DEGREE OF DOCTOR OF PHILOSOPHY IN
COMPUTER ENGINEERING AND SCIENCE

DOCTORATE SCHOOL IN
INFORMATION AND COMMUNICATION TECHNOLOGIES

XXVI Cycle

UNIVERSITY OF MODENA AND REGGIO EMILIA
Department of Engineering "Enzo Ferrari"

Ph.D. DISSERTATION

Mobile Visual Recognition of Shapes and Places

Candidate
Michele Fornaciari

Tutor
Prof. Andrea Prati
Prof. Rita Cucchiara

The Director of the School
Prof. Giorgio Matteo Vitetta

Abstract

The growth of mobile devices capabilities makes them suitable to perform complex processing tasks. This greatly widens the range of algorithms that can be run directly on the mobile device, therefore enabling the spread of many new applications unfeasible until few years ago.

Researches in Computer Vision can now exploit the growing computational capabilities of mobile devices equipped with high quality cameras, as long as many other built-in sensors. However, several limitation of mobile devices, with respect to traditional desktop computers, must take into account. Despite the hardware improvements, computational capabilities and memory availability may present a severe issue, as well as limited battery life and network connectivity. Also, in the mobile context the user directly interacts with the device so that real time response is often required.

Such limitations suggest that moving the computation towards mobile devices is not a mere porting of existing algorithms. Optimized code may run on the device, but most applications require further processing or data that cannot be found directly on the mobile device. Building mobile applications requires to design algorithms that fit in the mobile system architecture composed in by the mobile device itself, a remote server and the network connectivity in between.

The purpose of the recently born field of Mobile Vision is to face these issues. Mobile Vision is not only about optimizing computer vision algorithms to run on limited hardware, but also about defining mobile-oriented paradigms for algorithms, and application designs to meet a particular mobile vision system architecture, exploiting the set of sensors available on the mobile device, and taking advantage of the role played by the user in a mobile context.

The goal of this thesis is twofold. Firstly, it explores the improvements brought so far thanks to Mobile Vision, providing a thorough analysis of the literature in this field and focusing on the open challenges. The architectural solutions and the optimization techniques required to run mobile vision applications are then discussed. Secondly, it proposes two novel applications, namely an algorithm that make the ellipse detection task feasible on mobile device in real-time, and a lightweight approach to visual place recognition to provide on the fly useful content to users through intuitive and natural interaction.

Sommario

L'incremento delle capacità computazionali dei dispositivi mobili rende possibile l'esecuzione su questi dispositivi di algoritmi di elevata complessità computazionale, permettendo la realizzazione di applicazioni impensabili fino a pochi anni fa.

Adesso la comunità scientifica di Visione Artificiale può sfruttare le migliori caratteristiche dei dispositivi mobili, corredati di fotocamere di qualità elevata e molti altri sensori. Tuttavia occorre tenere in considerazione una tutta una serie di limitazioni proprie dell'ambito mobile. Nonostante i continui miglioramenti, la capacità computazionale e la memoria disponibile non è ancora paragonabile ai tradizionali computer. L'autonomia limitata della batteria e i problemi di connessione posso presentare un serio problema. Inoltre, nel contesto mobile l'utente interagisce direttamente con il dispositivo e con l'applicazione, e quindi è richiesto un tempo di risposta adeguato.

Queste limitazioni mostrano come l'utilizzo di algoritmi di Visione Artificiale su dispositivi mobili non sia immediato. Anche se del codice ottimizzato può essere eseguito sul dispositivo, la maggior parte delle applicazioni richiedono ulteriori dati e capacità computazionale. Lo sviluppo di applicazioni mobili richiede la progettazione di algoritmi che si adattino all'architettura del sistema composto dal dispositivo mobile, da un server remoto e da una connessione di rete che li collega.

Lo scopo della Visione Mobile, campo scientifico nato di recente, è proprio di affrontare queste problematiche. Il suo scopo non è solo quello di ottimizzare gli algoritmi di Visione Artificiale in modo che possano essere eseguiti su dispositivi con prestazioni limitate, ma anche di definire l'architettura dei sistemi, di sfruttare i sensori disponibili, e di avvantaggiarsi del ruolo dell'utente in rapporto con il dispositivo.

Questa tesi si pone due obiettivi principali. Il primo è esplorare i miglioramenti apportati finora grazie ai principi della Visione Mobile, fornendo un'ampia analisi della letteratura associata e focalizzandosi sulle sfide rimaste ancora aperte. Inoltre sono discusse le soluzioni architetturali e le tecniche di ottimizzazione richieste per eseguire una applicazione in ambito mobile. Il secondo è di proporre due nuove applicazioni, cioè un algoritmo che permette di eseguire in tempo reale su un dispositivo mobile il riconoscimento di forme ellittiche, e un approccio computazionalmente leggero al riconoscimento visuale di luoghi che permette di fornire contenuti tempo reale all'utente attraverso un'interazione naturale.

Contents

List of Figures

List of Tables

Chapter 1

Introduction to Mobile Vision

Since a few years ago, it was difficult to imagine an off-the-shelf smart phone with computational capabilities comparable with mid-range computers and equipped with a wide range of sensors, including high quality cameras. The increasing performance of mobile devices is providing many new opportunities for the scientific community. This is especially true in the field of computer vision, where smart phone cameras guarantee high quality images (see Fig. 1.1) and videos, and mobile devices have enough computational power to process these data on board.

This has largely fostered the emerging research area of *mobile vision* [81] in the past several years. Research in mobile vision is not only about making computer vision algorithms work in real-time on mobile device with less computational capacity and power. Even though the mobile device has several limitations with respect to standard platform such as desktop computers and servers, the sensors embedded on mobile devices provide a new set of data that may serve to complement the visual information.

Human factors are also another important aspect to take into account because of the strong interaction between a user and the carried mobile device.

This chapter explores the new field of the mobile vision. The use of mobile devices in place of standard computers presents strengths and weaknesses. A thorough analysis of the literature details the solution proposed so far, highlighting novel opportunities. Such application, however, must deal with different limitations proper of mobile devices.

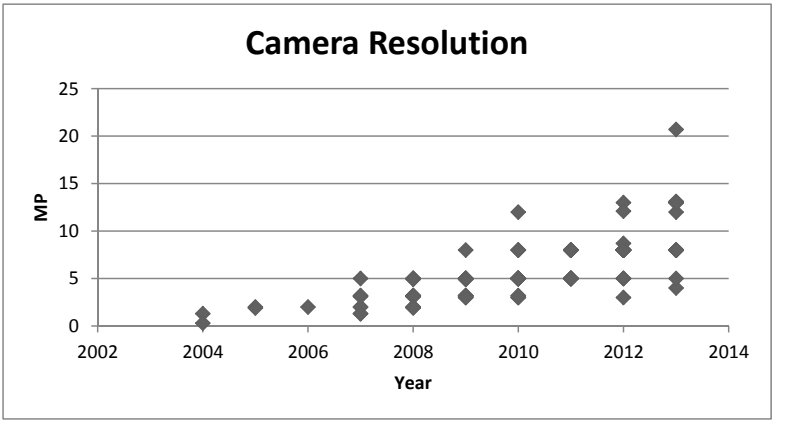


Figure 1.1: Camera Quality.

1.1 Limitations of Mobile Devices

New opportunities for a large set of applications are linked to the employment of computer vision technologies on mobile devices. Mobile vision addresses the challenges posed by the usage on such devices of algorithm designed for standard computers.

1.1.1 Limited Battery Life

Mobile devices are by definition not powered through cables, but by a battery which guarantees only limited autonomy. While battery technology will undoubtedly improve over time, as depicted in Fig. 1.2, the need to be sensitive to power consumption will not diminish: a powered-off device is, of course, useless.

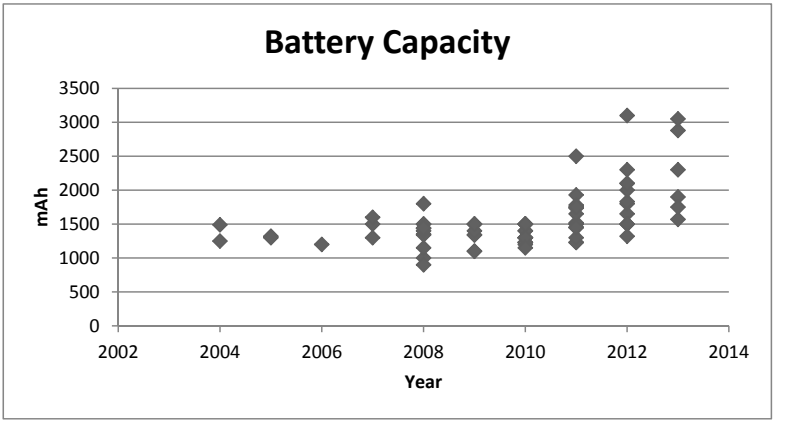


Figure 1.2: Battery Capacity.

This poses several limits to the kind of computations that can be performed on the device. First, very long-lasting algorithms are not guaranteed to finish before the battery is discharged. Because of this (and several other aspects that will be covered in the following) mobile devices can not be considered as just another computational unit. Second, applications should not drain all battery power, but should make a proper use to guarantee the other functionality of the mobile device. Consequently mobile vision applications must pose particular attention to power consumption, for example demanding computationally expensive algorithm to a remote server, limiting the use of the sensors, keeping the display off if not neces-

sary, avoiding to send big amount of data on the network, and so on.

1.1.2 Limited Computational Power

Although mobile devices are becoming more and more powerful, as depicted in Fig. 1.3, their computational power is still not sufficient to handle computationally intense tasks. From this perspective, migrating much of the computing into the cloud or to a remote server is essential.

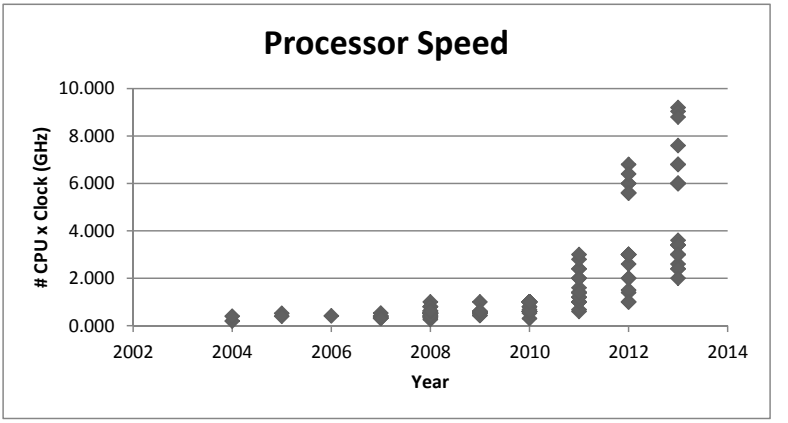


Figure 1.3: CPU speed.

1.1.3 Limited Storage Capabilities

Most mobile vision applications are driven by large amounts of annotated visual data which can not be stored locally on the mobile device. Also in this case, the computation must be migrated to a remote server where storage is not an issue. Other available solutions range from designing lightweight databases, to limiting the size of stored data, for example using compressed or low size descriptors. Such optimization techniques will be further explored in Chapter 2.

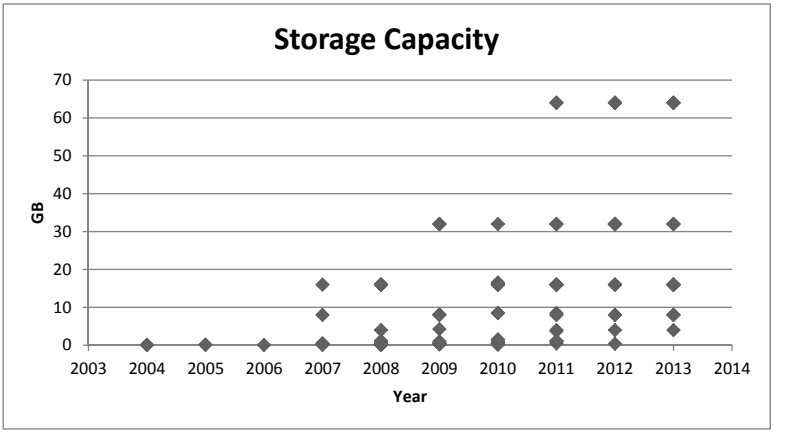


Figure 1.4: Storage Capacity.

1.1.4 Limited Connectivity

When a task may not be performed locally on the mobile device, because the aforementioned issues, the application must communicate with a remote server through the network. Some places may offer reliable, high-bandwidth wireless connectivity, while other may only offer low-bandwidth connectivity. Outdoors, a mobile client may have to rely on a low-bandwidth wireless network with gaps in coverage. As such, network delays and bandwidth limitations must then be taken into account. In Fig. 1.5 is reported the time required to perform an image matching query. Sending the whole image compact features has different impact with respect to the network bandwidth.

In general, however, it is very important to send the minimum amount of information on the network because of billing issues or battery consumption. This can be achieved by pre-processing data on the mobile device and by sending to the remote server low-size features.

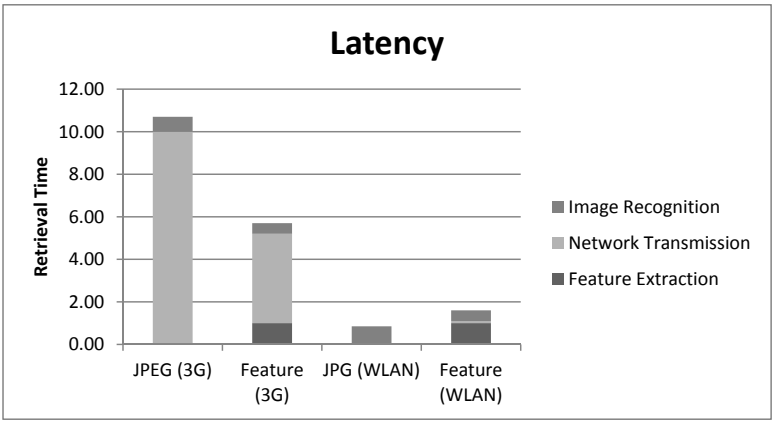


Figure 1.5: Storage Capacity.

1.1.5 Real Time Interaction

Besides differences regarding the hardware, another important difference is related to the role of the user. In most mobile vision application the user is part of the system. In order to make the interaction with the user feasible, as well as enhancing the user experience, mobile vision application usually have very strict real time requirements.

1.2 Related Works and New Opportunities

Mobile phones have evolved into powerful image and video processing devices, equipped with high-resolution cameras, color displays, and hardware-accelerated graphics. They are also equipped with GPS, magnetometer, gyroscope and many other senors, and connected to broadband wireless networks.

Such evolution fostered the development of a wide number of applications. This is also witnessed by the growing number of scientific papers in

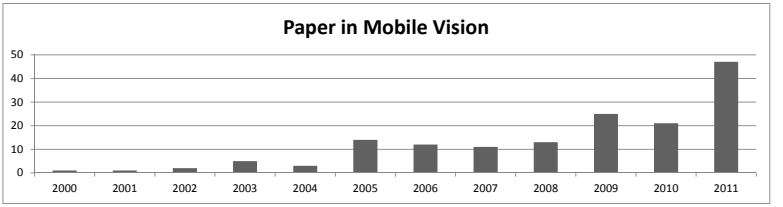


Figure 1.6: Work in Mobile Vision in recent years.

the field of mobile vision (see Fig. 1.6).

1.2.1 Mobile Visual Recognition

Visual place recognition, Location-based services and mixed/augmented reality have become popular topics in recent years, largely in the context of consumer applications where user-centered visual computing is essential. In essence, all these applications start from a snapshot image taken by a user from the camera on the mobile device. This photo will then be matched against a pre-annotated image database to extract useful information that is provided to the user, such as movies reviews, restaurant menus, historical descriptors of a nearby building, and so on.

1.2.1.1 Image Based Retrieval

The advances of mobile device enables a new class of applications that use the camera phone to initiate search queries about objects in visual proximity.

This kind of applications may be useful in everyday activities like product recognition. In scenarios where a customer wants to easily get information about products, such as books or CDs, these applications allow to retrieve the requested information simply taking a snapshot with the camera-phone of the product [30, 167, 169, 195]. The query is processed in

the device and query data are sent over the wireless network to a remote server to be recognized against a large database of products. The results are then sent back to the customer. Even if such applications are relatively simple, since book and CD covers are rigid planar objects with a texture, they present several issues typical of mobile vision. The variance of the snapshot, due to different light conditions or motion blur, as well as image geometric distortions require to extract robust features for a reliable matching. Aside well-known descriptors typical of Computer Vision such as SIFT, SURF, and Bag-of-Words approach, novel features are presented to allow a compact representation and cope the limited bandwidth networks. Semi-local visual parts grouped from local features are proposed in [195]. This solution requires to take multiple shots or a video clip in order to remove noise and clutter, but significantly improves the precision of retrieval. Also, real time response to the user is essential, and recognition algorithms should scale gracefully with the size of databases. In [167, 169] the Scalar Vocabulary Tree is used to efficiently reduce the search space to a small subset of most likely database matches.

Mobile devices are now frequently used to download and watch videos. However videos watched on a TV are typically not related to the same video available online. Consequently to find the same video to watch the remainder may be a long and boring activity. In [26] is presented a system to recognize not only the video, but also the exact scene, so that the user can easily resume the video and watch it on the mobile device. To overcome the issue that a single snapshot may not contain enough useful features for an accurate recognition, they propose to take a short video and to automatically select the best frames. This guarantees low latency in query acquisition and transmission.

Mobile image matching approaches have also been used to increase road safety by recognizing traffic signs and alerting the user [103]. Inventory of road sign may also be achieved relying on the GPS sensor available on the

mobile device [12].

Other applications range from a cooking recipes recommendation system, which is able to recognize the food ingredients[110, 192], to the recognition of electronic components [4].

The importance of these applications is witnessed by the interest of the industry with application like SnapTell, Kooaba, and Google Goggles.

1.2.1.2 Visual Place Recognition

The main challenge for visual localization is the rapid and accurate search for images related to the current recording in a large georeferenced dataset. In the computer vision community many methods, such as [99, 145] use the detected local features to generate the visual words which will be used to compute the image descriptors such as Bag of Features for retrieval use. Indexing structures such as vocabulary trees [117] are then used to organize the generated image descriptors to get a fast retrieval system. While promising, most of these works rely mainly on visual analysis for recognition and are centered on PC environments and are not feasible on mobile devices.

With recent advances in mobile computing, the demand for visual place recognition or landmark identification on mobile device is gaining much interest. Image captured with a mobile device are used to retrieve the spatially closest image from a georeferenced dataset [147]. The visual appearance may dramatically change between the image of a building stored in the dataset and a query. Different lighting conditions which may cause shadows and reflections, different viewpoint and the presence of dynamic objects such as pedestrians or cars may produce great variation in the images. Besides mentioned issues, mobile visual place recognition may rely on some sort of localization, provided by the GPS sensors or at least by the cell-ID of the network provider.

In mobile applications, the mobile user takes a query image, and then

transmits it to a remote server to identify its corresponding image through visual searching. The database images are commonly represented by BOF descriptors, and the inverted indexing is used to organize these visual descriptors for retrieval use. In [148] retrieval performance is significantly increased by composing partial vocabularies based on the uncertainty about the location of the client, thus efficiently integrating prior-knowledge into the matching process.

Network latency may be reduced by implementing the algorithm directly on the mobile device [65, 160]. This is achieved in [160] by compressing and incrementally update the features, derived from SURF, on the mobile phone. Scalability is guaranteed by pruning irrelevant features based on the proximity to the user. A vector quantization strategy is adopted also in [65], combining the Transform Coding and Residual Vector Quantization.

The location recognition accuracy may be incremented by fusing the inertial sensors and computer vision techniques [66]. The GPS may be used to retrieve only images falling in nearby locations cells [160], to search local vocabulary trees to speed up the visual searching process [31] or to design location discriminative vocabulary coding to achieve low bit rate transmission [88]. In [65] the VLAD descriptor is integrated with the GPS data. In [32] is proposed an effective method that employs an integration of content and context analysis to perform landmark recognition. For the content analysis a new bag-of-words framework was developed, while the contest analysis involves fusion of location and direction information. The performance of visual and inertial sensors is further improved designing an efficient Adaboost algorithm [100].

Other approaches rely on 3D models. Exploiting vanishing points in query images and thus fully removing 3D rotation from the recognition problem allows to simplify the feature invariance to a purely homothetic problem [6]. 3D coordinates may also be obtained by projecting features

on a digital model [181].

1.2.2 Mixed/Augmented Reality

With the increasing quality and pixel resolution of smart phones cameras, they can replace dedicated digital cameras in many circumstances. For some applications, smart phones may be superior as they typically have much more computational capacity and additional sensors, enabling mobile applications on the fly such as mixed and augmented reality applications [120].

One of the central issue in augmented reality is to track a planar object moving relatively to a camera. This task is difficult due to several factors, such as illumination changes and motion blur.

First applications relied on fiducial markers because their detection is less computationally demanding [176]. The use of 2D visual codes, such as QR codes, attached to physical objects allow to retrieve object-related information and functionality providing a natural way of interaction with these objects [138]. More accurate techniques allow to enhance tracking performance also in case of degradation in the effective image, that can happen when the target is distant from the camera [83] or in the case of visual clutter [177].

Instead of a marker, in [38, 97] the human hand is used as a distinctive pattern. The virtual object is rendered on the palm and reacts to hand and finger movements.

In outdoor environments, the application needs to match the camera-phone image with location-tagged images. To avoid network latency, the features stored directly on the phone may be compressed and incrementally updated [160]. Other techniques recur to the 3D reconstruction of the scene [5]. Sensor fusion techniques based on GPS and gyroscope may improve the accuracy [153, 196]

Hybrid approaches based on the client-server architecture have also

been proposed. The task of recognizing an object and tracking it on the user screen is split into a server-side and a client-side task, respectively [60, 69, 113].

Object recognition and tracking may be based on the 3D geometry of the related object, generated from common 3D CAD models [11, 62].

Tracking features using motion vectors obtained directly from the video coder has been proposed in [159]. For the tracking of weakly textured objects the MSER descriptor has been proposed [46] for detecting the required contours in an efficient manner and to apply random ferns as efficient and robust classifier for tracking. Other approaches rely on the shape of the target, which can thus be also a sketch [71]. The approach described in [123] aims at tracking object without texture, but relies on the use of depth information. In [171] instead, the proposed method based on the geometrical features aims at detecting a wide variety of textures, including handwritten text, low-textured images, traditional black and white markers, and even random dot markers.

Aside for gaming purposes, augmented reality technologies have been applied also for the measurement, 3D modeling and visualization of furniture or other objects [157]. In [84] is presented an application for augmented reality registration on wind farms.

Once the pose of a target object has been estimated, it is displayed on the screen of the mobile device. The methods [93, 118] deal with the rendering of such objects, adapting illumination and shadows according to the scene to make virtual object indistinguishable from real objects.

1.2.3 Text Recognition and Translation

The translation of a text framed by a mobile camera may be very useful in several circumstances.

Text Recognition is usually performed by standard Optical Character Recognition methods. They typically presents low recognition rate when

the text presents perspective distortions or is not properly aligned, centered, zoomed, or illuminated. Consequently, after a detection step, a pre-processing involving perspective transformation and illumination balancing is required [17, 49, 170]. Different techniques are adopted for text detection. Assumption on the shape, such as rectangular sign [17] may guarantee good detection accuracy, but less generality.

Other methods rely on a texture segmentation approach based on Gabor filters [49], or on edge-enhanced MSER-based algorithms [170]. The approach in [124] instead focuses first on single letters, relying on the geometry of the edges contour. Efficient rules allows then to quickly find the reminder of the word. The simplest solution is the manual initialization by tapping on the screen [57].

A specific method was developed for detecting Korean signs [45]. Text detection is performed using a simple edge-based method, under several assumption such as the position of the text in the image.

1.2.4 Soft Biometrics

Modern smart phone not only have the memory capacity to store large amounts of sensitive data, such as contact details and personal photos, but they also provide access to persoanl data stored on the internet, such as on social networking sites or email. Although passwords provide protection against unauthorized access to this data it presents several drawbacks.

Biometric authentication [166] is an alternative that allow unique identification of the user and is not easy to be lost or stolen. While biometric system may be based also on fingerprints and voice recognition, mobile vision applications are focused on the use of the camera.

1.2.4.1 Face Recognition and Authentication

Using the mobile camera is possible to authenticate the user from the facial features.

The first step is to detect the face of the user. Many methods [24, 70, 115, 162] rely on the well-known method proposed by Viola and Jones [175] or its variations. For close range face detection, a cascade asymmetric principal component discriminant analysis is proposed, enhanced with a focus attention strategy [134].

Typical algorithms for face recognition on mobile device rely on correlation filters, Individual PCA and FisherFaces [173]. While Individual PCA and FisherFaces work in the image domain, correlation filters work in the frequency domain and offer advantages such as shift-invariance, the ability to accommodate in-class image variability, and closed-form expressions. Computationally efficient minimum average correlation energy filters have been used also minimize computation on the device [115]. Face recognition may be accomplished through color-based skin detection and verification with fiducial points to reduce errors [146]. Fiducial components have been used also to build a facial graph model which involves both of appearance and geometric facial information, is built for face representation [1]. Salient parts of the face are used to improve the accuracy. Through eyes template matching in specific regions of the face, then the local binary pattern is adopted for fast face recognition [24, 70]. Wavelet domain feature vectors have been proposed for accuracy, efficiency and adaptability [85]. In [76] are discussed methods based on the appearance using Support Vector Machines to handle large dimension feature vectors, and probabilistic classifiers based on Gaussian Mixture Models. Real time training on mobile device [39] is obtained extracting local face features using some local random bases and then sequential neural network is trained incrementally. This allow to drastically reduce computational time. Complete systems for faces verification are discussed in [42, 135]

1.2.4.2 Emotion and Expression Analysis

Nowadays most smart phone are equipped with both rear and frontal camera. This could lead to application able to analyze the expression of the user. The industry started to take interest in such features: the Samsung Smart Pause feature uses the front camera to sense when the user is looking at the device and pauses video playback when the user looks away from the screen, while an iPhone app allow to unlock the device with a smile.

Application that are able to recognize the users emotion may provide useful feedback. Some application already exist [96, 130], but they do not rely on the camera. The use of computer vision techniques for emotion recognition may provide an interesting contribution.

1.2.5 On-board Image and Video Processing

The growing computational capabilities of modern mobile device make several image and video processing algorithms feasible directly on the device.

1.2.5.1 Video Rectification

In [136] is presented a method to efficiently rectify videos from mobile device equipped with rolling shutter cameras. The distortion model based on 3D camera rotation is extended using multiple knots across a frame in order to enable the algorithm to detect non-constant motion during frame capture. Rolling shutter distortions are corrected also using measurements from accelerometer and gyroscope sensors [74]. Low-pass filters are used to obtain the output camera trajectory. In [56] the camera motion is parametrized as a continuous curve with knots at the last row of each frame. Curve parameters are solved using non-linear least squares over inter-frame correspondences.

1.2.5.2 Image Editing

Effective methods to create a new composite image by removing, adding, and moving objects in an image have been proposed [188]. First, a gradient vector field is created from the gradients in the source image, and then updated by inserting or removing object gradients and filling areas with a best-fit patches approach. A diverge vector field is computed and used for recovering. Other methods to remove objects from images or videos are presented in [78, 94, 152], first selecting the object to remove and then adopting inpainting techniques.

1.2.5.3 Panorama Building

Many applications provide the capabilities to build panorama images by stitching together several images of the same subject with slightly different points of view.

Approaches based on dynamic programming find a good boundary along which to merge the images, and a translation smoothing process using instant image coning removes merging artifacts [184]. A minimal-cost path is used in [185] as an optimal seam to label images. Overlapping images are cut along the seam and then merged together. By finding optimal seams is possible to avoid ghosting and blurring problems caused by moving objects and small registration errors. In [187] graph cut optimization is used for finding optimal seams in overlapping areas in the source images to create the composite images.

In [186] is proposed a computational and memory efficient method, based on a mask-based image blending approach.

1.2.5.4 Structure Estimation and Scene Reconstruction

The estimation of 3D structure and motion from 2D images is a central problem in computer vision, and now, thanks to technologies advances, is

possible also on mobile device.

In [73] is derived a dynamic system describing the motion of the camera and the image formation. An extended Kalman filter is applied for estimation of both structure and motion. The method described in [77] deals with the issues of structure from motion algorithm in case of rolling shutter video exploiting the continuity of camera motion both between and across a frame. In [174] is introduced a system which constructs a textured geometric model of the user's environment as it is being explored. This is achieved by organizing 3D features into roughly planar surfaces and applying stereo analysis. A SLAM (Simultaneous Localization and Mapping) system based on keyframes, as long as the adaptations to mitigate the impact of the imaging deficiencies of the device is described in [92].

Starting from a set of panoramic images, a coarse 3D model of the environment can be estimated by using a cheap on-line space carving approach [122]. The method described in [61] behaves like a panorama mapping or a SLAM system depending on the motion of the camera.

1.2.6 Tourism And Cultural Heritage

The availability of methods for mobile place recognition and object detection allows the development of applications for tourism and cultural heritage. In such context users often require to know where they are or what they looking at, in order to get useful information about a place or a piece of art [43].

In [121] is proposed a variant of the standard SIFT descriptor by selecting only informative keys for the identification of urban objects. Scene recognition systems [102] enable tourists to access description of outdoor environments.

Many application have been proposed to enhance the user experience in museums. The SURF descriptor is adopted in [9] for the recognition of ob-

jects of art. In [52] is presented a museum guidance systems for lightweight object recognition based on neural networks. This enables museum visitors to identify exhibits by capturing photos of them [14]. A dynamic network configuration adapts to the current visibility situation while keeping the memory and processing requirements at a minimum. In the future, entering a museum might automatically trigger the transformation of the mobile phone of a visitor into a piece of personal guidance equipment [15]. The tourist guide system developed in [35] presents to city visitors information tailored to both their personal and environmental contexts. For improving user experience during museum visits, markerless augmented reality systems have been proposed [44, 164] to overlay information on the user mobile screen. The project described in [3] provides customized and detailed multimedia information also in a archaeological site.

Chapter 2

Mobile Vision Architectures

The architecture of a mobile vision system is composed by a mobile device, carried by a user, and connected to a remote server through a wireless network, as depicted in Fig. 2.1. This architecture is due the fact that mobile applications usually require resources that are not available directly on the mobile device, and thus need to be retrieved from a remote server through a network connection.

In the field of computer vision, tasks are performed directly by a powerful-

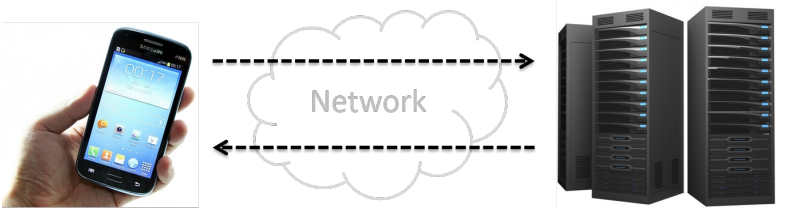


Figure 2.1: Mobile vision general architecture.

enough computer (or even a cluster, or in the cloud), where processing is usually batch-based since the response time is not a major issue. Mobile applications, instead, pose a unique set of challenges (see Sect. 1.1). What part of the processing should be performed on the mobile client, and what part is better carried out at the server? On the one hand, transmitting a JPEG image could take tens of seconds over a slow wireless link. On the other hand, extraction of salient image features is now possible on mobile devices in seconds or less. This leads to several possible client-server architectures.

Some computational intense algorithm are not feasible on the mobile device, either for limited computational capabilities or memory requirements, and are thus more suited to run on a remote server which then sends the results back to the mobile device. Some algorithm need a large amount of data to complete. As an example, image matching algorithm usually require a huge amount of images in order to provide meaningful results. If these data can not be stored on the mobile device because of storage limitations, the solution is to store them remotely. The matching algorithm that need such data must then be run remotely to efficiently access the required data.

Instead of saving all data on the internal storage, it is also possible to retrieve the needed data online. On one hand, this task may be accomplished directly by the mobile device. However, the limitations of the mobile network for connectivity, bandwidth or billing policies may present a severe issue. On the other hand, remote servers are usually equipped with stable, fast and cheap connectivity. As a result, if the amount of traffic on the network is large, it can be demanded to the remote server.

On the top of these considerations, the design choice regarding how to distribute the computation throughout the system results to be crucial to guarantee the correct functioning of the application. The paradigm widely adopted in Mobile Vision is to accurately analyze the data flow of the

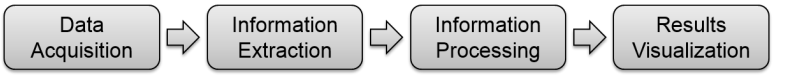


Figure 2.2: Typical processing steps in a mobile vision system.

application. The data flow of a typical mobile vision system is summarized in Fig. 2.2. Every mobile vision application need to collect some sort of data from the sensors, which usually is an image from the on-board camera, though data from accelerometer, gyroscope and other sensors may be useful as well. Accordingly to the computational capabilities of the mobile device, some pre-processing may be executed directly on the mobile device. If the application requires a large amount of data to process the query, usually the query is executed remotely, where the storage capacity is not an issue. Also, the query processing may be very computationally intense, and is this usually performed on remote server with sufficient computational power. The amount of data to send remotely through the network should be usually be kept at a minimum, but relies on the capabilities of the mobile device to efficiently compute lightweight features. If the final processing has been performed remotely, the results need to be returned to the mobile device for the presentation to the user.

In this chapter different mobile vision architectures and optimization techniques to overcome the typical limitations proper of mobile devices are investigated.

2.1 Mobile Vision System Architectures

The feasibility and the performance of a mobile vision application depend heavily on the underlying system architecture. The criteria for the appropriate design choice should take into account the limitations of the mobile device, the quality of the network connectivity, the amount of data and

computation required to answer the query, and the application responsiveness. Four mobile vision architecture may be identified:

- The whole algorithm is performed on the mobile device. All needed data are stored locally (Sect. 2.1.1).
- The mobile client transmits a query image to the server. The algorithm then runs entirely on the server, including the processing of the query image (Sect. 2.1.2).
- The mobile client processes the query image, extracts features and transmits feature data through the network. A portion of the algorithm run on the server using the feature data as input (Sect. 2.1.3).
- The mobile client may be connected also to other devices to create a collaborative network (Sect. 2.1.4).

2.1.1 Mobile Device Only Architecture

A mobile application can be self contained on the mobile device, as depicted in Fig. 2.3. All data required by the application are locally stored, and the computational power of the device is enough to handle all the processing. Since there is not remote server, unless the application itself needs to retrieve information online, the network connectivity is not required.

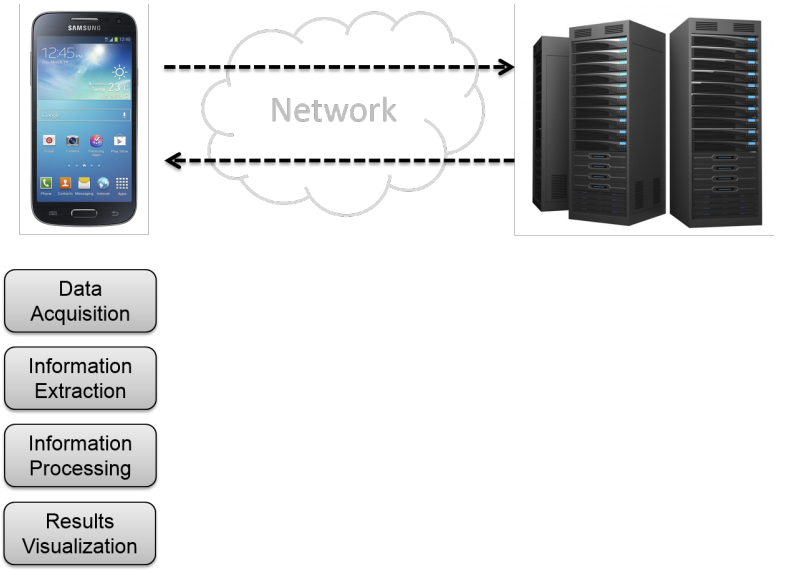


Figure 2.3: Mobile device only architecture.

This architecture guarantees the fastest responsiveness because of the lack of network transmissions or server side computation delays. Applications with strict real time constraints and that do not require large amount of data should be structured according to this architecture. Augmented Reality applications 1.2.2 are a notable example of applications that requires only data about the target and the objects to be drawn,that must run smoothly for the constant user interaction, and where the processing required can be carried out directly by the mobile device. Also the whole range of On-Board Video and Image Processing applications 1.2.5, such as Video Rectification, Image Editing, or Panorama Building applications, run locally on the device.

As already mentioned, applications that rely on this architecture must perform all processing on the device. Methods developed for standard desk-

top computers may perform poorly on such limited hardware. Algorithm optimization is the key to move Computer Vision algorithms to the mobile device. As discussed later in Sect. 2.2, several typology of optimization exist, such using dedicated hardware or re-designing lightweight algorithms. Through appropriate optimization, also the size of data may be reduced, to the extent that even datasets may be stored locally on the device, widening the range of possible application that can benefit from this low-latency architecture.

2.1.2 Remote Server Only Architecture

The mobile device can only serve as an input and presentation device, as depicted in Fig. 2.4. The mobile application retrieves the images from the camera, and eventually data from other available sensors, and sends them directly to the remote server, with no processing involved on the device. The whole processing is demanded to the server.

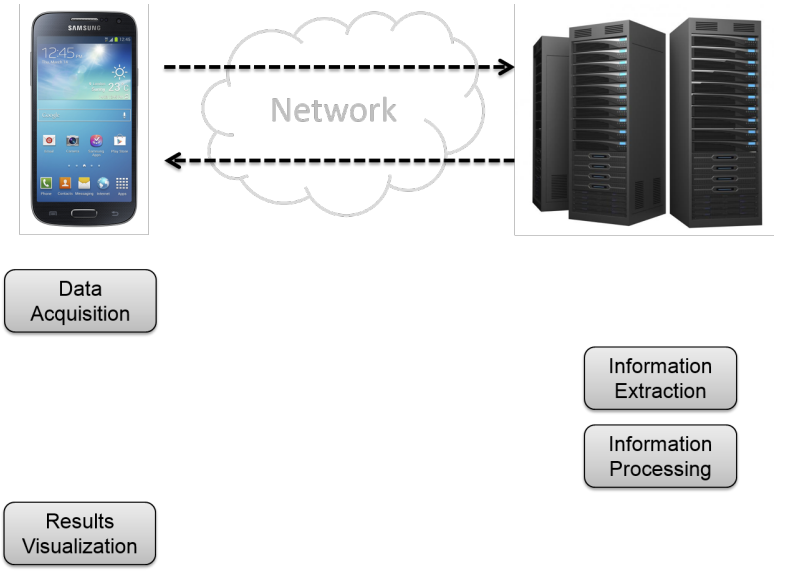


Figure 2.4: Remote server only architecture.

Because the raw data are sent remotely without pre-processing, network delays may represent a severe issue. This architectural solution was desirable when the processing power of mobile device did not permitted the execution of any computer vision algorithm, which now are widely adopted to extract low-size features in order to send only a limited amount of data on the network.

The advantages of this architecture, such as the exploitation of the server capabilities in term of computational power, storage and large bandwidth, can be achieved by relying on an architecture that involves some pre-processing on the mobile device.

2.1.3 Hybrid Architecture

The mobile device is not only an input and presentation device, but also performs a portion of the processing, usually to reduce the size of data sent through the network to the remote server. This allows to overcome the issues of the remote server only architecture of Sect. 2.1.2.

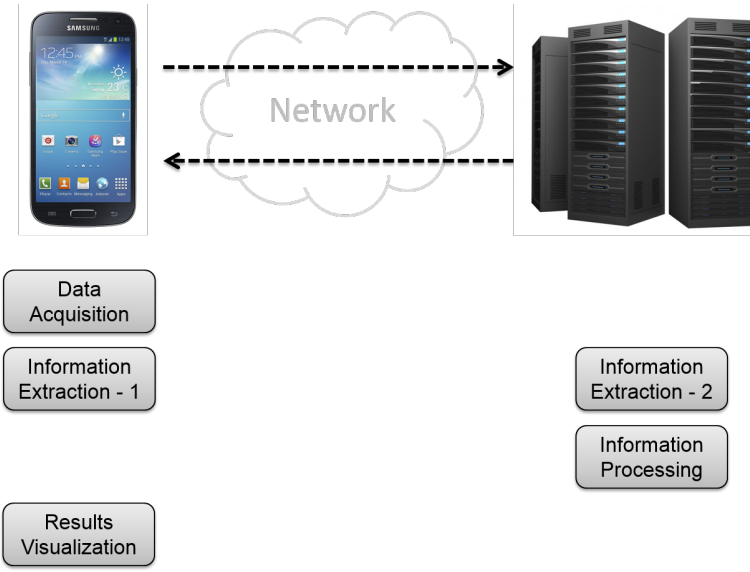


Figure 2.5: Hybrid architecture.

Most mobile vision applications, such as Mobile Visual Recognition applications (Sect. 1.2.1), conform to this architecture (Fig. 2.5). The balance obtained by splitting the computation between the mobile device and the server offers many advantages. Computationally intense task may be demanded to the remote server, without incurring in severe network latency thanks to the on-device data size reduction. Applications have also access to the large amount of data made available by the server.

However, this solution presents also some drawbacks. Because of the processing on the device, the optimization techniques presented in Sect. 2.2 must be taken into account. Also, since the network is an integral part of the architecture, network connectivity should always be present in order to guarantee the correct application functioning.

2.1.4 Mobile Network Architecture

The mobile device, still connected to a remote server, may also be connected to other mobile devices, as depicted in Fig. 2.6.

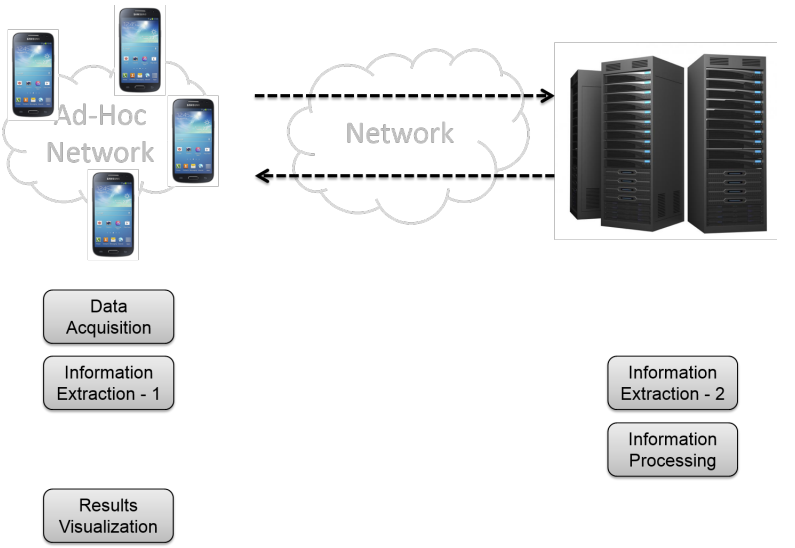


Figure 2.6: Mobile network architecture.

With this architectural solution, mobile devices can interact in a cooperative network. They can exchange data and information with the other nodes of the network to supply to missing data. Each device is also able to interact with a remote server to receive information not available from

the other connected devices.

This architecture may be very useful when the connection to the server is limited or not reliable. When a large amount of data must be retrieved on the remote server and is needed by all the members of the cooperative network, each device can download a different portion of the data, and then retrieve the remaining data from the other network devices. Another advantage is that each mobile device can produce a portion of data, which can than be collectively used to create an aggregated information. Structure from motion (Sect. 1.2.5.4) or panorama building application (Sect. 1.2.5.3) may benefit from the collection of simultaneous data from different viewpoints.

Nevertheless, there is the need of protocols to create and manage the network, mutual trust among devices is mandatory when sharing data, and potential battery power consumption issues may occur due to multiple active connections.

2.2 Optimization

2.2.1 Optimization of the Computation

The computational power is increasing very fast, but due to factors like limited battery life and overheating computationally intense task should be avoided. What can be computed in real-time on a desktop computer is unlikely to run at the same speed on a mobile device. How to enable standard computer vision task to be executed in acceptable time on a mobile device has been an issue tackled largely in literature.

First, faster or memory efficient descriptors can be used to make algorithm based on such descriptors feasible on the mobile device (2.2.1.1). Second, algorithms could be re-designed so to achieve the same results with less computational effort(2.2.1.2). Last, low level code optimization

may guarantee very efficient algorithms on dedicated hardware, like GPU (2.2.1.3).

2.2.1.1 Optimization of the Descriptor

Most of computer vision methods rely on local descriptors. However their computation is not always feasible in real time on the mobile device due to its hardware limitations. Improving the efficiency of descriptors, in terms of computational cost and memory requirements allows to achieve a major speed-up in the whole method.

Running a SURF (Speeded Up Robust Features) detector on mobile devices remains too slow to support emerging applications such as mobile augmented reality. Porting it without adapting the algorithm to account for mobile platform limitations could result in significant run time degradation. In [190], two mismatches between the SURF algorithm and the mobile hardware that cause substantial slow-down of the point detection process are identified. First, a mismatch between the data access pattern and the small cache size. Second, a mismatch between the huge amount of branches and high pipeline hazard penalty. Two techniques are proposed to address these issues: tiled SURF and gradient moment based orientation assignment. Tiled SURF improves data locality and greatly reduces memory traffic. A method for determining the optimal tile sizes, named content-aware tiling, is designed to minimize runtime and maximize detection accuracy. To avoid the penalties caused by pipeline hazards, the original orientation operator is replaced with branching-free gradient moment computations.

An efficient implementation of the SURF descriptors has also been proposed in [33]. Several improvements to the basic algorithm to reduce memory requirements are implemented to enable the execution of the feature detector on mobile phones. A new sampling schema is proposed, and floating points computations are reduced by using a lookup table for the Gaus-

sian filter and by approximating the *arctan* function. To speed up the re-sampling process, mipmaps are pre-computed using the round-up algorithm. The resulting algorithm is on average 30% faster and uses half of the memory.

An approach based on heavily modified SIFT descriptors enables natural feature tracking from textured planar objects at frame rates up to 20 Hz [178, 179]. The original SIFT algorithm uses Difference-of-Gaussians (DoG) to perform a scale-space search that not only detects features but also estimates their scale. Although several faster implementations based on the original definition have been proposed, the approach is inherently resource intensive and therefore not suitable for real-time execution on mobile phones. To achieve this, the DoG corner detection has been replaced with the FAST corner detector, with the non-maxima suppression that is known to be really fast, but still provides a high repeatability. The modified descriptor results to be not scale invariant, but scale is taken into account providing in the descriptor database features from all meaningful scales. By describing the same feature multiple times over various scales, memory is traded for speed to avoid a CPU-intensive scale-space search.

A novel highly efficient, robust and distinctive binary descriptor called Local Difference Binary (LDB) is proposed in [189] for augmented reality applications. LDB directly computes a binary string for an image patch using simple intensity and gradient difference tests on pairwise grid cells within the patch. A multiple gridding strategy is applied to capture the distinct patterns of the patch at different spatial granularities. The final descriptor is obtained concatenating a subset of highly-variant and distinctive bits. Compared with the state-of-the-art binary descriptor BRIEF, LDB has similar computational efficiency, achieves a grater accuracy and allow for a 5 times faster matching.

2.2.1.2 Optimization of the Algorithm

Efficient algorithms and implementation choices allow to perform a given task with less computational effort. In the literature, a lot of attention is dedicated to the task of face detection.

In [133] is discusses how to achieve real-time software-based implementation of The Viola-Jones object detection algorithm on mobile devices that have relatively limited processing and memory capabilities. The first optimization regards data reduction, by reducing the size of the images, by shifting sub-images by a wider range, by increasing the scale size step and by limiting the face dimension. The second uses key-frames and narrowed detection area to limit the amount of search. Fixed point computation guarantees a major speedup in the tree navigation. Similar optimization techniques have been adopted also in [131], even if this work is not based on the Viola-Jones algorithm, but on skin color detection and face shape recognition. They propose several lookup tables for computing the Gaussian Mixture Model for each pixel for the skin detection, in order to consider different lighting conditions.

In [161] is proposed a novel feature optimization method to build a cascade Adaboost face detector for real-time applications on mobile phones. AdaBoost algorithm selects a set of features and combines them into a final strong classifier. However, conventional AdaBoost is a sequential forward search procedure using the greedy selection strategy, and thus redundancy cannot be avoided. The design of embedded systems must find a good trade-off between performances and code size due to the limited amount of resource available in a mobile phone. To address this issue, a novel Genetic Algorithm post optimization procedure for a given boosted classifier is proposed, which leads to shorter final classifiers and a speedup of classification. This GA-optimization algorithm results to be very suitable for building application of embedded and resource-limit device.

For close range face detection, in [134] is proposed to limit the number

of scanning windows. Also, the covariance matrices of the face/eye class are less reliable compared to the covariance matrices of the non-face/non-eye class for close range faces. Therefore, a larger weight should be assigned to the face/eye class when building the covariance mixture matrix to remove the unreliable dimensions. The proposed cascade subspace face/eye detector utilizes the focus-attention strategy in all aspects and detects eyes at precise locations at an acceptable speed.

Real-time face verification on a mobile phone is achieved using computationally efficient Minimum Average Correlation Energy Filters [115]. The Correlation Filters are optimized using a fast fixed-point implementation of 2D Fourier Transform. Also, an efficient algorithm is presented for synthesizing these filters efficiently in order to minimize computation load and making the system practical with real-time enrollment of users.

The paper presented in [39] focused on the challenging problem of learning a large number of samples sequentially within mobile devices in real-time by presenting a real-time training algorithm for face detection related applications. Face features are extracted using some local random bases and then a sequential neural network is trained incrementally with these features. This leads the possibility of training model parameters in rel-time in mobile devices, and the method may be extended to other computer vision and pattern recognition techniques.

2.2.1.3 Dedicated Hardware

The performance of algorithms may be increased by recurring to dedicated hardware. The capability of GPU on mobile devices opens a new era for mobile computing and can enable many computationally demanding computer vision algorithms on mobile devices [34, 47, 154].

An implementation of the SIFT descriptors that incorporates the powerful graphics processing unit in mobile devices is described in [137]. By methodically partitioning the computation, compressing the data for mem-

ory transfers, and taking into account the unique challenges that arise out of the mobile GPU, the proposed method is able to achieve a speedup of 4-7 times over an optimized CPU version. Additionally, the energy consumption is reduced by 87% per image.

In [105] is described a face tracking approach that uses efficient gray-scale invariant texture features, namely Local Binary Patterns, and boosting. The GPU is used in the pre-processing and the feature extraction phase. An hybrid CPU-GPU ray tracer is designed in [114] to provide a realistic three-dimensional visualization on mobile devices. This ray tracer exploits the availability of CPU and GPU architectures to fully support reflection, refraction, hard shadows, and dynamic scenes. The GPU may be also used in the context of building panorama images in the reconstruction computations [13]. Object recognition and match processing have been hardware accelerated to achieve a 20 times speed-up in the context of augmented reality in [95].

2.2.2 Optimization of Descriptor Size

In the mobile environment the size of data may represent a severe issue. When the database is small, it can be stored on the phone and image retrieval algorithms can be run locally. When the database is large, it has to be placed on a remote server and retrieval algorithms are run remotely. In each case, feature compression (Sect. 2.2.2.1) is key to decreasing the amount of the data transmitted, and thus, reducing network latency. A small descriptor (Sect. 2.2.2.2) also helps if the database is stored on the mobile device. The smaller the descriptor, the more features can be stored in limited memory.

2.2.2.1 Compression Schema

Several compression schemes have been proposed to reduce the bit rate of SIFT descriptors for computer vision applications [23]. In the context of mobile vision, other methods have been proposed.

For mobile image retrieval, efficient data transmission may be achieved by sending only the query features, which are composed by the descriptor and the location within the image. The first is used to find candidate matching images, while the second is used as a geometric consistency check. In [168] is investigated how to compress the location information and how lossy compression affects the geometric consistency check. The location information is converted into a location histogram and a context-based arithmetic coding with location refinement method is then proposed to code the histogram. The proposed compression scheme achieves 12.5 times rate reduction compared to the floating point representation.

Image retrieval pipelines are usually based on bag-of-words matching, where the original order in which features are extracted from the image is discarded. Consequently, a set of feature from a query image can be transmitted in any of the $m!$ possible orderings. A coding scheme based on digital search trees [22, 29] reduces the size of a set of features by approximately $log_2(m!)$. The Type Coding schema for compressing distributions [21] is proposed as an improvement with respect to digital search trees. Optimal Entropy Constrained Vector Quantization code-books allow to perform as SIFT being 16 times smaller.

Typically, descriptors are extracted based solely on the visual content of a query, and the location cues from the mobile end are rarely exploited. In [88] is presented a Location Discriminative Vocabulary Coding (LDVC) scheme, which achieves extremely low bit rate query transmission, discriminative landmark description, as well as scalable descriptor delivery in a unified framework. The first contribution is a compact and location discriminative visual landmark descriptor, which is offline learned in two step.

First, spectral clustering is adopted to segment a city map into distinct geographical regions, where both visual and geographical similarities are fused to optimize the partition of cityscale geo-tagged photos. Second, two schemas to learn LDVC in each region are proposed: i) a Ranking Sensitive PCA and ii) a Ranking Sensitive Vocabulary Boosting. Both schemes embed location cues to learn a compact descriptor, which minimizes the retrieval ranking loss by replacing the original high-dimensional signatures. The second contribution is a location aware online vocabulary adaption. A single vocabulary is stored in the mobile end, which is efficiently adapted for a region specific LDVC coding once a mobile device enters a given region.

Another example that exploits location information is [109]. A temporally coherent keypoint detector, and design efficient interframe predictive coding techniques for canonical patches and keypoint locations are proposed. The goal is to transmit each patch with as few bits as possible by simply modifying a previously transmitted patch. This enables server-based mobile augmented reality where a continuous stream of salient information, sufficient for image-based retrieval and localization, can be sent over a wireless link at a low bit-rate. This technique achieves a similar image matching performance at 1/15 of the bit-rate when compared to detecting keypoints independently frame-by-frame.

The work described in [65] deals with the problem of city scale on-device mobile visual location recognition. An efficient vector quantization strategy is designed by combining the Transform Coding and Residual Vector Quantization. The visual descriptors can then be compressed into only a few bytes while providing reasonable searching accuracy. Another work in this area by performing the feature quantization on the device and transferring compressed bag of words vectors to the remote server [149]. To cope with the limited processing capabilities of mobile device, the quantization of high dimensional feature descriptors has to be performed at very low

complexity. To this end, the novel Multiple Hypothesis Vocabulary Tree is proposed. The probability of assigning matching features descriptors to the same visual words is increased by introducing an overlapping buffer around the separating hyperplanes to allow for a soft quantization and an adaptive clustering approach. This result in a 10 times faster query time with respect to descriptors quantized using a k-means tree.

2.2.2.2 Low Size Descriptor

While a compression scheme allows to reduce the size of standard descriptors, a better approach is to design a descriptor with compression in mind. Of course, such a descriptor still has to be robust and highly discriminative at low bit rates. Ideally, it would permit descriptor comparisons in the compressed domain for speedy feature matching. Further, training step should be avoided so that the descriptor is not dependent on any specific data set. Finally, the compression algorithm should have low complexity so that it can be efficiently implemented on mobile devices. To meet all these requirements simultaneously, the Compressed Histogram of Gradients [20] has been proposed. It is built by using a histogram-of-gradient descriptor and by explicitly exploiting the anisotropic statistics of the underlying gradient distributions. This allows to apply quantization and compression scheme that work well for distributions to produce compact descriptors, that are 16 times shorter than standard SIFT.

In [25, 27, 28] is presented a new architecture for searching a large database directly on a mobile device, which can provide numerous benefits for network-independent, low-latency, and privacy-protected image retrieval. A key challenge for on-device retrieval is storing a large database in the limited RAM of a mobile device. To address this challenge, a new compact, discriminative image signature called the Residual Enhanced Visual Vector (REVV) is developed. This is optimized for sets of local features which are fast to extract on mobile devices. REVV outperforms existing

compact database constructions in the mobile visual search setting and attains similar retrieval accuracy in large-scale retrieval as a Vocabulary Tree that uses 25 times more memory. Fast on-device search with REVV enables the system to achieve latency around 1 second per query regardless of external network conditions. The compactness of REVV allows it to also function well as a low-bitrate signature that can be transmitted to or from a remote server for an efficient expansion of the local database search when required.

The work in [59] presents the i-SIFT descriptors, obtained by applying the Informative Feature Approach on SIFT descriptors. The i-SIFT approach tackles three key bottlenecks in SIFT estimation: i-SIFT will i) improve the recognition accuracy with respect to class membership, ii) provide an entropy sensitive matching method to reject non-informative outliers and more efficiently reject background, iii) obtain an informative and sparse object representation, reducing the high dimensionality of the SIFT keypoint descriptor and thin out the number of training keypoints using posterior entropy thresholding.

Chapter 3

Fast and Effective Ellipse Detection on Mobile Devices

One of the most important challenges in computer vision is to run computer vision algorithm on mobile device. The porting of an algorithm to a mobile device is more than a mere translation of the code or adaptation to a new operating system. It usually require to carefully evaluate the design choices of the algorithm, and, if necessary, optimize the code or design a brand new lightweight algorithm.

Aside from the inherently slower hardware compared to desktop computers, it is very important to obtain a real time response: the user expects to interact with the application running the algorithm, and is not willing to wait too long for it to finish.

If it is possible to speedup computationally intense algorithms in a traditional environment, through hardware enhancement or by leveraging on

some kind of parallelism, it is not feasible to adopt the same solutions on mobile device. The optimization must carefully optimize computationally intense procedures, usually recurring to GPU computational capabilities, or design new procedures to accomplish the same task with less computation or memory requirements. In order to guarantee real time response, usually newly designed procedures rely on heuristic or approximated results. Particular attention must be paid in provide at least the same qualitative results as the original algorithm.

Ellipse detection is a problem which falls in this category. The estimation of the 5 parameters of the elliptic curve is very computationally intense and memory greedy. Even if ellipse detection has a long tradition in pattern recognition, an efficient solution is still missing. Several papers addressed ellipse detection as a first step for several computer vision applications, but most of the proposed solutions are too slow to be applied in real time on large images or with limited hardware resources. From the early algorithms based on the Hough transform several improvements have been proposed, in order to minimize computation or memory requirements, or to provide more accurate detection. However, despite some attempts to speed up the computation by leveraging on the GPU or by provide a new formulation, ellipse detection is still not feasible in real time, while achieving the same detection accuracy.

In this chapter a novel algorithm specifically designed to run on a mobile device for fast and effective ellipse detection is proposed. Thorough experiments on synthetic data and large and challenging real images datasets show that the proposed algorithms outperforms state of the art methods, being as much as, or even more, accurate and running in real time even on a mobile device. The proposed algorithm relies on an innovative selection strategy of arcs which are candidate to form ellipses and on the use of Hough transform to estimate parameters in a decomposed space. The final aim of this solution is to represent a building block for new generation

of smart-phone applications which need fast and accurate ellipse detection also with limited computational resources.

3.1 Introduction to Ellipse Detection

The recognition of geometrical shapes formally described by a mathematical model has a long tradition in pattern recognition. The attempts of finding new efficient solutions for detecting parametric shapes in noisy and cluttered images resulted in successful algorithms for lines and circles. They are based on accumulations / voting procedures (e.g. Hough–based methods), interpolation, curve fitting, and so on.

Similarly, the detection of ellipses has been often addressed in the past, although ellipses are more complex parametric curves due to the larger number of parameters. Ellipse detection is the starting point for many computer vision applications, since elliptical shapes are very common in nature and in hand-made objects. For instance, ellipse detection can be used in wheels detection [40], road sign detection and classification [156], object segmentation for industrial applications [163], automatic segmentation of cells from microscope imagery [163], pupil/eye tracking [158], and many more. With the advent of powerful mobile technologies for everyone and the spreading of new generations of smart-phones, the request of new applications running on these devices increased enormously. Despite their limitations, mobile devices are now powerful enough to enable on-board processing of complex data, including images and videos, allowing unprecedented capabilities in terms of applications. As a consequence, the scientific community has recently found large interest to image/video processing on smart-phones, often called *embedded* or *mobile vision* [80]. Possible applications range from real time object/person tracking [180], content-based retrieval of framed scene with markerless object recognition [53], face detection [132] (and possibly recognition), blind people aid for

movement [67], etc.. As such, even though the advances in technology and multi-core processors will allow ever faster computation, keeping ellipse detection fast has the merit of allowing ever more complex computer vision applications.

Consequently, in this paper we present a new solution for very fast ellipse detection in real images. We choose arcs belonging to the same ellipse very quickly and very reliably by working at arc-level, instead of pixel-level, and by relying an innovative selection strategy. The ellipse center is then estimated exploiting the property of the midpoints of parallel chords, and remaining parameters are estimated accumulating votes in a decomposed parameter space. The good trade-off between efficiency (in the order of 10 ms per image) and accuracy makes this approach a proper candidate for implementation on mobile devices.

3.2 Related Works

The importance of ellipse detection in image processing is witnessed by the large amount of works present in the literature.

Most of the methods for ellipse detection rely on the Hough Transform - HT (or its variants) to estimate the parameters. Since an ellipse is analytically defined by five parameters, these methods try to overcome the main problem of a direct application of standard HT, which is a 5D accumulator. McLaughlin *et al.*[112] rely on the randomized version of HT (RHT) and aim at reducing the memory usage using proper data structures. Lu *et al.*[107] iteratively focus only on the points with higher probability to belong to the a single ellipse, thus reducing the parameter space to five 1D accumulators. A very common and memory efficient approach has been proposed by Xie *et al.*[183] and Chia *et al.*[37], where four parameters are geometrically computed, estimating in a 1D accumulator the last one. Basca *et al.*[8] speeded up the method of Xie *et al.*using RHT, thus consid-

ering only a small random subset of the initial pairs of points. HT-based methods greatly suffer from noise (which includes both background noise and points belonging to different ellipses) which "dirties" the accumulator. Also, they are computationally intense and rather slow because of the voting procedure on a huge number of edge points combinations.

Instead of reducing the space dimensionality by means of strong assumptions, Aguado *et al.*[2] propose a decomposition of the parameter space: parameters are estimated in consecutive steps, leveraging previous results. Zhang *et al.*[194] avoid unnecessary computations for those combinations of points that can not lie to the same ellipse boundary by carefully selecting starting edge points.

Other approaches rely more heavily on the symmetry between the points on the boundary. Some methods [79, 98, 191] first find and analyze symmetry axes, estimating parameters using the HT, others [182] instead rely on symmetric relationships among boundary points and then adopt a least square fitting method.

All aforementioned methods start the estimation from sets of points, eventually selected according to some kind of geometric constraints. However, when considered unrelated to its neighbors, an edge pixel does not contribute significantly to a correct ellipse detection. A better characterization could be achieved using sets of connected edge pixels, i.e. *arcs*, which can be generated by linking short straight lines [36, 40, 90, 101, 108], splitting the edge contour [72, 104, 116, 126], or validating connected edge pixels [129]. The ellipses parameters are obtained using ellipse fitting methods [51, 127] on a reduced set of arcs, which are obtained grouping arcs according to their relative position and constraints on the curvature [40, 90, 101, 116, 126], or ellipse fitting error [36, 72, 104, 108, 129].

Most of the works present in the literature claim high detection accuracy. However, these results are validated mostly on a few synthetic images, and rarely on more than 10 real images, except [40, 126]. The exe-

cution time for methods that claim to be fast or real-time [40, 79, 90, 101, 108, 116, 194] has been computed on few images as well, and may increase significantly on different kind of images. In this paper we present a novel method (preliminary works can be found in [54, 55] for ellipse detection that results to be much faster than other state-of-the-art methods, while achieving similar or even better detection performance. We also present two annotated datasets (available on-line) of real images on which we tested both fast and effective methods for a fair evaluation.

3.3 Method Description

We present a novel algorithm for fast ellipse detection designed for real-time performance on real world images. It first selects combination of arcs belonging to the same ellipse and then estimates its parameters via the Hough Transform in a decomposed parameter space. Let us first describe the overall procedure, as outlined in Fig. 3.1.

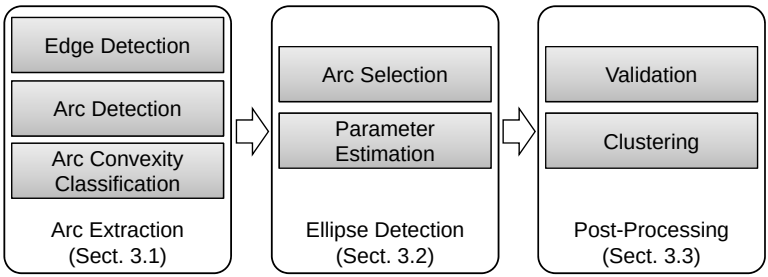


Figure 3.1: Flowchart of the algorithm.

As first step arcs are extracted from the edge mask and classified in four classes according to their convexity. We classify edge pixels in two main directions according to their gradient phase and group 8-connected edge pixels in the same direction class to form *arcs*. Their quality is also im-

proved removing short or straight arcs. Arcs are then classified according to their convexity, computed in a robust and efficient way. By combining the two classifications it is possible to assign each arc to a quadrant, in analogy with the final configuration in a Cartesian plane as depicted in Fig. 3.2(c). The method is tailored for the detection of visible ellipses, defined as having the boundary partially visible in at least three quadrants. Consequently we search for combinations of three arcs, called *triplets*, each belonging to a different quadrant. To avoid the combinatorial explosion, we select only triplets formed by arcs that satisfy three criteria based on convexity, mutual position and same pairwise estimated center. A selected triplet forms a *candidate* ellipse and, already knowing its center, we estimate the remaining three parameters in a decomposed Hough space requiring three 1D accumulators. Candidate ellipses are then validated according to the fitness of the estimation with the actual edge pixels. Since an ellipse may be supported by different triplets, multiple detections with slightly different parameters can be generated. We deal with multiple detections using a fast clustering procedure in the parameter space.

The following subsections will describe the different phases of the algorithm in detail.

3.3.1 Arc Extraction

In this phase we extract arcs from the input image, first by detecting edge points, then grouping them in arcs, and finally classifying arcs based on edge direction and convexity.

3.3.1.1 Edge Detection

Like most, also the proposed method begins analyzing the edge points, where every edge point $e_i = (x_i, y_i, \theta_i)$ is defined by its position x_i, y_i and the phase of the gradient θ_i. An edge detector is applied on the input

image in order to obtain a set of edge points. We chose the Canny edge detector [18] with automatic thresholding[1] because of its properties of good detection, good localization and minimal response. In the detected edge points the gradient phase θ_i is obtained through the Sobel operator (already computed in the Canny algorithm).

3.3.1.2 Arc Detection

Every edge point e_i is classified by the function $\mathcal{D} : e_i \rightarrow (+, -)$ into two main directions according to θ_i (Fig. 3.2(a)). We do not require accurate values of θ_i, and the classification $\mathcal{D}$ is simply done by checking the signs of the Sobel derivatives dx and dy:

$$\mathcal{D}(e_i) = \text{sign}(\tan(\theta_i)) = \text{sign}(dx) \cdot \text{sign}(dy) \tag{3.1}$$

We discard edge points lying on the classification boundary, i.e. with horizontal ($dy = 0$) or vertical ($dx = 0$) gradient direction. An *arc* α^k is formed by linking connected edge points of the same direction class:

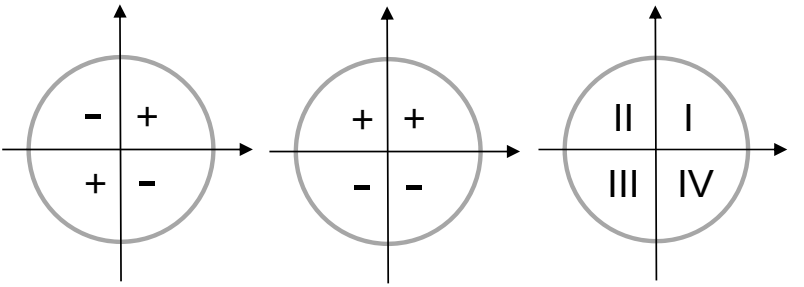
$$\alpha^k = \{(e_1^k, \ldots, e_{N^k}^k) : \mathcal{D}(e_i^k) = \mathcal{D}(e_j^k), \forall i, j \wedge \mathit{Connected}\left(e_{i-1}^k, e_i^k\right)\} \tag{3.2}$$

where N^k represents the number of edge points belonging to the arc α^k and $\mathit{Connected}\left(e_{i-1}^k, e_i^k\right)$ verifies the 8-connectivity of two consecutive edge points.

By extension, we define the arc direction $\mathcal{D}(\alpha^k)$ as the direction class of its points. We also define $L^k \equiv e_1^k$ as the *left* end point of α^k, $R^k \equiv e_{N^k}^k$ as the *right* one, $M^k \equiv e_{\lceil N^k/2 \rceil}^k$ as the *middle* edge point, BB^k as the bounding box having L^k and R^k as non adjacent vertices, and OBB^k as the oriented minimum area rectangle [58] enclosing all edge points $e_i^k \in \alpha^k$.

Some arcs α^k are not salient enough to characterize an ellipse and are

[1] https://gist.github.com/egonSchiele/756833



(a) Directions $\mathcal{D}$ (b) Convexities $\mathcal{C}$ (c) Quadrants $\mathcal{Q}$

Figure 3.2: Functions $\mathcal{D}, \mathcal{C}, \mathcal{Q}$. See the text for further details on these functions.

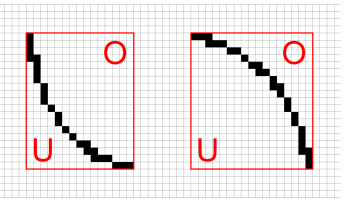


Figure 3.3: Convexity classification.

readily removed: very short arcs ($N^k < Th_{length}$) that are mainly due to noise and arcs containing mostly collinear points (shortest side of $\text{OBB}^k < Th_{obb}$), thus not belonging to the curved boundary of an ellipse. The values of these parameters are investigated in Sect. 3.4.2.

3.3.1.3 Arc Convexity Classification

Every arc α^k is classified by the function $\mathcal{C} : \alpha^k \rightarrow (+, -)$ as having the convexity upward $(+)$ or downward $(-)$ (Fig. 3.2(b)). By construction, all the points $e_i^k \in \alpha^k$ lie inside BB^k, and divide it in two regions: we call U^k the region *under* the arc, and O^k the region *over* it (Fig. 3.3) computed with Alg. 1. We find the convexity by comparing the areas of U^k and O^k:

Algorithm 1 Get the convexity $\mathcal{C}(\alpha)$ of the arc α, when $\mathcal{D}(\alpha) = +$. Swap $area\_O$ and $area\_U$ when $\mathcal{D}(\alpha) = -$.

```
function GetConvexity(Point α[])
    N ← α.length()
    left ← α[0]
    right ← α[N − 1]
    current_x ← left.x
    area_O ← 0
    for i = 1 → N do
        if α[i].x ≠ current_x then
            area_O ← area_O + |α[i].y − left.y|
            current_x ← α[i].x
        end if
    end for
    area_BB ← |right.x − left.x| ∗ |right.y − left.y|
    area_U ← area_BB − N − area_O
    if area_U > area_O then
        return +
    end if
    if area_U < area_O then
        return −
    end if
    if area_U = area_O then
        discard the arc
    end if
end function
```

$$\mathcal{C}(\alpha^k) = \begin{cases} + & \text{, if } Area(\mathrm{U}^k) > Area(\mathrm{O}^k) \\ - & \text{, if } Area(\mathrm{U}^k) < Area(\mathrm{O}^k) \end{cases} \tag{3.3}$$

Equal areas $Area(\mathrm{U}^k)$ and $Area(\mathrm{O}^k)$ implies that a meaningful convexity can not be determined (as in the case of inflexion points) and therefore the arc is discarded.

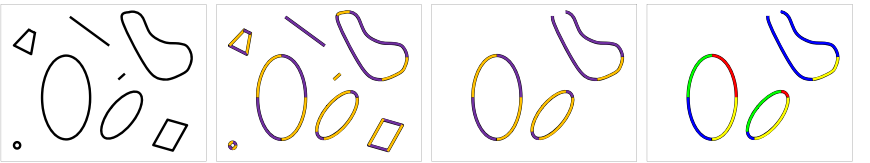
Given the functions $\mathcal{D}$ and $\mathcal{C}$, for every arc α^k we can define the function

$\mathcal{Q} : \alpha^k \rightarrow \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$, which maps α^k to its quadrant (Fig. 3.2(c)):

$$\mathcal{Q}(\alpha^k) = \begin{cases} \mathrm{I} & , \text{if } \langle \mathcal{D}(\alpha^k), \mathcal{C}(\alpha^k) \rangle = \langle +, + \rangle \\ \mathrm{II} & , \text{if } \langle \mathcal{D}(\alpha^k), \mathcal{C}(\alpha^k) \rangle = \langle -, + \rangle \\ \mathrm{III} & , \text{if } \langle \mathcal{D}(\alpha^k), \mathcal{C}(\alpha^k) \rangle = \langle +, - \rangle \\ \mathrm{IV} & , \text{if } \langle \mathcal{D}(\alpha^k), \mathcal{C}(\alpha^k) \rangle = \langle -, - \rangle \end{cases} \tag{3.4}$$

The chosen number of classes is four because it is the lowest number that allows to compute the convexity using Alg. 1, thus reducing the number of triplets combination and allowing to generate edge with enough curvature.

In order to better understand the arc detection phase, Fig. 3.4 reports a toy example on a synthetic image where the same color corresponds to the same direction or quadrant.



(a) Edge points. (b) Direction classification. (c) Removal of short and straight arcs. (d) Quadrants classification

Figure 3.4: Toy example of arc detection. Best viewed in colors.

3.3.2 Ellipse Detection

We define a *candidate* ellipse $\mathcal{E}_i$ as a *triplet*, i.e. a set of three arcs $\tau^{abc} = (\alpha^a, \alpha^b, \alpha^c)$ that satisfy a set of criteria and are, thus, likely to belong to the same ellipse. The parameters of each $\mathcal{E}_i$ are then computed in a HT framework.

3.3.2.1 Arc Selection Strategy

Given N_α as the number of arcs in the image, the set $\mathcal{T}^0$ of all possible triplets will contain $\binom{N_\alpha}{3}$ elements. This number could be extremely high and most of triplets are composed by arcs that do not belong to the same ellipse: a large amount of computation will be wasted to estimate parameters for false detections. The goal of the selection strategy is to generate a subset of triplets containing only triplets τ^{abc} whose arcs belong to the same ellipse boundary.

We define a *pair* of arcs as $p^{ab} = \left(\alpha^a, \alpha^b\right)$. Thus, a triplet can also be defined as two pairs sharing an arc: $\tau^{abc} = \left\{\left(p^{ab}, p^{dc}\right) | \, \alpha^b \equiv \alpha^d\right\}$. The selection strategy first selects pairs whose arcs satisfy constraints on *(i)* convexity and *(ii)* mutual position. Then it computes the ellipse center assuming that both arcs lie on the same ellipse boundary. Finally, it finds a candidate ellipse as a triplet composed by two pairs that *(iii)* imply the same center.

The first constraint on convexity ensures that each pair is composed by arcs in subsequent quadrants (in counterclockwise order):

$$\mathcal{A}(p^{ab}) = \begin{cases} \top & \text{, if } \left(\mathcal{Q}(\alpha^a), \mathcal{Q}(\alpha^b)\right) \in \{(\mathrm{I},\mathrm{II}), (\mathrm{II},\mathrm{III}), (\mathrm{III},\mathrm{IV}), (\mathrm{IV},\mathrm{I})\} \\ \bot & \text{, otherwise} \end{cases} \tag{3.5}$$

where "$\top$"and "$\bot$" stand respectively for *true* and *false*. The subset $\mathcal{T}^1 \subseteq \mathcal{T}^0$ is composed only of triplets whose pairs satisfy the first constraint:

$$\tau^{abc} \in \mathcal{T}^1 \iff \left(\tau^{abc} \in \mathcal{T}^0\right) \wedge \mathcal{A}(p^{ab}) \wedge \mathcal{A}(p^{dc}) \tag{3.6}$$

The second constraint on mutual position discards pairs whose arcs are incoherent with the same elliptic shape. It is defined, with reference to Fig.

3.5, by the boolean function:

$$\mathcal{M}(p^{ab}) = \begin{cases} \top \text{ , if } \big(\mathcal{Q}(\alpha^a), \mathcal{Q}(\alpha^b)\big) \equiv (I, II) & \wedge \big(L^a.x \gtrsim R^b.x\big) \\ \top \text{ , if } \big(\mathcal{Q}(\alpha^a), \mathcal{Q}(\alpha^b)\big) \equiv (II, III) & \wedge \big(L^a.y \gtrsim L^b.y\big) \\ \top \text{ , if } \big(\mathcal{Q}(\alpha^a), \mathcal{Q}(\alpha^b)\big) \equiv (III, IV) & \wedge \big(R^a.x \lesssim L^b.x\big) \\ \top \text{ , if } \big(\mathcal{Q}(\alpha^a), \mathcal{Q}(\alpha^b)\big) \equiv (IV, I) & \wedge \big(R^a.y \lesssim R^b.x\big) \\ \bot \text{ , otherwise} \end{cases} \tag{3.7}$$

where L and R are the leftmost and rightmost extrema of the arc, respectively, as defined in Sect. 3.3.1.2 and the inequalities $\gtrsim$ and $\lesssim$ are defined with a tolerance Th_{pos}, investigated in Sect. 3.4.2. The subset $\mathcal{T}^2 \subseteq \mathcal{T}^1$ contains only triplets whose pairs respect the first and the second constraints:

$$\tau^{abc} \in \mathcal{T}^2 \iff \big(\tau^{abc} \in \mathcal{T}^1\big) \wedge \mathcal{M}(p^{ab}) \wedge \mathcal{M}(p^{dc}) \tag{3.8}$$

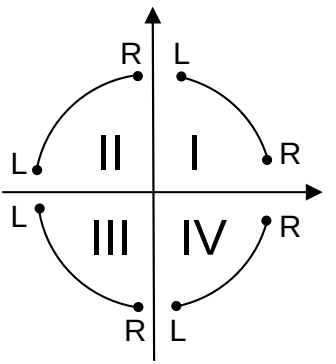


Figure 3.5: Mutual Position.

The third constraint verifies whether the three arcs $\alpha^a, \alpha^b, \alpha^c$ lie on the boundary of the same ellipse or, equivalently, that the centers C^{ab}, C^{dc}

implied by the pairs p^{ab}, p^{dc} of the triplet τ^{abc} are closer than a tolerance $Th_{centers}$ (see Sect. 3.4.2):

$$\mathcal{H}(\tau^{abc}) = \begin{cases} \top & \text{, if } C^{ab} \approx C^{dc} \\ \bot & \text{, otherwise} \end{cases} \tag{3.9}$$

The subset $\mathcal{T}^3 \subseteq \mathcal{T}^2$ contains only triplets whose pairs satisfy all constraints:

$$\tau^{abc} \in \mathcal{T}^3 \iff \left(\tau^{abc} \in \mathcal{T}^2\right) \wedge \mathcal{H}(\tau^{abc}) \tag{3.10}$$

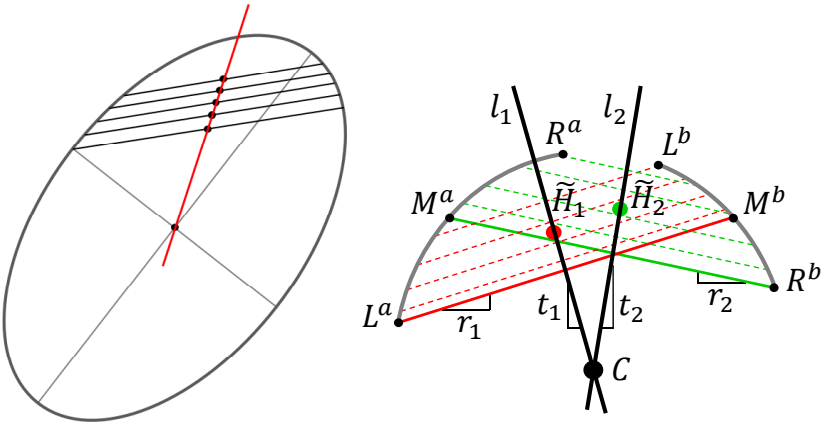
Each triplet $\tau_i^{abc} \in \mathcal{T}^3$ is a candidate ellipse $\mathcal{E}_i$.

First and second constraints are very fast to compute, and are still quite discriminative. The third is more complex, but is computed on a narrower set, and greatly improves the quality of candidate ellipses. More details on the performance of the selection strategy are available in Sect. 3.5.6.

3.3.2.2 Center Estimation

We estimate the ellipse center C^{ab} for a given arc pair p^{ab} by means of a well known geometric property of ellipses: *the midpoints of parallel chords are collinear* [191], as shown in Fig. 3.6(a). Thus, the intersection of two lines connecting the midpoints of two different sets of parallel chords is the ellipse center.

In order to find the center we need to generate two sets of parallel chords which are not parallel with each other. For clarity, we illustrate the procedure for the pair p^{ab} with reference to Fig. 3.6(b). We generate two sets of N_s chords, parallel respectively to the lines $\overline{L^a M^b}$ and $\overline{R^b M^a}$ (being M the midpoint of the arcs as defined in Sect. 3.3.1.2), having slope r_1^{ab}, r_2^{ab}. The influence of the parameter N_s on the overall performance is investigated in Sect. 3.4.2. We call H_1^{ab}, H_2^{ab} the two sets of their midpoints. The two lines l_1^{ab} and l_2^{ab} intersecting the points in H_1^{ab} and H_2^{ab} will intersect in the



(a) The midpoints of a set of parallel chords and its center are collinear.

(b) Geometric features to compute the center.

Figure 3.6: Method for estimating the ellipse center.

center C^{ab}. However, especially in real images, the points in H_1^{ab}, H_2^{ab} are affected by noise and are not perfectly collinear, and the lines l_1^{ab}, l_2^{ab} have to be robustly estimated.

We estimate the slopes t_1^{ab}, t_2^{ab} of the lines l_1^{ab}, l_2^{ab} using a fast variant of the robust Theil-Sen estimator (Alg. 2) [111] as the medians of the two sets of computed slopes S_1^{ab}, S_2^{ab}. We also generate the points $\tilde{H}_1^{ab}, \tilde{H}_2^{ab}$ whose coordinates are the medians of the coordinates of the points in H_1^{ab}, H_2^{ab}. We choose the median approach among other statistical measures to be consistent with the strategy of the Theil-Sen estimator, which adopts the median approach for its robustness to outliers.

Thus, given an arc pair p^{ab}, the coordinates of the center C^{ab} can be

Algorithm 2 Get the slope of the line best fitting the midpoints of parallel chords.

```
function GetSlope(Point midpoints[])
    middle ← midpoints.length()/2
    for i = 0 → middle do
        x1 ← midpoints[i].x
        y1 ← midpoints[i].y
        x2 ← midpoints[middle + 1 + i].x
        y2 ← midpoints[middle + 1 + i].y
        slope ← (y2 − y1)/(x2 − x1)
        S[i] ← slope
    end for
    return Median(S)
end function
```

computed as follows (see Fig. 3.6(b)) (superscripts omitted for clarity):

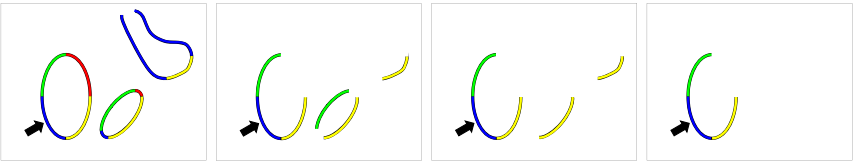
$$C.x = \frac{\tilde{H}_2.y - t_2\tilde{H}_2.x - \tilde{H}_1.y + t_1\tilde{H}_1.x}{t_2 - t_1} \tag{3.11}$$

$$C.y = \frac{t_1\tilde{H}_2.y - t_2\tilde{H}_1.y + t_1t_2(\tilde{H}_1.x - \tilde{H}_2.x)}{t_2 - t_1} \tag{3.12}$$

We say that the centers C^{ab}, C^{dc} of two pairs p^{ab}, p^{dc} coincide and satisfy the third constraint of the selection strategy (Eq. (3.9)) if they lie within a given distance $Th_{centers}$ (see Sect. 3.4.2) which accounts for image noise. Following the toy example reported in Fig. 3.4, Fig. 3.7 shows how the selection strategy works for the arc pointed by the arrow.

3.3.2.3 Parameter Estimation

The ellipse parameters are estimated only for those triplets $\tau^{abc} = \left(p^{ab}, p^{dc}\right)$ that satisfy the three selection strategy constraints. Since, by definition, the lines l_1, l_2 of the two pairs p^{ab}, p^{dc} should intersect the ellipse center, their pairwise intersection should always identify the center. However in



(a) $\alpha^b \equiv \alpha^d$ is the pointed arc. Find α^a and α^c.

(b) Constraint on convexity.

(c) Constraint on mutual position.

(d) Constraint on centers.

Figure 3.7: Toy example of selection strategy. Best viewed in colors.

noisy images this is rarely true. For a better estimation, we consider the ellipse center (x_c, y_c) as the median of the coordinates of a set of 7 points, consisting of the two centers C^{ab}, C^{dc}, their mean, and the other 4 intersection of the estimated lines $\{l_1^{ab} \cap l_1^{dc}, l_1^{ab} \cap l_2^{dc}, l_2^{ab} \cap l_1^{dc}, l_2^{ab} \cap l_2^{dc}\}$ as represented by gray points in Fig. 3.8.

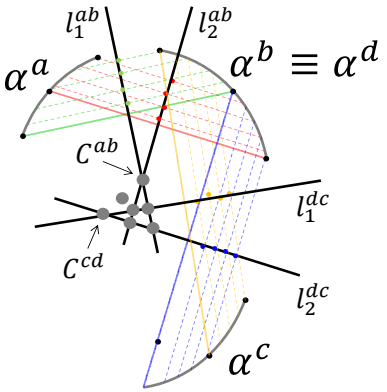


Figure 3.8: Estimated center of the ellipse.

In order to find the remaining parameters, the parameter space is decomposed as described in [2] in semi–axes ratio $N = B/A$ and orientation ρ. After the derivation of [2, 48, 194] the values of N and ρ are computed

(α^a, α^b)		(α^d, α^c)	
q_1	q_2	q_3	q_4
r_1^{ab}	$S_1^{ab}[s]$,$\forall s$	r_1^{dc}	$S_1^{dc}[s]$,$\forall s$
r_1^{ab}	$S_1^{ab}[s]$,$\forall s$	r_2^{dc}	$S_2^{dc}[s]$,$\forall s$
r_2^{ab}	$S_2^{ab}[s]$,$\forall s$	r_2^{dc}	$S_2^{dc}[s]$,$\forall s$
r_2^{ab}	$S_2^{ab}[s]$,$\forall s$	r_1^{dc}	$S_1^{dc}[s]$,$\forall s$

Table 3.1: Values to be assigned to q_1, q_2, q_3, q_4 to estimate N and ρ.

as:

$$N = \begin{cases} N_+ & \text{if } N_+ \leq 1 \\ 1/N_+ & \text{otherwise} \end{cases} \tag{3.13}$$

$$\rho = \begin{cases} \arctan(K_+) & \text{if } N_+ \leq 1 \\ \arctan(K_+) + \frac{\pi}{2} & \text{otherwise} \end{cases} \tag{3.14}$$

where:

$$\gamma = q_1 q_2 - q_3 q_4 \tag{3.15}$$

$$\beta = (q_3 q_4 + 1)(q_1 + q_2) - (q_1 q_2 + 1)(q_3 + q_4) \tag{3.16}$$

$$K_+ = \frac{-\beta + \sqrt{\beta^2 + 4\gamma^2}}{2\gamma} \tag{3.17}$$

$$N_+ = \sqrt{\frac{(q_1 - K_+)(q_2 - K_+)}{(1 + q_1 K_+)(1 + q_2 K_+)}} \tag{3.18}$$

One vote is accumulated for each combination of the parameters q_1, q_2, q_3, q_4, as reported by row in Table 3.1. Setting the values of q_1, q_3 to the slope of the parallel chords, we vary the values of q_2, q_4 with the slope of all lines computed by the Theil-Sen estimator (Alg. 2). The values of N and ρ are then the highest peaks in the two 1D accumulators.

Given the values N and ρ, the value of the major semi–axis A is:

$$A = A_x / \cos(\rho) \tag{3.19}$$

where:

$$x_0 = \frac{(x_i - x_c) + (y_i - y_c)K}{\sqrt{K^2 + 1}} \tag{3.20}$$

$$y_0 = \frac{-(x_i - x_c)K + (y_i - y_c)}{\sqrt{K^2 + 1}} \tag{3.21}$$

$$A_x = \sqrt{\frac{x_0^2 N^2 + y_0^2}{N^2 (K^2 + 1)}} \tag{3.22}$$

The value of A is estimated in a 1D accumulator considering as (x_i, y_i) every edge point e_i of the three arcs $\alpha^a, \alpha^b, \alpha^c$, and taking the highest peak. The value of the last parameter, minor semi–axis B, is then:

$$B = A \cdot N \tag{3.23}$$

3.3.3 Post–Processing

The selection strategy defines three necessary, but not sufficient, conditions for the detection of an ellipse. Candidate ellipses must then be validated to remove false detections. Also, multiple triplets may be found on the boundary of the same ellipse, generating multiple detections of the same ellipse which need to be merged. Since the number of candidate ellipses is much less than the number of arcs, merging multiple detections is more efficient than finding all the arcs lying on the boundary of a same ellipse at an early stage.

3.3.3.1 Validation

A quality measure is assigned to each candidate ellipse $\mathcal{E}_i$. Recalling that we can get the position of a point $(\overline{x}_i, \overline{y}_i)$ with respect to the boundary of $\mathcal{E}_i$ by means of the ellipse equation:

$$f(\overline{x}_i, \overline{y}_i, \mathcal{E}_i) = \begin{cases} = 1 & \text{, if } (\overline{x}_i, \overline{y}_i) \text{ is on the boundary of } \mathcal{E}_i \\ > 1 & \text{, if } (\overline{x}_i, \overline{y}_i) \text{ is outside the boundary of } \mathcal{E}_i \\ < 1 & \text{, if } (\overline{x}_i, \overline{y}_i) \text{ is inside the boundary of } \mathcal{E}_i \end{cases} \tag{3.24}$$

we can define the set $\mathcal{B} = \{(\overline{x}_i, \overline{y}_i) : |f(\overline{x}_i, \overline{y}_i, \mathcal{E}_i) - 1| < 0.1\}$ that contains the points that are close to the boundary. The score $\sigma \in [0, 1]$ summarizes how well the points of the three arcs composing $\mathcal{E}_i$ fit the boundary of the estimated ellipse:

$$\sigma = \frac{|\mathcal{B}|}{|\alpha^a| + |\alpha^b| + |\alpha^c|} \tag{3.25}$$

A candidate ellipse $\mathcal{E}_i$ with $\sigma > Th_{score}$ is considered as *valid*, otherwise as a false detection and is discarded.

3.3.3.2 Clustering

Multiple valid detections of the same ellipse are clustered by adopting a variant of the approach described in [128], which allows to assess the similarity of two ellipses $\mathcal{E}_i, \mathcal{E}_j$ comparing the distances between centers (eq.

(3.26)), axes (eq. (3.27) and (3.28)), and rotation (eq. (3.29)) separately:

$$\delta_c = \sqrt{(\mathcal{E}_i.x_c - \mathcal{E}_j.x_c)^2 + (\mathcal{E}_i.y_c - \mathcal{E}_j.y_c)^2} < \min\left(\mathcal{E}_i.B, \mathcal{E}_j.B\right) \times 0.1 \tag{3.26}$$

$$\delta_a = (|\mathcal{E}_i.A - \mathcal{E}_j.A| / \max(\mathcal{E}_i.A, \mathcal{E}_j.A)) < 0.1 \tag{3.27}$$

$$\delta_b = (|\mathcal{E}_i.B - \mathcal{E}_j.B| / \min(\mathcal{E}_i.B, \mathcal{E}_j.B)) < 0.1 \tag{3.28}$$

$$\delta_\rho = \begin{cases} \frac{\angle(\mathcal{E}_i.\rho - \mathcal{E}_j.\rho)}{\pi} < 0.1 & \text{, if } \left(\frac{\mathcal{E}_i.B}{\mathcal{E}_i.A} < 0.9\right) \wedge \left(\frac{\mathcal{E}_j.B}{\mathcal{E}_j.A} < 0.9\right) \\ 0 & \text{, otherwise} \end{cases} \tag{3.29}$$

Ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$ are considered as equivalent according to the function $\Delta : (\mathcal{E}_i, \mathcal{E}_j) \rightarrow (\top, \bot)$:

$$\Delta\left(\mathcal{E}_i, \mathcal{E}_j\right) = \bigwedge\left(\delta_c, \delta_a, \delta_b, \delta_\rho\right) \tag{3.30}$$

All valid ellipses are ordered by decreasing score and compared one by one with the center of each cluster by means of the function Δ. We consider the highest score ellipse as the center of a given cluster. If the current ellipse can not be assigned (i.e. is not equivalent) to any cluster, it becomes the center of a new cluster. The parameters are set according to [128].

3.4 Discussion on the method

This section summarizes the novel contributions of the proposed method, as well as the differences with other methods, and presents a thorough discussion on its parameters are presented.

3.4.1 Novelty and Comparison

Arc generation In the arc detection step (Sect. 3.3.1.2), the coarse gradient direction is computed by means of the Sobel derivatives. Other approaches, such as the Fast Line Extraction (FLE) of Kim *et al.*[91], approximate curved lines with short lines, and compute the gradient direction from the relationship between their starting and ending points. We compared the gradient computed using Sobel derivatives and FLE on the same data as in [91], i.e. various ellipses which have an axis ranging from 20 to 100 pixels, by fixing the other axis at 100 pixels. Figure 3.9 shows the error between the two methods and the direction mathematically computed. Both the average and the maximum errors of our method are smaller than FLE. A more challenging test is reported in Tab. 3.2, where, among others, we show the performance on real images of our method and its variant that computes the direction of the gradient relying on FLE, namely [91] + Ours. These results show that the use Sobel derivatives is both efficient and accurate enough.

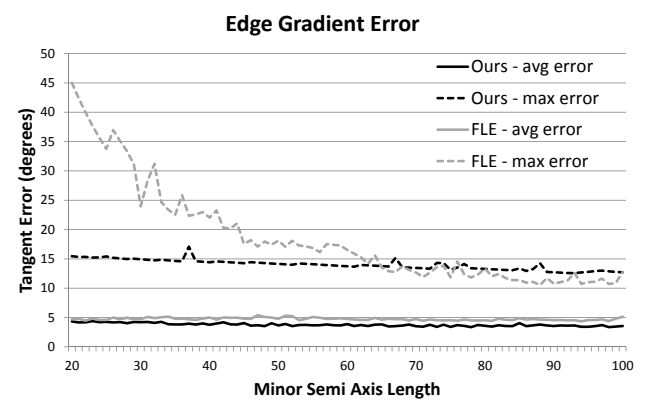


Figure 3.9: The maximum and average error of our method and Fast Line Extraction [91].

Arc detection Arcs are detected by labeling connected edge points within the same direction class, avoiding time-demanding refinements such as the search for inflexion points or sharp turn [116, 126], or junction points [72, 104]. On one hand, this procedure will affect the performances in synthetic datasets explicitly designed to test the robustness of algorithm in particular scenarios, as reported in Sect. 3.5.4. On the other hand, the simplicity of the algorithm allows to achieve in real–time (in the order of milliseconds) state-of-the-art results (or even better) on real images, as confirmed by the experiments reported in Sect. 3.5.5.

Convexity The convexity is computed by counting the number of pixels under the arc. Unlike the method of Zhang *et al.*[194], this algorithm (Alg. 1) is robust to thick edges and do not need angle estimation. Other methods to compute convexity can not be applied: the method of Guil *et al.*[68] is suitable only for concentric ellipses, while the method of Prasad. *et al.*[126] computes the convexity only in relation with another arc.

Selection strategy constraints There are several works in the literature that adopt criteria to reduce the search space. However, we present a new set of efficient and effective criteria: *(i)* straight arcs are removed by thresholding the shortest side of their oriented bounding box; *(ii)* arcs are selected according to their convexity at arc level, rather than edge point level as in [194]; arcs are also selected according to their *(iii)* mutual position and *(iv)* implied center, without the need to compute a search region as in [126] or relying on ellipse fitting algorithms to find the center first [101].

Center computation The ellipse center is computed exploiting the property of the midpoints of parallel chords [191]. It does not require angle estimations and is thus well suited for real world images, where the edges and gradient directions may be very noisy. This property has been already

exploited in [19, 79, 191] to compute the center, first by finding symmetric points through horizontal and vertical scans in the image, and then by computing the pairwise intersection of the symmetric axes estimated via the HT. The proposed method presents several improvements: *(i)* parallel chords are not bounded to be horizontal or vertical; *(ii)* the procedure is performed on pair of arcs instead of the whole image, thus *(iii)* the two symmetric axes are computed by a simple line estimator, and *(iv)* there is only one intersection, i.e. the center. The center may be computed also relying on other methods, which however have some drawbacks for real time processing of real images. The property of the tangents [150, 193] requires accurate edge gradient estimation. RANSAC [108] is robust to outliers but is not guaranteed to find the optimal solution or to converge within a given time. Circle fitting [40, 90] or ellipse fitting [36, 72, 101, 104, 116, 126, 129] methods are dependent on the amount of noise.

Parameter Estimation The ellipse parameters are estimated as soon as a valid triplet is selected, clustering at a later stage multiple detections, if any. This procedure is opposed to every other method, where the grouping procedure aims at retrieving first all arcs that belong to the same ellipse, and then estimates its parameters. The proposed formulation overcomes the limitation of [2, 194] to use 2D accumulators: it is optimized for the detection of a single ellipse at a time, thus needing only 1D accumulators to estimate rotation, axes ratio and major semi–axis length.

Clustering By testing the similarity between two ellipses in the parameters space, the clustering method is much faster than evaluating their overlap as in [126], and the function Δ is more accurate than thresholding the Euclidean distance in the parameter space as in [8]. Differently from [128], the test on the center is related to the ellipse size, instead of the image size: as a results, ellipses with same parameters are clustered similarly

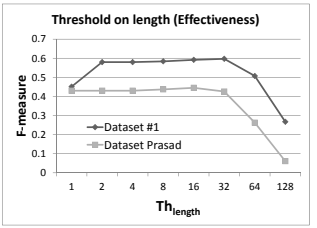
regardless the image size.

3.4.2 Parameter selection

The proposed method is influenced by 5 parameters, Andrea: which may be considered too many, making the algorithm tuning hard. However, their effect on the performance, both detection effectiveness and execution time, is discussed, showing either their negligible influence on the performance or an easy procedure to tune them. The parameters are tested on two datasets, detailed in Sect. 3.5.5, namely Dataset Prasad proposed by Prasad *et al.*[126] and Dataset #1 proposed in this paper.

Spurious edge and noise removal During the preprocessing, as discussed in Sect. 3.3.1.2, we remove short and straight arcs, that are mainly due to noise and do not have enough curvature to contribute to the detection of an ellipse. We define as short those arcs shorter than Th_{length} (parameter #1). The influence of the parameter Th_{length} is investigated in Fig. 3.10, where it is clear that either including all edges (left side of the graph) or removing too many edges (right side) has negative impact on the detection effectiveness. The best results are obtained with values of Th_{length} between 2 and 32. However, as clearly depicted in Fig. 3.10(b), values lower than 8 are very time demanding. As a result, we set $Th_{length} = 16$.

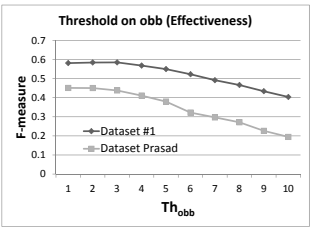
Edges are defined as straight if the shortest side of the respective oriented bounding box is smaller than Th_{obb} (parameter #2). Figure 3.11 shows the influence of this parameter on the performance. Both the detection effectiveness (Fig. 3.11(a)) and the execution time (Fig. 3.11(b)) decrease with values greater than 3 (Fig. 3.11(a)). We set $Th_{obb} = 3$ as a good trade-off.



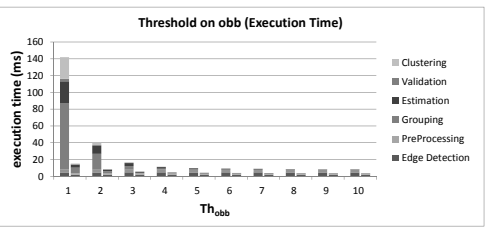
(a) F-measure.

(b) Execution time on Dataset #1 (left column) and Dataset Prasad (right column)

Figure 3.10: Performance varying Th_{length}.



(a) F-measure.

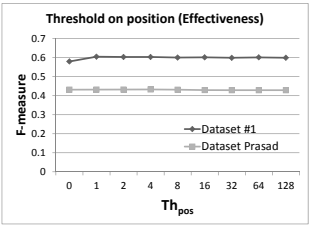


(b) Execution time on Dataset #1 (left column) and Dataset Prasad (right column)

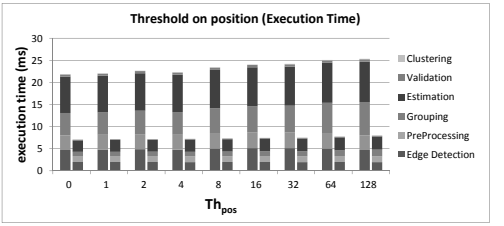
Figure 3.11: Performance varying Th_{obb}.

Mutual Position The second constraint of the selection strategy (Sect. 3.3.2.1) forms arc pairs only for arcs with coherent mutual position, which is computed analyzing the relationships among the extrema of the arcs, with a tolerance Th_{pos} (parameter #3). In Fig. 3.12 we show that the impact of Th_{pos} on the method is negligible. We set $Th_{pos} = 1$ to gain some effectiveness (Fig. 3.12(a)) and keep the execution time at the minimum (Fig. 3.12(b)).

Distance between estimated centers As discussed in Sect. 3.3.2.2, the third constraint of the selection strategy discards arc pairs whose com-



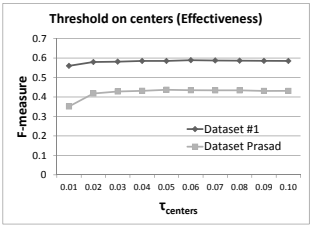
(a) F-measure.



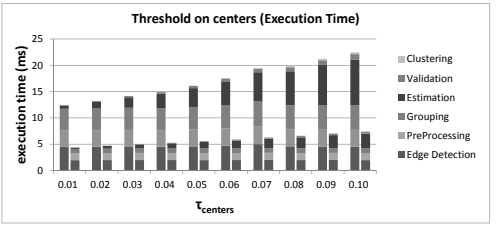
(b) Execution time on Dataset #1 (left column) and Dataset Prasad (right column)

Figure 3.12: Performance varying Th_{pos}.

puted centers are more distant than $Th_{centers} = \tau_{centers} \times$ image diagonal. As shown in Fig. 3.13, for $\tau_{centers}$ (parameter #4) greater than 0.02 there is no significant improvement in the detection effectiveness (Fig. 3.13(a)), but the computational time increases because more triplets are considered as valid (Fig. 3.13(b) shows that the increased time is due to the estimation step). A value of $\tau_{centers} = 0.05$ that guarantees top effectiveness and still fast execution time is then selected.



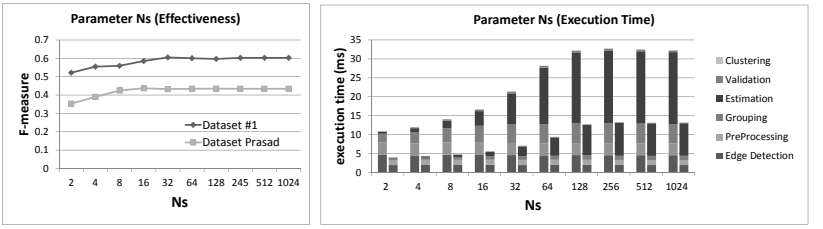
(a) F-measure.



(b) Execution time on Dataset #1 (left column) and Dataset Prasad (right column)

Figure 3.13: Performance varying $\tau_{centers}$, where ($Th_{centers} = \tau_{centers} \times$ image diagonal).

Number of Parallel Chords As described in Sect. 3.3.2.1, our method finds N_s (parameter #5) chords parallel to a given one. Considering all chords starting from each edge point of the arc α^a, i.e. $N_s = |\alpha^a|$, can be very time consuming and not useful to better locate the ellipse center. A more efficient approach is to consider fewer chords, starting from a subset of edge points sampled at regular intervals on the arc, to minimize the execution time and still allow a good estimation of the center. Figure 3.14(b) shows that the time used for the evaluation step increases with N_s, while the other steps of the algorithm are basically stable. Moreover, Fig. 3.14(a) shows that parameter N_s does not need to be greater than 32 to achieve the top performance. In general, the best tradeoff between accuracy and speed is obtained with $N_s \in [8, 32]$, that allows to avoid unnecessary computation and still to estimate accurately the parameters by using only the points on the selected arcs. Consequently, we set $N_s = 16$.



(a) F-measure.

(b) Execution time on Dataset #1 (left column) and Dataset Prasad (right column)

Figure 3.14: Effectiveness and execution time varying N_s.

3.5 Experimental Results

We perform a thorough set of tests of the proposed method, on both synthetic and real images datasets, evaluating its performance and comparing it with other state-of-the-art methods.

3.5.1 Evaluation metrics

We evaluated the performance of the algorithms in terms of *execution time* and *detection effectiveness*, according to the evaluation methodology of [126]. Detected and ground-truth ellipses are compared according to the following overlap ratio:

$$D = 1 - \frac{\text{count}(\text{XOR}(\mathcal{E}_1, \mathcal{E}_2))}{\text{count}(\text{OR}(\mathcal{E}_1, \mathcal{E}_2))} \tag{3.31}$$

where $\mathcal{E}_1$ and $\mathcal{E}_2$ are respectively the ground-truth ellipse and a valid detection. If the detected and the ground-truth ellipses have overlap ratio $D > D_0$, with $D_0 = 0.95$ for synthetic images and $D_0 = 0.8$ for real images, then a match is counted (true positive, TP); if a ground-truth ellipse does not have a match with any of the detected ellipses, then a miss is found (false negative, FN); finally, if the detected ellipse does not have a match with any of the ground-truth ellipses, it corresponds to a false positive (FP). According to these definitions the *detection effectiveness* is computed in terms of the F-measure.

3.5.2 Other methods

In order to show the performance of the proposed method, we compare it in terms of both execution time and detection effectiveness against other state-of-the-arts methods. Since the main feature of the proposed method is the ability to run in real–time still achieving good detection results, we selected among the methods reported in Sect. 3.1 the fastest and the most effective ones:

- the work of Basca *et al.*[8], which applies directly the RHT, randomly selecting an initial subset among all possible point pairs;

- the work of Zhang *et al.*[194], which, after applying a point selection

strategy, estimates the parameters in a decomposed parameter space in multiple steps;

- the work of Libuda *et al.*[101], which iteratively links small edges into arcs, till it obtains an elliptic shape;
- the work of Prasad *et al.*[126], which groups arcs according to edge curvature and convexity.

All methods are implemented in C++, except the one of Prasad *et al.*which is in Matlab. The methods of Zhang *et al.*and Basca *et al.*have been reimplemented according to the respective papers, while the source code of Libuda *et al.*and Prasad *et al.*is available on-line.

In the experiments on real datasets (Sect. 3.5.5.1) we also evaluate three variants of our method to better motivate our choices:

- instead of relying on the Sobel derivatives to compute the gradient direction for edge pixels, we adopt the Fast Line Extractor of Kim *et al.*[91];
- instead of estimating the ellipse parameters via the Hough Transform in a decomposed space, we used the direct least square ellipse fitting algorithm of Fitzgibbon *et al.*[51];
- same as the previous case, but using the unconstrained, non–iterative least square based geometric ellipse fitting method "Ellifit" of Prasad *et al.*[127].

Their code is available on-line. The Fast Line Extractor is in C++, while the two ellipse fitting algorithms are in Matlab. In order to guarantee a fair comparison, the execution time of the programs in Matlab is scaled by a factor[1]. All experiments were executed without code parallelization on

[1] After testing the execution time of a set of functions in C++ and Matlab, we found a scale factor of 50 to be a good average approximation.

a PC with an Intel Core i7. Our method was also executed on a Samsung Galaxy S2.

3.5.3 Robustness to rotation, axes ratio and size

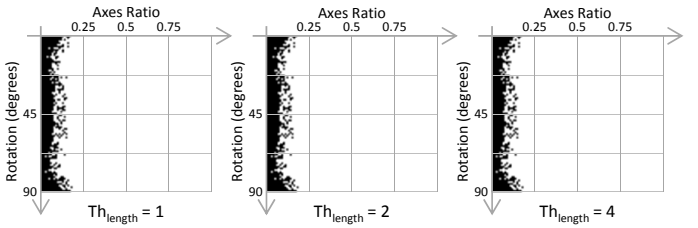
To investigate the working limits of the evaluated algorithms with respect to ellipse rotation, axes ratio and axes size, we created two datasets composed of automatically–generated synthetic images of size 400×400, each containing a single ellipse without noise.

The first dataset contains 9100 images with fixed parameters $x_c = y_c = 200$ and $A = 100$, with B varying so that the axes ratio B/A ranges from 0 to 1 (step of 0.01) and orientation ρ ranging from 0° to 90° (step of 1°). The results of this evaluation are reported in Fig. 3.15 so that a pixel in the image is white if the ellipse with corresponding axes ratio (along horizontal axis) and orientation (along vertical axis) has been correctly detected, or black otherwise.

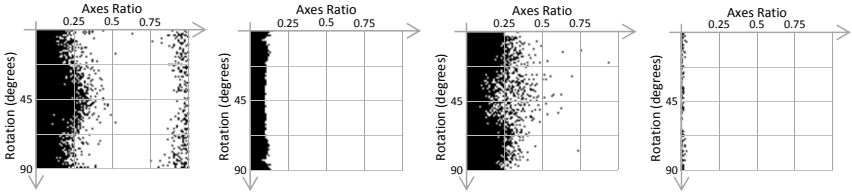
The second dataset contains 10000 images with fixed parameters $x_c = y_c = 200$ and $\rho = 30^\circ$, with A ranging from 1 to 100 (step of 1) and B varying so that the axes ratio B/A ranges from 0 to 1 (step of 0.01). The results of the evaluation are reported in Fig. 3.16.

As expected, all the algorithms i) are basically robust to orientation (no significant vertical asymmetry is noticeable in Fig. 3.15), ii) fail when the axes ratio is near to 0, thus when ellipses degenerate into straight lines (leftmost part of graphs in both Fig. 3.15 and Fig. 3.16), and iii) have difficulties detecting small ellipses (top part in Fig. 3.16), i.e. when ellipse boundaries are composed by very few pixels. Ellipses on the first synthetic dataset are large enough and the detection is insensitive to the variation of the parameter Th_{length} (Fig. 3.15(a)).

The method of Prasad *et al.*[126] performs extremely good on these datasets, being very robust to ellipse orientation, axes ratio and size, followed by the the method of Basca *et al.*[8], which performs really good



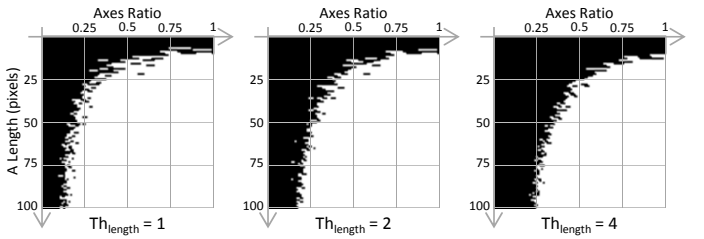
(a) Ours, varying Th_{length}



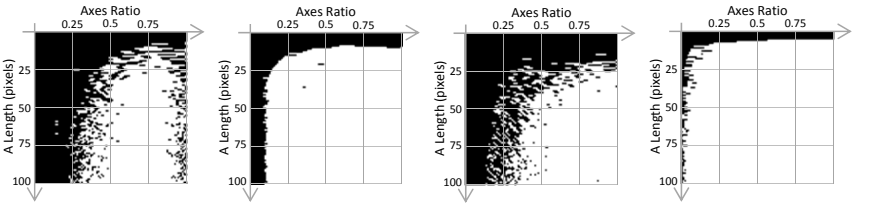
(b) Zhang [194] (c) Basca [8] (d) Libuda [101] (e) Prasad [126]

Figure 3.15: Working conditions, with respect to rotation (vertical axis, from 0° to 90°) and axes ratio B/A (horizontal axis, from 0 to 1).

in single ellipse, noise-free images. The method of Zhang *et al.*[194] fails in the detection of most of the ellipses. This is caused by its selection strategy, which is not able to correctly identify the ellipse center in case of too few samples which do not generate a peak in the 2D accumulator and, consequently, other parameters are wrongly estimated. Moreover, this method has some difficulties in the detection of ellipses with axes ratio near to 1 (rightmost part of Fig. 3.15(b) and Fig. 3.16(b)), i.e. ellipses degenerating into circles. Since most applications aim at detecting ellipses because they are a perspective transformation of circles, the non–detection of circles could represent an issue. The method of Libuda *et al.*[101] and ours have almost similar working conditions. Ours, however, appears to be more robust, while the high number of constraints in [101] can not deal with some configurations of edge points. As shown in Fig 3.16(a) varying



(a) Ours, varying Th_{length}



(b) Zhang [194] (c) Basca [8] (d) Libuda [101] (e) Prasad [126]

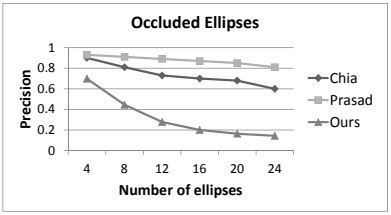
Figure 3.16: Working conditions, with respect to major semi-axis length (vertical axis, from 1 to 100) and axes ratio B/A (horizontal axis, from 0 to 1).

the parameter Th_{length}, the algorithm is able to detect small ellipses, but at the cost of more execution time (see Sect. 3.4.2).

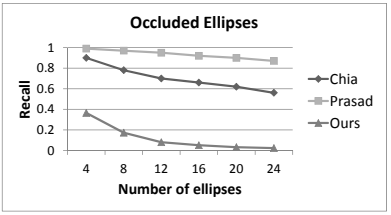
3.5.4 Dataset of Chia *et al.*

We report the tests on the synthetic datasets proposed in [36]. These datasets include three challenges: occluded ellipses, overlapped ellipses, and ellipses affected by salt–and–pepper noise. Being an edge linking method, we focused on the first two datasets as in [126], because the proposed method needs connected edge pixel to work and consequently is not suited to handle salt-and-pepper noise. Moreover, in case of salt-and-pepper noise, a simple pre-processing such the application of a median filter will remove the noise. The datasets consist of images of size $\lambda \times \lambda$, in which each im-

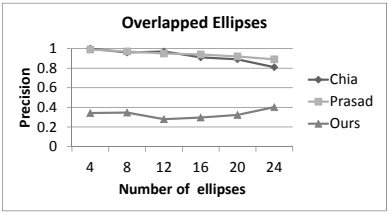
age contains η different ellipses, with $\lambda = 300$ and $\eta \in \{4, 8, 12, 16, 20, 24\}$. The parameters of the ellipses, arbitrarily located within the image, are generated randomly with the value of the semi-axes in $[\lambda/30, \sqrt{\lambda^2 + \lambda^2}/2]$. Figure 3.17 shows the performance of our method, as well as the results obtained by Chia *et al.*[36] and Prasad *et al.*[126]. These two methods perform really well on these datasets: their careful selection of edge segments to be split or merged handles consistently cases of occlusion and overlapping. The proposed method, instead, aims at simplifying the arc extraction in order to run in real-time in real world images. This limitation is highlighted by the poor performance obtained on such challenging, yet synthetic datasets.



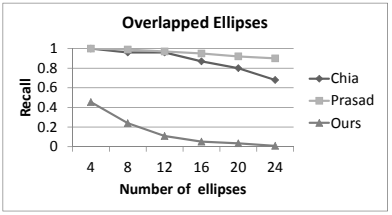
(a) Precision



(b) Recall



(c) Precision

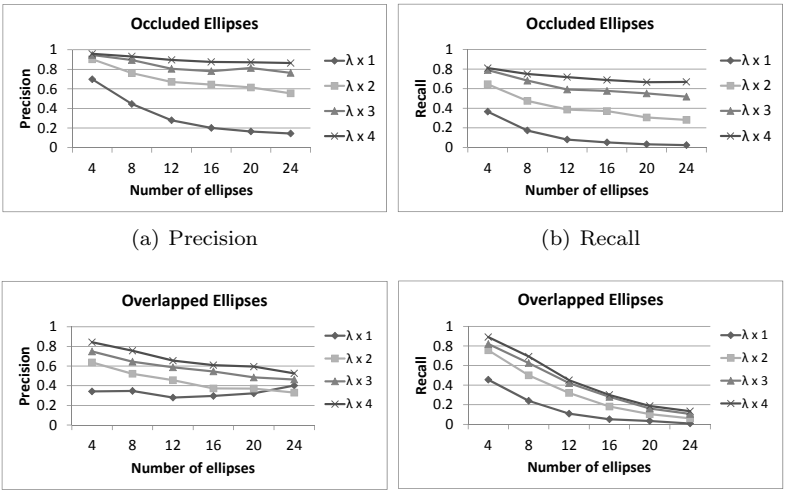


(d) Recall

Figure 3.17: Evaluation on Dataset Chia [36].

These results are due to the fact that in such small images the number of edge pixels for each ellipse is very limited. In order to ground this motivation to our poor performance, the original datasets have been upscaled

by multiplying the parameter λ by a factor of $1, 2, 3, 4$. As a consequence, ellipses are larger but still present the same challenges as in the original datasets. The results shown in Fig. 3.18 demonstrate that our method significantly improves the performance on the new datasets, confirming that in the presence of enough information the proposed method is able to detect very well overlapped or occluded ellipses.



(a) Precision (b) Recall

(c) Precision (d) Recall

Figure 3.18: Evaluation on Dataset Chia [36] of the proposed method increasing the value of λ.

3.5.5 Real Datasets

Since the goal of the proposed method is to work in real time in real scenarios, we tested the methods on three datasets containing real world images: a portion of the dataset used in Prasad *et al.*[126] called "Dataset Prasad" hereinafter, and two datasets we created collecting a total of 1029

images.

Dataset Prasad is composed by the portion of data that are still available on-line of the dataset used in [126], which consists of 198 images out of the original 400.

Our Dataset #1 is composed of 400 real images containing elliptic shapes, collected from MIRFlickr and LabelMe repositories. The images from the first repository are high quality and mainly focused on a single object, while images from the second one have lower resolution, are more noisy and represent scenes containing different objects.

The major reason for the development of the proposed method is the capability to run in real–time on embedded devices such as smart-phones. As a consequence, we created Dataset #2 collecting several videos using a Samsung Galaxy S2 and selecting a total amount of 629 frames at the resolution of 640x480, typical of most smart-phone applications. Ellipse detection on this kind of images is very challenging due to varying lighting conditions and images blurred by motion and autofocus.

All images in Dataset #1 and Dataset #2 have been manually annotated and are publicly available on-line [1] for future comparisons. Regarding Dataset Prasad we relied on the given ground truth.

Figures 3.19 and 3.20 show some statistics about the datasets. On one hand, in Dataset Prasad the number of small ellipses (i.e. with semi-major axis length shorter than 20 pixels) is very high compared to Dataset #1 and Dataset #2. On the other hand, in Dataset #1 and Dataset #2 most of the images contain few ellipses, while in Dataset Prasad the number of ellipses per image is more uniformly distributed.

3.5.5.1 Results on Real Datasets

We evaluated the effectiveness of the selected methods on the three datasets collecting the F-measure varying the threshold on the score (Th_{score}) of

[1] `http://imagelab.ing.unimore.it/imagelab/ellipse/ellipse_dataset.zip`

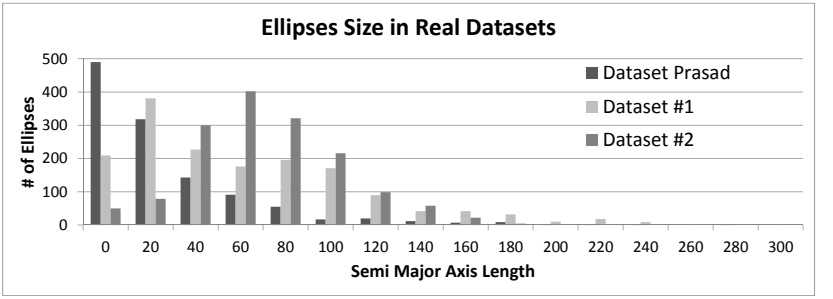


Figure 3.19: Number of ellipses per length of the major semi–axes in real datasets.

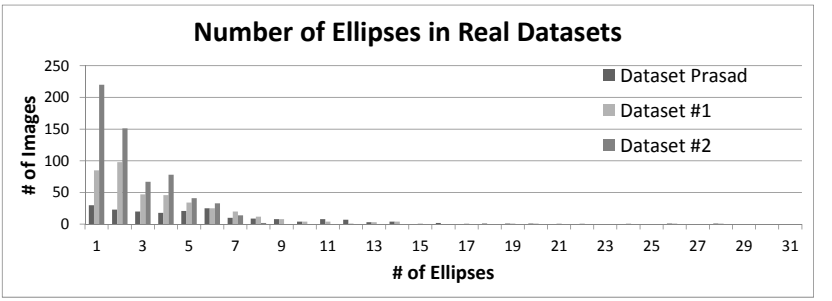


Figure 3.20: Images per number of ellipses in real datasets.

the detected ellipses for each image, as reported in Fig. 3.21 for Dataset Prasad, Fig. 3.22 for Dataset #1 and Fig. 3.23 for Dataset #2. We reported the highest F-measure value in Table 3.2. Our method results to be very effective in the detection of ellipses in real images.

In order to evaluate the speed, also reported in Table 3.2, for Dataset Prasad and Dataset #1 we collected the execution time while running all methods on a PC, while for Dataset #2 we collected the execution time while running our method on a smart-phone, namely a Samsung Galaxy S2. For this purpose, we developed an Android and OpenCV application which calls the C++ function for detecting ellipses via the Java Native Interface

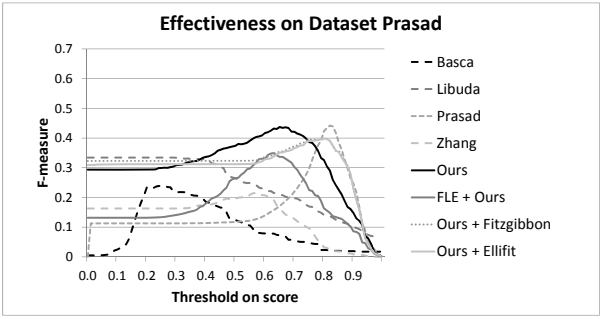


Figure 3.21: Graph reporting the F-measure varying Th_σ on Dataset Prasad.

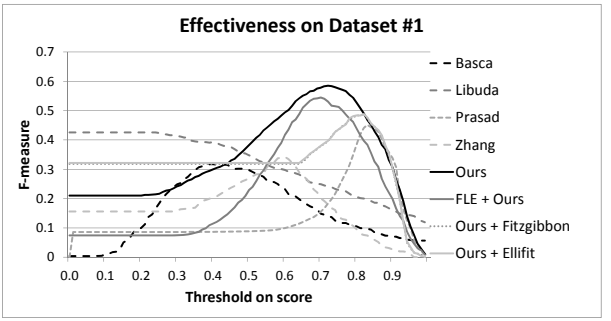


Figure 3.22: Graph reporting the F-measure varying Th_σ on Dataset #1.

(JNI). Since, it was impossible to test the accuracy in real–time directly on the smart-phone due to the lack of a ground truth, all frames have been annotated and the effectiveness test was performed off-line on the PC. The execution time of our method, instead, was collected while running the application directly on the smart-phone filming the same scenes.

Dataset Prasad We show the execution time breakdown for each processing step for all the compared algorithms in Table 3.3. The method of Prasad *et al.*[126] performs the best in terms of effectiveness (F-measure

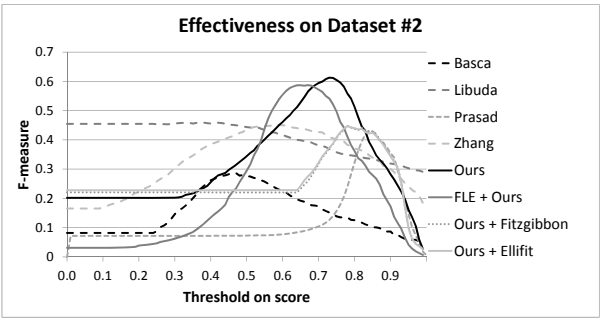


Figure 3.23: Graph reporting the F-measure varying Th_σ on Dataset #2.

	Dataset Prasad		**Dataset #1**		**Dataset #2**	
	F-measure	Time	F-measure	Time	F-measure	Time
[8]	23.98%	134.98	31.77%	684.31	28.84%	N/A
[101]	33.47%	7.85	42.58%	18.95	45.88%	N/A
[126]	**44.18%**	158.32	45.12%	1084.85	43.28%	N/A
[194]	21.53%	431.95	34.21%	5591.5	45.06%	N/A
Ours	43.70%	**5.56**	**58.52%**	**15.96**	**61.13%**	**45.82**
[91] + Ours	34.04%	8.39	54.44%	25.98	58.72%	N/A
Ours + [51]	39.72%	7.02	48.93%	22.77	44.86%	N/A
Ours + [127]	39.78%	7.43	48.82%	20.44	44.70%	N/A

Table 3.2: Average effectiveness and execution time (in milliseconds) on the three datasets. N/A = implementations on mobile devices not available.

of 44.18%), but results to be quite slow. Most of the time is spent in the grouping step; also its clustering method based on the overlap ratio is very slow. Our method is the second one in terms of effectiveness (F-measure of 43.70%, very close to Prasad *et al.*[126]) and is the fastest with an execution time of 5.56 ms. Our poorer performance is mainly due the fact that this dataset contains a large amount of small ellipses, as shown in Fig. 3.19. Due to its working limits and to the trade-off considered in the parameter setting, our method is not able to detect about the 38% of the ellipses present in this dataset. However, the reported F-measure value demonstrates that our method is very effective in the detection of mid-sized

	Basca	Libuda	Prasad	Zhang	Ours	Kim Ours	Ours Fitzibbon	Ours Prasad
Edge Detection	2.28	2.21	2.30	2.33	1.96	2.29	2.36	2.36
Pre-Processing	1.85	2.42	30.97	1.95	1.33	4.20	1.44	1.44
Grouping	0.00	2.44	67.48	0.04	0.84	0.81	1.47	1.47
Estimation	119.92	0.78	1.68	427.60	1.26	0.99	1.09	1.06
Validation	0.00	0.00	0.13	0.00	0.14	0.09	0.30	0.31
Clustering	10.92	0.00	55.76	0.02	0.03	0.01	0.36	0.78
Total	134.97	7.85	158.32	431.95	**5.56**	8.40	7.02	7.43

Table 3.3: Execution time breakdown on Dataset Prasad.

or large ellipses (where the major semi axis A is larger than about 10-15 pixels). Moreover, this dataset contains also ellipses that are so distorted that are not detected by any other methods, as depicted in third line of Fig. 3.24. The variants of our method that rely on an ellipse fitting algorithm perform also quite well, while the implementation with the Fast Line Extractor is less effective. The method of Libuda *et al.*[101] has good execution time but quite low detection performance. The methods of Basca *et al.*[8] and Zhang *et al.*[194] have very poor overall performance: most of the time is spent to estimate the parameters for a large number of edge pixel combinations.

Dataset #1 In Table 3.4 we report the execution time breakdown for all the algorithms. The two methods that work directly on single edge points,

	Basca	Libuda	Prasad	Zhang	Ours	Kim Ours	Ours Fitzgibbon	Ours Prasad
Edge Detection	5.93	5.92	4.66	5.23	4.54	5.09	5.45	5.45
Pre-Processing	5.77	5.47	102.81	4.68	3.14	10.22	3.11	3.11
Grouping	0.00	6.44	366.30	0.33	4.23	5.94	4.29	4.29
Estimation	611.05	1.11	4.48	5581.25	3.56	4.29	9.57	6.80
Validation	0.00	0.00	0.32	0.00	0.40	0.43	0.34	0.41
Clustering	61.56	0.00	606.28	0.04	0.07	0.01	0.29	0.36
Total	684.31	18.95	1084.85	5591.55	**15.96**	25.99	23.06	20.44

Table 3.4: Execution time breakdown on Dataset #1.

i.e. Zhang *et al.*[194] and Basca *et al.*[8], have again the worst performance. The selection strategy of Zhang *et al.*requires more computation, but leads to more accurate solutions with respect to random point selection of Basca *et al.*that suffers the amount of noise present in real images. The method of Libuda *et al.*[101] confirms to be very fast. The method of Prasad *et al.*[126] has good performances, but its arc extraction and grouping procedure are very time demanding. Our algorithm and its variants achieve the best performances in terms of both effectiveness and execution time.

Dataset #2 The results (see Table 3.2) regarding the effectiveness on Dataset #2 confirm the consideration reported for the other datasets. The method of Prasad *et al.*[126] is quite accurate, but not as much as in synthetic scenarios or its own dataset. The method of Libuda *et al.*[101] confirms average results, and the methods based on single edge pixels [8, 194] still have very low performance. Our method has the highest F-measure value, and its variants achieve good performance as well. The average execution time on the smart-phone is 45.82 ms, demonstrating the real–time performance of the proposed method.

Some results of the tested methods on the three datasets are depicted in Fig. 3.24, 3.25 and 3.26, respectively.

3.5.6 Selection Criteria

The goal of the selection strategy is to discard as soon as possible those triplets whose arcs do not lie on the same ellipse boundary. This impacts both speed and detection effectiveness, avoiding further computation for triplets which are actually false detections. As clarified in Sect. 3.3.2.1, each step of the selection strategy selects a narrower subset: $\mathcal{T}^3 \subseteq \mathcal{T}^2 \subseteq \mathcal{T}^1 \subseteq \mathcal{T}^0$. The size of each subset and its ratio with respect to $\mathcal{T}^0$ are reported in Table 3.5. These values, computed as the average for all images in Dataset Prasad and Dataset #1, demonstrate the effectiveness of the

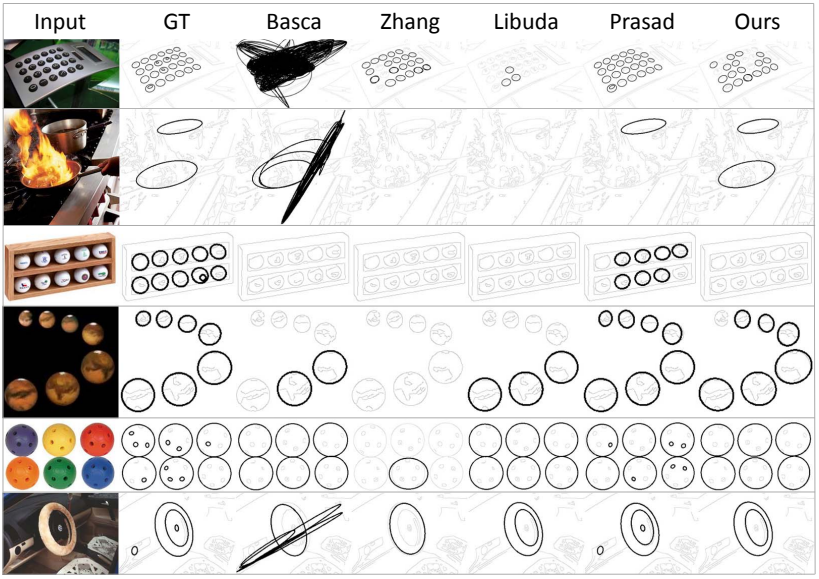


Figure 3.24: Results on Dataset Prasad.

selection strategy criteria.

Set	Dataset Prasad		Dataset #1	
	Avg. # triplets	% triplets	Avg. # triplets	% triplets
$\mathcal{T}^0$	44514	100.00 %	261019	100.00 %
$\mathcal{T}^1$	15600	35.04 %	86221	33.03 %
$\mathcal{T}^2$	2702	6.07 %	11450	4.38 %
$\mathcal{T}^3$	112	0.25 %	323	0.12 %

Table 3.5: Average number of triplets after applying the constraints.

3.5.7 Known Limitations of Our Method

Our method relies on a very simple procedure to generate arcs. On one hand this guarantees a major speed–up, on the other hand it presents some drawbacks. The method is not able to split correctly arcs that presents

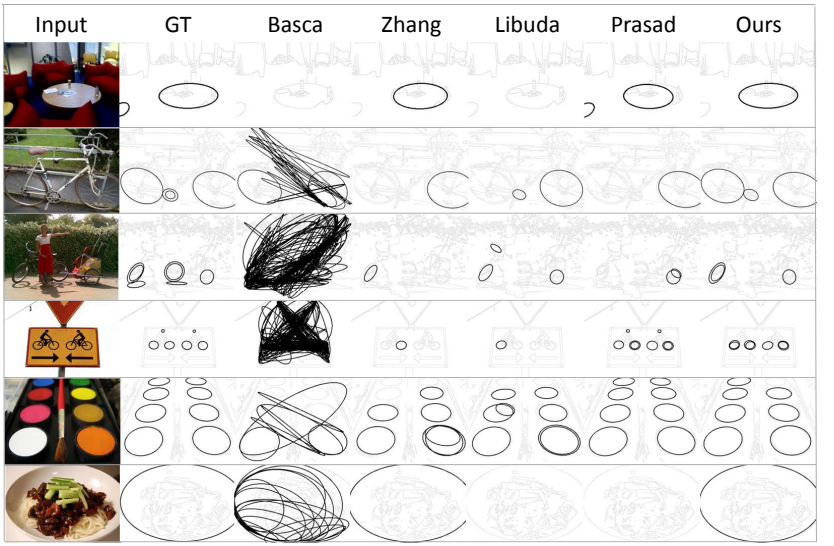


Figure 3.25: Results on Dataset #1.

inflexion or junction points, and it splits well shaped arcs spanning trough different quadrants. Small ellipses or ellipses with fragmented boundary are difficult to detect because arcs may result to be too short or without enough curvature. Also, when the ellipses have very low axes ratio arcs may be considered as straight lines and discarded as well. These limitations are highlighted by the poor performance on the synthetic datasets of Chia *et al.*in Sect. 3.5.4.

The selection strategy allows to significantly speed up the grouping procedure. However, it assumes that an ellipse has at least three arcs in different quadrants. This assumption does not always hold, and consequently some kind of occluded ellipses, such as well–shaped semi–ellipses, can not be detected.

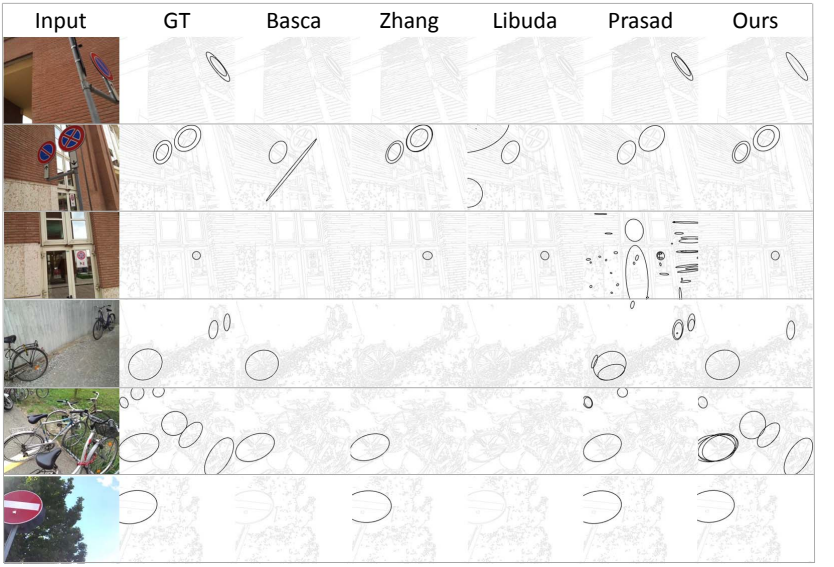


Figure 3.26: Results on Dataset #2.

3.6 Conclusions

In this chapter we proved that a very fast and accurate ellipse detection is feasible also with limited hardware resources as in the case of smartphones. The main point is to shift the focus of interest from edge points to arcs (or parts of arcs) and, instead of providing an exhaustive search, to start selecting only arcs compatible with an elliptical shape. Our approach has been extensively analyzed (with particular focus on the influence of the most critical parameter) and compared with four state-of-the-art methods, resulting superior than them on real images in terms of trade-off between detection effectiveness and execution time.

Despite its performance, our approach is based on several assumptions which can lower its effectiveness compared with other methods in particular conditions. However, based on our experiments, the cases on which

these assumptions do not hold are not frequent in real images and the overall improvement in terms of efficiency is worth this slight detection loss, especially when aiming to a real-time implementation on a mobile device.

Chapter 4

Visual Place Recognition

The diffusion of powerful mobile devices has posed the basis for new applications implementing directly on the devices sophisticated computer vision and pattern recognition algorithms. Because of the natural interaction of the user with its smart phone, a new set of applications is being developed. A very popular question is to know what are we looking at, and get information about it. The ubiquity of mobile device equipped with high quality cameras provide the opportunity to naturally get information about what we are filming.

Research in visual place recognition aims at solving this problem: by shooting a picture with the camera of a mobile device, retrieve all information regarding what is in the picture. Many application may take advantage from this technology, like tourism, advertising, and so on.

In this chapter is described the implementation of a complete system for automatic recognition of places localized on a map through the recognition of significant signs by means of the camera of a mobile device. Novel classification algorithm based on the innovative use of bag-of-words on ORB features are proposed. The recognition is achieved using a simple

yet effective search scheme which exploits GPS localization to limit the possible matches. This simple solution brings several advantages, such as the speed also on limited-resource devices, the usability also with limited training samples and the easiness of adapting to new training samples and classes. The overall architecture of the system is based on a REST-JSON client-server architecture. The experimental results have been conducted in a real scenario and evaluating the different parameters which influence the performance.

4.1 Introduction to Mobile Visual Search

Mobile vision is a rather recent research field aiming at developing sophisticated computer vision algorithms on board of mobile devices. More specifically, this field addresses the use of commercial smart phones and tablets as embedded devices, thanks to their increasing capabilities to process complex data (such as images and videos) in real time and to the availability of several good-quality sensors on board.

This research field must not be considered as a mere re–engineering task of existing algorithms on a different hardware platform, since it poses several challenges which require a complete re–design of the algorithms themselves and can be of interest for the scientific community: the limited computational and memory resources available, the extreme mobility and limited connectivity, and the power constraints under which these tetherless devices operate.

The use of mobile devices as people-centric sensors and processing units opens to new and impressive classes of applications, ranging from real time object/person tracking, content-based retrieval of framed scene with marker-less object recognition, face detection (and possibly recognition), blind people aid for movement, etc.. This paper addresses an application with the final aim of recognizing logos for localization purposes. The

proposed application needs to identify precisely a place where the user is located, supposing the GPS-based localization is not available or reliable (both for limited precision and for the co-existence of several significant locations in the neighborhood). This identification can be used for city marketing (similarly to the "check in" used in apps like Foursquare) or for gaming purposes. Other sensors may be employed too (e.g., the gyroscope to understand the direction the user is looking at), but precise localization is still required. Moreover, the place identification through image analysis allows the system to work properly also indoor (e.g., in a mall) where GPS-based localization is unreliable.

With these premises, this paper proposes a computer vision algorithm which recognizes significant locations in real-time. The algorithm is based on ORB features which are then quantized with a newly-proposed bag-of-words (BoW) model for binary descriptors. At the best of our knowledge, this is the first time a BoW model for a binary descriptor (as ORB is) is used in the context of logo recognition. Moreover, to take into account the aforementioned bandwidth limitations, the BoW histograms are compressed using an efficient and effective compression algorithm. Server side, this compressed descriptor is used to match with the trained classes in a database and the resulting ranking is returned to the user on the device. Experiments on real images with challenging logos are reported.

4.2 Related Works

The problem we are addressing is somehow similar to logo recognition [7, 125], in that we aim at recognizing the location depicted in the query image by means of appearance. Logo recognition solutions rely on segmented logos or brands and learn particular configurations of local descriptors to match with the query [139, 144]. Since in our case the segmentation is not available, we consider the images as a whole and search for visually similar

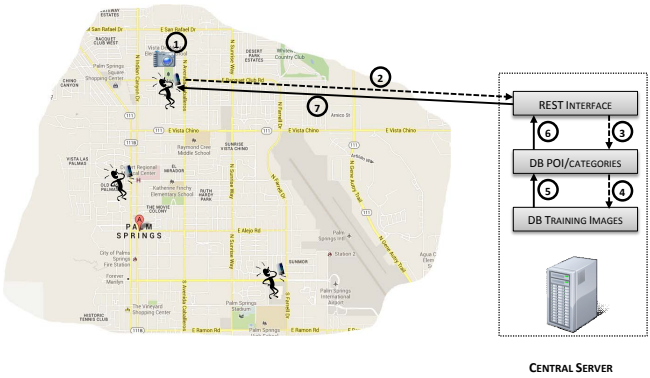


Figure 4.1: Overall description of the client-server architecture.

images, treating the problem as an image matching one [64]. However, beside the lack of segmentation, these methods are not directly applicable because we do not have large datasets for the comparison [151]. Also, they are in general computationally demanding and thus not suited for user interaction.

Most of the methods mentioned above rely on feature detection and matching. The SIFT keypoint detector and descriptor [106], although over a decade old, have proven to be remarkably successful in a number of applications using visual features, including object detection and recognition, image stitching, scene classification, etc. This descriptor has been also extended to color images in the form of RGB-SIFT, Opponent-SIFT and C-SIFT, as described by van de Sande *et al.* in [172]. However, it requires an intensive computational effort, especially for real-time systems, or for low-power devices such as cellphones. This has led to an increased research for replacements with simpler descriptors with lower computation demands. This trend started with SURF [10], but since then a lot of other descriptors have been proposed in literature, always focusing not only on performance but also on speed: we can refer to VLAD [86], BRIEF [16],

DAISY [165] among the most recent proposals. There has also been research aimed at speeding up the computation of SIFT, most notably with GPU devices [155], or the exploitation of approximate nearest neighbor techniques, starting from LSH [82] up to product quantization [87].

Also a great variety of global features has been proposed to tackle the retrieval problem, for example color histograms, GIST [119] and HOG [41]. Usually these features are easier to compute and do not require the quantization step typical of the bag-of-words model which is necessary to create a global representation (an histogram of visual words) from the aforementioned local descriptors.

In a recent paper, Rublee *et al.*[143] propose a very fast binary descriptor based on BRIEF, called ORB, which is rotation invariant and robust to noise. They demonstrate through experiments how ORB is up to two orders of magnitude faster than SIFT, while performing as well in many situations. The efficiency is tested on several real-world applications, including object detection and patch-tracking on a smartphone. The investigation of variance under orientation was critical in constructing ORB and decorrelating its components, in order to get good performance in nearest-neighbor applications. An interesting aspect is that the authors have also contributed a BSD licensed implementation of ORB to the community, via OpenCV 2.3. We provide a short description of the ORB descriptor in Section 4.4.1.

4.3 System Overview

The overall sketched architecture of the proposed system is shown in Fig. 4.1. The system follows a standard client-server architecture, where the client side is composed of the users' mobile devices, while the server side is a central computer devoted to keep the databases updated and to provide the logo classification. When dealing with mobile vision (and mobile

computing in general) there is always a crucial tradeoff between the local (on-device) computation and the remote processing. When moving towards the latter, the communication overhead required to send the raw data to the remote server can be unacceptable, especially because of the cost of transmission of the device (not-flat rates) and the delay introduced to transmit large amount of data. On the opposite side, local processing requires a sufficiently-powerful device to avoid long waits and it consumes a lot of battery.

In our system, the balance is reached as follows. First, the user will use the developed App to take a picture of a logo (phase 1 in Fig. 4.1). It is worth saying that the picture does not have to be perfect, and it can be at some degree blurred by the user's movement, it can contain small logos, and the logo can be acquired in a tilted, partial or occluded way, at least at a reasonable extent. The only strong requirement is that the picture does not contain other logos, at a resolution and quality superior to that of the searched logo. Once the picture has been taken, it is processed locally on the device to extract and compress, in real time, a set of features and the corresponding bag-of-words (BoW) descriptor (further details are reported in Section 4.4). The compressed descriptor is sent to the remote server (phase 2 in Fig. 4.1). In order to obtain en efficient communication between the clients and the server we used the REST (REpresentational State Transfer) architecture for distributed systems [50]. The main advantage is that REST architecture provides a simple interface with the server where the requests are sent through a HTTP GET with parameters passed through the URL. The server responds (phase 7 of Fig. 4.1) using the JSON (JavaScript Object Notation) format which codes the response as a formatted string.

In order to prepare the JSON response the server needs to identify the list of POIs (Points Of Interest) which potentially match with the logo in the picture taken by the users. This list is ordered based on the

score computed by the algorithm and returned to the user's device. This identification is achieved by matching the descriptor sent by the user with the descriptors of the trained class, as detailed in Section 4.4. First of all, the set of potential POIs (with corresponding categories) is retrieved by the database (phase 3 in Fig. 4.1). This retrieval can be narrowed with respect to the total number of POIs in the database by exploiting the rough GPS-based location of the device (which is sent together with the descriptor in phase 1), if available.

Given the set of potential POIs, the database containing the trained images for each one is queried (phase 4) and the server-side of the machine learning algorithm is activated (see Section 4.4.4). This algorithm returns (phase 5) the ordered list of POIs ranked according to a score, which is then sent back to the device through phases 6 and 7.

4.4 Logo Recognition through Bag-of-Words and ORB features

The system requires to efficiently extract some informative elements from the acquired images on the device, so the choice of the ORB descriptor (4.4.1) is straightforward and particularly effective, given also the availability of specific implementations for mobile devices. What we need is a way to quickly summarize these local binary descriptors, which also allows us to reduce the amount of information sent to the server. We employ two solutions for this task, leveraging a variation of the k-means algorithm (4.4.2) and a very simple, but effective compression scheme (4.4.3). Finally, server-side, we detect the image class, thus the sign queried by the user, by means of a simple ranking technique (4.4.4) which uses a subset of the training images, selected according to their GPS position.

4.4.1 ORB descriptor

The ORB descriptor (Oriented FAST and Rotated BRIEF) builds on the well-known FAST keypoint detector [142] and the recently-developed BRIEF descriptor [16].

The original FAST proposal implements a set of binary tests over a patch, by varying the intensity threshold between the center pixel and those in a circular ring around the center. The Harris corner measure [75] has been used to provide an evaluation of the corner intensity. In ORB the missing orientation information of FAST are instead complemented with Rosin's corner intensity [141]. In particular, the moment m_{pq} of a patch (region) R is computed as:

$$m_{pq} = \sum_{x,y} x^p y^q R(x, y). \tag{4.1}$$

We further compute the centroid $\mathbf{c}$ (boldface is used for vectors) as

$$\mathbf{c} = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right), \tag{4.2}$$

and by constructing a vector from the patch center to the centroid $\mathbf{c}$, we define the relative orientation of the patch as

$$\omega = \text{atan2}\left(m_{01}, m_{10}\right). \tag{4.3}$$

The patch description has been provided starting from the BRIEF operator [16], a bit string representation constructed from a set of binary intensity tests. Given a smoothed image patch R of an intensity image I, a binary test τ can be performed as:

$$\tau\left(R, \mathbf{u}, \mathbf{v}\right) = \begin{cases} 1: & R(\mathbf{u}) < R(\mathbf{v}) \\ 0: & R(\mathbf{u}) \geq R(\mathbf{v}) \end{cases}. \tag{4.4}$$

Given a set of intra-patch locations

$$L = \begin{pmatrix} \mathbf{u}_1, \ldots, \mathbf{u}_n \\ \mathbf{v}_1, \ldots, \mathbf{v}_n \end{pmatrix}, \tag{4.5}$$

the final feature $\mathbf{b}$ is defined as an n-dimensional vector of binary tests:

$$\mathbf{b}(R) = \sum_{1 \leq i \leq n} 2^{i-1} \tau(R, \mathbf{u}_i, \mathbf{v}_i). \tag{4.6}$$

The intra-patch locations for the tests influences the quality of the descriptor itself. A solution could be a grid–sampling–based set of sets by taking into consideration the patch orientation, so multiplying these locations with the rotation matrix. However, by analyzing the distribution of the tests, this solution brings a loss of variance and increase the correlation among the binary tests (since tests along the edge orientation statistically produce similar outcomes). This heavily impacts the descriptor effectiveness, describing redundancy more than distinctiveness. To solve the problem, the authors [143] employed a learning algorithm, sampling tests from 5×5 subwindows of the 31×31 patch window chosen for the descriptor, running each test against all training patches. The result is a predefined set of 256 tests called rBRIEF.

4.4.2 A Bag of Words Model for Binary Descriptors

While histogram based features are directly ready to be used in image classification or retrieval tasks, local features require an additional quantization step to be transformed into global image features. The classic approach is to employ k-means clustering using Euclidean distance between feature vectors, and this has proved to be effective, even if computationally demanding during the training phase.

Unfortunately when dealing with a vector of binary features, Euclidean distance is not the metric of choice, and the average vector is undefined.

Algorithm 3 K-majority algorithm

Given a collection D of binary vectors
Randomly generate k binary centroids C
repeat
  for $d \in D$ **do** ▷ Assign data to centroids
    $c_d \leftarrow \arg\min_{c \in C} HammingDistance(c, d)$
  end for
  for $c \in C$ **do** ▷ Majority voting
    for $d \in D | c_d = c$ **do**
      v accumulates d votes
    end for
    $c' \leftarrow Majority(v)$
  end for
until centroids not changed

```
unsigned HammingDistance (__m128i *x, __m128i *y) {
  __m128i xorValue = _mm_xor_si128(*x,*y);
  return (unsigned)__popcnt64(xorValue.m128i_u64[0])
    + (unsigned)__popcnt64(xorValue.m128i_u64[1]);
}
```

Figure 4.2: Example Hamming distance function in C language, using SSE4 instruction on a 64bit architecture.

A reasonable and effective distance between binary vectors is the Hamming distance (the number of different bits in corresponding positions), but still no average is provided. We could tackle the problem reverting to k-medoids (PAM algorithm) [89], but this would require the computation of a full distance matrix between the elements to be clustered, even worsening the problem. Therefore, to compute the centroid of a set of binary vectors based on the Hamming distance, we introduce a voting scheme. In particular, corresponding elements of each vector vote for 0 or 1. For determining each element of the centroid, the majority rule is used, with ties broken randomly. We call this variation of the Lloyd algorithm "k-majority

algorithm" and we resume it in Algorithm 3.

The algorithm processes a collection of D binary vectors and seeks for a number k of good centroids, that will become the visual dictionary for the Bag-Of-Words model. Initially, these k centroids are determined randomly (line 2). At each iteration, the initial step (lines 4-6) is the assignment of each binary vector to the closest centroid: the current binary vector d is therefore labelled with the index of the closest centroid. This part is essentially shared by many common clustering algorithms. The second step (lines 7-12) is the majority voting used to redefine the vector clustering. For each cluster c, we take into consideration every binary vector d belonging to it. Every bit of an accumulator vector v is increased by 1 if the corresponding bits in d is 1. At the end, the majority rule is used to form the new centroid c': for each element v_i in v, if the majority of vectors voted for 1 as bit value in v_i, then c'_i takes 1, otherwise c'_i takes 0. The algorithm iterates until no centroids are changed during the previous iteration.

A fundamental advantage of the k-majority approach is that both the Hamming distance and the majority voting step can work on the byte packed vector string. In particular the Hamming distance may be implemented leveraging both SSE instructions and specific bitwise hardware instructions. An optimized version of an Hamming distance function is provided in Fig. 4.2.If specialized hardware instructions are not available, it is still possible to employ some smart bitwise operations as those proposed in `http://graphics.stanford.edu/~seander/bithacks.html#CountBitsSetParallel` By using Hamming distance and majority voting in the cluster assignment step, which has to go through all elements to be clustered, we can obtain a speedup over classical k-means in the order of 100.

4.4.3 BoW Descriptors Lossless Compression Scheme

To reduce the bandwidth requirements of the system (which is of paramount importance when mobile devices are considered), an extremely simple and fast lossless compression scheme is applied to the BoW histograms. The first observation is that these descriptors are significantly sparse and become more and more sparse at growing size of the codebook. For this reason, similarly to the JPEG coding of AC coefficients, we employ a RLE of zeros and describe every non zero value as the number of zero values before the current one.

The second observation is that the distribution of nonzero value is definitely non uniform, with some values found many times and others really seldom. Huffman coding could be applied, but this would require a specific dictionary construction for every descriptor or a predefined dictionary which could be biased toward the specific dataset in use and possibly become not really useful at changing conditions. For these reasons we apply a universal coding technique, the Elias gamma code, known also as Exp-Golomb coding as used in the H.264/MPEG-4 AVC and Dirac video compression standards.

For the sake of simplicity, the original BoW histograms are stored as a stream of 32 bits wide floating point values, so we start our representation with a list of nonzero floating point values ordered by the most probable to the least probable ones, then encode the floating point values as an Elias encoded run length of zeros and an Elias encoded index pointing to the value table. The run lengths are not exactly exponentially distributed, so Elias coding is not the perfect choice for their representation, but this still allows for a very effective representation at small values, without adding a large overhead to longer representations.

In Table 4.1 we report the average compression obtained both with GZip standard compression applied over the uncompressed binary data and with our compression scheme. Our approach produces descriptor which are

Table 4.1: Compression tests comparisons. For both GZip and our compression scheme the average size in bytes and the average compression ratio are reported

centers	uncompressed binary	compressed gzip	compressed our	our/gzip
32	128	121 (1.06)	117 (1.10)	0.96
64	256	171 (1.50)	148 (1.73)	0.87
128	512	229 (2.23)	165 (3.11)	0.72
256	1,024	297 (3.45)	197 (5.20)	0.66
512	2,048	390 (5.25)	253 (8.09)	0.65
1K	4,096	517 (7.93)	328 (12.49)	0.63
2K	8,192	674 (12.16)	426 (19.24)	0.63
4K	16,384	854 (19.20)	542 (30.25)	0.63
8K	32,768	1,057 (31.01)	669 (48.99)	0.63
16K	65,536	1,300 (50.42)	804 (81.50)	0.62
32K	131,072	1,582 (82.86)	947 (138.40)	0.60

on average 30% smaller then GZip compression, while being much faster, because of the use of Elias encoding. In fact no search is required and the representation is a straightforward variable-length representation of a binary coded integer value. Higher compression rates could be obtained, at the price of a higher complexity.

4.4.4 Similarity Search

Once the query descriptor is received by the server, it is compared with the descriptors of all training images. The number of comparison can be largely reduced using only the descriptors of the locations that lie within a given radius from the current user position, provided by the GPS embedded on the mobile device. The descriptors are thus ranked according to the similarity measure given by the histogram intersection. We determine the similarity for each class computing the Average Precision (AP) on this ranked list as if the given class were the correct one, and propose back to

the user the list of classes sorted according to their AP.

With this simple scheme we avoid the need to train a multi-class classifier, with several advantages: i) the exhaustive similarity search is still feasible thanks to the GPS position constraint; ii) there is no need to retrain every time a new image (of an existing or new class) is added because we can compute the global descriptors of these images leveraging the same cluster centers; iii) the number of images per class may not be balanced, thanks to the AP re-ranking and, actually, iv) a class may be composed even by a single image.

4.5 Experimental results

The proposed system has been deployed on a wide range of smart phones and tablets, ranging from Samsung Galaxy Tab to Sony Xperia Z. The most important differences are computational power (from single core @ 1 GHz to quad core @ 1.5 GHz) and camera resolution (from 3.2 MP to 13 MP). In order to evaluate the accuracy, we tested offline the similarity search on two datasets. The first dataset, Significant Signs (SS), consists of 614 images: 464 depict shots of 96 different significant locations, the remaining 150 are noise. As depicted in Fig. 4.3, within each class each photo depicts the same location, taken from different points of view, with different devices, in different light conditions. The second dataset is the FlickrLogos-32 dataset [140] (FL32), which contains 30 photos showing brand logos for 32 logo classes, as long as 6000 non–logo images. It is meant for the evaluation of multi-class logo recognition as well as logo retrieval methods on real world images.

As local image descriptor we selected the ORB (see Sect. 4.4.1), because it is very fast to compute, and, being a binary descriptor, allows us for fast clustering via the K-majority algorithm (see Section 4.4.2) that can be easily accomplished on-board of a mobile device. We also aim at

Figure 4.3: Samples from the reference dataset, 1 column per class. Images depict the same location with different viewpoints, light conditions and camera resolutions.

demonstrating that the ORB descriptor provides also better retrieval than SIFT [106]. The accuracy obtained generating the global descriptor starting from ORB and SIFT, varying the number of cluster centers of the BoW model, i.e. the dimension of the global descriptor, has been evaluated. We computed the cluster centers starting from the local descriptors of all images, including noise, and used all images, except noise, as queries. The evaluation of the *Mean Average Precision (MAP)* in Fig. 4.4 shows that global descriptors generated from ORB perform in general much better than SIFT. Moreover, ORB provides the correct result as first response more often than SIFT, as depicted in Fig. 4.5.

We want our system to work irrespective of the device, so we must handle photos taken with different cameras at various resolutions. Since we work under the assumption that the logo is centered in the image and is clearly visible (some examples are reported in Fig. 4.3), we overcome the issue simply resizing the images. This allows us to normalize the sizes, so that their local descriptors, which are computed on fixed size patch, have comparable meanings. We report in Fig. 4.6 and Fig. 4.7 the results

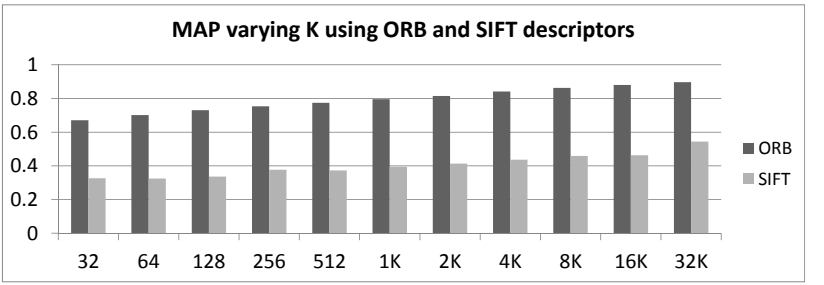


Figure 4.4: Mean Average Precision computed on the SS dataset varying the number K of cluster centers.

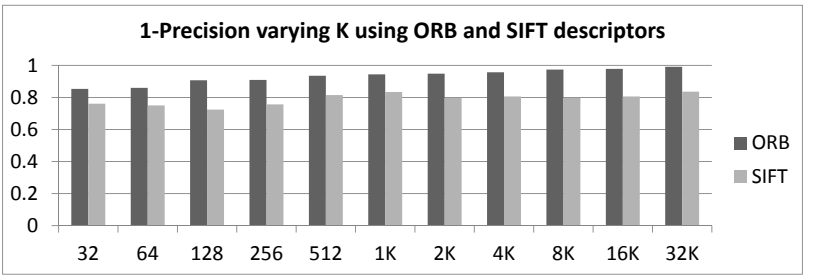


Figure 4.5: 1-Precision computed on the SS dataset varying the number K of cluster centers.

(obtained with $K = 512$ cluster centers) varying the dimensions of the resized images on the two datasets. The size of the images does not affect much the quality of the results, so we can use smaller sizes that guarantee a faster local descriptor computation.

We investigate also the number K of cluster centers of the BoW model, which directly affects the size of the global descriptor and the computational time. We tested our ORB–based descriptor on images resized to 320×240 on both datasets. First we train the BoW for both datasets using 1 random image per class, and repeated the evaluation 10 times, and using as queries all other images. The average values are reported as the

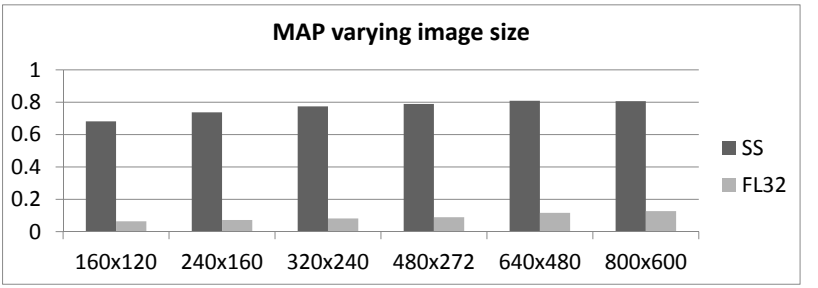


Figure 4.6: Mean Average Precision computed on the two datasets (SS and FL32) varying the size of the images.

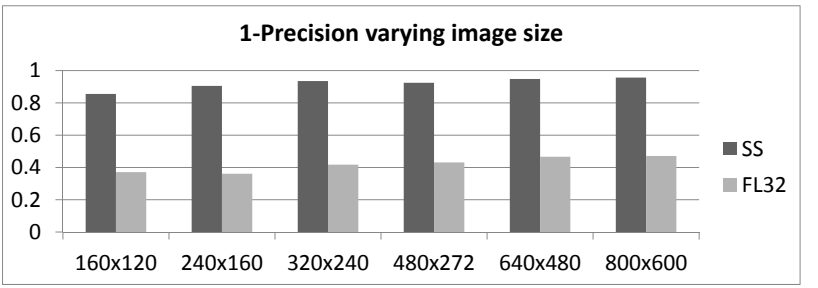


Figure 4.7: 1-Precision computed on the two datasets (SS and FL32) varying the size of the images.

darker bars (noted as SS-1 and FL32-1) in Fig. 4.8 and Fig. 4.9. They show that the performance increases with the number K, but very good results are already achieved with low values of K that allow for fast computation. Second, we train the BoW for both datasets using all images for each class, using again all images as queries. The results are reported as the brighter bars (noted as SS-All and FL32-All) in Fig. 4.8 and Fig. 4.9. On the one hand, adding more images helps the search, since the training set will more easily contain images similar to the query. This is clearly visible in Fig. 4.9, where the brighter bars are much higher than the darker bars. On the other hand, the overall performance is better with respect to

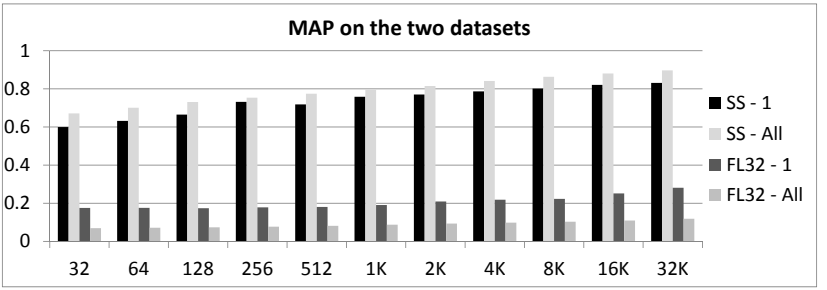


Figure 4.8: Mean Average Precision computed on the two dataset (SS and FL32), using 1 or All images as training samples.

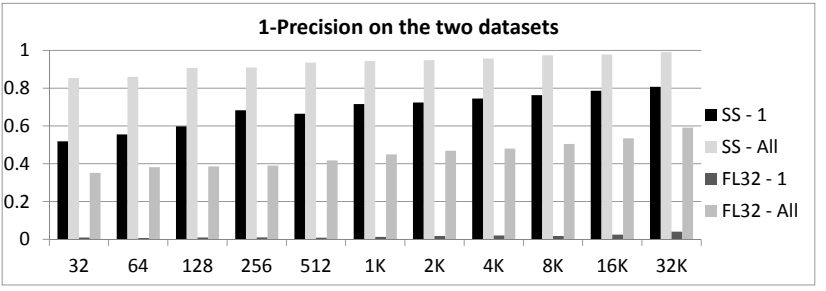


Figure 4.9: 1-Precision computed on the two dataset (SS and FL32), using 1 or All images as training samples.

our dataset, but gets worse for the FlickrLogos32 dataset (Fig. 4.8). This is because adding more information in the training phase provides worse results when the images are very diverse. When images depict exactly the same location, as in our dataset, adding more examples helps in dealing the different viewpoints and lighting conditions.

4.6 Conclusions

The proposed solution for visual place recognition has been developed for mobile application. As such, there has been the need to find efficient solutions feasible on commercial smart phones with limited resources and communication constraints. The proposed novel solution, based on ORB descriptors with BoW optimized for binary features, resulted to be an excellent trade-off between accuracy and efficiency, as demonstrated by our tests. The exploitation of GPS localization to reduce the set of possible matches in the training images also contributed to have a highly-responsive accurate system.

The proposed system is thus capable to retrieve information about a given place, provided that its visual description is available. As such, this application is very useful in contexts of advertising or tourism, where the visual description is provided by the same subjects that benefit from increased visibility.

Chapter 5

Conclusions

The increasing capabilities of mobile devices allow nowadays the development of a wide range of applications. Mobile Vision deals with the limitations of mobile devices, that do not allow Computer Vision tasks to run directly on the device, proposing several optimization strategies. Different architectural solutions allow to benefit of the unique features of mobile devices as well as of the computational and storage capabilities of standard computers. A thorough review of the literature in the field of Mobile Vision, regarding both applications and optimization techniques, is presented.

In this thesis are also proposed two novel methods that advance the state-of-the-art in the context of mobile pattern recognition and mobile visual place recognition, respectively.

The first method concerns the real-time detection of elliptic shapes. Most of the methods present in the literature are inherently too slow to run on a mobile device, or apply heuristic to speed up the process at the cost of losing in detection accuracy. The presented method overcomes these limitations by relying on heuristic constraints that guarantee to perform the most computational expensive tasks only on a very narrow set of elements.

High accuracy is achieved by relying on arcs as features, instead of single edge points. Novel approximate and fast solutions are given to accomplish the different sub-problems of the algorithm. This novel formulation makes the ellipse detection task feasible in real-time on mobile devices.

The second method deals with the problem of mobile visual place recognition. Several methods proposed in the literature rely on ad-hoc low-size or more informative descriptors to allow fast computation or feature matching. The proposed approach instead relies on efficient state-of-the-art binary descriptor and on a novel formulation of the bag-of-feature model to handle binary data. The resulting global descriptor allows for fast extraction and matching, while maintaining the discriminating capabilities for accurate matching. Because of its sparsity, the descriptor may be efficiently compressed to limit the amount of data sent through the network, thus reducing the latency and providing a very fast response for an effective user interaction.

References

[1] A.E. Abdel-Hakim and M. El-Saban. Face authentication using graph-based low-rank representation of facial local structures for mobile vision applications. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 40–47, 2011. doi: 10.1109/ICCVW.2011.6130220. 14

[2] Alberto S. Aguado, Eugenia Montiel, and Mark S. Nixon. On using directional information for parameter space decomposition in ellipse detection. *Patt. Rec.*, 29(3):369–381, 1996. 43, 55, 62

[3] Massimo Ancona, Marco Cappello, Marco Casamassima, Walter Cazzola, Davide Conte, Massimiliano Pittore, Gianluca Quercini, Naomi Scagliola, and Matteo Villa. Mobile vision and cultural heritage: the agamemnon project. In *Proc. IEEE of the 1st Int. Workshop on Mobile Vision (IMV06),(Graz, Austria, 2006)*, 2006. 18

[4] B. Anton and P. Svasta. Electronic components identified by computer vision using mobile devices. In *Design and Technology in Electronic Packaging (SIITME), 2011 IEEE 17th International Symposium for*, pages 179–182, 2011. doi: 10.1109/SIITME.2011.6102713. 9

[5] Clemens Arth, Daniel Wagner, M. Klopschitz, A. Irschara, and

D. Schmalstieg. Wide area localization on mobile phones. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pages 73–82, 2009. doi: 10.1109/ISMAR.2009.5336494. 11

[6] Georges Baatz, Kevin Köser, David Chen, Radek Grzeszczuk, and Marc Pollefeys. Leveraging 3d city models for rotation invariant place-of-interest recognition. *International journal of computer vision*, 96(3):315–334, 2012. 10

[7] M.A. Bagheri and Q. Gao. Logo recognition based on a novel pairwise classification approach. In *Artificial Intelligence and Signal Processing (AISP), 2012 16th CSI International Symposium on*, pages 316–321, 2012. 87

[8] C.A. Basca, M. Talos, and R. Brad. Randomized hough transform for ellipse detection with result clustering. In *Computer as a Tool, 2005. EUROCON 2005.The International Conference on*, volume 2, pages 1397 –1400, 2005. 42, 62, 67, 69, 70, 71, 77, 78, 79

[9] Herbert Bay, Beat Fasel, and Luc Van Gool. Interactive museum guide: Fast and robust recognition of museum objects. In *Proceedings of the first international workshop on mobile vision*, May 2006. 17

[10] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vision Image Understanding*, 110(3):346–359, 2008. 88

[11] Daniel Beier, R Billert, B Bruderlin, Dirk Stichling, and Bernd Kleinjohann. Marker-less vision based tracking for mobile augmented reality. In *Mixed and Augmented Reality, 2003. Proceedings. The Second IEEE and ACM International Symposium on*, pages 258–259. IEEE, 2003. 12

[12] A Benesova, Yuriy Lypetskyy, A Lucas Paletta, Andreas Jeitler, and Evelyn Hödl. A mobile system for vision based road sign inventory. In *in Proc. 5th International Symposium on Mobile Mapping Technology*. Citeseer, 2007. 9

[13] M. Bordallo López, J. Hannuksela, O. Silvén, and M. Vehviläinen. Graphics hardware accelerated panorama builder for mobile phones. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 7256 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, February 2009. doi: 10.1117/12.816511. 33

[14] Erich Bruns and Oliver Bimber. Adaptive training of video sets for image recognition on mobile phones. *Personal and Ubiquitous Computing*, 13(2):165–178, 2009. 18

[15] Erich Bruns, Benjamin Brombach, Thomas Zeidler, and Oliver Bimber. Enabling mobile phones to support large-scale museum guidance. *MultiMedia, IEEE*, 14(2):16–25, 2007. 18

[16] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Proc. Eur. Conf. Comput. Vision*, pages 778–792, 2010. 88, 92

[17] A. B. Cambra and A.C. Murillo. Towards robust and efficient text sign reading from a mobile phone. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 64–71, 2011. doi: 10.1109/ICCVW.2011.6130223. 13

[18] John Canny. A computational approach to edge detection. *IEEE Trans. on PAMI*, 8(6):679 –698, 1986. 46

[19] R. Chan and W.-C. Siu. Fast detection of ellipses using chord bisectors. In *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90.,*

1990 International Conference on, pages 2201–2204 vol.4, 1990. doi: 10.1109/ICASSP.1990.115997. 62

[20] V. Chandrasekhar, G. Takacs, D. Chen, S. Tsai, R. Grzeszczuk, and B. Girod. Chog: Compressed histogram of gradients a low bit-rate feature descriptor. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2504–2511, 2009. doi: 10.1109/CVPR.2009.5206733. 36

[21] V. Chandrasekhar, Y. Reznik, G. Takacs, D. Chen, S. Tsai, R. Grzeszczuk, and B. Girod. Quantization schemes for low bitrate compressed histogram of gradients descriptors. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 33–40, 2010. doi: 10.1109/CVPRW.2010.5543242. 34

[22] V. Chandrasekhar, Y. Reznik, G. Takacs, D.M. Chen, S.S. Tsai, R. Grzeszczuk, and B. Girod. Compressing feature sets with digital search trees. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 32–39, 2011. doi: 10.1109/ICCVW.2011.6130219. 34

[23] Vijay Chandrasekhar, Mina Makar, Gabriel Takacs, David Chen, Sam S. Tsai, Ngai man Cheung, Radek Grzeszczuk, Yuriy Reznik, and Bernd Girod. Survey of sift compression schemes, 2010. 34

[24] Bin Chen, Jie Shen, and Helei Sun. A fast face recognition system on mobile phone. In *Systems and Informatics (ICSAI), 2012 International Conference on*, pages 1783–1786, 2012. doi: 10.1109/ICSAI.2012.6223389. 14

[25] David Chen and Bernd Girod. Memory-efficient image databases for mobile visual search. *IEEE Multimedia*, 99(PrePrints):1, 2013.

ISSN 1070-986X. doi: http://doi.ieeecomputersociety.org/10.1109/MMUL.2013.46. 36

[26] David Chen, Ngai-Man Cheung, Sam Tsai, Vijay Chandrasekhar, Gabriel Takacs, Ramakrishna Vedantham, Radek Grzeszczuk, and Bernd Girod. Dynamic selection of a feature-rich query frame for mobile video retrieval. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 1017–1020. IEEE, 2010. 8

[27] David Chen, Sam Tsai, Vijay Chandrasekhar, Gabriel Takacs, Huizhong Chen, Ramakrishna Vedantham, Radek Grzeszczuk, and Bernd Girod. Residual enhanced visual vectors for on-device image matching. In *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on*, pages 850–854. IEEE, 2011. 36

[28] David Chen, Sam Tsai, Vijay Chandrasekhar, Gabriel Takacs, Ramakrishna Vedantham, Radek Grzeszczuk, and Bernd Girod. Residual enhanced visual vector as a compact signature for mobile visual search. *Signal Processing*, 93(8):2316–2327, 2013. 36

[29] David M Chen, Sam S Tsai, Vijay Chandrasekhar, Gabriel Takacs, Jatinder Singh, and Bernd Girod. Tree histogram coding for mobile image matching. In *Data Compression Conference, 2009. DCC'09.*, pages 143–152. IEEE, 2009. 34

[30] David M Chen, Sam S Tsai, Ramakrishna Vedantham, Radek Grzeszczuk, and Bernd Girod. Streaming mobile augmented reality on mobile phones. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pages 181–182. IEEE, 2009. 7

[31] David M Chen, Georges Baatz, K Koser, Sam S Tsai, Ramakrishna Vedantham, Timo Pylvanainen, Kimmo Roimela, Xin Chen, Jeff

Bach, Marc Pollefeys, et al. City-scale landmark identification on mobile devices. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 737–744. IEEE, 2011. 10

[32] Tao Chen, Kim-Hui Yap, and L-P Chau. Integrated content and context analysis for mobile landmark recognition. *Circuits and Systems for Video Technology, IEEE Transactions on*, 21(10):1476–1486, 2011. ISSN 1051-8215. doi: 10.1109/TCSVT.2011.2161413. 10

[33] Wei-Chao Chen, Yingen Xiong, Jiang Gao, Natasha Gelfand, and Radek Grzeszczuk. Efficient extraction of robust image features on mobile devices. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–2. IEEE Computer Society, 2007. 29

[34] Kwang-Ting Cheng and Yi-Chu Wang. Using mobile gpu for general-purpose computing a case study of face recognition on smartphones. In *VLSI Design, Automation and Test (VLSI-DAT), 2011 International Symposium on*, pages 1–4, 2011. doi: 10.1109/VDAT.2011.5783575. 32

[35] Keith Cheverst, Nigel Davies, Keith Mitchell, Adrian Friday, and Christos Efstratiou. Developing a context-aware electronic tourist guide: some issues and experiences. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 17–24. ACM, 2000. 18

[36] A.Y.-S. Chia, S. Rahardja, D. Rajan, and M.K. Leung. A split and merge based ellipse detector with self-correcting capability. *IEEE Trans. on Image Processing*, 20(7):1991 –2006, 2011. x, 43, 62, 71, 72, 73

[37] A.Y.S. Chia, M.K.H. Leung, How-Lung Eng, and S. Rahardja. Ellipse detection with hough transform in one dimensional parametric space.

In *Proc. of IEEE Intl Conf on Image Processing*, volume 5, pages V –333 –V –336, 2007. 42

[38] Junyeong Choi, Hanhoon Park, Jungsik Park, and Jong-Il Park. Bare-hand-based augmented reality interface on mobile phone. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 275–276, 2011. doi: 10.1109/ISMAR.2011.6143899. 11

[39] Kwontaeg Choi, Kar-Ann Toh, and Hyeran Byun. Realtime training on mobile devices for face recognition applications. *Pattern Recognition*, 44(2):386 – 400, 2011. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/j.patcog.2010.08.009. URL `http://www.sciencedirect.com/science/article/pii/S003132031000395X`. 14, 32

[40] T. Cooke. A fast automatic ellipse detector. In *Digital Image Computing: Techniques and Applications (DICTA), 2010 International Conference on*, pages 575–580, 2010. doi: 10.1109/DICTA.2010.102. 41, 43, 44, 62

[41] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, pages 886–893, 2005. 89

[42] M. Dantone, L. Bossard, T. Quack, and L. Van Gool. Augmented faces. In *IEEE International Workshop on Mobile Vision (ICCV 2011)*, 2011. 14

[43] Nigel Davies, Keith Cheverst, Alan Dix, and Andre Hesse. Understanding the role of image recognition in mobile tour guides. In *Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, pages 191–198. ACM, 2005. 17

[44] Paul Debenham, Graham Thomas, and Jonathan Trout. Evolutionary augmented reality at the natural history museum. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 249–250, 2011. doi: 10.1109/ISMAR.2011.6092400. 18

[45] Toan Nguyen Dinh, Jonghyun Park, and GueeSang Lee. Low-complexity text extraction in korean signboards for mobile applications. In *Computer and Information Technology, 2008. CIT 2008. 8th IEEE International Conference on*, pages 333–337. IEEE, 2008. 13

[46] M. Donoser, P. Kontschieder, and H. Bischof. Robust planar target tracking and pose estimation from a single concavity. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 9–15, 2011. doi: 10.1109/ISMAR.2011.6092365. 12

[47] Andrew Ensor and Seth Hall. Gpu-based image analysis on mobile devices, 2011. 32

[48] Armando Fernandes. A correct set of equations for the real-time ellipse hough transform algorithm, 2009. 55

[49] Silvio Ferreira, Vincent Garin, and Bernard Gosselin. A text detection technique applied in the framework of a mobile camera-based application. In *Proceedings of the First International Workshop on Camera-based Document Analysis and Recognition (CBDAR)*, 2005. 13

[50] Roy T. Fielding and Richard N. Taylor. Principled design of the modern web architecture. *ACM Trans. Internet Technol.*, 2(2):115–150, May 2002. ISSN 1533-5399. 90

[51] A. Fitzgibbon, M. Pilu, and R.B. Fisher. Direct least square fitting of ellipses. *IEEE Trans. on PAMI*, 21(5):476 –480, 1999. 43, 68, 77

[52] Paul Föckler, Thomas Zeidler, Benjamin Brombach, Erich Bruns, and Oliver Bimber. Phoneguide: museum guidance supported by on-device object recognition on mobile phones. In *Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*, MUM '05, pages 3–10, New York, NY, USA, 2005. ACM. ISBN 0-473-10658-2. doi: 10.1145/1149488.1149490. URL `http://doi.acm.org/10.1145/1149488.1149490`. 18

[53] Paul Föckler, Thomas Zeidler, Benjamin Brombach, Erich Bruns, and Oliver Bimber. Phoneguide: museum guidance supported by on-device object recognition on mobile phones. In *Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*, pages 3–10. ACM, 2005. 41

[54] M. Fornaciari and A. Prati. Very fast ellipse detection for embedded vision applications. In *Distributed Smart Cameras (ICDSC), 2012 Sixth International Conference on*, pages 1–6, 2012. 44

[55] M. Fornaciari, R. Cucchiara, and A. Prati. A mobile vision system for fast and accurate ellipse detection. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 52–53, 2013. 44

[56] P.-E. Forssen and E. Ringaby. Rectifying rolling shutter video from hand-held devices. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 507–514, 2010. doi: 10.1109/CVPR.2010.5540173. 15

[57] V. Fragoso, S. Gauglitz, S. Zamora, J. Kleban, and M. Turk. Translatar: A mobile augmented reality translator. In *Applications of*

Computer Vision (WACV), 2011 IEEE Workshop on, pages 497–502, 2011. doi: 10.1109/WACV.2011.5711545. 13

[58] H. Freeman and R. Shapira. Determining the minimum-area encasing rectangle for an arbitrary closed curve. *Commun. ACM*, 18(7):409–413, 1975. 46

[59] G. Fritz, C. Seifert, and L. Paletta. A mobile vision system for urban detection with informative local descriptors. In *Computer Vision Systems, 2006 ICVS '06. IEEE International Conference on*, pages 30–30, 2006. doi: 10.1109/ICVS.2006.5. 37

[60] S. Gammeter, A. Gassmann, L. Bossard, T. Quack, and L. Van Gool. Server-side object recognition and client-side object tracking for mobile augmented reality. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 1–8, 2010. doi: 10.1109/CVPRW.2010.5543248. 12

[61] S. Gauglitz, C. Sweeney, J. Ventura, M. Turk, and T. Hollerer. Live tracking and mapping from both general and rotation-only camera motion. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pages 13–22, 2012. doi: 10.1109/ISMAR.2012.6402532. 17

[62] Juergen Gausemeier, Juergen Fruend, Carsten Matysczok, Beat Bruederlin, and David Beier. Development of a real time image based object recognition method for mobile ar-devices. In *Proceedings of the 2nd international conference on Computer graphics, virtual Reality, visualisation and interaction in Africa*, pages 133–139. ACM, 2003. 12

[63] Bernd Girod, Vijay Chandrasekhar, David M Chen, Ngai-Man Cheung, Radek Grzeszczuk, Yuriy Reznik, Gabriel Takacs, Sam S Tsai,

and Ramakrishna Vedantham. Mobile visual search. *Signal Processing Magazine, IEEE*, 28(4):61–76, 2011.

[64] K. Grauman and T. Darrell. Efficient image matching with distributions of local invariant features. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 627–634 vol. 2, 2005. 88

[65] T. Guan, Y. He, J. Gao, J. Yang, and J. Yu. On-device mobile visual location recognition by integrating vision and inertial sensors, 2013. ISSN 1520-9210. 10, 35

[66] Zhenwen Gui, Yongtian Wang, Yue Liu, and Jing Chen. Outdoor scenes identification on mobile device by integrating vision and inertial sensors. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*, pages 1596–1600, 2013. doi: 10.1109/IWCMC.2013.6583794. 10

[67] Claudio Guida, Dario Comanducci, and Carlo Colombo. Automatic bus line number localization and recognition on mobile phonesa computer vision aid for the visually impaired. In *Image Analysis and Processing–ICIAP 2011*, pages 323–332. Springer, 2011. 42

[68] N Guil and E.L Zapata. Lower order circle and ellipse hough transform. *Patt. Rec.*, 30(10):1729 – 1744, 1997. 61

[69] Jaewon Ha, Kyusung Cho, F.A. Rojas, and H.S. Yang. Real-time scalable recognition and tracking based on the server-client model for mobile augmented reality. In *VR Innovation (ISVRI), 2011 IEEE International Symposium on*, pages 267–272, 2011. doi: 10.1109/ISVRI.2011.5759649. 12

[70] A. Hadid, J.Y. Heikkila, O. Silven, and M. Pietikainen. Face and eye detection for person authentication in mobile phones. In *Dis-*

tributed Smart Cameras, 2007. ICDSC '07. First ACM/IEEE International Conference on, pages 101–108, 2007. doi: 10.1109/ICDSC.2007.4357512. 14

[71] N. Hagbi, O. Bergig, J. El-Sana, and M. Billinghurst. Shape recognition and pose estimation for mobile augmented reality. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pages 65–71, 2009. doi: 10.1109/ISMAR.2009.5336498. 12

[72] Kwangsoo Hahn, Sungcheol Jung, Youngjoon Han, and Hernsoo Hahn. A new algorithm for ellipse detection by curve segments. *Patt. Rec. Letters*, 29(13):1836 – 1841, 2008. 43, 61, 62

[73] S. Haner and A. Heyden. A step towards self-calibration in slam: Weakly calibrated on-line structure and motion estimation. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 59–64, 2010. doi: 10.1109/CVPRW.2010.5543256. 17

[74] Gustav Hanning, Nicklas Forslow, Per-Erik Forssn, Erik Ringaby, David Trnqvist, and Jonas Callmer. Stabilizing cell phone video using inertial measurement sensors. In *Computational Methods for the Innovative Design of Electrical Devices'11*, pages 1–8, 2011. 15

[75] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. Alvey Vision Conf.*, pages 147–151, 1988. 92

[76] Timothy J. Hazen, Eugene Weinstein, Bernd Heisele, Alex Park, and Ji Ming. Multi-modal face and speaker identification for mobile devices. In *Face Biometrics for Personal Identification: Multi-Sensory Multi-Modal Systems*, page 123138. Springer, 2006. 14

[77] J. Hedborg, E. Ringaby, P.-E. Forssen, and M. Felsberg. Structure and motion estimation from rolling shutter video. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 17–23, 2011. doi: 10.1109/ICCVW.2011.6130217. 17

[78] J. Herling and W. Broll. Pixmix: A real-time approach to high-quality diminished reality. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pages 141–150, 2012. doi: 10.1109/ISMAR.2012.6402551. 16

[79] Chun-Ta Ho and Ling-Hwei Chen. A fast ellipse/circle detector using geometric symmetry. *Patt. Rec.*, 28(1):117 – 124, 1995. 43, 44, 62

[80] Gang Hua, Yun Fu, Matthew Turk, Marc Pollefeys, and Zhengyou Zhang. Introduction to the special issue on mobile vision. *Int. J. Comput. Vision*, 96(3):277–279, 2012. 41

[81] Gang Hua, Yun Fu, Matthew Turk, Marc Pollefeys, and Zhengyou Zhang. Introduction to the special issue on mobile vision. *International Journal of Computer Vision*, 96(3):277–279, 2012. ISSN 0920-5691. doi: 10.1007/s11263-011-0506-3. URL `http://dx.doi.org/10.1007/s11263-011-0506-3`. 1

[82] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. ACM Symp. Theor. Comput.*, pages 604–613, 1998. 89

[83] Eisuke Ito, T. Okatani, and K. Deguchi. Accurate and robust planar tracking based on a model of image sampling and reconstruction process. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 1–8, 2011. doi: 10.1109/ISMAR.2011.6092364. 11

[84] Juett J and Essl G. Real-time computer vision for heading correction in mobile augmented reality registration on wind farms. *Proceedings of the Workshop on Mobile Vision and HCI (MobiVis). Held in Conjunction with Mobile HCI*, 2012. 12

[85] Sabah Jassim, Harin Sellahewa, and Johan-Hendrik Ehlers. Wavelet-based face verification for mobile personal devices. *Biometrics on the Internet*, page 81, 2005. 14

[86] Herve Jegou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, pages 3304–3311, 2010. 88

[87] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011. 89

[88] Rongrong Ji, Ling-Yu Duan, Jie Chen, Hongxun Yao, Junsong Yuan, Yong Rui, and Wen Gao. Location discriminative vocabulary coding for mobile landmark search. *International Journal of Computer Vision*, 96(3):290–314, 2012. ISSN 0920-5691. doi: 10.1007/s11263-011-0472-9. URL `http://dx.doi.org/10.1007/s11263-011-0472-9`. 10, 34

[89] L. Kaufman and P.J. Rousseeuw. Clustering by means of medoids. In *Int. Conf. Stat. Data Anal.*, pages 405–416, 1987. 94

[90] Euijin Kim, Miki Haseyama, and Hideo Kitajima. Fast and robust ellipse extraction from complicated images. In *Proc. IEEE Intl Conf. on Information Technology and Applications*, 2002. 43, 44, 62

[91] Euijin Kim, Miki Haseyama, and Hideo Kitajima. Fast line extraction

from digital images using line segments. *Systems and Computers in Japan*, 34(10):76–89, 2003. x, 60, 68, 77

[92] Georg Klein and David Murray. Parallel tracking and mapping on a camera phone. In *Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '09, pages 83–86, Washington, DC, USA, 2009. IEEE Computer Society. ISBN 978-1-4244-5390-0. doi: 10.1109/ISMAR.2009.5336495. URL `http://dx.doi.org/10.1109/ISMAR.2009.5336495`. 17

[93] M. Knecht, C. Traxler, W. Purgathofer, and M. Wimmer. Adaptive camera-based color mapping for mixed-reality applications. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 165–168, 2011. doi: 10.1109/ISMAR.2011.6092382. 12

[94] O. Korkalo, M. Aittala, and S. Siltanen. Light-weight marker hiding for augmented reality. In *Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on*, pages 247–248, 2010. doi: 10.1109/ISMAR.2010.5643590. 16

[95] Seung Eun Lee, Yong Zhang, Zhen Fang, Sadagopan Srinivasan, Ravi Iyer, and Donald Newell. Accelerating mobile augmented reality on a handheld platform. In *Computer Design, 2009. ICCD 2009. IEEE International Conference on*, pages 419–426. IEEE, 2009. 33

[96] Sungwon Lee, Choong seon Hong, Yong Kwi Lee, and Hyun soon Shin. Experimental emotion recognition system and services for mobile network environments. In *Sensors, 2010 IEEE*, pages 136–140, Nov 2010. doi: 10.1109/ICSENS.2010.5690670. 15

[97] Taehee Lee and T. Hollerer. Handy ar: Markerless inspection of augmented reality objects using fingertip tracking. In *Wearable Com-*

puters, 2007 11th IEEE International Symposium on, pages 83–90, 2007. doi: 10.1109/ISWC.2007.4373785. 11

[98] Yiwu Lei and Kok Cheong Wong. Ellipse detection based on symmetry. *Patt. Rec. Letters*, 20(1):41–47, 1999. 43

[99] Yunpeng Li, Noah Snavely, and DanielP. Huttenlocher. Location recognition using prioritized feature matching. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision ECCV 2010*, volume 6312 of *Lecture Notes in Computer Science*, pages 791–804. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-15551-2. doi: 10.1007/978-3-642-15552-9_57. URL `http://dx.doi.org/10.1007/978-3-642-15552-9_57`. 9

[100] Zhen Li and Kim-Hui Yap. Content and context boosting for mobile landmark recognition. *Signal Processing Letters, IEEE*, 19(8):459–462, 2012. 10

[101] Lars Libuda, Ingo Grothues, and Karl-Friedrich Kraiss. Ellipse detection in digital image data using geometric features. In Jos Braz, Alpesh Ranchordas, Helder Arajo, and Joaquim Jorge, editors, *Advances in Computer Graphics and Computer Vision*, volume 4 of *Communications in Computer and Information Science*, pages 229–239. Springer Berlin Heidelberg, 2007. 43, 44, 61, 62, 68, 70, 71, 77, 78, 79

[102] Joo-Hwee Lim, Yiqun Li, Yilun You, and J.-P. Chevallet. Scene recognition with camera phones for tourist information access. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 100–103, 2007. doi: 10.1109/ICME.2007.4284596. 17

[103] Wong Hwee Ling and Woo Chaw Seng. Traffic sign recognition model on mobile device. In *Computers Informatics (ISCI), 2011 IEEE Symposium on*, pages 267–272, 2011. doi: 10.1109/ISCI.2011.5958925. 8

[104] Zhi-Yong Liu and Hong Qiao. Multiple ellipses detection in noisy environments: A hierarchical approach. *Patt. Rec.*, 42(11):2421 – 2433, 2009. 43, 61, 62

[105] Miguel Bordallo López, Henri Nykänen, Jari Hannuksela, Olli Silvén, and Markku Vehviläinen. Accelerating image recognition on mobile devices using gpgpu. In *IS&T/SPIE Electronic Imaging*, pages 78720R–78720R. International Society for Optics and Photonics, 2011. 33

[106] D.G. Lowe. Object recognition from local scale-invariant features. In *Proc. IEEE Int. Conf. Comput. Vision*, volume 2, pages 1150–1157, 1999. 88, 99

[107] Wei Lu and Jinglu Tan. Detection of incomplete ellipse in images with strong noise by iterative randomized hough transform (irht). *Patt. Rec.*, 41(4):1268–1279, 2008. 42

[108] F. Mai, Y.S. Hung, H. Zhong, and W.F. Sze. A hierarchical approach for fast and robust ellipse extraction. *Patt. Rec.*, 41(8):2512 – 2524, 2008. 43, 44, 62

[109] Mina Makar, Sam S. Tsai, Vijay Chandrasekhar, David Chen, and Bernd Girod. Interframe coding of canonical patches for mobile augmented reality. In *Proceedings of the 2012 IEEE International Symposium on Multimedia*, ISM '12, pages 50–57, Washington, DC, USA, 2012. IEEE Computer Society. ISBN 978-0-7695-4875-3. doi: 10.1109/ISM.2012.18. URL `http://dx.doi.org/10.1109/ISM.2012.18`. 35

[110] Takuma Maruyama, Yoshiyuki Kawano, and Keiji Yanai. Real-time mobile recipe recommendation system using food ingredient recognition. In *Proceedings of the 2nd ACM international workshop on*

Interactive multimedia on mobile and portable devices, IMMPD '12, pages 27–34, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1595-1. doi: 10.1145/2390821.2390830. URL `http://doi.acm.org/10.1145/2390821.2390830`. 9

[111] Jiri Matousek. Randomized optimal algorithm for slope selection. *Information Processing Letters*, pages 183–187, 1991. 53

[112] Robert A. Mclaughlin. Randomized hough transform: Improved ellipse detection with comparison. Technical report, Univ. of Western Australia, 1998. 42

[113] J Menzel, Michael Königs, and Leif Kobbelt. A framework for vision-based mobile ar applications. In *Proceedings of the Workshop on Mobile Vision and HCI (MobiVis). Held in Conjunction with Mobile HCI*, 2012. 12

[114] J.H. Nah, Y.S. Kang, K.J. Lee, S.J. Lee, T.D. Han, and S.B. Yang. Mobirt: an implementation of opengl es-based cpu-gpu hybrid ray tracer for mobile devices. In *ACM SIGGRAPH ASIA 2010 Sketches*, page 50. ACM, 2010. 33

[115] C.K. Ng, M. Savvides, and P.K. Khosla. Real-time face verification system on a cell-phone using advanced correlation filters. In *Automatic Identification Advanced Technologies, 2005. Fourth IEEE Workshop on*, pages 57–62, 2005. doi: 10.1109/AUTOID.2005.42. 14, 32

[116] Thanh Minh Nguyen, S. Ahuja, and Q.M.J. Wu. A real-time ellipse detection based on edge grouping. In *Proc. of IEEE Intl Conf on Systems, Man and Cybernetics*, pages 3280 –3286, 2009. 43, 44, 61, 62

[117] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2161–2168, 2006. doi: 10.1109/CVPR.2006.264. 9

[118] Derek Nowrouzezahrai, Stefan Geiger, Kenny Mitchell, Robert Sumner, Wojciech Jarosz, and Markus Gross. Light factorization for mixed-frequency shadows in augmented reality. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 173–179, 2011. doi: 10.1109/ISMAR.2011.6092384. 12

[119] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *Int. J. Comput. Vision*, 42(3):145–175, 2001. 89

[120] Thomas Olsson and Markus Salo. Online user survey on current mobile augmented reality applications. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 75–84, 2011. doi: 10.1109/ISMAR.2011.6092372. 11

[121] L. Paletta, G. Fritz, C. Seifert, P. Luley, and A. Aimer. A mobile vision service for multimedia tourist applications in urban environments. In *Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, pages 566–572, 2006. doi: 10.1109/ITSC.2006.1706801. 17

[122] Qi Pan, Clemens Arth, Gerhard Reitmayr, Edward Rosten, and Tom Drummond. Rapid scene reconstruction on mobile phones from panoramic images. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 55–64, 2011. doi: 10.1109/ISMAR.2011.6092370. 17

[123] Youngmin Park, V. Lepetit, and Woontack Woo. Texture-less object tracking with online training using an rgb-d camera. In *Mixed and*

Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on, pages 121–126, 2011. doi: 10.1109/ISMAR.2011.6092377. 12

[124] M. Petter, V. Fragoso, M. Turk, and Charles Baur. Automatic text detection for mobile augmented reality translation, 2011. 13

[125] H. Pourghassem. A hierarchical logo detection and recognition algorithm using two-stage segmentation and multiple classifiers. In *Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on*, pages 227–231, 2012. 87

[126] Dilip K. Prasad, Maylor K.H. Leung, and Siu-Yeung Cho. Edge curvature and convexity based ellipse detection method. *Patt. Rec.*, 45(9):3204 – 3221, 2012. 43, 61, 62, 63, 67, 68, 69, 70, 71, 72, 73, 74, 76, 77, 79

[127] Dilip K. Prasad, Maylor K.H. Leung, and Chai Quek. Ellifit: An unconstrained, non-iterative, least squares based geometric ellipse fitting method. *Patt. Rec.*, 46(5):1449 – 1465, 2013. 43, 68, 77

[128] D.K. Prasad and M. K H Leung. Clustering of ellipses based on their distinctiveness: An aid to ellipse detection algorithms. In *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, volume 8, pages 292–297, 2010. doi: 10.1109/ICCSIT.2010.5564932. 58, 59, 62

[129] Yu Qiao and S.H. Ong. Arc-based evaluation and detection of ellipses. *Patt. Rec.*, 40(7):1990 – 2003, 2007. 43, 62

[130] Kiran K. Rachuri, Mirco Musolesi, Cecilia Mascolo, Peter J. Rentfrow, Chris Longworth, and Andrius Aucinas. Emotionsense: A mobile phones based adaptive platform for experimental social psychology research. In *Proceedings of the 12th ACM International*

Conference on Ubiquitous Computing, Ubicomp '10, pages 281–290, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-843-8. doi: 10.1145/1864349.1864393. URL http://doi.acm.org/10.1145/1864349.1864393. 15

[131] M Rahman, Jianfeng Ren, and Nasser Kehtarnavaz. Real-time implementation of robust face detection on mobile platforms. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1353–1356. IEEE, 2009. 31

[132] M Rahman, Jianfeng Ren, and Nasser Kehtarnavaz. Real-time implementation of robust face detection on mobile platforms. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1353–1356. IEEE, 2009. 41

[133] Jianfeng Ren, Nasser Kehtarnavaz, and Leonardo Estevez. Real-time optimization of viola-jones face detection for mobile platforms. In *Circuits and Systems Workshop: System-on-Chip-Design, Applications, Integration, and Software, 2008 IEEE Dallas*, pages 1–4. IEEE, 2008. 31

[134] Jianfeng Ren, Xudong Jiang, and Junsong Yuan. A fast and accurate cascade subspace face/eye detector on mobile devices. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 84–91, 2011. doi: 10.1109/ICCVW.2011.6130227. 14, 31

[135] Jianfeng Ren, Xudong Jiang, and Junsong Yuan. A complete and fully automated face verification system on mobile devices. *Pattern Recogn.*, 46(1):45–56, January 2013. ISSN 0031-3203. doi: 10.1016/j.patcog.2012.06.013. URL http://dx.doi.org/10.1016/j.patcog.2012.06.013. 14

[136] Erik Ringaby and Per-Erik Forssn. Efficient video rectification and stabilisation for cell-phones. *International Journal of Computer Vision*, 96(3):335–352, 2012. ISSN 0920-5691. doi: 10.1007/s11263-011-0465-8. URL `http://dx.doi.org/10.1007/s11263-011-0465-8`. 15

[137] Blaine Rister, Guohui Wang, Michael Wu, and Joseph R Cavallaro. A fast and efficient sift detector using the mobile gpu. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013. 32

[138] Michael Rohs and Beat Gfeller. Using camera-equipped mobile phones for interacting with real-world objects. In *Advances in Pervasive Computing*, pages 265–271, 2004. 11

[139] Stefan Romberg and Rainer Lienhart. Bundle min-hashing for logo recognition. In *Proceedings of the 3rd ACM International Conference on Multimedia Retrieval (ICMR)*, ICMR '13, pages 113–120, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2033-7. 87

[140] Stefan Romberg, Lluis Garcia Pueyo, Rainer Lienhart, and Roelof van Zwol. Scalable logo recognition in real-world images. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, ICMR '11, pages 25:1–25:8, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0336-1. 98

[141] Paul L. Rosin. Measuring corner properties. *Comput. Vision Image Understanding*, 73(2):291–307, 1999. 92

[142] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Proc. Eur. Conf. Comput. Vision*, pages 430–443, 2006. 92

[143] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Proc. IEEE Int. Conf. Comput. Vision*, 2011. 89, 93

[144] H. Sahbi, L. Ballan, G. Serra, and A. Del Bimbo. Context-dependent logo matching and recognition. *Image Processing, IEEE Transactions on*, 22(3):1018–1031, 2013. ISSN 1057-7149. 87

[145] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–7, June 2007. doi: 10.1109/CVPR.2007.383150. 9

[146] C. Schneider, N. Esau, L. Kleinjohann, and B. Kleinjohann. Feature based face localization and recognition on mobile devices. In *Control, Automation, Robotics and Vision, 2006. ICARCV '06. 9th International Conference on*, pages 1–6, 2006. doi: 10.1109/ICARCV.2006.345308. 14

[147] G. Schroth, R. Huitl, D. Chen, M. Abu-Alqumsan, A. Al-Nuaimi, and E. Steinbach. Mobile visual location recognition. *Signal Processing Magazine, IEEE*, 28(4):77–89, 2011. ISSN 1053-5888. doi: 10.1109/MSP.2011.940882. 9

[148] G. Schroth, R. Huitl, M. Abu-Alqumsan, F. Schweiger, and E. Steinbach. Exploiting prior knowledge in mobile visual location recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 2357–2360, 2012. doi: 10.1109/ICASSP.2012.6288388. 10

[149] Georg Schroth, Anas Al-Nuaimi, Robert Huitl, Florian Schweiger, and Eckehard Steinbach. Rapid image retrieval for mobile location recognition. In *Acoustics, Speech and Signal Processing (ICASSP),*

2011 IEEE International Conference on, pages 2320–2323. IEEE, 2011. 35

[150] P.-K. Ser and W.-C. Siu. Novel detection of conics using 2-d hough planes. *IEE Proc. Vision, Image and Signal Processing*, 142(5):262 –270, 1995. 62

[151] Abhinav Shrivastava, Tomasz Malisiewicz, Abhinav Gupta, and Alexei A. Efros. Data-driven visual similarity for cross-domain image matching. *ACM Trans. Graph.*, 30(6):154:1–154:10, December 2011. ISSN 0730-0301. 88

[152] S. Siltanen. Texture generation over the marker area. In *Mixed and Augmented Reality, 2006. ISMAR 2006. IEEE/ACM International Symposium on*, pages 253–254, 2006. doi: 10.1109/ISMAR.2006.297831. 16

[153] Gilles Simon. Tracking-by-synthesis using point features and pyramidal blurring. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 85–92, 2011. doi: 10.1109/ISMAR.2011.6092373. 11

[154] Nitin Singhal, In Kyu Park, and Sungdae Cho. Implementation and optimization of image processing algorithms on handheld gpu. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 4481–4484. IEEE, 2010. 32

[155] Sudipta N. Sinha, Jan-Michael Frahm, Marc Pollefeys, and Yakup Genc. Feature tracking and matching in video using programmable graphics hardware. *Mach. Vision Appl.*, 22(1):207–217, 2011. 89

[156] A. Soetedjo and K. Yamada. Fast and robust traffic sign detection. In *Systems, Man and Cybernetics, 2005 IEEE International Conference*

on, volume 2, pages 1341–1346, 2005. doi: 10.1109/ICSMC.2005.1571333. 41

[157] Rahul Swaminathan, Renato Agurto, and Simon Burkard. Mobiliar: Augmented reality for virtual furnishing. *Proceedings of the Workshop on Mobile Vision and HCI (MobiVis). Held in Conjunction with Mobile HCI*, 2012. 12

[158] Lech Świrski, Andreas Bulling, and Neil Dodgson. Robust real-time pupil tracking in highly off-axis images. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 173–176. ACM, 2012. 41

[159] Gabriel Takacs, Vijay Chandrasekhar, Bernd Girod, and Radek Grzeszczuk. Feature tracking for mobile augmented reality using video coder motion vectors. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 141–144. IEEE, 2007. 12

[160] Gabriel Takacs, Vijay Chandrasekhar, Natasha Gelfand, Yingen Xiong, Wei-Chao Chen, Thanos Bismpigiannis, Radek Grzeszczuk, Kari Pulli, and Bernd Girod. Outdoors augmented reality on mobile phone using loxel-based visual feature organization. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, MIR '08, pages 427–434, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-312-9. doi: 10.1145/1460096.1460165. URL `http://doi.acm.org/10.1145/1460096.1460165`. 10, 11

[161] Xusheng Tang, Zongying Ou, Tieming Su, and Pengfei Zhao. Cascade adaboost classifiers with stage features optimization for cellular phone embedded face detection system. In *Advances in Natural Computation*, pages 688–697. Springer, 2005. 31

[162] Qian Tao and R. Veldhuis. Biometric authentication for a mobile personal device. In *Mobile and Ubiquitous Systems: Networking Services, 2006 Third Annual International Conference on*, pages 1–3, 2006. doi: 10.1109/MOBIQ.2006.340409. 14

[163] Christian Teutsch, Dirk Berndt, Erik Trostmann, and Michael Weber. Real-time detection of elliptic shapes for automated object recognition and object tracking. In *Electronic Imaging 2006*. International Society for Optics and Photonics, 2006. 41

[164] A.B. Tillon, I. Marchal, and P. Houlier. Mobile augmented reality in the museum: Can a lace-like technology take you closer to works of art? In *Mixed and Augmented Reality - Arts, Media, and Humanities (ISMAR-AMH), 2011 IEEE International Symposium On*, pages 41–47, 2011. doi: 10.1109/ISMAR-AMH.2011.6093655. 18

[165] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(5):815–830, 2010. 89

[166] P. Tresadern, T.F. Cootes, N. Poh, P. Matejka, A. Hadid, C. Levy, C. McCool, and S. Marcel. Mobile biometrics: Combined face and voice verification for a mobile platform. *Pervasive Computing, IEEE*, 12(1):79–87, 2013. ISSN 1536-1268. doi: 10.1109/MPRV.2012.54. 13

[167] Sam S. Tsai, David Chen, Jatinder Pal Singh, and Bernd Girod. Rate-efficient, real-time cd cover recognition on a camera-phone. In *Proceedings of the 16th ACM international conference on Multimedia*, MM '08, pages 1023–1024, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-303-7. doi: 10.1145/1459359.1459561. URL `http://doi.acm.org/10.1145/1459359.1459561`. 7, 8

[168] Sam S Tsai, David Chen, Gabriel Takacs, Vijay Chandrasekhar, Jatinder P Singh, and Bernd Girod. Location coding for mobile

image retrieval. In *Proceedings of the 5th International ICST Mobile Multimedia Communications Conference*, page 8. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009. 34

[169] Sam S Tsai, David Chen, Vijay Chandrasekhar, Gabriel Takacs, Ngai-Man Cheung, Ramakrishna Vedantham, Radek Grzeszczuk, and Bernd Girod. Mobile product recognition. In *Proceedings of the international conference on Multimedia*, pages 1587–1590. ACM, 2010. 7, 8

[170] Sam S. Tsai, Huizhong Chen, David M. Chen, Georg Schroth, Radek Grzeszczuk, and Bernd Girod. Mobile visual search on printed documents using text and low bit-rate features. In Benot Macq and Peter Schelkens, editors, *ICIP*, pages 2601–2604. IEEE, 2011. ISBN 978-1-4577-1304-0. 13

[171] Hideaki Uchiyama and E. Marchand. Toward augmenting everything: Detecting and tracking geometrical features on planar objects. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 17–25, 2011. doi: 10.1109/ISMAR.2011.6092366. 12

[172] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1582–1596, 2010. 88

[173] Krithika Venkataramani, S. Qidwai, and B. Vijayakumar. Face authentication from cell phone camera images with illumination and temporal variations. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 35(3):411–418, 2005. ISSN 1094-6977. doi: 10.1109/TSMCC.2005.848183. 14

[174] Jonathan Ventura and Tobias Hollerer. Online environment model estimation for augmented reality. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pages 103–106. IEEE, 2009. 17

[175] Paul Viola and Michael Jones. Robust real-time object detection. In *International Journal of Computer Vision*, 2001. 14

[176] Daniel Wagner and Dieter Schmalstieg. History and future of tracking for mobile phone augmented reality. In *Ubiquitous Virtual Reality, 2009. ISUVR'09. International Symposium on*, pages 7–10. IEEE, 2009. 11

[177] Daniel Wagner, Tobias Langlotz, and Dieter Schmalstieg. Robust and unobtrusive marker tracking on mobile phones. In *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*, pages 121–124. IEEE, 2008. 11

[178] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and D. Schmalstieg. Pose tracking from natural features on mobile phones. In *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*, pages 125–134, 2008. doi: 10.1109/ISMAR.2008.4637338. 30

[179] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and D. Schmalstieg. Real-time detection and tracking for augmented reality on mobile phones. *Visualization and Computer Graphics, IEEE Transactions on*, 16(3):355–368, 2010. ISSN 1077-2626. doi: 10.1109/TVCG.2009.99. 30

[180] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Real-time detection and tracking for augmented reality on mobile phones. *Visualization and Computer Graphics, IEEE Transactions on*, 16(3):355–368, 2010. 41

[181] Frank Lorenz Wendt, Stéphane Bres, Bruno Tellez, and Robert Laurini. Markerless outdoor localisation based on sift descriptors for mobile applications. In *Image and Signal Processing*, pages 439–446. Springer, 2008. 11

[182] Wen-Yen Wu and Mao-Jiun J. Wang. Elliptical object detection by using its geometric properties. *Patt. Rec.*, 26(10):1499 – 1509, 1993. 43

[183] Yonghong Xie and Qiang Ji. A new efficient ellipse detection method. In *ICPR*, pages 957–960, 2002. 42

[184] Yingen Xiong and K. Pulli. Fast image stitching and editing for panorama painting on mobile phones. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 47–52, 2010. doi: 10.1109/CVPRW.2010.5543259. 16

[185] Yingen Xiong and Kari Pulli. Fast image labeling for creating high-resolution panoramic images on mobile devices. In *Multimedia, 2009. ISM'09. 11th IEEE International Symposium on*, pages 369–376. IEEE, 2009. 16

[186] Yingen Xiong and Kari Pulli. Mask-based image blending and its applications on mobile devices. In *Proc. of SPIE Vol*, volume 7498, pages 749841–1, 2009. 16

[187] Yingen Xiong and Kari Pulli. Gradient domain image blending and implementation on mobile devices. In *Mobile Computing, Applications, and Services*, pages 293–306. Springer, 2010. 16

[188] Yingen Xiong, Dingding Liu, and K. Pulli. Effective gradient domain object editing on mobile devices. In *Signals, Systems and Computers,*

2009 Conference Record of the Forty-Third Asilomar Conference on, pages 1256–1260, 2009. doi: 10.1109/ACSSC.2009.5469959. 16

[189] Xin Yang and Kwang-Ting Cheng. Ldb: An ultra-fast feature for scalable augmented reality on mobile devices. In *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, pages 49–57. IEEE, 2012. 30

[190] Xin Yang and Kwang-Ting (Tim) Cheng. Accelerating surf detector on mobile devices. In *Proceedings of the 20th ACM international conference on Multimedia*, MM '12, pages 569–578, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1089-5. doi: 10.1145/2393347.2393427. URL `http://doi.acm.org/10.1145/2393347.2393427`. 29

[191] Peng-Yeng Yin and Ling-Hwei Chen. New method for ellipse detection by means of symmetry. *Journal of Electronic Imaging*, 3(1): 20–29, 1994. 43, 52, 61, 62

[192] Kawano Yoshiyuki and Yanai Keiji. Real-time Mobile Food Recognition System. In *The Third IEEE International Workshop on Mobile Vision*, June 2013. 9

[193] H.K. Yuen, J Illingworth, and J Kittler. Detecting partially occluded ellipses using the hough transform. *Image and Vision Computing*, 7 (1):31 – 37, 1989. 62

[194] Si-Cheng Zhang and Zhi-Qiang Liu. A robust, real-time ellipse detector. *Patt. Rec.*, 38(2):273–287, 2005. 43, 44, 55, 61, 62, 67, 70, 71, 77, 78, 79

[195] Yinghua Zhou, Xin Fan, Xing Xie, Yuchang Gong, and Wei-Ying Ma. Inquiring of the sights from the web via camera mobiles. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 661–664, 2006. doi: 10.1109/ICME.2006.262532. 7, 8

[196] ZhiYing Zhou, Jayashree Karlekar, Daniel Hii, Miriam Schneider, Weiquan Lu, and Stephen Wittkopf. Robust pose estimation for outdoor mixed reality with sensor fusion. In Constantine Stephanidis, editor, *Universal Access in Human-Computer Interaction. Applications and Services*, volume 5616 of *Lecture Notes in Computer Science*, pages 281–289. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-02712-3. doi: 10.1007/978-3-642-02713-0_30. URL `http://dx.doi.org/10.1007/978-3-642-02713-0_30`. 11