions placed in the center of R_{L0}, R_{L1} and R_{L2}.

Once the pyramidal levels are defined an iterative process is run. In this iterative process the match uses firstly the lowest pyramidal levels (W_{L2} and R_{L2}).Then, the result is propagated to the higher levels (seen Figure 4.4).

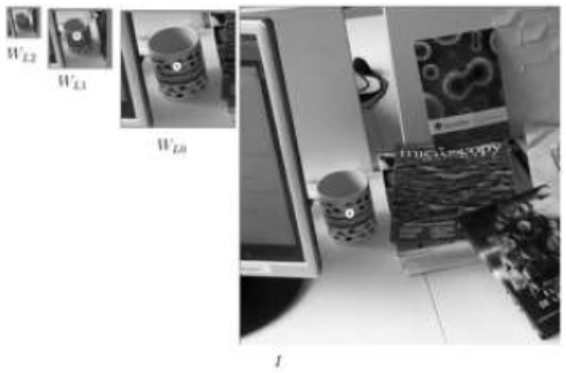


Figure 4.4: Iterative process where the similarity of the reference patches in different pyramidal levels is calculated. The propagation of the adjusted feature from the lowest pyramidal level to the current image can be seen as a result of the process.

So, firstly R_{L2} is found in W_{L2} using the cross correlation coefficient and the 8×8 region places in the center of R_{L2}. The function returns a position in W_{L2} of the region with the highest similarity to R_{L2}. This position is propagated to the next level, i.e W_{L1}.The search area of W_{L1} is redefined taking into account the position calculated in the previous step, obtaining W_{L0} , which is smaller than W_{L1}. The processes are repeated and R_{L1} is found in W_{L0} .

This iteration process is repeated until the position is propagated to the final level, W_{L0}, and finally to the original image I. In Figure 4.4 the propagation of the point from different levels can be seen.

This process is done for the n points, obtaining a set with n features that will be used to find the extrinsic parameters of the current camera, using DLT method.

4.3 Perspective compensation system

Perspective compensation system is the proposed system in this thesis which involves WaPT algorithm. It is an extension of BVLS. So, taking the architecture of BVLS as a basis, the architecture of the PCS is defined (Figure 4.5).

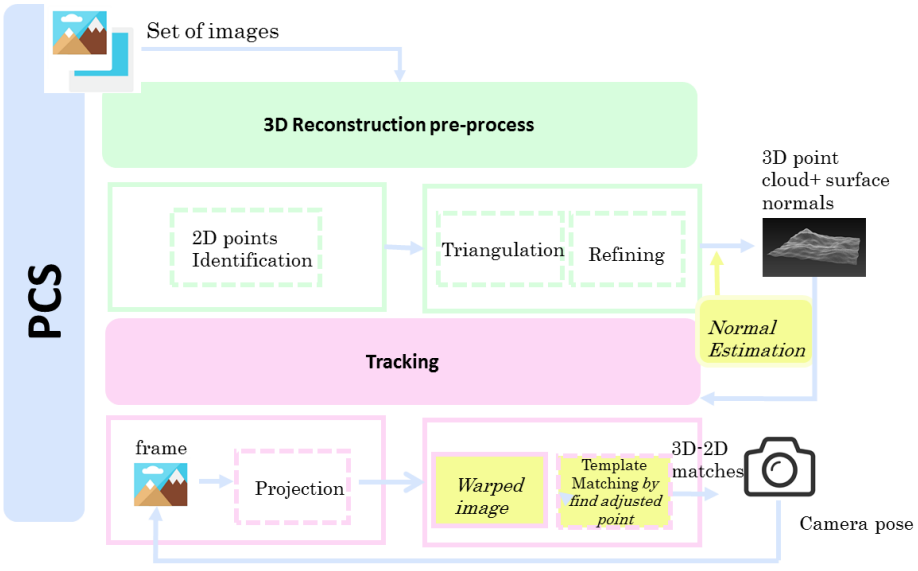


Figure 4.5: The PCS detailed architecture

4.3.1 *Tracking*

System Overview 4.1: *Tracking*

Goal To know the pose in the real world of the user or camera in real time

Input

- `Im`: images captured by the camera in real time
- `PC`: 3D point cloud obtained in the *3D reconstruction pre-process*
- `PCn`: normal vector for each point of the 3D point cloud
- `KF`: *keyFrames* camera pose matrices

Output

- $[\mathbf{R}|\vec{t}]$: Camera pose for each image

WaPT is applied in *tracking*, exactly in Patch Fitting. A new module denominate Warped Image is shown in the architecture of PCS (Figure 4.5). This is where the patch is warped , i.e where feature plane and transfer the patch are applied. In addition, Figure 4.5 shows how the template matching is applied using a searching process defined by WaPT (find adjusted point).

The System Overview 4.1 shows in a resumed way, the input and output data of this task. Take notice that the normal vectors, associated to each 3D point of the point cloud, are given to this stage. These normals are used by WaPT to warp the patch, to apply perspective compensation.

4.3.2 *3D reconstruction pre-process*

System Overview 4.2: *3D reconstruction pre-process*

Goal: To obtain the 3D point cloud which will work as a map of the environment, as well as the orientation of each point
Input:

- `Im`: set of environment images

Output

- `PC`: 3D point cloud
- `PCn`: normal vector for each point of the 3D point cloud
- `KF`: *keyFrames* camera pose matrices

To be able to apply perspective compensation in *tracking*, the orientation of each 3D point has to be known. In this way, *3D reconstruction pre-process* will provide them. A new module is added to this purpose (see Figure 4.5). The objective of this module is to estimate the normal vector of each 3D point. This vector defines the orientation of the 3D point.

Consequently, as a result of this task executed in pre-process, it is not only the 3D point cloud, it is also the surface normal vector for each point of the 3D point cloud (see System Overview 4.2).

4.4 Concept validation

This section provides the initial research steps in this thesis. Once the hypothesis has been stated, the next step was to make a concept validation of the algorithm. These results were published in 2015 [11]. The formal validation of the thesis hypothesis was published in 2018 [12].

As previously commented, in between, other authors provided more results that support the hypothesis [17][55]. However, neither of them provided a formal demonstration of the hypothesis proved in this thesis.

The aim is to analyse similarity between patches. So, Transferred patch and Reference patch similarity is measured for two situations: (i) when WaPT is applied, i.e the Reference patch is obtained by warping and (ii) when WaPT is not applied.

In first place, this section explains how the surface normal is computed

for each 3D point[1]. Section 4.4.2 explains the experiments. The last section analyses the results.

4.4.1 Normal estimation algorithm

The goal of this algorithm is to estimate the normal for each 3D point of the 3D point cloud obtained by SfM. To achieve this objective a minimisation process has been designed and implemented. This algorithm will be performance in *3D Reconstruction pre-process* of the PCS (Figure 4.5).

In order to understand this algorithm is important to remember how WaPT obtains the Reference patch (Section 4.2) using a normal vector. Figure 4.3 shows this process.

The main idea of the this algorithm is to follow the same process in order to obtain the property surface normal: given any 3D point, its projection onto a *keyFrame* is used to define a Transferred patch. Then, this Transferred patch is back-projected onto the plane associated to the 3D point. This new polygon is projected into Reference *keyFrame* generating Reference patch (see Figure 4.3).

Nevertheless, how can be this process done if the normal vector is unknown? What is the meaning of this process, if it is precisely the normal vector which we want to estimate? It is here, where the minimisation process makes sense. The objective of the minimisation algorithm is to find the normal vector that minimises the image difference between the two image patches. So the process is launched in a minimisation process.

This work uses the LM [40]. This algorithm is an iterative process, where a random normal is taken as the initial guess. The plane defined by the normal is then used to transfer the Transferred patch to the Reference *keyFrame*. Afterwards the algorithm evaluates the difference between the Transferred patch and the Reference patch.

The minimisation algorithm adjusts the normal so that the difference between all the Transferred patches with the Reference patch is the smallest. The algorithm exits when the difference between the last iteration and the current one does not exceed a fixed threshold.

How to evaluate the difference between patches? The NCC coefficient is used. The NCC coefficient is a measure of similarity of two images, where a perfect match will be 1 and a perfect mismatch will be -1. Deeply explanation of NCC is done in Section 4.1.1.2.

[1] Chapter 3 explains how the 3D point cloud is computed using conventional algorithms. Using SfM algorithm most concretely.

WaPT algorithm looks for the highest similarity between Reference and Transferred patches. So, the value of the NCC coefficient has to be as high as possible. However, notice that this value is used in a minimisation process. For this reason the objective function shown in Equation 4.11 is used to evaluate the difference between patches, being $crossCorrelation_i$ the NCC coefficient of the patch applied in frame i, and (α, β) the two angles used in the parameterisation of a normal.

$$min(\alpha, \beta) \sum_i (1 - crossCorrelation_i(\alpha, \beta)) \tag{4.11}$$

4.4.2 Experiments

In order to performance mentioned concept validation, some experiments must be carried out. PCS (Section 4.3) and BVLS (Section 3.3) will be launched for this purpose.

3D Reconstruction pre-process is launched as explained in Section 3.3.1 for both systems. However, PCS also applies the Normal estimation algorithm, just after 3D point cloud is obtained, as exposed in Figure 4.5.

Then, *tracking* tasks will be launched for both systems. The performance of Patch Fitting algorithm is different for each system. For BVLS Patch Fitting is carried out as explained in Section 3.3.2. Nevertheless, in PCS WaPT is applied at this point.

It has to be remembered that to measure the patches similarity is the goal of this experiment. It will evaluate the viability of the WaPT algorithm and PCS. For this purpose, for each current image when Patch Fitting is carried out and patches obtained, their appearance will be measured using NCC for both systems.

4.4.3 Results

The results of WaPT (implemented in PCS) are compared to results obtained by BVLS, which assumes that the orientation of the 3D points always faces the camera.

With this purpose, an indoor video sequence was recorded where a camera motion around the same work place is visualised. The video sequence is built by approximately 70 frames with 620×480 resolution.

As mentioned previously, in order to evaluate the viability of WaPT algorithm the similarity between patches is calculated.

In the first experiment, the similarity of the Reference patch and the Transferred patch is calculated for $n = 50$ points. The average of the NCC

coefficient of all the points for each frame is calculated. Figure 4.6 shows the evolution of these averages during the video sequence: PCS is represented with a dark blue line whereas the approach used by BVLS is represented with a light blue line.

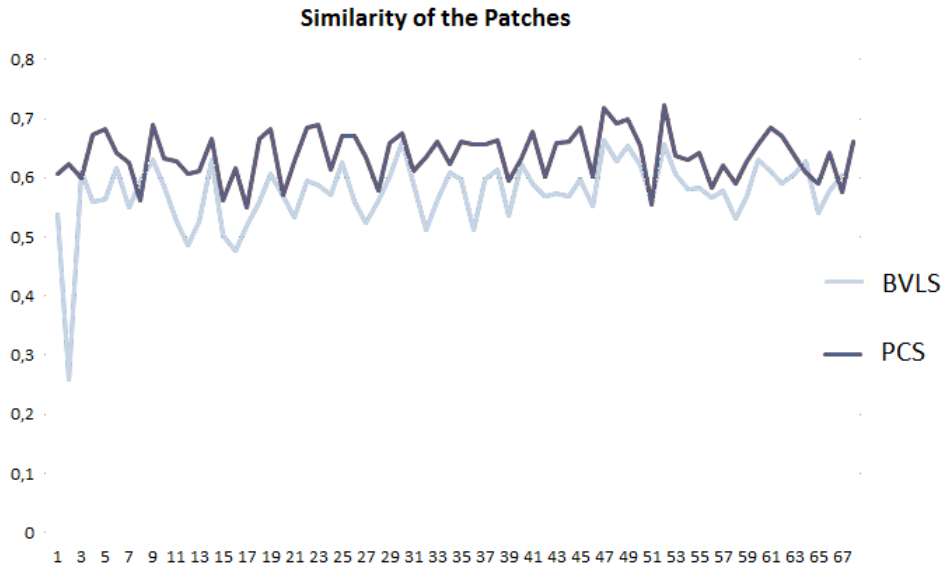


Figure 4.6: Evolution of the NCC average values

The average NCC coefficient value for PCS algorithm is 0.639 and in the case of the approach used by BVLS is 0.574, i.e the similarity of the patches is higher in PCS than in BVLS.

On the other hand, for each frame the median and the quartiles of the NCC coefficient values are calculated. These statistics are used to measure the stability of both algorithms. Figure 4.7 shows the box-plots for the first 10 frames for PCS and Figure 4.8 depicts the same information for BVLS.

With the aim of making the comparison visually simpler, only the first 10 frames are provided. These 10 frames are representative of the performance of the systems.

4.4.4 Discussion

The first discussion point is the difference between the average NCC coefficient values appreciated in Figure 4.6. As it can be seen in the figure, the WaPT algorithm, i.e PCS obtains higher NCC coefficient values during the

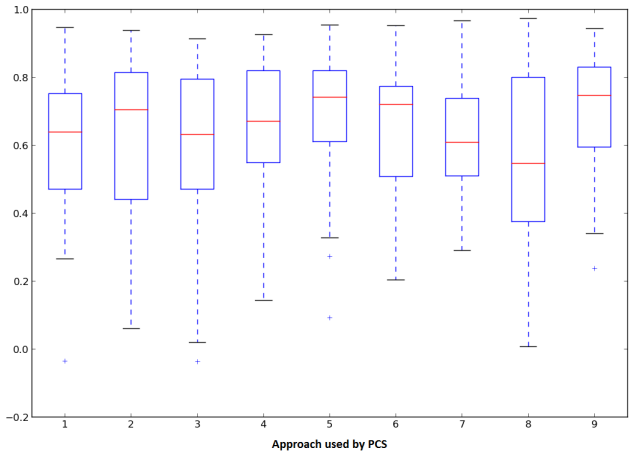


Figure 4.7: Box-plots of the NCC coefficient values in the first 10 frames. PCS system.

video sequence.

These values define the similarity between images, and have to be as close as possible to 1. Therefore, applying WaPT algorithm the template matching process is more accurate due to the similarity of the patches, i.e. WaPT algorithm obtains more similar patches than approach used by BVLS. Furthermore, the average of NCC coefficient values in both algorithms confirms that statement.

The average value for PCS $(0,639)$ is 11% higher than the value obtained for the approach used by BVLS $(0,574)$. It means that taking into account that similarity is measured in range $[-1,1]$, PCS lacks a 18% to achieve a perfect match while approach used by BVLS lacks a 29%.

The graphics shown in Figures 4.7 and 4.8 represent the distribution of a continuous variable (NCC coefficient) for both algorithms respectively. The 3rd quartiles in the PCS are higher than in the case of approach used by BVLS.

Regarding the median values, the same trend is observed. The median values in PCS are higher than in BVLS. As a general trend, it is appreciated that when WaPT is applied more similar patches are got.

The box-plots graphics demonstrate it; the 3rd quartiles, the median values and even the minimum values are higher in the case of PCS. All exposed data confirms that, in this test, the WaPT algorithm improves the similarity between patches.

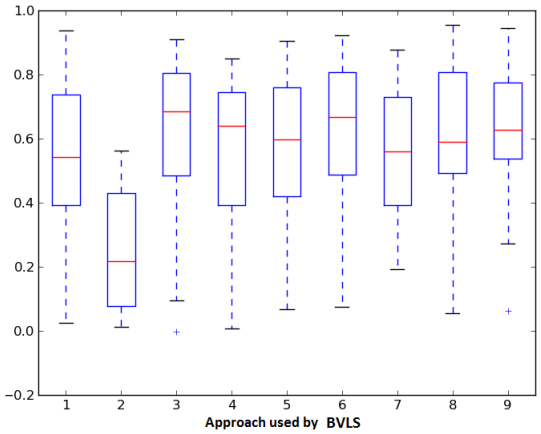


Figure 4.8: Box-plots of the NCC coefficient values in the first 10 frames. BVLS system

The results are promising, the accuracy in the template matching process should be improved as a consequence of applying WaPT due to the highest similarity between patches. Consequently, more formal experiments should be done in order to prove that statement. The next chapter will be focused in that purpose.

Chapter 5

Objective verification of the hypothesis

Model-based localisation systems rely on matching 3D points, previously identified in an environment, with their corresponding 2D projections measured in each frame captured by the camera. That is, given a set of 3D points in the model, their projections into an image must be found. The number of correct matches found is the number of inliers that will be used to solve an equation system which calculates the camera pose. So, there are two main facts that spoil the stability of the camera pose estimation: (i) a small number of inliers, or (ii) most of the inliers appear concentrated in a small region of the frame.

BVLS (Section 3.3) uses template matching technique in order to obtain the 3D-2D matches. This technique is fast in computational terms and does not tend to drift. Even more, it is robust against illumination changes. Nevertheless, it is not robust against perspective distortions created by camera motion. Consequently, the method may fail to obtain 3D-2D correspondences affecting the stability of the tracking process.

The main goal of the research presented in this thesis, is to prove, in an objective way, that considering 3D point orientation more accurate matches are obtained, increasing the number of inliers.

The benefits provided by considering 3D point orientation have been reported (see Section 2.4). However, they have not been formaly demostrated in the literature the literature.

In this thesis, the results obtained are evaluated using statistical hypothesis testing (statistical inference) methods. Thereby, it is ensured that results are analysed in an objective way and that they are not generated by

chance.

This chapter is divided into four sections. In the first one, basis of statistical hypothesis testing methods are explained. Then, the experimental set-up is defined. Later, the environment developed to carry out the experiments is explained. It is based on PCS. Finally, the results obtained by the experiments are exposed.

5.1 Statistical Inference

Statistical inference pretends to affirm or make inferences about a population from the results obtained from a sample or samples of that population. It means that some tests are done to the sample, and the results obtained are extrapolated to the population.

For example, let us consider that we want to know how good (from 1 to 10) qualifies on average the population of a country the president. On the other hand, we want also to know if old people (those over 50 years) qualify higher than young people (those under 50) on average.

It is very expensive (in time and resources) to ask entire population one by one. So, instead of ask to everybody, a sample (or samples) of this population is used. The sample must be representative. The average value obtained using the sample, can be affirmed if the sample is representative. So, it is needed to prove that the sample represents the whole population.

Statistical hypothesis is an affirmation about a specific characteristic (parameter) of a population, i.e the average value assigned by people over 50 is higher than the average value assigned by young people (under 50). On the other hand, **hypothesis contrast** is the tool used by statistical inference to judge if a property that is assumed in a statistical population is compatible with what is observed in a sample of that population. So, it is a procedure by which it is decided if the statistical hypothesis should be maintained or rejected [53].

In this way, how the statistical hypothesis should be presented? At this point, two important concepts emerge to be taken into account:

- **Null Hypothesis, H_0:** It is which affirms that the difference between the true value of the parameter and its hypothetical value is due to chance.
- **Alternative Hypothesis, H_α:** It is the complementary one. The rejection of H_0 implies the acceptance of H_α. It is usually named as experimental hypothesis.

The following steps must be taken in order to accept the hypothesis or reject it. Then, the explanation of each step is done.

1. Define hypothesis
2. Define significance level
3. Verify hypothesis
4. Make a decision

Define hypothesis The null hypothesis and the alternative one must be defined. With this purpose, it is important to know the different types of hypothesis that are defined in the literature [53]. The parameter (of the population) to be evaluated is known as θ (for example, the average value which the president is qualified). On the other hand, it is known θ_0 as the value that can be assigned to θ.

It is known as simple hypothesis, when the null hypothesis or alternative one assign an unique value (θ_0) to the parameter, i.e to θ. The correct notation is as in Equation 5.1. It means that the null hypothesis is: the population parameter θ value is θ_0.

$$H_0 : \theta = \theta_0 \tag{5.1}$$

Compound hypothesis, is that which will be true for more than one population parameter value. A range of values is defined for that parameter. Returning to the example, let us define the null hypothesis as follow: the average value with which the president has been qualified by the population is at least 5. The hypothesis is true to any qualification higher than 5.

Usually it is contrasted a simple null hypothesis ($H_0 : \theta = \theta_0$) against a compounded alternative one. For example, it would be wanted to contrast this simple null hypothesis against the alternative one which affirms that the actual value of θ is higher than θ_0 (see Equation 5.2).

$$\begin{aligned} H_0 &: \theta = \theta_0 \\ H_\alpha &: \theta > \theta_0 \end{aligned} \tag{5.2}$$

It can be also that the alternative one declares that the actual value of θ is less than θ_0 as in Equation (5.3). These kinds of hypotheses are denominated, unilateral.

$$\begin{aligned} H_0 &: \theta = \theta_0 \\ H_\alpha &: \theta < \theta_0 \end{aligned} \tag{5.3}$$

It is also possible to contrast exposed simple null hypothesis to alternative one, which declares that the value of θ is anything other than θ_0 (see Equation 5.4). It is know as bilateral alternative.

$$\begin{aligned} H_0 &: \theta = \theta_0 \\ H_\alpha &: \theta \neq \theta_0 \end{aligned} \tag{5.4}$$

To sum up, in Equation 5.5 are defined all the combinations of null and alternative hypothesis possibles.

$$\begin{aligned} &H_0 : \theta = \theta_0 \ vs \ H_\alpha : \theta > \theta_0 \\ &H_0 : \theta = \theta_0 \ vs \ H_\alpha : \theta < \theta_0 \\ &H_0 : \theta = \theta_0 \ vs \ H_\alpha : \theta \neq \theta_0 \\ &H_0 : \theta \leq \theta_0 \ vs \ H_\alpha : \theta > \theta_0 \\ &H_0 : \theta \geq \theta_0 \ vs \ H_\alpha : \theta < \theta_0 \end{aligned} \tag{5.5}$$

According to these possibilities and the characteristics of the case to be analysed, the null and alternative hypothesis should be defined.

Define significance level Once the hypotheses are established, it is important to know with what level of certainty are they going to be rejected or accepted. The significance level, which is denoted as α, is the probability of rejecting the null hypothesis when it is true. Error type I and Error type II are defined as:

- **Error type I:** The null hypothesis (H_0) is rejected when it actually is true.
- **Error type II:** Accept the null hypothesis (H_0) when it is not true.

So, the significance level (α) is the probability to Error type I happen. We can denote as, $P(ErrorTypeI) = \alpha$ or $P(Reject\ H_0|H_o\ is\ true) = \alpha$. Since the null hypothesis must be rejected or accepted, it is denoted the probability of accept the H_o when it is true as, $P(accept\ H_0|H_0\ is\ true) = 1 - \alpha$.

On the other hand, Error type II is denoted as follow: $P(ErrortypeII) = \beta$, i.e $P(Accept\ H_0|H_0\ is\ false) = \beta$. Consequently, the probability of reject the null hypothesis when it is false is expressed as: $P(Reject\ H_0|H_0\ is\ false) = 1 - \beta$.

Two other concepts appear together with those: rejected area and acceptance area. Rejected area groups all these values which are getting away from H_0 (because they are very big or very small) that it is very unlikely to happen when H_0 is true. It means, this zone is associated to significance level (α). Usually, in scientific field, this probability is established in 0.01 or 0.05 [53].

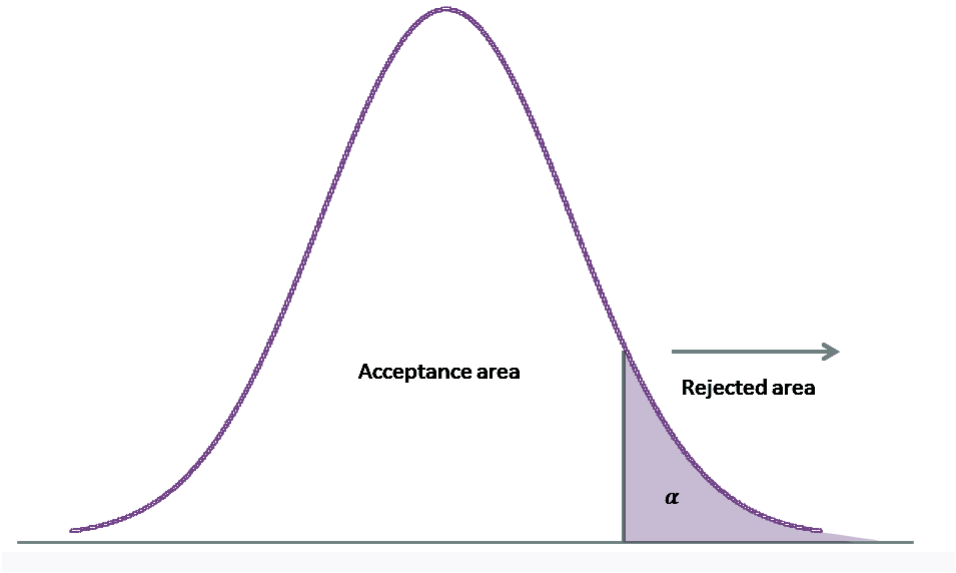
Figure 5.1(a) and Figure 5.1(b) show both areas (rejected and acceptance) in a contrast statistics defined by the significance level.

Verify hypothesis At this point, the hypotheses to be rejected and accepted (null and alternative one) are known. It is also known with what level of significance are they going to be rejected or accepted. So, how can be verified those hypotheses? extracting a sample whose size has been decided in the previous step and obtaining from it the corresponding statistic. With this purpose different statistical tests, which will be applied to the sample, can be found in the state of the art. Pearson's Chi-squared test and Mann-Whitney U test are the tests used in this thesis in order to reject or accept the hypothesis defined.

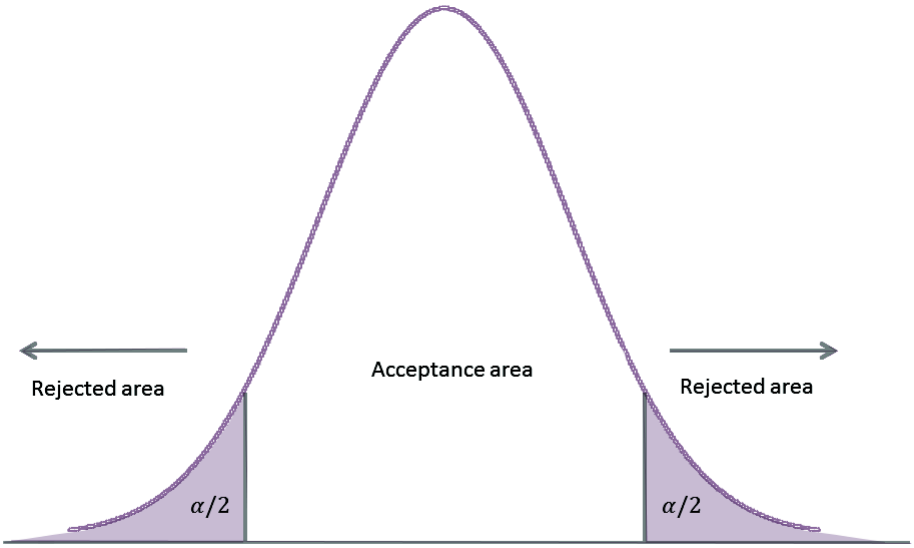
Statistical tests can be divided in two categories: parametric tests and non-parametric tests. Parametric statistical methods usually assume that samples have an specific distribution, often a Gaussian distribution. If the data sample is not a Gaussian, then non-parametric test must be used. So, the first thing to do before applying the statistical test, is to check if the sample data follows a normal distribution or not .

To check that, Pearson's Chi-squared is usually used. It is a statistical test applied to sets of categorical data to evaluate how likely it is that any observed difference between the sets arose by chance. So, basically it is used to analyse categorical data (for example, male or female, normal or not normal) [49] [47].

If Pearson's Chi-squared test confirms that samples do not follow a normal distribution then, U-Mann-Whitney test should be applied [51]. This test is applied to accept (or reject) the hypothesis. This is the case of this research.



(a) Acceptance and rejected area are established according to defined significance level (α). Unilateral Hypothesis



(b) Acceptance and rejected area are established according to defined significance level (α). Bilateral Hypothesis

Figure 5.1: Acceptance and rejected area definition

Make a decision Actually, H_0 is rejected when the probability of occurrence is less than the level of significance. It is denominated critical level to this probability and it is represented as *p-value*. Mentioned statistical test will calculate this *p-value* (p_v) for given samples. Usually , in the state of the art, it is established the *p-value* in: $p_v < 0.01$, $p_v < 0.05$ or $0.01 < p_v < 0.05$. It means that if the *p-value* is less than the significance level established, then the null hypothesis will be rejected.

5.2 Experimental Set-up

Experimental Set-up

- **What to measure?:** The number of inliers
- **For:** BVLS and PCS systems
- **Where?:** in a virtual model of a city
- **How?:** using two different camera motions
  - **Path I:** camera translates only
  - **Path II:** camera translates and rotates during the movement
- **Null hypothesis (to refuse):** BVLS and PCS generate the same number of inliers
- **Alternative hypothesis:** PCS generates more number of inliers than BVLS

The main goal of the research presented in this thesis, is to prove that model-based tracking processes, which use template matching techniques, improve their results when they take into account the perspective distortions generated during the camera motion.

The camera pose estimation accuracy depends on the number of good 3D-2D matches found (inliers). So, the experiments should measure the number of inliers generated by the systems.

With this purpose, the experiments should be carried out under two conditions: (i) taking into account the orientation of the 3D points, i.e PCS system and (ii) completely ignoring it, BVLS system.

In order to ensure that the results (number of inliers) obtained by both

systems (BVLS and PCS) are not generated by chance, the statistical significance of the results has to be calculated. With this purpose and taking into account the statistical inference theory explained in Section 5.1, the null hypothesis and the alternative one have to be defined. The idea is that if there is not any significant difference between both systems, the sample values generated by them for inliers should follow the same probability distribution with the same parameters. More formally, the null hypothesis and the alternative are stated as defined in Equation 5.6.

$$\begin{aligned} H_0 &= E(x_0) - E(x_1) = 0 \\ H_\alpha &= E(x_1) - E(x_0) < 0 \end{aligned} \tag{5.6}$$

where $E(x_0)$ is the average of inliers obtained by PCS and $E(x_1)$ the average of inliers obtained by BVLS. The objective is to prove that the average value of the sample x_0 is higher than the average value of the sample x_1, i.e., the method that takes into account the surface orientation generates more inliers in average. For this reason, the null hypothesis to refuse, is that the average values of the samples are the same.

The set-up of the experiments should allow to control all the variables, specially those which have an effect on the results. It should be ensured that the results obtained by each system in the experiments, is a direct consequence of using the 3D points orientation (or because the system ignores it). For this reason, the scenario where the systems will be launched must be a controlled one which also serves as a ground truth.

When a map of a 3D real environment is created, it is difficult to guaranty its precision. On the other hand, creating map of a 3D virtual environment can be done with high precision.

Therefore, instead of using a model of a real environment in which localise the camera, the experiments use a synthetic one. Experiments are performed in a virtual model using virtual cameras. A model of a virtual city neighbourhood has been used. It is shown in Figure 5.2. From now onwards the model will be denominated 'city'.

Two experiments are defined to analyse results in different ways. The idea is to simulate different camera movements around the environment. In that way, the influence of using 3D points orientation (or not) can be checked for different situations:

- In the first one (Path I), the camera does not rotate, i.e the camera makes a translation movement along the streets.

Figure 5.2: The 3D model of the city used to generate the ground truth

- In Path II, the camera orientation changes smoothly along the trajectory.

5.3 Experiment environment

The experiments are performed using a ground truth. Videos are recorded moving a virtual camera through a virtual city. In this way, all the camera poses are perfectly known before hand. Later on, when BVLS and PCS will compute that poses, they can be compared with the poses that created the video (ground truth).

Figure 5.3 enumerates the modules and processes that constitute the experiment environment. Section 5.3.1 describes the generation of the ground truth. Section 5.3.2 specifies some parameters used along the *tracking* task.

5.3.1 Ground truth generation

The experiments will use a ground truth. This means that all the data used in the *tracking* task is exact: 3D point cloud and normals. Even more, this process generates the image for each step of a camera path. In this way, each camera pose is known.

Next, camera paths generation and ground truth generation are explained. It should be noted that the *3D reconstruction pre-process* differs from that explained in Section 3.3.1 and Section 4.3.2. In this case the 3D

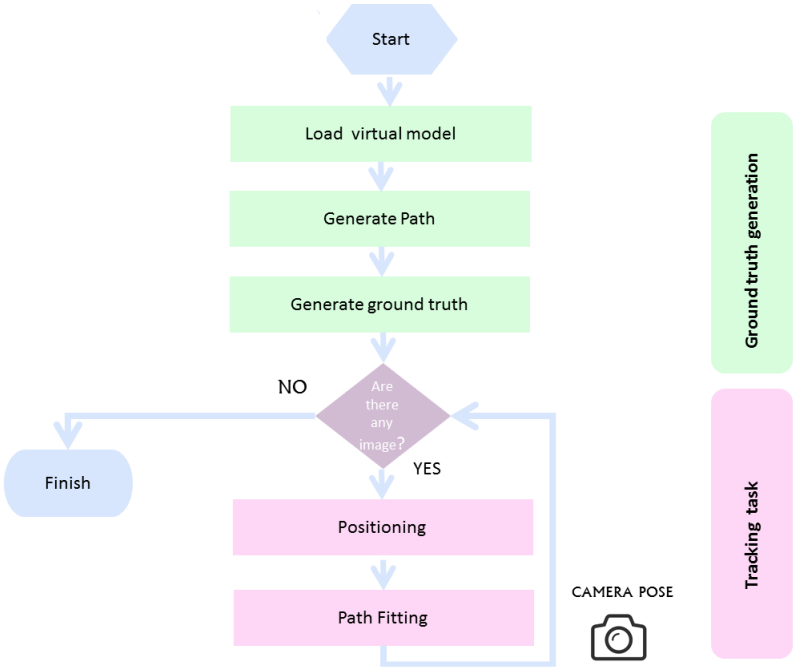


Figure 5.3: The flowchart which details the steps to be followed in order to develop defined experimental set-up

model is known before the camera images are created.

5.3.1.1 Generate camera path

First of all, the virtual model of the city (Figure 5.2) is loaded in the graphic engine. Next step is to simulate a route that the camera follows in that city. With this purpose, the virtual camera is fixed in two places: the initial one, where the route would start, and the final one, where the route would finish. From now on, they will be denominated main cameras.

As explained in Section 5.2 two experiments are defined: Path I and Path II. They represent different ways to make a route. So, positions and orientations of a discrete number of frames ($m = 150$) are computed between main cameras, defining these paths. In order to compute the position of each frame ($m = 150$) between the main frames, a linear interpolation is used. To obtain the orientation, the Slerp coefficient is used. For each frame, the virtual image 'seen' by the virtual camera is computed using the graphics

engine.

5.3.1.2 Generate ground truth

This step obtains all the output data of a *3D reconstruction pre-process*, in a controlled way. It means that, it will compute with total precision the value of each parameter.

The input are all the images got in the previous step, which represent the images captured by the camera. Some of these images are going to be chosen also as *keyFrames*, i.e as the set of images used in order to get the 3D point cloud.

For the experiments here presented, the image captured by the initial camera, is used for estimate the 3D point cloud, i.e there is only one *keyFrame* and it is that obtained by the initial camera. Then, FAST detector [67] is used in order to obtain a set of 2D points in the *keyFrame*. These 2D points are the features within the image. As the position of the camera is known the 3D point on the model that corresponds to each characteristic 2D point can be computed with total precision. A raycasting technique is used for this purpose. For each 3D point, its surface normal is also computed and stored. These 3D points are the 3D representation of the environment, the 3D point cloud.

5.3.2 *Tracking* task

Once the reference model is obtained, the *tracking* tasks has to be launched. This section describes the development details.

It is important to remark that the main difference between PCS and BVLS is the use of a 3D points orientation to warp the patch in Patch Fitting (see Figure 3.9 and Figure 4.5). In other words, PCS applies WaPT algorithm and BVLS does not.

The images which simulate the camera movement into the virtual city are created in the ground truth (Section 5.3.1). Now BVLS and PCS must localise each image: find the camera pose for each image. These algorithms were explained previously. Here some implementation details and parameters will be specified.

5.3.2.1 Approximate projection

First of all, the 3D points of the 3D point cloud should be projected in the current image. It is important to remark that the first image is set as the

keyFrame. It means that the images that will perform the role of images captured by the camera in real-time are all except the first one.

For these experiments, the camera matrix of the image used as *keyFrame* is used to project the 3D points in the first image of this stage.

The 3D point cloud can be very large. In order to get a balance in which, there are enough points to estimate the camera pose matrix and not be expensive in computational terms, a subset of points are projected. With this purpose, n points (in experiments $n = 250$) from the 3D point cloud are randomly chosen as features. The points are chosen uniformly distributed to be sure that there is not a concentration of inliers in a small region of the image.

5.3.2.2 Patch Fitting

In these experiments BVLS and PCS apply the pyramid search process for template matching. Experimentally it was found that a 128×128 (W) search area gives good results in 620×480 images. In addition, patches size was fixed in 32×32 (R).

In order to define the Reference patch, BVLS and PCS take a window around the projection of each 3D point cloud in the Reference *keyFrame*.

5.4 Experiments and Results

Based on the experimental set-up described in Section 5.2, two experiments have been defined to prove the hypothesis (denominated Path I and Path II). The ground truth defined is based on the virtual model of a city. Both experiments use the same model where 3D points (3D point cloud) and their normals are known with precision. In both experiments, the camera movement is simulated fixing an initial and final cameras. Then, the camera is moved from the initial one to the final one differently: for Path I (Section5.4.1) the camera moves from initial position to the final one doing a translation movements only. On the other hand, for Path II (Section 5.4.2) the camera starts in he same initial position and finish in the same final position but moves from one to another translating and rotating, i.e the camera orientation changes smoothly along the trajectory.

All the 3D points in the model will have the same Reference *keyFrame*, the initial frame. These experiments cover two usual situations in tracking systems. The goal is to know whether the tracking systems which compute perspective compensation generate more inliers than tracking systems which do not. In the same way, it is important to know whether this difference is

significant or not, and in which situations an important improvement can be seen.

Before analysing the results, it can be expected that the initial frames in both paths correspond to best computational cases for the methods that do not consider perspective compensations (BVLS). As the camera moves perspective deformations increase. Camera rotations increase even faster perspective deformation problems, as Path II will show.

As was mentioned, the number of inliers generated for both methods, is the key of this work. So it is important to know, which points are considered inliers. Tracking algorithms discriminate 3D–2D matches as inliers or outliers solving equation system used to find camera position. However, in this analysis we use a more precise discrimination method. In this case, the precise position of the camera is known before hand and also the 3D point cloud. So, it can be computed the exact projection of each 3D point onto the current frame. The re-projection error is the difference between the projection found by the tracking system and its exact projection. Those points with a re-projection error less than 10 pixels are considered inliers.

5.4.1 Path I

A virtual model of a city is used. In Figure 5.4(a), two main camera positions can be seen. In order to have a significant number of samples (m), 150 new camera positions are interpolated between these two main cameras. Figure 5.4(b) shows the images obtained by the camera in the first, intermediate and last positions. In the path presented in this section, the camera motion bears translations only.

Then, the tracking process is launched (*tracking* task), for the m frames using PCS and BVLS. For each camera, the re-projection error of each point n (250 points) is calculated.

Figure 5.5 shows the number of inliers obtained in all the positions (frames) for both methods. As it can be seen in the graphic, the number of inliers in tracking systems that take into account the perspective compensation is higher in all the frames. For PCS, the average percentage of inliers is the 65% (65.66%) of the points, and for BVLS is 33%.

However, this is not enough. As mentioned previously in Section 5.2 , in order to show that those results are not generated by chance, the statistical hypothesis testing should be used. In this manner, it can be confirmed that results can be widespread to the population. So, defined null hypothesis has to be refused. The null and alternative hypothesis are defined in Equation 5.6 (Section 5.2).

(a) The initial and final camera positions (green and red cubes)

(b) the images which represent the initial, intermediate and final camera positions. First image = green camera, second image = the central image of the trajectory between the initial and final cameras and third image =red camera

Figure 5.4: Path I

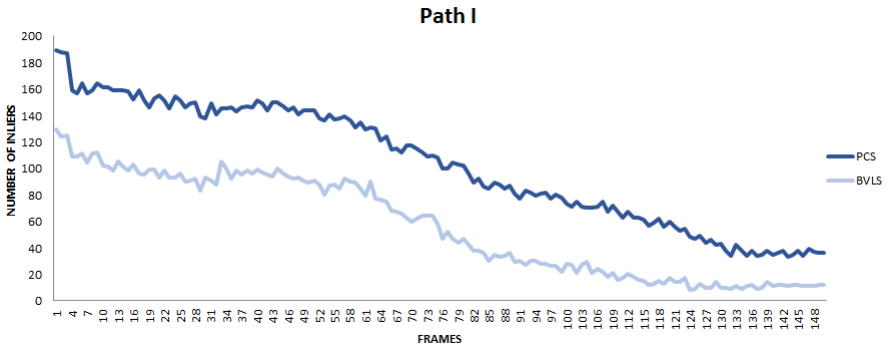
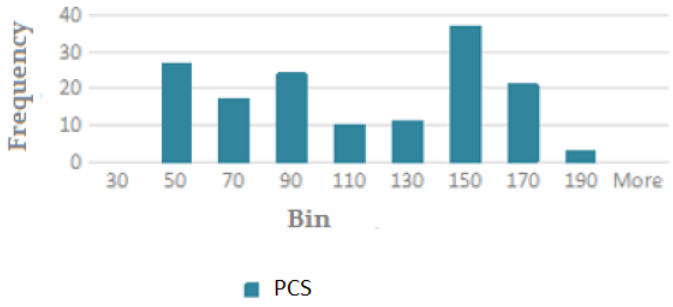
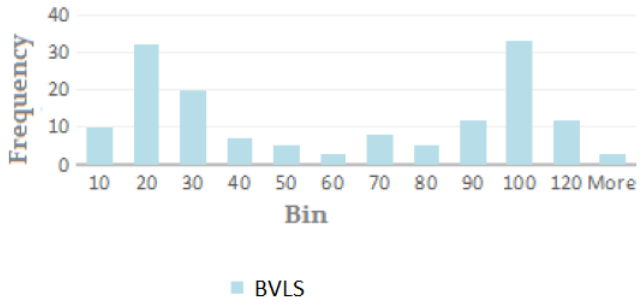


Figure 5.5: Number of inliers obtained for each frame for both systems, PCS and BVLS. Path I



(a) Histogram for PCS



(b) Histogram for BVLS

Figure 5.6: Histograms for Path I

There are many significant tests in the state of the art, depending on the features of the samples. Firstly, it is important to know whether the samples follow a normal distribution or not. Depending on this feature, the significance test that must be applied differs. In this way, histograms for both systems (PCS and BVLS) are analysed.

In Figure 5.6, the histograms for both samples can be seen. Frequency axis represents the number of frames of the sample which has a specific number of inliers (Bin). As can be seen in these graphics, the samples, specially the first one (Figure 5.6(a)), does not seem to follow a normal distribution. However, it has to be verified whether both samples follow a normal distribution.

With this purpose, the samples are analysed using a Chi-squared test [47]. The normal test developed in Python is used for this goal. The function returns a *p-value* of the sample. Usually, the level of significance (α) in these

kinds of test is defined as $\alpha = 0.01$.

The normal test for the sample x_0 has a *p-value* of $9.31 \times 10{-}42$ and for the sample x_1 is 0.89. The first sample (x_0) value is clearly less than $\alpha = 0.01$, which means that it does not follow a normal distribution.

For this reason, the test that has been used in order to refuse H_0 hypothesis and accept the alternative one (H_α) (see Equation 5.6) is the Mann–Whitney U test[51] which is used with no parametric samples. The Python implementation of the Mann–Whitney U test is used in this work. Both samples (two 150 size arrays) are the input data of this function, and it returns the one-sided *p-value*. The *p-value* shows the probability of obtaining the results that we obtain if H_0 is true.

So, if the *p-value* is less than the significance level configured, it means that we can reject H_0 and accept the alternative one (H_α). The significance level for the experiments presented in this work is established in $\alpha = 0.01$, which is the usual value in scientific experiments, to prove the alternative hypothesis with high probability.

The result of the Mann–Whitney U test is $p_v = 2.01 \times 10^{-14}$. As can be seen, the null hypothesis can be refused. So, it can be claimed that the tracking systems that use the surface normal in template matching process (perspective compensation) generate more inliers than the ones that do not take into account the perspective deformations.

As it was mentioned in the introduction part of this chapter (Chapter 5), the increase in inliers supports the accurate estimation of the camera extrinsic. In this way, the difference between the estimate cameras and the cameras of the ground truth is calculated in both systems (PCS and BVLS). The results are shown in Table 5.1.

Method	**Rotation**	**Translation**	**Error** $< 10°$
PCS	3.98°	$0.0427m$	78%
BVLS	5.91°	$0.0764m$	72%

Table 5.1: Difference between the estimated cameras and ground truth cameras for both methods (PCS and BVLS) in Path I

The average rotation and translation error between the estimated camera (from obtained inliers) and the ground truth cameras are exposed in this table (rotation and translation columns). Rotation values are represented in degrees and translation in meters. The % of the frames that were successfully estimated for both methods (which depends on the inliers) are also exposed ($Error < 10°$ column). It is considered a good estimation those which have

(a) The initial and final camera positions (green and red cubes)

(b) the images which represent the initial, intermediate and final camera positions. First image = green camera, second image = the central image of the trajectory between the initial and final cameras and third image =red camera

Figure 5.7: Path II

got less than 10 degrees of error in rotation.

5.4.2 Path II

The virtual model of the same city is used. But in this case, the simulated camera motion bears rotation and translation changes. In addition, the initial and last positions of the cameras are not facing the buildings of the path. It means that there is a perspective deformation between the initial frames, and the frames used to generate the 3D point cloud.

In Figure 5.7(a), the two main camera positions defined can be seen as well as all the camera positions interpolated between them. Figure 5.7(b) shows the images obtained by the camera in the first, intermediate and last positions. In the path presented in this section, the cameras are rotated, and the majority of the points are not facing the camera.

Then, the tracking process is launched (m = 150 frames) using PCS (uses perspective compensation) and BVLS (does not). Only the points with a re-projection error less than 10 pixels are considered inliers.

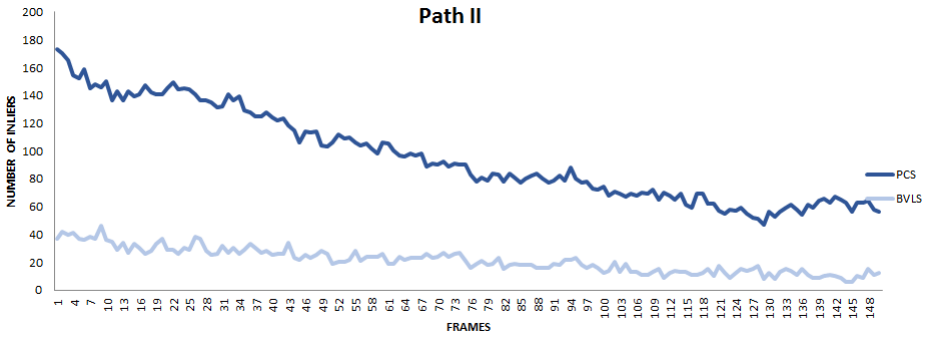


Figure 5.8: Number of inliers obtained for each frame for both systems, PCS and BVLS. Path II

Figure 5.8 shows the number of inliers obtained in all the cameras for both methods. As can be seen in the graphic, the number of inliers in tracking systems that apply perspective compensation is higher in all the images. The average percentage of inliers in PCS is 65% (64.94%) and in BVLS 14%.

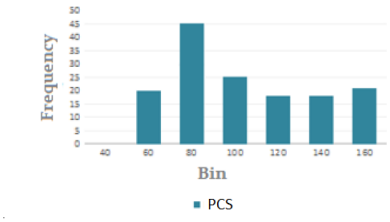
In this path, the inliers difference between both methods is significantly higher than in Path I. In order to prove this trend is not a result of chance, the statistical significance of the results has to be calculated, as it has been done for Path I.

Firstly, it is important to know whether the samples follow the normal distribution. In Figure 5.9, the histograms of the samples can been seen. The Chi-squared test is applied to prove the samples do not to follow the normal distribution.

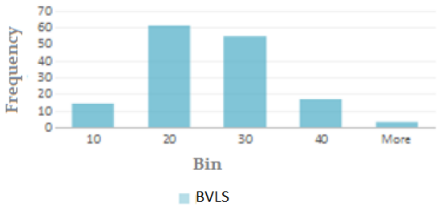
The *p-value* for both methods is: PCS $= 3.11 \times 10^{-8}$ and BVLS $= 0.0321$. The *p-value* of the sample which represents PCS is less than 0.01, so it does not follow a normal distribution (See Figure 5.9(a)).

For this reason, the test that we have to use in order to refuse the H_0 hypothesis and accept the alternative one (H_α) is the Mann–Whitney U test. The Python implementation of the function is used, as in Path I. The value of the *p-value* is 1.04×10^{-50} which is less than $\alpha = 0.01$. So, it can be claimed that the tracking systems that use perspective compensation during the template matching process generate more inliers than the ones that do not.

As far as camera estimations concern, Table 5.2 shows the difference in rotation, translation and the percentage of correctly estimated cameras for both methods in this path.



(a) Histogram for PCS



(b) Histogram for BVLS

Figure 5.9: Histograms for Path II

Method	**Rotation**	**Translation**	**Error** $< 10°$
PCS	$2.37°$	$0.033m$	60%
BVLS	$5.72°$	$0.075m$	25%

Table 5.2: Difference between the estimated cameras and ground truth cameras for both methods (PCS and BVLS) in Path II

5.4.3 Nearest *keyFrame*

The previous test uses as Reference *keyFrame* in Patch Fitting process, the *keyFrame* where the 3D point appeared for the first time. A new experiment is designed using an improved Reference *keyFrame* assignment. A *keyFrame* is assigned as reference if it is the nearest *keyFrame* to the current image. The algorithm implemented will be called NkF (Nearest *keyFrame*).

It could be expected that using a nearer *keyFrame* as Reference *keyFrame*, perspective distortion problem could be solved. This section analyses this topic.

In that case, the experiment is carrying out only for Path II. The results are compared with that obtained when the Reference *keyFrame* is the *keyFrame* where the 3D point cloud is seen firstly (from now on FkF).

For BVLS, the % of inliers obtained when NkF is applied is 43%. How-

ever, when FkF is applied it is 14%. The Figure 5.10 shows the evolution of the number of inliers during the camera motion for both tests (NkF and FkF). BVLS obtains for each frame more inliers (or at least the same) for NkF.

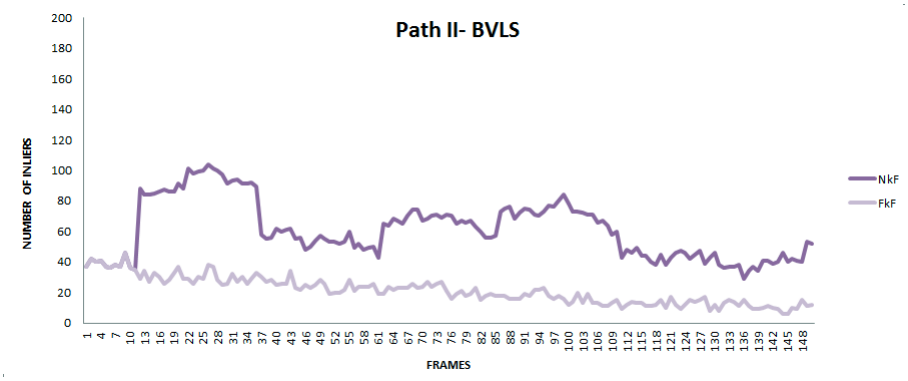


Figure 5.10: Number of inliers obtained in each frame for NkF and FkF in BVLS.

On the other hand, for PCS, the inliers % applying NkF is 79%, instead of 65% obtained applying FkF. The evolution of the number of inliers during the camera motion is shown in Figure 5.11. As for BVLS, PCS obtains more inliers (or at least the same) for NkF than for FkF.

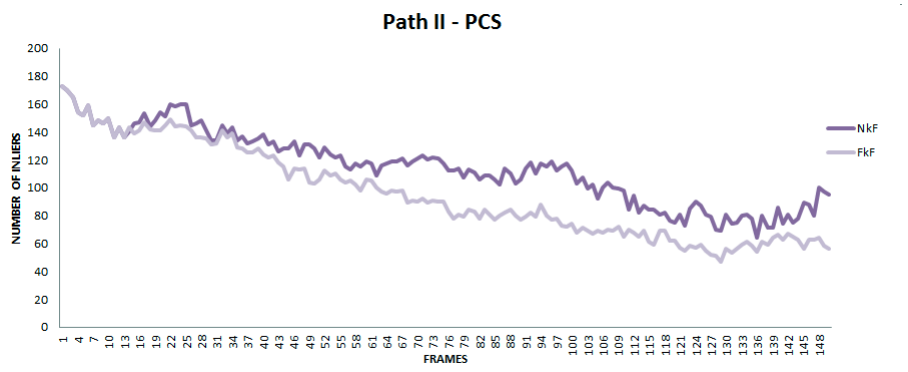


Figure 5.11: Number of inliers obtained in each frame for NkF and FkF in PCS.

In Table 5.3 the results obtained by both methods (PCS and BVLS) using FkF and NkF Reference *keyFrame* assignment modes are shown.

Finally, in Figure 5.12 the evolution of the number of inliers for all the systems analysed are shown together.

Method	FkF	NkF
PCS	65%	79%
BVLS	14%	43%

Table 5.3: The % of inliers obtained for each method, PCS and BVLS, using the two different ways to select Reference *keyFrame*

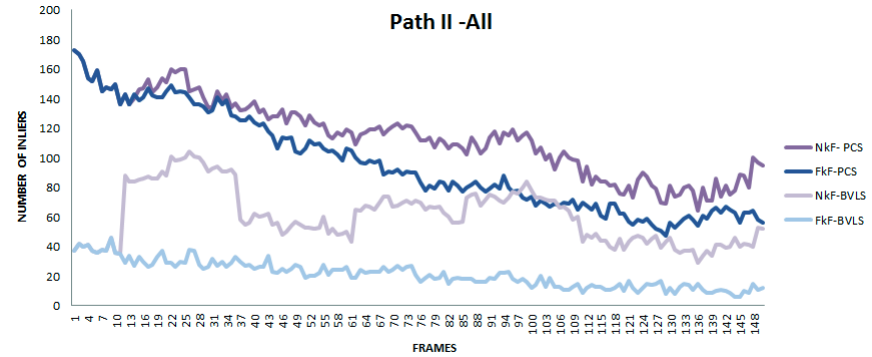


Figure 5.12: Number of inliers obtained in each frame for NkF and FkF in BVLS and PCS

It has to be noticed that in order to be able to apply NkF the implementation of the ground truth was briefly changed. In that case, not only the first image was chosen as *keyFrame*, every 25 images was considered a *keyFrame*. In addition, in the *tracking* task, a function had to be added, in order to calculate what of the *keyFrames* were the nearest to the current frame.

5.5 Discussion

The target of this section is to discuss the results presented in Section 5.4.

Inliers These results prove that the method that consider perspective compensation (PCS) generates more inliers than those who ignore it (BVLS).

In this way, it can be seen that the average value of the sample x_0 (PCS) is higher in both paths than the sample x_1 (BVLS) with significance levels below 0.1% (level of certainty more than 99%). So, **the research presented statistically proves the fact that using the normal surface**

in template matching methods improves the number of inliers in a tracking system.

The results from Sections 5.4.1 and 5.4.2 also provide the following additional knowledge.

In first place, Figure 5.5 and Figure 5.8 clearly show one fact: along both camera paths, perspective compensation always provides more inliers, even in the initial frames when the camera is close to the Reference *keyFrame*. This behaviour is explained realising that: in all the frames there are 3D points whose orientation is not parallel to the camera orientation. So, even in the initial frames, perspective compensation improves the number of found inliers.

This fact, stands out even more taking into account the additional tests done to Path II. When NkF is used in order to compare the patches, the percentage of inliers obtained by both methods (PCS and BVLS) goes up. But, even in that case the percentage of inliers obtained by PCS (79%) is higher than for BVLS (43%).

On the other hand, it has to be also noticed that using NkF elevates the percentage of inliers obtained for both methods (PCS and BVLS) in comparison to using FkF. But, the increase suffered by BVLS is clearly superior to that suffered by PCS. While BVLS increases the percentage of inliers from 14% to 43%, PCS increases from 65% to 79%. NkF improves the results but it does not solve the perspective distortion problem. The best result of BVLS, which is using NkF, is 43% of inliers. It is less than PCS without NkF improvement, which is 65%.

Another global fact can be also deduced from these results. It can be easily observed that the increment of inliers in the Path II is significantly higher than in the Path I: Path II gets 50% more inliers while Path I gets 32%. It means that taking into account the normal surface is especially beneficial when orientation disparities between Reference *keyFrame* and camera increase. This happens when camera motion includes rotations, a fact that is quite common in real scenarios.

In both camera paths and in both methods, the number of inliers decreases as the camera moves away from the Reference *keyFrame*. Experts would expect this behaviour. However, the results at the end of the paths show that this reduction might be critical when perspective compensation is not used.

This critical fact is exacerbated in Path II, where the camera also slightly rotates along its trajectory. An analysis of the path end (last 20% frames) shows that using perspective compensation a 51% (average value) of the 3D points projected into the image were found as inliers. However when it was

not used perspective compensation, only 10% were found as inliers. Even more, the best result obtained without perspective compensation is 23%, while the worst result achieved using perspective compensation is 41%.

Camera estimations The results shown in Table 5.1 and Table 5.2 indicate that camera estimations improve when surface normal is used, due to the number of inliers used in this computation.

In the case of Path II, where the camera movement suffers a rotation changes, improvements are more significant: the average rotation error is 32% better in the case of PCS respect to BVLS for Path I. But, this improvement increases up to 58% in the case of the Path II.

However, the most relevant information is given by the column named "Error $< 10°$": using PCS, the percentage of camera positions which are successfully estimated by the system increases up in 35% for Path II.

Computational Cost Regarding computational cost, when perspective compensation is not computed, the 150 frames require 11.6 seconds for Path I and 19 seconds for Path II. When perspective compensation is computed these times increase a 17% for Path I and 24% for Path II.

It should be taken into account that Patch Fitting process without perspective compensation takes a 89% of the whole time for Path I and a 93% for Path II. Adding, the computational cost of the *Warping Module* does not increases these percentages (91% Path I, and 94% Path II).

Perspective compensation computing cost does not heavily penalised the computing cost of the whole system.

Trade off These facts suggest that the use of perspective compensation should be intelligent: activate perspective compensation when the number of inliers begins to drop in the tracking task. However, there is another situation where perspective compensation may improve the tracking. In some frames it may happen that they seem to have enough number of inliers, but when the linear equations are solved, the solution falls in a local minima. This may happen when there is an apparent enough number of inliers, but most of them are packed in a small area of the image. Perspective compensation may alleviate this problem providing a more disperse set of inliers.

In a nutshell, these results show the benefits provided by perspective compensation and they open new research about how perspective compensation should be integrated into the whole tracking system.

Part III

Concluding Remarks

Chapter 6

Conclusions and future work

The research work done along this thesis is exposed in this chapter in a summary way. A general description of the work as well as the contributions of the dissertation are explicated. Furthermore, a list of publications which prove how this work has been validated with the research community is also expounded. Finally, the dissertation ends with some ideas for future research lines that could be followed.

In that way, the chapter is divided as follow: first section is dedicated to explain in a summary way all the research work done during the thesis, as well as the conclusions obtained from the results. In the second section, the contributions of the thesis are enumerated and explained. Then, the list of all the publications which validate the research exposed in the thesis are expounded. Finally, the last section is dedicated to define some future work ideas that allow the continuation of the thesis work opening new research lines to be explored.

6.1 Research work summary and conclusions

Real time localisation is an essential task within AR domain. It allows AR applications to place virtual element in the real environment completely aligned with the objects of that environment. Visual localisation systems use a **model** as reference in order to perform the localisation task. They are denominated **model-based visual localisation systems**. Many systems use a 3D point cloud as a model (environment map or map).

The first step of this thesis was a new localisation concept proposal, where two main steps are defined: first, a model-based visual localisation system is applied locally to small environments. At the same time, that

map obtained locally is used to extend and update the map in the Cloud. Finally, this large map is used recursively to improve the local maps and localisation (Barrena et.al 2013).

This new concept opens new research lines in the field of collaborative model-based visual localisation systems. The thesis improves them using techniques which measure the similarity between images, i.e **template matching techniques**.

The main goal of model-based visual localisation systems is to stablish a relationship between 3D points and 2D points which are their correspondences in the images. Template matching techniques obtain these matches measuring the similarity between images. As the camera moves, the **perspective distortion** problem degradates performance.

The hypothesis defined in Section 1.2, claims that considering a **surface normal** vector associated with each 3D point, **perspective distortion** problems are palliated.

With the objective of proving that hypothesis, the following steps were done in the thesis:

1. A model-based visual localisation system based on template matching was designed and implemented as a baseline. Along the dissertation it is denominated BVLS. This system does not palliate the perspective distortion problem.

2. An algorithm, which has been denominated WaPT was designed in order to solve the perspective distortion problem. In addition, taking BVLS as a base, a model-based visual localisation system which integrates WaPT was designed and implemented. It is denominated PCS. It also involves an algorithm which estimates the surface normal of each 3D point.

3. Finally, a set of experimental set-ups were designed and performed in order to prove the hypothesis. A preliminary test was done in order to validate the concept exposed in WaPT algorithm. Then, more exhaustive tests were designed and performed using statistical inference theory in order to prove in an objective way the hypothesis.

WaPT algorithm, presented in the Chapter 4, proposes a new model for visual localisation systems. Besides the 3D point cloud, a surface normal vector for each point is also stored. When the 3D point cloud is created a minimization process is also run in order to estimate the best normal for each point.

In the tracking process these normals are used to obtain a more precise matching. In this way, perspective distortions are reduced.

Section 4.4.3 presents the preliminary test. Although they did not proved the hypothesis, those results were promising.

Later on, the hypothesis placed was statistically proven by the results presented in Sections 5.4.1 and 5.4.2.

Section 5.4 analyses the knowledge achieved with the experiments shown in Chapter 5. It shows that perspective compensation provides a significant increment in the number of **inliers**. Inliers are correct 3D-2D correspondences. This increment is higher as the camera moves. As expected, when the camera rotates along the path the benefits provided by perspective compensation increase significantly.

When the camera rotates and moves, the number of inliers becomes critical without perspective compensation. In this case, the increment of inliers provided by the method is quite significant: without perspective compensation only average 14% of the projected points are identified as inliers, while using it the average increases to 65%.

Section 5.4.3 shows that other improvements in the patch fitting process, do not solve the perspective distortion problem.

6.2 Contributions

This section summarises the contributions of the thesis.

1. A new global visual localisation proposal. This concept is based on a collaborative reconstruction process. A single huge and very precise map will be generated and stored in Cloud. Individual users will make use of a local section of this map for their visual localisation. They will also contribute to its continuous update (Barrena et.al 2013).

2. An extension to the 3D point cloud model. It is used in model-based visual localisation algorithms. This thesis adds a surface normal to each 3D point (Barrena et.al 2015).

3. An algorithm to estimate the surface normal vector for each 3D point (Barrena et.al 2015)

4. An algorithm which compensates the image distortion suffered during the camera motion in localisation processes which use template matching (Barrena et.al 2015 & Barrena et.al 2018).

5. A model-based visual localisation system which uses template matching is adapted and improved applying this algorithm (4) (Barrena et.al 2015 & Barrena et.al 2018).

6. The perspective compensation system (PCS) provides much better results than optimal reference *keyFrame* selection: the nearest *keyFrame* (NkF)

7. Ground truth generation. A process to generate a ground truth in order to launch experiments where all the variables are controlled. The ground truth also plays the role of a 3D reconstruction acquirement process where the surface normals, estimated by the proposed algorithm, can be compared with the exact ones (Barrena et.al 2018).

6.3 Relevant publications

The scientific contributions of this dissertation have been presented to the scientific community in a series of publications in international conferences as well as a journal. Below is the list with all these publications:

- International JCR journal
  - Barrena, N., Sánchez, J. R., Ugarte, R. J., & García-Alonso, A. (2018). Proving the efficiency of template matching-based markerless tracking methods which consider the camera perspective deformations. In *Machine Vision and Applications,* 29(4), 573-584.
- International Conferences: algorithm proposal
  - Barrena, N., Sánchez, J. R., & García-Alonso, A. (2013). A distributed and collaborative vSLAM framework for real-time localisation in huge environments for mobile devices. In *Eurographics 2013-Posters* (pp. 11-12). The Eurographics Association.
  - Barrena, N., Sánchez, J. R., & García-Alonso, A. (2015). WaPT-Surface Normal Estimation for Improved Template Matching in Visual Tracking. In *VISAPP (3)* (pp. 496-503).
- International conferences: algorithm applications

- Barbadillo, J., Barrena, N., Goñi, V., & Sánchez, J. R. (2014, December). Collaborative E-Learning Framework for Creating Augmented Reality Mobile Educational Activities. In *International Conference on Ubiquitous Computing and Ambient Intelligence* (pp. 52-59). Springer, Cham.

- Barrena, N., Navarro, A., García, S., & Oyarzun, D. (2016). CoolTour: VR and AR Authoring Tool to Create Cultural Experiences. In *Intelligent Interactive Multimedia Systems and Services 2016* (pp. 483-489). Springer, Cham.

- Barrena, N., Navarro, A., & Oyarzun, D. (2016, April). A Flexible and Easy-to-Use Platform to Create Advanced Edutainment Applications. In *International Conference on Technologies for E-Learning and Digital Entertainment* (pp. 291-300). Springer, Cham.

- Celis, R. D., Barrena, N., Sánchez, J. R., & Ugarte, R. J. (2016, June). Registration of deformable objects using a depth camera. In *the 24th International conference in central Europe on computer graphics, visualization and computer vision 2016* (WSCG'2016).(pp. 33-40). In cooperation with Eurogrpahics.

- Ugarte, R. J., Barrena, N., Díez, H. V., Alvarez, H., & Oyarzun, D. (2016, September). Augmented reality system to assist in manufacturing processes. In *Proceedings of the XXVI Spanish Computer Graphics Conference* (pp. 75-82). Eurographics Association.

- International conferences: AR related

  - Azpiazu, J., Siltanen, S.,Multanen, P., Mäkiranta, A., Barrena, N., Díez, A., Agirre, J., Smith, T. (2011, November). Remote support for maintenance tasks by the use of Augmented Reality: the ManuVAR project. In *IX Congress on virtual reality applications (CARVI 2011)* (pp.)

  - Smith, T., Diez, A., Barrena, N., Azpiazu, J., & Ibarbia, J. A. (2011, September). Remote Maintenance Support in the Railway Industry. In *Joint Virtual Reality Conference* (JVRC 2011) (pp. 20-21).

6.4 Future work

Considering the scope of work of this thesis and the further research lines identified during the dissertation, future work is exposed in this section. They are mentioned according to the contribution to which they belong.

- **Collaborative model-based visual localisation approach:** a new global visual localisation proposal has been introduced in this thesis (see Section 1.1.1).

  The accuracy of the global visual localisation system depend on the accuracy of the huge map. For that reason, the issue of *extend and update the huge map* properly from the local maps obtained, is essential. In that way, two main task are determined: first of all identify in which part of the huge map it has to be added the small one and secondly, how to add them correctly.

  On the other hand, *model-based visual localisation system*, which is carried out locally and for small environments, has to be adapted to use the huge map. This pose re-estimation process, which has to handle with a selection of the huge map, should be also analysed.

  Summarising, the following future work is identified according to the collaborative model-based visual localisation approach:

  1. **Matching between small local map and the huge global one**. The design and implementation of an algorithm should be done. This algorithm will be an hybrid system which matches both, 3D point clouds obtained in the local process and their 2D images, with the 3D point cloud from the big map. Hybrid matching process will identify the correct pose in the huge map where the small map has to be added.
  2. **Affine 3D reconstruction techniques**. In addition, affine 3D reconstruction techniques should be also analysed and improved in order to add the small map to the big one accurately.
  3. **Improving local pose estimation** An algorithm to improve the initially estimated pose. This algorithm should use a selection of a huge map to improve the localisation. So, first of all, this selection should be done properly according to the first localisation, and secondly the pose estimation should be re-calculated.

- **Warped Template Patch Tracking, WaPT:** This is the main contribution of this thesis. It involves two algorithms: one for a surface

normal estimation and another one which uses these normals in order to improve the accuracy on template matching based model-based visual localisation systems.

It has to be remembered, that this thesis was based on model-based visual localisation systems which use a pre-calculate 3D reconstruction of the environment, i.e it use SfM method.

WaPT involves a surface normal estimation process in 3D reconstruction acquirement task. A first approximation of the algorithm has been integrated in a model-based visual localisation system based on SfM. They estimate the 3D point cloud off-line, in a pre-process .

However, vSLAM (visual Simultaneous Localisation and Mapping) is a technique which create a 3D point cloud in real time. It would be interesting to integrate WaPT in that kind of localisation technique. Surface normal estimation, is a minimisation process, which could be a performance problem to carried out on-line. Study and improve this method to integrate it in a vSLAM process is a very interesting challenge.

Bibliography

[1] Abdel-Aziz, Y. (1971). Direct linear transformation from comparator coordinates into object space in close-range photogrammetry. In *Proceedings of the ASP Symposium on Close-Range Photogrammetry, 1971*, pages 1–18. American Society of Photogrammetry.

[2] Agrawal, M. (2007). Practical camera auto calibration using semidefinite programming. In *Motion and Video Computing, 2007. WMVC'07. IEEE Workshop on*, pages 20–20. IEEE.

[3] Alahi, A., Ortiz, R., and Vandergheynst, P. (2012). Freak: Fast retina keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–517.

[4] Azpiazu, J., Siltanen, S., Multanen, P., Mäkiranta, A., Barrena, N., Díez, A., Agirre, J., and Smith, T. Remote support for maintenance tasks by the use of augmented reality: the manuvar project.

[5] Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S., and MacIntyre, B. (2001). Recent advances in augmented reality. Technical report, NAVAL RESEARCH LAB WASHINGTON DC.

[6] Bailey, T., Nieto, J., Guivant, J., Stevens, M., and Nebot, E. (2006). Consistency of the ekf-slam algorithm. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3562–3568. IEEE.

[7] Barbadillo, J., Barrena, N., Goñi, V., and Sánchez, J. R. (2014). Collaborative e-learning framework for creating augmented reality mobile educational activities. In *International Conference on Ubiquitous Computing and Ambient Intelligence*, pages 52–59. Springer.

[8] Barrena, N., Navarro, A., García, S., and Oyarzun, D. (2016a). Cooltour: Vr and ar authoring tool to create cultural experiences. In *Intelligent Interactive Multimedia Systems and Services 2016*, pages 483–489. Springer.

[9] Barrena, N., Navarro, A., and Oyarzun, D. (2016b). A flexible and easy-to-use platform to create advanced edutainment applications. In *International Conference on Technologies for E-Learning and Digital Entertainment*, pages 291–300. Springer.

[10] Barrena, N., Sánchez, J. R., and García-Alonso, A. (2013). A distributed and collaborative vslam framework for real-time localisation in huge environments for mobile devices. In *Eurographics 2013-Posters*, pages 11–12. The Eurographics Association.

[11] Barrena, N., Sánchez, J. R., and García-Alonso, A. (2015). Wapt surface normal estimation for improved template matching in visual tracking. In *Proceedings of the 10th International Conference on Computer Vision Theory and Applications (VISAPP)*, volume 3, pages 496–503. SCITEPRESS.

[12] Barrena, N., Sánchez, J. R., Ugarte, R. J., and Alonso, A. G. (2018). Proving the efficiency of template matching-based markerless tracking methods which consider the camera perspective deformations. *Machine Vision and Applications*, 29(4):573–584.

[13] Barron, J. L., Fleet, D. J., and Beauchemin, S. S. (1994). Performance of optical flow techniques. *International journal of computer vision*, 12(1):43–77.

[14] Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517.

[15] Bouguet, J.-Y. (2001). Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4.

[16] Brunelli, R. (2009). *Template matching techniques in computer vision: theory and practice*. John Wiley & Sons.

[17] Charmette, B., Royer, E., and Chausse, F. (2016). Vision-based robot localization based on the efficient matching of planar features. *Machine Vision and Applications*, 27(4):415–436.

[18] Davison, A. J. (2003). Real-time simultaneous localisation and mapping with a single camera. In *Ninth IEEE International Conference on Computer Vision*, pages 1403–1410. IEEE.

[19] Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007). Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:1052–1067.

[20] Fox, D., Hightower, J., Liao, L., Schulz, D., and Borriello, G. (2003). Bayesian filtering for location estimation. *IEEE pervasive computing*, (3):24–33.

[21] Fuentes-Pacheco, J., Ruiz-Ascencio, J., and Rendón-Mancha, J. M. (2015). Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, 43(1):55–81.

[22] Furukawa, Y. and Ponce, J. (2010). Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376.

[23] Goesele, M., Snavely, N., Curless, B., Hoppe, H., and Seitz, S. M. (2007). Multi-view stereo for community photo collections. In *IEEE 11th International Conference on Computer Vision, ICCV*, pages 1–8. IEEE.

[24] Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer.

[25] Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.

[26] Hartley, R. I. (1994a). An algorithm for self calibration from several views. In *Cvpr*, volume 94, pages 908–912. Citeseer.

[27] Hartley, R. I. (1994b). Self-calibration from multiple views with a rotating camera. In *European Conference on Computer Vision*, pages 471–478. Springer.

[28] Hartley, R. I. and Sturm, P. (1997). Triangulation. *Computer vision and image understanding*, 68(2):146–157.

[29] Kato, H. and Billinghurst, M. (1999). Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Augmented Reality, 1999.(IWAR'99) Proceedings. 2nd IEEE and ACM International Workshop on*, pages 85–94. IEEE.

[30] Kirchner, M. R. (2016). Automatic thresholding of sift descriptors. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 291–295. IEEE.

[31] Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small ar workspaces. In *6th IEEE and ACM International Symposium on Mixed and Augmented Reality. ISMAR*, pages 225–234. IEEE.

[32] Koenderink, J. J. and Van Doorn, A. J. (2007). Affine structure from motion. *JOSA A, Journal of the Optical Society of America*, 8(2):377–385.

[33] Lepetit, V., Fua, P., et al. (2005). Monocular model-based 3d tracking of rigid objects: A survey. *Foundations and Trends® in Computer Graphics and Vision*, 1(1):1–89.

[34] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

[35] Lucas, B. D., Kanade, T., et al. (1981). An iterative image registration technique with an application to stereo vision.

[36] Luong, Q.-T. and Faugeras, O. D. (1997). Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of computer vision*, 22(3):261–289.

[37] Milgram, P. and Kishino, F. (1994). A taxonomy of mixed reality visual displays. *IEICE TRANSACTIONS on Information and Systems*, 77(12):1321–1329.

[38] Molton, N., Davison, A. J., and Reid, I. (2004). Locally planar patch features for real-time structure from motion. In *BMVC*, pages 1–10.

[39] Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B., et al. (2002). Fastslam: A factored solution to the simultaneous localization and mapping problem. *Aaai/iaai*, 593598.

[40] Moré, J. J. (1978). The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer.

[41] Mostofi, N., Moussa, A., Elhabiby, M., and El-Sheimy, N. (2014). Rgb-d indoor plane-based 3d-modeling using autonomous robot. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40(1):301.

[42] Mourtzis, D., Zogopoulos, V., and Vlachou, E. (2017). Augmented reality application to support remote maintenance as a service in the robotics industry. *Procedia CIRP*, 63:46–51.

[43] Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, 2(331-340):2.

[44] Naimark, L. and Foxlin, E. (2002). Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, page 27. IEEE Computer Society.

[45] Perveen, N., Kumar, D., and Bhardwaj, I. (2013). An overview on template matching methodologies and its applications. *IJRCCT*, 2(10):988–995.

[46] Piskunov, N., Medkov, K., et al. (1983). *Cálculo diferencial e integral*, volume 1. Mir.

[47] Plackett, R. L. (1983). Karl pearson and the chi-squared test. *International Statistical Review/Revue Internationale de Statistique*, pages 59–72.

[48] Quan, L. (1996). Self-calibration of an affine camera from multiple views. *International Journal of Computer Vision*, 19(1):93–105.

[49] Rana, R., Singhal, R., et al. (2015). Chi-square test and its application in hypothesis testing. *Journal of the Practice of Cardiovascular Sciences*, 1(1):69.

[50] Ristic, B., Arulampalam, S., and Gordon, N. (2003). *Beyond the Kalman filter: Particle filters for tracking applications*. Artech house.

[51] Rosner, B. and Grove, D. (1999). Use of the mann–whitney u-test for clustered data. *Statistics in medicine*, 18(11):1387–1400.

[52] Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer.

[53] San Martín, R. and Pardo, A. (1989). Psicoestadística. contrastes paramétricos y no paramétricos. *Madrid: Pirámide*.

[54] Schmalstieg, D. and Hollerer, T. (2016). *Augmented reality: principles and practice*. Addison-Wesley Professional.

[55] Schmidt, A. (2016). Prediction-based perspective warping of feature template for improved visual slam accuracy. In *Man–Machine Interactions 4*, pages 169–177. Springer.

[56] Shi, J. and Tomasi, C. (1994). Computer vision and pattern recognition. In *1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 593–600.

[57] Sivaraman, S. and Trivedi, M. M. (2013). Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795.

[58] Smith, T., Diez, A., Barrena, N., Azpiazu, J., and Ibarbia, J. A. (2011). Remote maintenance support in the railway industry. In *Joint Virtual Reality Conference (JVRC2011)*, pages 20–21.

[59] Strasdat, H., Davison, A. J., Montiel, J. M., and Konolige, K. (2011). Double window optimisation for constant time visual slam. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2352–2359. IEEE.

[60] Tomasi, C. and Kanade, T. (1991). Detection and tracking of point features.

[61] Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (1999). Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer.

[62] Tsai, R. Y. (1986). An efficient and accurate camera calibration technique for 3d machine vision. *Proc. of Comp. Vis. Patt. Recog.*, pages 364–374.

[63] Ugarte, R., Barrena, N., Diez, H., Alvarez, H., and Oyarzun, D. (2016). Augmented reality system to assist in manufacturing processes. In *Proceedings of the XXVI Spanish Computer Graphics Conference*, pages 75–82. Eurographics Association.

[64] Vávra, P., Roman, J., Zonča, P., Ihnát, P., Němec, M., Kumar, J., Habib, N., and El-Gendi, A. (2017). Recent development of augmented reality in surgery: a review. *Journal of healthcare engineering*, 2017.

[65] Wagner, D., Langlotz, T., and Schmalstieg, D. (2008). Robust and unobtrusive marker tracking on mobile phones. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 121–124. IEEE Computer Society.

[66] Wagner, D. and Schmalstieg, D. (2007). *Artoolkitplus for pose tracking on mobile devices*. na.

[67] Watman, C., Austin, D., Barnes, N., Overett, G., and Thompson, S. (2004). Fast sum of absolute differences visual landmark detector. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 5, pages 4827–4832. IEEE.

[68] Wu, C., Clipp, B., Li, X., Frahm, J.-M., and Pollefeys, M. (2008). 3d model matching with viewpoint-invariant patches (vip). In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

[69] Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22.

[70] Zhao, Z., Ye, D., Zhang, X., Chen, G., and Zhang, B. (2016). Improved direct linear transformation for parameter decoupling in camera calibration. *Algorithms*, 9(2):31.