Universidad Politécnica de Madrid

Escuela Técnica Superior
de Ingenieros de Telecomunicación

DETECTION AND TRACKING OF VANISHING POINTS IN DYNAMIC ENVIRONMENTS

DETECCIÓN Y SEGUIMIENTO DE PUNTOS DE FUGA EN ENTORNOS DINÁMICOS

Ph. D. Thesis
TESIS DOCTORAL

MARCOS NIETO DONCEL
(Ingeniero de Telecomunicación, UPM)

2010

Departamento de Señales, Sistemas y Radiocomunicaciones
Escuela Técnica Superior de Ingenieros de Telecomunicación
Universidad Politécnica de Madrid

DETECTION AND TRACKING OF VANISHING POINTS IN DYNAMIC ENVIRONMENTS

DETECCIÓN Y SEGUIMIENTO DE PUNTOS DE FUGA EN ENTORNOS DINÁMICOS

Ph. D. Thesis
TESIS DOCTORAL

Autor:

Marcos Nieto Doncel

Ingeniero de Telecomunicación
Universidad Politécnica de Madrid

Director:

Luis Salgado Álvarez de Sotomayor

Profesor titular del Dpto. de Señales, Sistemas y Radiocomunicaciones
Universidad Politécnica de Madrid

2010

TESIS DOCTORAL

DETECTION AND TRACKING OF VANISHING POINTS IN DYNAMIC ENVIRONMENTS

DETECCIÓN Y SEGUIMIENTO DE PUNTOS DE FUGA EN ENTORNOS DINÁMICOS

Autor: Marcos Nieto Doncel
Director: Luis Salgado Álvarez de Sotomayor

Tribunal nombrado por el Mgfco. y Excmo. Sr. Rector de la Universidad Politécnica de Madrid, el día . . . . de . . . . . de 2010

Presidente	D. . . . .
Vocal	D. . . . .
Vocal	D. . . . .
Vocal	D. . . . .
Secretario	D. . . . .

Realizado el acto de defensa y lectura de la Tesis el día . . . . de . . . . . de 2010
en . . . . . .

Calificación: . . . . . .

EL PRESIDENTE LOS VOCALES

EL SECRETARIO

A Usoa y Katixa

Acknowledgements

Habiendo coincidido en el tiempo mi lectura de tesis con mi marcha de Madrid, aprovecho esta sección como despedida y resumen de los años que he pasado trabajando en la Universidad.

Creo poder decir que mi entrada en el GTI, hace unos seis años, cambió el rumbo de mi vida. Aquí me he desarrollado personal y profesionalmente, y he conocido a la gente que ha terminado siendo parte de mi vida. Recuerdo una charla con mi tutor Luis Salgado, en la que le comentaba que no estaba seguro de querer hacer una tesis doctoral. Demasiado tiempo, demasiado esfuerzo. Sin embargo él insistió y me aseguró que a la larga lo agradecería. En efecto, así ha sido, de forma que mi mayor agradecimiento es para Luis.

En segundo lugar mi mente me lleva a mi familia. A mis padres y mi hermana, que me dieron una educación que me ha hecho tener hoy los principios que me rigen y orientan mis decisiones. A mi nueva familia, Usoa y Katixa, que le dan sentido a lo que soy ahora y me han hecho madurar más de lo que podía imaginar.

Debo confesar que muchos lunes me he despertado contento de poder ir al GTI. Esto ha sido posible gracias a este inigualable grupo humano. Habéis hecho que merezca la pena, y me marcho convencido de haber aprovechado mi tiempo aquí, haciendo buenas amistades, de esas que no se rompen.

Por orden cronológico: Carlos Roberto, gran samaritano, eres el pegamento del grupo, lo mantienes unido con tu forma de ser. Juan Carlos, casi todos los nuevos están hartos de oirme contar cosas que dijiste. La gente piensa que hablo de seres mitológicos. Carlos Cuevas, nunca he conocido, y estoy seguro de no conocer jamás, a alguien que primero hace, luego dice y a veces piensa. Qué feliz eres, cómo te admiro. Raúl, no podré agradecerte lo suficiente tus consejos matemáticos, sobre cómics y rock and roll: brazzo Raúl. Jon, aunque al principio pensé que venías a robarme a Usoa, luego me di cuenta de que eras un amigo de verdad, y tu forma de ser me ha hecho ver las cosas con más calma, y a veces era justo lo que necesitaba. Víctor, ahora quedas sólo entre los Carlos, has de ponerles freno, confío en ti. Pablo, Gianlu, Maykel, Fili, César, Toñi, Esther, Sasho, Massimo... espero que no os enfadéis si os agrupo a todos, seguro que sabéis que sólo tengo buenas palabras para vosotros, por tomar café

conmigo, invitarme a cenar cuando estaba sólo y triste, por hacerme pensar que sé explicar las cosas...

Para Narciso tengo una mención especial, y es que como jefe de grupo, debo agradecer que me permitiera entrar en el GTI, cuando aún no conocía a Usoa y mi vida podría haber tomado un rumbo catastrófico en las fauces de las consultorías. Fernando, como pilar del GTI te debemos todos mucho, y yo más, por haberte cargado de trabajo cuando peor te venía y haberlo resuelto a pesar de todo. F, Julián, sin duda estoy en deuda con vosotros, así que si un día os acercáis al norte, llamad a mi puerta y os invito a unos zuritos.

Menciono rápidamente a otros más fugaces en el GTI, que si no pueden enfadarse y esta vez no quiero dejarme a nadie: Kike, José, podríais haber esperado un par de meses antes de iros, gracias a ello he tenido que soportar la pesada carga de ser el doctorando más veterano desde el principio...; Hui, y LiHui, por reiros de todas las bromas; Marcos, Alcorcón ya no vale nada desde que te fuiste, Laurent, Alberto, Ainhoa, Juan, Alessandro, todos aquellos de la época del PFC. También a los que colaboraron conmigo aunque fuese obligados por Luis: Sharko, Dani, Alfonso, Pieter, Matheus, Ivana, Kristina, Álex, por estrellarte conmigo, y tantos otros que no me caben ya.

Va por ustedes,

Marcos Nieto

Abstract

This thesis is about the detection and tracking of vanishing points in images, its application to rectification of planes, and the additional considerations that the road scenario imposes on these topics.

The core element of the dissertation is the obtention of vanishing points, for which a number of contributions are introduced in the field of robust estimation through optimization procedures. RANSAC (RANdom Sampling And Consensus) and the EM (Expectation-Maximization) algorithm are studied for initialization and refinement purposes, respectively. The results show that these proposals use the information of the images more efficiently and with better performance than other approaches in the literature.

Two chapters cover the extraction of the required information from the images in order to feed the estimation methods. On the one hand, images of structured environments (those containing rectilinear elements such as corridors, buildings, roads, etc.), contain line segments that can be detected using the SSWMS (Slice Sampler and Weighted Mean Shift) algorithm, proposed in this thesis. This method has been conceived to work in real-time and to represent a trade-off between accuracy and robustness. On the other hand, focusing on the specific road scenario, the information of the images is extracted through more complex procedures, which are based on recursive Bayesian classification and model fitting.

Finally, the required conditions for the rectification of planes using vanishing points are determined, together with an evaluation of the recovered geometric properties of the plane according to the available information of the scene. Besides, paying special attention to the road scenario, a number of case studies are identified, for which corresponding practical methods for plane rectification are proposed.

Along this thesis, results are presented individually for each proposed strategy, comprising feature extraction, vanishing point estimation, road modeling and plane rectification. As a practical application, these strategies are jointly used as a vision-based ADAS (Advanced Driver Assistance Systems) that generates and updates, dynamically, the rectified images and the resulting road model.

Resumen

Esta tesis aborda la detección y seguimiento de puntos de fuga en imágenes, su aplicación para la rectificación de planos y las consideraciones adicionales que plantea el escenario de carretera.

El elemento central de la tesis es la obtención de puntos de fuga, para lo cual se presentan diversas contribuciones en el ámbito de la estimación robusta mediante procesos de optimización basados en RANSAC (RANdom Sampling And Consensus) y el algoritmo EM (Expectation-Maximization) para inicialización y refinado, respectivamente. Los resultados demuestran que estas propuestas, en comparación con otros métodos, permiten utilizar de forma más eficiente y con mejores resultados la información de las imágenes.

En torno a estas estrategias de estimación, se incluyen dos capítulos dedicados a la extracción de características de las imágenes necesarias para la detección de estos puntos. Por un lado, para imágenes de entornos estructurados (aquellos con elementos rectilíneos, como pasillos, edificios, carreteras, etc.), se propone obtener segmentos de línea mediante el algoritmo SSWMS (Slice Sampler and Weighted Mean Shift). Este método ha sido diseñado para funcionar en tiempo real y obtener resultados precisos y robustos. Por otro lado, cuando nos centramos en el escenario de la carretera, la información de las imágenes se obtiene mediante procedimientos más complejos, en los que se recurre a métodos de estimación recursiva Bayesiana, y al ajuste de modelos.

Finalmente, se determinan las condiciones necesarias para la rectificación de planos a partir de puntos de fuga, y se analiza el tipo de propiedades geométricas del plano que se recupera en función de la información utilizada. Además, con especial énfasis en el escenario de la carretera, se identifican una serie de casos de estudio para los cuales se proponen métodos prácticos de generación de imágenes rectificadas.

A lo largo de esta tesis se han presentado resultados individuales en cada una de las estrategias presentadas, tanto en extracción de características, estimación de puntos de fuga, generación de modelos de carretera y obtención de imágenes rectificadas. Como aplicación práctica, se plantea la utilización de estas estrategias en una esquema de ADAS (Advanced Driver Assistance Systems) basado en visión en el que se actualizan, dinámicamente, las imágenes rectificadas, y los modelos de carretera resultantes.

Contents

List of Figures

List of Tables

Glossary of acronyms

ADAS	Advanced Driver Assistance Systems
CCA	Connected Component Analysis
CPL	Calibrated Point-line
CS	Consensus Set
DAS	Driver Assistance Systems
DLT	Direct Linear Transform
EM	Expectation-Maximization
EPM	End-points error minimization
EPx	End-points error maximization
HD	High Definition
HT	Hough transform
IAC	Image of the Absolute Conic
IPM	Inverse Perspective Mapping
ITS	Intelligent Transportation Systems
LCDR	Lane Change Detection Range
LCFPR	Lane Change False Positive Rate
LDW	Lane Departure Warning
LSD	Line Segment Detector
MFC	Microsoft Foundation Class
MH	Metropolis-Hastings
MLESAC	Maximum Likelihood Estimator Sample and Consensus
MS	Mean Shift
MSAC	M-estimator Sampling and Consensus
MSS	Minimal Sample Set
O	Orientation
OS	Orientation with scale
pEM	Point-based EM
PL	Point-line

PPHT	Probabilistic Progressive Hough Transform
pUEM	Point-based Uniform EM
RANSAC	Random Sampling and Consensus
RHT	Random Hough transform
RP	Recall & Precision
SHT	Standard Hough transform
SQP	Sequential Quadratic Programming
SRS	Sequential Random Sampling
SSWMS	Slice Sampling Weighted Mean Shift
SVD	Singular Value Decomposition
vpEM	Vector-point EM
vpUEM	Vector-point Uniform EM
YUDB	York Urban Database
WMS	Weighted Mean Shift

Chapter 1

Introduction

This thesis is a dissertation about the detection and tracking of vanishing points in images and its application to plane rectification. The thesis investigates relevant topics related to this purpose, such as the extraction of features, the computation of vanishing points, and the projective geometry behind the plane rectification task. As a result, several new methods and algorithms are devised for each one of these aspects for generic scenarios, and special strategies are added to handle the particularly challenging "road scenario"[1].

Plane rectification refers to the task of finding the projective transformation that rectifies a perspective image of a plane (e.g. the road, a wall, the floor) into a fronto-parallel view of it, as shown in the examples of figure 1.1. This transformation is typically known as planar homography [46], and can be defined as an invertible transformation or linear relationship between points on planes that maps straight lines to straight lines. The objective is to recover geometric properties of lines and points, such as parallelism, angles, length ratios, metric distances, etc. The example images of figure 1.1 illustrate this target, and show that the perspective distortion can not be completely removed from a single view as one dimension is lost through the projection of 3D world elements to their 2D image. This is the reason why only the elements of the plane are correctly rectified and elements above the plane, such as vehicles, trees or balconies, appear distorted in the transformed domain.

This transformation is very commonly used in computer vision applications, such as mosaics and panorama generation [52], rectified photography [21], epipolar stereo alignment [39], traffic applications [72], etc. The recovery of geometric information can be used to reduce the complexity of the models to be fitted to the image data, or to extract features according to some known geometric properties, such as parallelism, orthogonality, etc. Specifically, in video-based traffic applications, the road is assumed

[1] This scenario is described further in this introduction and in great detail in chapter 4.

(a) (b) (c) (d)

Figure 1.1: Rectification of imaged planes; (a-b) example of road image and its fronto-parallel view; and (c-d) example of facade rectification.

to be locally flat (as done, for instance, by Bertozzi and Broggi [14], Kim [58], Klappstein et al. [60], McCall and Trivedi [74], Southall and Taylor [107]), so that the plane rectification is applied to obtain the so-called bird's-eye view of the road ahead the vehicle[2]. In this scenario, there is a number of benefits derived from this computation: lane markings are supposed to be parallel in this domain, lanes appear with their actual width up to scale, the complexity of the curvature analysis is reduced and also the relative speed and position of the vehicles are imaged without perspective distortion, allowing to estimate metric distances between points on the road plane.

The computation of homographies for plane rectification is a well known problem in the literature (generic approaches are described by Hartley and Zisserman [46]), and it is typically carried out by finding reference elements in the images, such as known length ratios, known angles, and especially vanishing points.

The vanishing points are elements of great interest in the computer vision field, since they are the main source of information for structured environments. The simplest definition is that *a vanishing point is a point in the image plane on which images of parallel lines of the scenario seem to converge* [85].

The importance of vanishing points comes from the fact these image points correspond to three-dimensional directions in the space. There is a number of applications that can take advantage of the computation of these points, such as plane rectification [69], camera calibration [68], single view metrology [33], compass estimation [32], etc.

Vanishing point estimation is a major topic in the computer vision research community since decades, hence there are many noteworthy works related to their detection in images, like the ones by Almansa et al. [2], Barnard [12], Košecká and Zhang [65], Liebowitz [68], Lutton et al. [71], McLean and Koyyuri [75], Pflugfelder [93], Schindler and Dellaert [99], only to mention a few. Nevertheless, from a practical

[2]This transformation is also known as Inverse Perspective Mapping (IPM) in the traffic applications literature [14].

point of view, there are many aspects to work on. Namely, new contributions should search not only for improved accuracy, but for efficiency (how much information is required by the method?), flexibility (what type of information can it use?), automatism (can it be implemented for any computer vision task?) and speed (can it be really used for online systems?).

Even though there exists a wide variety of vanishing point strategies, most of them share a common fundament: they use algorithms that search for intersection points of image elements with orientation such as lines or gradient-pixels. A vanishing point necessarily implies the presence of an intersection point, though, for an unknown scenario, it is not possible to assert that an intersection point corresponds to a three-dimensional direction (i.e. a vanishing point). To be able to distinguish between intersection points that actually correspond to vanishing points, information about the scenario is required. For instance, there are many methods that take advantage of the relationship between vanishing points, such as the assumption that there are three mutually orthogonal directions in the scene as described by Coughlan and Yuille [32].

Considering challenging scenarios, it is important to include the prior information that allows to compute intersection points and determine that they actually correspond to vanishing points. Specifically, the road scenario is an outdoor uncontrolled environment, which can be extremely variable regarding light conditions, lack of visibility due to the weather, instability caused by irregularities of the road, driver maneuvers, accelerations, etc. Besides, in these images, there are typically very few visual features (such as edges or corners) that can be used to compute vanishing points. Therefore, in this thesis, part of the work deals with this problematic scenario, including a specific road modeling stage, which helps to compute vanishing points and to accomplish the road plane rectification task.

1.1 Motivation and objectives

The core idea of the dissertation was born in the field of video-based driver assistance systems. Within them, the rectification of planes represents a great tool to transform the image and restore geometric properties of the road plane.

Nevertheless, the related literature does not explicitly pay attention to the correct computation of this transformation along time, since it is typically computed once during the system setup and kept fixed [14, 104]. This approach is clearly unrealistic in any real application and even more in the road scenario, where the camera is mounted on a fast moving vehicle, subject to vibrations, instability, road slopes, etc.

Facing this problem from the projective geometry point of view, this transformation is understood as the metric rectification (up to scale) of the road plane. The rectification requires the computation of a planar homography that contains enough

information to remove the distortion between planes [46]. For this purpose, vanishing points can be computed, since they determine directions in the space, and also can be used to retrieve other information about the environment, such as length ratios, known orientations or reference distances. According to the number of known vanishing points, the relationship between them and any other information about the scenario, the obtained plane rectifications vary from projectivities to similarities. The work of Liebowitz and Zisserman [69] about stratified plane rectification depicts this granularity in detail.

Therefore, plane rectification is the final objective of this dissertation, in which the road scenario is tackled with special interest, since it is extremely complex for its dynamism, lack of reliable information and general unpredictability.

Hence, the core topic of the thesis is the robust[3] detection and tracking of vanishing points as the main actors for the plane rectification problem. For that reason, the proposal of innovative methods for their robust computation constitute an important part of this thesis. Besides, the extraction of features has been given a significant relevance in the research contributions of the thesis, including a general line segment method and specific feature extraction methods for the road scenario.

As a summary, the following points index the main objectives of the work:

- Design and develop robust vanishing point detection strategies, which can be used to track multiple vanishing points along time in challenging scenarios.
- Propose a line segment detector that finds a trade-off between robustness and efficiency and that overcomes the drawbacks of currently existing methods.
- Design road modeling methods that exploit the prior information about the specific road scenario, in which lanes, vehicles and other elements may contain useful information for the plane rectification task.
- Propose and evaluate plane rectification alternatives according to the information contained in the detected vanishing points and in the obtained road model for the specific road scenario.

1.2 Outline of the dissertation

The thesis is organized starting from the feature extraction algorithm (the line segment detector), following with the vanishing point detection methods, the road modeling strategies and ending with the proposed alternatives for plane rectification. At each

[3]Robustness is understood in this context as the ability of achieving accurate results in the presence of a significant proportion of outliers or clutter.

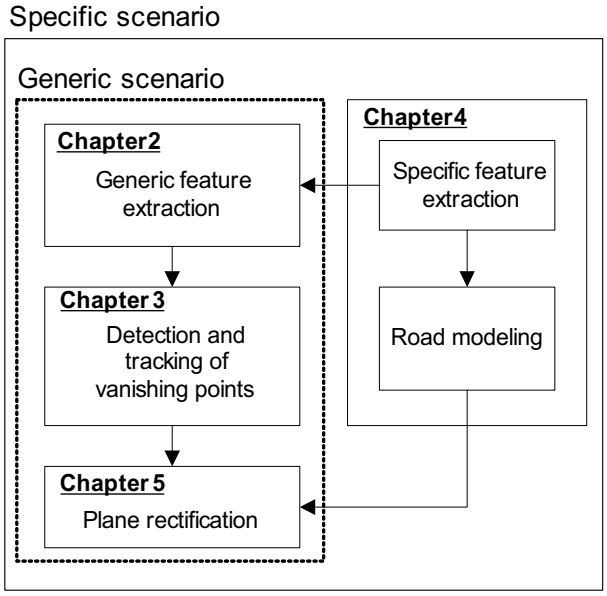


Figure 1.2: Block diagram of the thesis outline. Chapters are organized according to the definition of a generic and a specific scenario, for which the modules described in chapter 4 are added to the process.

chapter the problems are exposed in a theoretical and practical way, addressing the previous related work found in the literature, the description of the proposed new methods to solve them, and the obtained results. The chapter structure is illustrated in figure 1.2, and described as follows:

- Chapter 1 introduces the main topics discussed in this thesis, including the objectives and research contributions.

- Chapter 2 describes a new general method for feature extraction, which is a line segment detector, denominated *Slice Sampling Weighted Mean Shift* (SSWMS).

- Chapter 3 contains all the proposed algorithms for vanishing point detection and tracking, including several versions for robust estimation (using RANSAC-like methods) and different approaches to the use of the Expectation-Maximization algorithm for vanishing point estimation.

- Chapter 4 is divided in two parts. The former accounts for the designed Bayesian classifier that extracts the specific information from the road images. The latter describes the proposed road models that use this information in order to feed the plane rectification strategy with richer information.

- Chapter 5 goes into the details of the plane rectification problem related to the generic and specific scenario, in which the different prior information about the environment allows different levels of distortion removal.

- Chapter 6 concludes the thesis summarizing the main achievements, discussing the problems presented, the obtained results, and the future work.

1.3 Research contributions

There are different proposals in this thesis that can be considered as significant research contributions, which are supported by a number of international publications.

The research lines and publications that emerged during the elaboration of this thesis are mentioned here:

Robust vanishing point estimation The most important contributions in this field are the methods for robust estimation of vanishing points based on RANSAC-like algorithms, as well as the EM-based methods. As a result, three articles have been accepted for publication in international conferences [83, 84, 85]. The application of vanishing points on other well-known related problem, video-mosaicing, has been also published recently [86].

Road-scenario modeling As a specific target, the road scenario has received a significant effort during the elaboration of this thesis. On the one hand, specific techniques for vanishing point detection and tracking in this scenario were designed and published [82]. On the other hand, road modeling strategies, such as a specific feature extraction for road image segmentation [88], hierarchical bipartite graphs for road modeling [90], computation of multiple-lanes model of the road [89], or other works related to the detection and tracking of vehicles in this domain [3, 5, 6, 7, 8, 9]. Recently, an article related to this and other topics has been submitted to the "Machine Vision and Applications" journal.

Plane rectification Several methods for plane rectification have been proposed applying projective geometry concepts and particularizing in the road scenario. There are several works that used plane rectification subject to the constraints of the road scenario to model the ego-motion of the own vehicle [87], to analyze the curvature of the road [90], or the presence of multiple lanes [84, 89].

Stochastical filtering Particularly interesting are the contributions in the field of Bayesian filtering, such as the works in particle filtering (for which an article has been published related to the management of multiple objects in a single filter

[91]), or Bayesian classifiers, which have been used to model dynamic environments [88].

Line segment detection Related to the generic feature extraction techniques, it is remarkable the contribution in the field of line segments detection. The proposed method (SSWMS) represents a trade-off solution for fast and robust detection of features. This work has been submitted for review into the "Pattern Analysis and Applications" journal.

Chapter 2

Generic feature extraction

2.1 Introduction

Straight lines or line segments within an imaged scene can be of great help to infer its geometric properties and also the projection process that converts 3D world elements into a 2D image. Man-made environments typically contain multiple line segments oriented towards a number of common directions (belonging to flat surfaces such as the ground, doors, walls, or buildings). For that reason, line segments can be used as low level features for conventional computer vision problems, such as the detection of vanishing points [2, 65], autocalibration [93], plane rectification techniques [69, 72], 3D reconstruction [70] or object tracking [103].

In this chapter the problem of finding line segments in real images is faced in a probabilistic way. An image is considered as a set of observations, $I = \{(x_k, y_k)\}_{k=1...N}$, corresponding to the pixels of the image, and the detection of line segments considers a subjacent probability density function $p(x, y)$ that represents the probability of a pixel to belong to a line segment. One of the main contributions of the proposed strategy is on the generation of an estimate $\hat{p}(x, y)$ of this distribution from the analysis of the eigenvalues of the tensor matrix associated to the image pixels. Besides, the real-time performance of the proposed strategy is achieved by the use of a sequential sampling technique that selects pixels in the image that likely belong to line segments according to $\hat{p}(x, y)$. Line segments are obtained through the Bresenham growing algorithm enhanced with the use of weighted Mean Shift (wMS) procedures that provides accurate fits and robustness against noise.

The target of this method is the obtention of line segments in any kind of real images. Therefore, the results section evaluates the obtained line segments for a wide variety of real images. Nevertheless, since one of the major feats of this approach is its low computational cost, its performance is better exploited for vision applications

working on sequences of images. For instance, plane rectification strategies based on the computation of vanishing points in moving camera sequences of structured environments [65, 85], which require real-time operation while accurate and meaningful line segment detections.

2.2 Related work

Most approaches in the related literature use edges extracted at pixel level from the images as basic information. The Sobel edge detector [48] is typically used as an approximation to the first order spatial derivatives $I_x(x, y)$ and $I_y(x, y)$. For each pixel k at location (x_k, y_k) the module and orientation of the gradient can also be estimated as described in [34]:

$$||\nabla I(x_k, y_k)|| = \sqrt{I_x^2(x_k, y_k) + I_y^2(x_k, y_k)} \tag{2.1}$$

$$\theta(x_k, y_k) = \arctan\left(\frac{I_y(x_k, y_k)}{I_x(x_k, y_k)}\right) \tag{2.2}$$

Once obtained, the information of edges at pixel level can be used in different ways to search for line segments. Two main groups of techniques can be identified: those based on accumulation in parametric spaces, such as the methods based on the Hough transform (HT) [1, 10, 34, 40, 59, 63, 109]; and pixel clustering strategies in the image plane [22, 44, 56, 119].

2.2.1 Hough transform

The former group basically performs a parametric transformation to the set of edge pixels, from which lines are extracted searching for local maxima in the transform domain. Note that the Standard HT (SHT) detects lines, but not line segments, which require addressing additional considerations for their detection, as done by Khan et al. [56] or Yuen et al. [125]. Some other works just face the problem of finding lines, as done by Stephens [109], Xu et al. [124], Kiryati et al. [59] or recently Aggarwal and Karl [1] with their Regularized HT.

The major drawback of the works based on the HT is the excessive algorithm complexity [34, 109], which typically avoids its use on online applications. Random sampling applied to the HT (RHT) has been originally proposed by Xu et al. [124], and further improved by Kiryati et al. [59] to render efficient line segment detectors. A pair of edge points is selected at random to accumulate a single vote on the transformed domain. These approaches are the so-called "many-to-one" voting schemes, in contrast to the "one-to-many" corresponding to the SHT, which dramatically reduce

the computational cost of the HT. Several variants of the RHT have been proposed by Kälviäinen et al. [63], which improve the sampling distribution by means of a two-steps random process: windows of fixed or also random size are randomly selected in the image, and the RHT is then applied to search for maxima. Walsh and Raftery [120] proposed an importance sampling procedure to improve random sampling within the RHT.

The resulting line segments can be further grouped into longer ones as done by Bandera et al. [10], which uses Mean Shift procedures to cluster line segments in the transform domain and obtain a single representative for each cluster.

Nevertheless, the main problem of these approaches is that they achieve fast results at the cost of reducing their accuracy, giving a large number of false negatives (missdetections due to the termination criteria), especially in very fast detectors as the PPHT (Progressive Probabilistic HT) by Galambos et al. [40]. Besides, they require a large set of application-dependant threshold values to be tuned to obtain adequate results. As final remark, the need of such a number of user-defined thresholds in this type of approaches reduces its applicability to unsupervised systems as well as making them more prone to errors.

2.2.2 Pixel clustering

This group first clusters sets of connected pixels in a coarse manner according to some common property (e.g. their orientation) and afterwards compute a line segment estimate for each set. The work by Burns et al. [22] has been used as reference by further authors such as Khan et al. [56] or von Gioi et al. [119]. Besides, the use of connected component analysis (CCA) was first introduced in this context by Nevatia and Babu [81] and followed by other authors as Khan et al. [56] or Košecká and Zhang [65]. Analogously, Yuen et al. [125] propose a connected version of the HT, which selects ar random an edge pixel and then apply a one-dimensional accumulation on the angle parameter.

In this field, some of the works exploit the information contained in the eigenvalues and vectors of the image tensor matrix [121], as done by many others as Guru et al. [44], Khan et al. [56] or Košecká and Zhang [65]. Typically, the straightness of the group of pixels is measured as a function on the eigenvalues (λ_1, λ_2). However, it is not straightforward to classify a pixel, given its eigenvalues, into these three categories. As mentioned by Rosten and Drummond [96], many authors have created unbounded functions that evaluate the "cornerness" of pixels by the computation of λ_2/λ_1, λ_2, or approximations to the relationship between eigenvalues that skip their computation using directly the elements of the tensor matrix, such as the well known Harris corner function.

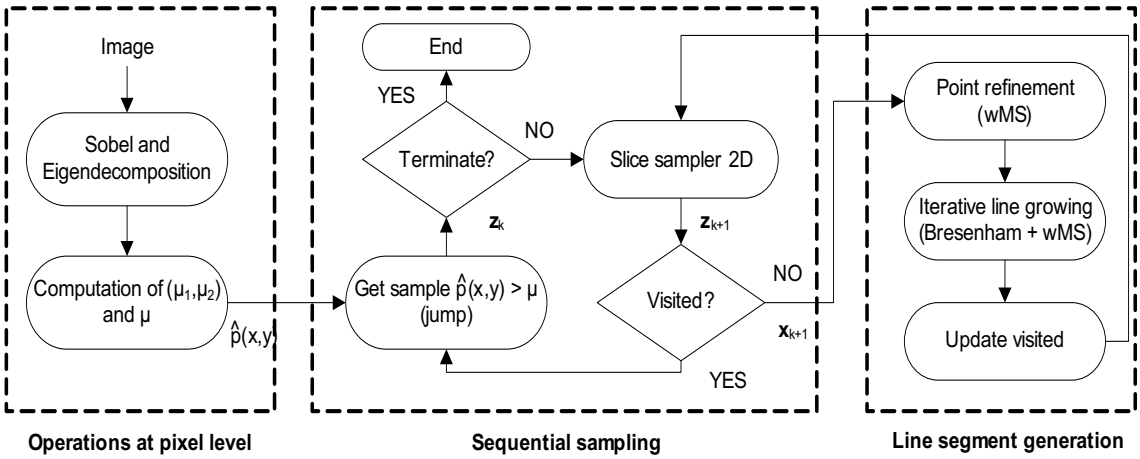


Figure 2.1: Flow chart of the proposed strategy. The image is processed in order to obtain the likelihood distribution $\hat{p}(x, y)$ defined by the parameters (μ_1, μ_2). The mean value of $\hat{p}(x, y)$, μ, is used to generate the first sample of the algorithm and start running the slice sampler, which sequentially generates samples, $\mathbf{z}_k$. The last stage is the line segment generation, in which the samples are refined and used as starting point for the iterative line growing algorithm that finally generates the line segment.

2.3 Approach overview

The proposed strategy shares some properties with methods of the two abovementioned groups, although it can not be classified as belonging to any of them. On the one hand, it uses a sequential sampling strategy, which is an improved version of random sampling approaches applied with the RHT. On the other hand, it exploits the eigenvalues and vectors information as also done by some pixel clustering methods, although in a novel way by the computation of the likelihood function $\hat{p}(x, y)$.

The flow chart of the proposed method (which will be denoted as SSWMS, *Slice Sampling Weighted Mean Shift*), is depicted in figure. 2.1. As shown, it is composed of three main stages: the computation of the likelihood function, the sequential sampling and the line segment generation.

The first step estimates the likelihood distribution over the image I such that $\hat{p}(x, y)$ is the likelihood value of a pixel to belong to a line segment. This distribution is parameterized by (μ_1, μ_2), which are statistics automatically obtained from the image. The distribution is defined using the image tensor matrix, and their corresponding eigenvalues. This process is guided by the idea that pixels that belong to line segments must satisfy two criteria: (i) they have significant gradient magnitude; and (ii) there is only one dominant direction in their neighborhoods. These concepts and the associated methods are described in detail in further sections.

The proposed sequential sampling, which is based on the slice sampling algorithm [16, 78], sequentially selects pixels in the image (denoted as $\mathbf{z}_k = (x_k, y_k)$ for clarity), that are good candidates to belong to line segments, according to the computed likelihood $\hat{p}(\mathbf{z}_k)$. The design of this stage ensures that no repeated samples are selected, so that $\mathbf{z}_k \neq \mathbf{z}_c \ \forall c < k$. If the slice sampler proposes a sample $\mathbf{z}_k$ that has been already visited by the algorithm, the next sample is selected randomly satisfying $\hat{p}(x, y) > \mu$, where μ is the mean value of $\hat{p}(x, y)$ computed for all the pixels of the image.

Given a candidate pixel proposed by the sampling technique, the process arrives to the line segment generation stage. It starts with the point refinement module, which uses Mean Shift (MS) procedures on a multidimensional space composed of the position of the pixels and their dominant local orientation, denoted as $\mathbf{x}_k = (x_k, y_k, \theta_k)$. MS searches for a local maxima on this space, i.e., it looks for the pixel in the neighborhood of the candidate pixel that most likely belongs to a line segment.

Starting from this local maxima, an efficient line growing strategy, based on the Bresenham algorithm [18] is applied to connect pixels until the end points of the line segment are reached. This scheme is applied iteratively to enhance the accuracy of the generated line segments. All the pixels swept by the generated line segment are marked as visited, and thus not available to be selected anymore during the sequential sampling stage.

The overall result of the strategy is that the sequential selection of candidates with the slice sampling algorithm is much more efficient than processing the whole image in search of line segments (as done by most works based on pixel clustering techniques), and even than randomly selecting points on the image without any guiding criteria (as the works about random sampling on the Hough domain). On the other hand, the accuracy on the line segment fitting process is provided by the MS procedure that is used to refine the candidates and generate good starting and ending points for the growing algorithm.

2.4 Sequential Sampling Strategy

In this section we describe the sequential sampling stage, which is based on a general sampling technique: the slice sampling algorithm. Also the computation of the likelihood function for line segments and the initialization and termination processes are presented.

2.4.1 Slice sampling

The slice sampling algorithm [78], is a general sampling strategy, born in the field of inference methods based on numerical sampling (typically known as Markov Chain

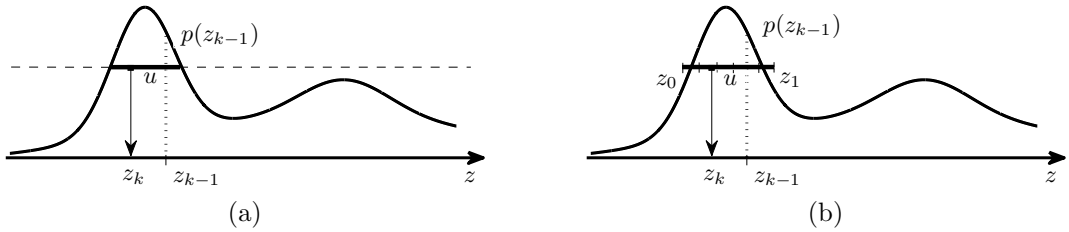


Figure 2.2: Univariate slice sampling: (a) the uniform u value determines the slice through $p(z)$; and (b) the practical implementation uses fixed length steps to determine the range in which z is uniformly sampled.

Monte Carlo techniques) [16]. It allows to sequentially obtain samples, $\{\mathbf{z}_k\}_{k=1}^{N}$ from a arbitrary target pdf, $p(\mathbf{z})$. The only requirement to apply this algorithm is that the value $p(\mathbf{z})$ can be evaluated for any given value of $\mathbf{z}$.

As described in [16], the slice sampling improves the results, in terms of efficiency, of typical sampling approaches based on the Metropolis-Hastings (MH) algorithm [41]. This algorithm (MH) has an important drawback that makes it inefficient for the proposed line segment detector as it is sensible to the step size, given by the proposal distribution. If it is chosen too small, the process behave as a random walk, which makes the algorithm converge very slowly and, on the contrary, if it is too large, the rejection rate may be very high, hence not achieving accurate results. The advantage of slice sampling is due to its ability to automatically adapt its step size according to the characteristics of the pdf.

For a better understanding of how it works, let us first consider the univariate case: $p(z)$. Slice sampling works by augmenting z with an auxiliary random variable u and then sample from the joint (z, u) space [78]. Given the previous sample z_{k-1}, u is uniformly drawn in the range $[0, p(z_{k-1})]$. Fixed u, the sample z_k is obtained from the "slice" through the distribution defined by $\{z : p(z) > u\}$. This criterion is illustrated in figure 2.2 (a). Nevertheless, it is difficult to find the limits of the slice and thus to draw a sample from it. For that reason an approximation is done by means of creating a quantized local slice, delimited by z_0 and z_1 as shown in figure 2.2 (b). To obtain these limits, the value $p(z)$ is evaluated at left and right of z_{k-1} using fixed length steps (the quantification step) until $p(z) < u$. The next sample, z_k, is obtained by uniformly sampling on this range (iteratively until $p(z_k) > u$).

The sampling has to be carried out on a two-dimensional space for line segment detection. The proposed approach is to obtain samples by sequentially applying one-dimensional slice sampling at each dimension, x and y. Figure 2.3 illustrates this procedure, depicting the contour plot of a zoom on a two-dimensional, $p(\mathbf{z})$, function. For a given sample $\mathbf{z}_{k-1} = (x_{k-1}, y_{k-1})$, depicted as a black square, the one-dimensional

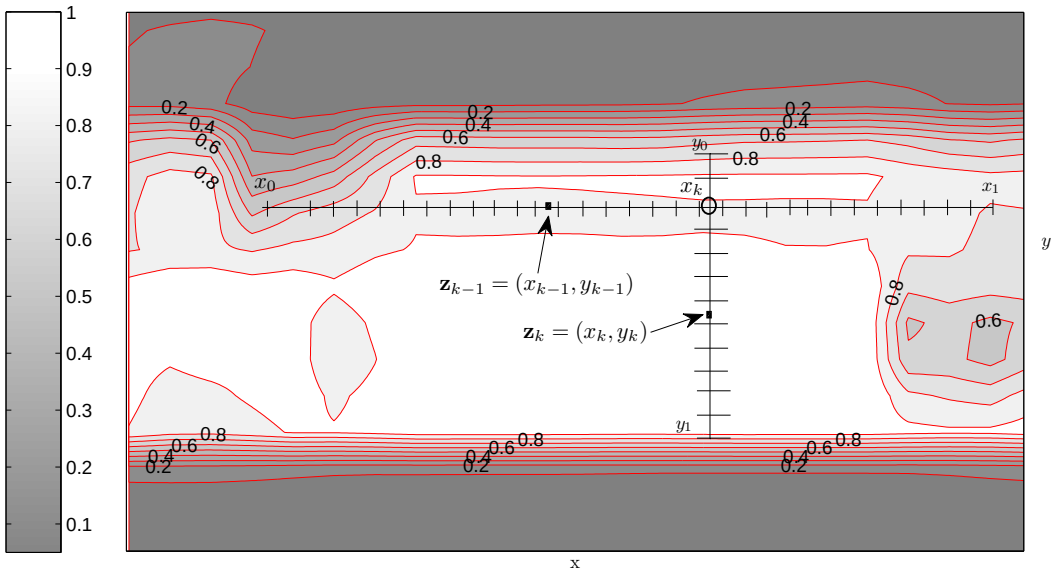


Figure 2.3: 2D slice sampling example. The level set plot allows observing that $\mathbf{z}_{k-1}$ is located in a region with levels between 0.6 and 0.8. The slice procedure on x produces the region limited by x_0 and x_1. The selected random sample x_k is fixed and the region between y_0 and y_1 is obtained for the y dimension. The final sample is $\mathbf{z}_k = (x_k, y_k)$.

slice criterion is applied first on x: y_{k-1} is fixed and a new x_k is delivered. This value is then fixed (represented as a circle) and the one-dimensional procedure is repeated for y. The result is the new two-dimensional sample $\mathbf{z}_k = (x_k, y_k)$. Hence, the proposed 2D slice sampling is a combination of two 1D slice steps, which results in a fast an efficient 2D sampling. As a remark, note that if the order of the one-dimensional sampling is reversed, a different final sample $\mathbf{z}_k$ could be generated, though this fact does not affect the correct performance of the strategy.

2.4.2 Line segment likelihood

The slice sampling algorithm requires the construction of a target pdf (probability density function), which is, in this case, the likelihood of the image pixels to belong to line segments.

Different methods have been addressed in the literature to characterize image pixels to belong to corners, homogeneous regions or line segments. Most of them use the covariance matrix or image tensor matrix for each pixel and their associated eigenvalues, (λ_1, λ_2) with $\lambda_1 > \lambda_2$, and vectors, $(\mathbf{e}_1, \mathbf{e}_2)$. For the sake of clarity the main concepts are summarized in this section, although more details can be found in works as [121].

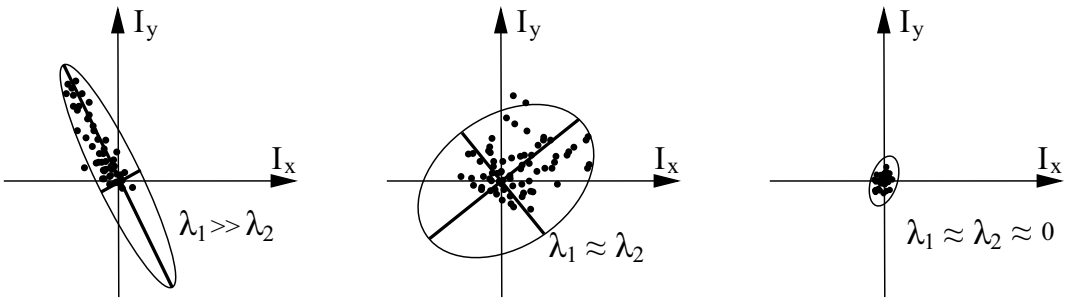


Figure 2.4: Eigenvalues illustrating gradient structures. Black dots are the (I_x, I_y) values of the neighbor pixels of an example pixel that corresponds to: (a) a line segment; (b) a corner or an heterogeneous gradient region; and (c) an homogeneous region.

The image tensor matrix can be defined as:

$$S = \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} \tag{2.3}$$

where I_x and I_y are the partial derivatives of the image at the pixel. Typically, these values are obtained using the Sobel operator, that works as an efficient gradient approximation [38]. Eigen-decomposition is applied to matrix S to obtain the eigenvalues (λ_1, λ_2), satisfying that $\lambda_1 > \lambda_2$, and the associated eigenvectors $(\mathbf{e}_1, \mathbf{e}_2)$. This decomposition can be obtained solving the second order equations given by $\det(S - \lambda I) = 0$, yielding:

$$\lambda_{1,2} = -\frac{S_{11} + S_{22} \pm \sqrt{(S_{11} + S_{22})^2 + 4S_{12}^2}}{2} \tag{2.4}$$

where S_{ij} are the elements of matrix S.

The two eigenvectors, $\mathbf{e}_1 = (A_1, B_1)$ and $\mathbf{e}_2 = (A_2, B_2)$ can be retrieved by, respectively, fixing $A_1 = 1$ and $A_2 = 1$ and solving for B_1 and B_2 as

$$B_{1,2} = -A_{1,2}\frac{S_{11} - S_{22} \pm \sqrt{(S_{11} - S_{22})^2 + 4S_{12}^2}}{2S_{12}} \tag{2.5}$$

and normalizing afterwards so that $\|(A_1, B_1)\| = 1$ and $\|(A_2, B_2)\| = 1$. Since the eigenvectors are orthogonal, $\mathbf{e}_2$ represents the dominant gradient orientation in the neighborhood of the pixel.

The eigenvalues illustrate the dispersion of the gradient along their associated eigenvectors [96] as can be seen in figure 2.4. For that reason, a point that belongs

to a line segment is described by a significant value of λ_1, while λ_2 is close to zero. Analogously, a corner is given by a pair of eigenvalues with similar magnitude both being far from zero, whereas pixels inside homogeneous regions have both eigenvalues close to zero. These cases are depicted in figure 2.4, where the obtained eigenvalues are represented as the axis of the ellipse fitting the gradient distribution of the neighborhood of the considered pixel.

In this work we propose to handle the eigenvalues information by means of a novel function composition, based on the eigenvalues of the pixels of the whole image to generate a function that maps back the likelihood value of a pixel given its particular eigenvalues. The function satisfy two criteria: on the one hand, it returns high values only for pairs of eigenvalues for which one of them is much higher than the other; and, on the other hand, it decreases rapidly when the eigenvalues are both high or low. The function is defined as:

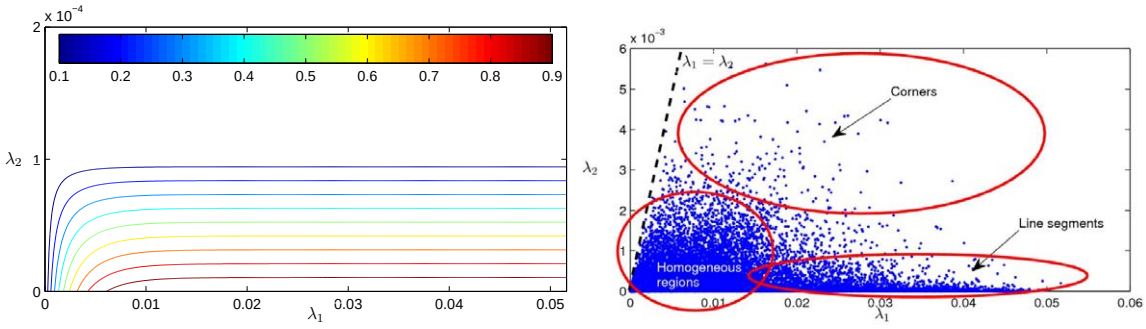
$$\begin{aligned} p = g \circ f : I &\rightarrow (\lambda_1, \lambda_2) \rightarrow \mathbb{R} \\ (x, y) &\mapsto \hat{p}(x, y) = g(f(x, y)) \end{aligned} \tag{2.6}$$

where $f(x, y)$ is the function that gives, for each pixel (x, y), the pair of corresponding eigenvalues (λ_1, λ_2), and $g(\lambda_1, \lambda_2)$ is the function that determines the likelihood of a pair of eigenvalues to correspond to a line segment, defined as:

$$g(\lambda_1, \lambda_2) = \left(1 - \exp\left(-\frac{\lambda_1}{\mu_1}\right)\right) \exp\left(-\frac{\lambda_2}{\mu_2}\right) \tag{2.7}$$

where the parameters (μ_1, μ_2) are, respectively, the average values of the eigenvalues computed over the whole image. The function $g(\lambda_1, \lambda_2)$ is the product of two independent functions on each eigenvalue. The first term, depending on λ_1, penalizes the response for those pixels having their largest eigenvalue too small, i.e., those that likely belong to an homogeneous region. The second term, depending on λ_2, enhances the response for those pixels with a very low smallest eigenvalue, thus penalizing those corresponding to corners, which are described by high values of λ_1/μ_1 and λ_2/μ_2. The result is a good representation of the likelihood of a pixel to belong to a line segment. This function shows high response values for pixels with a significant largest eigenvalue, and a very low smallest one. For one example image, this function is illustrated in figure 2.5 (a), while the associated scatter plot of (λ_1, λ_2) is shown in figure 2.5 (b).

As an example, the $\hat{p}(x, y)$ values of all the pixels in an image have been computed. Figure 2.6 shows, in (a) the original image, which contains large homogeneous regions, significant line segments and corners; (b) shows an scaled representation of the line segment likelihood, where pixels with higher intensity have higher probability to belong to line segments. Note that the corners of the image are represented in dark in (b), while the edges, which are all of them straight, are painted in white.



(a) (b)

Figure 2.5: The level sets of function $g(\lambda_1, \lambda_2)$ shown in (a). In (b) a scatter plot of the set of pairs (λ_1, λ_2) for an example image.

2.4.3 Initialization and termination

The initialization of the algorithm is done by the iterative selection of pixels uniformly along the image until a pixel is found with $\hat{p}(x, y) > \mu$, where μ is the mean value of $p(x, y)$ computed on the whole image. This initialization ensures a good candidate point for the line generator stage of the algorithm. The reason is that the histogram of $\hat{p}(x, y)$ follows typically a decreasing exponential distribution as most pixels of the image do not belong to line segments which form a mode close to zero. So, the selection of starting points above the mean ensures that we are not selecting points that do not likely belong to line segments. In the next steps, slice sampling is applied sequentially, finding automatically pixels with $\hat{p}(x, y)$ similar to the one of the initialization. Each time a line segment is generated, all the pixels that belong to it are removed from the set of pixels that can be selected by the slice sampling algorithm.

A major virtue of this strategy is that typically only one candidate is required to generate a whole line segment, which marks as "visited" all its pixels so that the sampling strategy never draws a sample for that segment again. For that reason, it may happen that the slice sampling algorithm finds out that all the pixels surrounding the previous sample have been marked as visited, so that no more segments need to be detected in that region of the image. Therefore, the initialization criterion is applied again to produce a new start for the slice sampling algorithm, which will be referred to as a jump. Hence this strategy allows to jump naturally from one region to another in the image according to the line segments generated in previous steps.

The proposed scheme sequentially selects the pixels of the image in order of importance, and allows to sweep all the pixels till the end. Nevertheless, it is also possible to determine a termination criteria according to the available computational resources, for example stopping when a maximum number of jumps is reached.

In summary, the sampling strategy sequentially generates samples $\mathbf{z}_k = (x_k, y_k)$,

with an associated orientation, given by $\theta_k = \arctan(B/A)$, with A and B computed as in (2.5). This is used by the line segment generation step to generate the line segments.

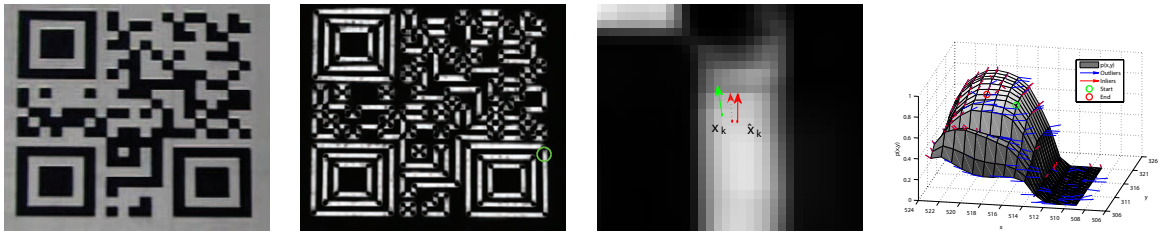
2.5 Line segment generation

For this stage of the algorithm, the sample $\mathbf{z}_k$ and its associated normal orientation θ_k are used to compose the vector, or edge-point, $\mathbf{x}_k = (x_k, y_k, \theta_k)$, which will guide the line segment generation process. This process is composed of three independent stages, as shown in figure 2.1 (point refinement, line growing and update of visited pixels), which are applied on each new candidate $\mathbf{x}_k$. The point refinement step applies a multidimensional weighted Mean Shift (wMS) procedure. It starts on $\mathbf{x}_k$ and searches for a better representative edge-point of the line segment, denoted as $\hat{\mathbf{x}}_k = (\hat{x}, \hat{y}, \hat{\theta}_k)$. The line growing step searches for the two end points of the line segment under analysis. It is based on an oriented local growing algorithm that starts on $\hat{\mathbf{z}}_k$ governed by $\hat{\theta}_k$. The final step marks as visited the pixels that are swept by the line segment.

2.5.1 Refinement of the sample

The probabilistic nature of the slice sampling process delivers samples, $\mathbf{x}_k$, with high value of $p(x, y)$, that likely belong to a line segment. Nevertheless, in their neighborhoods there might be edge-points that better represent the position and orientation of the line segment. The refinement step searches for this representative, denoted as $\hat{\mathbf{x}}_k$, using a multidimensional wMS procedure, inspired on the well known Mean Shift algorithm [30, 45]. Mean Shift is an iterative, non-parametric algorithm that can be used to find local maxima of an unknown density function from which a set of samples is available. At each iteration, MS operates on an starting sample and moves towards the most dense region of the search space in the neighborhood.

MS is used in an innovative way, such that the presented formulation is similar to the one introduced by Comaniciu and Meer [30] but modified to consider additional information of the image: the abovementioned pdf. Therefore, the proposed formulation takes into account two types of information. On the one hand, the orientation of the pixels, which is assumed to be coherent for the pixels that belongs to the same line segment, and, on the other hand, the probability of each pixel to belong to a line segment given by $\hat{p}(x, y)$. The former information (the orientation), is used as in the standard MS, since it moves towards more dense regions of that space (orientation), i.e. towards pixels whose neighbors are more coherent in orientation. The latter information, the pdf $\hat{p}(x, y)$, is not treated like that, since we do not look for coherence in the neighborhood but for the highest possible value of the pdf in the



(a) (b) (c) (d)

Figure 2.6: The original image in shown in (a), and the scaled pdf map in (b). (c) shows a zoom of the highlighted region (located at the bottom right part of the image shown in (b)). The sample $\mathbf{x}_k$ is shown in green, and the refined $\hat{\mathbf{x}}_k$ is shown in red. The dotted vector represents an intermediate iteration of wMS; finally (d) is the 3D representation of $\hat{p}(x, y)$ values of the considered region.

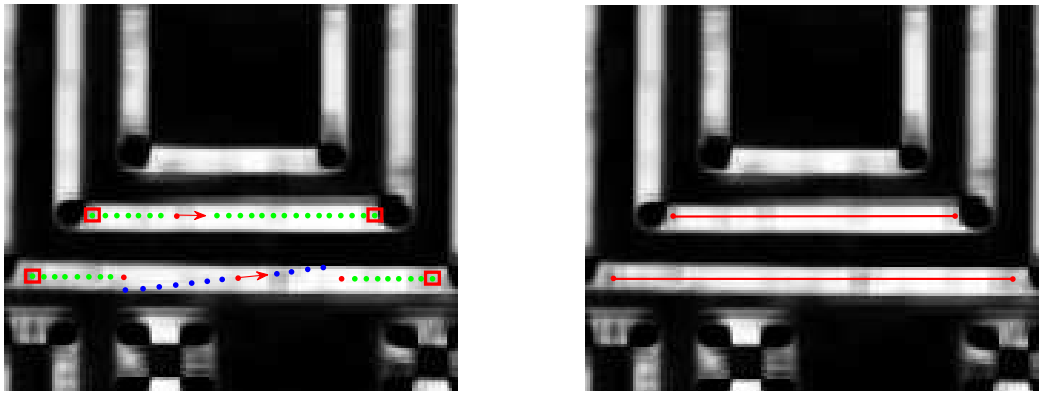
neighborhood. To achieve this combination of information, a wMS procedure is defined, where pixels are weighted by their $\hat{p}(x, y)$ value, such that the relevance of each pixel is given by this weight. The combined result is that the wMS tends to move towards orientation-coherent regions but with high values of $\hat{p}(x, y)$.

Figure 2.6 (d) shows a 3D representation of the values of $\hat{p}(x, y)$ corresponding to the zoomed region shown in (c): the elevation associated to each pixel corresponds to its line segment likelihood level, which is the weight applied during the wMS procedures. As observed, the pixels that belong to the line segment have higher weights, which are the target of the wMS procedures. Details are given in appendix A.

The application of the wMS procedure generates mean shift vectors at each iteration towards the refinement of $\mathbf{x}_k$. An example is shown in figure 2.6 (c). The candidate vector-point given by the slice sampling algorithm is clearly near to a line segment, but the refined version, $\hat{\mathbf{x}}_k$, is much more accurate in position and orientation.

2.5.2 Line growing algorithm

The refined sample, $\hat{\mathbf{x}}_k = (\hat{x}_k, \hat{y}_k, \hat{\theta}_k)$, is used as a starting point for the generation of the line segment. The line segment is delimited by its end points, which are found using an oriented growing strategy inspired on the Bresenham algorithm [18]. This method is illustrated with the example upper line segment of figure 2.7 (a): starting from the edge-point $(\hat{x}_k, \hat{y}_k)$, corresponding to the arrow, the pixels along the orientation $\hat{\theta}_k$ (shown as dots) are connected while their orientations are in the range $\hat{\theta}_k \pm \Delta\theta$. The experiments have shown that the best value for this parameter is $\Delta\theta = \frac{\pi}{8}$, which is a good choice for all kind of images, and it is also proposed by other authors that work with CCA techniques [23, 119]. The extreme points (marked with squares in the figure) of the obtained cluster are the target end-points of the line segment.



(a) (b)

Figure 2.7: Oriented growing algorithm: (a) performance for two different growing cases, one of them with an accurate starting orientation (top), and the other (bottom) with a refinement step that corrects the orientation; (b) the obtained line segments for both cases are correct.

This process is applied iteratively until convergence is reached. For this purpose, an error measurement is defined for the set of points, indexed by i, covered by the line segment at iteration j, computed as follows:

$$\epsilon_i = \frac{1}{M} \sum_{i=1}^{M} \mid \theta_i - \hat{\theta}_j \mid \tag{2.8}$$

where θ_i is the orientation of each point of the line segment, $\hat{\theta}_j$ is the orientation that guides the line growing algorithm and M is the number of pixels connected by the growing algorithm. Note that the maximum error at each pixel is $\mid \theta_i - \hat{\theta}_j \mid \leq \Delta\theta$, hence $0 \leq \epsilon_i \leq \Delta\theta$. The process is considered to have converged when $\epsilon_j \geq \epsilon_{j-1}$, i.e., when the new growing iteration returns a set of pixels that does not fit better to the growing direction θ_j than the previous iteration. The value of $\hat{\theta}_j$ for each new iteration is obtained as the orientation of the segment that joins the refined end-points of the previous iteration. These points are refined as described in section 2.5.1 by means of the application of one wMS procedure to each of them. The process is guaranteed to converge since the number of candidate end-points for a line segment is finite, and so are the orientations $\hat{\theta}_j$, so that there is a global minimum error ϵ_j. Nevertheless, experiments have shown that, in average, the process converges in two or three iterations.

The reason of using this iterative strategy is to compensate potential small deviations between $\hat{\theta}_k$ and the actual line segment orientation. An example is depicted in the lower line segment of figure 2.7 (a). As shown, the application of two new

wMS procedures, one on each end-point restarts the growing algorithm to find a more accurate line segment, achieving convergence at the second iteration. Figure 2.7 (b) shows the obtained segments for the examples shown in (a).

2.6 Complexity of the algorithm

In this section an estimate of the complexity of the whole algorithm is provided. For this purpose let us assume that sums, products, divisions and the rest of basic operations are equally costly (as done by other authors of related works, such as Guru et al. [44]).

The number of operations of the proposed method is approximately proportional to the number of pixels of the image, N. The most time consuming part of the algorithm is the computation of the estimate of the likelihood function $\hat{p}(x, y)$, since it involves the calculation of the eigenvalues and vectors for all the pixels of the image. The operations that are carried out are the calculation of the Sobel approximation to the spatial derivatives $O(16N)$, the computation of the gradient orientation $O(2N)$, the obtention of the eigenvalues $O(10N)$ and the computation of μ, $O(N)$. The result is $O(29N)$ which is proportional to the size of the image.

The rest of computations are done for each line segment and consume much less processing resources. The Bresenham algorithm plus the wMS-based refinement is $O(L + 3R)$, where L is the average length of a line segment, and $R = r \times r$ is the squared spatial bandwidth of wMS. $O(L)$ refers to the computation of the position of each candidate pixel in the growing strategy, and $O(3R)$ corresponds the execution of 3 MS procedures, one for setting the starting point of the growing strategy, and one for each end-point to refine its position. Note that $M(L + 3R) \ll 29N$ for average values of $L = 50$ pixels, $M = 400$ line segments and $N = 600 \times 400$ pixels.

This theoretical analysis is reflected in the results of next sections, especially in table 2.1, where the processing time of the algorithm is evaluated for the detection of different number of line segments.

2.7 Tests and results

This section includes the description of different tests applied on the developed SSWMS algorithm, which are focused on demonstrating its performance in terms of speed, accuracy, and flexibility. A comparison with other related state of the art line segment detectors is also presented.

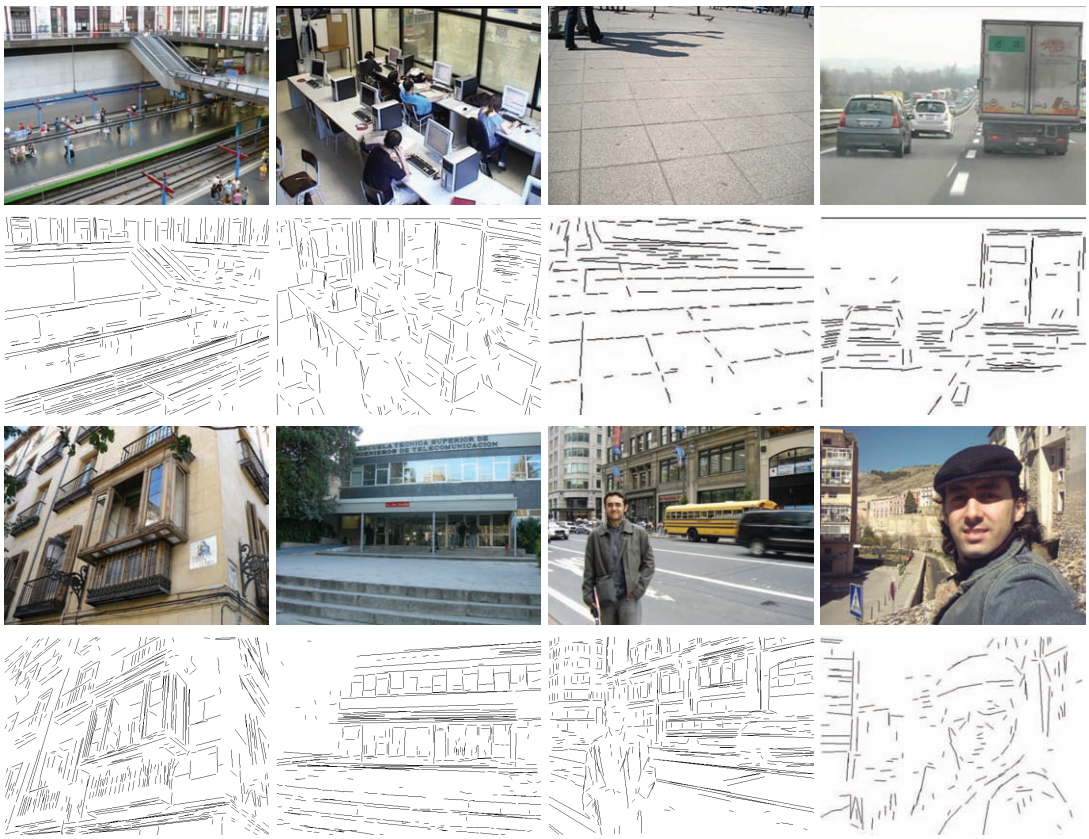


Figure 2.8: Some results of the SSWMS applied on different real images.

2.7.1 Performance analysis

For the tests, a solution in C++ programming language has been implemented, which uses OpenCV v1.1 libraries, running the tests on a Core2Duo processor at 2.2 GHz. A large set of images of several environments, such as buildings, roads, facades, portraits, etc. have been collected and processed, with different sizes, and properties like contrast or brightness. Figure 2.8 shows some of these images and their associated detected line segments. Observe that most of significant line segments in the images are correctly detected, achieving great accuracy and resulting in a very reduced number of false negatives or miss-detections[1]. Furthermore, it is also remarkable that the algorithm generates very few false positives in areas of the image that contain edges but without actual line segments, such as the leafs of the trees, the people or the small elements at the far distance. Ordered from left to right and from top to bottom, the third image shows a very interesting property of the SSWMS thanks to the sequential sampling step: it provides satisfactory results for all the straight edges of the image

[1]Section 2.7.4 shows some numerical values of these statistics.

Table 2.1: Average process time (in milliseconds) for different image sizes and requested line segments.

	Requested number of line segments			
Image Size	50	100	500	∞
180×144	16	16	16	16
320×240	31	31	31	31
360×288	46	47	47	47
640×480	125	125	156	156
1024×720	281	281	313	390
1366×768	391	391	422	563
1280×1024	485	484	531	688
1680×1050	641	656	687	922
1600×1200	703	703	735	1000
1900×1200	812	844	846	1172
3072×2304	2578	2578	2609	3781

even though there are some (the ones corresponding to the shadows) that are much stronger than the rest of edges in the image. This would cause problems in methods that process sequentially based on sorting the edge points of the image according to their magnitude.

Regarding processing times, the algorithm was designed to perform in real-time[2] for medium size images, achieving an average of 125 ms (8 fps) for images about 640×480 pixels and below 50 ms (20 fps) for 360×288 pixels, which are typical image sizes for most online video processing applications. The processing time increases linearly with the size of the images, so that larger ones, with higher resolutions such as 1280×720, 1440×1080 or 1920×1080 require processing times of 430 ms (2.32 fps), 650 (1.5 fps), and 1000 ms (1 fps), respectively.

2.7.2 Discussion about parametrization

It must be considered that the processing times presented before correspond to the execution of the SSWMS until it has searched for line segments along the whole image. The sequential and probabilistic nature of the algorithm allows to stop computing when a requested number of line segments has been obtained. Typically, for most

[2]Real-time in image processing is the ability of a system to process images as soon as they are available from the source, and deliver a result before the next image of the sequence arrives. Considering video sequences, the input frame rate is typically 25 (fps). Therefore, output frame rates below this number are not strictly real-time. Nevertheless, the concept of real-time is used along this thesis in a broad sense, so that frame rates between 5 and 25 (fps) are still considered as real-time.

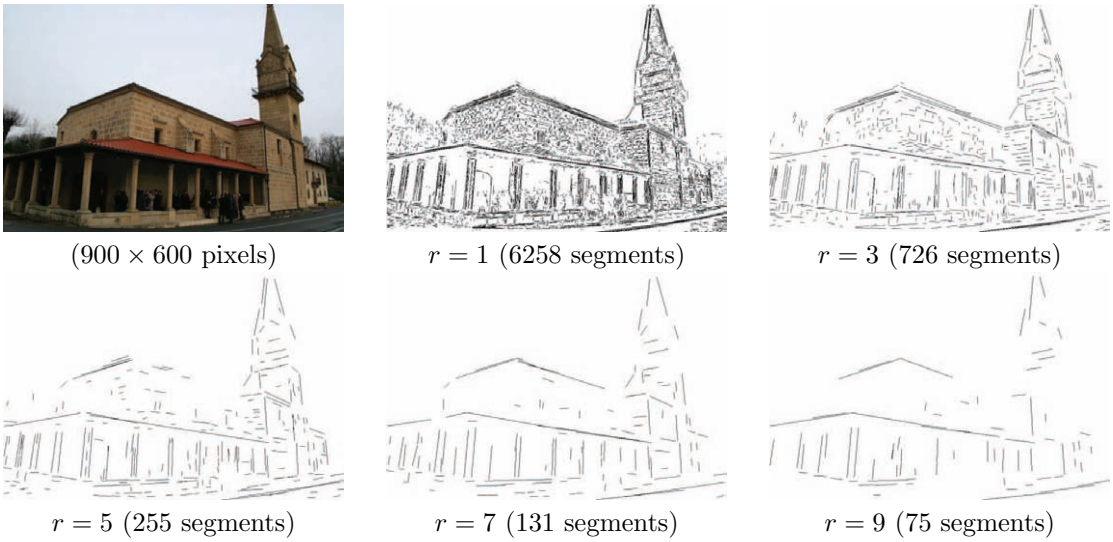


Figure 2.9: Results of the SSWMS line segment detection using different values of the Mean Shift spatial component of the bandwidth, r.

computer vision applications, only the most representative line segments are of utility. Table 2.1 shows the time consumed by the SSWMS for the detection of different number of line segments for different image sizes. On the one hand, the computational time required for images with resolutions below 640×480 do not significantly vary for different target numbers of line segments, as the most consuming part of the algorithm is the computation of the eigenvectors. Therefore, for medium and small images, it is worthy to apply the full search, with no parameters, as the gain of time is negligible. On the other hand, for larger image sizes, the reduction of time when reducing the number of requested line segments is more significant. For instance, the application of the SSWMS on an image with 1600×1200 pixels, would spend near 1000 ms, while applying the SSWMS limiting the search to the first 500 line segments, would result in a reduction of processing time in about 30%. In any case, detected line segments are those ones having more intense values of $\hat{p}(x, y)$, which are typically the most significant lines in the image.

The only parameter of the algorithm that has to be fixed, as mentioned along the paper, is the spatial component of the bandwidth vector of the MS procedures, $h_x = h_y$ that we will denote here as r. By default it is fixed to $r = 3$, which makes the system perform excellent for medium size images. Nevertheless, an even better performance can be obtained by automatically adapting this value according to the size of the images. The bandwidth is an indicator of the resolution of the detector, i.e., the minimum distance between two detected line segments. For that reason, as the images increase in size better results are obtained increasing the size of this parameter.

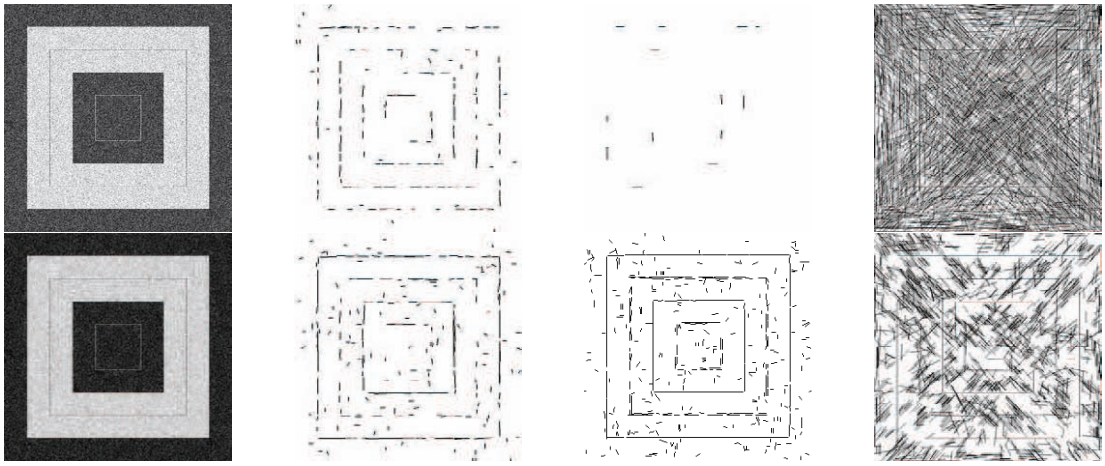


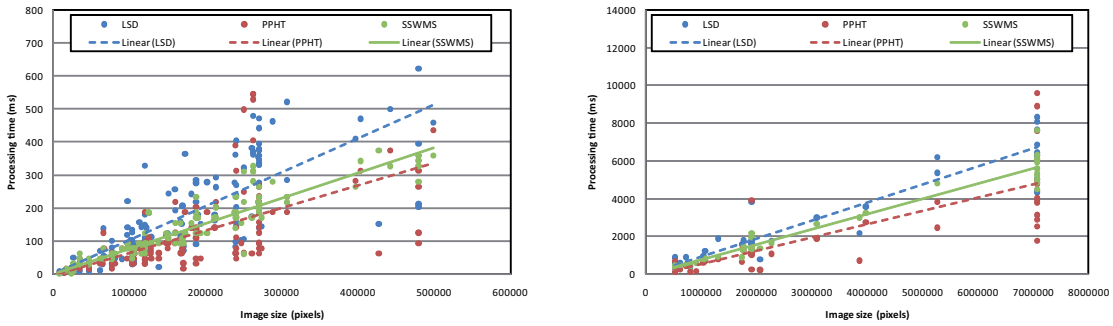
Figure 2.10: Comparison of the performance of the compared methods against noise. The upper row shows the results obtained from the noisy image; and the second row shows the results for the smoothed version of the image.

The results for an example image using different values of r is shown in figure 2.9. As shown, for small values of r, the number of detected line segments is much higher. Increasing this parameter makes the system work towards the detection of dominant line segments. For instance, for $r = 9$, the number of detected line segments is 75, which mostly correspond to the main edges of the building.

2.7.3 Comparison with other methods

Two line segment detection algorithms have been selected to compare with the proposed method: the LSD (Line Segment Detector [119]) and the PPHT (Progressive Probabilistic Hough Transform [40]) for which efficient implementations are available (the authors of LSD offer their implementation in their website, while the PPHT is a very well-known algorithm that has been efficiently implemented as a function inside the OpenCV 1.1 libraries). The motivation of this selection is that this implementation of the PPHT is a very good representative of Hough-based methods as well as, in our knowledge, it is the fastest method in the literature. The LSD has been recently published and it offers linear-time operation and very accurate results, and is selected for comparison as it is the best algorithm derived from the Burns method [22].

The first test was conducted to check the performance of these algorithms in the presence of noise. An example image, with significant added noise and the detected line segments obtained with the different methods is shown in figure 2.10. The upper row of this figure illustrates the robustness of the SSWMS against noise: the LSD



(a) (b)

Figure 2.11: Process time of the SSWMS, PPHT and LSD methods for more than 200 images with different sizes: (a) images up to about 600×900 pixels; and (b) larger images up to 3702×2304 pixels. The colored lines represents the linear tendency of the data along the image size axis. As shown, the faster method is the PPHT, while the SSWMS shows higher performance than LSD in both cases.

is unable to detect segments, the PPHT produces hundreds of noisy line segments, while the SSWMS detects, partially splitted, the most important line segments in the image. In the second row, a Gaussian filter is used to remove partially the noise, as suggested by von Gioi et al. [119] to improve its accuracy in the presence of noise. In this situation, both SSWMS and LSD show similar performance, while the PPHT still generates too many noisy segments.

Figure 2.11 shows a scatterplot of the processing time of the proposed approach against the size of the image (in pixels) compared to the PPHT and LSD algorithms: (a) shows the comparison for small and medium size images, and (b) for large images. The PPHT is in average 10% faster than SSWMS considering the whole set of images used for the comparisons (more than 200 images), while the LSD is between 10% (for large images) and 30% (for small and medium size images) slower than the SSWMS. Note that for the comparison we had to tune the parameters of the PPHT for each image independently to obtain the best results in terms of speed and accuracy (such as the resolution of the transform space, the minimum vote threshold, and minimum length of the line segments), while the SSWMS needs no parameter tuning. The accuracy of these methods is illustrated, for different images in figure 2.12. As shown, the PPHT shows the higher number of false detections and missdetections. The LSD offers very good results for almost all images, specially those that have high-contrast edges. Nevertheless, for some images, such as the one showing the ground (first row in figure 2.12) it fails to detect important line segments, and it produces too many line segments, retrieving multiple replied segments that add no information about the scene. In all these cases, the SSWMS shows excellent performance, detecting the most important line segments with a very reduced number of false detections and

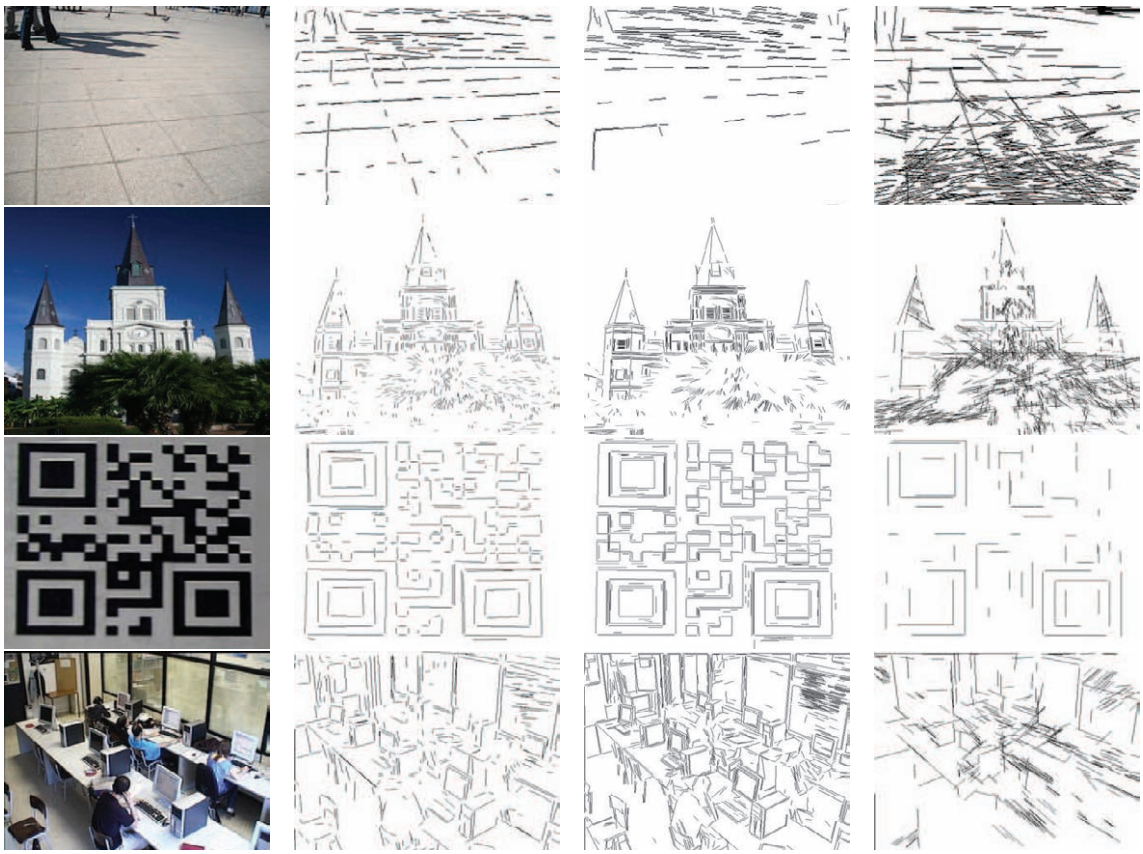


Figure 2.12: Comparisons of the line segments obtained for different images applying different methods.

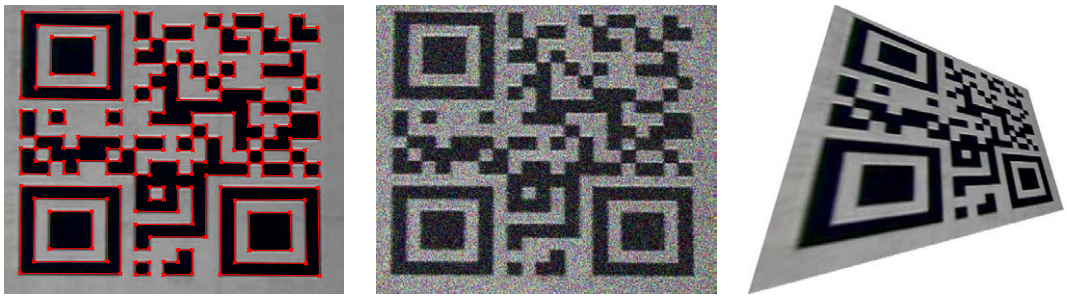
missdetections.

2.7.4 Recall and precision

To compare the performance in terms of recall and precision of the three considered methods the ground truth line segments of a real image has been created, as shown in figure 2.13 (a). The recall and precision (RP) evaluation metrics can be computed as:

$$\text{Recall}(\%) = \frac{\#\text{ Correct detections}}{\#\text{ Ground truth line segments}} \tag{2.9}$$

$$\text{Precision}(\%) = \frac{\#\text{ Correct detections}}{\#\text{ Total detections}} \tag{2.10}$$



(a) (b) (c)

Figure 2.13: Example image used to test recall and precision: (a) original image with ground truth line segments; (b) image with added gaussian noise; and (c) image with perspective distortion.

This way, recall is related to the number of missdetections, and precision to the number of false alarms. In this context, a detected line segment is considered to be a correct detection if it is similar to an existing ground truth line segment in terms of relative orientation, distance between mid-points and length. The RP values have been obtained for this image and for a number of modifications of it. The results are shown in figure 2.14, where the values corresponding to the original image are shown with a thick black edge[3]. The modifications of the original image are done in three different ways, designed to evaluate the performance of the methods in typical conditions of real images. The graph showing "variable size" depicts the RP values obtained increasing the size of the image. It evaluates the response of the methods against variations of the length of the line segments and the distance between them; "variable noise" stands for the addition of Gaussian noise with standard deviation values ranging from 0 to 0.07 in 0.01 steps. An example image with noise is shown in figure 2.13 (b). The introduction of noise shows how flexible are the algorithms to provide good results when the spatial coherence of pixel information (such as gradients) is lost or corrupted due to the noise. Finally, "variable perspective distortion" illustrates the results obtained by applying perspective transformations on the image, such as the one shown in figure 2.13 (c). This type of transformation implies a non-homogeneous modification of the image area. Hence, the length of the line segments is non-uniformly modified. Therefore, this test illustrates the ability of the methods to compute long and short line segments in the same image.

The results related to the modification of the size of the image indicates that the SSWMS renders very accurate results with very low missdetections and false alarms,

[3]SSWMS RP: 91.73/95.35; LSD RP: 90.35/73.33; PPHT: RP: 38.19/89.91. These numbers mean that both the LSD and SSWMS offer great results for this image, although the LSD delivers more false alarms. The PPHT suffer more missdetections and thus its recall value is very low.

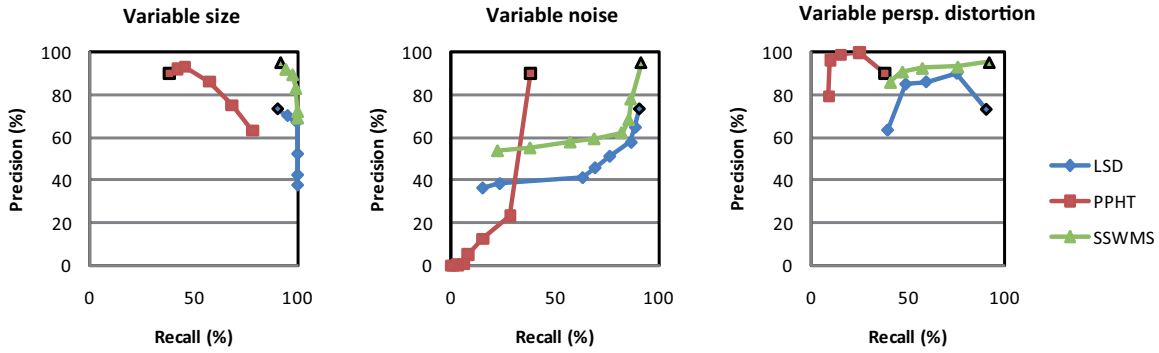


Figure 2.14: Recall & Precision graphs for different situations: varying size, adding noise and perspective distortion for the three methods. The starting point is marked with a thick black edge, and is the same for the three graphs.

while the performance of the LSD decreases as it provides more false alarms as the size of the image increases. The recall of the three methods increases since it is easier to detect all the existing line segments as the image increases.

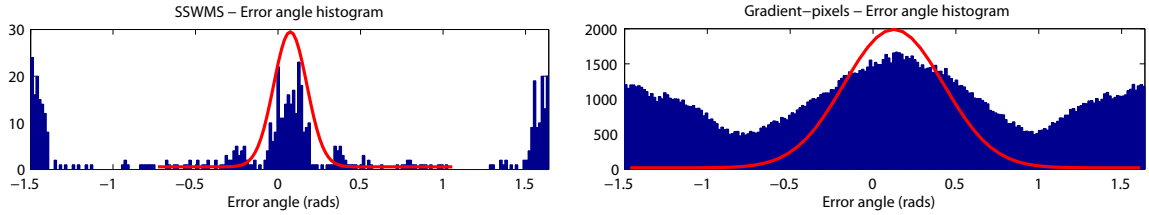
The addition of noise, as already commented in the example of figure 2.10, is very harmful for all methods. Therefore, RP curves tend to decrease both the recall and precision values, although to a lesser extent for the SSWMS algorithm, which still shows RP above 50/50 for added noise with 0.05 standard deviation.

The results corresponding to the images with perspective distortion show that the recall values decrease as the number of line segments whose length is reduced grows. Nevertheless, the SSWMS algorithm still keeps a very good value of RP especially compared with the PPHT, which is not able to detect a large number of line segments due to the distortion.

2.8 Example application

As a link to the following chapters, this subsection justifies the use of the SSWMS algorithm and explains its advantages for the computation of vanishing points compared to the use of gradient information at pixel level provided by a simple edge detector.

The vanishing point detection strategies that are described in chapter 3 are formulated as optimization processes that estimate the values of the parameters of the vanishing points that minimize a defined error function with respect to the set of data obtained from the image. The data samples can be any feature that has orientation, such as gradient-pixels (pixels with enough intensity gradient magnitude) or line segments. The equivalence comes from the fact that both features can be expressed



(a) (b)

Figure 2.15: Error angle histograms: (a) line segments obtained with SSWMS; and (b) gradient-pixels

similarly as a line in homogeneous coordinates (for details about homogeneous coordinates, see Hartley and Zisserman [46]). Any gradient pixel, defined by a position and a gradient vector $\{\mathbf{r}_i, \mathbf{g}_i\}$, can be expressed as a line $\mathbf{l}_i = \mathbf{r}_i \times (\mathbf{r}_i + \lambda \mathbf{m}_i)$, where $\mathbf{m}_i$ is the the normal to the gradient vector, and λ is any real number. Analogously, any line segment is described as a pair of end-points $\mathbf{l}_i = \mathbf{a} \times \mathbf{b}$, where the expression of these points is augmented into homogeneous coordinates as $\mathbf{a} = (a_x, a_y, 1)^\top$ and $\mathbf{b} = (b_x, b_y, 1)^\top$.

Regarding the number of data, it has been shown that generic sparse sets of edges are more accurate than gradient maps for the whole image [37]. This way, the use of gradient pixels for vanishing point estimation (which is reasonable for many applications and platforms since the Sobel operator is typically easy to implement in an efficient way), is subject to the introduction of a sub-selection stage that selects just a few pixels of the image. Otherwise, robust approaches that compute vanishing points, such as RANSAC or EM, would suffer a great decrease of their performance due to the excessive number of data sample to handle in their iterative procedures. As shown in this chapter, the number of line segments delivered by the SSWMS algorithm varies according to the bandwidth value or the size of the image. Though, in general, this number ranges from 50 to 500 line segments, which are perfectly handled by the subsequent vanishing point strategies.

The comparison in terms of error distributions is supported by the following experiment, which use the public set of images of the York Urban Database (YUDB)[4] [37], for which ground truth vanishing points and calibration is available. For each image of the YUDB the SSWMS has been run and an evaluation is obtained about the orientation error of the obtained line segments with respect to each vanishing point. This error is related to the angle between the line segment and the line that joins the vanishing point and the mid-point of the line segment[5]. Figure 2.15 (a) shows an example histogram of the error of the detected line segments with respect one vanishing

[4]More details about this database are given in chapter 3.
[5]Chapter 3 enters into details about this error function.

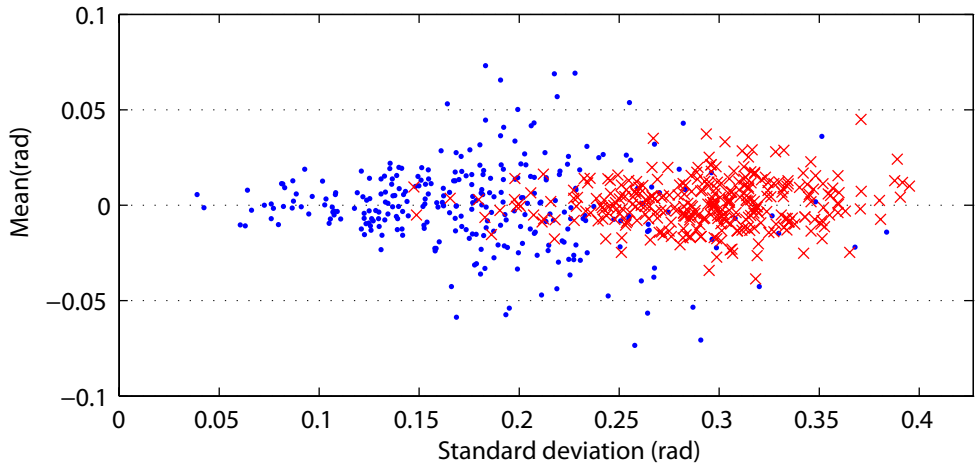


Figure 2.16: Scatter plot of the mean and standard deviation of the normal fit of the inliers mode of the error histogram. The results of the normal fit of the line segments obtained with SSWMS is shown in dots. The normal fit corresponding to gradient-pixels is shown with crosses. Realize that there are 306 data points, since each image of the YUDB contains three vanishing points.

point. The mode around zero correspond to the error of the line segments actually meeting at the vanishing point. The rest of the histogram corresponds to line segments not meeting that vanishing point. As shown, the peak can be characterized with a normal distribution with a mean close to zero and a low variance.

The actual error with respect that vanishing point of the image information can be characterized by the histogram of the orientation error of the gradient-pixels of the image. This histogram is shown in figure 2.15 (b). As can be observed, the error distribution is more peaked for the line segments, which illustrates that the use of SSWMS reduces the error in the orientation and thus increases the accuracy of the information that is used to compute vanishing points. This experiment is repeated for all the vanishing points of all the images of the YUDB, obtaining mean and standard deviation values. The result is illustrated in figure 2.16, which shows a scatter plot of the obtained statistics, showing with dots the results for the SSWMS line segments, and with crosses the results of the gradient-pixels. As can be observed, the distribution of error of the SSWMS is more concentrated on zero mean values, with lower standard deviation. The reason is simple: one long line segment gets less orientation error than its corresponding set of gradient pixels, since it has been generated with a robust growing algorithm.

Therefore, the SSWMS can be said to be a wise tool to select oriented features of the image, without redundancy, and with less orientation error with respect to vanishing points. As a conclusion, it is always better to use the SSWMS rather than

simple gradient pixels obtained with the Sobel operator.

2.9 Conclusions

In this chapter a novel approach for accurate and fast detection of line segments in images has been proposed, which is specially devoted for real-time vision applications. It has been designed to adapt itself to the characteristics of the images, hence there is no need to tune any input parameter. It is based on a sequential sampling strategy, which uses the slice sampling algorithm to draw pixels in the image that likely belong to line segments. At each drawn sample, an iterative line growing strategy, based on Mean Shift is applied which finally finds the end-points of the line segment.

The outstanding performance of this strategy has been demonstrated in terms of accuracy, flexibility and, mainly, speed. A comparison with other line segment detection methods has been carried out. It has shown that the proposed method offers an excellent trade-off between very accurate methods and fast ones. One of the major abilities of the proposed algorithm is that it finds the most important line segments for any type of images, including noisy ones, without need of tuning any parameter.

Chapter 3

Detection and tracking of vanishing points

3.1 Introduction

This chapter introduces the proposed methods in the field of vanishing point estimation. The first section presents an extended state of the art, which classifies existing works attending to different aspects and helps to identify the most significant contributions of the methods that are described in this chapter.

The presented contributions are classified in three sections: (i) the definition of an error function and its comparison with other existing approaches; (ii) the design of robust methods based on modifications of the RANSAC algorithm; and (iii) optimization algorithms for simultaneous estimation of lines and vanishing points based on the EM algorithm.

3.2 Related work

Next subsections describe the most relevant methods for vanishing point estimation found in the literature. They are described and classified according to several aspects. In first place, the workspace on which the methods are applied: detecting vanishing points means obtaining the values of the parameters that characterize these points. Therefore, the parametrization determines the workspace, or analysis space on which algorithms search for vanishing points. Secondly, the image information that is used, since different image features can be used to determine vanishing points, such as lines, line segments, edges or gradient-pixels. Another important criterion is the type of approach. The detection of vanishing points requires two steps, clustering or grouping,

which refers to the selection of image features that meet at a common vanishing point, and estimation, which, given a group of image features, determines the position of the common vanishing point. Nevertheless, these steps can be done in different ways, and thus a number of different methods have arisen in the literature. Finally, the number of vanishing points and the assumed relationships between them. For instance, it is quite typical to search for tuples of mutually orthogonal vanishing points.

As a summary, at the end of this section, table 3.1 classifies the most relevant works and highlights their research contributions.

3.2.1 Workspace

Vanishing points can be described mathematically in different ways, since they, on the one hand, are 2D image entities, but on the other hand, correspond to 3D space directions. Therefore, it is appropriate to carry out a first classification of techniques according to the type of parameterization they use. This formulation guides the selection of methods and tools that are discussed further on.

Image plane

Some authors have chosen the image plane itself as work space, such as Caprile and Torre [25], McLean and Koyyuri [75], Sekita [101], and more recently Minagawa et al. [76], Suttorp and Bücher [111]. For instance, Suttorp and Bücher [111] put forward that vanishing points in road scenes typically lie inside the limits of the image, so that it is enough to work on the image plane to estimate their positions. The concept of "vanishing hull" was introduced by Hu et al. [50] as the intersection of the fan shape regions defined by an edge-point given a edge error model. Not restricted to one single vanishing point, other authors, such as McLean and Koyyuri [75], Sekita [101], or Minagawa et al. [76] define error distances on the image plane to perform optimization processes to estimate the position of the vanishing points. Concretely, Minagawa et al. [76] use the EM algorithm and the normal distance between the vanishing points and the lines that contain the line segments.

Gaussian sphere (unit sphere)

Barnard [12] proposed to project the image plane, which is, by definition, an unbounded space, on the unit sphere (also known as Gaussian sphere[1]), centered at the optical center of the camera. The main target was to obtain a bounded space on which operations are simplified and thus finite and infinite vanishing points are treated equally. The sphere has been used, in different ways, by many authors, such us Barnard [12], Magee and Aggarwal [73], Quan and Mohr [94], Collins [29], Brillault-O'Mahony [19], Lutton et al. [71] or Shufelt [105].

[1]The concepts related to the unit sphere are explained in appendix B, as they help to understand some parts of this thesis.

The image plane information is mapped into the sphere by means of the projective transformation given by the calibration matrix of the camera. The image plane information is then said to be calibrated and normalized, i.e. the information is mapped into a sphere located at the optical center of the image, with a radius of one, and the Z-axis of the sphere is aligned with the optical axis. For this reason, the works in the literature that make use of the sphere cannot be applied to calibrate a camera through the detection of vanishing points (which turns out to be the most typical application of the detected vanishing points).

Nevertheless, Košecká and Zhang [65] shown that it is not strictly required to know the exact calibration of the camera to detect vanishing points, as the intersection between lines is invariant to projective transformations. Even so, this projection is not coherent with the calibration matrix of the camera that acquired the image, so it is not possible to infer geometric properties of the scene from the vanishing points (details are given in appendix B).

Most of former works [12, 73, 94, 71] make explicit use of the unit sphere using its surface, appropriately tesselated as accumulation space to search for maxima that determine the presence of dominant vanishing points. Further works that use this calibration (Košecká and Zhang [65], Antone and Teller [4]), rather than accumulate, use the probabilistic EM framework on the projective plane, which is equivalent to work on the unit sphere, as explained in the next subsection.

Projective plane

As an alternative to the image plane, and as a mathematical formalization of the unit sphere parametrization, emerged the works that consider the projective plane to analytically treat infinite points in a natural way (David [36], Kanatani [55], Liebowitz [68], Pflugfelder [93], Rother [97]). Analogously to the unit sphere, this search requires to augment the dimension of the vanishing points by considering the homogeneous coordinates representation.

These works constructed over previous approaches, which were based on the sphere, proposing error distances that do not depend on the position of the vanishing point. Recently, Rother [97] stated that distances defined on the image plane are not maintained after the projective transformation into the sphere, so that, for example, if two line segments undergo a rigid movement on the image plane, the distance between their great circles[2] is not constant on the sphere. For Pflugfelder [93], this fact ruled out the sphere as search space in favor of the projective plane.

Other spaces

Other authors have proposed alternative transformed spaces to handle infinite vanishing points [24, 102, 117]. Tuytelaars et al. [117] defined an intelligent partition of the parametric line space (computed as the Hough transform) by considering three

[2]The concept of great circles is also explained in appendixB

bounded sub-spaces that were used to detect vanishing points and also vanishing lines, and similar approaches were proposed by Almansa et al. [2] and Seo et al. [102]. The work by Cantoni et al. [24] focused on the polar space obtained as the polar Hough transform of the points on the image. In this space, finite vanishing points are represented as sinusoids, which are estimated through linear least squares optimization.

3.2.2 Image information

Computing vanishing points requires to obtain image features that indicate an orientation or a line, so that groups of image features determine sets of lines that might meet at a common vanishing point. This way, putative image features are, obviously, lines (that can be obtained easily with the Hough transform, or some of its variants, like the PPHT [40] mentioned in chapter 2), line segments, or simply gradient-pixels[3].

Line segments are the image features most typically used in the literature. Barnard [12] already used them, as well as Collins [29] that assumed the availability of clusters of line segments meeting common vanishing points. Many other authors have used line segments, such as Brillault-O'Mahony [19], Shufelt [105], Antone and Teller [4], Rother [97], Liebowitz [68], Košecká and Zhang [65], Almansa et al. [2], Hu et al. [50], Seo et al. [102], Suttorp and Bücher [111], Trinh and Jo [115], Kalantari et al. [54] and Pflugfelder [93]. Other works use instead points (Tuytelaars et al. [117], Minagawa et al. [76], Cantoni et al. [24]), edges or gradient-pixels (Schindler and Dellaert [99], Barinova et al. [11]), lines (McLean and Koyyuri [75]) or texture (Ribeiro and Hancock [95]).

3.2.3 Type of approach

Vanishing point estimation strategies are typically composed of two fundamental steps, denominated clustering and estimation. The former takes care of the classification of the image features into groups or clusters sharing a common vanishing point. The latter considers the information of the group and estimates the position of the vanishing point.

Nevertheless, these two steps can be carried out in many different ways, which can be classified first between iterative and non-iterative approaches. Furthermore, iterative strategies can be as well subclassified into robust (based on RANSAC) and optimal approaches (based on EM).

In the following subsections, some of the most relevant approaches in the field are

[3]This name refers to pixels for which their associated gradient vector has been computed for instance, with the approximation given by the Sobel operator. Other names for the same concept are edge-pixels, edgels, edge-points, or simply edges.

described according to the abovementioned taxonomy.

3.2.3.1 Clustering and estimation

In former works, these two steps have been carried out separately using different nature techniques: clustering methods can be coarsely divided into those that use traditional clustering algorithms (such as k-means), and those that use accumulation spaces on which search for maxima. The estimation stage requires the definition of a cost or error function between the vanishing point and the image features, which is typically minimized using optimization methods.

For the clustering step, former works on vanishing point detection followed the proposal of Barnard [12], which used an accumulator defined on the unit sphere in a similar way to the Hough transform. The idea is to divide the sphere into accumulation cells and increase the votes of a cell according to the number of great circles that pass through it. Once the voting process is finalized, cells with higher number of votes correspond to potential vanishing points. There are authors that have used this procedure to estimate the position of the vanishing points as the position of the cells with higher votes [12, 71, 94, 105], although the accuracy these methods obtain depends on the resolution of the quantization.

Barnard [12] realized that a regular quantization of the cells in θ and ϕ on the sphere makes that each cell corresponds to regions of the image with different areas. Therefore a post-processing stage is applied on the count of the cells to reduce the impact of this effect. Other authors insisted on this topic, trying to reduce the committed error associated to the size of the cells [71, 73]. Specifically, Lutton et al. [71] proposed a semi-regular quantization (regular in θ and irregular in ϕ). In the same way, Quan and Mohr [94] proposed a complex scheme, which used a hierarchical Hough transform to find a trade-off between the computational cost and the accumulator resolution. Alternatively, Shufelt [105] uses the regular quantization proposed by Barnard, but introduces a swath error model on the line segments. This error model compensates the problem of regular cells and permitts much more accurate results.

Tuytelaars et al. [117] explicitly uses the Hough transform, in one of the most interesting works about accumulation spaces, since not only the position of vanishing points and the vanishing lines that join them is computed, but also an intelligent transformation that manage infinite points is proposed: a cascade of three consecutive Hough transforms is applied, which is used to transform from edge points to lines, from lines to intersection (vanishing) points, and then vanishing lines. Nevertheless, as a bounded space, this strategy undergoes the same problem as the sphere accumulators: cells more distant to the center of the defined Hough spaces cover larger and larger regions of the image plane.

Perhaps, the most outstanding work of the last decade corresponds to Almansa

et al. [2]. A partition of the image plane is proposed in such a way that each cell of the accumulator gathers the same probability that an image line passes through it. This approach was shown to be very robust against outliers without any prior knowledge of the scene.

As an alternative to accumulation spaces, some authors have used a variety of clustering techniques. Košecká and Zhang [65], McLean and Koyyuri [75] and Seo et al. [102] presented similar analysis of the orientation histogram of the image line segments to obtain initial clusters on which operate. Previously, Magee and Aggarwal [73] computed the complete set of intersection points between all pairs of line segments in the sphere to search clusters using k-means. Analogously, Barinova et al. [11] introduced Mean-Shift as a non-parametric clustering tool, while Ribeiro and Hancock [95] used a texture-based transformation on which vote using spectral components on the sphere.

Given the clusters of image features, the estimation of vanishing points requires the definition of a cost or error function. This function illustrates, under any parametrization, the error that exists between a vanishing point and an image feature that is expected to pass through that vanishing point. Hence, the residual error of a hypothesized vanishing point with respect all the elements of the cluster can be computed and minimized to search for the optimal vanishing point.

One of the first approaches using error functions is the one of Caprile and Torre [25], which does not define an explicit error function but obtain the position of the vanishing point as the mean of the set of intersections of all pairs of line segments in the cluster. McLean and Koyyuri [75] introduced the first approach based on optimization methods. After clustering, the estimation of the vanishing point is carried out as a least squares problem defining the error function as the normal distance of the great circles with respect to the vanishing point and a gradient descend method to minimize the error. The approaches of Brillault-O'Mahony [19], and more significantly Shufelt [105], introduced the concept of error models for line segments. Hence, each line segment is understood as the result of a noisy image processing technique. Shufelt proposed to estimate the vanishing point by modeling the line segments as swaths of the sphere, with integer-pixel accuracy.

The point-line distance between the vanishing point and the line defined by the line segment has been used in the image plane by some authors [55, 76, 101] as cost function. Nevertheless, as pointed out by Pflugfelder [93], this type of error distances has no geometric sense, as it depends on the position of the vanishing point. To overcome this drawback, alternative error functions, defined on the projective plane have been proposed [68, 97]. The strategy by Rother [97], and a similar one by David [36], uses the error function defined as the squared sum of the distances between the end-points of line segments and the line formed by the vanishing point and the mid-point of the line segment. However, Liebowitz [68] explains that this method is not optimum under the

consideration of bivariate Gaussian noise in the position of the end-points. He proposes the analytic optimal solution, for which it is necessary to employ optimization tools for non-linear least squares, such as the Levenberg-Marquardt algorithm [46].

3.2.3.2 Joint strategies

Many authors have converged to the use of iterative joint strategies, which contain the grouping and estimation stages as an alternating process. Specifically, the most used techniques are those based on robust estimation tools, such as RANSAC or its variants, and optimization strategies based on mixture models using the EM algorithm.

The methods based on RANSAC (RANdom Sampling And Consensus) [54, 93, 97, 115], work iteratively by the selection of a sufficient set of image features, which are used to compute a candidate vanishing point and then find the set of features that are coherent with it. Once this consensus set has been determined, the vanishing point can be re-estimated considering the information of all the elements of the group. To determine more than one vanishing point, Pflugfelder [93] removed the line segments that were used to compute one vanishing point and restart the RANSAC iteration searching for new vanishing points. The procedure is repeated until there are not enough line segments to determine new vanishing points.

The EM (Expectation-Maximization) algorithm has been shown to be a very powerful tool for recursive estimation and clustering. As opposite to RANSAC, EM allows assigning each line segment to different vanishing points with a particular probability. The iterative process runs until the position of all vanishing points (whose number must be provided initially) have been determined [4, 36, 65, 76, 93, 101].

The work by Schindler and Dellaert [99] is noteworthy, as an EM model with on-off mixture models for edge points on the projective plane is proposed. This work does not require additional information from the scenario (which is denominated as Atlanta world) in terms of the relationship between detected vanishing points, as opposite to many works that search for the three vanishing points that corresponds to the three orthogonal directions of the Euclidean space [71, 94, 97], typically known as Manhattan worlds [32].

However, there exists a number of problems that shall be solved in order to use the EM algorithm. First, it requires the prior knowledge of the number of vanishing points to use a mixture model adapted to the problem. Košecká and Zhang [65] obtains this information through an initial clustering stage based on orientation histograms. In second place, the EM algorithm is very sensible to the initialization, as any multidimensional space minimization technique. If the positions of the vanishing points are initialized far from the target minima, the EM is prone to get stuck in incorrect local likelihood minima. Antone and Teller [4], Košecká and Zhang [65] and Minagawa et al. [76] used the Hough transform as initialization method for their EM-based strategies.

3.2.4 Number of vanishing points

The number of vanishing points that are obtained by each approach is strongly related to the purposes of the computer vision application that use them. For instance, methods that compute vanishing points for camera autocalibration need, at least, three vanishing points, while rectification of planes just require two.

This way, there are methods that fit models assuming the existence of a single vanishing point [24, 29, 68, 75], although several vanishing points can be obtained by applying sequentially these approaches with an appropriate filtering of the image feature set (i.e. the set of image features used to compute the first vanishing point are removed from the complete set in order to compute the next vanishing point). Nevertheless, other approaches do actually consider multiple vanishing points in their models, so that they are computed simultaneously [4, 50, 54, 65, 76, 99, 102, 117].

These models can be additionally adapted to consider assumed relationships between vanishing points. For instance, the Manhattan world hypothesis by Coughlan and Yuille [32], take advantage of the assumption that there are three vanishing points and that they are mutually orthogonal. Other works have constructed over this approach, as the Atlanta world by Schindler and Dellaert [99], which assumes the presence of several independent sets of three mutually orthogonal vanishing points.

Typically, the orthogonality assumption is done for autocalibration purposes, since three such vanishing points contain enough information to retrieve the image of the absolute conic (IAC) and thus the camera calibration matrix [68].

3.2.5 Conclusions

Several conclusions can be obtained from the analysis of the state of the art, which guide the contributions of this thesis in the field of vanishing point estimation.

Although this has been a major topic in the computer vision research community, there are some aspects that still require novel contributions. For instance, in the opinion of the author, a more deep evaluation of the existing error functions shall be carried out, to compare their performance inside optimization procedures. Some authors propose their own error functions doubting about previous approaches, but without entering into more specific considerations. For instance, Rother [97] and Tardif [112] point out that the hypothesis of Liebowitz [68][4] is not justified, but without actually comparing their performance. Therefore, in this thesis, the new error function that is proposed (that hypothesizes a different noise model), is compared in terms of performance with existing methods in the literature, showing its advantages and disadvantages.

[4]Gaussian noise in the position of the end-points of line segments.

Ref.	Year	Image features	Workspace	Contribution	Num. Vps.
Barnard[12]	1983	LSegs.	Sphere	Hough accumulation on sphere surface	3
Magee[73]	1984	-	Sphere	Search for clusters of intersections.	-
Quan[94]	1989	-	Sphere	Hierarchical Hough transform	-
Caprile[25]	1990	Vps.	Image plane	Vps. matching	-
Collins[29]	1990	Clustered LSegs.	Sphere	Bingham statistics	1
Brillault[19]	1991	LSegs.	-	Error model using LSegs. length	-
Lutton[71]	1994	-	Sphere	Semiregular sphere quantification	3 orth.
McLean[75]	1995	Lines	Image plane	Cluster and estimate approach	1
Tuytelaars[117]	1998	Points	Hough	Cascaded Hough transform	N and Vls.
Shufelt[105]	1999	LSegs.	Sphere	Voting scheme using swath error model	-
Antone[4]	2000	LSegs.	Sphere	EM algorithm, angle error model	N
Minagawa[76]	2000	Points	Image plane	EM algorithm for lines and Vps. and Vl.	N and Vl.
Ribeiro[95]	2000	Texture	Sphere	Spectral voting and search for maxima	N
Rother[97]	2000	LSegs.	Image plane	Mid-point error model, unbounded search	3
Cantoni[24]	2001	Points	Hough/Image plane	Voting and least squares sinusoid	1
Liebowitz[68]	2001	LSegs.	Projective plane	Optimal MLE solution for LSegs.	1
Košecká[65]	2003	LSegs.	Unc. Sphere	EM algorithm, outliers control and refinement	N
Almansa[2]	2003	LSegs.	Image plane	Equiprobable vanishing regions	N
Schindler[99]	2004	Edge points	Projective plane	EM algorithm, Atlanta world	N
Hu[50]	2006	LSegs.	Image plane	Convex hull	N
Seo[102]	2006	LSegs.	ICIS	Inverted Coordinate Image Space	N
Suttorp[111]	2006	LSegs.	Image plane	Kalman tracking for road environment	1
Trinh[115]	2006	LSegs.	Projective plane	MSAC estimation	3
Barinova[11]	2007	Edges	Unc. Sphere	Class-specific edges, Mean Shift and EM	3
Kalantari[54]	2008	LSegs.	Unc. Sphere	RANSAC clustering	N
Pflugfelder[93]	2008	LSegs.	Projective plane	Online scheme with EM algorithm	N

LSegs. : Line segments; Vps. : Vanishing points; Vl. : Vanishing line; - : Unknown or unspecified; Unc. : Uncalibrated

Table 3.1: Classification of vanishing point estimation methods.

Moreover, the existing methods based on RANSAC for robust computation of vanishing point can be improved by using the existing variants MSAC and MLESAC, which have been shown to show better performance than simple RANSAC [114].

Finally, addressed by Minagawa et al. [76], the EM algorithm can be used to provide simultaneously information regarding vanishing points and support lines[5], which can be of great interest for plane rectification, as it will be shown in chapter 5. Nevertheless, there are a number of weaknesses in that approach that must be solved for a more general application. The last contribution, and the most innovative of this chapter, is the upgrade of this basic EM method, in order to provide robust, efficient and more general solutions for the simultaneous computation of vanishing points and support lines.

[5]From here one we will denominate support line to those lines that pass through the vanishing point and whose parameters are as well obtained through the EM estimation process.

3.3 Error function

Several error functions have been proposed in the literature to characterize the error between an image feature and a vanishing point. Point-line distances in the image plane [76, 115] are the simplest ones, but only can be applied when the vanishing point is close to the image frame. Their major drawback is that these distances are not valid for infinite vanishing points, as the further a vanishing point is from the image frame, the larger the error will grow. Though, if the calibration matrix is known, the coordinates of the image plane can be mapped to the unit sphere. Hence, the point-line distance becomes a relative difference between two 3D orientations [65], and infinite vanishing points can be handled.

The best results are obtained with specific error functions defined for line segments. Liebowitz [68] proposed an error function based on the sum of the point-line distance of each end-point of the line segment with respect to a line that joins the vanishing point and the line segment. This error function, that has been modified by other authors [97, 112], requires a non-linear optimization process to obtain the vanishing point that minimizes the global error.

Nevertheless, more general approaches [37, 99] consider the orientation error as the workspace on which apply optimization methods. This allows to apply the error function indifferently to gradients-pixels and line segments. For instance, Schindler and Dellaert [99] considers the angle between the normal orientation of a pixel gradient and the line joining the vanishing point and that pixel.

Inspired by this approach, a new error function is proposed, based on the hypothesis that the noise affecting the orientation of image features is Gaussian. This is upheld by the fact that the SSWMS (as described in chapter 2), and other line segment detectors [119], generate line segments by means of orientation-based growing strategies. In the following subsections we introduce the proposed error function, whose performance is compared with that of the most significant error functions found in the literature.

3.3.1 Orientation-based error function

Let $\mathbf{r} = (x, y, 1)^\top$ be the homogeneous coordinates of a point of the image plane with significant gradient magnitude, defined by the components of the gradient vector $\mathbf{g} = (g_x, g_y)^\top$. Typically these points can be obtained in images applying any operator that generates an approximation to the gradient vector at each pixel of the image. Besides, a line segment is defined by a pair of end-points, $\{\mathbf{a}, \mathbf{b}\}$, such that it defines the line $\mathbf{l} = \mathbf{a} \times \mathbf{b} = (l_1, l_2, l_3)^\top$ in homogeneous coordinates. Its associated reference point is the mid-point $\mathbf{r} = \frac{\mathbf{a}+\mathbf{b}}{2}$. Analogously, any gradient pixel can be expressed as a line, as described in chapter 2, section 2.8.

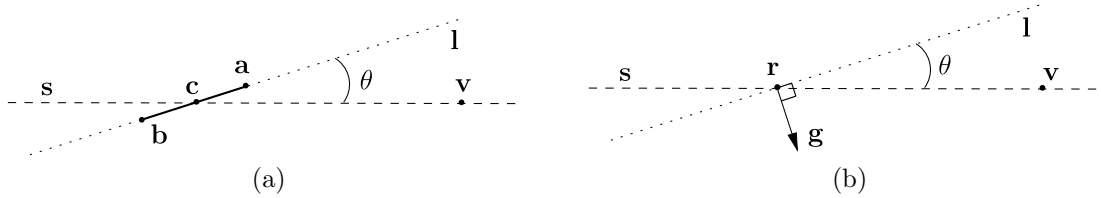


Figure 3.1: Relative orientation between the orientation of the measured feature and the vanishing point ((a) line segments and (b) edge-points). When the vanishing point lies on $\mathbf{l}$, then the cost is minimum ($d(\mathbf{x}, \mathbf{v}) = |\sin\theta| = 0$).

Let consider from here on a generic data sample as the set $\mathbf{x} = \{\mathbf{r}, \mathbf{l}\}$, independently their elements are gradient-pixels or line segments.

The error between a vanishing point $\mathbf{v} = (v_1, v_2, v_3)^\top$ in homogeneous coordinates and an orientation given by the data sample $\mathbf{x}$ is defined as the absolute value of the sine of the angle θ between $\mathbf{l}$ and a line, $\mathbf{s} = (s_1, s_2, s_3)^\top$, which joins $\mathbf{v}$ and $\mathbf{r}$, and that can be computed as $\mathbf{s} = \mathbf{v} \times \mathbf{r}$:

$$d(\mathbf{x}, \mathbf{v}) = |\sin(\theta)| = \frac{|-l_2 s_1 + l_1 s_2|}{\sqrt{l_1^2 + l_2^2}\sqrt{s_1^2 + s_2^2}} \tag{3.1}$$

These concepts are illustrated in figure 3.1 for the two types of considered data samples: (a) line segments and (b) gradient-pixels. The absolute value is considered because we are only interested in the relative deviation between the orientation of the lines, not in its sign, so that $\theta \in [0, \pi]$. Besides, the use of the sine function is motivated by the fact that, for low values of the deviation angle, which is the case for inliers, $\sin\theta \simeq \theta$, thus avoiding the arctangent computation for large amounts of data.

To qualify the contributions of each data sample to the global error minimization process, this approach is extended by adding a scale factor to (3.1) that weights the importance of image features in the optimization process. This way, the scaled orientation-based error function can be defined as $d'(\mathbf{x}, \mathbf{v}) = S \cdot d(\mathbf{x}, \mathbf{v})$, where S is a factor that weights the importance of image features. For instance, in the case of using gradient-pixels, S is the magnitude of the gradient, so that more intense edges are considered more important. For line segments, the scale could be related to their length or any quality measurement obtained in the feature extraction process. As it will be shown, the use of this information makes the system to perform better against outliers.

Considering Gaussian noise in the defined error function, the likelihood of a data sample, indexed with i, to meet a given vanishing point $\mathbf{v}$ is:

	Workspace	Data samples	Error function	Cost	Estimation
Point-line (PL) [115]	Image plane $\mathbf{v}_{2D} = (v_x, v_y, 1)^\top$	$\mathbf{l} = (l_1, l_2, l_3)^\top$	$d(\mathbf{l}, \mathbf{v}) = \mathbf{l}^\top \mathbf{v}_{2D}$	$C = \sum_{i=1}^{N} d^2(\mathbf{l}_i, \mathbf{v})$	Least squares
Cal. Point-line (CPL) [65]	Unit sphere $\mathbf{v} = (v_1, v_2, v_3)^\top$ $\mathbf{v}' = K^{-1}\mathbf{v}$	$\mathbf{l}' = K^\top \mathbf{l}$	$d(\mathbf{l}', \mathbf{v}') = \mathbf{l}'^\top \mathbf{v}'$	$C = \sum_{i=1}^{N} d^2(\mathbf{l}'_i, \mathbf{v}')$	Least squares
End-p. min. (EPM) [68]	Projective plane $\mathbf{v} = (v_1, v_2, v_3)^\top$	$\mathbf{l} = \mathbf{a} \times \mathbf{b}$ $\mathbf{a} = (a_x, a_y, 1)^\top$ $\mathbf{b} = (b_x, b_y, 1)^\top$	$d(\mathbf{a}, \mathbf{s}) = \frac{\mathbf{a}^\top \mathbf{s}}{\sqrt{s_1^2 + s_2^2}}$ $d(\mathbf{b}, \mathbf{s}) = \frac{\mathbf{b}^\top \mathbf{s}}{\sqrt{s_1^2 + s_2^2}}$ $\mathbf{s} = \underset{\mathbf{s}}{\operatorname{argmax}} \left(d^2(\mathbf{a}, \mathbf{s}) + d^2(\mathbf{b}, \mathbf{s})\right)$	$C = \sum_{i=1}^{N} d^2(\mathbf{a}_i, \mathbf{s}) + d^2(\mathbf{b}_i, \mathbf{s})$	Levenberg-Marquardt
End-p. max. (EPx) [112]	Projective plane $\mathbf{v} = (v_1, v_2, v_3)^\top$	$\mathbf{l} = \mathbf{a} \times \mathbf{b}$	$d(\mathbf{a}, \mathbf{s}) = \frac{\mathbf{a}^\top \mathbf{s}}{\sqrt{s_1^2 + s_2^2}}$ $\mathbf{s} = \mathbf{v} \times (\mathbf{a} + \mathbf{b})/2$	$C = \sum_{i=1}^{N} d^2(\mathbf{a}_i, \mathbf{s})$	Levenberg-Marquardt
Orientation (O)	Projective plane $\mathbf{v} = (v_1, v_2, v_3)^\top$	$\mathbf{l} = (l_1, l_2, l_3)^\top$	$d(\mathbf{x}, \mathbf{v}) = \frac{\lvert -l_2 s_1 + l_1 s_2 \rvert}{\sqrt{l_1^2 + l_2^2}\sqrt{s_1^2 + s_2^2}}$ $\mathbf{s} = \mathbf{v} \times (\mathbf{a} + \mathbf{b})/2$	$C = \sum_{i=1}^{N} d^2(\mathbf{x}_i, \mathbf{v})$	Levenberg-Marquardt
Or. with scale (OS)	Projective plane $\mathbf{v} = (v_1, v_2, v_3)^\top$	$\mathbf{l} = (l_1, l_2, l_3)^\top$	$d(\mathbf{x}, \mathbf{v}) = S\frac{\lvert -l_2 s_1 + l_1 s_2 \rvert}{\sqrt{l_1^2 + l_2^2}\sqrt{s_1^2 + s_2^2}}$ $\mathbf{s} = \mathbf{v} \times (\mathbf{a} + \mathbf{b})/2$	$C = \sum_{i=1}^{N} d^2(\mathbf{x}_i, \mathbf{v})$	Levenberg-Marquardt

Table 3.2: Summary of the error functions considered for comparison.

$$p(\mathbf{x}_i|\mathbf{v}) \propto \exp\left(-\frac{1}{2\sigma^2} d^2(\mathbf{x}_i, \mathbf{v})\right) \tag{3.2}$$

Given a set of i.i.d. data samples, $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$, actually meeting at a single vanishing point (i.e., all data are inliers)[6] the global likelihood function is:

$$p(\mathcal{X}|\mathbf{v}) = \prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{v}) \tag{3.3}$$

Provided that the conditional distribution (3.2) is normal, the maximum likelihood estimation is solved by minimizing the sum of squared costs $\mathcal{C} = \sum_{i=1}^{N} d^2(\mathbf{x}_i, \mathbf{v})$. However, this is a difficult problem to solve analytically as $\mathcal{C}$ is highly non-linear. Therefore, approximate numerical methods for non-linear optimization are required. In this work we propose to use the Levenberg-Marquardt algorithm [46], which has been already used by other authors whose error functions are non-linear [68, 112].

3.3.2 Tests & discussion

The aim of this section is to compare the performance of the proposed orientation-based error function with others in the literature. For that purpose, the error map concept is first introduced, and then used to visually compare the behavior of the

[6] The classification between inliers and outliers is discussed in section 3.4.

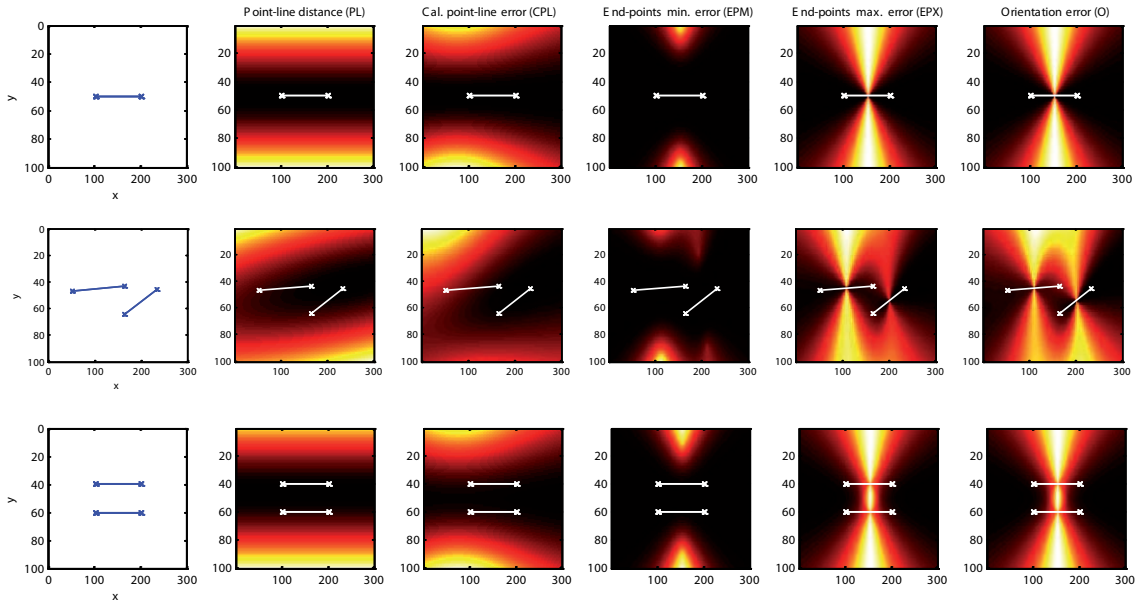


Figure 3.2: Error maps generated for example synthetic line segments.

different methods according to different representative data configurations. In second place, a number of tests are carried out with both synthetic data and real data sets to evaluate the efficiency and accuracy of the compared methods.

The methods we are comparing with are the ones mentioned in the first paragraphs of section 3.3. Their properties are summarized in table 3.2: (PL) point-line error; (CPL) calibrated point-line error; (EPM) end-points minimum error, and (EPx) end-points maximum error. Our approaches will be denoted as O (Orientation) and OS (Orientation with scale).

Let us remark one last issue before entering into the discussion. Typically lens distortion affects the computation of vanishing points since straight lines in the scene are shown as non-linear elements. Lens distortion correction does not enter into the targets of this thesis so that we have used synthetic data which does not have this type of distortion, and images captured by cameras which have shown very low lens distortion (using enough large focus). Nevertheless, if required, it is possible to correct this distortion by assuming some model (like a first-order radial lens distortion [93]) and compute their parameters with a calibration tool.

3.3.2.1 Error maps

First let us evaluate how the error maps generated with the mentioned error functions of the literature look like using some synthetic data examples. These maps are simply

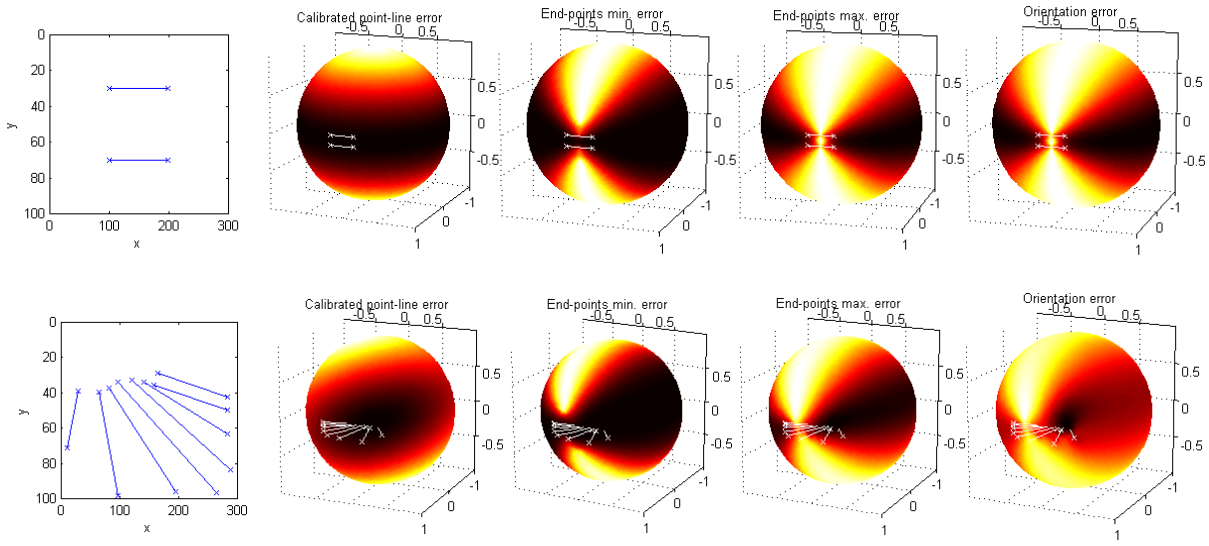


Figure 3.3: Error maps generated for synthetic examples in the unit sphere.

obtained by computing the value of the error function for all the putative values of the vanishing point given a set of data samples, i.e. this error is computed as the sum of the error produced by each data sample at each position of the space in which the vanishing point is defined. For instance, figure 3.2 illustrates the error maps considering the image coordinates of vanishing points, for different sets of line segments. In these error map representations, darker regions correspond to points in the image with low associated error (which means that are likely to be the vanishing points). Brighter regions depict high error positions. The first row illustrates the case of having a single line segment, which shows quite clearly the differences between the methods. For instance, the point-line distance delivers an error map that grows with the orthogonal distance of the point to the line segment. As shown, the darkest region is a horizontal stripe, parallel to the line segment. The calibrated point-line behaves similarly, and its error map is equivalent to the one of PL, but with an added projective distortion (nevertheless, this example is better understood considering the unit sphere representation, as it will be shown in figure 3.3). The EPx method and the proposed orientation-based method result about the same error map for this simple case, as the error grows with the angle with respect the mid-point of the line segment. The EPM method is similar to these ones, but with a wider dark margin near the line segment. The second row includes two converging line segments, and the associated error maps. Attending to the error maps of the methods EPx and O, the only perceptive difference is that the error-map of the proposed orientation-based method is much sharper, so that the transitions between dark and bright regions are much more abrupt. For larger sets of line segments, the commented difference makes that the proposed approach

generates error maps with less local minima, i.e. the global minima is much deeper and thus the estimation process is less prone to get stuck in a strong local minima.

The third row has been intentionally included to depict the limitation of this image-plane error map representation. The error map of lines converging outside the limits of the image can not be fully seen, as is the case for parallel lines, that meet at the infinity. Figure 3.3 show the error maps in the unit sphere, for all the methods except for the point-line distance, that can not be projected into the sphere since its error is unbounded in the projective plane (the distance from a line to a infinite vanishing point is infinite). The two parallel line segments case is now shown in the first row, where it can be observed that the low error regions are at the equator of the sphere. With this representation we can see that the CPL method provides a solution for infinite vanishing points, but is still not enough accurate. The value of the error at the equator is constant, so that the vanishing point position for this simple case is not unique. Better accuracy is provided by the EPM, EPx and orientation-based method compared with the CPL, since the error maps offer no ambiguity and a single vanishing point can be obtained[7]. The second row includes a larger set of line segments meeting approximately, with some added noise, at a common vanishing point. In this case, the error map of the proposed orientation-based method is clearly more peaked around the best vanishing point, i.e. the dark region around the vanishing point is much smaller and the error grows faster as we move from the vanishing point. Although it is clear from the figures that all the methods provide low error values around the same vanishing point, we should demand them to give these low values only in that regions and not in the rest of the space (as done by the orientation-based method). This is important since estimation methods based on iterative procedures may be initialized randomly in the sphere, and the sharper the error map is, the lower number of iterations are required. These concepts are analyzed in the following subsection, which also includes examples of error maps for line segments obtained from real images.

3.3.2.2 Quantitative comparison

Specific tests are defined to evaluate the estimation process associated to each error function. The comparisons are made in terms of accuracy and efficiency.

The accuracy is tested according to the error committed in the estimations of vanishing points with respect to the ground truth vanishing points. For this purpose, synthetic data tests are carried out, for which exact ground truth is available. The efficiency is afterwards tested according to the number of iterations required by the

[7]Realize that in this visualization, vanishing points correspond to 3D vectors passing through the center of the sphere, i.e. representing a 3D direction, which intersects the sphere surface in two points. Nevertheless, the sign of the direction is ignored so that only half of the sphere shall be considered. The entire spheres are shown for a better visualization.

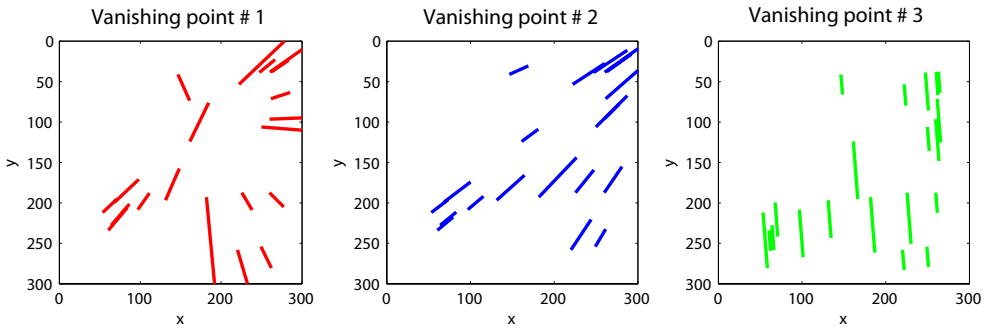


Figure 3.4: Ground truth line segments for three different vanishing points: from left to right, inside and outside the limits of the image frame, and in the infinity.

non-linear optimization procedures. For this test, real images are used, since in these cases the difference in efficiency is much more noticeable.

Synthetic data tests

Three random sets of line segments meeting at three different vanishing points are generated. Figure 3.4 shows the three sets of ground truth line segments meeting at three vanishing points, which have been defined to be (#1) inside the image boundaries, (#2) outside, and (#3) in the infinity. Additionally, Gaussian noise is added to corrupt the data samples, in two different ways: (a) the position of the end-points and (following the hypothesis of Liebowitz [68]); and (b) in the orientation of the data sample (according to the assumption described in this thesis). The standard deviation of the noise distributions are set to $\sigma_x = \sigma_y = 3$ (pixels) in the case of bivariate noise in the position of the end-points, and $\sigma_\theta = 0.1$ for the orientation noise. These are the typical values of the standard deviation for detected line segments with the proposed method (described in chapter 2).

All the estimation methods (PL, CPL, EPM, EPx, O and OS) are applied to the different data sets, and the estimation error is computed as the sine of the relative angle between the detected vanishing point, $\hat{\mathbf{v}}$, and the ground truth vanishing point, $\mathbf{v}_{gt}$. Realize that these points have been calibrated to the unit sphere and thus represent 3D space directions, since we have normalized the data using a ground truth calibration of the synthetic view. Hence, this error ranges from 0 for perfect matching to 1 for orthogonal directions.

The tests were executed 200 times, adding random noise to the line segments as explained above. The obtained results are shown in figure 3.5. The first row shows the average error obtained for the different methods, using the two types of noise models, for the three considered vanishing points. On the one hand, the results for vanishing point # 1 are very accurate for all methods, since it corresponds to the easiest situation. On the other hand, the point-line methods offer the highest errors, for both noise types and for the three vanishing points. As expected, the end-point

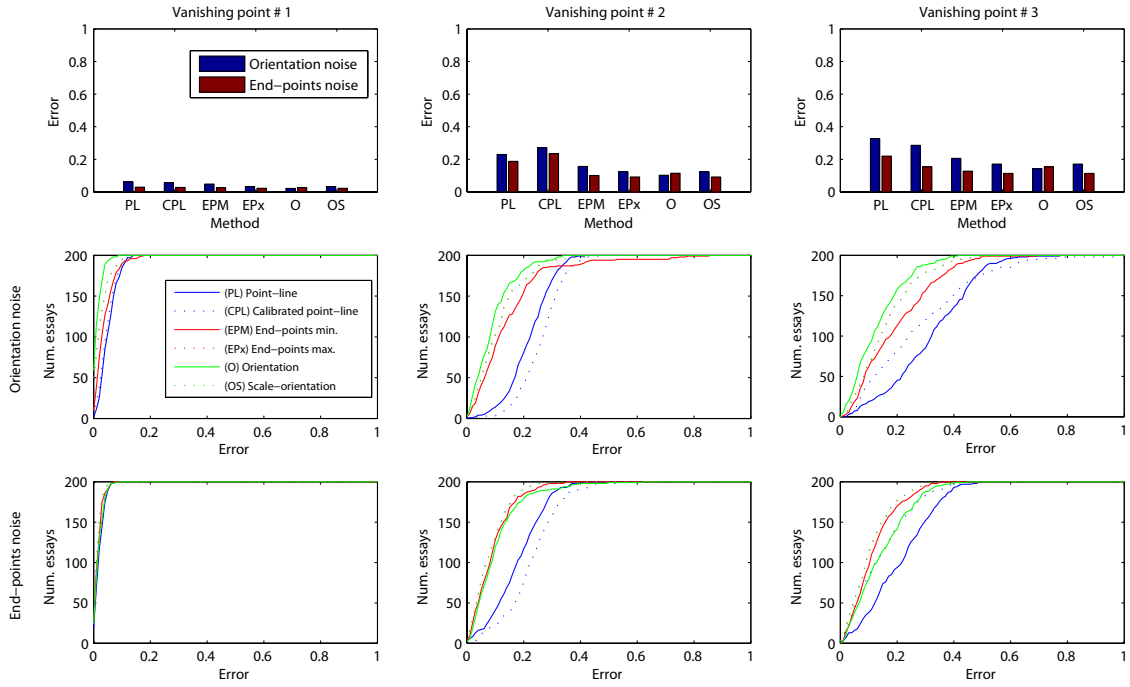


Figure 3.5: Results of the synthetic tests for 200 essays. The first row illustrates the average error for each method at each vanishing point. The second row shows the cumulative error histograms for the orientation noise assumption. The third row shows the corresponding cumulative error histograms for the end-point noise model.

methods (EPM and EPx) show the lower error for the end-point noise model, and analogously, the orientation-based methods (O and OS) perform better assuming the orientation noise model. Nevertheless, the comparison of the performance of these methods is more clearly depicted by the figures in the second and third row. These figures illustrate the cumulative histograms of the committed error, respectively for orientation and end-points noise model, which point out the error distribution for the complete set of tests. As shown, the vanishing point # 1 is well estimated by all the methods, and there is no significant difference between them. However, for vanishing points # 2 and especially # 3, the end-points methods (both EPM and EPx) render more error than the proposed orientation-based approaches. Nevertheless, under the hypothesis of bivariate Gaussian noise in the position of the end-points, the difference between the results for EPM, EPx and the proposed approaches (O and OS) is not significant.

Real images tests

An available image database has been used to test the performance of the proposed methods with real images: the York Urban Data Base (YUDB) [37], composed by 102

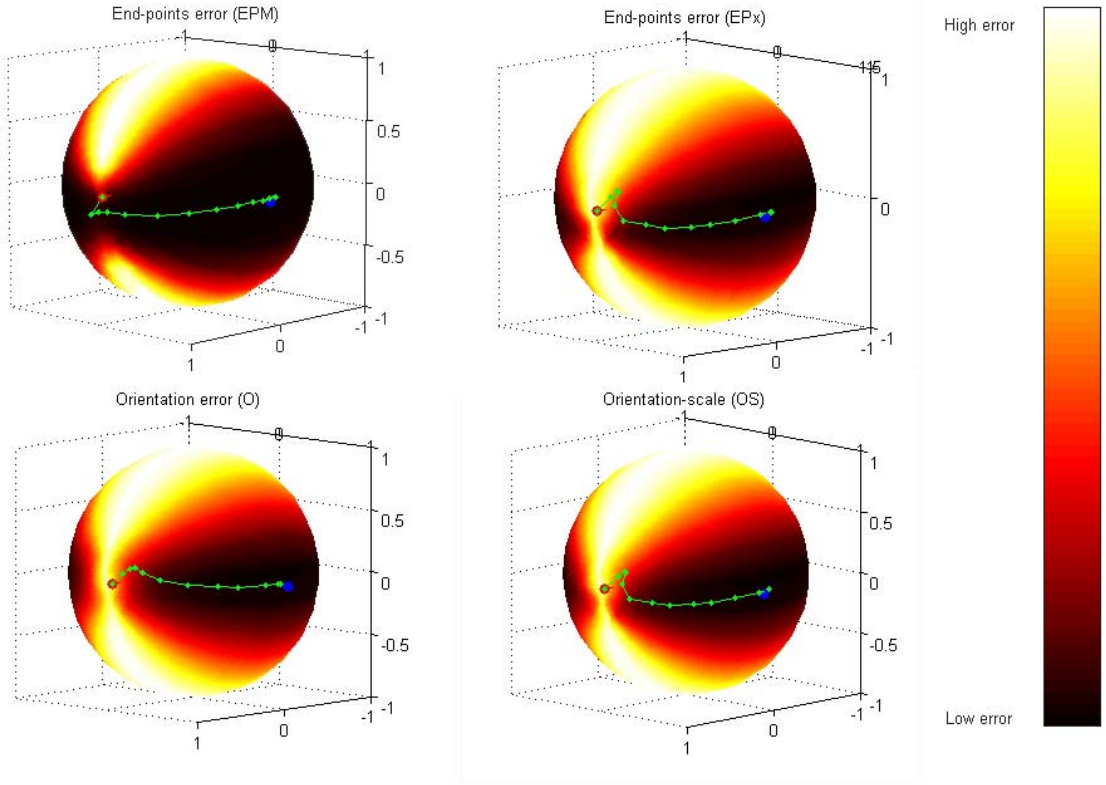
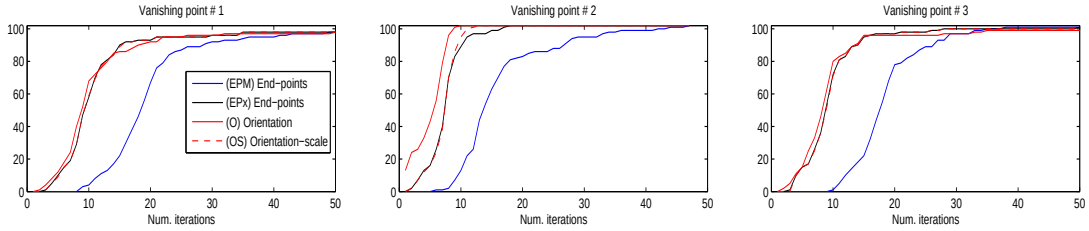


Figure 3.6: Error maps for an example line segment set, and the associated iterative procedure: in red the initialization point, in green the different iteration steps, and the blue point depicts the final estimated vanishing point.

images of structured environments (corridor, facades, parkings, etc.) for which the calibration information is available as well as the ground truth of the three dominant vanishing points. The YUDB has been selected since it also includes the ground truth line segments meeting at the dominant vanishing points. Although manually selected, these line segments are noisy, as well as the ground truth vanishing points, which are computed with the estimation method proposed by Collins [29].

We can generate error maps on the unit sphere for each vanishing point of each image considering their corresponding sets of line segments as done for the synthetic data examples. This way we can easily find the position of the vanishing point that provides the lowest error, and test the different non-linear iterative methods (EPM, EPx, O and OS) using the same randomly obtained initialization point. Specifically, we want to test the required number of iterations of the Levenberg-Marquardt algorithm for the different error functions to reach the ground truth vanishing points. Figure 3.6 shows the unit sphere error map for one vanishing point of a real image. The initialization is shown as a red point on the sphere surface, and the iterations are shown as a green



(a) (b) (c)

Figure 3.7: Cumulative histograms of the number of required iterations for the non-linear optimization procedures for the three different vanishing points of the set of images of the YUDB.

path. The final estimation is depicted in blue (the last steps of the path are very close to the final estimation, so that they can not be seen in the illustration). As expected, all methods offer the same final estimation. Nevertheless, the orientation-based methods display narrower error valleys, which lead to faster convergence of the optimization procedures (the first steps of the path are longer and thus the process arrive to the target in less iterations). This is a key advantage of the proposed approach: in case we have a limited or fixed available number of iterations, the committed error is lower than with the other methods. The error valleys are deeper and thus in few iterations the estimation has already reached very low error regions, while the other methods typically spend some more iterations without significant error reduction.

Figure 3.7 shows the cumulative histogram of the required number of iterations for the different methods for all the vanishing points of all the images of the data base. As shown, the number of iterations provided by the proposed orientation-based methods are lower in average for all vanishing points, especially for the second one (which corresponds to the vertical vanishing point of the images in the YUDB). Regarding the end-points methods, the one by Tardif [112] clearly improves the results of the method by Liebowitz [68] and obtains very similar results to the ones of the proposed methods for the vanishing points #1 and #2.

At this point, we can discuss about the two proposed methods (O and OS), which only differ in the use of the scale factor. In general terms, as shown by the described tests, the performance of these methods is very similar in average, but slightly better for the orientation-based method without scale. In the case of synthetic data, all data samples have been corrupted by the same noise level, such that the assumption that longer or stronger image features are less noisy is not kept. Hence, the orientation-based with scale method can not provide better results. For the real data tests, the reasoning is similar, since the used data samples come from manual selection, which again shall produce noisy line segments incoherent with the mentioned noise hypothesis. Nevertheless, for sets of line segments automatically obtained with a line

segment detector (such as the SSWMS described in chapter 2), which may contain both long and short line segments, we have observed that the use of scale factor indeed increase the performance of the methods.

3.4 Robust estimation

As revised in section 3.2.3.2, there are several methods that use RANSAC to compute vanishing points. For instance, Trinh and Jo [115] use MSAC and the point-line distance between vanishing points and image features. Pflugfelder [93] obtains good results applying the error function proposed by Liebowitz [68] with the RANSAC framework. To determine more than one vanishing point, Pflugfelder [93] removes the line segments that were used to compute one vanishing point and restarts the RANSAC algorithm to search for new vanishing points. The procedure is repeated as many times as desired or until there are not enough line segments to determine new vanishing points. We will follow this recursive scheme using the proposed orientation-based error function since it provides better convergence properties, and introducing the MSAC and MLESAC variants of the RANSAC algorithm[8].

3.4.1 RANSAC for vanishing point estimation

The robust estimation of vanishing points using RANSAC is done iteratively, such that at each iteration, given the complete set of observations $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{N}$, a single dominant vanishing point is hypothesized, and each line segment is assumed to be drawn from a mixture model of two classes, labeled as $c = \{ON, OFF\}$. These classes indicate respectively that the line segments meets at the vanishing point or not (i.e. it is an inlier or an outlier). The likelihood model is defined as follows:

$$p(\mathbf{x}_i|\mathbf{v}) = \sum_{c} p(c)p(\mathbf{x}_i|\mathbf{v}, c) \tag{3.4}$$

where the probability $p(\mathbf{x}_i|\mathbf{v}, c = ON)$ is modeled as a normal distribution as in (3.2), and $p(\mathbf{x}_i|\mathbf{v}, c = OFF)$ is a uniform distribution. The value of the uniform distribution is computed as the inverse of the highest expected error, so that, for the defined error function, the maximum error corresponds to the case for which the orientation of $\mathbf{s}$ is orthogonal to $\mathbf{l}$ and the sine of the angle is 1. This way $p(\mathbf{x}_i|\mathbf{v}, c = OFF) = 1$.

Summarizing, the RANSAC algorithm proceeds iteratively in two main steps: (i) a minimal sample subset (MSS) of data is randomly selected in order to generate a

[8]Related to these variants of RANSAC, two publications summarize the contributions described in this chapter, one devoted to MSAC [83], and the other to MLESAC [85].

hypothesis of the vanishing point. In this case, the minimal set is composed by two line segments whose best estimation is $\mathbf{v}^* = \mathbf{l}_i \times \mathbf{l}_j$; and (ii) the consensus set, $CS(\mathbf{v}^*)$, i.e. the subset of line segments that are coherent with the hypothesis is computed as

$$CS(\mathbf{v}^*) \triangleq \{\mathbf{l}_i \in \mathcal{X} : d^2(\mathbf{x}_i, \mathbf{v}^*) \leq \delta\} \tag{3.5}$$

where δ is a certain error threshold governed by the statistics of the data (details are given in section 3.4.1.2). These two steps are repeated until the probability of finding a better consensus set is below the convergence threshold. Here resides the difference between RANSAC, MSAC and MLESAC (as a special case, MLESAC is described in detail in section 3.4.1.1): while basic RANSAC ranks the consensus sets according to their cardinality, i.e. the number of elements of the set, the MSAC version ranks them according to a global cost function that gathers individual contributions of inliers and outliers. The total cost of an hypothesis of the MSAC iterative procedure can be expressed as $\mathcal{C}(\mathcal{X}, \mathbf{v}^*) = \sum_{i=1}^{N} \rho(\mathbf{x}_i, \mathbf{v}^*)$, where $\rho(\mathbf{x}_i, \mathbf{v}^*)$ is defined as:

$$\rho(\mathbf{x}_i, \mathbf{v}^*) = \begin{cases} 1 & , d^2(\mathbf{x}_i, \mathbf{v}^*) \leq \delta \\ 0 & , \text{otherwise} \end{cases} \tag{3.6}$$

and analogously, for the MSAC algorithm as:

$$\rho(\mathbf{x}_i, \mathbf{v}^*) = \begin{cases} d^2(\mathbf{x}_i, \mathbf{v}^*) & , d^2(\mathbf{x}_i, \mathbf{v}^*) \leq \delta \\ \delta & , \text{otherwise} \end{cases} \tag{3.7}$$

which is a re-descending estimator that provides better results in the presence of outliers, since the contribution of the outliers to the error is limited by the threshold value δ.

As a final remark, the minimal sample subset selection can be carried out more efficiently for both RANSAC and MSAC by using any type of sampling strategy that is more peaked than a purely random process. Thinking in sampling as the procedure in which each data sample is given a certain probability of being selected, random sampling assigns the same probability to all samples. A more efficient approach is to assign to each sample a value that encodes somehow its importance. For instance, line segments are assigned their length, and gradient pixels their gradient magnitude, since longer line segments and stronger gradient pixels are more likely to be less noisy and, typically, more likely to be inliers.

3.4.1.1 MLESAC

MLESAC defines a more accurate ranking system than RANSAC and MSAC. Provided that the likelihood value of each line segment with respect to the hypothesized

vanishing point is available, the total cost is computed as the global likelihood function $p(\mathcal{X}|\mathbf{v}^*) = \prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{v}^*)$. The main problem with this approach is that the prior probability, $p(c = ON)$, is unknown and must be estimated[9]. Nevertheless, this estimation can be done with a one-dimensional EM optimization procedure [114]. Typically, using $p(c = ON) = 0.5$ as initialization, convergence is achieved in just 3 or 4 iterations.

The convergence of the MLESAC algorithm is related to the mean log-likelihood value, which is given by:

$$\log\left(p(\mathcal{X}|\mathbf{v}^*)\right) = \frac{1}{N}\sum_{i=1}^{N} \log \sum_{c_l} p(c_l)\ p(\mathbf{x}_i|\mathbf{v}^*) \tag{3.8}$$

At each iteration, indexed as k, the improvement of mean log-likelihood is checked as:

$$\Delta \log\left(p(\mathcal{X}|\mathbf{v}_k^*)\right) = \frac{\log\left(p(\mathcal{X}|\mathbf{v}_k^*)\right) - \log\left(p(\mathcal{X}|\mathbf{v}_{k-1}^*)\right)}{\log\left(p(\mathcal{X}|\mathbf{v}_{k-1}^*)\right)} \tag{3.9}$$

When $\Delta \log\left(p(\mathcal{X}|\mathbf{v}_k^*)\right)$ is below a certain threshold ϵ, the process is considered to have converged. Typically it is accurately enough to use $\epsilon = 10^{-5}$.

3.4.1.2 RANSAC and EM

Within the RANSAC framework, the hypothesis and test steps can be though as equivalent to the E-step of the EM algorithm, since they comprise the selection of a minimal sample subset, as well as the computation of the corresponding consensus set.

Given a vanishing point hypothesis, the threshold δ that divides line segments between inliers and outliers is selected such that $p(d^2(\mathbf{x}_i, \mathbf{v}) \leq \delta) = P_{\text{inlier}}$. It can be computed as $\delta^2 = \sigma^2 F^{-1}_{\mathcal{X}^2_n}(P_{\text{inlier}})$. The value of P_{inlier} is typically set to 95%, and σ is the expected standard deviation of the noise of the inliers. In the case of one-dimensional noise, $n = 1$, (as the case for the orientation-based error function), assuming that the maximum deviation angle is $\theta_{\max} = \pi/16$, and that $2.5\sigma = \sin(\pi/16)$ swaths the 95% of the probability, then $\sigma = 0.065$ and $\delta = 0.01623$.

Following the analogy of the EM algorithm, we can define the M-step of the RANSAC strategy as the re-estimation of the vanishing point considering the contribution of all the line segments of the consensus set. The likelihood of the consensus set is defined as in (3.3), considering only the inliers, and provided that the conditional distribution (3.2) is normal, the maximum likelihood estimation is solved by minimizing the sum of squared costs $\mathcal{C} = \sum_{i=1}^{N} d^2(\mathbf{x}_i, \mathbf{v})$. As already commented, this is a difficult problem to solve analytically as the expression to maximize is highly

[9]Note that $p(c = OFF) = 1 - p(c = ON)$.

Algorithm	Error function	VP. #1	VP. #2	VP. #3	Average(%)
MSAC	O	92 / 98	100 / 102	91 / 101	94.01
MSAC	OS	94 / 98	99 / 102	91 / 101	94.35
MLESAC	O	89 / 98	95 / 102	86 / 101	89.70
MLESAC	OS	91 / 98	96 / 102	84 / 101	90.05

Table 3.3: Detection results for the different proposed error functions and robust schemes.

non-linear. Therefore approximate numerical methods for non-linear optimization are required, such as the Levenberg-Marquardt algorithm.

3.4.2 Tests and discussion

The robustness of the proposed schemes is evaluated comparing the angle error between the estimated vanishing points and the ground truth vanishing points in the YUDB[10]. The image information that feeds the proposed methods is obtained with the line segment detector described in chapter 2.

Although there are 102 images in the data base, not all of them contain three vanishing points. Thus, we have observed than 98 images have information about the first horizontal vanishing point (numbered as vanishing point # 1), 102 about the vertical one (numbered as vanishing point # 2), and 101 about the second horizontal vanishing point (numbered as vanishing point # 3). The errors are measured in angles, since the camera calibration is available, and makes that image plane positions of vanishing points become 3D vanishing directions (as explained in appendix B). To simplify the evaluation of the results, let us consider the detection rate as the main feat of the proposed robust schemes. In that sense, a vanishing point is considered to have been correctly detected when the computed error angle is below 10 degrees with respect the ground truth, otherwise, the estimation is considered to have failed (mainly due to the presence of outliers).

Line segments are detected using the SSWMS method, described in chapter 2, using a bandwidth value $r = 3$ and running the algorithm until all the pixels in the image have been evaluated. In average, for the images of the YUDB, whose size is 640×480, the number of obtained line segments is around 450. Besides, in average, the proportion of outliers is 25% compared to the sum of inliers of each vanishing point (i.e. line segments that meet at any of the three main vanishing points). This way, for the most dominant vanishing point, the proportion of outliers is about 75%

[10]Though accurate, the ground truth vanishing points of the YUDB have been computed using a particular method [29], so that we have preferred to compute more accurate ground truth vanishing points finding the lower error point on the sphere error maps.

Algorithm	Error function	VP. #1	VP. #2	VP. #3	Average(%)
RANSAC	EPL	85 / 98	97 / 102	89 / 101	89.98
MSAC	EPL	92 / 98	99 / 102	90 / 101	93.36
MSAC	EPx	90 / 98	97 / 102	87 / 101	91.03
MSAC	CPL	85 / 98	92 / 102	80 / 101	85.38

Table 3.4: Detection results using popular error function in the literature with the proposed robust schemes.

(considering as outliers also the inliers for the other vanishing points), 50% for the second vanishing point and 25% for the third.

First, different strategies are compared, which result of the combination of the two proposed error functions (orientation (O) and orientation with scale (OS)) with the two presented robust schemes, MSAC and MLESAC. The results are summarized in table 3.3, where the detection rates are shown for each vanishing point. As observed, the detection rate is in average above the 90% detection ratio, being the MSAC-OS the with scheme that shows the best results. MLESAC offers slightly worse results, probably due to the presence of the additional EM stage that computes the prior coefficients, which can be, by its own, a source of error in the computation of CS ranking since it heavily depends on the initialization. Table 3.3 also reflects that the detection rates for the vanishing point #2 are higher than for the other vanishing points. The reason is that this vanishing point in the YUDB corresponds to the vertical vanishing point in structured scenarios, for which typically more inlier line segments are available. Vanishing point # 3 is the one that obtain worse results since it corresponds to the second (in terms of importance) horizontal vanishing point. Hence, the number of line segments meeting at it is typically lower, and thus the estimations are more prone to errors.

Second, the performance of other related error functions in the literature is also compared in similar terms. Different alternatives result from the combination of the analyzed robust schemes with the error functions already mentioned in previous sections. In particular, the MSAC algorithm is used, as being the best robust scheme, with the EPL, EPx and CPL error functions, and also the RANSAC algorithm with the EPL error function, since it has been explicitly addressed in the literature [93]. The results are shown in table 3.4. The best detection rates are obtained with the MSAC-EPL strategy. It is noteworthy that this method obtains the more accurate results, compared to others, such as MSAC-EPx, although still having worse numbers than the proposed approach MSAC-OS.

In summary, we have seen that the results provided by the the proposed combination of the MSAC algorithm and the defined orientation-scale (OS) error function provide the most robust results, compared to the different combinations of robust

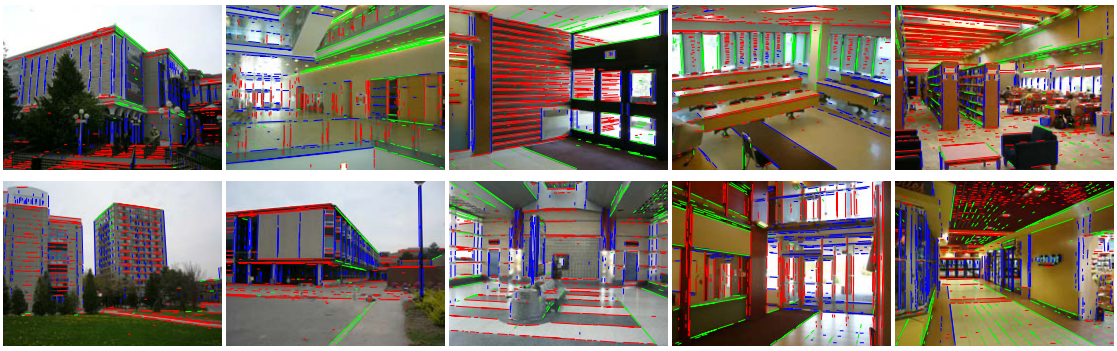


Figure 3.8: Examples of the use of the proposed MSAC algorithm in images of the YUDB. Line segments are colored according to the vanishing point they meet.

algorithms and error functions.

Figure 3.8 shows several examples of the results obtained applying the proposed method (MSAC scheme with the orientation-scale (OS) error function). The line segments are shown colored according to the vanishing point they have been associated to. In general terms, the application of this method leads to excellent results and correct line segment classifications for structured scenarios, which include non-structured elements such as trees, pedestrians, etc.

3.4.3 Vanishing point tracking with RANSAC

Although not initially foreseen, the described robust schemes can be easily adapted to track vanishing points in sequences of images. The number of iterations of RANSAC-based approaches can be dramatically reduced if the random sampling process that selects the MSS is substituted by a deterministic algorithm that uses the information of the estimations of the previous time instant.

If the motion of the camera is moderate, it can be assumed that vanishing points between consecutive images of a sequence do not show significant changes in their position. This way, the error between the data samples detected in time t, and the previous estimation of vanishing points, $\mathbf{v}_{t-1}$, in time $t-1$, can be computed. Data samples are then sorted according to their associated error values, so that data samples with lower associated error values are selected as MSS first, and new vanishing point candidates, $\mathbf{v}_t^*$, are computed. Typically, the first candidate is accurate enough to obtain a very high ranked CS, thus not requiring new iterations to be carried out. In our experience, we have observed that, in average, very few iterations (2 to 4 in average) are required for a very accurate estimation of the vanishing points, dramatically reducing the cost of the MSAC algorithm.

To illustrate this technique, the MSAC-OS algorithm (which provides the best

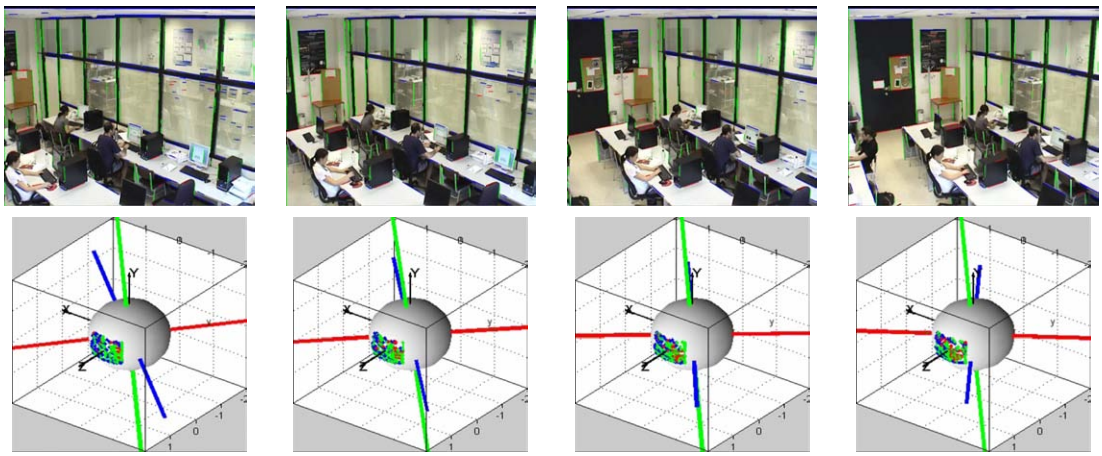
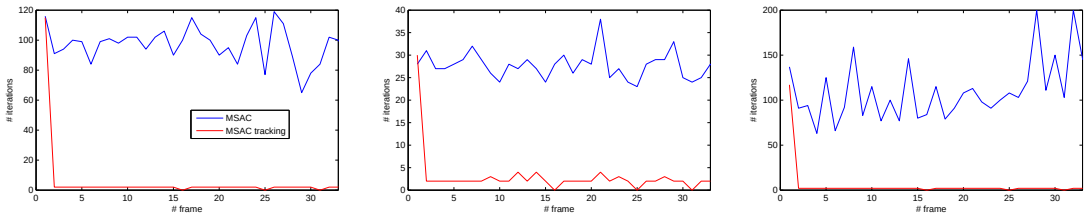


Figure 3.9: Example images of a typical surveillance video sequence in an indoor scenario. Line segments are colored according to the vanishing point they meet at. In the lower row, a representation on the unit sphere is shown, where the three main vanishing points are shown as thick colored lines.



(a) (b) (c)

Figure 3.10: Number of iterations of the MSAC algorithm with and without using the prediction information.

performance among the different robust alternatives) has been tested applied to a video sequence of an indoor structured scenario, with and without using the abovementioned MSS selection. Figure 3.9 shows some example images of this sequence. The scene corresponds to a surveillance environment, monitored with PTZ cameras (that stands for pan, tilt and zoom cameras). For the example sequence shown in the figure, the rotation incurred by the camera is on the vertical axis, which is depicted in the figures of the bottom, where the two horizontal vanishing points, shown in red and blue, rotates along the green, vertical one.

The comparative results are shown in figure 3.10. The graph in (a) correspond to the vertical vanishing point, and (b) and (c) correspond to the other two vanishing points. As can be observed, the number of iterations of the tracking version of the MSAC algorithm are always very low, ranging between 2 and 4, except for the first

frame, that requires a full MSAC computation. Without the prediction information, the number of iterations is typically very high to obtain similar values of likelihood.

3.5 Optimal estimation of vanishing points

As described in the state of the art, the EM algorithm has been successfully used for the estimation of vanishing points, both for single images [99] or sequences of images [93], as a refinement procedure which needs moderately accurate initialization. This thesis sets out the use of the explained robust schemes based on RANSAC, which serves as initialization, followed by a refinement step based on the EM algorithm.

Among all the related works, the one by Minagawa et al. [76] is noteworthy due to the definition of a novel set of parameters to be estimated in the EM framework. Apart from the position parameters of the vanishing points, the parameters of several lines that pass through vanishing points are also considered, as well as the vanishing line defined by those vanishing points. The maximization of this model implies the simultaneous optimization over the vanishing line, vanishing points and supporting lines[11] (those passing through them), which groups, in a single elegant step, several stages typically needed after vanishing point estimation. The use of the EM algorithm ensures the convergence of the model (when appropriately initialized), which includes not only vanishing points but also the most significant support lines. In the author's opinion, the use of support lines also helps to provide more accurate results for structured scenarios. In general, sub-sets of line segments can provide better estimations than the whole set of inliers of a vanishing point. The most reliable line segments are precisely those that are aligned in groups, or in dominant lines, rather than isolated line segments, which are typically more prone to be accidentally meeting at the vanishing point and thus increasing the committed estimation error.

The methods proposed in this section are constructed over the initial work by Minagawa et al. [76] aiming to overcome its main drawbacks. In first place, this work does not consider a couple of important aspects for its practical use. On the one hand the likelihood model considers point information only, without using gradient or orientation information, optimizing over point-line distances. On the other hand, there is not an explicit control of outliers, showing no experiments about how the optimization process behave in the presence of points that do not actually belong to the support lines. These two problems make this alternative not efficient in the usage of information and not robust in real situations that, typically, may contain a significant number of outliers (as discussed in section 3.4.2). To overcome these limitations, different contributions have been conceived: (i) the inclusion of orientation information in the likelihood model (given by gradient-pixels or line segments); and

[11] Through this thesis we will refer to these lines as "support lines" or "main lines".

(ii) the addition of a new component to the mixture model that allows to handle the presence of outliers. These two improvements are included in the first algorithm, (from hereon denominated as "image-plane EM algorithm"), which is described in section 3.5.1.

Second, Minagawa et al. define the image plane as the search space, which is an important limitation for the estimation of vanishing points that are far from the limits of the image (such as infinite vanishing points) as mentioned in the analysis presented in section 3.2. Therefore, in line with other works in the EM literature [65, 99], a second strategy is proposed (analogously, it will be denoted as "projective-plane EM algorithm"), which extends the solution proposed in the first approach considering the estimation of multiple vanishing points in the projective plane, simultaneously with the most significant support lines. This strategy is described in section 3.5.2.

This way, the two next subsections introduce the two proposed strategies in detail. The first one is an adapted solution for vanishing points not far from the image boundaries. This approach is described in great detail to explain some considerations about the maximization process of the EM algorithm that are shared with the second approach. The projective-plane EM algorithm, as an extension of the image-plane EM, is a more complex solution for general vanishing points configurations.

3.5.1 Image-plane EM algorithm

3.5.1.1 Likelihood model

As mentioned in section 3.3.1, let us consider a data sample $\mathbf{x}_i = \{\mathbf{r}_i, \mathbf{l}_i\}$, independently if it comes from a gradient-pixel or a line segment. Since the image-plane EM algorithm works on the image plane, let us consider a point as a 2D entity $\mathbf{r}_i = (x_i, y_i)^\top$. Besides, to simplify the equations of this section, let us consider the normal direction in the image plane of the line $\mathbf{l}_i$ as $\mathbf{g}_i = (g_{xi}, g_{yi})^\top$.

Support lines in the image plane are parameterized in the homogeneous form as $\mathbf{l}_j = (a_j, b_j, c_j)^\top$, with the constraint $a_j^2 + b_j^2 = 1$, where points (x, y) satisfy $a_j x + b_j y + c_j = 0$ if they belong to the line. This form is preferred over other notations (such as slope-intercept or polar), as it allows natural derivation of the maximization equations of the EM algorithm. The point-line distance in the image plane between a point, $\mathbf{r}_i$, and a line, $\mathbf{l}_j$, is defined as:

$$d(\mathbf{r}_i, \mathbf{l}_j) = \frac{\mathbf{r}_{ih}^\top \mathbf{l}_j}{\|\mathbf{m}_j\|} = \frac{a_j x_i + b_j y_i + c_j}{\sqrt{a_j^2 + b_j^2}} \tag{3.10}$$

where $\mathbf{r}_{ih}$ represents the point $\mathbf{r}_i$ in homogeneous coordinates, $\mathbf{r}_{ih} = (x_i, y_i, 1)^\top$; and $\mathbf{m}_j = (-b_j, a_j)^\top$ is the director vector of the support line. Therefore, a point lies in

a line, if $d(\mathbf{r}_i, \mathbf{l}_j) = 0$ holds. Considering Gaussian measurement noise, the likelihood of a point $\mathbf{r}_i$ to belong to a line j parameterized with $\mathbf{l}_j$ is defined as:

$$p(\mathbf{r}_i|\mathbf{l}_j) \propto \exp\left(-\frac{1}{2\sigma_{\rho,j}^2} d^2(\mathbf{r}_i, \mathbf{l}_j)\right) \tag{3.11}$$

where $\sigma_{\rho,j}$ is the standard deviation of the considered measurement noise.

For each data sample, the gradient vector $\mathbf{g}_i$ depicts the normal to the direction of the edge at $\mathbf{r}_i$. The difference between the orientation of a line j and that of the data sample is defined as:

$$d(\mathbf{g}_i, \mathbf{l}_j) = \frac{\mathbf{g}_i^\top \mathbf{m}_j}{\|\mathbf{g}_i\|\|\mathbf{m}_j\|} \tag{3.12}$$

This expression is equivalent to the cosine of the angle defined by the vectors. If the normal of a data sample is orthogonal to the direction of the line, equation (3.12) yields zero. Considering, as in the case of the position, a normal error distribution, the likelihood of a vector $\mathbf{g}_i$ to represent the normal orientation of a line is given by:

$$p(\mathbf{g}_i|\mathbf{l}_j) \propto \exp\left(-\frac{1}{2\sigma_{\phi,j}^2} d^2(\mathbf{g}_i, \mathbf{l}_j)\right) \tag{3.13}$$

where $\sigma_{\phi,j}$ is the standard deviation of the considered noise.

Assuming that the measurement of the position of data samples and their orientation are independent, the joint likelihood of $\mathbf{x}_i$ to represent a line $\mathbf{l}_j$ is given by:

$$p(\mathbf{x}_i|\mathbf{l}_j) = p(\mathbf{r}_i|\mathbf{l}_j)p(\mathbf{g}_i|\mathbf{l}_j) \tag{3.14}$$

Introducing (3.11) and (3.13) into (3.14), the expression of the likelihood of a data sample $\mathbf{x}_i$ with respect to a line j is expanded as:

$$p(\mathbf{x}_i|\mathbf{l}_j) \propto \exp\left(-\frac{1}{2}\left(\frac{(a_j x_i + b_j y_i + c_i)^2}{\sigma_{\rho,j}^2} + \frac{(a_j g_{yi} - b_j g_{xi})^2}{\sigma_{\phi,j}^2}\right)\right) \tag{3.15}$$

3.5.1.2 Vanishing point restriction

In the case the line $\mathbf{l}_j$ also passes through a given point, $\mathbf{v} = (v_x, v_y)^\top$, then the independent term of the line parameters is set as $c_j = -a_j v_x - b_j v_y$, and expression (3.10) now depends on $\mathbf{v}$ as:

$$d(\mathbf{r}_i, \mathbf{l}_j, \mathbf{v}) = \frac{a_j(x_i - v_x) + b_j(y_i - v_y)}{\sqrt{a_j^2 + b_j^2}} \tag{3.16}$$

Equations (3.11) and (3.14) are modified accordingly, whereas (3.13) remains unmodified since it does not depend on c_j:

$$p(\mathbf{r}_i|\mathbf{l}_j, \mathbf{v}) \propto \exp\left(-\frac{1}{2\sigma_{\rho,j}^2} d^2(\mathbf{r}_i, \mathbf{l}_j, \mathbf{v})\right) \tag{3.17}$$

$$p(\mathbf{x}_i|\mathbf{l}_j, \mathbf{v}) = p(\mathbf{r}_i|\mathbf{l}_j, \mathbf{v}) p(\mathbf{g}_i|\mathbf{l}_j) \tag{3.18}$$

where equation (3.18) is the expression that links each data sample with a support line that passes through a given vanishing point. For convenience, let us rewrite the likelihood as $p(\mathbf{x}_i|\mathbf{v}, \theta_j)$ where we have grouped the parameters of the line and the standard deviation of the error distributions as $\theta_j = \{\mathbf{l}_j, \sigma_{\rho,j}, \sigma_{\phi,j}\}$.

3.5.1.3 Mixture model likelihood

If we consider now M lines, the likelihood of a data sample is given by the mixture model:

$$p(\mathbf{x}_i; \Theta, \Omega) = \sum_{j=1}^{M} \omega_j \; p(\mathbf{x}_i|\mathbf{v}, \theta_j) \tag{3.19}$$

where $\Theta = \{\mathbf{v}, \theta_j\}_{j=1}^M$ and $\Omega = \{\omega_j\}_{j=1}^M$, are the parameters of the likelihood distribution. Note that the weighting factors must satisfy $\sum_{j=1}^M \omega_j = 1$, as they can be seen as the prior probability values for each component of the mixture. The error of the different support lines is assumed to follow the same type of distribution. Moreover, it is quite typical to assume that all lines share the same deviation values. If there is not any prior knowledge about the lines, it is better to work with a common pair $\{\sigma_\rho, \sigma_\theta\}$ for all the components of the mixture.

Let $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$ be the set of N data samples of the image. The joint probability density of the data samples, assumed to be i.i.d (independent and identically distributed), is given by:

$$p(\mathcal{X}|\Theta) = \prod_{i=1}^{N} p(\mathbf{x}_i; \Theta, \Omega) \tag{3.20}$$

which can also be seen as a likelihood function on the parameters of the mixture model $\mathcal{L}(\Theta|\mathcal{X}) = p(\mathcal{X}|\Theta)$. The log-likelihood function transforms the product of (3.20) into a summation, expanding the expression of the mixture as in (3.19):

$$\log \mathcal{L}(\Theta|\mathcal{X}) = \sum_{i=1}^{N} \log \sum_{j=1}^{M} \omega_j \; p_j(\mathbf{x}_i|\mathbf{v}, \theta_j) \tag{3.21}$$

The values of the parameters Θ and Ω are obtained using the Maximum Likelihood Estimation of the log-likelihood expression. The problem to be solved is enounced as follows:

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmax}} \left(\mathcal{L}(\Theta|\mathcal{X})\right) = \underset{\Theta}{\operatorname{argmax}} \left(\log \mathcal{L}(\Theta|\mathcal{X})\right) \tag{3.22}$$

where it is explicitly denoted that the maximization of the log-likelihood yields the same result as the likelihood itself and it is typically easier to solve. However, partial derivatives of the log-likelihood with respect each parameter give expressions that depend on the derivative of a natural logarithm of a sum, corresponding to the components of the mixture model, which are still not analytically tractable.

3.5.1.4 EM algorithm

The EM strategy offers a solution to this problem. Assuming the existence of a set of hidden variables $\mathcal{Y}$, which are used only internally inside the algorithm, the problem can be analytically solved [15]. Let us consider $\mathcal{X}$ as the incomplete data set, and $\mathcal{Z} = \{\mathcal{X}, \mathcal{Y}\}$ as the complete data set, including the unobserved data. The values of these hidden variables, $\mathcal{Y}$, inform about which component of the mixture generated each single observed data $\mathbf{x}_i$. That is, having $y_i \in \{1, 2, \ldots, M\}$, for each $i \in \{1, 2, \ldots, N\}$, $y_i = j$ if $\mathbf{x}_i$ belongs to the $j - th$ mixture component $p(\mathbf{x}_i|\mathbf{v}, \theta_j)$. Therefore, the expression of the complete log-likelihood is:

$$\log \mathcal{L}(\Theta|\mathcal{Z}) = \sum_{i=1}^{N} \log \left(\omega_{y_i} p(\mathbf{x}_i|\mathbf{v}, \theta_{y_i})\right) \tag{3.23}$$

which is much simpler to operate with than the incomplete log-likelihood.

The EM algorithm proceeds iteratively [15], applying two steps: the E-step, that implies the differentiation of the analytical expression of the function to maximize; and the M-step, which actually maximizes over the target parameters. At each iteration, the algorithm considers the whole data set, $\mathcal{X}$, and the previous estimation of the parameters, Θ^*.

Specifically, the E-step finds the expected value of the complete-data log-likelihood with respect to the unknown data $\mathcal{Y}$ given the observed data $\mathcal{X}$ and the current parameter estimates, Θ^*. That is:

$$Q(\Theta, \Theta^*) = E_{\mathcal{Y}}\left[\log(\mathcal{X}, \mathcal{Y}|\Theta)|\mathcal{X}, \Theta^*\right] = \int_{\mathbf{y}\in\Upsilon} \log p(\mathcal{X}, \mathbf{y}|\Theta) f(\mathbf{y}, \mathcal{X}, \Theta^*) d\mathbf{y} \tag{3.24}$$

where $\mathbf{y}$ is a particular realization of the set of random variable $\mathcal{Y}$ and Υ is the complete set of possible $\mathbf{y}$ vectors. For the defined mixture model, this expression is (based on derivations shown by Bilmes [15]:

$$Q(\Theta, \Theta^*) = \sum_{j=1}^{M}\sum_{i=1}^{N} \log(\omega_j) p(j|\mathbf{x}_i, \Theta^*) + \sum_{j=1}^{M}\sum_{i=1}^{N} \log(p(\mathbf{x}_i|\mathbf{v}, \theta_j)) p(j|\mathbf{x}_i, \Theta^*) \tag{3.25}$$

where the conditional probability of the samples to belong to a specific component is given by:

$$p(j|\mathbf{x}_i, \Theta^*) = \frac{\omega_j^* p(\mathbf{x}_i|\mathbf{v}, \theta_j^*)}{\sum_{k=1}^{M} \omega_k^* p(\mathbf{x}_i|\mathbf{v}, \theta_k^*)} \tag{3.26}$$

From here on, $p(j|\mathbf{x}_i, \Theta^*)$ will be denoted as γ_{ij} for simplicity, since it has a single value for each data sample, indexed by i with respect to each mixture model component, indexed by j.

The first term of $Q(\Theta, \Theta^*)$, is maximized over the weights ω_j independently from the second term, which can be maximized over the set of parameters θ_j. These processes compose the M-step, which searches for the maximum values of the considered parameters according to the existing data set and the previous parameter estimation.

The maximization of the first term of (3.25) is done by differentiation with respect to ω_j and equaling to zero. The problem is enounced taking into account the restriction $\sum_{j=1}^{M} \omega_j = 1$ and applying Lagrange multipliers. It can be shown that the estimation is $\hat{\omega}_j = \frac{1}{N}\sum_{i=1}^{N} \gamma_{ij}$, which, as expected, is independent of the rest of parameters of the model.

The maximization of the second term of (3.25) is more complex as it involves several variables, which are dependent one from each other. For the parameters that govern the considered M lines, the differentiation is applied over σ_ρ, σ_ϕ first, and secondly over a_j, b_j and c_j.

Extending the second term of (3.25) with (3.15), and knowing that the components of the mixture are Gaussian, the expression to be maximized is:

$$Q_2 = \sum_{j=1}^{M}\sum_{i=1}^{N}\Big(-2\log\sigma_\rho - 2\log\sigma_\phi - \frac{1}{2\sigma_\rho^2}(a_j(x_i - v_x) + b_j(y_i - v_y))^2 - \frac{1}{2\sigma_\phi^2}(a_j g_{y,i} - b_j g_{x,i})^2\Big)\gamma_{ij} + \lambda(a_j^2 + b_j^2 - 1) \tag{3.27}$$

The differentiation over σ_ρ and σ_ϕ is done first. As abovementioned, we will suppose at this point that all lines share the same deviation values, which yields:

$$\hat{\sigma}_\rho^2 = \frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{N}\left(((x_i - v_x)a_j + (y_i - v_y)b_j)^2\right)\gamma_{ij} \tag{3.28}$$

$$\hat{\sigma}_\phi^2 = \frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{N}\left((a_j g_{y,i} - b_j g_{x,i})^2\right)\gamma_{ij} \tag{3.29}$$

For the maximization of the line parameters, Lagrange multipliers are required to add the restriction $a^2 + b^2 = 1$. Removing the constant elements not dependent on any of the line parameters the differentiation is carried out for the following expression:

$$Q_2 = \sum_{j=1}^{M}\sum_{i=1}^{N}\Big(-\frac{1}{2\hat{\sigma}_\rho^2}(a_j(x_i - v_x) + b_j(y_i - v_y))^2 - \frac{1}{2\hat{\sigma}_\phi^2}(a_j g_{y,i} - b_j g_{x,i})^2\Big)\gamma_{ij} + \lambda(a_j^2 + b_j^2 - 1) \tag{3.30}$$

For convenience, let us rewrite (3.30) defining $\tilde{x}_i$ and $\tilde{y}_i$ as $\tilde{x}_i = x_i - v_x$ and $\tilde{y}_i = y_i - v_y$:

$$Q_2 = \sum_{j=1}^{M}\sum_{i=1}^{N}\Big(-\frac{1}{2\hat{\sigma}_\rho^2}(a_j\tilde{x}_i + b_j\tilde{y}_i)^2 - \frac{1}{2\hat{\sigma}_\phi^2}(a_j g_{y,i} - b_j g_{x,i})^2\Big)\gamma_{ij} + \lambda(a_j^2 + b_j^2 - 1) \tag{3.31}$$

The maximization of the sum over the M elements is achieved by maximizing each element. Considering only one component, the partial derivatives deliver the following expressions:

$$\frac{\partial Q_2}{\partial a_j} = \sum_{i=1}^{N}\Big(\frac{1}{\hat{\sigma}_\rho^2}(a_j\tilde{x}_i^2 + b_j\tilde{x}_i\tilde{y}_i) + \frac{1}{\hat{\sigma}_\phi^2}(a_j g_{y,i}^2 - b g_{x,i} g_{y,i})\Big)\gamma_{ij} + 2a_j\lambda = 0 \tag{3.32}$$

$$\frac{\partial Q_2}{\partial b_j} = \sum_{i=1}^{N}\Big(\frac{1}{\hat{\sigma}_\rho^2}(a_j\tilde{x}_i\tilde{y}_i + b_j\tilde{y}_i^2) + \frac{1}{\hat{\sigma}_\phi^2}(-a_j g_{x,i} g_{y,i} - b g_{x,i}^2)\Big)\gamma_{ij} + 2b_j\lambda = 0 \tag{3.33}$$

which is a linear system of equations on the line parameters. For the sake of simplicity in the notation, we use the expected value of a generic data set $\{z_i\}_{i=1}^N$, defined as $E(z) = \sum_{i=1}^N z_i \gamma_{ij}$ provided that $\sum_{i=1}^N \gamma_{ij} = 1$. This way, (3.32) and (3.33) are rewritten as:

$$a\left(\frac{E(\tilde{x}^2) - E^2(\tilde{x})}{\hat{\sigma}_\rho^2} + \frac{E(g_y^2)}{\hat{\sigma}_\phi^2} - 2\lambda\right) + b\left(\frac{E(\tilde{x}\tilde{y}) - E(\tilde{x})E(\tilde{y})}{\hat{\sigma}_\rho^2} - \frac{E(g_x g_y)}{\hat{\sigma}_\phi^2}\right) = 0 \quad (3.34)$$

$$a\left(\frac{E(\tilde{x}\tilde{y}) - E(\tilde{x})E(\tilde{y})}{\hat{\sigma}_\rho^2} - \frac{E(g_x g_y)}{\hat{\sigma}_\phi^2}\right) + b\left(\frac{E(\tilde{y}^2) - E^2(\tilde{y})}{\hat{\sigma}_\rho^2} + \frac{E(g_x^2)}{\hat{\sigma}_\phi^2} - 2\lambda\right) = 0 \quad (3.35)$$

The matrix form of the linear problem defined by equations (3.34) and (3.35) is obtained by moving the elements depending on λ to the right term, yielding

$$C\mathbf{n}_j = \lambda \mathbf{n}_j \quad (3.36)$$

where $\mathbf{n}_j = (a_j, b_j)^\top$, and $C = \frac{1}{\hat{\sigma}_\rho^2} A + \frac{1}{\hat{\sigma}_\phi^2} B$. Matrices A and B contain, respectively, the information given by the weighted samples of position and orientation:

$$A = \begin{pmatrix} E(\tilde{x}^2) - E^2(\tilde{x}) & E(\tilde{x}\tilde{y}) - E(\tilde{x})E(\tilde{y}) \\ E(\tilde{x}\tilde{y}) - E(\tilde{x})E(\tilde{y}) & E(\tilde{y}^2) - E^2(\tilde{y}) \end{pmatrix} \quad (3.37)$$

$$B = \begin{pmatrix} E(g_y^2) & -E(g_x g_y) \\ -E(g_x g_y) & E(g_x^2) \end{pmatrix} \quad (3.38)$$

The meaning of these matrices is easier to understand if we analyze them independently. On the one hand, solving $A\mathbf{n}_j = \lambda \mathbf{n}_j$ gives the line parameters, $(\hat{a}_j, \hat{b}_j)$, as in standard total least squares line fit using points. On the other hand, solving $B\mathbf{n}_j = \lambda \mathbf{n}_j$ gives an estimate of $(\hat{a}_j, \hat{b}_j)$ in such a way that gradient polarity is properly taken into account. Matrix B has actually the form of the well known second moment matrix (or second structure tensor), but with switched diagonal terms such that the smaller eigenvector gives the normal to the edge. Using the second moment matrix to characterize gradients has the advantage that gradient orientations in opposing directions are properly handled (i.e., gradients in opposing directions do not cancel each other as it would be the case with the simple mean of the gradient components).

Therefore, (3.36) is a weighted combination of the contribution of the position information, contained in A, and the orientation information given in B, with the restriction that the parameters are normalized. This combination takes advantage of the two different sources of information to generate more accurate estimations. Moreover, as it will be shown in the test analysis, the estimator composed by these matrices achieves faster convergence than simple point-based EM strategies.

Differentiation of (3.30) with respect to the vector $\mathbf{v}$ requires rewriting it in vector format:

$$Q_2 = \sum_{j=1}^{M}\sum_{i=1}^{N}\Big(-\frac{1}{2\hat{\sigma}_\rho^2}(\mathbf{r}_i^\top N_j \mathbf{r}_i + 2\mathbf{r}_i^\top N_j \mathbf{v} - \mathbf{v}^\top N_j \mathbf{v}) - \frac{1}{2\hat{\sigma}_\phi^2}(\mathbf{m}_j^\top \mathbf{g}_i)^2\Big)\gamma_{ij} + \lambda(a_j^2 + b_j^2 - 1) \quad (3.39)$$

where $N_j = \mathbf{n}_j\mathbf{n}_j^\top$ is a 2×2 symmetric matrix. Knowing that $\frac{\partial}{\partial \mathbf{u}}[\mathbf{u}^\top A\mathbf{u}] = 2A\mathbf{u}$ for any $k \times k$ symmetric matrix A and $k \times 1$ vector $\mathbf{u}$, the differentiation yields:

$$\frac{\partial Q_2}{\partial \mathbf{v}} = \sum_{j=1}^{M}\sum_{i=1}^{N}\big(2\mathbf{r}_i^\top N_j - 2N_j\mathbf{v}\big)\gamma_{ij} = 0 \quad (3.40)$$

From this expression, the estimation of the vanishing point is given by working out $\mathbf{v}$ as:

$$\hat{\mathbf{v}} = \left(\sum_{j=1}^{M}\sum_{i=1}^{N} N_j\gamma_{ij}\right)^{-1}\sum_{j=1}^{M}\sum_{i=1}^{N}\mathbf{r}_i^\top N_j\gamma_{ij} \quad (3.41)$$

As expected, the maximum likelihood estimation of a_j and b_j depend on $\mathbf{v}$ as in (3.37), and viceversa, as shown in (3.41). Therefore, the maximization process must be done in a two-steps process, which is known as gradual M-step [76]. Hence, it is necessary to proceed fixing one of this parameters and updating the other, so that in two steps, both parameters are updated.

3.5.1.5 Control of outliers

The selection of the number of components of the mixture model is an important issue. This is a very well known problem for real images, where two possibilities arise: (i) the data set is actually distributed in a fixed number of well-defined clusters. Although this approach has been already addressed by Minagawa et al. [76], the presence of outliers would dramatically corrupt their results (as the estimation holds on a least squares procedure); (ii) a more realistic scenario considers that there is not a clearly defined number of clusters of the data. Instead, data is, to some extent, smoothly distributed in the image and partially grouped around significant clusters.

The second case is addressed in these sections, where there are a number of significant lines (support lines), gathering a large set of data samples, but also a non-negligible number of them not belonging to any line. These data samples will be considered as outliers for the model.

Therefore, in our EM scheme proposal, the presence of outliers is handled with the inclusion of an quasi-uniform distribution as an additional component for the mixture

model, that is denoted as outlier distribution. That way, M components are assumed by the EM to represent the data samples drawn from the M most significant lines in the scene. The rest of the data samples are absorbed by the outlier distribution. Hence, in our approach, the complete mixture model has $M+1$ components.

The mixture model (3.19) is rewritten as:

$$p(\mathbf{x}_i;\Theta,\Omega)=\sum_{m=1}^{M+1}\omega_m\ p_m(\mathbf{x}_i|\mathbf{v},\theta_m)=\omega_{out}\ p_{out}(\mathbf{x}_i)+\sum_{j=1}^{M}\omega_j\ p(\mathbf{x}_i|\mathbf{v},\theta_j) \tag{3.42}$$

where the components $j=1,2,\ldots,M$ are the ones already described, and $p_{out}(\mathbf{x}_i)$, is a normal distribution with fixed large variance, which behaves approximately as a uniform distribution. The use of such a normal distribution has been already addressed by other authors like Košecká and Zhang [65]. The reason of choosing this approximation to the uniform instead of the uniform itself is because the EM algorithm for a mixture of Gaussians is much more simple to compute, and the equations derived in previous paragraphs are still valid with the addition of this distribution. The parameters of the outlier distribution are kept fixed, i.e. the M-step does not modify its values, so that the normal distribution is not modified and is always similar to the uniform.

The weight of the outlier distribution is also kept fixed as ω_{out}. This way, the equations of the EM algorithm are modified accordingly. The conditional probability, shown in (3.26), now includes the uncertainty component:

$$p(j|\mathbf{x}_i,\Theta^*)=\frac{\omega_j^* p(\mathbf{x}_i|\mathbf{v},\theta_j)}{\omega_{out}p(\mathbf{x}_i)+\sum_{k=1}^{M}\omega_k^* p(\mathbf{x}_i|\mathbf{v},\theta_k)} \tag{3.43}$$

Analogously, the estimation of the weights of the components at each iteration is now given by:

$$\hat{\omega}_j=\frac{(1-\omega_{out})\sum_{i=1}^{N}\gamma_{ij}}{\sum_{j=1}^{M}\sum_{i=1}^{N}\gamma_{ij}} \tag{3.44}$$

Note that the M-step is not modified since the parameters to be estimated remain the same.

3.5.1.6 Tests and results

In this section several experiments are described, whose results demonstrate the benefits of the proposed image-plane EM algorithm, addressing different problems and how this approach faces them and improve upon the basic approach by Minagawa et al. [76].

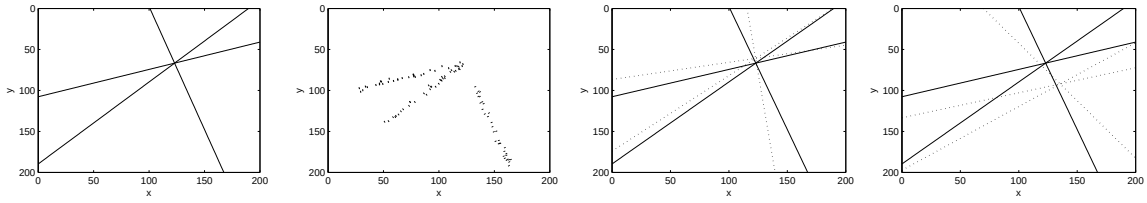


Figure 3.11: Ground truth lines used for the experiments and noisy edge-points drawn from them: (a) three converging lines; and (b) the corresponding data set. In (c) and (d) two different initializations are shown in dotted lines.

Along this section and its subsections, the proposed image-plane EM algorithm will be denoted in different ways according to which of the defined feats are used at each test: the image-plane EM algorithm will be denoted as vpEM (i.e. vector-point EM) when considering only the use of orientation information inside the EM framework with the basic pEM (point EM) framework by Minagawa et al. [76]. Analogously, when dealing with the control of outliers, the method will be denoted as vpUEM, where the "U" character points out the use of an additional quasi-Uniform distribution. This way, vpEM refers to the proposed image-plane EM algorithm without using the additional outlier distribution, but using the orientation information of data samples, and vpUEM is the complete image-plane EM algorithm.

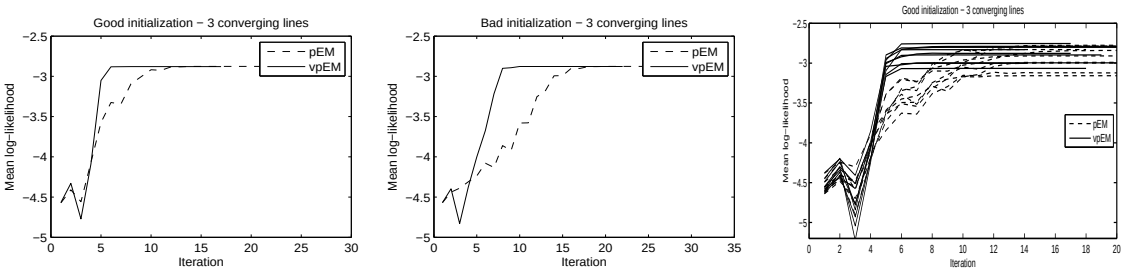
First, the performance of the vpEM is compared with the pEM. The aim is to demonstrate that since the vpEM includes the orientation information of data samples, it results in a better utilization of the image information with respect to the pEM. Results can be summarized as achieving faster convergence, requirement of less amount of data to achieve equal accuracy, and better response against erroneous initializations.

Different tests have been carried out using synthetic data, i.e. data that is obtained by noisy sampling a set of ground truth simulated gradient-pixels in the plane. Secondly, the benefits of the introduction of the outlier distribution are discussed, based on the results for different synthetic examples which includes a significant number of outliers.

The real images case is considered afterwards, where the importance of the quasi-uniform distribution to handle outliers is clearly demonstrated to achieve the required level of accuracy.

Synthetic data tests

The usage of orientation information for line fitting is a clear advantage compared to pointwise information. Intuitively, this is because each data sample, $\mathbf{x}_i = \{\mathbf{r}_i, \mathbf{l}_i\}$, directly gives a measurement of the parameters of the support line, while two points are required to define a line (which is the case of the pEM).



(a) (b) (c)

Figure 3.12: Values of mean log-likelihood (in (a,b)), $\bar{\mathcal{L}}(\Theta|\mathcal{X})$, for the two cases shown in figure 3.11 (c) and (d). In (c), the convergence is shown for the case in which the number of data samples is reduced.

For the experiments, let us consider an example of ground truth support lines meeting at a common vanishing point, which are depicted in figure 3.11. A set of $N = 100$ simulated gradient-pixels are drawn from these lines, adding noise to their position and orientation, as defined by the likelihood model of section 3.5.1.1. For these experiments we set $\sigma_\rho = 5$ (pixels) and $\sigma_\phi = 0.1$ (rads), which are values similar to those that could be expected for real images.

We consider two initializations of the parameters to be estimated in each case. These are depicted in figure 3.11 (c) and (d). One of them is close to the ground truth, while the other is significantly different. The pEM and vpEM algorithms are executed on these examples until they converge. Convergence is defined as it was for MLESAC (see section 3.4.1.1), using the mean log-likelihood value, which is given now by:

$$\bar{\mathcal{L}}(\Theta|\mathcal{X}) = \frac{1}{N}\sum_{i=1}^{N}\log\sum_{j=1}^{M}\omega_j\ p(\mathbf{r}_i|\mathbf{v},\theta_j) \tag{3.45}$$

Realize that the model $p(\mathbf{r}_i|\mathbf{v},\theta_j)$ is used in (3.45) instead of $p(\mathbf{x}_i|\mathbf{v},\theta_j)$, as defined in (3.18), to be able to fairly compare the results of pEM and vpEM, since the pEM algorithm does not include the gradient information of the data samples, contained in $\mathbf{x}_i$ but not in $\mathbf{r}_i$.

Although in all cases both pEM and vpEM converge to approximately the ground truth set of parameters, the vpEM algorithm shows faster convergence, i.e. it requires less number of iterations to converge. Figure 3.12 (a,b) shows the evolution of the mean log-likelihood at each iteration for the pEM and vpEM for the two different initializations. Realize that the EM procedure shows some instability, and in some iterations $\Delta\bar{\mathcal{L}}_t < 0$. This is due to the applied gradual M-step, where two steps are required to compute the whole set of parameters, such that at each single M-step the

Table 3.5: Mean number of iterations for pEM and vpEM (100 tests)

Parameters noise	# it. pEM	# it. vpEM
$\{\sigma_{a,b} = 0.10, \sigma_{v_{x,y}} = 10\}$	8.4	4.7
$\{\sigma_{a,b} = 0.15, \sigma_{v_{x,y}} = 15\}$	9.2	5.1
$\{\sigma_{a,b} = 0.20, \sigma_{v_{x,y}} = 20\}$	12.7	6.5
$\{\sigma_{a,b} = 0.25, \sigma_{v_{x,y}} = 25\}$	16.4	7.8

likelihood is not guaranteed to be increased. Nevertheless, considering two consecutive iteration steps, $\Delta\bar{\mathcal{L}}_t > 0$, which is in line with the expected convergence property of the EM algorithm.

The experiment was repeated 100 times, using different initializations of the parameter set (the position of the vanishing point and the orientation of the lines), with similar results. For convenience, the parameters were initialized randomly, using zero mean Gaussian noise and a defined standard deviation for each parameter. Table 3.5 shows the results for different values of the noise added to the initialization parameters. The results are shown in terms of mean number of iterations to converge for both the pEM and vpEM algorithms. The contents of the table reveal that the number of iterations required by the pEM increases constantly as the initialization get worse, while for the vpEM, the number of iterations grows much slowly than for pEM. The conclusion is that the utilization of orientation information makes the system much faster and less dependent on how much the initialization deviates from the ground truth model parameters.

Another important property of the vpEM is that convergence is achieved even for situations in which the number of data samples is reduced. Conversely, the pEM decreases its performance severely in these situations. The same synthetic data set shown in figure 3.11 (b) is used, and the convergence results are shown in figure 3.12 (c) when the number of data samples suffers a 10% to a 90% reduction. It is clear from the figure that the convergence of the vpEM is fixed in approximately 5 iterations, while the pEM needs more iterations as the set of samples is reduced (from 8 to 12 iterations). For sub-sets with less data samples both methods fail to converge.

Finally, the performance of the vpUEM (i.e., the complete proposed method, including the component inside the mixture model to handle outliers) is tested in the presence of noisy data samples randomly distributed among the image. Note that this situation can not be handled by pEM since its aims is to find the maximum likelihood of its model for the whole set of data, including the outliers, which actually do not follow any of the support lines. Some results of the experiment are shown in figure 3.13. As shown, different number of outliers are added to the data set of the example of figure 3.11. The outlier distribution of the mixture model makes that most part of outliers are discarded for the estimation process of the hypothesized lines, so that the

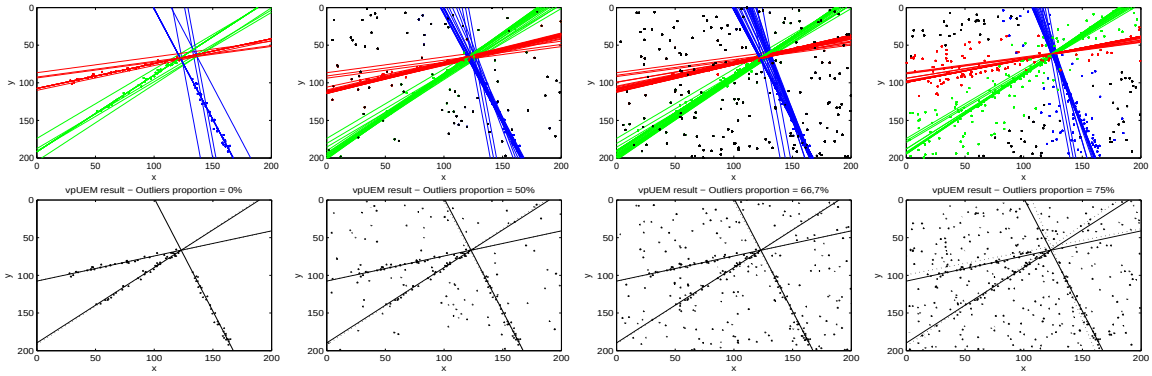


Figure 3.13: Convergence of vpEM for different outlier ratios: 0%, 50%, 66.7% and 75% from left to right. The upper row shows the iterative process that leads to the result. The lower row shows, in dotted lines, the result of the iteration process. In solid lines, the ground truth.

final result is correct. The vpUEM shows excellent results in all cases and only begins to suffer small errors when the proportion of outliers reaches values around 75%. The iteration process of the vpUEM algorithm is shown in the upper row of the figure, and the final result, in the bottom row. The color code is the following: blue, red and green lines are fitted using the corresponding colored data samples, which are automatically selected. The black data samples are those absorbed by the uncertainty component.

Real data examples

Real images as those shown in figure 3.14 clearly contain edges not only meeting at one vanishing point, but also oriented to other vanishing points and arbitrary shapes (people, shadows, etc.) not meeting any vanishing point and that are considered as outliers for the estimation process. Nonetheless, the vpUEM, as demonstrated with the synthetic data tests, has been designed to cope with this difficulty, handling the presence of outliers and working well even with dramatically erroneous initializations.

In this section we will consider images that actually contain lines that intersect on $\mathbb{R}^2$, that can be classified into two groups: (i) images having a vanishing point inside the limits of the image well supported by the edge dataset. This scenario is typically analyzed in online applications for which the translational movement of the camera follows the vanishing direction, being of great interest for many applications such as hand-held camera reconstruction of indoor scenes, obstacle detection for vision based road safety systems, etc.; (ii) the cases where the vanishing point is outside the image are also considered, e.g. many typical pictures of buildings or surveillance cameras in indoor environments for 3D reconstruction, image rectification, etc.

The examples given in figure 3.14 illustrate the performance of the system for the abovementioned cases. It depicts the initialization of the parameter set corresponding

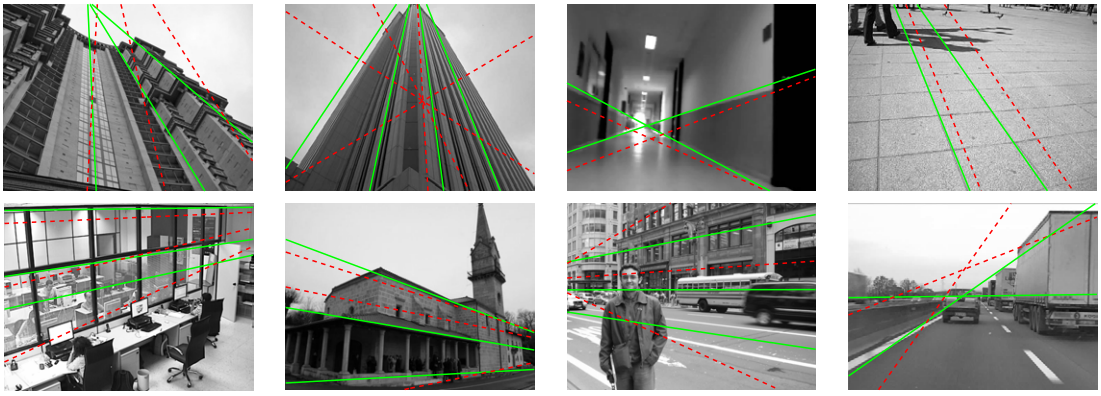
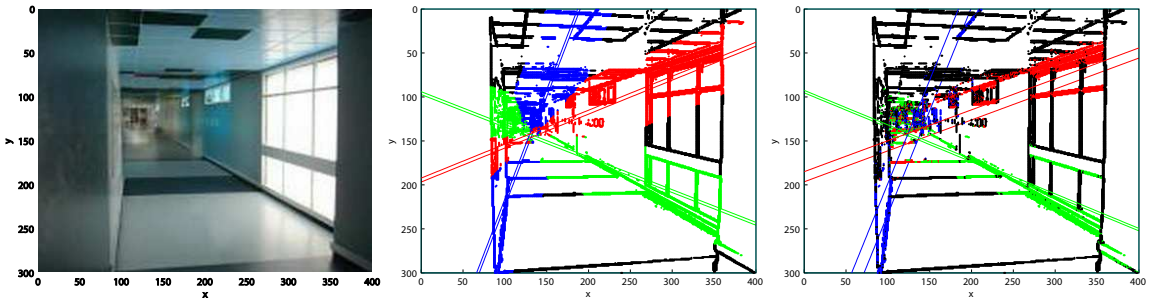


Figure 3.14: Example of the use of vpUEM on different real images.

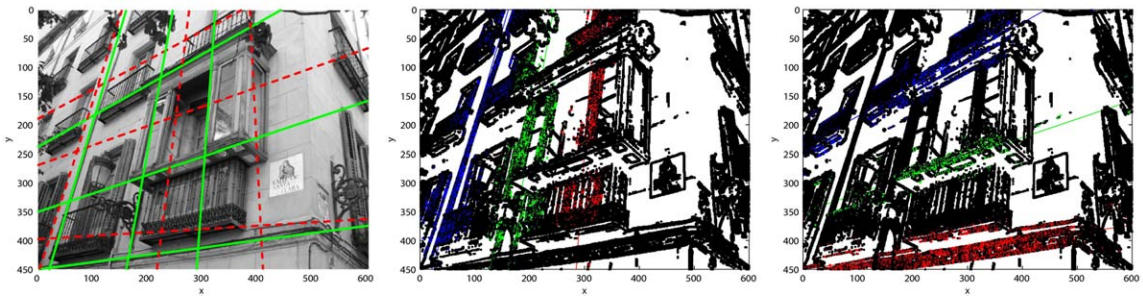


(a) (b) (c)

Figure 3.15: Orientation information makes the classification of samples more accurate and allows finding better estimations at each iteration. Therefore, the color code assignment is used only for visualization purposes: (a) original image of a corridor; (b) classification of samples without using the vector-point information (pUEM); and (c) classification using vpUEM.

to M straight lines (shown dashed and red in the figure) intersecting at a single vanishing point, and the result given by the vpUEM as solid-green lines. For these tests $M = 3$ or $M = 4$ is enough to achieve fast convergence and accurate results. As shown, the result of the vpUEM is very accurate for all the examples, even for the difficult ones, such as those of images containing arbitrary shapes (like people, shadows, vehicles, etc.), and for quite erroneous initializations. In simpler cases, as those of images that contain a significant number of straight lines meeting the vanishing point, such as those of skyscrapers, vpUEM reaches a very accurate fit of the model to the data even with extremely inaccurate initializations, such as the one shown in the second example of figure 3.14 (from left to right and up to bottom).

The example of figure 3.15 shows the importance of the orientation information in the case of real images. One iteration step of the vpUEM is shown in (c), for the



(a) (b) (c)

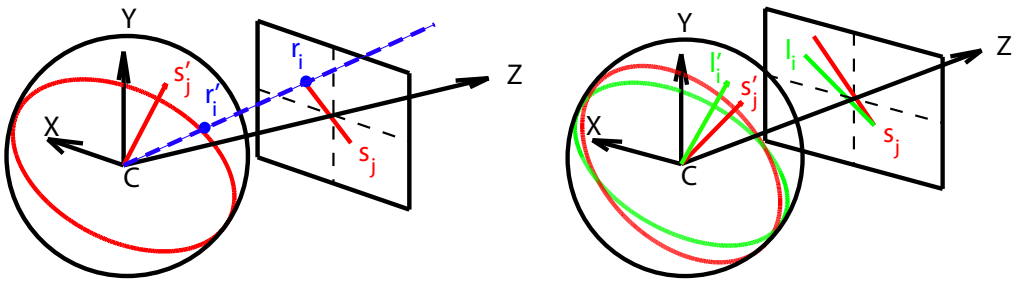
Figure 3.16: Estimation of multiple vanishing points: (a) two vanishing points detection as the intersection of the two sets of lines; (b) and (c) show the soft assignment of data samples to the components of the mixture in the last iteration of the vpUEM algorithm.

example image of (a), where data samples are colored accordingly to the component for which they have higher conditional probability, $p(j|\mathbf{x}_i, \Theta^*)$. The vpUEM obtains significantly better classification that pUEM (the version of the vpUEM that do not use the orientation information of edge points), shown in (b), even with important initialization error.

The presence of multiple vanishing points is naturally handled by running as many vpUEM procedures as desired, as done by the RANSAC approaches described along this chapter. Figure 3.16 shows one example of the application of the vpUEM strategy on an image of a facade containing two dominant vanishing points. In (a), the initialization and result of both vanishing points are depicted following the same color code as in previous figures. The data samples that are not classified to any support lines are depicted in (b) and (c) as black dots. Note that, for each vanishing point, the data samples meeting at the other vanishing point act as outliers. Figures (b) and (c) illustrate the data samples that have been assigned during the last E-step to each support line respectively for both vanishing points. The amount of data samples in black with respect to the colored ones illustrate the proportion of outliers that the vpUEM has managed at each situation. At each iteration, these outliers are sequentially absorbed by the outlier distribution of the mixture model until the estimation of the vanishing point considers only the colored points shown in (b) and (c), achieving a very high accurate estimation of vanishing points.

3.5.2 Projective-plane EM algorithm

Once the proposed image-plane EM strategy has been described, the next subsections explain the extension for a more general solution in the projective plane. The concepts



(a) (b)

Figure 3.17: Point-line and line-line distances in the projective plane. The unit sphere visualization helps to understand the proposed distances as relationship between vectors joining the optical center C and points in the surface of the sphere. For clarity, in this figure, calibrated coordinates are depicted with a tilde.

about data calibration and normalization explained in appendix B are used along these subsections[12].

3.5.2.1 Problem statement

Given a measured data set, such as a set of line segments, or significant gradient pixels, $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{N}$, vanishing point estimation techniques typically focalize only on the optimization of the parameters of the vanishing point, $\mathbf{v}$ (or the set of vanishing points).

In this section, the dimensionality of the problem is augmented, since the set of parameters to be optimized is given now by a set of vanishing points $\{\mathbf{v}_k\}_{k=1}^{V}$ and a set of support or main lines that pass through them, $\{\mathbf{s}_{jk}\}_{j=1,k=1}^{M,V}$. Therefore, the set of parameters to be estimated is $\Theta = \{\{\mathbf{v}_k\}_{k=1}^{V}, \{\mathbf{s}_{jk}\}_{j=1,k=1}^{V,M}, \sigma_\phi, \sigma_\rho\}$, where σ_ϕ and σ_ρ are the standard deviation for the normal distributions in orientation and position, respectively.

3.5.2.2 Likelihood models

Let us define the relationship between data samples and the set of support lines. The likelihood model is governed by two equations, which link each data sample, $\mathbf{x}_i = \{\mathbf{r}_i, \mathbf{l}_i\}$, with lines meeting at a common vanishing point, $\{\mathbf{s}_j\}_{j=1}^{M}$ in a similar way as in (3.14):

[12]To avoid unnecessary notation, the tilde (introduced in appendix B) will be removed from the equations, and all variables will be supposed to be calibrated, except for the cases in which explicitly uncalibrated data is treated, and for the figures that show both calibrated and uncalibrated data.

$$p(\mathbf{x}_i|\mathbf{s}_j) = p(\mathbf{r}_i|\mathbf{s}_j)p(\mathbf{l}_i|\mathbf{s}_j) \tag{3.46}$$

where now the point-line and line-line distances are given by three-vector dot products:

$$p(\mathbf{r}_i|\mathbf{s}_j) \propto \exp\left(-\frac{1}{2\sigma^2_{\rho,jk}}\left(\mathbf{r}_i^\top \mathbf{s}_j\right)^2\right) \tag{3.47}$$

$$p(\mathbf{l}_i|\mathbf{s}_j) \propto \exp\left(-\frac{1}{2\sigma^2_{\phi,jk}}\left(1-\left(\mathbf{l}_i^\top \mathbf{s}_j\right)^2\right)\right) \tag{3.48}$$

Figure 3.17 illustrates these functions. The point-line distance, $\mathbf{r}_i^\top \mathbf{s}_j$ in (3.47) assumes that vectors are normalized such that $||\mathbf{r}_i|| = ||\mathbf{s}_j|| = 1$. Therefore, it corresponds to the cosine of the angle defined by the vector that joins the optical center C with the point $\mathbf{r}_i$ and the normal vector of the plane defined by the great circle that corresponds to the line $\mathbf{s}_j$[13]. This way, if the point belongs to the line in the image plane, its associated 3D vector is orthogonal to the normal vector of the interpretation plane, and (3.47) is maximum.

The line-line distance, considering also normalized vectors, is related to the sine of the angle between the interpretation planes corresponding to $\mathbf{l}_i$ and $\mathbf{s}_j$. Concretely, given that $\cos(\theta) = \left(\mathbf{l}_i^\top \mathbf{s}_j\right)$ and $\sin^2(\theta) = (1-\cos^2(\theta))$ then $\sin^2(\theta) = 1-\left(\mathbf{l}_i^\top \mathbf{s}_j\right)^2$. Therefore, if the lines are equal, these normal vectors are coincident and the sine of their angle is zero, so that the likelihood as in (3.48) is maximum.

The combination of these two likelihood models provide a more robust approach than if only one type of information is considered.

The presence of multiple data samples and multiple lines to be fitted is handled in a similar way as shown in section 3.5.1. The result is a likelihood model for a mixture model of M support lines clustered in V groups corresponding to different vanishing points in the scene plus an additional component to handle outliers:

$$p(\mathbf{x}_i;\Theta,\Omega) = \sum_{k=1}^{V}\sum_{j=1}^{M}\omega_{jk}p(\mathbf{x}_i|\theta_{jk}) + \omega_{\text{out}}p_{\text{out}}(\mathbf{x}_i) \tag{3.49}$$

where $\Theta = \{\theta_{jk}\}_{j=1,k=1}^{M,V}$ and $\theta_{jk} = \{\mathbf{s}_{jk}, \sigma_{\phi,jk}, \sigma_{\rho,jk}\}$, and $\Omega = \{\{\omega_{jk}\}_{j=1,k=1}^{M,V}, \omega_{\text{out}}\}$. This expression already includes a component of the mixture devoted for outliers, such that the weights sum to one: $\left(\sum_{k=1}^{V}\sum_{j=1}^{M}\omega_{jk}\right) + \omega_{\text{out}} = 1$. The outlier distribution can be

[13]This plane is denominated "Edge Plane" by Antone and Teller [4], although is more frequently used the term "Interpretation plane" as done by Pflugfelder [93].

defined as in section 3.5.1. Note that although the standard deviation of the support lines could be different according to the indexes j and k, for simplicity we will consider that there is a common shared value $\sigma_{\phi,jk} = \sigma_\phi$ and $\sigma_{\rho,jk} = \sigma_\rho \; \forall j = 1 \dots M, k = 1 \dots V$.

The presence of multiple i.i.d. data samples yields the incomplete likelihood function $p(\mathcal{X}|\Theta) = \prod_{i=1}^{N} p(\mathbf{x}_i; \Theta, \Omega)$, and the log-likelihood function, which now includes the summation over line clusters indexed by k:

$$\log \mathcal{L}(\Theta|\mathcal{X}) = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{V} \sum_{j=1}^{M} \omega_{jk} p(\mathbf{x}_i|\theta_{jk}) + \omega_{\text{out}} p_{\text{out}}(\mathbf{x}_i) \right) \tag{3.50}$$

Provided the likelihood model for data samples and support lines, let us now consider the relationship between vanishing points and data samples. This is necessary because the definition of likelihood model in (3.50) allows to compute the parameters related to the sets of lines $\{\mathbf{s}_{jk}\}_{j=1,k=1}^{M,V}$, but not the positions of the vanishing points $\{\mathbf{v}_k\}_{k=1}^{V}$ since their parameters do not appear in the expression (3.46). Therefore, it is required to add additional models to consider the position of the vanishing points.

For this purpose, any of the evaluated error functions described in section 3.3 can be used. For instance, the most simple approach could be to use the calibrated point-line distance, $\mathbf{l}_i^\top \mathbf{v}$. This distance is equivalent to the point-line distance $\mathbf{s}_j^\top \mathbf{r}_i$ that was illustrated in figure 3.17. Nevertheless, it has been shown in section 3.3 that a better approach is based on the orientation deviation between $\mathbf{l}_i$ and a line joining the vanishing point and a reference point, which is the mid-point of the line segment (the so-called OS error function). Therefore, we propose to use the error function defined in equation (3.1). It is important to realize that this error function is defined on uncalibrated coordinates. Therefore, the likelihood of a data sample $\mathbf{x}_i$ given a vanishing point $\mathbf{v}$ is:

$$p(\mathbf{x}_i|\mathbf{v}) \propto \exp\left(-\frac{1}{2\sigma^2} d^2(\mathbf{x}_i, \mathbf{v}) \right) \tag{3.51}$$

where $d(\mathbf{x}_i, \mathbf{v})$ is defined as in equation (3.1).

3.5.2.3 EM algorithm

Provided the definition of the likelihood models, the EM procedure can be described. Analogously to the solution for the image plane, given in subsection 3.5.1, the process has to be done gradually, i.e. with several M-steps and, for this case, also several E-steps.

The following paragraphs summarize the proposed way to proceed, which updates in its first M-step the position of the vanishing points, $\{\mathbf{v}_k\}_{k=1}^{V}$, and in the second

one, the parameters of the support lines, $\{\mathbf{s}_{jk}\}_{j=1,k=1}^{M,V}$. Another possible way is the opposite, i.e. update first the position of the support lines and then the position of the vanishing points. Nevertheless, in the author's opinion, it is much less prone to errors to update the vanishing point in the first iteration, so that the E-step "classifies" data samples according to which vanishing point are they meeting, and then update the support lines.

The proposed sequence of steps, at each iteration, is the following:

1. **E-step(1)** Compute the values of conditional probability for the set of data samples with respect to the set of vanishing points and corresponding support lines. Given the current best estimation of the parameters of the model, these probabilities are computed as:

$$p(j,k|\mathbf{x}_i,\Theta^*) = \frac{\omega^*_{jk} p(\mathbf{x}_i|\theta^*_{jk})}{\omega_{\text{out}} p_{\text{out}}(\mathbf{x}_i) + \sum_{k'=1}^{V}\sum_{j'=1}^{M} \omega^*_{j'k'} p(\mathbf{x}_i|\theta^*_{jk})} = \gamma_{ijk} \tag{3.52}$$

   where $p(\mathbf{x}_i|\theta^*_{jk})$ is computed as defined in (3.46).

2. **M-step(1)** Once the conditional probabilities have been computed, the first maximization step estimates the position of the set of vanishing points according to the selected error function.

   In the case of using the proposed orientation-based (OS) error function in uncalibrated coordinates, the conditional probabilities γ_{ik} can be used as the scale factor (realize that the index j has disappeared from γ_{ijk} as for the estimation of each vanishing point the dimension corresponding to the support lines is marginalized).

   This way, a Levenberg-Marquardt iterative procedure is run in order to update the vanishing point $\mathbf{v}_k$. The initialization of this procedure could be random, although we have found better results if the initialization is carried out with the calibrated point-line (CPL) minimization, which is a linear approach and can be solved with least squares. As a linear optimization problem, the least squares solution for each $\mathbf{v}_k$ is given in closed form as the eigenvector with smallest eigenvalue of the eigenproblem (as used by Collins [29], Košecká and Zhang [65]):

$$\left(L^\top \Gamma^\top \Gamma L\right) \mathbf{v}_k = \lambda \mathbf{v}_k \tag{3.53}$$

   where L is a $N \times 3$ matrix, for which each row corresponds to the data sample $\mathbf{l}_i$ in row format, and Γ is a $N \times N$ diagonal matrix of weights, corresponding to the conditional probabilities γ_{ik} from $i = 1, \ldots, N$.

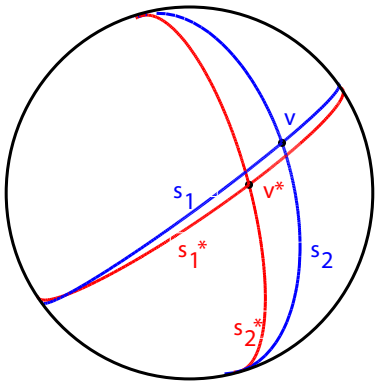


Figure 3.18: Given the new position of the vanishing point, $\mathbf{v}$, the lines $\mathbf{s}_1^*$ and $\mathbf{s}_2^*$ are re-estimated yielding $\mathbf{s}_1$ and $\mathbf{s}_2$ such that they actually pass through $\mathbf{v}$ and the angular distances, $\mathbf{s}_j^\top \mathbf{s}_j^*$, are minimized.

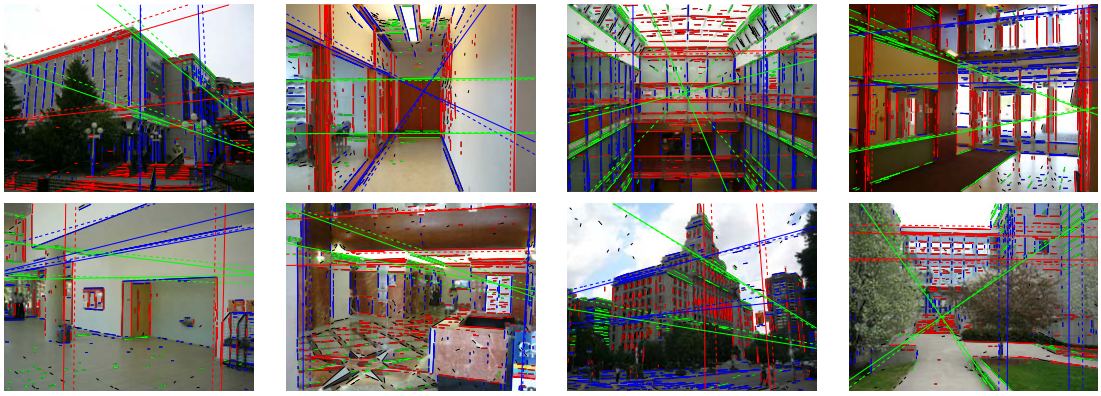
3. **M-step(2)** Within the image-plane EM algorithm, the parameterization of the support lines includes the position of the vanishing point as independent variable. Hence, each time the vanishing point is estimated, the offset of these lines is automatically updated, while keeping their orientation. However, for the projective version of the algorithm, the vanishing point has to be included as a restriction in the optimization problem rather than an added parameter. For this reason, after updating the vanishing point in the M-step(1) it is required to recompute the parameters of these lines so that they meet at the updated vanishing point. This concept is illustrated in figure 3.18. The problem is enounced, for each line $\mathbf{s}_{jk}$ as:

$$\hat{\mathbf{s}}_{jk} = \underset{\mathbf{s}_{jk}^*}{\operatorname{argmax}} \left(\mathbf{s}_{jk}^\top \mathbf{s}_{jk}^*\right)^2 \text{ subject to } \mathbf{s}_{jk}^\top \mathbf{v}_k = 0 \tag{3.54}$$

   The solution to this problem can be found using iterative methods for non-linear minimization with restrictions. For instance, a sequential quadratic programming (SQP) method has been used, implemented into the optimization MATLAB toolbox.

4. **E-step(2)** After the re-estimation of the support lines, the computed conditional probabilities have to be updated according to the new parameters of the support lines. Therefore, the values of γ_{ijk} are re-computed as in (3.52).

5. **M-step(3)** Finally, using the updated conditional probabilities and the likelihood function defined in (3.50), the parameters of the set of support lines have to be updated. The problem to be solved for each line $\mathbf{s}_{jk}$ is defined as



(a) (b) (c) (d)

Figure 3.19: Examples of the application of the proposed EM algorithm in the projective plane for multiple vanishing point estimation. The upper row show cases in which the initialized vanishing points (the intersection of the dashed lines) is quite correct and thus the EM algorithm just works as a refinement step. The bottom row of images show more difficult cases, in which the initialization of some vanishing points is significantly incorrect. For instance the blue vanishing point in (a), or the green one in (b). In these cases, the proposed strategy correct these errors and provides high accurate estimations of vanishing points.

$$\hat{\mathbf{s}}_{jk} = \underset{\mathbf{s}^*_{jk}}{\operatorname{argmax}} \log \mathcal{L}(\Theta|\mathcal{X}) \text{ subject to } \mathbf{s}_{jk}^\top \mathbf{v}_k = 0 \text{ and } ||\mathbf{s}_{jk} = 1|| \tag{3.55}$$

where, again, this problem must be solved using iterative methods for non-linear minimization with restrictions as in M-step (2).

3.5.2.4 Testing and discussion

Some of the properties of the projective-plane EM algorithm have been already evaluated in the tests corresponding to the image-plane EM (section 3.5.1.6), since they share several basic aspects: the usefulness of the orientation information, the effectiveness of the outlier distribution to absorb outliers, etc. Therefore, this section focalizes on other aspects, such as the importance of the support lines, how to practically initialize them as well as vanishing points, or the treatment of infinite vanishing points.

The proposed projective-plane EM algorithm requires an initialization of the parameters to be estimated: V vanishing points and each corresponding set of M support lines. For the vanishing points, the estimations provided by the MSAC-OS algorithm are used as initialization, since it was shown to provide the best results in section 3.4.2. The support lines must also be initialized, so that a line segment clustering

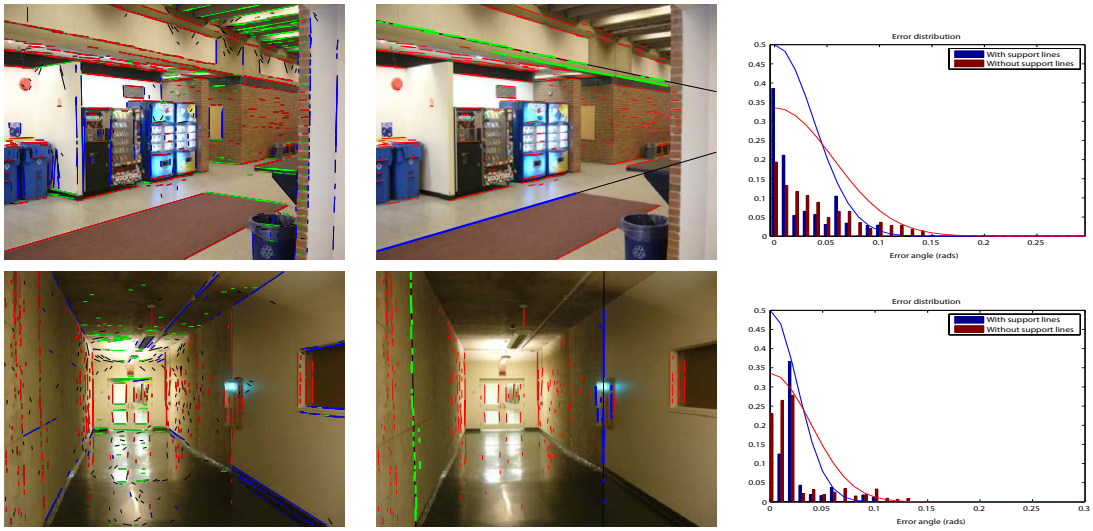
stage need to be conducted. For this purpose, the Hough transform can be used on the set of data samples $\mathcal{X} = \{\mathbf{r}_i, \mathbf{l}_i\}_{i=1}^{N}$, such that each point $\mathbf{r}_i$ is transformed into a sinusoid $\rho = x_i \sin\theta + y_i \cos\theta$. Then, the votes of data samples are accumulated on the quantized cells defined by the sinusoid in the accumulator. The accumulation is done such that the count of each one of these cells is increased by the length of the line segment. This way, the votes of longer line segments are more important than those of shorter ones. Typically it is enough to search for two separated maxima that correspond to the two most significant line segments clusters.

The images of the YUDB are used to test the performance of the proposed projective-plane EM algorithm. Some examples of the result are shown in figure 3.19. These examples show, in dashed lines, the initialization of the support lines according to the Hough transform based clustering approach, colored according to the vanishing point they meet at. In solid lines, the estimation of the lines after running the EM algorithm. Regarding the initialization, the upper row show examples for which the initialized support lines are close to actual significant lines of the image; the projective-plane EM algorithm applied on these cases result in a refinement of the initial parameters, finding the best fits for the support lines.

The second row addresses the ability of the EM algorithm to correct not so accurate initializations. As shown, some of the initialized support lines are incorrectly oriented mainly due to an incorrect vanishing point initialization. This lack of accuracy of the MSAC-OS method to provide good vanishing point initializations can be due to two main factors: (i) the set of detected features is not dense enough to provide a sufficient number of line segments for all vanishing points (such as the second row cases, (a) and (b)); or (ii) there is a large proportion of inliers that are not clustered in main lines, and that have higher orientation error than the samples actually clustered into main lines. The MSAC-OS considers all the inliers and thus its estimation can be affected by the presence of such false-inliers[14] (examples of the second row, (c) and (d)). The application of the projective-plane EM algorithm corrects these erroneous initializations and rectifies the position of the vanishing points and the corresponding support lines. The result, as shown in the examples of figure 3.19 is that the proposed method accurately determines the three main directions of the scene as well as the main support lines passing through them, including vanishing points inside the limits of the image, outside, and in the infinity (for instance, cases (b) and (c) of upper row show, respectively, in green and red, two parallel support lines meeting at the infinite).

The computation of the support lines during the optimization process is the reason

[14]In fact, some of these inliers are actually noisy line segments accidentally meeting the vanishing point under consideration, such as those in the trees, or the floor. Their presence heavily disturb the correct estimation of vanishing points as they show high error values (although in the limit of acceptance by the MSAC procedure).



(a) (b) (c)

Figure 3.20: Error distribution with and without using the support lines: (a) classification of line segments given by the MSAC-OS algorithm; (b) line segments corresponding to two support lines computed by the projective-plane EM algorithm; (c) comparison of the normalized histograms (weighted according to the length of the segments) of the orientation error.

for such good performance [15]. The support lines act as a subselection process that filters which line segments will contribute to the estimation of the vanishing point. This way, line segments are selected only if they are clustered around significant lines, discarding other line segments (that could also meet at the vanishing point) if they are isolated in the image. The advantage comes from the hypothesis that line segments that are clustered into lines have less error with respect to vanishing points than isolated or sparse line segments, which could be in fact outliers accidentally meeting at the vanishing point with higher error. The proposed method, by using this line clustering, discards these false inliers and the accuracy of the estimation is enhanced. This is precisely what makes the MSAC-OS to fail (and actually any estimation method that does not subselect inliers according to this criterion), in the cases shown in figure 3.19 second row, (c) and (d).

This property of the proposed projective-plane EM algorithm is exemplified in figure 3.20. Column (a) shows the classification of line segments obtained applying the MSAC-OS algorithm for two example images of the YUDB. In (b), an example pair of

[15] These lines can also be used for any subsequent computer vision algorithm. For instance, as it will be detailed in following chapters, plane rectification can take advantage of the presence of lines meeting at vanishing points.

support lines for one vanishing point are shown in solid black lines. The corresponding line segments associated to these support lines (whose conditional probability γ_{ijk} are higher than to any other component of the mixture model) are highlighted in thick green and blue lines, while the rest of line segments associated to the same vanishing point are shown in red. The graphs of column (c) show the normalized histogram (summing to one) of the orientation error weighted according to the length of the line segments. Also, a normal fit is provided for the two datasets: one corresponding to the complete set of inliers provided by MSAC-OS, and the other corresponding to the subset of line segments selected according to the support lines computed by the EM algorithm. As shown, the use of support lines provides narrower error distributions for the two cases. The upper row show a case in which the support lines actually correspond to two very long, clear lines in the scene, and thus the orientation error histogram is much narrower than the complete set of line segments, which contain a large number of the so-called false inliers. The bottom row shows an example in which line segments are all of similar length, and the support lines can not be fitted to a clear set of long line segments, but to an arbitrary set of aligned line segments. Nevertheless, the result is that there is also a gain in the reduction of error deviation.

Finally, we have compared the results obtained with the proposed projective-plane EM algorithm and the results obtained with the MSAC-OS algorithm, which does not subselect data samples to compute vanishing points for the whole set of images of the YUDB.

Provided that there are 102 images in the database, and a total number of 301 vanishing points, the MSAC-OS algorithm obtains correct detections (below 10^o with respect the ground truth vanishing points) for $284/301 = 94.35\%$ vanishing points. The average error achieved by this method is 3.5^o.

Considering these 284 correct initializations, the projective-plane EM algorithm refines the obtained vanishing points and reduces the error dow to around 1^o for $247/284 = 86.97\%$ cases. For the cases in which the MSAC-OS obtains inaccurate initializations, mainly due to the abovementioned false-inliers problem, the EM algorithm improves the estimations more than 5^o in average for $28/284 = 9.86\%$ cases.

Nevertheless, the projective-plane EM algorithm also commits some errors, and for $9/284 = 3.17\%$ cases, the estimation of the vanishing point is worst than the one given by the MSAC-OS algorithm. These are the cases in which there are not enough line segments supporting the vanishing points, or they are not clustered into dominant lines but actually distributed along the whole image.

3.6 Conclusions

In this chapter, different vanishing point detection methods have been proposed. Robust approaches based on RANSAC have been analyzed as well as its use with different error functions (including the proposed orientation-based function). Besides, the EM algorithm has been described for the simultaneous computation of vanishing points and support lines, for two different workspaces: the image-plane and the projective-plane.

As a summary, several conclusions can be extracted from the discussions presented in this chapter. The evaluation of the state of the art regarding vanishing point estimation pointed out that an effective way to proceed consists on using a two-steps approach. On the one hand, robust methods, like RANSAC can be used as initialization in presence of outliers and still give correct detections of vanishing points without any prior knowledge of the scene. In particular, this strategy requires the definition of an error function between the vanishing points and the image features (e.g. line segments or gradient-pixels). In this aspect, a novel error function, based on an orientation difference, has been proposed, which has been shown to produce as accurate results as some of the most popular of the literature while reducing the number of iterations of the required non-linear procedures. In the related discussion, it has been also shown that, MSAC, the well-known variant of RANSAC used in this context increase the robustness of the detections of vanishing points.

In general, for static images of structured scenarios, the proposed method based on MSAC and the orientation with scale (OS) error function can be combined to obtain robust estimation of vanishing points with very high detection rates.

In case of having sequences of images, it has been shown that the RANSAC algorithm can be slightly modified to dramatically reduce the required number of iterations to converge. This can be done by means of using the estimation of vanishing points in previous time instants as a prediction that guides the selection of MSS, which otherwise would be done randomly. Therefore, this approach can be used for tracking vanishing points through sequences of images.

The EM algorithm has been included as an iterative refinement step that can start with the results provided by the initialization strategy (such as the one based on MSAC-OS) and increase the accuracy of the estimations. For this purpose it is required to initialize the support lines that meet at each vanishing point, for instance by using the Hough transform. Typically it is enough to use two support lines per vanishing point. In case vanishing points are inside the limits of the image, or at least not too far away (such as for images of corridors or roads), it is better to use the image-plane EM, as it is robust and accurate in these cases and also is a linear approach that achieves fast convergence. If infinite vanishing points have to be computed, it is strictly required to use the projective-plane EM algorithm.

Part of the excellent performance of the proposed EM approaches is due to the

use of the mentioned support lines. These lines help to subselect image features that are clustered in main groups and that are hypothesized to be less noisy than the rest of image features. For this purpose, several considerations have arisen during the design of these methods. For instance, the proposed image-plane EM incorporates the orientation information of image features in the linear model proposed by Minagawa et al. [76], as well as the treatment of outliers by means of an additional outlier distribution in the mixture model. The projective-plane EM algorithm represents an innovative contribution that has been not treated in the literature for the simultaneous estimation of multiple vanishing points and their support lines in the projective plane, i.e. handling both finite and infinite vanishing points.

As far as tracking is concerned, the EM algorithm can correct the predictions provided by methods like MSAC and absorb their error while the motion of the camera is not very high and the position of vanishing points does not vary significantly between consecutive images of the sequence. These concepts will be exemplified in next sections, in which the proposed vanishing point estimation methods are applied on a specific scenario (the road scenario), and for an specific task (plane rectification).

Chapter 4

Road scenario

4.1 Introduction

This chapter introduces a particular scenario in which the methods described in chapter 3 for vanishing point estimation are applied. It will be called the "road scenario", since it refers to a forward looking camera installed inside a moving vehicle observing the road ahead. Hence, the images captured by the camera typically show a pronounced perspective effect (as those shown in figure 4.1), and contain elements of interest such as lane markings and other vehicles. This is the scenario addressed by video-based driver assistance systems (DAS), which aim to provide applications such as lane departure warning (LDW), lane invasion, collision prevention, pedestrian detection, sign recognition, etc.

Significant benefits come out from the computation of vanishing points in this environment. For instance, chapter 5 describes the generation of fronto-parallel views of the road, which are helpful for different tasks (lane markings modeling, vehicle detection, etc.). Within this view, the perspective distortion of the road is removed, observation models are simplified and the dynamics of the imaged vehicles are measured without distortion. This transform is greatly referenced along this chapter, as it is used by the algorithms that are described in the chapter (an example image resulting from this transformation is shown in figure 4.2). The reason for its usage is twofold. On the one hand, the observations are simplified, e.g. lane markings appear parallel and with similar width, and thus easier to estimate. On the other hand, the image processing algorithms (for instance, the SSWMS line segment detector) perform much better in this transformed domain, since the information is better distributed in the image, without perspective distortion.

However, the vanishing point estimation methods described in chapter 3, strongly depend on the quality of the detected image features in terms of accuracy, amount

Figure 4.1: Example images of the road scenario, including reflections, vehicles, illumination changes, etc.

of data, proportion of outliers, etc. Within the road scenario, reliable computation of vanishing points from line segments or gradient-pixels is an arduous task. This environment, as it will be shown in next sections, poses a number of challenges for vision systems. On the one hand, it is highly dynamic, and thus can dramatically change its properties (illumination, color of the road, number of objects and their positions, presence of shadows, weather conditions, moving wipers, etc.) in very short time periods since it is an outdoor scenario with fast camera motion. On the other hand, the information that can be considered as reliable, such as that corresponding to the lane markings of the road, is very limited and, typically, on account of the scene dynamism, very difficult to obtain. Additionally, the motion of the scene entails more complexity, as it is composed of the motion induced by the vehicle itself with respect to static elements, and of other vehicles, typically moving at different speeds.

Therefore, the vanishing point estimation strategies described in chapter 3 can not be exploited with all their advantages unless additional processing is applied on the image observations to overcome the mentioned difficulties. In particular, we can take advantage of the presence of lane markings in the image. The main vanishing point of the scene is located in the point of the image in which lane markings seem to converge. Therefore, an obvious approach is to first find these lane markings in the image and then apply the line segment detection only on the selected regions.

Besides, within this scenario, the camera moves approximately in the direction of the mentioned main vanishing point of the scene, so that, typically, the position of this vanishing point between consecutive images does not significantly vary. This temporal coherence can also be exploited in order to improve the vanishing point estimations.

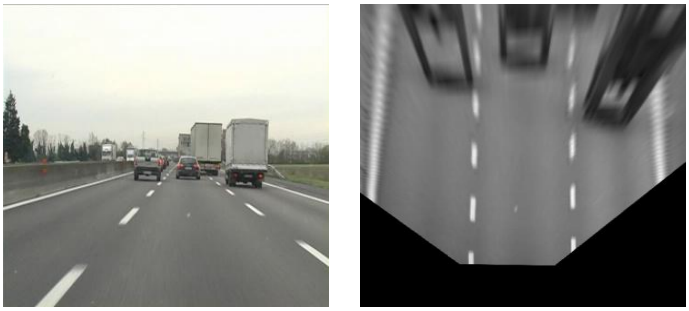


Figure 4.2: Example bird's-eye view of the road.

4.1.1 Road modeling - Overview

The work relative to the road scenario has been carried out in the framework of a video-based DAS, whose aim is to generate a road model from the sequence of images captured by the camera. The road model is understood as the set of parameters that characterize the properties of interest of the road. For instance, the number of visible lanes, their width or the geometry of the lane markings. Figure 4.3 shows the block diagram of the proposed system. As shown, it is composed by several blocks in a recursive fashion. Particularly, the estimation of the main vanishing point of the scene plays the major role in this scheme. The road model is built based on the knowledge of this vanishing point, which is used to determine the parameterization of the road plane[1], and to infer the geometry of the lane markings. As abovementioned, the application of the methods described in chapter 3 has to be preceded by some processing stages. The three main stages are: plane rectification, feature extraction, and lane modeling.

Plane rectification Input images are first transformed to obtain a new image in which the road is seen without perspective distortion. The transform is carried out using the information of the previous estimation of the vanishing point, $\tilde{\mathbf{v}}_{t-1}$. The used method is the one described as case 2.2 in chapter 5 table 5.1. The rest of the modules of the system work on this transformed domain to take advantage of its mentioned properties.

Feature extraction Lane markings are the elements of the road images which are more reliable to compute the vanishing point. Therefore, the system first detects the pixels of the transformed domain that more likely belong to lane markings by means of a Bayesian classifier, which is also used to detect other elements such

[1] As many other authors in the field [14, 58, 74, 104, 107], the road is assumed to be locally planar, since, in general, the incurred error is very small for the analysis range of an image (typically between 15 to 20 meters)

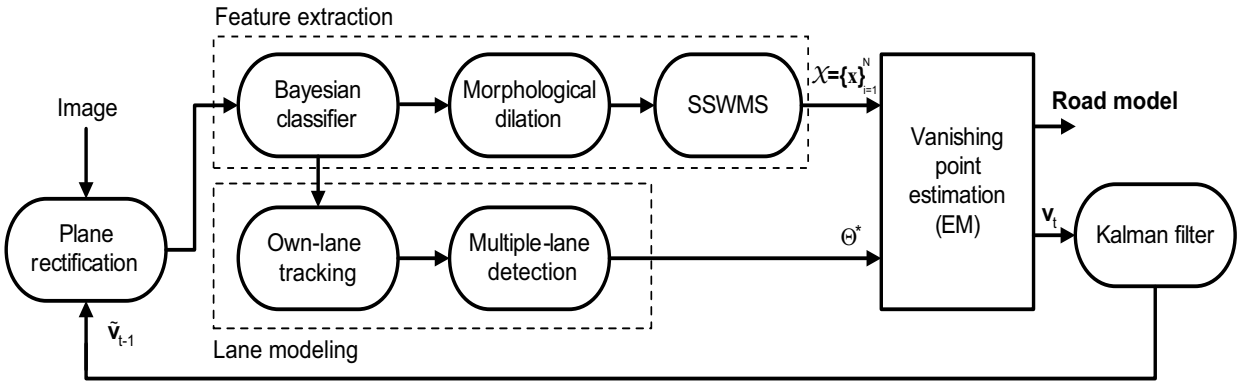


Figure 4.3: Block diagram of the proposed method for multiple-lane road model generation, and recursive vanishing point estimation and plane rectification, which is discussed in chapter 5 section 5.4.

as the pavement of the road or vehicles. A morphological dilation is applied on the lane marking pixels to define the regions of the image on which the SSWMS algorithm shall be applied. The result is a set of line segments $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{N}$ that feeds the vanishing point estimation algorithm.

Lane modeling The temporal coherence is properly handled by means of the modules that model the own lane and the adjacent lanes of the road. Within these modules, the observations coming from the Bayesian classifier are given continuity as the set of main or support lines modeling the lanes, Θ^*.

The projective-plane EM algorithm, as described in chapter 3 is used to estimate the vanishing point. As a result, the road model is built and can be used as the output of the DAS to provide services like lane departure warning, lane invasion, etc. Besides, the estimated vanishing point is time filtered with a Kalman filter in order to smooth the variations of the positions of the vanishing point and to generate more stable plane rectifications.

4.2 Related work

In this section we will focus on image processing strategies that are related to the detection of lane markings and the generation of road models.

Typically, road modeling strategies are divided into two main processing stages: feature extraction, which is the module that processes the images to generate observations from them; and model fitting, which contains both the definition of a road model

and the process that searches for the values of the parameters of the road model that best fit to the observed features.

4.2.1 Feature extraction

Most works found in the literature detect the lane markings which delimit the lane boundaries of the own lane as the main source of information of this scenario. Different approaches have been presented for this purpose in the recent literature, achieving great results ranging from fast alternatives to robust ones. Some authors assume the intensity values of the lane markings to be brighter than the rest of the image, and thus apply intensity thresholding by histogram analysis as González and Ozguner [43], Jeong and Nedevschi [51], Jiang et al. [53]. Similarly, in terms of simplicity, some alternatives apply edge detection and thresholding like Kluge and Lakshmanan [62], Lai and Yung [66], Li et al. [67], Schreiber et al. [100], using directly the Sobel operator (Shu and Tan [104]), the Canny method (Wang et al. [122]), or steerable filters like McCall and Trivedi [74], Nieto and Salgado [82]. More complex alternatives introduce morphological operations, as done by Tsai et al. [116], support vector machine (Kim [58]), or Markov Random Fields (Wojek and Schiele [123]).

Finally, there is a number of works that model lane markings as bright stripes in either the original image or the rectified plane, which are detected with intensity-bump analysis carried out with one-dimensional filters applied on each image row. The work by Bertozzi and Broggi [14] outstands as one of the first approaches that used this type of filtering in the rectified domain. Nevertheless, others followed this approach slightly varying the filtering step as Borkar et al. [17], Nieto et al. [87], Southall and Taylor [107]. There is also an interesting approach based on the generation of a time-fused image for a set of consecutive images. The obtained effect by using the temporal mean [17] or maximum [87] is that discontinuous lane markings become continuous and, although some distortion is induced, detections are significantly enhanced. Besides, some methods that extract basic features like intensity levels or edge information, use them in combination with the Hough transform to obtain lines or line segments corresponding to lane markings (Wang et al. [122]).

Nevertheless, no special attention is devoted to feature extraction, which is typically solved with simple approaches. The complexity of the proposals is typically more focused on the road model fitting, the data fusion with other sensors, etc. Some significant exceptions are those by Kim [58] that compared intensity-bump methods with some learning-based methods, determining that the best approach is based on Support Vector Machines. Veit et al. [118] compared the performance of different approaches based on intensity-bumps, global and local thresholding using ground truth images.

4.2.2 Model fitting

After the feature extraction stage, the obtained information is typically used to compute the values of the parameters that define the road model. The mathematical models used in the literature to fit the image observations vary from generic second order curves, such as parabolas (Kluge and Lakshmanan [62], McCall and Trivedi [74]), circles, to constrained cubic curves that approximate clothoids (Corridori and Zanin [31]), or even more complex spline shapes (Wang et al. [122]). Some works also consider parabolic curves in the road plane domain and the family of curves resulting from the camera projection (Kluge [61]). In general some assumption about parallelism or quasi-constant width of the lanes, in the rectified domain, is applied (Bertozzi and Broggi [14], Tsai et al. [116]).

The methods used to fit these models to the detected lane marking pixels vary from accumulation with the Hough transform, used not only for line fitting [17, 122], but also for higher order curve fitting (Li et al. [67]), to robust methods like Least Median Squares [61], or RANSAC [58]. The temporal coherence of the observations is also considered, provided that the images are captured by cameras with at least 10 fps. This way, stochastical filtering guides the fitting process through time propagation of probabilistic models. The Kalman filter has been commonly applied by authors such as Borkar et al. [17], Corridori and Zanin [31], McCall and Trivedi [74], Nedevschi et al. [79], since it is the optimal solution provided that the process and measurement noise is Gaussian. Nevertheless more complex approaches for non-linear, non-Gaussian models have been proposed, such as particle filters [58, 107].

Regarding the type of acquisition systems used, the most recent trends in this field have focused on 3D environment modeling using stereovision, due to its ability to recover depth information from the analysis of two synchronized video inputs. For instance, several works address lane markings models with subpixel accuracy using calibration information (Danescu et al. [35], Nedevschi et al. [79, 80]). Others use image alignment between the images of the stereo pair to detect volumetric objects on the road plane (Broggi et al. [20], Chang et al. [27]), or to enhance lane markings detection (Simond and Rives [106]).

While showing promising results in obtaining depth information, stereoscopic vision has a number of drawbacks: multi-view systems are typically not considered for real-time applications due to their built-in complexity, mainly related to the calibration process, the necessity of a synchronized acquisition system and the difficulties encountered in finding reliable correspondences between images (Stein et al. [108]). On the other hand, mono-camera systems are more cost-effective and hence more widely used in real applications (Zhou and Li [126]). Different mono-camera approaches have been proposed in the literature to address the road modeling (McCall and Trivedi [74]), including accurate lane markings models (Corridori and Zanin [31], Wang et al. [122]),

and vehicle detection and tracking (Chen et al. [28], Goecke et al. [42], Hoffmann et al. [49], ten Kate et al. [113]).

Some of these solutions provide coarse or incomplete results due to the intrinsic limitations of the mono-camera analysis: projective geometry is typically handled using appearance-based methods that may be faster than stereo [53] without evaluating the loss in robustness. However, some researchers compensate for this limitation by making prior assumptions about the environment, for example, by defining a constant relative pose of the camera with respect to the road (which is assumed to be always flat) (McCall and Trivedi [74], Shu and Tan [104]). However, such approaches reduce the system's capability to adapt to more realistic situations.

4.3 Road feature extraction

This section describes the proposed feature extraction procedure, which uses a powerful tool to obtain the required features (mainly the information about lane markings) in a reliable way. The proposed strategy is based on the Bayesian decision theory. The algorithm defines a parametric multiple-class likelihood model of the road, from which pixels are classified into different classes, with an associated probability of error.

The probabilistic framework handles all the available information in a simple and robust way, by defining the prior probabilities and the likelihood models, and by appropriately choosing the features that best characterize the classes that are to be identified. Therefore, it avoids to define and compute a large amount of deterministic cases or situations regarding illumination conditions, the presence of vehicles, motion, etc. Besides, it copes with the highly dynamic nature of the scenario since the likelihood models are adapted to the observations.

Three types of elements of interest are considered within any road-plane image:

- **Pavement**: light gray regions of the road.
- **Lane markings**: bright stripes painted on the road.
- **Objects**: dark elements, such as the lower parts of vehicles, their wheels, shadows, etc.

An extended version of the models is defined by adding an additional "unknown" class. In general, all pixels could be classified as belonging to one of the abovementioned classes. Nevertheless, if the properties (which will be described in next sections) of a pixel do not fully match any of the defined classes, the system is designed to set that pixel as belonging to an unknown element. This situation is quite typical in such outdoor uncontrolled scenario, since it may happen that un-modeled elements appear in the scene. Therefore, this procedure reduces the classification error, and provides some degree of uncertainty in the segmentation of the image.

4.3.1 Bayesian framework

Let $\mathcal{S} = \{P, L, O, U\}$ be the set of classes that represent, respectively, pavement, lane markings, objects and unidentified objects. The target of the classifier is to assign one of these classes to each pixel of the image.

Let X_i represent the event for which a pixel, indexed with its spatial coordinates inside the image (x, y), with an associated observation vector denoted as $\mathbf{z}_{xy}$, is classified as belonging to the class $i \in \mathcal{S}$. Using the Bayesian decision theory, this classification is carried out by selecting the class that maximizes the posterior conditional probability $P(X_i|\mathbf{z}_{xy})$, which is decomposed by the Bayes' rule as:

$$P(X_i|\mathbf{z}_{xy}) = \frac{p(\mathbf{z}_{xy}|X_i)P(X_i)}{P(\mathbf{z}_{xy})} \tag{4.1}$$

where $p(\mathbf{z}_{xy}|X_i)$ is the likelihood function, which denotes the probability that a pixel, according to its associated measurements, belongs to class i; $P(X_i)$ is the prior probability of each class and $P(\mathbf{z}_{xy})$ is the evidence, a scale factor that ensures that the posteriors sum to unity, computed as $P(\mathbf{z}_{xy}) = \sum_{i \in \mathcal{S}} p(\mathbf{z}_{xy}|X_i)P(X_i)$.

The result, for each pixel, is a set of posterior probabilities $\{P_{xy}(X_i|\mathbf{z}_{xy})\}_{i \in \mathcal{S}}$, which denote the probability that a pixel belongs to each defined class. Accordingly, each pixel of the image is classified as the class with the maximum posterior probability. The likelihood and the prior probabilities are computed as described in the following subsections.

4.3.2 Likelihood models

In this section the likelihood models are described as parametric functions, according to the expected properties of the considered image features with respect to the defined classes. Additionally, the estimation of their parameters is obtained through an optimization process using the Expectation-Maximization (EM) algorithm based on the observations of the image.

The appearance of the defined classes (that can be used to determine the type of likelihood and prior models) can be described as follows: the pavement is usually a homogeneous area in the transformed image (the bird's-eye view of the road), whose pixels have a similar intensity level with low variations among pixels; lane markings are represented as near-vertical bright stripes, usually surrounded by pavement pixels; and objects typically can be characterized by dark regions with intensity levels lower than the pavement.

With this information, it is possible to design pixel-level features that help to differentiate between classes. Two features have been used for this purpose: the intensity

or grayscale level, I_{xy}, and the response to the lane marking detector, L_{xy}. The combination of these features ensures a clear class differentiation, especially accurate for the lane markings class, thus allowing to reduce misclassifications.

The likelihood function of class i is defined as:

$$p(\mathbf{z}_{xy}|X_i) = p(I_{xy}|X_i)p(L_{xy}|X_i) \tag{4.2}$$

where the likelihood functions for each image feature are assumed to be conditionally independent.

4.3.2.1 Intensity feature

The likelihood functions for I_{xy}, namely $p(I_{xy}|X_P)$, $p(I_{xy}|X_L)$ and $p(I_{xy}|X_O)$ are all defined as normal distributions:

$$p(I_{xy}|X_P) \propto \exp\left(-\frac{1}{2\sigma_{I,P}^2}(I_{xy} - \mu_{I,P})^2\right) \tag{4.3}$$

and analogous expression for the other classes; parameterized by their respective mean and standard deviation: $\{\mu_{I,L}, \sigma_{I,L}\}$, and $\{\mu_{I,O}, \sigma_{I,O}\}$. Note that there is an implicit and necessary, condition that must be satisfied: $\mu_{I,O} < \mu_{I,P} < \mu_{I,L}$, since the dark objects are always darker, as well as lane markings are always clearer than the pavement. The model for the unknown class is defined as well as as a normal distribution with parameters $\{\mu_{I,U}, \sigma_{I,U}\}$ with large fixed variance, such that it is similar to a uniform distribution[2].

4.3.2.2 Lane marking detector

A new intensity-bump lane marking detector is proposed in this section, that is used to filter the image to generate an image where highlighted pixels are only those corresponding to the lane markings[3].

The detection is done by applying a one-dimensional filter to each row of the image, assuming that the appearance of the lane markings in this one-dimensional domain is given by pulses of high intensity values surrounded by darker regions. Therefore, the analysis is done by independently filtering each image row, indexed by y, denoted as $\{I_{xy}\}_{x=1}^{W}$, resulting in a new filtered data array $\{L_{xy}\}_{x=1}^{W}$, defined as:

[2]This question has been addressed also for the control of outliers in the mixture model of the proposed EM algorithms in chapter 3, section 3.5.1.5

[3]It is included in a recent publication [9]. Previous versions of this detector were also included in other publications [82, 87]

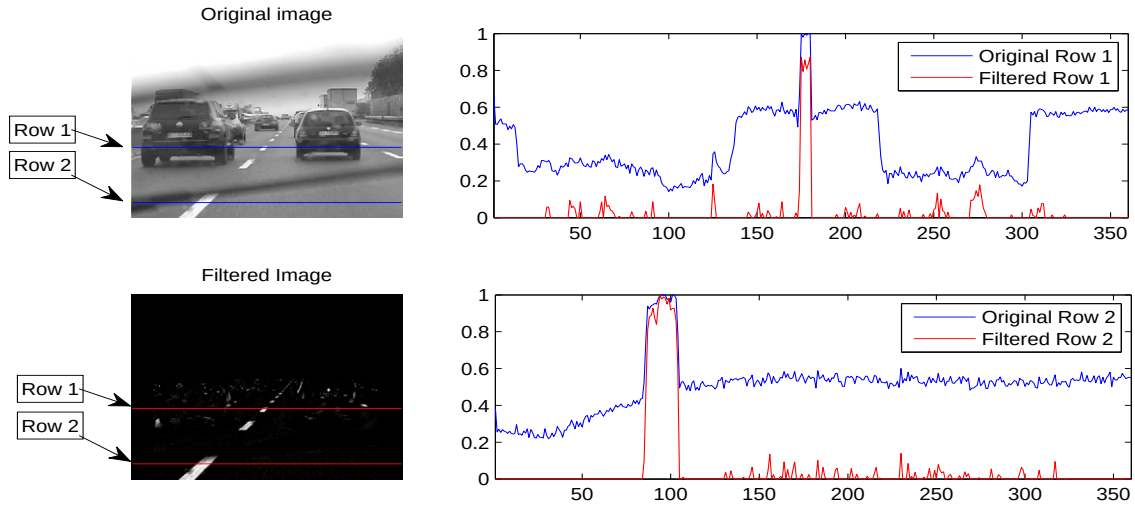
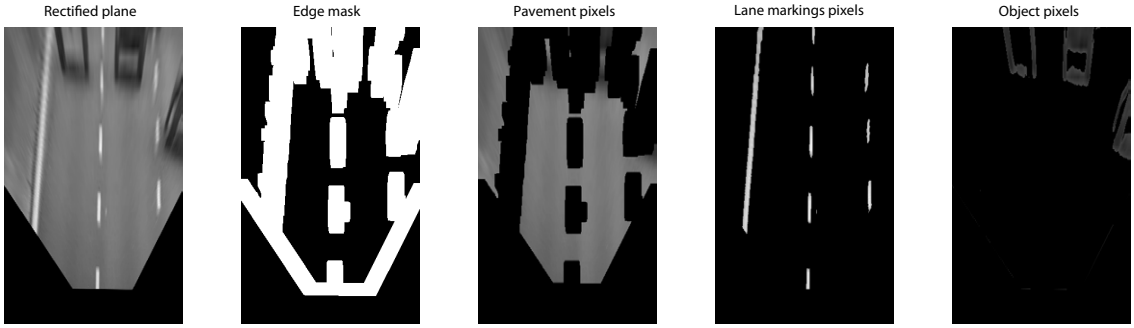


Figure 4.4: Lane marking detection for an example original image. For clarity, the response to the filter has been normalized between 0 and 1.

$$L_{xy} = 2I_{xy} - (I_{x-\tau,y} + I_{x+\tau,y}) - |I_{x-\tau,y} - I_{x+\tau,y}| \tag{4.4}$$

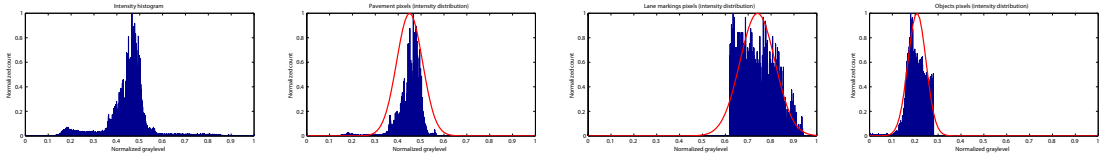
where τ is the width parameter that governs the filtering process[4]. This filter produces high responses for I_{xy} values that are higher than those of their left and right neighbors at distance τ, given by the substraction $2I_{xy} - (I_{x-\tau,y} + I_{x+\tau,y})$. The last term in (4.4) penalizes cases in which the difference of intensity between the left and right neighbors is significant, so that a higher response is given to positions with similar intensity values of the left and right neighbors. This last term makes the filter less prone to errors than other lane marking detectors presented in the literature [14, 87]. An example of the application of this filter to an original image is given in figure 4.4. In the example, two different rows are analyzed. The intensities of the original image in each row are shown in blue, while the result of the filter is shown in red. The lower row (row 2) is a simple example of excellent performance given a clear lane marking, even with the presence of obstructing elements, such as the wiper of the vehicle. The case of the upper row (row 1) is more complicated, as there are several abrupt changes in the intensity profile, due to the presence of vehicles. Nevertheless, as it is shown in the response profile, our method accurately detects the lane marking of interest and dismisses the clutter.

[4]The value of τ is selected accordingly to the expected width of the lane markings in the transformed domain. In case the filter is applied to the original images, as done in some examples of this subsection, τ is dependent on y, such that at the image row that contains the main vanishing point, $\tau = 0$ and at the bottom row of the image $\tau = \tau_{max}$.



(a) (b) (c) (d) (e)

Figure 4.5: Initialization of the parameters of the likelihood function for the intensity feature.



(a) (b) (c) (d)

Figure 4.6: Histograms of the different sets of pixels for the computation of the likelihood parameters for I_{xy}.

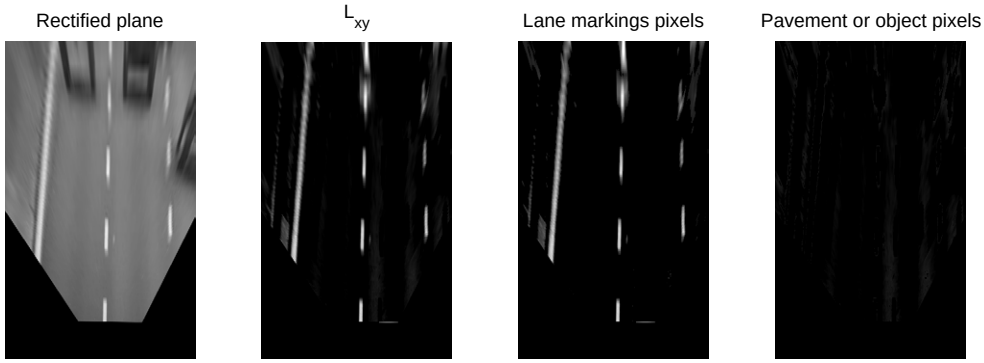
Regarding the likelihood functions associated to the proposed detector, lane markings are expected to provide high response values to the filter and low response values for the other classes. This way, the likelihood functions for L_{xy}, namely $p(L_{xy}|X_P)$, $p(L_{xy}|X_L)$, $p(L_{xy}|X_O)$ and $p(L_{xy}|X_U)$ are defined as normal distributions. The parameters of the distributions are $\{\mu_{L,P}, \sigma_{L,P}\}$, $\{\mu_{L,L}, \sigma_{L,L}\}$, $\{\mu_{L,O}, \sigma_{L,O}\}$ and $\{\mu_{L,U}, \sigma_{L,U}\}$. Although the response of the filter for pavement and objects is actually unpredictable, it is much more convenient to model them with normal distributions with high variance values. The unknown class is also modeled with a wide normal distribution, analogously to the unknown class for the intensity level likelihood model.

For this feature, the conditions $\mu_{L,O} < \mu_{L,L}$ and $\mu_{L,P} < \mu_{L,L}$ must hold, which mean that lane markings are always brighter in L_{xy} than pavement and dark objects.

4.3.2.3 Parameters estimation

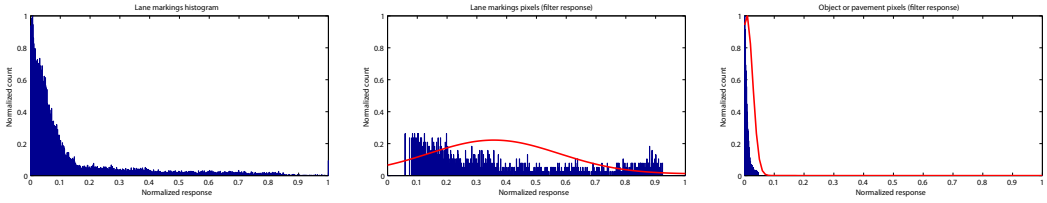
The parameters of the abovementioned functions are computed for each image of the sequence. Hence, the system dynamically adapts the Bayesian model in a sequential manner.

The EM algorithm for a mixture of Gaussians is used to estimate the parameters



(a) (b) (c) (d)

Figure 4.7: Initialization of the parameters of the likelihood function for the L_{xy}.



(a) (b) (c)

Figure 4.8: Histograms of the different sets of pixels for the computation of the likelihood parameters for L_{xy}.

that govern the likelihood functions for the defined classes since we defined all of them as normal distributions. The EM algorithm converges to the optimal solution if it receives a good initialization or start point. For that reason, we provide initial values of the parameters to be estimated (namely $\{\mu_{I,P}, \sigma_{I,P}, \mu_{I,L}, \sigma_{I,L}, \mu_{I,O}, \sigma_{I,O}, \mu_{I,U}, \sigma_{I,U}\}$ and $\{\mu_{L,P}, \sigma_{L,P}, \mu_{L,L}, \sigma_{L,L}, \mu_{L,O}, \sigma_{L,O}, \mu_{L,U}, \sigma_{L,U}\}$) in the following way: we can provide some coarse estimations for these parameters through the application of a preliminary analysis of the histogram of the image (an example image is given in figure 4.5 (a) and its associated histogram is shown in figure 4.6 (a)). The proposed approach extracts three groups of pixels from the image, one for each class. First, the putative pixels of the pavement class are obtained by dismissing pixels with high gradient value. For that purpose we generate a mask, as shown in figure 4.5 (b), that is used to remove the pixels with high gradient and their neighborhoods[5]. The value of the parameters for $p(I_{xy}|X_P)$ are then obtained as the sample mean and the sample standard deviation of the pixels of the resulting group, which is shown in figure 4.5 (c).

Lane markings and dark object classes are then extracted by thresholding the road-

[5]We have done this by applying first a Sobel gradient detection, an appropriate thresholding and a morphological dilation.

plane image at $\mu_{I,P} \pm 3\sigma_{I,P}$, respectively. These thresholds were chosen to satisfy the hypotheses $\mu_{I,O} < \mu_{I,P} < \mu_{I,L}$. In particular, the selection of $\mu_{I,P} \pm 3\sigma_{I,P}$ dictates that only those pixels falling outside the $99,999\%$ of the probability of belonging to the pavement are considered for modeling the two other classes.

The images in figure 4.5 (d) and (e) show the abovementioned criteria, and the corresponding sets of pixels that are likely belonging to the lane markings, and the objects class. The corresponding histograms for the images in figure 4.5 (c), (d) and (e) are shown in figure 4.6 (b), (c) and (d), respectively, with the associated normal fit that depicts the mean and standard deviation value that describe the histograms and that are used as the initialization of the mentioned parameters.

Regarding the lane markings feature a similar approach is followed, which is illustrated in figure 4.7. The image L_{xy} is computed (figure 4.7 (b)), and its histogram, shown in figure 4.8 (a) is assumed to be a mixture of two Gaussians: one is centered near zero (corresponding to the low values of L_{xy} corresponding to pavement and object pixels), and the other, with much larger variance, models the tail of the histogram (which depicts that the response of lane markings pixels ranges from moderate to very high values of L_{xy}). Since this histogram can be approximated by an exponential distribution, we propose to use the mean value of the distribution as the threshold that separates the two abovementioned modes. Two images are obtained through the thresholding stage, as shown in the examples of figure 4.7 (c) and (d). Then, the mean and standard deviation for the corresponding sets of pixels can be computed, obtaining $\{\mu_{L,L}, \sigma_{L,L}, \mu_{L,P} = \mu_{L,O}, \sigma_{L,P} = \sigma_{L,O}\}$. Note that for this feature the pavement and object class can not be distinguished, and thus they share the parameters of the likelihood function.

Within the EM algorithm, the likelihood functions according to the two defined features (intensity and response to the lane markings detector) are modeled as a mixture model:

$$p(I_{xy}|\{X_i\}_{i\in\mathcal{S}}) = \sum_{i\in\mathcal{S}} \omega_{i,I} p(I_{xy}|X_i) \tag{4.5}$$

$$p(L_{xy}|\{X_i\}_{i\in\mathcal{S}}) = \sum_{i\in\mathcal{S}} \omega_{i,L} p(L_{xy}|X_i) \tag{4.6}$$

where $\omega_{i,I}$ and $\omega_{i,L}$ are the weights of the corresponding mixture components. These coefficients represent the proportion of elements of the set (in this case the pixels of the image) that belong to each class. The EM algorithm has been designed to consider as initialization for these coefficients the actual proportion obtained in the classification of the previous time instant and estimates the new values for the current instant. The coherence between the values of these parameters is illustrated in figure 4.9, which shows the resulting intensity mean values computed along time for an example video

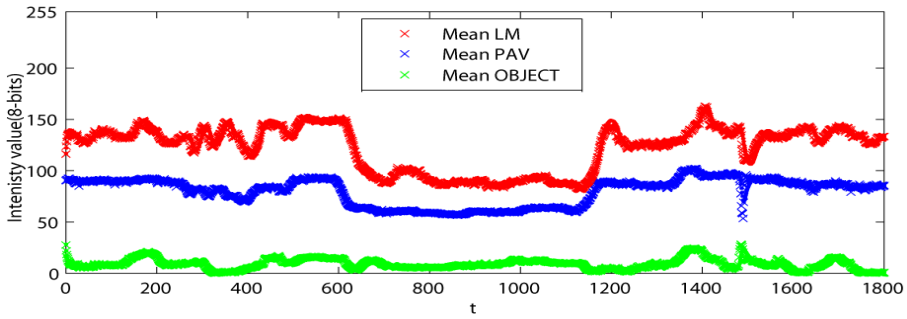


Figure 4.9: Variation of mean values for the classes lane markings (LM), pavement (PAV) and objects (OBJECT) along time in an example sequence.

sequence. Each colored curve corresponds to the variation of the mean of each class, lane markings, pavement and object.

The EM algorithm iterates until the whole set of parameters, including the mean and standard deviation of each normal distribution and the mixture component weights are computed[6]. As an exception, the unknown class is kept fixed, and not updated within the EM framework to ensure that a quasi-uniform distribution absorbs the putative outliers (analogously to the procedure carried out in chapter 3 with the estimation of vanishing points through the use of the different proposed EM algorithms and the additional outlier distribution).

Finally, the system comprises a control mechanism that is able to detect situations involving sudden visibility or illumination changes, such as when entering or exiting tunnels. In these situations, the system switches to a transitory state, in which the variables are not update according to observations in order to prevent the system from being corrupted. Meanwhile, the images are checked, and the transitory state finishes when the situation is stabilized. Therefore, this control scheme increases the overall robustness of the system, and avoids misleading the EM algorithm with wrong initializations.

To illustrate the effect of the application of the EM algorithm on the estimation of the parameters, let us consider the example depicted in figure 4.10. Within this example the EM is shown to correct an erroneous initialization of the model parameters as well as appropriately handle the weighting factors of the mixture.

Figure 4.10 illustrates the parameter estimation process for an example image, regarding the intensity level feature. The initialization is performed as described in previous paragraphs, which results in a set of parameters for the normal distributions,

[6]The E-step and M-step for a mixture of Gaussian are well known problems in many computer vision applications. Their expressions can be found in basic references as [15].

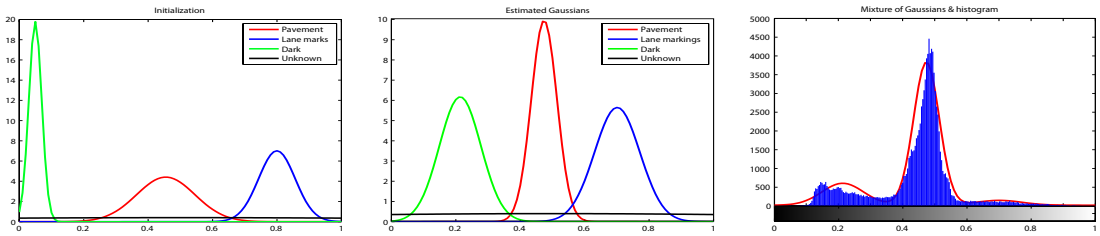


Figure 4.10: Initialization (a), and final estimation (b) of the mixture of Gaussians, which is depicted in (c) multiplying each Gaussian with its corresponding weight.

	Object	Pavement	Lane markings	Unknown
Initialization	25(%)	25(%)	25(%)	25(%)
Final estimation	19.37(%)	73.57(%)	6.02(%)	1.03(%)

Table 4.1: Estimation of the coefficients of the mixture model. The initialization is uniform, i.e. 25(%) for each class.

which are depicted in figure 4.10 (a). Note that, in this illustration, the scale of the distributions is the same, so that each distribution sums to 1 in the defined range (from 0 to 1). In black we show the distribution that corresponds to the "unknown" class. It is a normal distribution with a high standard deviation, so that it looks like and behaves as a uniform. This is an advantage of our approach, since at regions with high uncertainty, such as those corresponding to the intersection between classes, the unknown class has higher likelihood than the other classes.

The EM algorithm is used to refine the initialization and also to provide the values of the coefficients of the mixture. In figure 4.10 (b) we show the result of the EM algorithm for a mixture of 4 Gaussians, initialized as in figure 4.10 (a). After some iterations, the algorithm converges to the maximum likelihood value of the parameters. Note that the mean and standard deviation of the Gaussians have been modified to correct the error committed by the initialization stage (specially the green one, corresponding to the "object" class). The parameters of the distribution of the "unknown" class are not updated, so that we can always handle uncertainty as abovementioned. Besides, the EM computes the value of the coefficients of the mixture. For this simple example, we have initialized the value of the coefficients of the mixture uniformly, without using the information of the previous estimations. This way we show that the EM, even in the absence of prior information, can also modify significantly these values in order to adapt correctly the contribution of each component of the mixture to the data. The results are shown in table 4.1.

The resulting mixture of Gaussians (weighted according to the values obtained through the optimization process) is shown in figure 4.10 (c) together with the his-

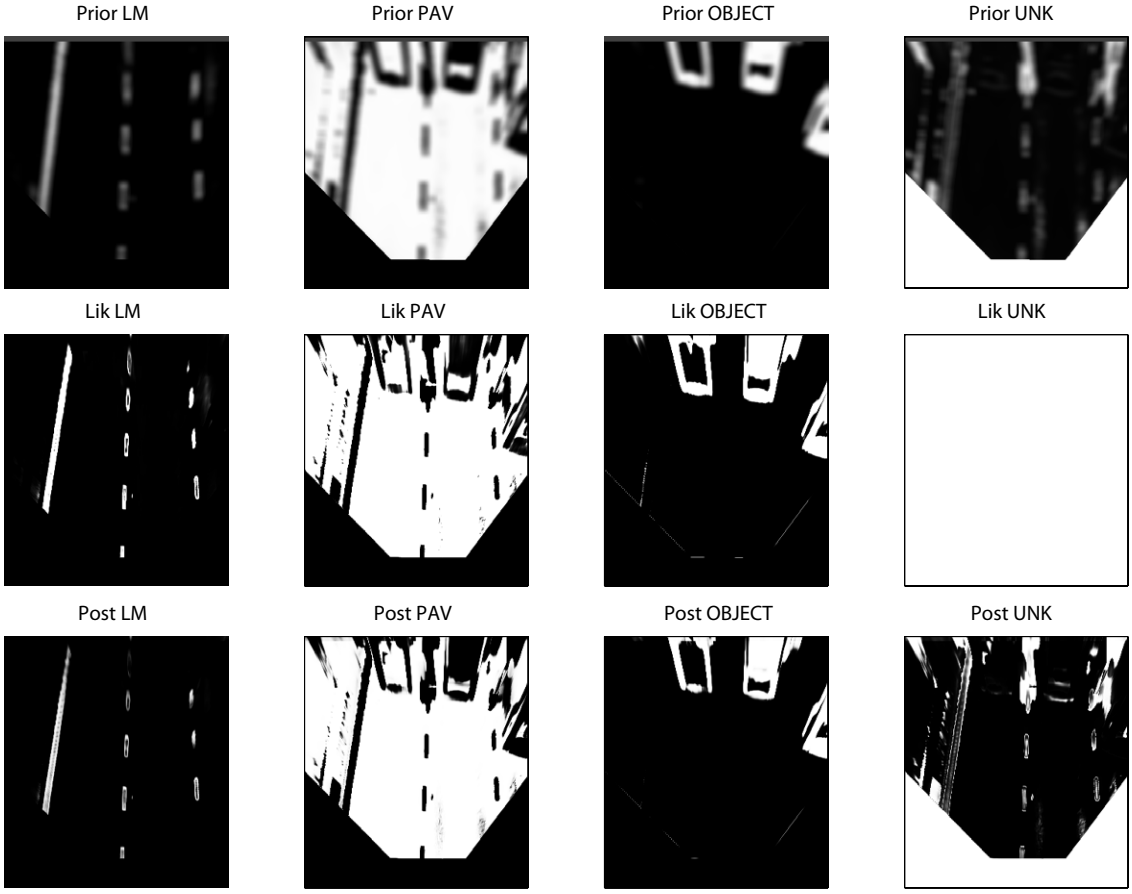
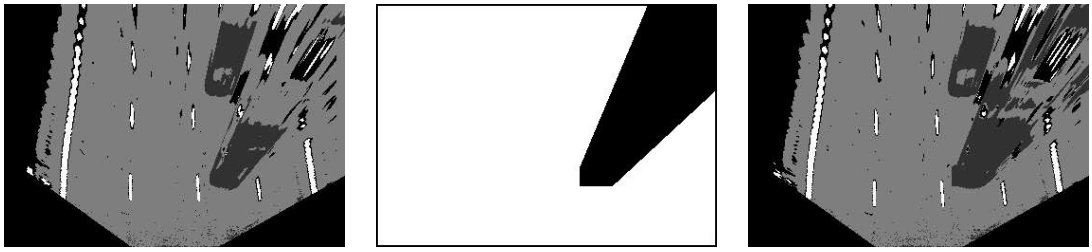


Figure 4.11: Probability maps for an example image.

togram of the intensity values of the example image. As can be observed, the result of the EM algorithm is a mixture that perfectly fits with the histogram. Therefore, the unbalance between amount of data of each class is handled naturally by the EM algorithm by assigning the corresponding weighting factors to the mixture components. For this example image, the weight of the unknown class is very low, which means that, for this feature, most of the pixels of the image can be correctly assigned to the other classes.

4.3.3 Prior probabilities

The prior probability of each defined class must be computed in order to obtain the final posterior probability for each pixel of the image. These prior probabilities represent prior knowledge of the probability of a pixel to belong to each class before examining its associated observations. Typically, prior information is obtained from



(a) (b) (c)

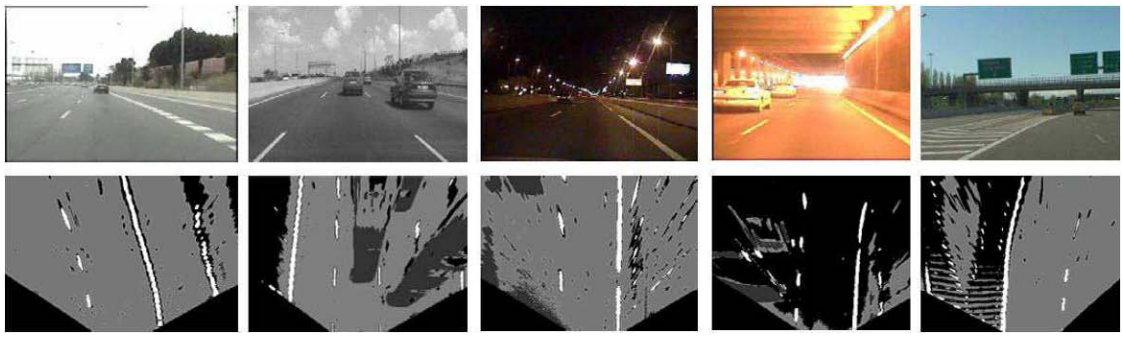
Figure 4.12: Prior probability map provided by vehicle tracking: (a) segmentation at instant k without prior information; (b) image mask derived from the previous detection of one vehicle in the transformed domain, the black pixels are predicted to belong to the vehicle; and (c) the segmentation result obtained using the prior information, where the pixels classified as belonging to the vehicle are more compact and easier to detect and model, as compared to those in (a).

the posterior probabilities from previous time instants, through the so-called dynamic or prediction models.

Different sources of information can be used to generate prior models. Specifically, if an estimation of the ego-motion of the camera is available[7], we can generate prior probability maps from the previous time instant posterior probabilities applying a translation that compensates the ego-motion and adding some Gaussian blurring. An example is given in figure 4.11 first row. Second and third rows show, respectively, the obtained likelihood images (i.e. the likelihood value of each class for each pixel, normalized between 0 and 1 for a better visualization), and the posterior images.

In the case in which there are vehicles in the scene, we use their prediction model (as described in published works related to vehicle detection and tracking [5, 6]), which defines the regions of the image that are more likely to contain vehicles in the next image via a binary mask. Figure 4.12 shows a comparison of the segmentation results obtained with and without the use of this prediction model. The mask shown in figure 4.12 (b) can be directly combined with the masks of figure 4.11 upper row, in order to obtain the complete prior maps. The prior for pavement, lane markings and unknown class can be multiplied by the mask. Hence, the probability to belong to one of these classes of the pixels that belong to the black region of the mask (as shown in Fig. 4.12 (b)), is zero. As a result, the pixels of this region will be more easily classified as belonging to the object class by the prior maps.

[7]It can be obtained easily from the rectified images through the computation of a simple translation plus rotation motion model, for instance, with point correspondences between consecutive images.



(a) (b) (c) (d) (e)

Figure 4.13: Example of the application of the Bayesian classifier. Note that the classifier has been applied for these examples to the transformed images, and not to the original ones. This is because further processing modules operate on this transformed domain.

4.3.4 Tests and discussion

Figure 4.11 shows the complete set of probability maps for an example image. Provided the intensity values of the rectified image of the road, I_{xy}, and the response to the lane markings filter, L_{xy}, the Bayesian classifier is applied as explained in the previous subsections, obtaining the posterior probability maps shown in the bottom row of figure 4.11. The prior probabilities are shown in the upper row for the different classes. The likelihood maps shown in this figure correspond to the product of the two distinct likelihood models described for the two image features, intensity and response to lane markings detector, including the unknown class, which is depicted as a uniform image.

As shown, the result is a posterior map that clearly classify pixels as belonging to pavement, lane markings and objects. The unknown class is mainly assigned to pixels in the upper part of the image, which is something expected, since this region of the transformed domain contains more uncertainty as it corresponds to very few pixels in the original image.

Figure 4.13 shows the resulting segmentation for a number of example images. In the segmented image, for clarity, the pixels have been colored according to their classification: the light gray pixels are those that likely belong to the pavement of the road, the dark gray pixels are those that are assigned to the wheels and shadows of the lower parts of the vehicles, and the white pixels belong to the lane markings. Black pixels are those that have not been classified as belonging to any of the three previous classes, and therefore remain as unknown pixels. The result is very accurate for all of them, except for the tunnel sequence shown in (d), in which the sudden illumination change makes the classifier set to unknown most of pavement pixels. Nevertheless,

this does not affect neither the classification of lane markings nor that of objects.

The first image, shown in (a), corresponds to a sequence where road elements appear very clearly and distinguishable in intensity, mainly due to the favorable illumination conditions. The segmentation shows correct classification of lane markings, as well as pavement pixels. As observed, the pixels classified as unknown correspond to the far distance, since these regions of the transformed domain correspond to few pixels of the original image, and thus contain very few information. The example in (b) is similar to the example in (a) but with the presence of vehicles, that appear clearly identified by clusters of pixels painted in dark gray. In this case, the black pixels of the left part of the image correspond to the median strip of the road, which does not fit the likelihood models for the defined classes, so that it is not misclassified and left as unknown. The color of the road, though being significantly different from the one of example (a) does not disturb the correct classification of the pavement pixels.

Examples (c) and (d) share the same difficulty: there are severe illumination problems that may lead to incorrect pixel classifications: in (c), which corresponds to a night sequence, the segmentation is clearly correct, but with some noisy misclassifications of isolated pixels. These errors, though, would not affect the road modeling stage, as it is robust against outliers due to the utilization of the EM algorithm. In (d), the illumination conditions are worse (due to strong saturation) but only corrupt the classification for pavement pixels, which are left as unknown, instead of being misclassified as belonging to any other classes. Analogously to the previous example, the road modeling stage, since only uses the lane markings pixels, is not affected in this situation. The utility of the unknown class is highlighted with these examples: it allows to avoid misclassifications of pixels when there is not enough information in the images, or it is confusing. It is always less risky to determine that a pixel belongs to an unknown element rather than erroneously classify it as another element.

The example in (e) shows a case in which there are lane markings on the road that are correctly set as unknown as they do not fit the likelihood models of the lane markings class and are actually not interesting for road model fitting (left pixels in the image).

The correct detection of lane markings is most of times subject to the correct detection of vehicles. The proposed approach, due to the different used likelihood models and the prior information is able to correctly classify pixels that belong to vehicles as belonging to the object class even in difficult situations. For instance, let us consider an specific challenging situation in which there are white vehicles on the image (or at least with high intensity values I_{xy}). A white vehicle could be, in fact, misclassified it we only consider the intensity level. Nevertheless, the response to the lane marking detector is only high for bright stripes like those corresponding to the lane markings. A wider white region in a row, like the one corresponding to a white vehicle results in low values after of L_{xy}, and thus the computed likelihood of the lane

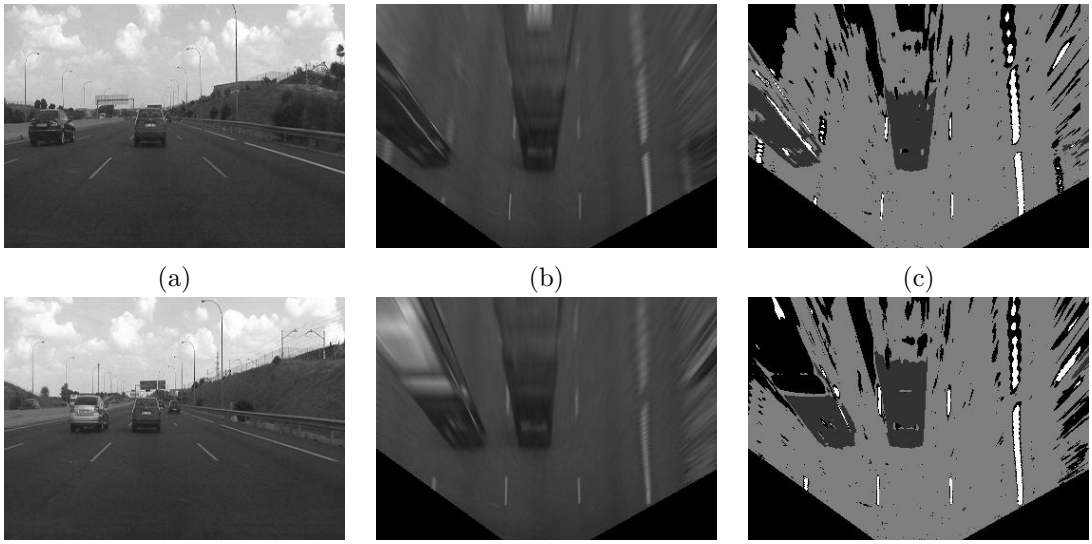


Figure 4.14: Detection of dark and white vehicles.

markings class, $p(\mathbf{z}_{xy}|X_L)$ for those pixels is very low. This question is exemplified in figure 4.14, in which images (a) and (d) correspond to two different instants of the same sequence. Images (b) and (e) are, respectively, the transformed images for the two original images, and (c) and (f) the resulting segmentation after applying the Bayesian classifier. Observe that the scenes are almost identical but for the fact that the left vehicle in (a) is dark, while in (d) it is as bright as the lane markings. Attending to the segmentation results, we can observe that both segmented images (c) and (f) are very similar, except for the pixels in the upper part of the left vehicle, which are classified as belonging to the object class for the dark vehicle in (c), and to the unknown class for the white vehicle in (f). The reason is that for these pixels of the white vehicle, none of the classes obtain high likelihood values: on the one hand, $p(I_{xy}|X_L)$ is high, but $p(L_{xy}|X_L)$ is low, and thus $p(\mathbf{z}_{xy}|X_L)$ is also low; on the other hand $p(I_{xy}|X_P)$ and $p(L_{xy}|X_P)$ are low, and also $p(I_{xy}|X_O)$ and $p(L_{xy}|X_O)$ are low, so that $p(\mathbf{z}_{xy}|X_P)$ and and $p(\mathbf{z}_{xy}|X_O)$ are both low. Therefore, the bright pixels of the vehicle are not correctly detected as object[8], but also not misclassified as lane markings and thus will not disturb further processing stages.

Summarizing, this set of examples shows the typical performance of the proposed segmentation under extremely different conditions: the result is that the classification

[8]For the sake of clarity, vehicle detection works based on this segmentation use only the pixels classified as object of the lower part of the vehicle, which are correctly detected in the example, since these part of the vehicle is the less distorted by the rectification of the image. Hence, the vehicle detection process is actually able to detect white vehicles without problems.

of pixels is correct most of times, suffering only some noisy classifications when the illumination conditions are severely unfavorable.

4.4 Lane modeling

Throughout the elaboration of this thesis, different road models have been devised, designed and implemented, ranging from simple ones, based on linear models, to complex and flexible ones, based on second order curvature lane models (details can be seen in works like [88, 89]). Nevertheless, we shall focus here on the linear road model, which is much more robust and sufficient for the main concern of the thesis (i.e., the estimation of vanishing points and its application to plane rectification). Note that in the lower part of the image, which corresponds to the road area that is closer to the camera, we can assume that the curvature of the road is negligible and, hence, we can apply estimation of linear models without committing significant errors in the estimation of the main vanishing point. This assumption is reasonable for non-urban roads, where the curvature level of the lanes is typically very reduced.

Specifically, we will describe a lane model that uses a Kalman filter to add the required temporal coherence to the lane markings detection. The spatial coherence is afterwards provided by the EM algorithm (as defined in chapter 3), that determines the position of the lines that best fit the lane markings, and that uses as initialization the results of the Kalman filter.

The following subsections describe first the "own-lane tracking" technique that is used to detect and track the position of the ego-vehicle inside its own lane with time, and then the use of the EM algorithm to obtain the parameters of the road model considering multiple lanes.

4.4.1 Own-lane tracking

The own lane tracking holds for the commonly used functionality of DAS systems that analyzes the evolution of the images and determines the width of the own lane, w_t, and the position of the vehicle within it, x_t. Hence, lane changes are also detected by analyzing the evolution of x_t and w_t, from which the transversal motion of the vehicle inside its lane can be monitorized through time.

This evolution can be easily described as a dynamic linear system that can be solved with a Kalman filter[9] defined by the following state-space equation:

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + B\mathbf{u}_t + \mathbf{n}_t \tag{4.7}$$

[9] Assuming that both the measurement and process noise are Gaussian.

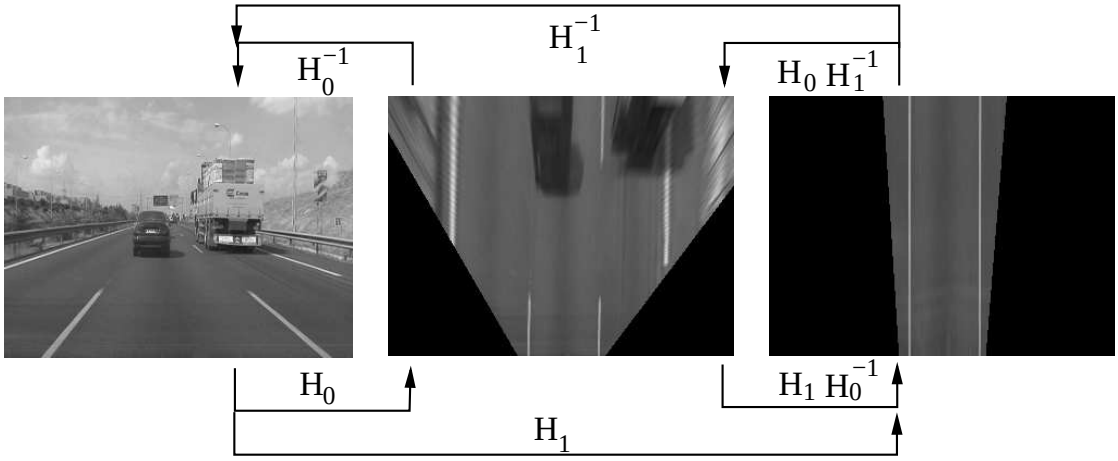


Figure 4.15: Image-plane to road plane transformations: H_0 image-plane to road-plane; H_1 image-plane to zoomed road-plane.

where t indexes time instants, and the state vector is $\mathbf{x}_t = (x_t, w_t, \dot{x}_t, \dot{w}_t)^\top$, where $\dot{x}_t$ stands for the variation of x_t from one instant to the other (i.e. the transversal velocity), and $\dot{w}_t$ for the corresponding variation of the width of the lane. The measurement vector is, at each instant, $\mathbf{z}_t = (x_t, w_t)^\top$, which is the instantaneous measurement of the target parameters (the measurement algorithm is discussed in further paragraphs). The transition matrix, A, and the input control matrix, B, are given by a constant-velocity model. The use of this model does not mean that we assume a constant velocity over all time; rather, the statistical model of the motion assumes undetermined variations of the instantaneous velocity (i.e. accelerations) with a Gaussian profile, modeled by $\mathbf{n}_t$. These dynamics are represented by the following matrices:

$$\mathrm{A} = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \quad \mathrm{B} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{4.8}$$

The control matrix is used to modify the estimation of the transversal position of the vehicle when lane changes are detected. The input control vector is obtained as:

$$\mathbf{u}_t = \begin{cases} w_t & \text{if } x_t > \frac{1}{2}(W + w_t) \\ -w_t & \text{if } x_t < \frac{1}{2}(W - w_t) \end{cases} \tag{4.9}$$

where W is the width of the image in pixels; hence, when the transversal position exceeds the boundary, at the left or right, of the estimated lane, the input control is activated and the transversal position is shifted.

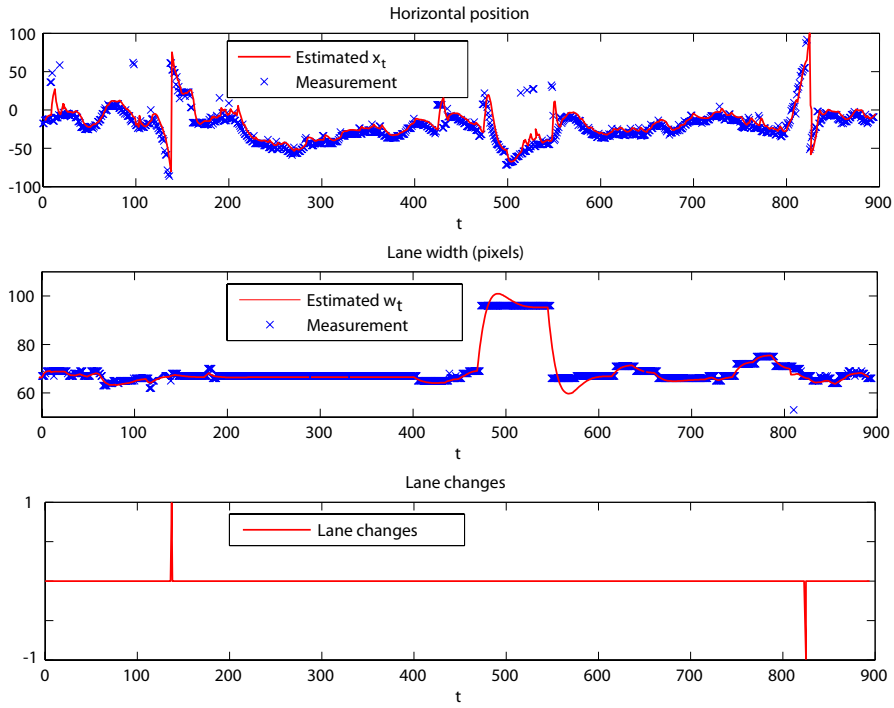


Figure 4.16: Estimation of the transversal position of the own-vehicle inside its lane (varying from 100 to -100% corresponding to the right-most and left-most position inside the lane), the width of the lane, and the detected lane changes.

For the computation of the measurement vector, $\mathbf{z}_t$, a zoomed transformed image is created. Figure 4.15 shows the relationship between the image plane, the road plane, and this zoomed road plane. As shown, the zoomed road plane contains only the very lower part of the original image plane in order to reduce the presence of vehicles or information about adjacent lanes. Within this zoomed image, only the lane markings that belong to the own lane are displayed. In addition lane markings can be modeled by straight lines with lower error.

The measurement vector is obtained using the Hough transform on the pixels classified as lane markings by the Bayesian classifier in the zoomed road plane. Two straight lines model the two-lane markings of the own lane. These lines intersect the bottom boundary of the image in two points. The distance between these points is the measurement w_t, while the difference between their middle point and the mid-bottom point of the image is x_t.

The tracking of the defined parameters is illustrated, for an example video sequence, in figure 4.16. In the upper figure, the estimated transversal position x_t is shown in solid line, whereas the instantaneous measurements are shown in blue crosses. It de-

picts a vehicle driving quite centered in its lane, and with two lane changes, correctly detected, and shown in the bottom figure. Nevertheless, there are some instabilities between frames 450 and 550, which are caused by incorrect detections in the Hough transform. The estimation of the width, w_t, is analogously shown in the central figure. The motorway of the analyzed sequence has a very steady lane width except for a few meters, where two lanes fuse after an entrance ramp. As shown, the noisy measurements are smoothed with the Kalman filter, which also allows for the prediction and correct detection of the lane change event.

4.4.2 Multiple-lane detection

This section presents a method for the extension of the detected own-lane to generate a road model with multiple lanes, which is illustrated in figure 4.17. We will make use of the information of the own-lane as well as other processing stages that have been proposed along this thesis, e.g. the line segment detector or the EM algorithm for vanishing point estimation.

The presence of adjacent lanes is hypothesized once the own-lane has been estimated, based on the assumption that the width of these lanes is similar to the one of the own-lane. Therefore, we can go back to the transformed domain and evaluate the presence of additional lane markings to complete the road model. An efficient approach is just to evaluate the pixels classified as lane markings of the rectified image, assuming that they form vertical stripes. This way, we can evaluate each column of the image, and count the number of pixels classified as lane markings. Figure 4.17 (e) illustrates, as a blue histogram-like profile, the obtained counts for each column for an example image. The peaks of this profile correspond to the expected transversal position of the lane markings, including the own lane. Using the information of the lane tracker we can divide this one-dimensional histogram into regions corresponding to each hypothesized lane marking and search for the maxima at each one. Provided that the own lane is always there, we can use the value of the maxima corresponding to its lane markings to specify an appropriate threshold to determine if the adjacent maxima is high enough to correspond to actual lane markings or not.

Finally, we arrive to the estimation of the number of lane markings that are visible in the transformed domain. This way, we can upgrade the road model to consider the transversal position and the width of the detected adjacent lanes.

The result is a vector Θ that contains the parameters of the lines corresponding to each detected lane marking. For instance, the parameter vector of the example illustrated in figure 4.17 consists on four vertical lines that can be expressed as $\Theta = \{\mathbf{s}_j\}_{j=1}^{4}$, where $\mathbf{s}_j$ is the 3-vector that characterizes each line as already defined in chapter 3 as the so-called support lines of the model.

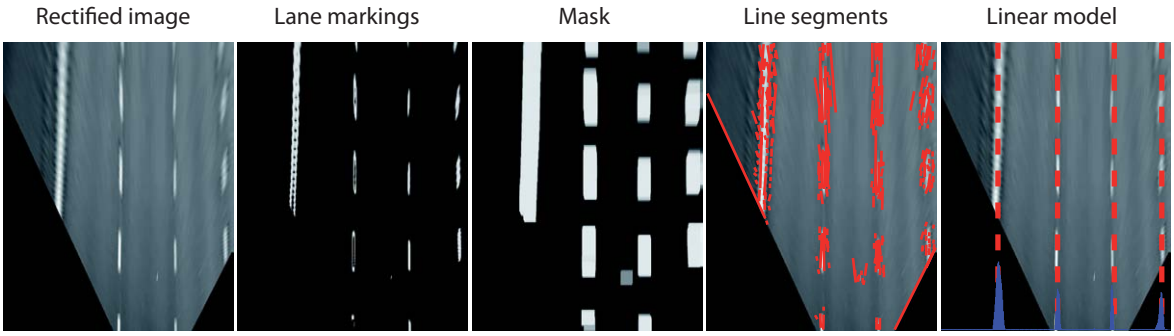


Figure 4.17: Estimation of the linear road model using the Bayesian classifier and the EM algorithm.

This information is enough to initialize the EM algorithm. At this point we combine the specific information of the scenario, obtained with the Bayesian classifier, and the generic method for detecting line segments described in chapter 2. The pixels that have been classified as lane markings are selected, as shown in the example of figure 4.17, where (a) is the rectified image, and (b) shows the pixels classified as lane markings painted with an intensity value associated to their corresponding $P(X_L|\mathbf{z}_{xy})$ value. To facilitate the task of the line segment detector, a morphological dilation is applied on this image. The result, shown in (c), can be seen as a mask that determines the points in the image where the SSWMS algorithm shall search for line segments. Specifically, the probability map $\hat{p}(x, y)$, used by the SSWMS algorithm, as described in chapter 2, is multiplied by this mask to ignore the potential line segments that fall outside the dilated regions around lane marking pixels (and thus speeding up the detection process). The example of figure 4.17 (d) shows the set of detected line segments, depicted in solid red lines, which are, although quite noisy, useful for the road model fitting.

Within this fronto-parallel view of the road plane, the expected vanishing point where support lines meet at is at the infinity (a vertical vanishing point). The EM defined for the projective plane can provide the corrected parameters of these lines. Typically, if the initialization is good, one iteration is enough to obtain very accurate results. The example of figure 4.17 (e) shows the obtained best fit for four lines meeting at a common vanishing point in the infinity.

4.4.3 System test

The methods described in this chapter, as well as other additional strategies (such as vehicle detection, lane presence verification or curvature detection) have been inte-

grated and developed in a real video-based DAS that works online for video sequences.

All the developments have been done in C++ programming language, under an MFC solution for Windows, using Direct Show primitives. This architecture allows real-time performance and on-board operation as well as continuous visualization of the processed data. For the trials, the acquiring system was composed of a forward looking digital video camera SONY HDR-HCR5E, installed near the rear mirror, and a FireWire connection to the processing system. For this set-up, the size of the images is 360×288, which allows the system to work near real-time, at 15 frames per second on average (including video visualization and output data generation) in a laptop Core2Duo at 2.2 GHz with 2 GB of RAM.

The on-road trials have been conducted in different roads in Madrid, Brussels, Milano, Torino, and the A4 Brescia-Padova motorway. From these trials, we have collected a large number of sequences (with a total length of around 150 minutes) from which we have gathered relevant output data. The target is the evaluation of the road model generated by the system and monitor its main output parameters, namely, the position of the vehicle inside its lane and the number of lane changes. Additionally, results of the complete DAS about other issues, such as curvature, detection rate of vehicles, position and dimensions of these vehicles, are addressed in different works [6, 88, 89].

Different scenarios are considered to demonstrate the ability of the system to adapt to the uncontrolled outdoor scenario, including changing illumination conditions, different pavement color, varied weather conditions, different type of lane markings, presence of vehicles that cause occlusions, etc. According to the results obtained through the tests, we can extract the following conclusions, which are supported by objective numerical results at the end of this section.

The position of the own vehicle inside its lane (a feature that depends on the lane tracker performance) is estimated with high accuracy in almost all situations. Some examples are shown in figure 4.18. The green overlaid region defines the closer part of the own lane, according to the detected lane markings, depicted with thick blue lines (note that these lines are represented in the original domain for a better visualization, and have been obtained by inverting the plane rectification transformation). The center of the lane is marked with a thin yellow line, and the center of the image with a shorter black line. The relative position of the vehicle in the lane is depicted with a numerical indicator, which is 0% at the center of the lane, and -100% and 100% at its left-most and right-most position, respectively. As shown, the width of the lane may vary, but it is accurately estimated by the system.

The lane change detection is one of the higher performance features of the developed system (as it is shown in table 4.2). The second row of figure 4.18 shows some detected lane changes, whose directions are indicated with a superimposed icon.

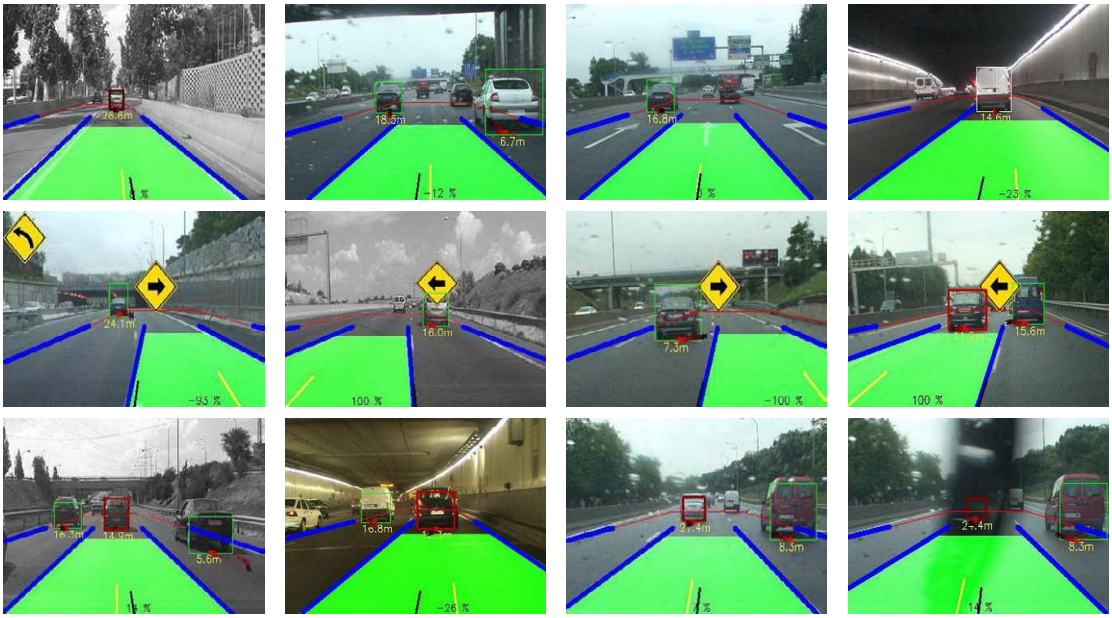


Figure 4.18: Examples showing the detected elements of the road: In the upper row some representative examples of the accuracy of the lane tracker are shown; the second row shows examples of lane change events; and the lower row illustrates vehicle detected in difficult scenarios.

Regarding the detection of multiple lanes, the system has shown excellent performance in most situations. As long as the detection of adjacent lane markings is correct, which is true for most situations, the system is able to hypothesize the presence of adjacent lanes to the own lane and include their estimations within the EM framework. Different examples illustrate this ability in figure 4.18: when driving in the right lane, the system is able to determine that there are no more lanes at the right, and analogously when driving in the left-most lane. When driving in the central lanes, three lanes are hypothesized at most. This number is not a limitation of the algorithm, but a design parameter, as beyond these, there are not typically enough lane markings pixels to take a reliable decision.

One of the major abilities of the proposed system is its demonstrated robustness against challenging, unpredictable situations. On-road trials featured, for instance, sudden illumination changes (e.g. when entering into tunnels, or passing below a bridge), appearance of elements such as wipers, drawings on the road, drops on the windscreen, irregular color of the pavement, reflections, etc. The proposed models rely on the robustness of the Bayesian classifier, that is able to handle these difficulties with the "unknown" class. As a conclusion, the system has been designed to search for and model lane markings, the color of the pavement and vehicles, and to not be deceived

	Type of scenario	
Event	Typical	Complex
No. of lane changes	204	211
No. of detected lane changes	196	197
No. of false positive lane changes	3	9
LCDR (%)	96.08	93.36
LCFPR (%)	1.47	4.27

Table 4.2: Detection results for different scenarios.

by this type of uncontrolled scenarios.

In order to reflect more objectively the performance of the designed system, a couple of relevant parameters regarding lane modeling have been defined, i.e. lane change detection rate (LCDR), and lane change false positive rate (LCFPR). The former is defined as the number of correct detections over the total number of events. The latter is defined as the number of false positives over the total number of events. They have been measured off-line for a large set of sequences. Those sequences involve two sets of scenarios: typical and complex. The former involves the typical driving conditions in motorways, this is, well-painted lane markings, variable illumination conditions with soft changes, low/medium traffic density, or favorable weather conditions (including mild rain). The latter comprises those conditions that are not so commonly encountered in motorways (such as dense traffic or variable illumination conditions with rapid and abrupt variations), or those that occur for short periods of time (such as tunnels, which entail artificial illumination conditions).

Table 4.2 shows the performance of the system, in terms of detection and false positive rates, for the two sets of scenarios. As can be observed, the lane change detection rate is very high (over 96 %) in a typical scenario. In this situation, the false positive rate is under 2 %. As for complex scenarios, the lane change detection rate decreases to 93.36 %. Although the performance is slightly reduced in difficult scenarios, the system still shows excellent accuracy.

Chapter 5

Plane rectification

The plane rectification problem is analyzed in detail in this chapter related to the estimation of vanishing points. Although it is a task that can be solved using other type of information, such as through point matching algorithms between different views of the plane [46], this chapter focalizes in alternatives based on vanishing points, and their application in the described road scenario.

Plane rectification is a problem conditioned to the available amount and type of information. Namely, the number of vanishing points, their relationships, the knowledge of the camera calibration, the relative pose of the camera with respect to the plane, the presence of support lines meeting at vanishing points, etc. Moreover, different results are achieved according to the properties that are recovered through the transformation, i.e. parallelism, angles, distances, etc. In that sense, this chapter is devoted to examine and evaluate the obtained results for different configurations of available information, addressing as well a number of proposal methods to rectify planes for the road scenario.

The structure of the chapter is as follows: in section 5.1 we explain the geometric concepts[1] behind such a transformation, and also which are the methods more typically used in the literature. Section 5.2 introduces the particularities of the road scenario with respect to the road plane rectification, and discuss about how the related works in the field of DAS compute and use this transformation. Finally, in section 5.3 we introduce a number of proposed approaches to rectify the road plane, using the methods described along this thesis for the computation of vanishing points and the road model. The presented cases are ordered in an incremental approach, assuming an increasing number of unknowns that must be retrieved to perform the road plane rectification.

[1] Following the reference book by Hartley and Zisserman [46].

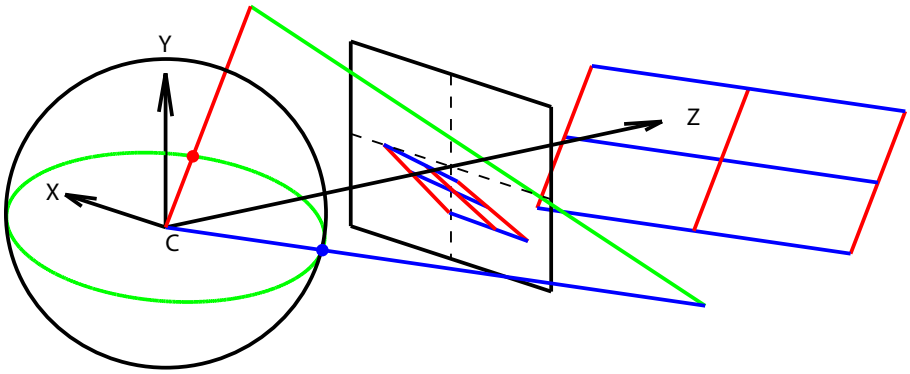


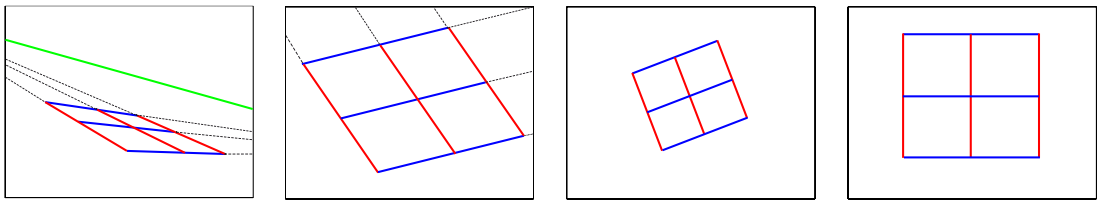
Figure 5.1: Representation of a plane, defined by two sets of parallel lines, their projection into the image plane, and the unit sphere visualization. The two main vanishing points in the image identify these directions of the space.

5.1 Concepts and methods

A world plane is projected by a camera into the image plane as illustrated in figure 5.1. For instance, the world plane can be defined by two sets of parallel lines, which are, in this example, mutually orthogonal for a better visualization. The target is to transform the image, which is shown in figure 5.2 (a) into a new one, as shown in (d), which is typically known as "fronto-parallel view", "bird's eye view", "inverse perspective mapping" of the plane, or simply "rectified image". The obtained image is actually a scaled version of the world plane, such that most of the geometric properties of the elements of the plane are recovered: parallelism, angles, area ratios, distances, etc.

In a formal way, the target is to compute the plane to plane homography that transforms points from the image plane to the world plane. This transformation is encoded by a 3×3 matrix, H, that defines the projective relationship between the two planes. Any point in the image plane, denoted in homogeneous coordinates as $\mathbf{x} = (x_1, x_2, 1)^\top$ is transformed into a point of the world plane as $\mathbf{x}' = H\mathbf{x}$.

Many works have been published related to plane rectification, although most of the relevant concepts and methods for this topic can be found in the book of Hartley and Zisserman [46], which includes the linear algebra notation that is used in this thesis. The following two subsections introduce two alternative methods to compute the target homography: (i) based on the computation of H through point-correspondences, and (ii) based on the construction of H using vanishing points and other geometrical constraint of the scene.



(a) (b) (c) (d)

Figure 5.2: Transformation between an original image (a) and a fronto-parallel view of it (d); images (b) and (c) show intermediate steps: in (c), the parallelism is recovered; and in (d) the angular information is also correct.

5.1.1 Point-correspondences

A point correspondence between the world and image plane, denoted in inhomogeneous coordinates as $(x, y) \leftrightarrow (x', y')$, has been shown to generate two equations for the elements of H, denoted as h_{ij} which corresponds to the i-th row and j-th column [46]:

$$x'(h_{31}x + h_{32}y + h_{33}) = h_{11}x + h_{12}y + h_{13}$$
$$y'(h_{31}x + h_{32}y + h_{33}) = h_{21}x + h_{22}y + h_{23}$$

Given four such correspondences[2] lead to eight equations that are sufficient to solve for H up to an unknown scale. Nevertheless, when more point correspondences are available, numerical methods can be applied. Although this problem can be solved reliably with non-linear methods, we briefly introduce here the DLT (Direct Linear Transformation) algorithm, described in [46], as it is much faster and simpler, and indeed is typically used as initialization for higher order approaches. We have used this method successfully for road plane rectification [88, 89].

Given a set of N point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x'}_i\}$, the DLT algorithm computes the cross product $\mathbf{x'}_i \times H\mathbf{x}_i = \mathbf{0}$, where $\mathbf{0} = (0, 0, 0)^\top$. If the j-th row of the matrix H is denoted by $\mathbf{h}^{j\top}$, and considering $\mathbf{x}'_i = (x'_{i,1}, x'_{i,2}, x'_{i,3})^\top$, the cross product yields:

$$\mathbf{x}'_i \times H\mathbf{x}_i = \begin{pmatrix} x'_{i,2}\mathbf{h}^{3\top}\mathbf{x}_i - x'_{i,3}\mathbf{h}^{2\top}\mathbf{x}_i \\ x'_{i,3}\mathbf{h}^{1\top}\mathbf{x}_i - x'_{i,1}\mathbf{h}^{3\top}\mathbf{x}_i \\ x'_{i,1}\mathbf{h}^{2\top}\mathbf{x}_i - x'_{i,2}\mathbf{h}^{1\top}\mathbf{x}_i \end{pmatrix} \tag{5.1}$$

[2]Provided that no three points are collinear.

It has been shown that (5.1) leads to

$$\begin{pmatrix} \mathbf{0}^\top & -x'_{i,3}\mathbf{x}_i{}^\top & x'_{i,2}\mathbf{x}_i{}^\top \\ x'_{i,3}\mathbf{x}_i{}^\top & \mathbf{0}^\top & -x'_{i,1}\mathbf{x}_i{}^\top \end{pmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0} \tag{5.2}$$

which has the form $A_i\mathbf{h} = \mathbf{0}$, where A_i is a 2×9 matrix, and $\mathbf{h}$ is a 9-vector made up of the entries of the matrix H. By assembling the A_i matrices into a single $2N \times 9$ matrix A, we obtain the equation $A\mathbf{h} = \mathbf{0}$ which is solved by obtaining the SVD of A.

5.1.2 Stratified plane rectification

Just as a short summary, we introduce in this subsection the concepts of stratified metric rectification of planes, described by Liebowitz and Zisserman [69], as an alternative to use point correspondences. This work clarifies the relationship between the recovered geometric properties of the plane according to the information (e.g. vanishing points) used to generate the transformation matrix.

The method proposed by Liebowitz and Zisserman [69] decomposes the target homography as the combination of three different transformations, each one holding different type of information:

$$H = H_s H_a H_p \tag{5.3}$$

where H_p is the matrix containing the projective information, H_a the affine, and H_s the similarity. The projective matrix, H_p, once known, can be applied to restore, mainly, the parallelism, which is lost in perspective images. As an example, let us follow the illustrations in figure 5.2. In (a) we have the perspective image of a world plane, where vanishing points are shown as finite points. By applying H_p we obtain a new image, shown in (b), where the projective distortion has been removed and, therefore, parallel lines are actually shown parallel. With this first step, vanishing points are sent to the infinite. We can generate this transformation by obtaining the parameters of the line at the infinity (shown in green in figure 5.2 (a)), which is the line that contains all the vanishing points of the plane[3]. The line at the infinity of the world plane can be obtained using two vanishing points, $\mathbf{v}_1$ and $\mathbf{v}_2$ in homogeneous coordinates, corresponding to directions that belong to the plane:

$$\mathbf{l}_\infty = (l_1, l_2, l_3)^\top = \mathbf{v}_1 \times \mathbf{v}_2 \tag{5.4}$$

The projective transformation matrix H_p can be then obtained as:

[3]There could be infinite vanishing points as infinite sets of parallel lines could be selected.

$$H_p = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{pmatrix} \tag{5.5}$$

The second step involves the restoration of angles, which is done by obtaining and applying the affine transformation H_a. This way, we transform the image in (b) into (c), where the sets of lines in red and blue are shown with their correct angular relationship (orthogonality in this example). This matrix is defined as:

$$H_a = \begin{pmatrix} \frac{1}{\beta} & \frac{-\alpha}{\beta} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.6}$$

where α and β specify the image of the circular points $\mathbf{I}$ and $\mathbf{J}$, which are a pair of conjugate points on the line at infinity which are transformed from coordinates $(1, \pm i, 0)^\top$ on the metric plane to $(\alpha \mp i\beta, 1, 0)^\top$ on the affine plane [69]. Different alternative options arise here to compute these coefficients. Liebowitz and Zisserman [69] propose to compute the coordinates of the circular points on the affine plane through using constraints such as a known angle on the plane, equality of two unknown angles or a known length ratio.

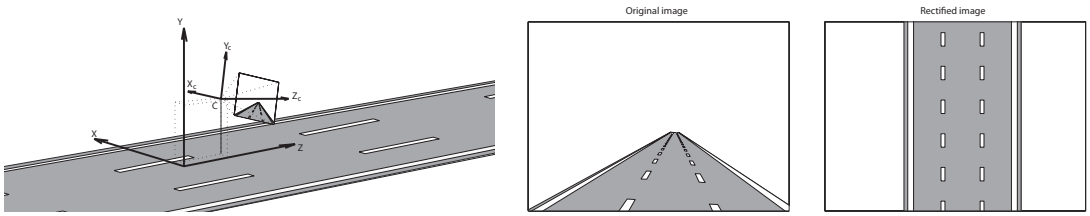
At this point the most important properties have been already recovered, although a similarity distortion remains, that can be removed applying H_s:

$$H_s = \begin{pmatrix} sR & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix} \tag{5.7}$$

that namely rotates using the 2×2 rotation matrix R, scales by s and translates according to $\mathbf{t} = (t_x, t_y)^\top$ the image in (c) to obtain the final rectified image (d). This can be done using a known length on the image and a reference coordinate system.

5.2 Road plane rectification

We can consider now the road scenario as a particularization of the generic plane rectification problem. The target is the same, but the relative pose of the camera with respect to the plane is now more restricted. Specifically, the camera is assumed to be installed inside a vehicle that is moving on the road, which is assumed to be locally flat, and that the camera is forward looking to the road ahead the vehicle. Figure 5.3 (a) illustrates this configuration. As shown, we have defined a world coordinate system, depicted as $\{X, Y, Z\}$ located on the surface of the road, with its axes aligned with the main directions of the road. The camera coordinate system addresses a relative



(a) (b) (c)

Figure 5.3: Plane rectification for the road scenario: (a) shows an scheme of the relationships between the camera, the imaged plane and the road plane; in (b) and (c), the original image and the target rectified image.

rotation and translation with respect to the world coordinate system. This rotation and translation illustrates that the camera is mounted inside the vehicle, near the rear mirror, such that it has some height above the road plane. On account of the dynamic environment, we will assume that the world coordinate system moves rigidly on the plane as the vehicle moves on the road, so that the Z-axis and X-axis are always locally parallel and orthogonal, respectively to the lane markings. The rotation of the camera coordinate system corresponds to the movement of the vehicle with respect to the lanes (for instance when the vehicle steers). The translation $\mathbf{c}$ is assumed to be fixed.

We shall discuss here about the adequacy of the flat plane model for the road surface. It is quite obvious that actual roads do have some curvature degree in different axis. Moreover, roads are designed pasting together stretches with different curvature in order to fuse them as a smooth path. For instance, cubic approximations to the clothoid are very typical on the Y-axis, and also for the X-axis, which determine varying slopes [31]. However, the parameters of these curves are designed following a comfort criterium, which is linked to the variation of curvature. At the maximum allowed speed in highways and mainly in motorways, curvature is very soft to fulfill comfort, and can be considered negligible for stretches of about 10 to 20 meters. Taking into account that this is approximately the range that we choose for the video analysis, the committed error fitting a plane to the road stretch is very reduced. The flat world assumption has been made by many authors (Bertozzi and Broggi [14], Kim [58], McCall and Trivedi [74], Shu and Tan [104], Southall and Taylor [107] to mention just a few) as it is accurate enough to fit road models, while significantly simplifies algorithms.

5.2.1 Pinhole camera projection

The projection process that generates the image shown in figure 5.3 (b) is described as a linear process using homogeneous coordinates for both the points in the 3D space and

the image coordinates. If we define a point in the 3D world as $\mathbf{X} = (X_1, X_2, X_3, X_4)^\top$ in homogeneous coordinates, its projection into the image plane combines two transformations: the former converts the point into the camera coordinate system, yielding $\mathbf{X}_c$, and the second projects it into the image plane, $\mathbf{x}$. These steps are depicted using the following expression:

$$\mathbf{x} = K\mathbf{X}_c = K(R|-R\mathbf{c})\mathbf{X} = P\mathbf{X} \tag{5.8}$$

where P is the so-called projection matrix, K is the camera calibration matrix, and R and $\mathbf{c}$ are, respectively, the abovementioned relative rotation and translation between the coordinate systems. Several considerations can be done independently for the intrinsic parameters (the camera calibration matrix), and the extrinsic parameters (the rotation matrix and translation vector), which are described in the following paragraphs.

5.2.1.1 Intrinsic parameters

The camera calibration matrix model that is considered in this thesis is

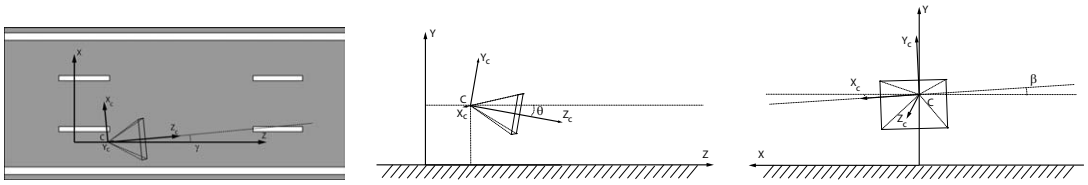
$$K = \begin{pmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.9}$$

where f is the focal distance in pixel units, and (x_0, y_0) are the coordinates of the principal point of the image, also in pixels. This matrix corresponds to an affine transformation that basically projects points of a ray that passes through the optical center of the camera into a single point of the image plane, with a translation in the image coordinate system given by the principal point.

Among the different considerations that we could discuss about K we highlight only one due to its relevance in the design of methods for plane rectification based on the computation of vanishing points. We can assert that the angular information of the space is contained in this matrix, since we can determine the relative angle between two rays passing through the optical center and intersecting the image plane in $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively. The cosine of the angle α between them is computed as:

$$\cos\alpha = \frac{\mathbf{v}_1'^\top \mathbf{v}_2'}{\|\mathbf{v}_1'\|\|\mathbf{v}_2'\|} = \frac{\mathbf{v}_1^\top \left(K^{-\top}K^{-1}\right)\mathbf{v}_2}{\sqrt{\mathbf{v}_1^\top \left(K^{-\top}K^{-1}\right)\mathbf{v}_1}\sqrt{\mathbf{v}_2^\top \left(K^{-\top}K^{-1}\right)\mathbf{v}_2}} \tag{5.10}$$

where $\mathbf{v}_1'$ and $\mathbf{v}_2'$ are the coordinates of the ray in the camera coordinate system, as discussed in appendix B. Typically, the matrix product $K^{-\top}K^{-1}$ is called the image of the absolute conic [46], and is denoted as ω. Just as a remark of the information



(a) (b) (c)

Figure 5.4: Yaw, pitch and roll angles, respectively in (a), (b) and (c) within the defined road scenario. The camera coordinate system is shown as $\{X_c, Y_c, Z_c\}$.

contained in vanishing points, there are excellent works that, provided pairs of mutually orthogonal vanishing points, derive equations on the unknowns of ω, and obtain as a result the camera calibration matrix [68, 93].

5.2.1.2 Extrinsic parameters

The rotation matrix is composed by three different rotations:

$$R = R_\theta R_\gamma R_\beta = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\gamma & 0 & -\sin\gamma \\ 0 & 1 & 0 \\ \sin\gamma & 0 & \cos\gamma \end{pmatrix} \begin{pmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (5.11)$$

where θ, γ and β correspond, respectively, to the pitch, yaw and roll angles, which are depicted in figure 5.4, and described as follows:

- **Pitch**: this is the angle of rotation with respect the X axis. It is typically large enough to show as much road as possible in the image. It may change due to slopes or bumps in the road.
- **Yaw**: angle of rotation with respect to the Y-axis. This angle depends on the installation of the camera. It is typically non-zero as the camera is usually installed at the left or right of the rear mirror, and rotated to show the own lane at the center of the image. This angle is related to the movement of the steering wheel.
- **Roll**: the angle of rotation with respect the Z-axis. If the camera is correctly installed, this angle should be zero, such that the horizon is shown as a horizontal line in the image.

Regarding the translation vector $\mathbf{c} = (c_x, c_y, c_z)^\top$, the height of the camera, c_y, can be considered approximately constant, while c_z and c_x are by definition fixed, since the camera coordinate system moves jointly with the world coordinate system as the vehicle moves on the road.

5.2.2 Related work

It is quite natural to find many works in the field of video-based driver assistance systems or traffic surveillance that use plane rectification. Nevertheless, a significant number of approaches do not pay attention to the dynamic nature of the environment and carry out the road plane rectification in an static way. Some works do not only assume that they exactly know both the intrinsic and extrinsic parameters of the camera, but also that their values do not vary over time, and thus perform a fixed plane rectification [14, 104]. Typically these works use this fixed model since they are focused on other topics, such as including complex road models [58, 107], parallax-based detection of vehicles and road surface [92], curvature analysis [104], or night-time algorithms for lane modeling [17].

These type of assumptions might be valid for surveillance system that do not model the dynamism of the road scenario [72], but in on-board systems it can lead to large errors in the rectification, due to the steering of the vehicle, its bumping, or slope changes of the road. Some authors have studied the dependence of the obtained transformation image according to the variability of the environment. For instance, Muad et al. [77] analyzes the impact of the pitch and yaw angles in the obtained rectified image in terms of radial distortion and parallelism. In this line, different authors have identified the pitch angle error (i.e. the error between the instantaneous pitch angle and that used to perform the rectification), as the main cause of the image distortion [13, 26, 53, 64, 87][4].

Purely vision-based approaches include the computation of the plane rectification as a first step, with an approximated pitch angle value, which is corrected later in different ways. For instance, Jiang et al. [53] compute the rectification with fixed parameters and checks the parallelism between the left and right lane markings, estimated as straight lines, correct the angles and re-estimate the rectification. Cerri and Grisleri [26] compute several rectifications using a range of pitch values, and then also checks the parallelism between the detected lane markings to determine which pitch angle was the correct one.

Other approaches explicitly use the vanishing points, such as Bücher [13], who assumes that the roll angle is zero and determines the pitch and yaw angle analytically from the position of the vanishing point. Besides, Klappstein et al. [60] use point correspondences between consecutive images to obtain the ego-motion and solve for the pitch and roll angles.

In the following section we discuss about different methods for obtaining plane rectification in the road scenario. Particularly, we will focalize mainly in the type

[4]To avoid the computation of the value of these critic angles through image processing techniques, some authors use specific odometry sensors, as inclinometers, that adequately used can retrieve the instantaneous rotation of the camera with respect to the road [64].

of information that we can obtain using vanishing points. For some cases we will obtain plane rectification up to affinity, i.e. recovering only the parallelism but not the angular information, as also done by other authors [47], which can be enough to apply, for instance, linear models of the road. Nevertheless, the target is to obtain at least plane rectification up to similarity, such that the only remaining distortion is an unimportant relative rotation, translation and scale.

5.3 Plane rectification: cases

The following subsections present the problem of plane rectification following an incremental complexity approach. This way, a number of methods are proposed according to the available information, which are specially designed to work in the challenging road scenario. Specifically we start with a simple case in which we have all the available information about the scene, as described in section 5.2.1, and thus we can easily obtain the target homography, H. The next steps increase the number of unknowns and thus the complexity of the required methods to obtain the homography.

At the end of this section we include a summary of the methods, schematized in table 5.1, which depicts, for each alternative, the assumptions done, the information that must be obtained and the achieved results.

5.3.1 Knowing the projection matrix

Many works, especially the former ones on road plane rectification [14, 53], perform a plane rectification through the application of a non-linear coordinate transformation. A detailed analysis of these equations reveals that they are actually using projective relationships between the coordinates of the road plane and the image plane assuming the knowledge of all the parameters of the projective matrix: i.e. the camera calibration matrix as well as the relative rotation and translation of the camera with respect to the road plane.

Following the linear notation introduced in section 5.2.1, the projective process of points of the road plane into the image plane can be described as a linear problem [46], using homogeneous coordinates. If we consider points in the 3D space that correspond to the road plane (with spatial coordinate $Y = 0$), the projective relationship shown in equation (5.8) becomes the following expression:

$$\mathbf{x} = \mathrm{P}\mathbf{X} = \begin{pmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{pmatrix} \begin{pmatrix} \mathrm{X} \\ 0 \\ \mathrm{Z} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{p}_1 & \mathbf{p}_3 & \mathbf{p}_4 \end{pmatrix} \begin{pmatrix} \mathrm{X} \\ \mathrm{Z} \\ 1 \end{pmatrix} = \mathrm{H}\mathbf{x}' \qquad (5.12)$$

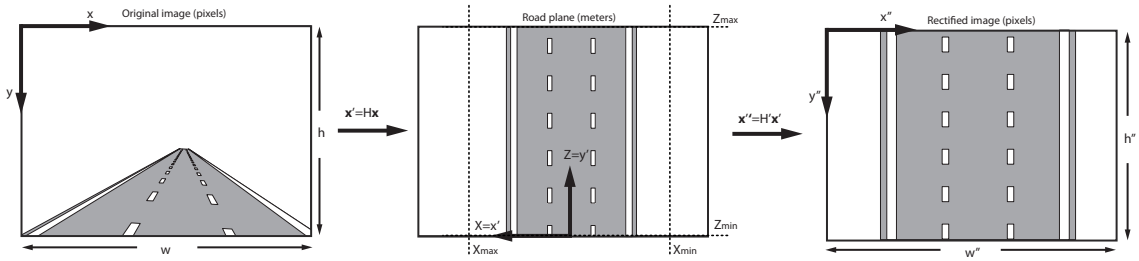


Figure 5.5: An additional affine transformation is required to obtain the targeted rectified image fitted in a given frame of $w'' \times h''$ pixels size.

which yields the target plane to plane homography H between $\mathbf{x}'$, which correspond to the coordinates of the points of the road plane, and the points on the image plane $\mathbf{x}$. This transformation is depicted in the two first images of figure 5.5 (from left to right). As shown, the obtained domain is the road plane itself, such that the coordinates (x', y') coincide with (X, Z). Therefore, we need an additional step to finally generate a synthetic finite image, (in pixels) of the road plane. Figure 5.5 shows this second step, which transforms the coordinates (x', y') into (x'', y'') as $\mathbf{x}'' = H'\mathbf{x}'$. From a practical point of view, we proceed obtaining H from P, and H' such that the rectified image fits into a defined frame size $w'' \times h''$, which is initially fixed according to memory allocation issues.[5]

In general, it is absolutely required to carefully select the output coordinate range, in order to represent adequately the stretch of road of interest. The generation of the fronto-parallel view of the road transforms the vanishing point of the lane markings into a point at the infinity, which, obviously, can not be displayed. Therefore, it is required to select a region of the original domain to be transformed, in order to obtain a finite rectified image. This way, it is in our hands how much road we want to be shown in the rectified image. Typically, we can select a range between 15 to 30 meters, in order to monitor the road and vehicles ahead that are represented with enough pixels in the original domain. In figure 5.5 we depict an example selection of the limits $(Z_{\max}, Z_{\min})$ and $(X_{\max}, X_{\min})$ to be displayed. The matrix H' is computed as:

$$H' = \begin{pmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{pmatrix} \tag{5.13}$$

where $s_x = -w''/(X_{\max} - X_{\min})$ and $s_y = -h''/(Z_{\max} - Z_{\min})$ are the scale factors for

[5]The mapping procedure loops over the pixels of the target image, and applies the inverse homography to find the pixel of the original image from which obtain the intensity level: $\mathbf{x}'' = (H'H)^{-1}\mathbf{x}$. Interpolation is used to obtain smoothed images with subpixel accuracy.

the abscise and ordinate axes. The translation factors are, respectively $t_x = \frac{w''}{2}$ and $t_y = h''$ provided that $X_{\max} = -X_{\min}$ and $Z_{\min} = 0$.

As shown in the right-most image of figure 5.5, the resulting image might be affected by affine distortion (i.e. anisotropic scaling in the x'' and y'' axes), provided that scale factors may be different to fit into the specified dimensions of the rectified image. Nevertheless, measurements on this rectified domain can be translated back into the actual coordinates of the road plane applying H'^{-1}.

5.3.2 Unknown extrinsic parameters

In this second case, we assume the knowledge of the camera calibration matrix, while the extrinsic parameters, i.e. the relative rotation of the camera with respect to the image plane, is left unknown. Obviously, this case is far more realistic than the previous one, since the rotation actually vary. For instance, the relative rotation of the vehicle with respect to in the Y-axis, which is associated to the movement of the steering wheel, is continuously changing.

On the other hand, the camera calibration matrix is an element that can be considered as static in many situations. This way, we can proceed obtaining an initial calibration and assume that it will be kept fixed, and only compute the relative rotation of the camera with respect to the road plane at each time instant.

Besides, we can also assume fixed translation of the camera coordinate system with respect to the world coordinate system, since the camera moves rigidly with respect to the vehicle, and compute it offline analogously to the camera calibration matrix. Nevertheless, if the translation is not available (e.g. we do not have access to the vehicle or just got the uncalibrated video sequences), we can still find rectified planes. The problem is that if we use an arbitrarily chosen value for $\mathbf{c}$, the resulting plane rectification will have an unknown anisotropic distortion: the worsen approximation we chose, the larger the distortion will grow.

This way, our target can be defined as to obtain the value of the pitch, roll, and yaw angles at each time instant in order to construct the rotation matrix R and the camera projection matrix P, as defined in (5.8), given that the camera calibration matrix K is available, and finally obtain the homography H as in (5.12).

There are, though, a number of variants to this problem, that we describe in the following paragraphs.

5.3.2.1 Null roll angle

If the camera is installed inside the vehicle with null roll angle, the problem is significantly simplified. This assumption holds if the camera is carefully installed without

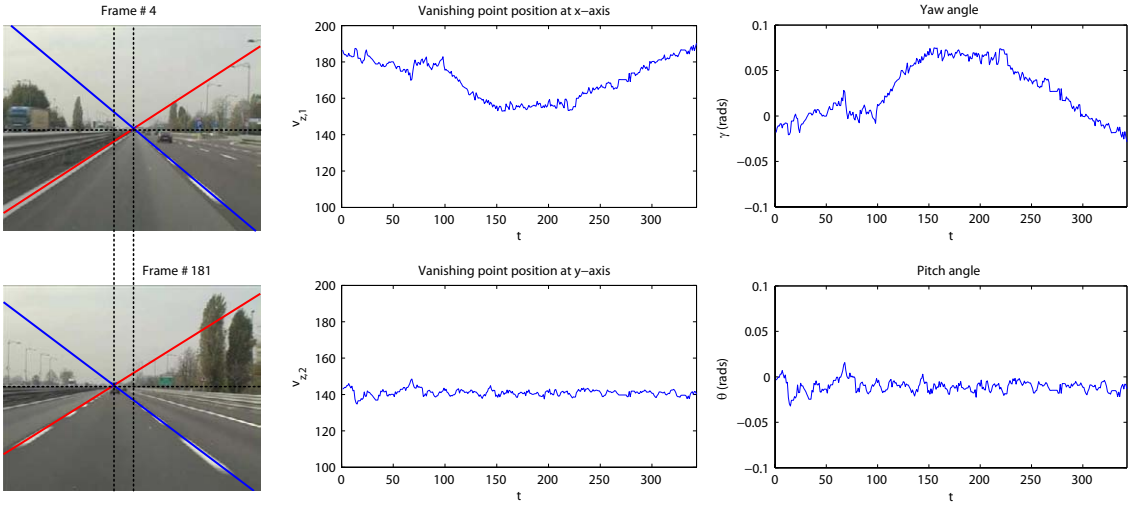


Figure 5.6: Example of variation of the pitch and yaw angle in a lane change manoeuver. The vanishing point is shown as the intersection of two colored lines for a better visualization.

rotation with respect the to Z-axis. An early version of the following method is described in [87] and used in [88].

This assumption implies that the line at the infinity of the road plane is exactly horizontal. Hence, if we determine its position in the image (defining its coordinate in the y-axis), we have recovered the affine properties, e.g. parallelism or area ratios, as well as the angular information, provided that we know the camera calibration matrix.

This way it is enough to compute the vanishing point associated to the lane markings of the road, $\mathbf{v}_z = (v_{z,1}, v_{z,2}, 1)^\top$, which belong to the line at the infinity and thus define it completely. We can do it with the specific methods described in chapters 3 and 4, using the EM algorithm and the Bayesian classifier to select the information of the image related to the lane markings.

The vanishing point is calibrated as $\mathbf{v}'_z = K^{-1}\mathbf{v}_z$ to project it into the camera coordinate system, as described in appendix B, so that relative angles θ and ϕ can be directly computed as

$$\theta = \arctan(v'_{z,2}) \tag{5.14}$$

$$\gamma = \arctan\left(-\frac{v'_{z,1}}{\cos\theta}\right) \tag{5.15}$$

These expressions come from the following argument: consider the point at the infinity corresponding to the vanishing point $\mathbf{v}_z$, and its projection into the image

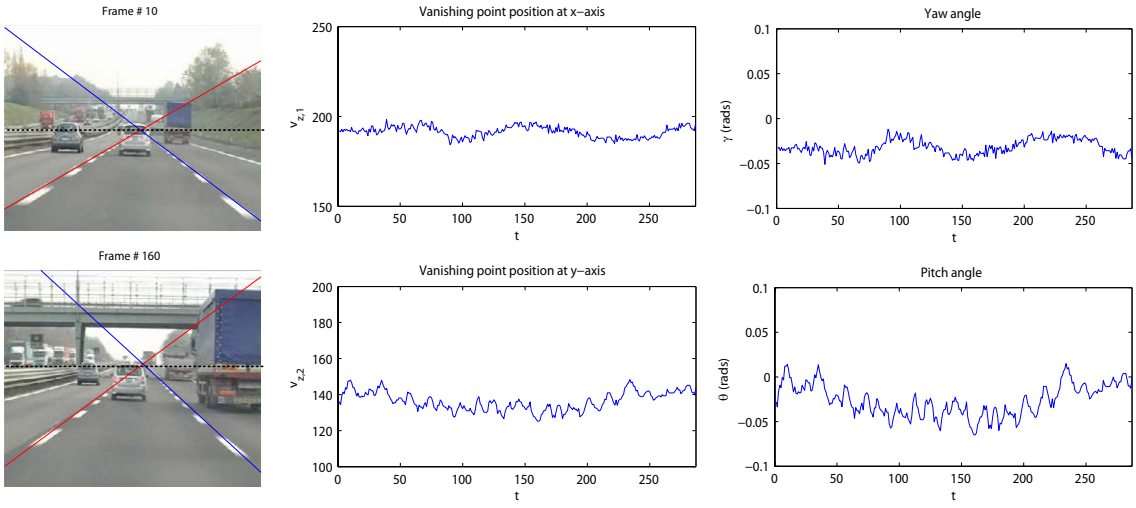


Figure 5.7: Example of variation of the pitch and yaw angle in a road slope change. The oscillation of the curves is due to the inherent instability of the motion of the camera.

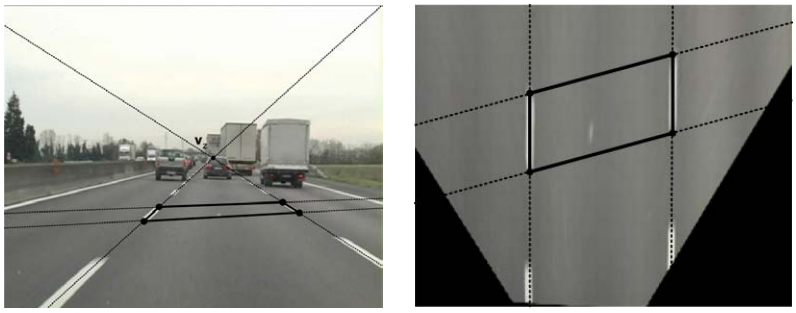
plane as:

$$\mathbf{v}_z = K(R|-R\mathbf{c}) \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = K \begin{pmatrix} \tan\gamma cos\theta \\ \tan\theta \\ 1 \end{pmatrix} \tag{5.16}$$

so that if we consider that we know the camera calibration matrix and then we can calibrate $\mathbf{v}_z$ as $\mathbf{v}'_z = K^{-1}\mathbf{v}_z$, then equation (5.16) give us two equations on the two unknowns θ and γ, which are worked out as in (5.14) and (5.15), respectively.

For a better understanding of these relationships, we can consider the following examples: if the roll angle is null, and the pitch angle is kept fixed, but the yaw angle varies (i.e. for instance when the vehicle steers), the coordinate in the X-axis of the vanishing points changes as $v'_{z,1} = \tan\gamma\cos\theta$, such that it moves in the image reflecting the movement of the vehicle. On the other hand, if the yaw angle is fixed but the pitch angle changes, the vanishing point changes its two coordinates as $v'_{z,1} = \tan\gamma\cos\theta$ and $v'_{z,2} = \tan\theta$. In the typical case that the yaw angle is not only static, but null, then $\tan\gamma = 0$ and the only changing coordinate is $v'_{z,2} = \tan\theta$, which means that the vanishing point moves vertically in the Y-axis of the image, but not horizontally (in the X-axis).

Figure 5.6 and figure 5.7 show two example sequences representing the abovementioned situations. The former depicts a lane change manoeuver, where the vehicle is steering to the right and makes two consecutive lane changes. The vanishing point



(a) (b)

Figure 5.8: Example of detection of the quadrilateral formed by two opposite lane markings strokes in the original image (a), and the rhomboid in the rectified image (b).

is detected using the EM algorithm described in chapter 3 for the image plane. As shown, the height of the vanishing point, $v'_{z,2}$ is almost constant, apart from some detection noise, so is the pitch angle. The transversal position of the vanishing point, $v'_{z,1}$ changes, which corresponds to a significant change of the yaw angle, which is maximum when the vehicle is at maximum steer, and returns to its initial values as the vehicle stabilizes its position in the final lane. The second example, shown in figure 5.7, although less pronounced than the previous one, illustrates the behavior of the vanishing point when a significant road slope appears. As explained before, the pitch angle is the one varies more significantly, following the movement of the vertical component of the vanishing point, $v'_{z,2}$. The yaw angle does not vary so much, since although it depends on the variation of the pitch angle, its influence is highly attenuated by the arctangent expression shown in equation (5.15). As a result, we obtain rectified images of planes which show parallel lane markings, with constant width, even in difficult situations where the described extrinsic parameters vary.

5.3.2.2 Unknown rotation

If none of the three rotation angles, pitch, yaw and roll are known, it is still possible to rectify the road plane by obtaining the line at the infinity of the road plane to remove the projective distortion. The angular information is provided by the camera calibration matrix.

The line at the infinity can be computed in several ways, but we propose two methods for the road scenario based on the computation of vanishing points: (i) we can compute two vanishing points, such that $\mathbf{l}_\infty = \mathbf{v}_1 \times \mathbf{v}_2$; or (ii) obtain the line at the infinity from the main vanishing point $\mathbf{v}_z$ and three equally spaced support lines.

As we have seen along this thesis, the main vanishing point of the road scenario, $\mathbf{v}_z$

can be obtained in most situations. However, it is not always possible to find reliable information about a second vanishing point of the road plane. Nevertheless, there are some situations in which we can obtain this information. This is a possibility that arises when there are discontinuous lane markings painted on the road[6], as in the example of figure 5.8. In this example, we can see that the lane markings strokes are not aligned to the X-axis and thus do not form a rectangle in the rectified domain. In many situations we have observed that they form instead a rhomboid[7], as can be seen in figure 5.8 (b). In (a) we see that the image of this rhomboid is a generic quadrilateral.

To obtain this quadrilateral from images we use the proposed Bayesian classifier, although any other lane markings detector could be applied. Then, we fit two line segments to each stroke, which can be done using the SSWMS line segment detector described in chapter 2[8].

This way we have obtained two vanishing points that are used to obtain the line at the infinity. Nevertheless, in case we can not detect these strokes, or the road is painted with continuous lane markings, there is a second alternative to proceed, which has been published in [84]. Schaffalitzky and Zisserman [98] proposed a method to compute the line at the infinity from a single vanishing point and the image of a set of three parallel lines that are equally spaced in the plane.

The expression of the line at the infinity is:

$$\mathbf{l}_\infty = \left((\mathbf{l}_0 \times \mathbf{l}_2)^\top (\mathbf{l}_1 \times \mathbf{l}_2)\right) + 2\left((\mathbf{l}_0 \times \mathbf{l}_1)^\top (\mathbf{l}_2 \times \mathbf{l}_1)\right) \tag{5.17}$$

where $\mathbf{l}_0$, $\mathbf{l}_1$ and $\mathbf{l}_2$ are the imaged lines of the three equally spaced lines.

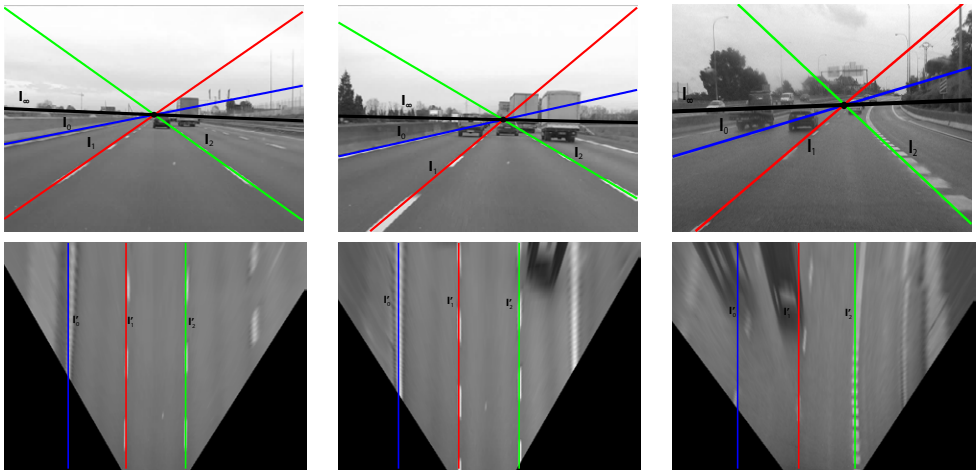
This method can be applied easily to the road scenario: we can detect three lines corresponding to three lane markings of two lanes, which meet at the vanishing point $\mathbf{v}_z$ and also satisfy that are equally spaced in the plane since lanes can be assumed to have the same width.

In summary, using any of the two proposed methods, we can compute the line at the infinity in order to recover the three rotation angles that define the rotation matrix of the camera with respect to the road plane. Particularly, provided that we know the camera calibration matrix, the computation of these angles is straightforward. We first obtain the value of the pitch and yaw angles, which are independent of the roll angle, using equations (5.14) and (5.15).

[6] In our experience, most of motorways in Madrid, Milano and Torino are painted in discontinuous lane markings, satisfying our hypothesis when there are at least three lanes in the road (such that the central one have discontinuous lane markings at left and right).

[7] Since these opposite strokes are actually aligned to the Z-axis, and of equal length, as well as the width of the lane they form is constant.

[8] An early version of this procedure is included in a recent publication [9].



(a) (b) (c)

Figure 5.9: Computation of $\mathbf{l}_\infty$ from the knowledge of the image of three equally spaced parallel lines in the road plane.

This way, if we define the vanishing points of the plane $\mathbf{V}_z = (0,0,1,0)^\top$ and $\mathbf{V}_x = (1,0,0,0)^\top$ in the 3D space, they must satisfy $\mathbf{l}'_\infty = \mathbf{v}'_x \times \mathbf{v}'_z$ in the camera coordinate system. Realize that we have to remove the perspective distortion by firstly calibrate the points and lines of the image as explained before.

After some operations, we arrive to the following expression:

$$\mathbf{l}'_\infty = ((R|-R\mathbf{c})\,\mathbf{V}_x) \times ((R|-R\mathbf{c})\,\mathbf{V}_z) = \begin{pmatrix} -\cos\gamma\sin\beta \\ -\cos\theta\cos\beta - \sin\theta\sin\gamma\sin\beta \\ \sin\theta\cos\beta - \cos\theta\sin\gamma\sin\beta \end{pmatrix} \tag{5.18}$$

The first element of the equality yields $l'_{\infty,1} = -\cos\gamma\sin\gamma$, such that we can work out β as:

$$\beta = \arcsin\left(-\frac{l'_{\infty,1}}{\cos\gamma}\right) \tag{5.19}$$

which is the last rotation angle needed to build the rotation matrix and to be able to generate the required homography from the projective matrix.

5.3.2.3 Unknown translation

If the relative translation between the coordinate systems, $\mathbf{c}$, is unknown, we propose a different approach for the computation of the rectified image. Since we would obtain

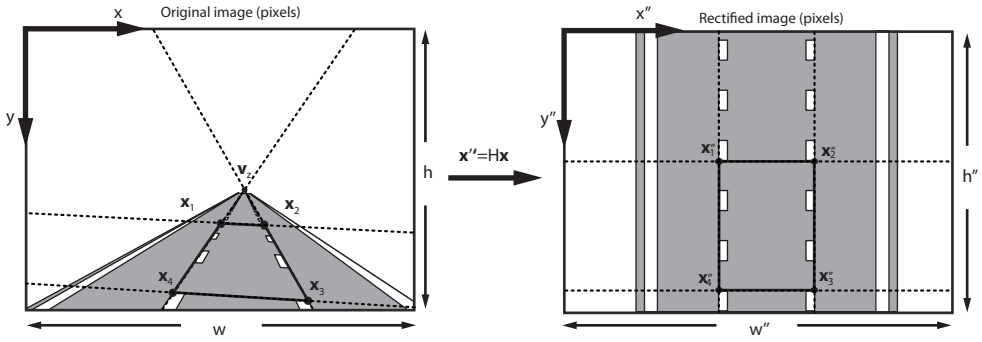


Figure 5.10: Computation of the homography between images from the correspondence of four points.

distorted images setting an arbitrary $\hat{\mathbf{c}}$, it is not worthy to compute H from the columns of P and then select the output range with unknown distortion as explained in previous cases.

It is much simpler to assume that we will not recover the anisotropic scaling and just work with the actually recovered geometric properties (basically, parallelism and angles), which are enough for tasks such as road modeling or vehicle detection. In that sense, we propose to use the obtained vanishing points, as described in previous subsections, that can be used to automatically select the regions of interest of the road plane, and directly compute the homography that transforms from $\mathbf{x}$ to $\mathbf{x}''$. Figure 5.10 depicts the idea. Provided two vanishing points detected on the image[9], we can draw two lines passing through each vanishing point such that we obtain four points on the image, $\{\mathbf{x}_i\}_{i=1}^4$.

The selection of these lines can be done with any arbitrary criterium. For instance, in the example figure, we have selected the lines such that they coincide with the lane markings. These points, as a result of the intersection of lines meeting at mutually orthogonal vanishing points, form a rectangle in the road plane (with unknown aspect ratio). Hence, we can choose the position of the points $\{\mathbf{x}_i''\}_{i=1}^4$ as a rectangle in the rectified image according to the desired width and height. The homography, H, can be obtained with any of the methods described in section 5.1.1, such as the DLT algorithm.

This method is very practical and it is recommended for applications that needs not to recover distances, and can work with affine-distorted versions of the road plane.

[9]That can be computed with any of the methods described in section 5.3.2.2.

5.3.3 Unknown extrinsic and intrinsic parameters

This last section is devoted to analyze the situation in which none of the parameters that govern the projection process are known. This is a difficult situation, since now we would have also to estimate the camera calibration matrix, K, to recover the angular information.

Although autocalibration (i.e. the automatic computation of K) is not an objective of this thesis, we present here some alternatives that we have evaluated in the road scenario, that leads to the autocalibration of the camera through the use of the information contained in the vanishing points.

5.3.3.1 Three mutually orthogonal vanishing points

This subsection introduces some concepts of camera autocalibration from vanishing points as they are commonly used in the related literature [68, 93].

It has been shown that three mutually orthogonal vanishing points leads to a number of equations that can solve for the parameters of the camera calibration, defined to have zero skew and square pixels [68]. As already mentioned in section 5.2.1.1, the cosine of the angle between two directions is given by $\mathbf{v}_1^\top \omega \mathbf{v}_2$ (if both directions are normalized), where $\omega = K^{-\top}K^{-1}$ is the image of the absolute conic. If the directions are orthogonal, they provide a equation on the elements of K:

$$\mathbf{v}_1^\top \omega \mathbf{v}_2 = 0 \tag{5.20}$$

If we have now three directions that are mutually orthogonal, we have three such equations:

$$\begin{aligned} \mathbf{v}_1^\top \omega \mathbf{v}_2 &= 0 \\ \mathbf{v}_1^\top \omega \mathbf{v}_3 &= 0 \\ \mathbf{v}_2^\top \omega \mathbf{v}_3 &= 0 \end{aligned}$$

Besides, the circular points introduce the required equations to complete the system:

$$\begin{aligned} \mathbf{I}^\top \omega \mathbf{I} &= 0 \\ \mathbf{J}^\top \omega \mathbf{J} &= 0 \end{aligned}$$

This way, there are five equations and five degrees of freedom of the IAC, so that it can be computed linearly with singular value decomposition. Finally, the obtention

of the camera calibration matrix is achieved by means of the Cholesky decomposition [68].

For the road scenario this method is not applicable. In general, we will have not enough information about three directions of the space, and no chance to compute three mutually orthogonal directions in such a dynamic scenario. Nevertheless, the concepts used by this method can be easily exported into an adapted autocalibration method for road scenarios, which is described in next subsection.

5.3.3.2 Tracking a pair of mutually orthogonal vanishing points

We have discussed in this chapter about methods to compute vanishing points, and we have seen that in some cases we can obtain both $\mathbf{v}_z$ and $\mathbf{v}_x$. We will operate under the assumption that the computed $\mathbf{v}_x$ (by means of the discontinuous lane marking strokes) is actually orthogonal to the main direction of the lane markings, $\mathbf{v}_z$. Realize that in some situations this hypothesis is not satisfied, as in the example image of figure 5.8, where the strokes are not correctly aligned.

Provided this hypothesis, we can obtain only one equation on the elements of the IAC, which is insufficient to obtain a solution. Nevertheless, we can take advantage of the temporal coherence of the video sequences to track the position of these vanishing points through time and obtain at least the three required equations:

$$\mathbf{v}_{z,t_1}^{\top}\omega\mathbf{v}_{x,t_1} = 0 \tag{5.21}$$

$$\mathbf{v}_{z,t_2}^{\top}\omega\mathbf{v}_{x,t_2} = 0 \tag{5.22}$$

$$\mathbf{v}_{z,t_3}^{\top}\omega\mathbf{v}_{x,t_3} = 0 \tag{5.23}$$

where t_1, t_2 and t_3 represents the time index for three images of the video sequence in which we have computed the two vanishing points. We have to pay attention here to the selection of these frames. It is not valid to select three consecutive frames with no variation of the yaw angle, for instance. We should select the more different situations that we find for a better numerical accuracy. For instance, a good opportunity arises when the vehicle steers, since both vanishing points are modified quite significantly.

Nevertheless, the abovementioned method is very prone to errors and we formulate it just as a theoretical contribution, but with many practical problems. In particular, it is very dependant on the presence of well aligned lane marking strokes, their correct computation, and on the accurate detection of vanishing points. This last point is, for this scenario and for some cases, not possible due to the absence of reliable information.

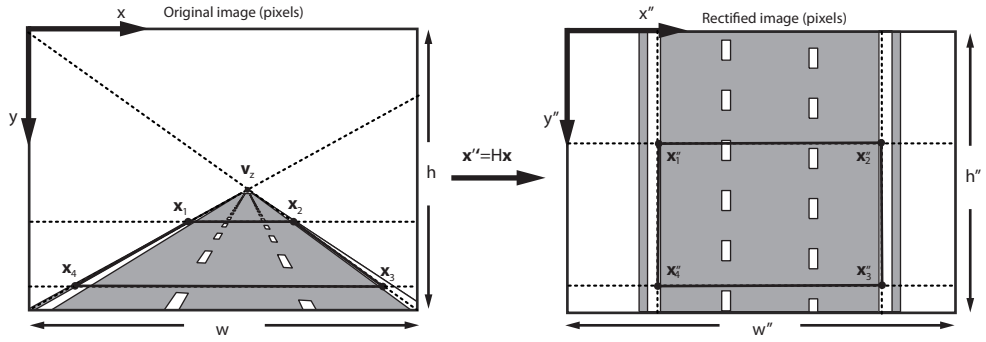


Figure 5.11: Computation of the homography between images from the correspondence of four points, with a resulting affine distortion.

5.3.3.3 Without computing the camera calibration matrix

We can think on a final case, in which the camera calibration is neither available nor computable, but we need to compute a plane rectification. We can do it, though, since the angular information is not recovered, the rectified image will have an unknown affine distortion.

The method just requires the computation of the main vanishing point of the road, $\mathbf{v}_z$. Then, we can select four points of the image as depicted in figure 5.11. As shown, these points are selected as if the actual position of the vanishing point $\mathbf{v}_x$ is at the infinity. Hence, the affine distortion is only correctly removed when this hypothesis is verified, i.e. when the yaw angle is zero[10]. The selection of these points could be done by joining the vanishing point with the lower corners of the images and selecting the points of intersection of those lines with two horizontal lines at different heights.

The example of figure 5.11 illustrates a situation for which the yaw angle is not zero. The result is a rectified image which show the road plane with a minor affine distortion (in real examples, this is almost visually unnoticeable).

For this reason, this approach could still be used when we lack the required information to perform better estimations, or when the quality of the images avoids the computation of additional elements, such as the line at the infinity or the second vanishing point, $\mathbf{v}_x$. This could be the case for ADAS applications that use video systems with small and cheap cameras without the required quality or image size, as it is described in [89].

[10]Therefore, even when the initial yaw is actually zero, if the vehicle steers, the yaw angle changes and the hypothesis is not satisfied.

Case	Prior knowledge			Elements to obtain	Procedure	Distortion removed
1	K	$\mathbf{c}$	(θ, γ, β)	-	-Reduce P to H as in (5.12) -Select range and get H -Apply $\mathbf{x}'' = H'H\mathbf{x}$	Projective Affine Similarity
2.1	K	$\mathbf{c}$	$\beta = 0$	$\mathbf{v}_z$	-Calibrate to $\mathbf{v}'_z$ -Obtain (θ, γ) as in (5.14) and (5.15) -Build P and proceed as case 1	Same as case 1
2.2	K		$\beta = 0$	$\mathbf{v}_z$	-Obtain $\mathbf{l}_\infty$ with $\mathbf{v}_z$ and $\beta = 0$ -Calibrate and obtain $\mathbf{v}'_x = \mathbf{l}'_\infty \times \mathbf{v}'_z$ -Select region of interest with 4 points	Projective Affine Unknown scale
2.3	K		$\beta = 0$	$\mathbf{v}_z$ and $\mathbf{c}$	-Proceed as in 2.1	Same as case 2.2
3.1	K	$\mathbf{c}$		$\mathbf{v}_z$ and $\mathbf{v}_x$	-Calibrate and obtain $\mathbf{l}'_\infty = \mathbf{v}'_z \times \mathbf{v}'_x$ -Obtain (θ, γ, β) as in (5.14), (5.15) and (5.19) -Select range as case 1	Same as case 1
3.2	K	$\mathbf{c}$		$\mathbf{v}_z$ and $\mathbf{l}_\infty$	-Calibrate and get $\mathbf{v}'_x = \mathbf{l}'_\infty \times \mathbf{v}'_z$ -Obtain (θ, γ, β) as in (5.14), (5.15) and (5.19) -Select range as case 1	Same as case 1
4.1	K			$\mathbf{v}_z$ and $\mathbf{v}_x$	-Obtain 4 points from $\mathbf{v}_z$ and $\mathbf{v}_x$ as in case 2.2	Same as case 2.2
4.2	K			$\mathbf{v}_z$ and $\mathbf{l}_\infty$	-Calibrate and get $\mathbf{v}'_x = \mathbf{l}'_\infty \times \mathbf{v}'_z$ -Obtain 4 points from $\mathbf{v}_z$ and $\mathbf{v}_x$ as in case 2.2	Same as case 2.2
5.1				$\{\mathbf{v}_{z,t}, \mathbf{v}_{x,t}\}_{i=1}^{3}$	-Obtain IAC -Cholesky decomposition to get K -Proceed as in case 4.1	Same as case 2.2
6.1				$\mathbf{v}_z$	-Select 4 points and obtain H	Projective Affine if $\gamma = \beta = 0$ Unknown scale

Table 5.1: Summary of plane rectification cases.

5.4 Summary and conclusions

Along this chapter, and specifically in this last section, we have introduced a number of alternatives for the computation of rectified images of the road plane. As we have seen, these methods directly depend on the amount of available information, and the results are not equal for all methods, since not all the geometric properties of the elements of the plane can always be retrieved.

As a summary, we present in table 5.1 an scheme of the different proposed methods, the type of information that is assumed to be known, the elements that must be computed and the obtained results.

In our experience in the road scenario, we have seen that the methods that find the best trade-off between the amount of prior knowledge and the reliability of the required detections are the ones denoted as case 3.2 and case 4.2. Both methods assume that the camera calibration matrix is known, though the rotation angles are all unknown. Case 3.2 has been explained as the method that recovers all the geometric properties of interest of the plane, namely removes the projective, affine and similarity distortion. The required elements are just the main vanishing point of the lane markings, $\mathbf{v}_z$ and the line at the infinity of the road plane. We have presented methods to reliably compute these variables, provided that there are at least two lanes in the road and that they have the same width.

Nevertheless, in the author's opinion, the case number 4.1 provides more flexibility, since it works under the assumption that the relative translation, $\mathbf{c}$, of the camera with respect to the road plane is unknown. This might be the case in which we had to work with video sequences not recorded by ourselves, such that we have had no access to the vehicle and we can not know the value of $\mathbf{c}$. As mentioned in the corresponding section, this lack of information implies that the similarity distortion can not be retrieved, and we can not obtain measurements of distances on the plane. However, we have used these rectifications successfully for different ADAS applications that do not target metric rectification but just the ability to generate relative measurements, such as the transversal position of the vehicle in the own lane, the relative speed of the vehicles ahead, etc.

Regarding the rest of cases, case 1 is quite unrealistic for dynamic scenarios, although we have included it here as a valid option for static environments, and also because some authors working on dynamic scenarios use it as a coarse approximation to the plane rectification. Cases 2.1 and 2.2 are almost as flexible as case 3.2 and 4.1, since the only added assumption is that the roll angle must be zero, which is quite typically accomplished in this scenario. In the absence of the value of the translation $\mathbf{c}$, two options arise: one of them (case 2.3) is just to guess a value that seems to be correct (typically, for a tourism vehicle the camera is about 1.2 meters height), and accept that there will be a small unknown similarity distortion, i.e. measurements of distances will not be metric, although only distorted in scale. The second option is depicted in case 2.2, which does not estimate nor guesses $\mathbf{c}$ and instead compute the homography as the result of selecting four points correspondences with the consequently unknown scale problem.

Case 4.1 is similar to case 4.2, although the presence of perfectly aligned lane markings strokes to compute $\mathbf{v}_x$ is an hypothesis not reliable in most situations. We have seen that many roads have, indeed, discontinuous lane markings, though not aligned in many cases, such that the application of the method corresponding to case 4.2 is prone to generate affine distortions.

Case 5.1 is the most complex of the proposed methods. In particular, it assumes

that we are able to compute pairs of orthogonal vanishing points of the road plane in different time instants, different enough to obtain accurate measurement of the camera calibration matrix. This situation is more than doubtful to happen, so that we just formulate it as a theoretical approach. Nevertheless, in more constrained scenarios, such as urban environments.

The last case, 6.1, has been included at the end not for its complexity, but, on the contrary, for its simplicity. Namely, if for some reason we do not have any information about the scenario: e.g. we have not the camera calibration matrix, we do not known the relative translation and we are not able to compute reliably nothing else than the main vanishing point of the road, we can still produce some nice results. In particular, provided the accurate detection of $\mathbf{v}_z$, we can select four fixed points of the image and retrieve a homography matrix that leads to the road plane rectification. We know that under these assumptions, the only recovered properties are those related to the projective distortion (basically, the parallelism). But, if the roll angle is zero, and the yaw angle is also zero (which is true if the camera is installed exactly forward looking, and the vehicle is not steering), the affine distortion is also recovered. This alternative could be useful for different ADAS applications (LDW, time to contact, etc.) if these assumptions are approximately satisfied so that the resulting distortion is limited.

Chapter 6

Conclusions and future work

A number of strategies and algorithms have been devised and described in this thesis for the detection and tracking of vanishing points and their application to the rectification of planes. The road scenario has been treated with special interest, and particular algorithms have been devised for it.

The following paragraphs comment the aspects that are believed to be novel, and discuss about the obtained results, pointing out the achieved enhancements over related methods, and also addressing their limitations. This discussion guides the future work section, in which potential evolutions of the presented work are summarized, as well as alternative methods for those presented along this thesis and other applications related to vanishing point estimation.

6.1 Conclusions

The first contribution, described in chapter 2, is related to the extraction of image features that can be used to compute vanishing points. The proposed SSWMS algorithm has been designed as an efficient alternative to existing methods for line segment detection. This approach represents a trade-off between computational efficiency and robustness. The former characteristic is always interesting for practical applications that work online, and especially for vision-based ADAS. The latter is the most challenging, as it means that the system is able to provide accurate results even with clutter and noise. Besides, the SSWMS algorithm does not require to tune any input parameter that affects the performance of the detections (although we have studied the influence of some of them in the results). The combination of these properties makes this approach a great choice for real applications that require online robust detections, like the one referred to the road scenario described in this thesis. A complete set of tests compare the performance of this approach with that of two of the most

outstanding works in the literature, the PPHT [40], and the LSD [119].

Chapter 3 is the core of the thesis, as it gathers the contributions in the field of vanishing point detection [83, 84, 85]. A number of methods have been proposed that determine the position of vanishing points in images through the use of the information contained in the image features, like the line segments provided by the SSWMS algorithm. The research contributions of this chapter can be described in two groups. On the one hand, robust estimation methods have been introduced, based on variants of the RANSAC algorithm. For these methods, an error function has been defined, which is an alternative to others typically used in the related literature (especially, the point-line distance in the calibrated sphere [65], and the functions defined on the end-points of line segments, which require non-linear optimization methods [68, 112]). This function provides narrower error maps on the projective plane, thus requiring less number of iterations in non-linear optimization procedures, and has been shown to provide excellent results in combination within the robust framework [83, 85].

On the other hand, two refinement alternatives have been proposed, which use the RANSAC-based algorithms as initialization and obtain enhanced accuracy in the detection of vanishing points. These methods, which are based on the EM algorithm, are novel contributions to the simultaneous estimation of vanishing points and support lines that pass through them [84]. These solutions have been defined, respectively, in the image plane, and in the projective plane. The former can be used for the detection of vanishing points which are inside the limits of the image. This approach has been constructed over the work by Minagawa et al. [76], with specific modifications that makes it better in different aspects (mainly speed, and robustness). The method on the projective plane provides excellent results for more general problems, since it treats equally all vanishing points, even if they are at the infinity. This method represents one of the most relevant contributions of this thesis, since, up to our knowledge, no other work in the literature seems to solve the simultaneous estimation of multiple vanishing points with their supporting lines in the projective plane. The use of support lines in the EM framework has been shown to be a positive feature, since it allows taking advantage of that in structured scenarios line segments tend to be clustered into significant lines, which are less noisy than sparse sets of line segments. This way, the EM algorithm automatically selects only the line segments that belong to these support lines and discard the rest of information, and hence achieving more accurate results.

The abovementioned methods have been applied in the complex road scenario, described in chapter 4, which is especially challenging in account of its dynamism and inherent variability. The detection of vanishing points in this scenario requires a number of additional considerations [82]. On the one hand, there is a lack of reliable information related to the vanishing points of the scene, so that a pre-processing stage is needed. The proposed method is based on a Bayesian classifier that works at pixel

level, for which different classes are defined [6, 88]. One of them corresponds to the lane markings of the road, which are the main actors for the detection of vanishing points [9, 7, 89, 87]. The classifier combines different image features, to provide better classification results. As a result, this approach can be used as an enhanced lane markings detector[1], which has been shown to adapt its internal parameters according to the observations of the image, such that the dynamism of the scene is thus absorbed, providing excellent detection results in a wide variety of scenarios.

On the other hand, a road model is defined on the transformed domain, which is a bird's eye view of the road plane. It allows to exploit the temporal coherence of the sequence in order to track the elements of interest of the road [8, 90]. Namely, the own lane is tracked with a Kalman filter that estimates the position of the left and right-hand lane markings of the lane through time. Hence, it serves as initialization for the application of the EM method proposed at the end of chapter 3. In the transformed domain, lanes are modeled as pairs of straight parallel lines, which are the support lines of an infinite vanishing point. Once computed, the rectification of the road plane is corrected according to the vanishing point, in case it is not at the infinity, meaning that the transformation has some error. The coherence provided by the road model has been tested for a large number of road sequences, showing very high detection rates.

The rectification of planes has been addressed in chapter 5, which includes a complete analysis of the alternatives that exist for the computation of the road plane rectification according to the available information. Some of the proposed methods represent significant research contributions in the vision-based ADAS field, since typically the transformation that is computed is not well adapted to the changes of relative orientation between the camera and the road plane. For instance, the one described as case 3.2 in section 5.4, restores the Euclidean properties of the road plane by computing the main vanishing point of the scene together with three lane markings [84]. This is achieved using the EM algorithm on the image plane and the obtention of the line at the infinity using the method by Schaffalitzky and Zisserman [98].

6.2 Future work

Since several topics have been treated along this thesis, we can pose now a number of future lines for each of them, corresponding to aspects that have not been solved or that require more analysis to improve the performance of the proposed strategies. Besides, new application fields are suggested for the devised methods.

[1] Although, as part of a complete vision-based ADAS, it also addresses the detection of vehicles and pavement areas of the image, for other applications, like collision prevention or detection of multiple lanes.

Feature extraction

The results obtained with the proposed SSWMS algorithm applied to the detection of vanishing points are satisfactory. Nevertheless, as a more ambitious target, it is possible to go further and obtain more complex image features. For instance, we could define as point-wise image feature the intersection between straight lines, as a subcase of corners. This type of feature could be of great interest for the estimation of vanishing points and even for estimation of planes as intersection of sets of parallel lines.

The SSWMS can be extended in this line, combining the search for straight lines and corners, and through the design of a more complex CCA algorithm with Mean Shift procedures that allows growing from a starting point to more than one direction.

Vanishing point estimation

Although several methods have been proposed for vanishing point estimation in this thesis, there are a few topics that have been not fully covered and thus can be of interest for future research lines.

The proposed methods are basically constructed over optimization procedures, such as RANSAC and EM. Nevertheless, other alternatives, as described in section 3.2, can also be very effective, like those based on the Hough transform. Particularly, the work by Almansa et al. [2] is a great probabilistic approach using Hough-like accumulators. In this line, it is also possible to explore the potentials of non-linear optimization process working on a optimal quantized space proposed by Almansa et al.

In the same line, recent works on optimization procedures have shown that there are methods that perform better than RANSAC. In particular the work by Subbarao and Meer [110], which is somehow related to non-parametric estimation and Mean Shift procedures, can be also investigated for robust vanishing point estimation.

Another possible evolution is the inclusion of prior knowledge of the relationship between vanishing points. For instance, particle filters with MRF (Markow Random Fields) as described by Khan et al. [57], is being analyzed. The MRF can be used to model the angular relationship between vanishing points. Three mutually orthogonal vanishing points can be then searched in an unified manner, which is more robust than searching for independent vanishing points.

Road modeling

The complexity of the road models can be increased, as done in some preliminary approaches [89, 90] with circumference models in the transformed domain. The vibration of the vehicle can also be better handled with non-linear dynamic models, e.g. the use of particles filters could be evaluated for this purposes, paying attention to the additional computational load that this framework generates.

All the works carried out related to road modeling have been developed using monocamera acquisition systems. Nevertheless, the use of multiple cameras opens a

new window of alternatives. For instance, if several cameras are installed inside the vehicle looking at the front, back and laterals of the vehicles, a new, richer transformed domain of the road could be built. More complex manoeuvres like overtakings could be handled. The management of multiple views is a problem itself and it could pose a number of additional problems in the described road scenario.

Camera autocalibration in dynamic environments

Among the presented cases for road plane rectification, we have included some of them that would require the exact computation of the camera calibration matrix. Typically, this is a tough problem in controlled scenarios, but much more in dynamic environments like the road scenario, in which the information is very limited and not reliable. For that reason it could be interesting to exploit the temporal coherence of the detected vanishing points and the motion of the vehicle. As the position of the vanishing point varies, more equations on the unknown parameters of the IAC are obtained. Provided that we are able to detect two vanishing points from the lane markigns quadrilateral, as shown in figure 5.8, along time, the relative angle between them in the road plane can be added as an unknown to the system of equations.

Methods and evaluation of this problem are an interesting research field that combines image processing in the road scenario and projective geometry concepts.

Appendix A

Weighted Mean Shift

MS works as a kernel-based non-parametric estimator of the pdf from which samples $\mathbf{x}_k$ are drawn. Let us consider a set of samples, $\{\mathbf{x}_i\}_{i=1}^{N}$, with dimension D, from an unknown density $f(\mathbf{x})$. The multivariate weighted kernel estimator of $f(\mathbf{x})$ is defined as:

$$\widehat{f}_{\omega}(\mathbf{x}) = \frac{1}{\sum_{i=1}^{N} \omega_i} \sum_{i=1}^{N} \omega_i K_H(\mathbf{x} - \mathbf{x_i}) \tag{A.1}$$

with the kernel defined as:

$$K_H(\mathbf{x}) = \frac{1}{|\mathrm{H}|^{\frac{1}{2}}} K\left(\frac{\mathbf{x}}{\mathrm{H}^{\frac{1}{2}}}\right) \tag{A.2}$$

where H is a symmetric positive definite $D \times D$ bandwidth matrix that specifies the "width" of the kernel at each dimension. As a fully parameterized H increases the complexity of the estimation [30], the bandwidth matrix H is chosen in this approach as a diagonal matrix containing the corresponding bandwidths for each dimension, with $D = 3$: $\mathrm{H} = diag(h_x^2, h_y^2, h_\theta^2)$. K is obtained from the product of symmetric univariate kernels:

$$K(\mathbf{x}) = \prod_{d=1}^{D} c_d k(x_d) \tag{A.3}$$

where x_d are each component of the vector $\mathbf{x}$, c_d are the normalization constants, and the function k is defined as the Epanechnikov kernel [30]. Substituting (A.3) into (A.2), and the result into (A.1) yields:

$$\widehat{f}_{\omega}(\mathbf{x}) = \frac{|\mathrm{H}|^{-\frac{1}{2}}C}{\sum_{i=1}^{N}\omega_i}\sum_{i=1}^{N}\omega_i\prod_{d=1}^{D}k\left(\left(\frac{x_d - x_{i,d}}{h_d}\right)^2\right) \tag{A.4}$$

where $x_{i,d}$ is the d-th component of data sample $\mathbf{x}_i$ and h_d is its corresponding bandwidth. Note that, as opposed to most approaches using MS, the described estimator considers not only a set of weighting factors for the kernels but also a multidimensional bandwidth matrix, which contains different bandwidth values, one for each dimension [45].

The modes in $f(\mathbf{x})$ that we are interested in, are located at the zeros of the gradient, $\nabla f(\mathbf{x})$. An estimator of the gradient of $f(\mathbf{x})$ is the gradient of $\widehat{f}_{\omega}(\mathbf{x})$, that can be represented as follows:

$$\nabla\widehat{f}_{\omega}(\mathbf{x}) = A\left[\sum_{i=1}^{N}\omega_i G(\mathbf{x}-\mathbf{x}_i)\right]\left[\left(\sum_{i=1}^{N}\omega_i G(\mathbf{x}-\mathbf{x}_i)\right)^{-1}\left(\sum_{i=1}^{N}\omega_i G(\mathbf{x}-\mathbf{x}_i)\mathbf{x}_i\right) - \mathbf{x}\right] \tag{A.5}$$

where A is a matrix gathering the constants and the bandwidth matrix:

$$A = \frac{2|\mathrm{H}|^{-\frac{1}{2}}C}{\sum_{i=1}^{N}\omega_i}H^{-1} \tag{A.6}$$

and $G(\mathbf{x})$ is a diagonal 3×3 matrix defined as:

$$G(\mathbf{x}) = diag\left(-k'\left(\frac{x^2}{h_x^2}\right), -k'\left(\frac{y^2}{h_y^2}\right), -k'\left(\frac{\theta^2}{h_\theta^2}\right)\right) \tag{A.7}$$

The function k' is obtained as the differentiation of the kernel k, as defined in (A.3).

The last bracket in (A.5) is the mean shift vector, which represents the difference between the weighted mean of the data samples and the center of the kernel:

$$m_w(\mathbf{x}) = \frac{\left(\sum_{i=1}^{N}\omega_i G(\mathbf{x}-\mathbf{x}_i)\mathbf{x}_i\right) - \mathbf{x}}{\sum_{i=1}^{N}\omega_i G(\mathbf{x}-\mathbf{x}_i)} \tag{A.8}$$

Appendix B

Unit sphere

Each line (or any image feature with an associated orientation, such as line segments or gradient-pixels) on the image plane define a plane that passes through the optical center of the camera model, denominated interpretation plane, which intersects a sphere centered as well at the optical center in the so-called "great circles". The intersections between these circles correspond to the intersection of the corresponding lines on the image plane. Figure B.1 depicts this concept.

An interesting property that comes out from these definitions is that a vanishing point is treated in this space as a 3D direction, instead of a 2D point on the image. The vanishing direction is then defined by the vector that joins the optical center and the point on the surface of the sphere which is the intersection of two or more great circles that correspond to lines meeting at a common vanishing point.

B.1 Data calibration and normalization

There is an important concept to clarify regarding the formulation of the estimation of vanishing points in the projective plane. Vanishing points are represented in homogeneous coordinates[1] as $\mathbf{v} = (v_1, v_2, v_3)^\top$. This representation is very useful since it allows to handle projective relationships with linear matrix operations. For instance, the projection of a point in the space, represented as well with homogeneous coordinates as $\mathbf{X} = (X, Y, Z, 1)^\top$, in the image plane is obtained as

$$\mathbf{x}_{\text{hom}} \;=\; K\,[I|\mathbf{0}]\,\mathbf{X} \tag{B.1}$$

where K is the camera calibration matrix. The image coordinates of this projected

[1] see [46] for details about these coordinates.

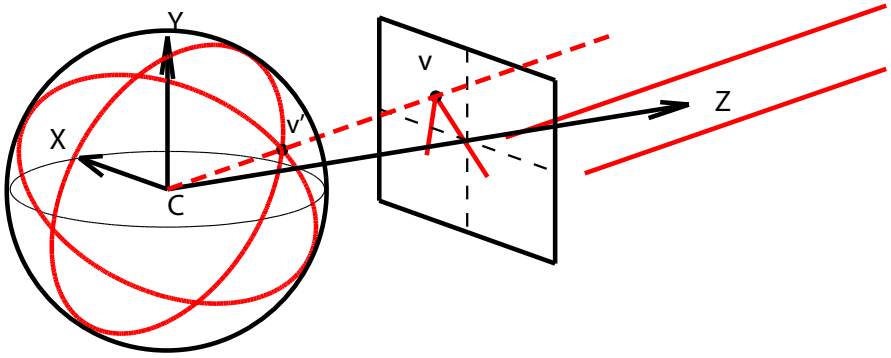


Figure B.1: Two parallel 3D lines are projected into the image plane converging into the vanishing point $\mathbf{v}$. Its calibrated and normalized version, $\mathbf{v}'$, can be seen as the ray joining the optical center, C as well as the intersection of the two great circles defined by the projected lines.

point are then obtained by dividing $\mathbf{x}_{\text{hom}}$ by its third coordinate, obtaining the inhomogeneous coordinates of the point in the image plane $\mathbf{x}_{\text{inh}} = \left(f\frac{X}{Z}, f\frac{Y}{Z}\right)^\top$.

One dimension is lost in the projection of a point in the space to the image plane, since any point lying in the ray that joins $\mathbf{X}$ with the optical center $\mathbf{C}$ is projected in the same image point [46]. From a geometric point of view, if the camera calibration matrix is known, the abovementioned ray can be retrieved from any 2D image point. This is of great help in the case of vanishing points since this ray actually correspond to the 3D direction of the parallel lines in the 3D world. Indeed, parallel lines, when projected into the image, converge into the vanishing point. This idea is illustrated in figure B.1.

To explain this concept, let us suppose that there are two parallel lines in the 3D world, whose common director vector is $\mathbf{D} = (X, Y, Z)^\top$. The intersection of these lines is at some point in the infinity, that we can represent in homogeneous coordinates as $\mathbf{V} = (X, Y, Z, 0)^\top$. This point is projected into the vanishing point $\mathbf{v} = K[I|\mathbf{0}]\mathbf{V}$. If we left-multiply both members of this expression by K^{-1} we obtain $K^{-1}\mathbf{v} = [I|\mathbf{0}]\mathbf{V} = (X, Y, Z)^\top$, which renders a new 3D vector $\mathbf{v}' = K^{-1}\mathbf{v}$. We will refer to this transformation as "data calibration" as other authors do [65]. Hence, the calibration of the vanishing points, in homogeneous coordinates yields the actual 3D direction of the parallel lines that converge to the vanishing point in their image projection. This transformation can also be understood as the process of augmenting the 2D coordinates of the image elements (points, lines) into the 3D coordinates under the camera coordinate system.

Data calibration is carried out in the following way: points are normalized by K^{-1} as $\mathbf{x}' = K^{-1}\mathbf{x}$ and lines by $K^\top$ as $\mathbf{l}'_i = K^\top\mathbf{l}_i$. Besides, to be able to represent this

elements on the unit sphere, an additional scale normalization must be done, such that the norm of the 3D vectors is the unity.

In the case that the camera calibration is not available, working with the homogeneous vector $\mathbf{v}$ entails an implicit distortion in the angular information. For instance, the cosine of the angle between two uncalibrated directions is dependent on an unknown distortion given by $\omega = K^{-\top}K^{-1}$:

$$\cos\alpha = \frac{\mathbf{v}_1'^{\top}\mathbf{v}_2'}{\|\mathbf{v}_1'\|\|\mathbf{v}_2'\|} = \frac{\mathbf{v}_1^{\top}\left(K^{-\top}K^{-1}\right)\mathbf{v}_2}{\sqrt{\mathbf{v}_1^{\top}\left(K^{-\top}K^{-1}\right)\mathbf{v}_1}\sqrt{\mathbf{v}_2^{\top}\left(K^{-\top}K^{-1}\right)\mathbf{v}_2}} = \frac{\mathbf{v}_1^{\top}\omega\mathbf{v}_2}{\sqrt{\mathbf{v}_1^{\top}\omega\mathbf{v}_1}\sqrt{\mathbf{v}_2^{\top}\omega\mathbf{v}_2}} \tag{B.2}$$

Nevertheless, the camera calibration matrix can be approximated easily by a default projective transformation based on the image dimensions, which reduces the abovementioned inherent distortion, as done by some authors [65], such as:

$$\tilde{K} = \begin{pmatrix} W & 0 & W/2 \\ 0 & H & H/2 \\ 0 & 0 & 1 \end{pmatrix} \tag{B.3}$$

where W and H are the width and height of the image in pixels, respectively.

Bibliography

[1] N. Aggarwal and W. C. Karl. Line detection in images through regularized Hough transform. *IEEE Transactions on Image Processing*, 15(3):582–591, 2006.

[2] A. Almansa, A. Desolneux, and S. Vamech. Vanishing point detection without any a priori information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4):502–507, 2003.

[3] D. Alonso, L. Salgado, and M. Nieto. Robust vehicle detection through multidimensional classification for on board video based. In *IEEE Proc. International Conference on Image Processing*, pages 321–324, 2007.

[4] M. E. Antone and S. Teller. Automatic recovery of relative camera rotations for urban scenes. In *Proc. Conf. Computer Vision and Pattern Recognition*, volume 2, pages 282–289, 2000.

[5] J. Arróspide, L. Salgado, M. Nieto, and F. Jaureguizar. On-board robust multiple vehicle detection and tracking using adaptive quality evaluation. In *IEEE Proc. International Conference on Image Processing*, pages 2008–2011, 2008.

[6] J. Arróspide, L. Salgado, M. Nieto, and F. Jaureguizar. Real-time vehicle detection and tracking based on perspective and non-perspective space cooperation. In *IS&T/SPIE Proc. International Conference on Real-Time Image and Video Processing*, pages 72440H–1–12, 2008.

[7] J. Arróspide, M. Nieto, L. Salgado, and R. Mohedano. Homography-based ground plane detection using a single on-board camera. *IET Intelligent Transportation Systems*, page (accepted), 2010.

[8] J. Arróspide, L. Salgado, and M. Nieto. Vehicle detection and tracking using homography-based plane rectification and particle filtering. In *IEEE Proc. International Conference on Image Processing*, page (accepted), 2010.

[9] J. Arróspide, L. Salgado, and M. Nieto. Vehicle detection and tracking using homography-based plane rectification and particle filtering. In *IEEE Proc. Intelligent Vehicles Symposium*, page (accepted), 2010.

[10] A. Bandera, J.M. Perez Lorenzo, J.P. Bandera, and F. Sandoval. Mean shift based clustering of Hough domain for fast line segment detection. *Pattern Recognition Letters*, 27(6):578–586, 2006.

[11] O. Barinova, A. Kuzmishkina, A. Vezhnevets, and V. Vezhnevets. Learning class specific edges for vanishing point estimation. In *Proc. of Graphicon*, pages 162–165, 2007.

[12] S. T. Barnard. Interpreting perspective images. *Artificial Intelligence Journal*, 21(4):435–462, 1983.

[13] T. Bücher. Measurement of distance and height in images based on easy attainable calibration parameters. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 314–319, 2000.

[14] M. Bertozzi and A. Broggi. GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing*, 7 (1):62–81, 1998.

[15] J. A. Bilmes. A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *Technical Report, Univ. of California, Berkeley*, 1998.

[16] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006.

[17] A. Borkar, M. Hayes, and M.T. Smith. Lane detection and tracking using a layered approach. In *Proc. Advanced Concepts for Intelligent Vision Systems*, pages 474–484, 2009.

[18] J. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30, 1965.

[19] B. Brillault-O'Mahony. New method for vanishing point detection. *Computer Vision, Graphics and Image Processing: Image Understanding*, 54(2):289–300, 1991.

[20] A. Broggi, M. Bertozzi, and A. Fascioli. Self-calibration of a stereo vision system for automotive applications. In *IEEE Proc. International Conference on Robotics and Automation*, volume 4, pages 3698–3703, 2001.

[21] P.G. Bryan, I. Corner, and D. Stevens. Digital rectification techniques for architectural and archaeological presentation. *The Photogrammetric Record*, 16(93): 399–415, 2003.

[22] J. B. Burns, A. R. Hanson, and E. M. Riseman. Extracting straight lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(4):425–455, 1986.

[23] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 8(6):679–698, 1986.

[24] V. Cantoni, L. Lombardi, M. Porta, and N. Sicard. Vanishing point detection: representation analysis and new approaches. In *Proc. International Conference on Image Analysis and Processing*, pages 26–28, 2001.

[25] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International Journal of Computer Vision*, (3):127–140, 1990.

[26] P. Cerri and P. Grisleri. Free space detection on highways using time correlation between stabilized sub-pixel precision IPM images. In *IEEE Proc. International Conference on Robotics and Automation*, pages 2223–2228, 2005.

[27] P. Chang, D. Hirvonen, T. Camus, and B. Southall. Stereo-based object detection, classification, and quantitative evaluation with automotive applications. In *IEEE Proc. Computer Vision and Pattern Recognition*, volume 3, pages 62–68, 2005.

[28] Y. Chen, M. Das, and D. Bajpai. Vehicle tracking and distance estimation based on multiple image features. *Proc. Canadian Conf. Computer and Robot Vision*, pages 371–378, 2007.

[29] R. Collins. Vanishing point calculation as statistical inference on the unit sphere. In *Proc. International Conference on Computer Vision*, pages 400–403, 1990.

[30] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24: 603–619, 2002.

[31] C. Corridori and M. Zanin. High curvature two-clothoid road model estimation. In *IEEE Proc. Intelligent Transportation Systems Conference*, pages 630–636, 2004.

[32] J. Coughlan and A. Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *Proc. International Conference on Computer Vision*, pages 941–947, 1999.

[33] A. Criminisi, I. Reid, and A. Zisserman. Single view metrology. *International Journal on Computer Vision*, 40(2):123–148, 2000.

[34] D. Dahyot. Statistical Hough transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(8):1502–1509, 2009.

[35] R. Danescu, S. Nedevschi, M.-M. Meinecke, and T.-B. To. A stereovision-based probabilistic lane tracker for difficult road scenarios. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 536–541, 2008.

[36] P. David. Detecting planar surfaces in outdoor urban environments. Technical Report AD-A488059; ARL-TR-4599, Army Research Lab., 2008.

[37] P. Denis, J. H. Elder, and F. J. Estrada. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *Proc. European Conference on Computer Vision*, volume 2, pages 197–210, 2008.

[38] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[39] A. Fusiello and L. Irsara. Quasi-euclidean uncalibrated epipolar rectification. In *Proc. International Conference on Pattern Recognition*, pages 1–4, 2008.

[40] C. Galambos, J. Kittler, and J. Matas. Progressive probabilistic Hough transform for line detection. *IEEE Proc. Computer Vision and Pattern Recognition*, 1:1554–1560, 1999.

[41] W. Gilks, S. Richardson, and D. Spiegelhalter. *Markov Chain Monte Carlo Methods in Practice*. Chapman and Hall/CRC, 1996.

[42] R. Goecke, N. Pettersson, and L. Petersson. Towards detection and tracking of on-road objects. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 416–421, 2007.

[43] J. P. González and U. Ozguner. Lane detection using histogram-based segmentation and decision tree. In *IEEE Proc. Intelligent Transportation Systems*, pages 346–351, 2000.

[44] D.S. Guru, B.H. Shekar, and P. Nagabhushan. A simple and robust line detection algorithm based on small eigenvalue analysis. *Pattern Recognition Letters*, 25 (1):1–13, 2004.

[45] B. Han, D. Comaniciu, Y. Zhu, and L. S. Davis. Sequential kernel density approximation and its application to real-time visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(7):1186–1197, 2008.

[46] R. I. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2004.

[47] Q. He and C.-H. Henry Chu. Lane detection and tracking through affine rectification. In *IAPR Proc. Conference on Machine Vision Applications*, pages 536–539, 2007.

[48] M. Heath, S. Sarkar, T. Sanocki, and K. Bowyer. Comparison of edge detectors: a methodology and initial study. *Computer vision and image understanding*, 69 (1):38–54, 1998.

[49] C. Hoffmann, T. Dang, and C. Stiller. Vehicle detection fusing 2d visual features. In *IEEE Proc. Intelligent Vehicle Symposium*, pages 280–285, 2004.

[50] J. Hu, S. You, and U. Neumann. Vanishing hull. In *3DPVT'06: Proc. Int. Symposium on 3D Data Processing, Visualization, and Transmission*, pages 448–455, 2006.

[51] P. Jeong and S. Nedevschi. Efficient and robust classification method using combined feature vector for lane detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(4):528–537, 2005.

[52] M. Jethwa, A. Zisserman, and A. Fitzgibbon. Real-time panoramic mosaics and augmented reality. In *Proc. British Machine Vision Conference*, pages 852–862, 1998.

[53] G. Y. Jiang, T.Y. Choi, S.K. Hong, J.W. Bae, and B.S. Song. Lane and obstacle detection based on fast inverse perspective mapping algorithm. In *IEEE Proc. International Conference on Systems, Man, and Cybernetics*, volume 4, pages 2969–2974, 2000.

[54] M. Kalantari, F. Jung, N. Paparoditis, and J.-P. Guedon. Robust and automatic vanishing points detection with their uncertainties from a single uncalibrated image, by planes extraction on the unit sphere. In *Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume 37-3A, pages 203–208, 2008.

[55] K. Kanatani. *Statistical optimization for geometric computation.* Elsevier, 1996.

[56] P. Khan, L. Kitchen, and E.M. Riseman. A fast line finder for vision-guided robot navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(11):1098–1102, 1990.

[57] Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1819, 2005.

[58] Z. Kim. Robust lane detection and tracking in challenging scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 9(1):16–26, 2008.

[59] N. Kiryati, Y. Eldar, and A. M. Bruckstein. A probabilistic Hough transform. *Pattern Recognition*, 24(4):303–316, 1991.

[60] J. Klappstein, F. Stein, and U. Franke. Applying kalman filtering to road homography estimation. In *Proc. ICRA 2007 Workshop: Planning, Perception and Navigation for Intelligent Vehicles*, 2007.

[61] K. Kluge. Extracting road curvature and orientation from image edge points without perceptual grouping into features. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 109–114, 1994.

[62] K. Kluge and S. Lakshmanan. A deformable-template approach to lane detection. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 54–59, 1995.

[63] H. Kälviäinen, P. Hirvonen, L. Xu, and E. Oja. Comparisons of probabilistic and non-probabilistic Hough transforms. In *Proc. of 3rd European Conference on Computer Vision (ECCV)*, pages 351–360, 1994.

[64] M. Kotb and S. Beauchemin. Generalizing inverse perspective. In *Proc. of the 2nd Canadian conference on Computer and Robot Vision*, pages 521–527, 2005.

[65] J. Košecká and W. Zhang. Video compass. In *Proc. European Conference on Computer Vision, LNCS 2350*, pages 476–491, 2003.

[66] A.H.S. Lai and N.H.C. Yung. Lane detection by orientation and length discrimination. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 30(4): 539–548, 2000.

[67] Q. Li, N.N. Zheng, and H. Cheng. Springrobot: A prototype autonomous vehicle and its algorithms for lane detection. *IEEE Intelligent Transportation Systems*, 5(4):300–308, 2004.

[68] D. Liebowitz. *Camera calibration and reconstruction of geometry.* PhD thesis, University of Oxford, June 2001.

[69] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *IEEE Proc. Computer Vision and Pattern Recognition*, volume 0, pages 482–488, 1998.

[70] D. Liebowitz, A. Criminisi, and A. Zisserman. Creating architectural models from images. *Computer Graphics Forum*, 18(3):39–50, 1999.

[71] E. Lutton, H. Maître, and J. Lopez-Krahe. Contribution to the determination of vanishing points using Hough transform. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(4):430–438, 1994.

[72] C. Maduro, K. Batista, P. Peixoto, and J. Batista. Estimation of vehicle velocity and traffic intensity using rectified images. In *IEEE International Conference on Image Processing*, pages 777–780, 2008.

[73] M. J. Magee and J. K. Aggarwal. Determining vanishing points from perspective images. *CVGIP*, 26:256–267, 1984.

[74] J. C. McCall and M. M. Trivedi. Video-based lane estimation and tracking for driver assistance: Survey, system, and evaluation. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):20–37, 2006.

[75] C. F. McLean and D. Koyyuri. Vanishing point detection by line clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1090–1095, 1995.

[76] A. Minagawa, N. Tagawa, T. Moriya, and T. Gotoh. Vanishing point and vanishing line estimation with line clustering. *IEICE Transactions Inf. & Syst.*, E83-D (7), 2000.

[77] A.M. Muad, A. Hussain, S.A. Samad, M.M. Mustaffa, and B.Y. Majlis. Implementation of inverse perspective mapping algorithm for the development of an automatic lane tracking system. In *IEEE Proc. Region 10 Conference (TENCON)*, volume 1, pages 207–211, 2004.

[78] R. Neal. Slice sampling. *Annals of Statistics*, 31:705–767, 2000.

[79] S. Nedevschi, R. Schmidt, T. Graf, R. Danescu, D. Frentiu, T. Marita, F. Oniga, and C. Pocol. 3d lane detection system based on stereovision. In *IEEE Proc. Intelligent Transportation Systems Conference*, pages 161–166, 2004.

[80] S. Nedevschi, F. Oniga, and R. Danescu. Increased accuracy stereo approach for 3d lane detection. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 42–49, 2006.

[81] R. Nevatia and K.R. Babu. Linear feature extraction and description. *Computer Graphics and Image Processing*, 13:257–269, 1980.

[82] M. Nieto and L. Salgado. Real-time vanishing point estimation in road sequences using adaptive steerable filter banks. In *Proc. Advanced Concepts for Intelligent Vision Systems, LNCS 4678*, pages 840–848, 2007.

[83] M. Nieto and L. Salgado. Non-linear optimization for robust estimation of vanishing points. In *IEEE Proc. International Conference on Image Processing*, page (accepted), 2010.

[84] M. Nieto and L. Salgado. Plane rectification through robust vanishing point tracking using the Expectation-Maximization algorithm. In *IEEE Proc. International Conference on Image Processing*, page (accepted), 2010.

[85] M. Nieto and L. Salgado. Real-time robust estimation of vanishing points through nonlinear optimization. In *IS&T/SPIE Proc. International Conference on Real-Time Image and Video Processing*, page (Invited paper), 2010.

[86] M. Nieto and L. Salgado. Automatic video mosaicing for surveillance using vanishing points. In *SPIE Newsroom*, 2010.

[87] M. Nieto, L. Salgado, F. Jaureguizar, and J. Cabrera. Stabilization of inverse perspective mapping images based on robust vanishing point estimation. In *IEEE Proc. Intelligent Vehicles Symposium 2007*, pages 315–320, 2007.

[88] M. Nieto, J. Arróspide, L. Salgado, and F. Jaureguizar. On-board video based system for robust road modeling. In *IEEE Proc. Content-Based Multimedia Indexing*, pages 109–116, 2008.

[89] M. Nieto, J. Arróspide, L. Salgado, and F. Jaureguizar. Robust multiple lane road modeling based on perspective analysis. In *IEEE Proc. International Conference on Image Processing*, pages 2396–2399, 2008.

[90] M. Nieto, L. Salgado, and F. Jaureguizar. Robust road modeling based on a hierarchical bipartite graph. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 61–66, 2008.

[91] M. Nieto, C. Cuevas, and L. Salgado. Measurement-based reclustering for multiple object tracking with particle filters. In *IEEE Proc. International Conference on Image Processing*, pages 4097–4100, 2009.

[92] K. Onoguchi, N. Takeda, and M. Watanabe. Planar projection stereopsis method for road extraction. *IEICE transactions on information and systems*, 81(9):1006–1018, 1998.

[93] R. Pflugfelder. *Self-calibrating cameras in video surveillance*. PhD thesis, Graz University of Technology, 2008.

[94] L. Quan and R. Mohr. Determining perspective structures using hierarchical Hough transform. *Pattern Recognition Letters*, 9:279–286, 1989.

[95] E. Ribeiro and E. R. Hancock. Perspective pose from spectral voting. In *IEEE Proc. Conference on Computer Vision and Pattern Recognition*, volume 1, pages 656–662, 2000.

[96] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *Proc. European Conference on Computer Vision*, volume 1, pages 430–443, 2006.

[97] C. Rother. A new approach for vanishing point detection in architectural environments. In *Proc. 11th British Machine Vision Conference*, pages 382–391, 2000.

[98] F. Schaffalitzky and A. Zisserman. Planar grouping for automatic detection of vanishing lines and points. *Image and Vision Computing*, 18:647–658, 2000.

[99] G. Schindler and F. Dellaert. Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In *Proc. Conf. on Computer Vision and Pattern Recognition*, pages 203–209, 2004.

[100] D. Schreiber, B. Alefs, and M. Clabian. Single camera lane detection and tracking. In *IEEE Intelligent Transportation Systems Conference*, pages 1114–1119, 2005.

[101] I. Sekita. On fitting several lines using the em algorithm. In *CVVC*, pages 107–109, 1994.

[102] K.-S. Seo, J.-H. Lee, and H.-M. Choi. An efficient detection of vanishing points using inverted coordinates image space. *Pattern Recognition Letters*, 27(2):102–108, 2006.

[103] D. Serby, E.-K. Meier, and L. Van Gool. Probabilistic object tracking using multiple features. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 2, pages 184–187, 2004.

[104] Y. Shu and Z. Tan. Vision based lane detection in autonomous vehicle. In *Proc. 5º World Congress on Intelligent Control and Automation*, pages 5258–5260, 2004.

[105] J. A. Shufelt. Performance evaluation and analysis of vanishing point detection techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(3):282–288, 1999.

[106] N. Simond and P. Rives. Homography from a vanishing point in urban scenes. In *IEEE/RSJ Proc. International Conference on Intelligent Robots and Systems*, volume 1, pages 1005–1010, 2003.

[107] B. Southall and C.J. Taylor. Stochastic road shape estimation. In *Proc. International Conference on Computer Vision*, pages I: 205–212, 2001.

[108] G. P. Stein, O. Mano, and A. Shashua. A robust method for computing vehicle ego-motion. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 362–268, 2000.

[109] R. S. Stephens. Probabilistic approach to the Hough transform. *Image Vision Computing*, 9(1):66–71, 1991.

[110] R. Subbarao and P. Meer. Beyond ransac: User independent robust regression. In *Workshop on 25 Years of RANSAC*, 2006.

[111] T. Suttorp and T. Bücher. Robust vanishing point estimation for driver assistance. In *IEEE Proc. Intelligent Transportation Systems Conference*, pages 1550–1555, 2006.

[112] J.-P. Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *IEEE Proc. International Conference on Computer Vision*, pages 1250–1257, 2009.

[113] T.K. ten Kate, M.B. van Leewen, S.E. Moro-Ellenberger, B.J.F. Driessen, A.H.G. Versluis, and F.C.A. Groen. Mid-range and distant vehicle detection with a mobile camera. pages 72–77, June 2004.

[114] P.H.S. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Journal of Computer Vision and Image Understanding*, 78(1):138–156, 2000.

[115] H.-H. Trinh and K.-H. Jo. Image-based structural analysis of building using line segments and their geometrical vanishing points. In *SICE-ICASE International Joint Conference*, pages 566–571, 2006.

[116] L.-W. Tsai, J.-W. Hsieh, C.-H. Chuang, and K.-C. Fan. Lane detection using directional random walks. In *IEEE Proc. Intelligent Vehicles Symposium*, pages 303–306, 2008.

[117] T. Tuytelaars, M. Proesmans, and L. Van Gool. The cascaded Hough transform. In *IEEE Proc. International Conference on Image Processing*, pages 736–739, 1998.

[118] T. Veit, J.-P. Tarel, P. Nicolle, and P. Charbonnier. Evaluation of road marking feature extraction. In *IEEE Conf. on Intelligent Transportation Systems*, pages 174–181, 2008.

[119] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1), 2009.

[120] D. Walsh and A.E. Raftery. Accurate and efficient curve detection in images: the importance sampling Hough transform. *Pattern Recognition*, 35:1421–1431, 2002.

[121] H. Wang, Y. Cheng, T. Fang, J. Tyan, and N. Ahuja. Gradient adaptive image restoration and enhancement. In *IEEE Proc. International Conference on Image Processing*, pages 2893–2896, 2006.

[122] Y. Wang, E. K. Teoh, and D. Shen. Lane detection and tracking using B-snakes. *Image and Vision Computing*, 22:269–289, 2004.

[123] C. Wojek and B. Schiele. A dynamic conditional random field model for joint labeling of object and scene classes. In *Proc. European Conference on Computer Vision*, pages IV: 733–747, 2008.

[124] L. Xu, E. Oja, and P. Kultanen. A new curve detection method: Randomized Hough Transform (RHT). *Pattern Recognition Letters*, 11(5):331–338, 1990.

[125] S.Y.K. Yuen, T.S.L. Lam, and N.K.D. Leung. Connective Hough transform. *Image and Vision Computing*, 11:295–301, 1993.

[126] J. Zhou and B. Li. Homography-based ground detection for a mobile robot platform using a single camera. In *IEEE Proc. International Conference on Robotics and Automation*, pages 4100–4105, 2006.