Universitat Autònoma
de Barcelona

Viewpoint Invariant Features and Robust Monocular Camera Pose Estimation

A dissertation submitted by **Luis Ferraz Colomina** at Universitat Autònoma de Barcelona to fulfil the degree of **Doctor en Informática**.

Bellaterra, December 9, 2015

Director: **Dr. Xavier Binefa Valls**
Universitat Pompeu Fabra
Departament de Tecnologies de la Informació i les Comunicacions

Tutor: **Dr. Ricardo Toledo Morales**
Universitat Autònoma de Barcelona
Departament de Ciències de la Computació

This document was typeset by the author using LaTeX 2_ε.

als meus avis

Acknowledgements

En primer lloc vull agrair al Xavier Binefa l'oportunitat que em va donar ja fa molts anys per entrar i conèixer el món de la recerca, començant pel projecte final de carrera i continuant repescant-me del món del desenvolupament d'eines de gestió per poder tornar al món de la recerca i poder presentar en última instància aquesta tesi doctoral. Moltes gràcies per confiar en mi i gràcies per tots els teus consells.

També estic en deute amb el Francesc Moreno, estic segur que si no fos per ell ara mateix no estaria escrivint aquestes línies. Ell em va ensenyar que collaborar amb altres investigadors permet tenir noves idees i ajuda a fer recerca al més alt nivell. Així doncs, també vull donar les gràcies a tots aquells amb els que he collaborat i espero continuar collaborant en un futur.

Vull agrair sobre manera a tots els meus companys de despatx i departament, molts dels quals considero grans amics, el seu suport quan les coses es torçaven, ajudant-me a veure aquella llum al final del túnel que tot doctorant busca amb força. No puc oblidar aquells inicis de tesi compartint despatx amb el Brais o les aventures amb el Marc, el Ciro i l'Oriol amb els que comparteixo una gran amistat i afecte. Tampoc puc oblidar al Ramon, amic, company de recerca i exsoci. Més tard van arribar molts altres grans companys, dels quals no vull deixar de mencionar al Pol, al Mateo, a l'Adria, a l'Edgar, a l'Eduard i a l'Antonio. També vull agrair a la Vane, la Magda i la resta de membres de la secretaria els bons moments que hem passat i que m'agradaria poder repetir de tant en tant, encara que ja no siguin al passadís de la UPF.

Per altra banda mai podré oblidar a la meva família, sobretot als meus pares que sempre m'han animat a finalitzar aquesta aventura de la qual espero n'estiguin orgullosos encara que mai tant com jo n'estic d'ells. Tampoc puc oblidar als meus avis, que ja no hi son, que amb el seu afecte sempre van fer que tot fos més fàcil.

Impossible no mencionar el paper de Norma en aquesta tesi, que m'ha aguantat durant tots aquests anys –i espero que n'aguanti molts més–, sempre m'ha donat suport i animat per no decaure i així finalitzar aquesta etapa de la nostra vida.

Per tot això i més, **moltes gràcies a tots!**

Resum

La pose de la càmera respecte a una escena del món real determina la projecció perspectiva de l'escena sobre el pla imatge. L'anàlisi de les deformacions entre parelles d'imatges degudes a la perspectiva i la pose de la càmera han portat a molts investigadors en Visió per Computador a tractar amb problemes com, la capacitat per detectar i buscar coincidències de les mateixes característiques locals a diferents imatges o recuperar per cada imatge la pose original de la càmera. La diferencia entre els dos problemes recau en la localitat de la informació que es mostra a la imatge, mentre en el cas de les característiques es busca la invariància local, per al cas de la pose de la càmera es busquen fonts d'informació més global, com ara conjunts de característiques locals.

La detecció de característiques locals és una peça clau per un ampli rang d'aplicacions de Visió per Computador donat que permet buscar coincidències i localitzar regions específiques de la imatge. A la primera part d'aquest treball la invariància de les característiques és abordada proposant algoritmes per millorar la robustesa a les pertorbacions de la imatge, canvis de perspectiva i capacitat de discriminació des de dos punts de vista: (i) detecció precisa de cantonades i taques a les imatges evitant redundàncies mitjançant el seu moviment a través de diferents escales, i (ii) aprenentatge de descriptors robustos. Concretament, proposem tres detectors invariants a escala on un d'ells detecta cantonades i taques simultàniament amb un increment de la cárrega computacional insignificant. També proposem un detector invariant afí de taques. Sobre descriptors, proposem aprendre'ls mitjançant xarxes neurals de convolució i grans conjunts de regions d'imatges anotades sota diferents condicions.

Malgrat que és un tema investigat durant dècades, l'estimació de la pose de la càmera encara és un repte. L'objectiu dels algorismes de Perspective-n-Point (PnP) és estimar la localització i orientació d'una càmera calibrada a partir de n correspondències 3D-a-2D conegudes entre un model 3D prèviament conegut d'una escena real i característiques 2D obtingudes d'una única imatge. A la segona part d'aquesta tesi l'estimació de la pose de la càmera és adreçada amb nous mètodes de PnP, els quals redueixen dràsticament el cost computacional permetent aplicacions en temps real independentment del nombre de correspondències. A més, proporcionem un mecanisme integrat de rebuig de correspondències incorrectes amb una càrrega computacional insignificant i un nou mètode per incrementar la precisió que modela l'error de reprojecció de cada correspondència.

Finalment per escenaris complexos i grans, amb potser centenars de milers de característiques, és difícil i computacionalment car trobar correspondències correctes. En aquest cas, proposem un mètode robust i precís per estimar la pose de la càmera. El nostre mètode

s'aprofita de classificadors d'alt nivell, que estimen la pose de la càmera de manera poc precisa, per tal de restringir les correspondències a ser utilitzades pels nostres precisos algorismes de PnP.

Abstract

Camera pose with respect to a real world scene determines the perspective projection of the scene on the image plane. The analysis of the deformations between pairs of images due to perspective and camera pose have led many Computer Vision researchers to deal with problems such as, the ability to detect and match the same local features in different images or recovering for each image its original camera pose. The difference between both problems lie in the locality of the image information, while for local features we look for local invariance, for camera pose we look for more global information sources, like sets of local features.

Local feature detection is a cornerstone of a wide range of Computer Vision applications since it allows to match and localize specific image regions. In the first part of this work local invariance of features is tackled proposing algorithms to improve the robustness to image perturbations, perspective changes and discriminative power from two points of view: (i) accurate detection of non-redundant corner and blob image structures based on their movement along different scales, and (ii) learning robust descriptors. Concretely, we propose three scale invariant detectors, detecting one of them corners and blobs simultaneously with a negligible computational overhead. We also propose one blob affine invariant detector. In terms of descriptors, we propose to learn them using Convolutional Neural Networks and large datasets of annotated image regions under different image conditions.

Despite being a topic researched for decades camera pose estimation is still an open challenge. The goal of the Perspective-n-Point (PnP) problem is to estimate the location and orientation of a calibrated camera from n known 3D-to-2D point correspondences between a previously known 3D model of a real scene and 2D features obtained from a single image. In the second part of this thesis camera pose estimation is addressed with novel PnP approaches, which reduces drastically the computational cost allowing real-time applications independently of the number of correspondences. In addition, we provide an integrated outlier rejection mechanism with a negligible computational overhead and a novel method to increase the accuracy by modelling the reprojection error of each correspondence.

Finally in the case of complex and huge scenarios, with maybe hundreds of thousands of features, is difficult and computationally expensive to be able to find correct 3D-to-2D correspondences. In this case, a robust and accurate top-down approach for camera pose estimation is proposed. Our approach takes advantage of high-level classifiers, which estimates a rough camera pose, in order to constrain the 3D-to-2D correspondences to be used by our accurate and robust to outliers PnP method.

Contents

1 Introduction **1**
- 1.1 Invariant Interest Point Detection . . . 2
- 1.2 Interest Point Description . . . 3
- 1.3 Robust Monocular Camera Pose Estimation . . . 4
- 1.4 Monocular Camera Pose Estimation in Complex Scenarios . . . 6
- 1.5 Thesis Overview . . . 7

2 Scale & Affine Invariant Interest Points **9**
- 2.1 Introduction . . . 10
  - 2.1.1 Overview . . . 11
- 2.2 Scale Invariance . . . 11
  - 2.2.1 Multi-Scale Representation . . . 11
  - 2.2.2 Operators to Detect Image Structures . . . 13
  - 2.2.3 Evolution and Automatic Scale Selection of Image Structures . . . 14
- 2.3 Affine Invariance . . . 18
  - 2.3.1 Affine Invariance from Geometry-based Operators . . . 18
- 2.4 Curvature Analysis on Grayscale Images . . . 20
  - 2.4.1 Principal Curvatures . . . 22
  - 2.4.2 First Fundamental Form . . . 22
  - 2.4.3 Second Fundamental Form . . . 23
  - 2.4.4 Principal Curvatures Estimation . . . 23
  - 2.4.5 Gaussian and Mean Curvature . . . 24
  - 2.4.6 Relations with Other Common Operators . . . 25
- 2.5 Scale Invariant Detection of Non-redundant Interest Points . . . 29
  - 2.5.1 Proposed Non-redundant Detector for Simultaneous Detection of Corner and Blob Structures . . . 31
  - 2.5.2 Computational Complexity . . . 32
- 2.6 Affine Invariant Blob Detection Fitting Gaussian Functions . . . 33
  - 2.6.1 Weighted Non-linear Least Squares Algorithm in order to Fit Gaussian Functions on Images . . . 33
  - 2.6.2 Proposed Shape Adaptation Algorithm . . . 36
  - 2.6.3 Computational Complexity . . . 38
- 2.7 Experimental Evaluation . . . 38
  - 2.7.1 Repeatability and Matching . . . 39

2.7.2 Precision and Recall . . . 44
2.7.3 Over-representation . . . 47
2.7.4 Homography Estimation . . . 48
2.8 Summary . . . 51

3 Learning of Discriminative Local Image Descriptors 53
3.1 Introduction . . . 53
3.1.1 Overview . . . 55
3.2 Related CNN-Based Approaches . . . 56
3.3 Introduction to Artificial Neural Networks and CNNs . . . 57
3.3.1 Back-propagation Procedure . . . 58
3.3.2 Learning Procedure . . . 58
3.3.3 Minibatch Stochastic Gradient Descent . . . 59
3.3.4 Convolutional Neural Networks . . . 59
3.4 Learning Deep Descriptors . . . 61
3.4.1 CNN-based Descriptors . . . 61
3.4.2 Stochastic Sampling Strategy and Mining . . . 63
3.4.3 Learning . . . 63
3.5 Experimental Evaluation . . . 64
3.5.1 Depth and Fully Convolutional Architectures . . . 65
3.5.2 Hidden Units Mapping, Normalization, and Pooling . . . 66
3.5.3 Mining . . . 66
3.5.4 Generalization & Comparison to State of the Art . . . 68
3.5.5 Qualitative Analysis . . . 71
3.5.6 Robustness to Rotation . . . 72
3.5.7 Wide-baseline Matching . . . 74
3.5.8 Deformation and Varying Illumination Dataset . . . 74
3.6 Summary . . . 77

4 Robust Monocular Camera Pose Estimation 79
4.1 Introduction . . . 80
4.1.1 Overview . . . 82
4.2 Related PnP Approaches . . . 82
4.3 Real-Time and Scalable PnP Solution . . . 83
4.3.1 Problem Statement . . . 84
4.3.2 Revisiting the EPnP Linear Formulation . . . 84
4.3.3 Reformulating the EPnP Linear Formulation . . . 86
4.3.4 Estimating Scale and Absolute Pose . . . 87
4.3.5 The Planar Case . . . 88
4.4 Algebraic Outlier Rejection in the PnP Problem . . . 89
4.4.1 Related Outlier Rejection Schemes . . . 89
4.4.2 Robust Null-Space Estimation . . . 91
4.4.3 Experimental Results . . . 93
Synthetic Experiments . . . 94
Number of Correspondences and Noise . . . 94
Robustness to Outliers . . . 97

Real Images . . . 101
4.5 Leveraging Feature Uncertainty in the PnP Problem . . . 102
4.5.1 Related Methods to Deal with Uncertainty . . . 104
4.5.2 Integration of Feature Uncertainties in Our Linear Formulation . . . 104
4.5.3 Dealing with Feature Uncertainties on Real Images . . . 106
4.5.4 Experimental Results . . . 108
Synthetic Experiments . . . 111
Real Images . . . 112
4.6 Summary . . . 115

5 Monocular Camera Pose Estimation in Complex Scenarios 117
5.1 Introduction . . . 117
5.1.1 Overview . . . 119
5.2 Related Methods for Camera Pose Estimation . . . 120
5.3 Building the 3D Model . . . 121
5.4 Merging Appearance and Geometric Methods . . . 122
5.4.1 Coarse Pose Estimation . . . 122
5.4.2 Fine Pose Estimation . . . 123
5.5 Experimental Results . . . 124
5.5.1 Image Retrieval for Coarse Pose Estimation . . . 125
5.5.2 Efficiency Assessment . . . 127
5.5.3 Accuracy . . . 128
5.6 Summary . . . 130

6 Conclusions and Future Work 131
6.1 Interest Point Detection . . . 131
6.2 Interest Point Description . . . 133
6.3 Robust Monocular Camera Pose Estimation . . . 134
6.4 Monocular Camera Pose Estimation in Complex Scenarios . . . 135
6.5 Future Work Combining Proposed Algorithms . . . 136

Bibliography 137

List of Tables

2.1 Curvature shape classification. Combining Gaussian curvature K and Mean curvature M the local shape of an image region is described. 24

2.2 Comparison in terms of computational complexity and cost between our scale invariant detectors and the Hessian-Laplace and Harris-Laplace detectors. $K(I)$ denotes the Gaussian curvature, $H(I)$ the Hessian matrix and $\mu(I)$ the second moment matrix computed for each image point. For each candidate point, ∇K is the gradient ascent/descent applied on the upper scale-space level and $\#3$ $(dxx + dyy)$ is a convolution by the Laplacian operator in 3 adjacent scale-space levels. $\#s$ denotes the number of scales and $\#m$ the number of candidate interest points. 33

2.3 Comparison in terms of computational complexity and cost between our affine detector and the Hessian-Affine detector. $\boldsymbol{\mu}(x)$ is the second moment matrix computed for a point $\mathbf{x}$ and $\mathbf{H}(\mathbf{x}_{neighbors})$ is the Hessian matrix computed in an image region near to $\mathbf{x}$. $\#3$ $(dxx + dyy)$ is a convolution by a Laplacian operator in 3 adjacent scale-space levels. $\#n$ denotes the number of iterations per point, $\#s$ denotes the number of scales for the re-detection step and $\#m$ is the number of scale-invariant interest points detected in the re-detection step. 38

2.4 Ratio of image regions over-represented by the scale invariant detectors for each evaluation sequence. The minimum ratio for each sequence is shown in bold. Self-repeatability has been computed with an overlapping error $< 30\%$. 47

2.5 Ratio of image regions over-represented by the affine invariant detectors for each evaluation sequence. The minimum ratio for each sequence is shown in bold. Self-repeatability has been computed with an overlapping error $< 30\%$. 48

2.6 Comparison of scale invariant detectors in terms of computational cost of the matching process for the registration task assuming interest points were previously extracted. The first column shows the mean number of features detected on each dataset image. The second column shows the computational cost required to match interest points and estimate each homography for each image pair. 50

2.7 Comparison of affine invariant detectors in terms of computational cost of the matching process for the registration task assuming interest points were previously extracted. The first column shows the mean number of features detected on each dataset image. The second column shows the computational cost required to match interest points and estimate each homography for each image pair. 51

3.1 Four top rows: effect of increasing batch size, without mining. Four bottom rows: with mining. Mining factors indicate the samples considered (s_p, s_n), the hardest 128 of which are used for training. Column 5 indicates the fraction of the computational cost spent mining hard samples. These experiments correspond to the validation set. 67
3.2 PR AUC for the generalized results over the three MVS dataset splits, for different mining factors. 69
3.3 Generalized results: PR AUC over the three MVS splits, and a new split with data from all three sets, against SIFT, BinBoost [134], and VGG [122]. We re-train VGG with data from two sets (rows 1-3) and all sets (row 4). 70
3.4 Computational cost for one descriptor (in batch). 70
3.5 Stereo matching, '3' vs '4'. 75
3.6 Stereo matching, '3' vs '5'. 75
3.7 Stereo matching, '3' vs '6'. 75
3.8 Stereo matching, '3' vs '7'. 75
3.9 Stereo matching, '3' vs '8'. 76
3.10 Results on the dataset of [121]. We evaluate over three different settings, corresponding to deformation changes only (Def.), illumination changes only (Ill.), and both simultaneously (Def.+Ill.). We show the mean accuracy of descriptor matches and highlight the top-performing descriptor for each of setting, in bold. 77

5.1 Performance of the proposed approach in the two models for different values of the parameter k. We also consider the case when *all* the model points are considered. 124
5.2 Time values in seconds for the different methods evaluated. The upper part of the table compares the two methods evaluated for the coarse estimation of the pose (Bag of words vs. Deep network features). The lower part reports the computation time of the methods used for fine pose estimation (RANSAC + OPnP vs. REPPnP). These results are obtained by computing the mean computation time for all the evalaution images. 128
5.3 Rotation and translation errors for the images shown in Figure 5.7 using REPPnP and RANSAC+OPnP with $k = 6$. 130

List of Figures

1.1 Example of over-detected image regions using scale invariant detectors. Left: Using a state-of-the-art blob detector [90]. Rigth: Using a state-of-the-art corner detector [91]. 3

1.2 Example of image regions under different viewing conditions such us illumination, blurring or perspective changes. Each column represent an specific image region. These samples belong to the Multi-view Stereo dataset (MVS) [9]. 4

1.3 Examples of the challenges we propose to solve. Top: Example of inlier/outlier correspondences, green lines denote inlier correspondences and black lines denote outlier correspondences. Bottom: Feature uncertainties on a real image region. Green dots for each feature denote the noise computed by reprojecting features detected in other images and black ellipses denote the noise model for each feature. 5

1.4 Example of the closest images in the training set to a given query image. Images (a), (b) and (c) denotes the three closest images in the training set. . . 6

2.1 Different multi-scale representations. Succesive smothing (left), pyramid combining smoothing and sampling (center) and hybrid approach where sampling is performed after several smoothed scales (right). In all these representations the smooth is computed convolving by a Gaussian kernel with standard deviation σ. 12

2.2 Automatic scale selection looking for local extreme responses. The response of the pixel X is compared with its 26 neighbors in 3×3 regions at the current and adjacent scales [84] (left). Example of evolution along the scales of a concrete blob structure created in a corner structure (red line) on a synthetic image using our approach (right). Green dots denotes extreme responses along the scales. 15

2.3 Examples of over-detection where interest points are represented by red circles. In the first row, over-detection on a synthetic image containing a corner structure is shown. Left image shows over-detection on a specific scale and right image along different scales. In the bottom image, over-detection using a standard blob detector, Hessian-Laplace [90] detector, on a real image is presented. 17

2.4 Comparison between the surfaces provided by the Gaussian curvature operator (left image) and the determinant of the second moment matrix (right image). Operators are evaluated on a synthetic image containing a corner structure. Being the red lines the level curves of the surfaces, the + simbol indicates the positive region of the surface and the - symbol indicates the negative region. 19

2.5 Gaussian curvature K for three different kinds of surfaces where $\mathbf{N}$ is the surface normal and k_1 and k_2 are the two principal curvatures: Saddle surface with $K < 0$ ($k_1 < 0$ and $k_2 > 0$), blob surface with $K > 0$ ($k_1 < 0$ and $k_2 < 0$) and edge surface with $K = 0$ ($k_1 = 0$ and $k_2 < 0$). Being the red lines the projections of the two main directions on the surface. 21

2.6 Evolution of the Gaussian curvature K in a corner structure. Corner structures can be considered as image regions that can be represented as elliptical, parabolic or hyperbolic regions at the same time. 25

2.7 Relations between the responses of several operators on a corner structure. Green, red and blue lines denote negative, zero and positive responses. Green and blue dots denote the extreme responses. 26

2.8 Relations between the responses of several operators on a blob structure. Green, red and blue lines denote negative, zero and positive responses. Green and blue dots denote the extreme responses. 27

2.9 Operator responses along the pixels belonging to the diagonal from the top-left corner to the bottom-rigth corner of the image showing the synthetic corner in Figure 2.7. Vertical dashed lines denote the corner location. Responses of M, $trace(H)$, $det(\mu)$ and Harris have been scaled. 28

2.10 Operator responses along the pixels belonging to the diagonal from the top-left corner to the bottom-rigth corner of the image showing the synthetic square in Figure 2.8. Vertical dashed lines denote the spatial location of corners and blob center. Responses of M, $trace(H)$, $det(\mu)$ and Harris have been scaled. 28

2.11 From scale-space levels to blob trajectories. (a) shows 3 hypothetical consecutive scale-space levels computed using Gaussian curvature on a 1D image (for clarity). (b) shows the trajectories obtained along the stacked scale-space levels shown in (a). Blob and corner trajectories P are represented by the green and yellow regions respectively. Being each trajectory denoted as set of relations $r_{i,j}$ between candidate interest points ζ_i^k. 30

2.12 Output of our scale invariant blob and corner detector on a synthetic image. The estimated blob and corner trajectories are shown in blue and yellow, respectively. In red are shown the selected interest points. 32

2.13 Output of our scale invariant blob detector (top image) and output of the proposed shape adaptation algorithm (bottom image). Trajectories appear on the top image in blue and interest points appear in red in both images. Comparing both images is noticeable that some scale invariant blobs do not converge to an affine blob. 34

2.14 Example of execution of our shape adaptation algorithm. (a) shows the initial isotropic blob in blue ($K = 0.0252$) and the solution (red ellipse) found in the first iteration of our method. (b) and (c) show the two next iterations of the algorithm. The green rectangles are the image regions used in the modeling process. In (d) the final result is shown, where the green ellipse represents the solution that maximizes the *Gaussian curvature* ($K = 0.0418$). 37

2.15 Comparison of scale invariant detectors in terms of repeatability, matching and precision/recall from evaluation sequences with viewpoint changes and blurring. 40

2.16 Comparison of scale invariant detectors in terms of repeatability, matching and precision/recall from evaluation sequences with zoom+rotation, JPEG compression and illumination changes. 41

2.17 Comparison of affine invariant detectors in terms of repeatability, matching and precision/recall from evaluation sequences with viewpoint changes and blurring. 42

2.18 Comparison of affine invariant detectors in terms of repeatability, matching and precision/recall from evaluation sequences with zoom+rotation, JPEG compression and illumination changes. 43

2.19 Example of interest points detected using our blob and corner detector (top) and Hessian-Laplace detector (bottom) on an image of the graffiti sequence. Our detector locates 2582 interest points ($\approx 50\%$ corners) and the Hessian-Laplace detector 2477 interest points. In pictures interest point locations are depicted as red crosses and treir scales as red circles. 46

2.20 Example of accuracy on a real image. In (a) and (b) affine invariant blobs found by our shape adaptation algorithm converge to the real blob: (a) initialized by our proposed scale invariant blob detector and (b) initialized by the Hessian-Laplace detector. In (c), blobs found by the Hessian-Affine detector are shown, which do not converge to real blobs. Red circles are the initially scale invariant detected blobs and black ellipses are the shape-adapted blobs. . 48

2.21 Comparison of scale invariant detectors in terms of $\varepsilon_{\tilde{H}_i}$ from evaluation sequences. $\varepsilon_{\tilde{H}_i}$ for each detector and sequence has been computed as the $median(\varepsilon_{\tilde{H}_i})$ of 10 Ransac-based homography estimations. 49

2.22 Comparison of scale invariant detectors in terms of $\varepsilon_{\tilde{H}_i}$ from evaluation sequences. $\varepsilon_{\tilde{H}_i}$ for each detector and sequence has been computed as the $median(\varepsilon_{\tilde{H}_i})$ of 10 Ransac-based homography estimations. 50

2.23 Examples of our affine blob detector for image registration tasks on image pairs with large variations. Green rectangles represent the initial image registered using the correct homography H and yellow rectangles using $\tilde{H}$). In the first row there are two examples of image pairs with high variations in scale and rotation $\varepsilon_{\tilde{H}} = 0.67$ and $\varepsilon_{\tilde{H}} = 2.70$. In the second row, the first image pair suffers large variations in the point of view ($\varepsilon_{\tilde{H}} = 4.35$) and the second image pair suffers a large degradation due to compression ($\varepsilon_{\tilde{H}} = 0.55$) 52

3.1 To train models with Siamese networks, we need pairs of corresponding and non-corresponding samples. (a) We use t-SNE [137] to display $\sim 100\,64\times64$ patches of 12 3D points from different images (see Figure 3.5 for examples). Corresponding regions are drawn with the same color. (b) We single out the red-circled patch, belonging to the blue point cloud, and consider all of its potential pairings. The line length encodes the closeness between this region and the rest: positive matches in blue, negative in red. Most pairs are easy to discriminate and ineffectual for training. (c) We mine the samples to obtain the closest negative (shortest red line) and the most distant positive (longest blue line). This simple strategy allows us to train discriminative networks over large datasets. 54

3.2 Schematic of a Siamese network, where pairs of input patches are processed by two copies of the same CNN, sharing all the internal parameters. 55

3.3 Example of a Convolutional Neural Network (CNN) for digits recognition. Convolutional and subsampling layers have associated a label, $a@b\times c$, where a denotes the number of feature maps and $b \times c$ the number of nodes for each feature map. The local receptive fields are shown as small black squares. Figure reproduced from [69]. 60

3.4 Different CNN configurations evaluated to learn descriptors. Each layer is described as $axbxcxdpool$ denoting the number of feature maps a, the size of the local receptive fields bxc and the size of the pooling and normalization region d. 62

3.5 Pairs of corresponding samples from the MVS dataset. Top: 'Liberty' (LY). Middle: 'Notre Dame' (ND). Bottom: 'Yosemite' (YO). Right: we compute the absolute value of the pixel difference between corresponding regions on each set and show their mean/std (black corresponds to small values). 64

3.6 Precision-Recall results for the evaluated architectures. Effect of network depth (up to 3 CNN layers), and fully convolutional networks vs networks with final fully-connected layer (NN1). PR AUC on the validation set for the top-performing iteration. 66

3.7 Precision-Recall results for experiments on hidden units, normalization, pooling. Fully convolutional CNN3 models with Tanh and ReLU, without normalization, and with max pooling instead of L2 pooling. The best results are obtained with Tanh, normalization, and L2 pooling (i.e. CNN3). PR AUC on the validation set for the top-performing iteration. 67

3.8 Effect of mining on filters. *Top:* first layer filters, before mining. *Middle:* same, after aggressive mining, filters are initialized with the filters in the top row. *Bottom:* difference between top and middle row. For visualization purposes all values are normalized such that the minimum value is displayed in black and the maximum in white for all the values in the group. Note that mining accentuates the maxima and minima of the filters. 68

3.9 PR curves for the generalized results over the three MVS dataset splits, for different mining factors. 69

3.10 Generalized results: PR curves over the three MVS splits, and a new split with data from all three sets, compared to SIFT, Binboost [134], and VGG [122]. We re-train VGG with data from two sets (columns 1-3) and all sets (column 4). 71

3.11 Samples of matches retrieved with our descriptor. Each row depicts the reference patch (in green) and the top matching candidates, sorted from left to right by decreasing similarity. The ground truth match is highlighted in blue. Left: Examples where the ground truth match is retrieved in the first position. Right: Examples in which the ground truth match is ranked at positions 2-6. The last row shows a failure case, highlighted in red, where the correct match is ranked 632/1000. 72

3.12 Robustness to Rotation. 73

3.13 Samples of the wide-baseline stereo dataset of [126]. 74

3.14 Samples of the deformation and varying illumination dataset of [121]. 76

4.1 PnP problem definition: given an input image and a known textured 3D model, the problem is to estimate the camera pose w.r.t. the 3D model in terms of $[\mathbf{R}|\mathbf{t}]$. Valid 3D-to-2D correspondences are denoted as black lines from the camera center (origin of the camera coordinates) trougth interest points on the 2D image up to 3D points on the model. 80

4.2 Example of the proposed Procrustes-based alignment. Left square contains, a comparison between the ground-truth $\mathbf{x}$ estimated without noise in the correspondences and the $\mathbf{x}$ estimated with noise. The blue control points are expresed in world coordinates. In the rigth side an example showing the evolution of $\mathbf{x}$ iterating our proposed method. First the control points in world coordinates (blue dashed) are aligned with the noisy $\mathbf{x}$ solution using equation (4.16), afterwards in orange is shown the new $\mathbf{x}$ solution computed using equation (4.18). 88

4.3 Our approach to simultaneously estimate pose and reject outliers. Top: Sample result, where green and black lines indicate the inlier and outlier matches, respectively. The 3D model is overlaid in blue in the input image, using the estimated pose. The overall process (outlier rejection and pose estimation) is done in less than 5 ms. Bottom-left: Our PnP solution iteratively computes the null-space of a linear system. Here we plot the matrix of this system, where rows represent correspondences. The error per match of the estimated null-space is plotted on the side of each matrix, and is used to detect outliers (black rows). The convergence of the null-space is plotted at the bottom of each matrix, where dotted lines depict the true null-space. Bottom-right: The null-space gives an estimate of the 3D model in camera coordinates. We finally compute the pose using the method proposed in Section 4.3.4. 90

4.4 Example of outlier rejection with 100 inliers (shown on the left-most part of the graphs) and 100 outliers. The blue line represents the algebraic error of each correspondence, and the horizontal black line is the threshold $\max(\varepsilon_{\text{max}}, \delta_{\text{max}})$ used in Eq. 4.21 to classify each match as inlier or outlier. δ_{max} is the only threshold we need to provide, and we set it to 0.017, which approximately represents an image noise of $\tau = 10$ pixels for a focal length $f = 800$. 93

4.5 Synthetic experiments for (a) non-planar objects and (b) planar objects. Varying the number of correspondences. 95

4.6 Synthetic experiments for (a) non-planar objects and (b) planar objects. Varying the amount of Gaussian noise. The color codes and line styles are the same as those used in Figure 4.5 96

4.7 Running time. Varying the number of outlier-free correspondences (left) and varying the % of outliers (right). In the left, the color codes and line styles are the same as those used in Figure 4.5. 97

4.8 Synthetic experiments for different levels of outliers in terms of relative error of the null-space of $\mathbf{Mx} = \mathbf{0}$ computed without removing outliers (left); and algebraic error for the inlier and outliers correspondences projected onto this null space (right). The deviation levels of the initially estimated null-space let to clearly identify inliers and outliers, based on their algebraic error. . . . 98

4.9 Synthetic experiments for different levels of outliers for (a) non-planar and (b) planar objects. Varying the % of outlier correspondences. The real inliers are fixed to 100 with $\sigma = 2$. Note that even for small percentage of outliers the obtained solutions are completely wrong. 99

4.10 Synthetic experiments, varying the % of outlier correspondences. The real inliers are fixed to 100 with $\sigma = 5$. The symbol $*$ represents the last value before the breakdown point. 100

4.11 Real image examples. Top: Inlier and outlier correspondences found by our method between a model (left) and a test image (rigth). Bottom: Examples of fitting using our method. Green and black dots represent the inliers and outliers respectively. 101

4.12 Example with a planar object. Estimated camera pose with respect to the 3D object model (left) for the book shown in the image (right). Green and black dots represent the inliers and outliers respectively and blue rectangle the object model projected on the image. 102

4.13 PnP problem with noisy correspondences. We assume a set of 2D feature points is given, with a particular noise model for each feature point, which is represented by one 2D ellipse. We also assume the correspondences (black lines) and 3D points (blue points) with respect to a 3D model are known. Our approach estimates a solution of the PnP problem that minimizes the Mahalanobis distances $\Delta\mathbf{u}_i$ shown in the left (red arrow). In the left the green rectangle and blue dots are the true projection of the 3D model. 103

4.14 Feature uncertainties on real images. Top: Example of a grid of reference views where uncertainties must be estimated. Bottom-Left: Example of feature point clouds (in green) and their Gaussian models (black ellipses) for V_1. Bottom-Right: Results of feature matching and uncertainties alignment against a test image. 107
4.15 Synthetic experiments for (a) non-planar objects and (b) planar objects. Varying the number of correspondences. 109
4.16 Synthetic experiments for (a) non-planar objects and (b) planar objects. Varying the amount of uncertainty. 110
4.17 Computation time for the non-planar case (left) and the planar case (right) in synthetic experiments varying the number of correspondences. The color codes and line styles are the same as those used in Figure 4.15. 112
4.18 Experiments with real images of a planar object. Each column depicts the errors (median and quartiles) as the camera pose move away from the reference view, in rotation (first row) and translation (second row) terms. In red we remark the angular distance we have selected to generate the grid of reference images. 113
4.19 Experiments with real images of a planar object. In these images the results provided by the EPPnP (red rectangle), the OPnP (yellow rectangle) and the CEPPnP (blue rectangle) are shown. 114

5.1 Problem definition: Given an input image (a) and a known textured 3D model of the environment (b), the problem is to estimate the pose of the camera that captured the image with respect to the model (c). The main challenge addressed in this chapter is to perform the correspondence of points efficiently and reliably for complex 3D that contain a large number of points. In this example, the model of the Sagrada Familia has over 100,000 points. 118
5.2 Overall scheme of the proposed method for accurate camera pose estimation using highly complex models. 119
5.3 Courtyard 3D model built and two sample images. 121
5.4 Comparison of the L2 distances between distance deep network features and the known angular distance between the test images and training images. Each row is sorted according to the angular distance. We can see a strong correlation between images that are close to each other and their descriptors obtained from the deep network. 125
5.5 Example of one input image of the Sagrada Familia (top) and one input image of the Courtyard (bottom) in the validation sets. Next to the input images similar images in the training set selected by our image retrieval algorithm ($k = 4$ and $k = 5$) and a plot showing the distance to the training images. . . 126
5.6 Analysis of computation time. Left: Computation time of the approaches with different values of k. The case *All* is equivalent to not doing any appearance based pre-computation over the 3D model. 127
5.7 Examples of pose estimation results for the Sagrada Familia (top) and Courtyard (bottom) models. For each image we show the reprojected 3D coordinates of the SIFT points using the groundtruth $\mathbf{R}$ and $\mathbf{t}$ (from Bundler) and the ones estimated by both REPPnP and RANSAC+OPnP with $k = 6$. 129

Chapter 1

Introduction

In the human eyes the images are formed on the retina and then our brain perceives a perspective projection of the real world. This perspective projection is mathematically explained by the pinhole camera model, which also explains the image formation process in most of the lens-based cameras. Since perspective depends on the camera location and orientation with respect to the scene, a small change in the location or orientation of the camera prompts a change in the perspective projection. This fact allows, from a single perspective image point of view recover the camera pose and having into account multiple images, comparing their perspective variations, to recover 3D information about a real world scene or object. In fact, the separation between the eyes prompts small differences between the perspective seen by each eye, allowing to our brain, apparently without effort, understand the world in three dimensions instead of only two dimensions. At the same time, our brain is able to estimate easily its location and orientation with respect to the real world even with only one eye. This work is centred in this second case, where using only single perspective images the camera pose parameters are accurately recovered with respect to known real world scenes.

Camera pose estimation in term of location and orientation from single images has been deeply studied by the Computer Vision community for decades. Most of the proposed methods are based in low-level algorithms, which analyse the perspective from a local point of view. Probably the most relevant low-level algorithms are those devoted to locate and describe the same local image regions along different views of a given scene or object. These local image regions, which are usually called features, not just look for invariance to changes in the perspective, but also light conditions, noise or blurring providing robust features to the matching algorithms. The matching task provides correspondences between images and real world scenes being the fundamental information source in order to recover the camera pose. For this reason in this work, in addition, we also studied the local behaviour of the image regions and their description.

In this chapter we introduce the main research areas and the problems addressed in each one from a low-level to high-level point of view. At the end of the chapter we present an overview of this manuscript.

1.1 Invariant Interest Point Detection

In the literature, there are several approaches in order to deal with the detection of interest image regions, which are usually called interest point detectors. The aim of interest points is the detection of local image regions, which can reliably be found in other images containing the same regions under perspective changes or other viewing conditions as blurring or illumination. Some approaches look for underlying local image structures such as, corners or blobs, assuming that these structures are discriminative enough while others look for highly discriminant image regions independently of the underlying structure.

All these local image regions are determined by a specific location, scale and shape. The intrinsic geometry of the image regions allow to categorize the interest point detectors in two main groups:

- **Scale invariant detectors**, which try to determine the most representative scales of each interest point in each image, trying to obtain the same interest points independently of the distance between the camera and the scene. These detectors deal with perspective variations assuming isotropic scale variations by means of an exhaustive analysis called multi-scale image analysis [17].

- **Affine invariant detectors**, which try to estimate the most representative underlying shape and scale of each interest point in each image, trying to obtain the same interest points independently of the camera viewpoint and distance with respect to the scene. These detectors try to deal with any perspective variation assuming image regions belongs to a planar region in the scene and considering that an affine transformation is a good approximation of the true perspective projection. These detectors, with exceptions [88], assumes that perspective variations will be always roughly detectable by scale invariant detectors, therefore they can be considered a post-processing step applied on scale invariant interest points [82, 91]. Hence, affine invariant detectors increase the precision in the location of image regions although with an increment in the computational cost.

Interest point detectors are widely used because they provide a sparse representation of the images, allowing to discard huge image regions. However, most of the detectors tend to over-detect image regions increasing the redundancy, reducing the sparsity, reducing the distinctiveness of each region and making difficult subsequent tasks as the matching process to identify each region along the images. Moreover, interest point detectors are usually designed to detect very specific kinds of image regions. Therefore, to detect complementary image regions several detectors must be applied. In Figure 1.1 an example of over-detection provided by two state-of-the-art scale invariant detectors is shown.

Contributions: Our contributions in this field are four invariant detectors avoiding over-detection. Improving the sparsity, the distinctiveness and simplifying the matching task between regions. We propose three novel scale invariant detectors. Concretely, one corner detector, one blob detector and one detector that simultaneously detects blob and corner structures with a negligible overhead with respect to compute only blobs or corners. We also propose a novel affine invariant detector of blob structures, based in our blob scale invariant

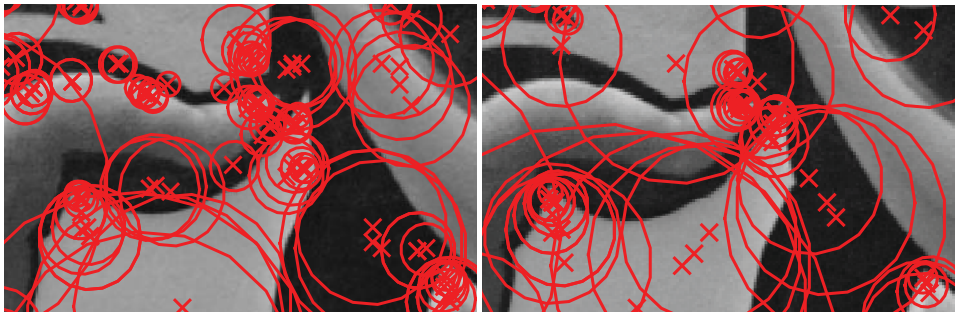


Figure 1.1: Example of over-detected image regions using scale invariant detectors. Left: Using a state-of-the-art blob detector [90]. Rigth: Using a state-of-the-art corner detector [91].

detector, that simultaneously computes the location and affinity of each region in contrast with the most standard method [91], which estimates location and affine shape in separated steps.

1.2 Interest Point Description

Once the invariant interest points are detected their underlying image regions have to be matched with the image regions found in other images. This matching can be performed by directly describing the regions from their pixel intensities or colours, obtaining descriptors of high dimensionality and highly sensitive to the viewing conditions such as perspective, noise or illumination. The most known descriptor that reduces drastically these issues is the SIFT descriptor [84].

At this moment, descriptors can be categorized in two groups. The main group, descriptors like SIFT and most of the descriptors in the literature that are hand-crafted. These descriptors follow a predefined methodology without having into account the specific discriminatory power contained in each image region. In addition, robustness to viewing conditions is provided by the predefined methodology, making researchers responsible of this task.

The other group of descriptors is relatively recent, it is formed by descriptors based on machine learning techniques allowing to learn how to compute the most robust and discriminative descriptors. Given a set of interest points obtained from several images representing all of them the same image region, these descriptors are able to learn to capture invariance to the viewing conditions represented in the set while at the same time, maximize the discriminatory power of each region. Figure 1.2 shows several examples of detected regions representing the same feature under different viewing conditions.

The second group of descriptors show better performance than the first one. Since they are based on learning, they are able to learn to deal with very difficult issues such as, location errors of the interest points, huge in-plane perspective variations or even those perspective variations produced by image regions that are not planar in the real 3D scene and can even be partially affected by self-occlusions.

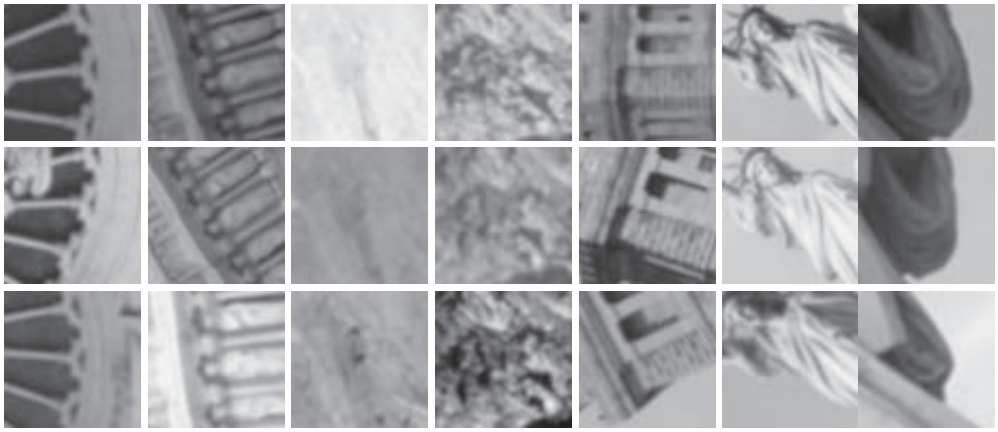


Figure 1.2: Example of image regions under different viewing conditions such us illumination, blurring or perspective changes. Each column represent an specific image region. These samples belong to the Multi-view Stereo dataset (MVS) [9].

Contributions: Our contribution in this area is a new machine learning based descriptor that is able to improve the discriminative power and robustness to viewing conditions of the state-of-the-art descriptors. Our descriptor is based on Convolutional Neural Networks (CNNs), a recent technique that for image classification has outperformed existing approaches by a large margin [66].

1.3 Robust Monocular Camera Pose Estimation

The monocular camera pose estimation problem consists on accurately determine the camera extrinsic parameters, location and orientation, from a single perspective image with respect to the 3D coordinates of a real world scene. In order to increase the accuracy most of the solvers require a calibrated camera allowing to take advantage of the internal geometry of the camera.

The fundamental information source used during the camera pose estimation are the 3D-to-2D correspondences between the 3D scene and the 2D image. These 3D-to-2D correspondences are typically computed by means of a matching task between a set of image features, obtained by means of the detection and description of interest points, and a set of previously known 3D scene features, which will be considered from now on the 3D scene model.

Depending on the number of 3D-to-2D correspondences the camera pose estimation problem can be also called the Perspective-n-Point (PnP) problem, where n represents the number of correspondences. Initial approaches solved the problem very efficiently for very small numbers of correspondences, e.g. P3P and P4P solvers, however they have the drawback of being very sensitive to noise. This problem was attenuated by exploiting data redundancy, which was achieved by solvers without a predefined number of correspondences, which are simply called PnP solvers.

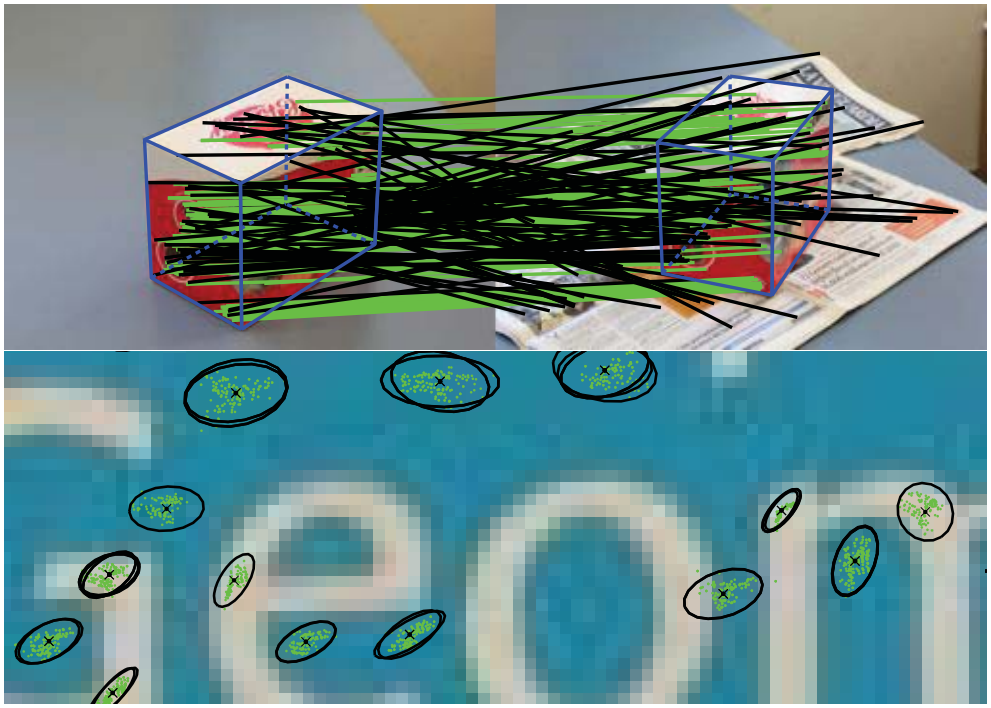


Figure 1.3: Examples of the challenges we propose to solve. Top: Example of inlier/outlier correspondences, green lines denote inlier correspondences and black lines denote outlier correspondences. Bottom: Feature uncertainties on a real image region. Green dots for each feature denote the noise computed by reprojecting features detected in other images and black ellipses denote the noise model for each feature.

The last decade has witnessed a wide body of literature in which the primary goal has been to build efficient and accurate solutions scalable to a large number of correspondences. However, the fundamental problem of mismatched correspondences (see Figure 1.3 Top) has not been directly handled by PnP solvers. These outlier correspondences are typically rejected in a preliminary stage using P3P solvers in combination with RANSAC-based schemes, reducing significantly the efficiency of the final solver. Moreover, usually the efficiency decrease linearly with respect to the number of correspondences. Therefore, a large number of correspondences also reduce significantly the efficiency of the solver.

In addition, although PnP solutions assume that correspondences may be corrupted by noise and show robustness against large amounts of it, none of them considers the particular structure of the noise associated to each correspondence (see Figure 1.3 Bottom), assuming that all the correspondences are affected by the same model of noise.

Contributions: Our contributions in this field are three real-time PnP solvers. We first reformulate the Efficient PnP solver [97] in order to increase its accuracy and efficiency providing almost constant computational cost for any number of correspondences, even thousands. We also propose an outlier rejection mechanism that is fully integrated in our pose

estimation pipeline with a negligible computational overhead. Finally, in order to further improve the accuracy of the estimated pose we model the particular structure of the noise for each correspondence and we reformulate the PnP problem as a maximum likelihood estimation.

1.4 Monocular Camera Pose Estimation in Complex Scenarios

In order to build valid 3D-to-2D correspondences, camera pose estimation algorithms need correctly matched features between 2D images and 3D models. When those 3D models contains hundreds of thousands of features is difficult and computationally expensive to be able to find correct 3D-to-2D correspondences.

In the literature this problem is handled by analysing the global appearance of the images. A global descriptor is computed for each input image and using machine learning approaches these methods estimate which image within a given training set is the closest one to each input image. An example of the results of these methods can be seen in Figure 1.4. The limitation of these approaches is that the camera pose they can estimate tends to be inaccurate, and highly depends on the spatial resolution at which the training images have been acquired. This limitations can be seen in Figure 1.4.

Contributions: Our contribution in this field is a new algorithm that combine the best of both worlds. We use a global descriptor in order to look for similar images in the training set providing an initial set of rough estimations of the camera pose. The initial estimations in its turn reduce the number of potential 3D-to-2D correspondences, and make standard camera pose estimation approaches applicable. In this stage, we take advantage of our very fast and robust to outliers PnP solver. The combinations of both ingredients result in a powerful pose estimation algorithm capable of dealing with very large models.

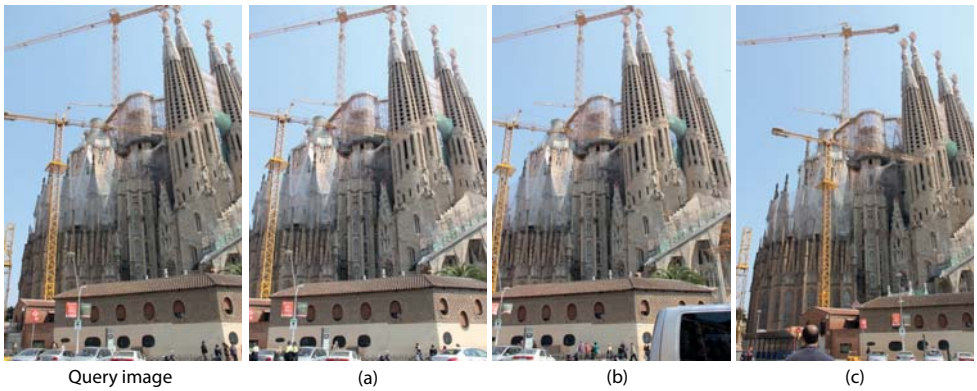


Figure 1.4: Example of the closest images in the training set to a given query image. Images (a), (b) and (c) denotes the three closest images in the training set.

1.5 Thesis Overview

This thesis is conceptually grouped into 4 chapters enclosing our contributions organized from low to high level. Each chapter has an introduction section in order to provide a high level overview of its state of the art, several sections explaining our contribution, one section showing our results in front of the state-of-the-art methods and a final summary. Finally, there is a last chapter pointing out the most important contributions of this thesis and some concluding remarks.

Chapter 2: Scale & Affine Invariant Detection of Non-Redundant Interest Points. In this chapter we introduce our four interest point detectors and their properties. We first give an introduction reviewing the state-of-the-art detectors and a short introduction to differential geometry applied to images. The following sections refer to our scale invariant detectors allowing to detect simultaneously corners and blobs avoiding redundancies and our affine invariant detector. Finally we summarize the results.

Chapter 3: Learning of Discriminative Local Image Descriptors. This chapter focusses on our work on learning of discriminative and robust descriptors. We first review the related work in terms of hand-crafted and machine learning based descriptors. Afterwards, we introduce Artifical Neural Networks and Convolutional Neural Networks which are the basis for our machine learning based descriptor, which is introduced in the next section. Finally, we explain and validate our method through experiments.

Chapter 4: Robust Monocular Camera Pose Estimation. In this chapter we introduce our three PnP methods to estimate the camera pose. We first review the main PnP methods in the state of the art and we introduce our fast and accurate PnP method. In subsequent sections we introduce our methods based on the previous one to deal with outlier correspondences and feature uncertainties. Each one of those sections validates the proposed methods through experiments on synthetic data and real images.

Chapter 5: Monocular Camera Pose Estimation in Complex Scenarios. This chapter focusses on our early work to be able to estimate as fast as possible the camera pose with high accuracy in complex 3D scenarios with even hundreds of thousand of features. We first review the related work and we explain the method used to build 3D models. Afterwards, we introduce our two steps methodology to estimate the camera pose. Finally, we validate our method on two real scenarios.

Chapter 6: Conclusions and Future Work. In the last chapter we give a high-level discussion of our contributions. We also sketch several directions for future research.

Chapter 2

Scale & Affine Invariant Detection of Non-Redundant Interest Points

Scale and affine invariant detectors are well suited for local feature extraction. State-of-the-art detectors usually tend to over-represent the local image structures associating several interest points for each local image structure. Therefore interest points are not completely distinguishable, losing the ability to clearly describe the uniqueness of each local image structure. In order to solve this issue we propose non-redundant detectors of blobs and corners, which try to describe each blob or corner image structure with a single interest point. In our approach, scale invariant interest points are located by means of the idea of blob/corner movement and blob/corner evolution (creation, annihilation and merging) along different scales by using an accurate description of the image provided by the Gaussian curvature, providing a global bottom-up estimation of the image structure. The Gaussian curvature allows to classify image regions in terms of their curvature as blobs, corners or edges.

Another issue with state-of-the-art detectors is that they are usually able to detect one single type of image structure, typically corners or blobs. Our approach allows to detect simultaneously both, corners and blobs, with negligible overhead.

In addition, we propose to refine the shape and location of scale invariant blobs by fitting an anisotropic Gaussian function, which minimizes the error with respect to the underlying image and simultaneously estimates both the shape and location, by means of a non-linear least squares approach.

A comparative evaluation of scale and affine invariant detectors is presented, showing a comparable performance to the state-of-the-art detectors in terms of repeatability and matching scores, while in terms of precision and recall our detectors outperform or obtain similar scores. In addition we demonstrate that our detectors do not over-represent blob and corner structures providing a non-redundant detection that improves distinctiveness and reduces the computational cost of future matching tasks. In order to verify the accuracy and computational cost reduction we have evaluated our detector in image registration tasks.

2.1 Introduction

This chapter is dedicated to detect scale and affine invariant interest points. Interest point detectors have been shown to be well suited for local feature extraction and they are the cornerstone of a wide range of Computer Vision applications as object recognition [84, 135], image registration [91], 3D reconstruction [1] or visual tracking [151]. The aim of interest points is the detection of local image structures, which can reliably be found in other images containing the same structures. Each one of these local image structures is typically determined by an image region in a concrete location with a specific scale and shape.

The intrinsic geometry of the local image structures allows to classify interest point detectors into two main groups. Scale invariant detectors, which describe each local image structure as a pixel location and a scale parameter, assume that local image structures can be determined by an isotropic image region. On the other hand, affine invariant detectors take into account that the shape of local image structures can be anisotropic. Usually, affine invariance is a post-processing step, which is applied on an initial set of interest points detected by a scale invariant detector [82, 91, 54, 70]. Therefore, the initial set of scale invariant interest points is crucial.

Besides of geometrical invariance, there are other important properties that interest points should satisfy. In [44, 112, 33] a set of properties that interest points should satisfy to be considered as optimal are proposed. From our point of view, besides geometrical invariance there are four main properties:

- **Robustness to image variations:** in order to be able to detect specific local image structures with accuracy even in presence of perturbations, such us noise, blurring or illumination changes.
- **Completeness:** to be able to detect at the same time a complete set of complementary types of local image structures such us, corners, edges or blobs. Usually, several detectors must be applied to fulfill this property.
- **Distinctiveness:** every interest region should clearly stand out in its neighbourhood being as distinctive as possible, avoiding redundant regions.
- **Uniqueness:** interest points should determine every local image structure with accuracy and uniqueness. Conferring an easy interpretation to each local image structure, which improves the distinctiveness between interest points.

Another interesting property, but this time for interest point detectors is that they should have as few control parameters as possible and they should have a clear semantic [33]. Finally, depending on the application there are properties more relevant than others. For example, distinctiveness property can be very useful for visual tracking [87] whereas for object recognition its lack can be also considered a feature [135]. However, [135] is a specific case and cannot be generalized.

The success of all these properties depends on the approach and the operators used in order to obtain the interest points. This chapter proposes an automatic scale selection approach

for interest points, taking advantage of their evolution over scale, and a set of accurate differential geometry operators to satisfy the explained properties in our scale and affine invariant interest points.

2.1.1 Overview

This chapter is organized as follows. In the next two sections scale and affine invariant detectors are reviewed. After that, in Section 2.4 a short introduction to differential geometry on images is done, relating the first and second order terms of the Taylor expansion with the curvature analysis. In Section 2.5 we present our scale invariant detectors of blobs and corners, which are based on Gaussian curvature. Moreover, in Section 2.5 we show how our scale invariant detectors inherit and enhance the ideas proposed by Lindeberg in [79] in order to extract with robustness the trajectories followed by blob and corner structures over scale. In Section 2.6 we present our shape adaptation algorithm based on Gaussian curvature and a non-linear least squares approach to fit Gaussian functions. Finally, in Section 2.7 an experimental evaluation is done. A comparative evaluation of our detectors with the state of the art is done under different imaging conditions in terms of geometrical invariance and distinctiveness in four experimental setups: repeatability and matching, precision and recall, over-representation and homography estimation.

2.2 Scale Invariance

The aim of interest points is the detection of local image structures, which can reliably be found in other images containing the same structures. Therefore, interest points not only have to locate the local image structures but also determine their characteristic scales. Thus, they are defined by three parameters, two for pixel coordinates and one for scale. These three parameters are estimated by using an exhaustive analysis, where scale parameter is split into a discrete set. This kind of analysis is commonly referred to as scale-space or multi-scale image analysis [17], where a multi-scale image representation is built in order to apply an automatic scale selection technique.

2.2.1 Multi-Scale Representation

The multi-scale image representation and its properties for interest point detection has been deeply studied in literature [17, 143, 63, 79]. This representation arises from the fact that relevant image structures are meaningful entities only over a restricted range of scales. Pixels in an image conform a discrete representation, similarly scale parameter can be also dicretized. Thus, the multi-scale representation is defined as a discrete set of scaled images. These scaled images must prevent the creation of spurious image structures when the scale is diminished, in this sense the work of Koenderink in [63] shows the set of scale-space levels must be governed by diffusion equation and each scale-space level represented by an smoothed version of the original image using Gaussian kernels.

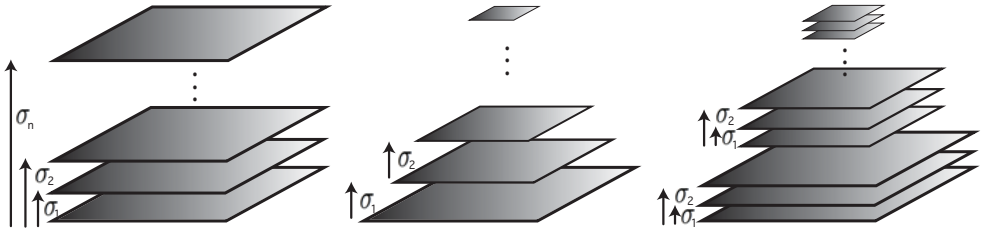


Figure 2.1: Different multi-scale representations. Succesive smothing (left), pyramid combining smoothing and sampling (center) and hybrid approach where sampling is performed after several smoothed scales (right). In all these representations the smooth is computed convolving by a Gaussian kernel with standard deviation σ.

Understanding an image as a two dimensional signal $\mathbf{I}(x, y)$ where x and y are the pixel coordinates, each level in the multi-scale representation $\mathbf{L}$ is built as,

$$\mathbf{L}(x, y, \sigma) = \mathbf{G}(\sigma) * \mathbf{I}(x, y) \tag{2.1}$$

where $*$ denotes convolution and $\mathbf{G}(\sigma)$ a Gaussian kernel with standard deviation σ,

$$\mathbf{G}(\sigma) = \frac{1}{2\pi\sigma^2} exp^{\frac{u^2+v^2}{2\sigma^2}} \tag{2.2}$$

where u and v represent kernel coordinates with respect to the kernel center.

In order to reduce the computational cost of building the multi-scale representation a pyramid based approach was proposed in [17], where the resolution of each scale-space level $\mathbf{L}(\cdot, \cdot, \sigma)$ is reduced after the smoothing. The drawbacks of pyramid representation are mainly two: it can produce aliasing and it increases the cost of relate the pixels coordinates along the scales. Lowe in [84] proposes a pyramid representation able to deal with these drawbacks by not reducing the resolution in all the scale-space levels. In Figure 2.2.1 all the enumerated multi-scale representations are shown.

These multi-scale representations require to pre-select the smoothing and sampling parameters. Lindeberg in [80] shows that the ratio between successive scales must be constant, allowing to maintain a uniform change of information between successive scale-space levels. Concretely, for the representation based on successive smoothing without sampling where the scales are parametrized by σ, equation (2.1), a set of n scales is defined as,

$$\sigma_k = r^k \sigma_0, \qquad 1 \leq k \leq n \tag{2.3}$$

where σ_0 represents the scale of the initial scale-space level and r is the ratio between successive scales. For example a commonly used ratio providing a good trade off between interest points localization accuracy and computational cost is 1.4 [80, 84, 91]. Lower ratios

allow to increase the localization accuracy at the expense of scale-space levels, which grows quickly.

Gaussian multi-scale representation is useful to detect specific kinds of low-level local image structures such as corners, edges, blobs or even spirals [6]. Those structures are typically found by building specific operators, which combine first and second order partial derivatives of smoothed images along the multi-scale representation. Those partial derivatives for a specific scale-space level can be computed taking into account the convolution properties,

$$\frac{\delta}{\delta_{x^\alpha y^\beta}}\mathbf{L}(x,y,\sigma) = \frac{\delta}{\delta_{x^\alpha y^\beta}}(\mathbf{G}(\sigma) * \mathbf{I}(x,y)) = \frac{\delta}{\delta_{x^\alpha y^\beta}}\mathbf{G}(\sigma) * \mathbf{I}(x,y) \tag{2.4}$$

where α and β denotes the order of the partial derivatives. This equation shows that partial derivatives can be computed directly on the Gaussian kernel.

Image structures in a given image are meaningful entities only over a restricted range of scales. That means operators used to detect those structures along the scales must provide a mechanism to locate their characteristic or relevant scales. In addition, those characteristics scales must be also detected independently of the image scale. To be able to compare different images and locate the same characteristic scales the aim is to obtain constant partial derivative responses between an image $\mathbf{I}$ and its scaled version $\mathbf{I}'$, with a known scaling factor s such that $\mathbf{I}(x,y) = \mathbf{I}'(sx,sy)$. In order to get this property the Gaussian partial derivatives must be normalized with respect to the scale factor [80]. Concretely, for the multi-scale representation based on smoothing without sampling, where the amplitude of partial derivatives increases according as the scale increases, the normalization parameter σ^γ allows to validate that

$$\begin{aligned} \sigma_k^{\alpha+\beta}\frac{\delta}{\delta_{x^\alpha y^\beta}}\mathbf{L}'(x',y',\sigma_k) &= \sigma_l^{\alpha+\beta}\frac{\delta}{\delta_{x^\alpha y^\beta}}\mathbf{L}(x,y,\sigma_l) \\ \sigma_k^{\alpha+\beta}\frac{\delta}{\delta_{x^\alpha y^\beta}}\mathbf{G}(\sigma_k) * \mathbf{I}'(x',y') &= \sigma_l^{\alpha+\beta}\frac{\delta}{\delta_{x^\alpha y^\beta}}\mathbf{G}(\sigma_l) * \mathbf{I}(x,y) \end{aligned} \tag{2.5}$$

where k and l denotes two different scales. From a more informal point of view, the normalization parameter σ^γ achieves that the integral of the positive part of the γ-order Gaussian partial derivatives becomes constant.

Gaussian multi-scale representations have been also related with the biological vision system. Neurophysiological studies show that receptive field profiles in the mammalian retina and the first stages in the visual cortex can be modelled by superposing Gaussian derivatives [144, 81]. In these respects, the scale-space framework can be seen as a theoretically well-founded paradigm for early vision.

2.2.2 Operators to Detect Image Structures

Scale invariant interest point detectors are more or less optimal depending on the operators used in order to build the multi-scale representation. These operators applied along the scales

should be responsible of the success of the properties explained in the previous section: robustness, completeness, distinctiveness and uniqueness. Basically, in order to find local image structures, there are two main kinds of operators:

Geometry-based operators. These operators are specifically designed to search for specific kinds of low-level local image structures, being the most studied corners and blobs. Since these operators typically combine several first and second order Gaussian partial derivatives to be computed, they can be directly integrated inside the multi-scale representation. Corner detection methods can be classified in two main approaches [94]: contour-based and intensity-based. The contour-based approach is based on an initial contour detection (usually by means of the Canny detector) and a posterior search of curvature maxima points along contours, e.g. [95, 94]. In contrast, the intensity-based approach directly computes a measure on the grayscale image, e.g. [141, 80, 150]. On the other hand, methods for blob detection follow the intensity-based approach by computing measures directly on the grayscale image, e.g. [80, 91, 84, 92].

Unfortunately sometimes is not enough with a single operator to detect interest points. Some geometry-based operators have shown to be better for spatial location and other ones for scale selection. For example in the Harris-Laplace [91] or the Hessian-Laplace [90] detectors spatial selection is done using the Harris and the determinant of Hessian operators respectively. Once extreme responses for each scale are selected the characteristic scales are evaluated using the Laplace operator, which has proof its ability to carry out this task [80, 84].

Appearance-based operators. These operators are specifically designed in order to search for distinctive image regions. Therefore it is not mandatory to relate image regions with specific kinds of low-level local image structures such as corners, edges or blobs. Interest point detectors based on these operators look for image regions with a distinctive pixel distribution in front of their neighbourhood, commonly by means of saliency measures over histograms [53, 29, 70]. This kind of measures can be also computed on a multi-scale representation, however its properties to relate responses over scale are lost since saliency measures cannot be directly related with specific kinds of low-level image structures.

An interesting approach combining geometry and appearance based operators is proposed in [26]. In this approach, while the geometry-based operator is used to decide the candidate image regions, the appearance-based operator is used to evaluate the stability of all the candidates along several scales. Thus, the appearance-based operator is not used to evaluate the distinctiveness but rather the stability. This detector tries to locate stable descriptors for future matching tasks.

2.2.3 Evolution and Automatic Scale Selection of Image Structures

In a multi-scale representation scheme according as the scale increases the fine structures vanish and high level structures show up. Therefore, scale invariance must be achieved by searching for, across several possible scales, extreme responses provided by operators combining scale normalized Gaussian partial derivatives. This fact was deeply studied by Lindeberg in [78, 79, 80] providing a general framework for automatic scale selection. Lindeberg shows how a local extremum over scale at a concrete scale also appears after a uniform rescal-

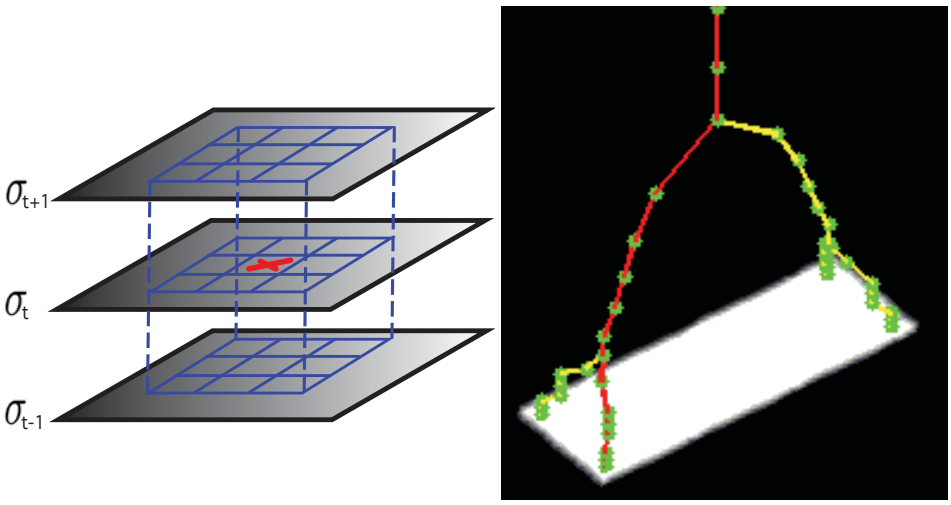


Figure 2.2: Automatic scale selection looking for local extreme responses. The response of the pixel X is compared with its 26 neighbors in 3 $\times$ 3 regions at the current and adjacent scales [84] (left). Example of evolution along the scales of a concrete blob structure created in a corner structure (red line) on a synthetic image using our approach (right). Green dots denotes extreme responses along the scales.

ing at an equivalent scale. He shows that extreme responses are characterized by the first and second order normalized derivatives with respect to the scale and space simultaneously. Lowe in [84] proposes the most commonly used automatic scale selection approach, which basically looks for extreme responses of spatial partial derivatives and approximates the scale second order partial derivatives by comparing spatial responses in a 3D neighbourhood (usually 3x3x3). This neighbourhood combines the spatial responses of a concrete scale with the spatial responses in the upper and lower scales (see Figure 2.2(left)).

An important property of geometry-based operators, which is not shared with appearance-based operators, is that they allow us to analyse the evolution of specific local image structures from their creation to their annihilation along the scales. This property provides the ability to select the location and scale maximizing the distinctiveness and uniqueness of each local image structure, in Figure 2.2(right) an example of evolution of a blob structure can be seen. However, most state-of-the-art detectors based on these operators do not try to take advantage of this property, at least totally. In fact standard automatic scale selection approaches, such us [84], can be considered as very local approximations of the evolution between consecutive scales.

From the analysis of the evolution of each local image structure emerge two issues. First, interest points should be extracted from the evolution of each local image structure and, second, local image structures at each scale should be represented by only one interest point.

There are two main drawbacks to represent each local image structure by only one interest point at each scale. The first one is the well-known problem of kernel discretization [77] and the second one is related with the ability of operators to represent each kind of im-

age structure. In general, state-of-the-art geometry-based operators look for low-level image structures by estimating second order partial derivatives, whether it is through Laplacian, as in [80], or through Hessian approaches, as in [90].

These second order approaches can be understood as approximations of the Mean curvature and the Gaussian curvature operators [25], which combine first order and second order partial derivatives. Both curvatures have shown their success in several corner detectors, being used as one dimensional operators on image contours [95, 94] or being used as two dimensional operators on grayscale images [141, 4, 23]. The Gaussian curvature operator allows us to understand images as surfaces with three types of regions, corresponding each one with specific low-level image structures. Concretely, blob structures are related with elliptic regions, contour structures or flat regions are related with parabolic regions, and finally, corner and saddle structures are related with hyperbolic regions (see Figure 2.5 for some examples).

In order to compute the evolution of each local image structure it is needed to link its extreme responses, provided by an scale normalized operator, along the scales belonging to the multi-scale representation. The first method to link the evolution of the kernel responses was proposed by Witkin in [143] for one dimensional signals. However, on two dimensional signals like images, linking the extreme responses belonging to an specific image structure along the scales is a complex task [79].

The evolution of local image structures was deeply studied by Lindeberg in [79] in the case of blob structures. Lindeberg proposed 4 events: blob annihilation, blob merging, blob split and blob creation. Lindeberg proposed a watershed-based blob detector, where the flooding stops when the intensity value of pixels fall bellow a certain value obtained from the saddle region between two blob structures. Finally, Lindeberg's method concludes with a matching mechanism between blobs at consecutive scales based on spatial coincidence. This approach tries to carry out an impossible task on complex images. It attempts to obtain a precise and detailed evolution of all the blobs on the image facing the problem with non-robust algorithms. In fact, the flooding stop criterium makes blob regions highly unstable, as short variations in the stopping value can produce large variations in the blob region. Moreover the matching algorithm presents difficulties when there are more than one matching candidates and matching is non-trivial when there are more than two candidates [79]. Therefore, Lindeberg's approach can hardly extract stable blobs and stable trajectories unless its robustness is improved.

In the case of the evolution of corner structures, Fidrich and Thirion in [31] propose an algorithm to extract the evolution of each corner structure by using iso-contours, obtaining one iso-contour for each scale space level. They propose to use a 'Marching Lines' algorithm in order to relate curvature extrema and inflection points of the planar curves represented by the iso-contours. From the set of related curvature extrema and inflection points they analyse the corner stability on medical images measuring the lifetime and maximum response of each corner. However this method is only valid for corner detection because it only measures the curvature in the contours of the objects.

Commonly used state-of-the-art detectors do not estimate the image structures evolution. For this reason the automatic scale selection method tend to assign several characteristic scales, and hence several interest points to each image structure carrying out an over-detection

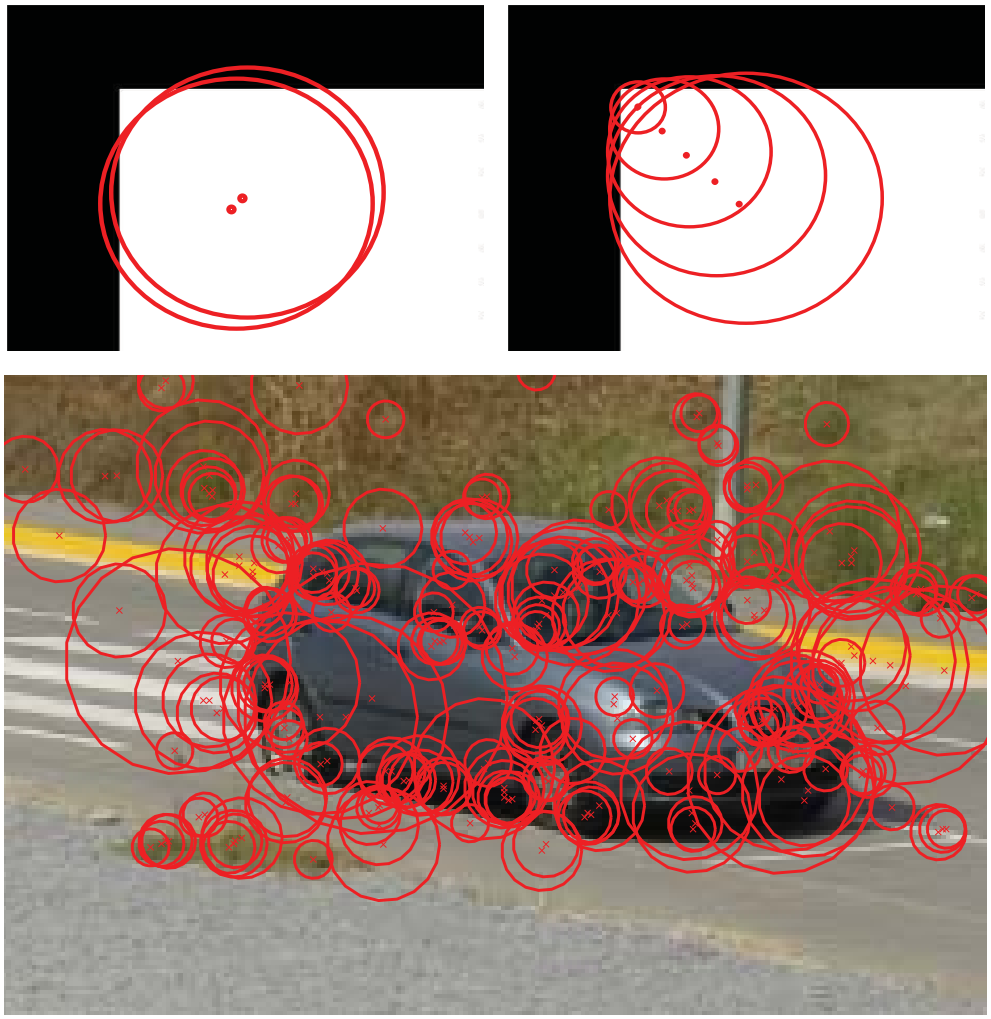


Figure 2.3: Examples of over-detection where interest points are represented by red circles. In the first row, over-detection on a synthetic image containing a corner structure is shown. Left image shows over-detection on a specific scale and right image along different scales. In the bottom image, over-detection using a standard blob detector, Hessian-Laplace [90] detector, on a real image is presented.

(Figure 2.3). This over-detection, based on redundant interest points, increases the computational complexity (which is superlinear in the number of interest points) of future matching tasks [10, 136]. In addition, the redundancy works against the distinctiveness and uniqueness properties.

Specifically, Brown *et al.* in [10] partially address this problem by proposing a post processing step, which filters the interest points in order to restrict their number (fixing the final number) so that they are spatially well distributed, using an adaptive non-maximal sup-

pression strategy. Their strategy uses an adaptive radius of the non-maximal suppression in order to avoid nearby corner detections (using a multi-scale Harris detector). However, their method does not take into account whether nearby corners are the same and only takes into account their strength.

In contrast, we propose detectors which analyse the evolution of blobs and corners with robustness, selecting the best interest point for each local image structure. Our approach is a robust bottom-up algorithm to analyse the image structures by using the accurate Gaussian curvature operator in combination with a gradient ascent scheme. In addition, our approach allows to analyse the evolution of both, blobs and corners, at the same time.

2.3 Affine Invariance

Affine invariant detectors not only looks for the spatial location and characteristic scale of interest points, they also estimate the underlying affine shape of local image structures. Multi-scale image analysis has shown to be successful in terms of detection of scale invariant interest points. This approach can also be used for affine invariant detection but, since the number of parameters is increased to five (spatial location, two scales and orientation), the complexity grows exponentially and the approach is no longer feasible. For this reason several affine invariant detectors have been proposed using a scale invariant detector as starting point.

The best known approach to detect affine invariances is the shape adaptation algorithm proposed by Mikolajczyk and Schmid in [91], which is generic for geometry-based operators. For appearance-based operators there are several approaches such as [54] or [70]. Nevertheless, none of them is generic since they depend on very specific scale invariant detectors. Another family of affine detectors is based on the watershed algorithm, showing very interesting results in terms of geometrical invariance and distinctiveness. Being their best example the Maximally Stable Extremal Regions (MSER) detector [88], which analyses the shape variations at different pixel intensity levels to locate stable and connected image regions.

A different approach is proposed by Morel and Yu in [96], they propose a novel framework for interest point detection based on the simulation of specific affine deformations on images in order to compute scale invariant interest points on each simulated image. Specifically, they propose to compute the SIFT detector [84] for each set of simulated images in order to extract a huge number of scale invariant interest points. Assuming that these interest points codify all the feasible deformations of the image, the affine invariance problem is converted into a matching problem. Thus, the computational cost of these algorithm is large in comparison with other detectors as [91, 88], although it has shown its good performance for object detection and image registration [96].

2.3.1 Affine Invariance from Geometry-based Operators

Mikolajczyk and Schmid in [91] propose a shape adaptation algorithm based on a previous work of Lindeberg and Gårding [82]. This algorithm estimates affine transformations in

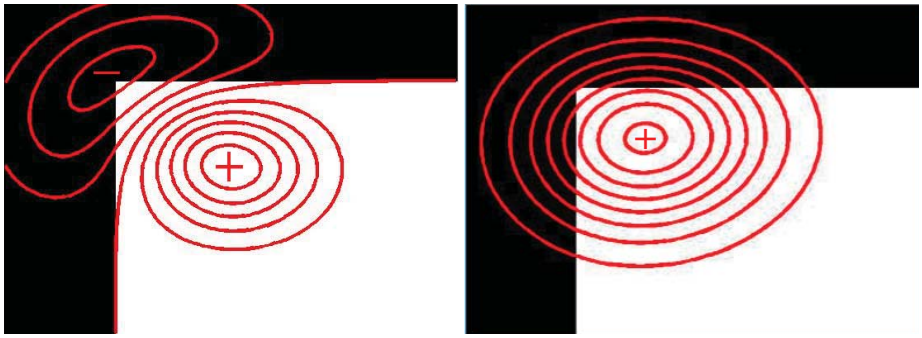


Figure 2.4: Comparison between the surfaces provided by the Gaussian curvature operator (left image) and the determinant of the second moment matrix (right image). Operators are evaluated on a synthetic image containing a corner structure. Being the red lines the level curves of the surfaces, the + simbol indicates the positive region of the surface and the - symbol indicates the negative region.

two steps, which are iteratively applied, the first one estimates the spatial location while the second one the affine deformation.

The very first step of these detectors consists in extracting a set of scale invariant interest points using a geometry-based scale invariant detector. Afterwards, the affine shape for each scale invariant interest point is vaguely estimated from the eigens of the second moment matrix, which can be considered as a rough approximation to the Hessian matrix. The most significant difference lies in that the Hessian matrix is either positive or negative definite while the second moment matrix is semi-definite positive. This can be seen in Figure 2.4, which compares the responses of the second moment matrix and Gaussian curvature, which as we explain in Section 2.4.6 is quite similar to the Hessian matrix response. In [67] is shown that substituting the second moment matrix by the Hessian matrix in the affine shape estimation algorithm the obtained results are similar.

Once the affine shape of an interest point is estimated, its spatial location is re-estimated applying again the previously used scale invariant detector on the transformed image domain (where the affine shape is undone). These two steps are iteratively computed until the eigenvalues of the second moment matrix show isotropy.

The second moment matrix is defined as,

$$\mu = \begin{pmatrix} \mu_{11} & \mu_{12} \\ \mu_{21} & \mu_{22} \end{pmatrix} = \mathbf{G}(\sigma) * \begin{pmatrix} \mathbf{L}_x^2 & \mathbf{L}_x\mathbf{L}_y \\ \mathbf{L}_x\mathbf{L}_y & \mathbf{L}_y^2 \end{pmatrix} \tag{2.6}$$

where $\mathbf{G}(\sigma)$ denotes an averaging operator, usually a Gaussian kernel, with an integration scale parameter σ. $\mathbf{L}_x$ and $\mathbf{L}_y$ denote the compact representation of equation (2.4) for the first-order partial derivative in x and y.

The second moment matrix, also referred to as structure tensor, summarizes the distribution of the grayscale gradients for a given point in terms of two principal directions and two non-negative magnitudes. Which can be considered an approximation of the directions

and the absolute values of the principal curvatures referred to the underlying shape. In [91], Mikolajczyk and Schmid assume that these principal directions and the ratio between the two rough principal curvatures can be used as a change of basis in order to apply affine transformations up to the underlying shape becomes isotropic. That means, the ratio is linearly used in order to estimate the size of an image region with the same curvature (contrast) in the two principal directions.

Both principal curvatures, from their definition [25], explain the shape in their principal directions by means of their sign and scalar values, however the second moment matrix loses partially this information since it is a semi-definite positive matrix. The product of the principal curvatures provides an important differential geometry invariant referred to as Gaussian curvature. Hence, depending on the sign and the value of the Gaussian curvature, the shape and the shape strength of each image region is estimated. Thereby, Gaussian curvature provides a fundamental information about the shape, which is not provided by the second moment matrix (see Figure 2.4).

From the shape information provided by Gaussian curvature, we propose a novel method to estimate affine invariance fitting known parametric functions on local surfaces. Specifically, in the case of blob structures we propose to fit Gaussian functions by means of a non-linear least squares approach. This approach allows us to take into account the pixel distribution on each image region in contrast with the second moment matrix, which only uses a very rough approximation of the absolute value of the principal curvatures. A very important property of our approach is that it allows us to simultaneously estimate the location and the shape of each blob structure. This avoids having to separate the algorithm into two steps, one of them exhaustive.

2.4 Curvature Analysis on Grayscale Images

The image behaviour in a local neighbourhood of a point can be approximated by the Taylor expansion, so the local neighbourhood can be simplified as an expansion of functions. The Taylor expansion of a local neighbourhood of a point $\mathbf{x}_0$ is,

$$\begin{aligned} \mathbf{I}(\mathbf{x}_0 + \mathbf{h}) &= \mathbf{I}(\mathbf{x}_0) + \nabla\mathbf{I}(\mathbf{x}_0)\mathbf{h} + \\ &+ \frac{1}{2!}\mathbf{h}^T\nabla^2\mathbf{I}(\mathbf{x}_0)\mathbf{h} + \mathbf{R}(\mathbf{h}) \end{aligned} \tag{2.7}$$

where $\mathbf{R}(\mathbf{h})$ is a residual.

Differential geometry, in general, allows to extract properties and features of points by means of first and second order terms of the Taylor expansion. While first order terms contain information about gradient distribution in a local neighbourhood, second order terms contain information about the local shape.

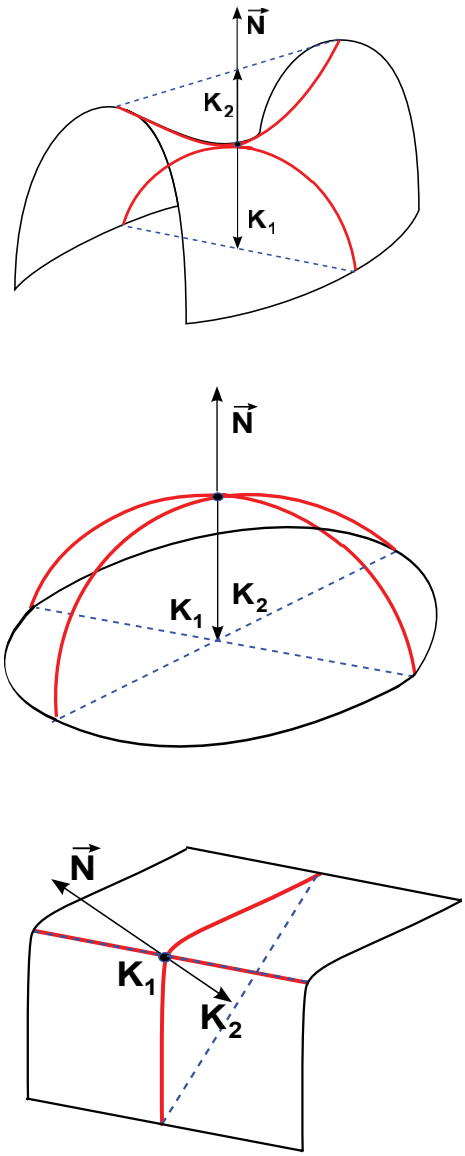


Figure 2.5: Gaussian curvature K for three different kinds of surfaces where $\mathbf{N}$ is the surface normal and k_1 and k_2 are the two principal curvatures: Saddle surface with $K < 0$ ($k_1 < 0$ and $k_2 > 0$), blob surface with $K > 0$ ($k_1 < 0$ and $k_2 < 0$) and edge surface with $K = 0$ ($k_1 = 0$ and $k_2 < 0$). Being the red lines the projections of the two main directions on the surface.

2.4.1 Principal Curvatures

A point $\mathbf{p}$ on an image $\mathbf{I}$ can be expressed in a parametric form as $\mathbf{p} = (x, y, \mathbf{I}(x, y))$ where x and y are the image coordinates. Assuming that images can be understood as differentiable functions [64], by blurring $\mathbf{I}$ with a kernel $\mathbf{G}$, involve that images are 3D surfaces which are treated locally as an specific type of patch (local surface) called *Monge patch* [64].

For each point $\mathbf{p}$ there is a tangent plane defined by a normal vector $\mathbf{N}$. Moreover, for each point $\mathbf{p}$ there are an infinite number of 1D curves passing through it, understanding each 1D curve as the intersection between a plane and the surface. From each 1D curve on a cutting plane and the point $\mathbf{p}$ a tangent vector on the plane can be computed, being its perpendicular vector on the plane the curvature vector [25]. The projection of this curvature vector on the normal vector $\mathbf{N}$ to $\mathbf{p}$ is an scalar called normal curvature. From the infinite number of normal curvatures to $\mathbf{p}$ there exist two relevant 1D curves denoting the curve with largest absolute normal curvature and its perpendicular curve. The normal curvatures of these two 1D curves are the principal curvatures k_1 and k_2 for the point $\mathbf{p}$ on the surface. In Figure 2.5 are shown some surfaces with their principal curvatures and the 1D curves related with them.

In order to compute k_1 and k_2 for a local image region is not necessary to compute all the feasible normal curvatures, both can be computed from the first and second fundamental forms. These fundamental forms are estimated computing the first and second order partial derivatives in the Taylor expansion showed in equation (2.7). Those partial derivatives are computed as in [64] using a straightforward method, convolving an image $\mathbf{I}$ with a differentiable kernel $\mathbf{G}$.

The principal curvatures k_1 and k_2 determine two important differential invariants, the Gaussian curvature K and the Mean curvature M.

2.4.2 First Fundamental Form

Given a point $\mathbf{p}$, the first order terms of the Taylor expansion conform its so-called Jacobian matrix,

$$\mathbf{J} = \begin{pmatrix} \frac{\partial \mathbf{p}}{\partial x} \\ \frac{\partial \mathbf{p}}{\partial y} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \mathbf{L}_x \\ 0 & 1 & \mathbf{L}_y \end{pmatrix} \tag{2.8}$$

where first-order partial derivatives $\mathbf{L}_x$ and $\mathbf{L}_y$ are estimated applying Gaussian derivative kernels on the image as in equation (2.4).

The first fundamental form $\mathbf{F}_I$ for the point $\mathbf{p}$ on an image $\mathbf{I}$ is defined as,

$$\mathbf{F}_I = \mathbf{J} * \mathbf{J}^T = \begin{pmatrix} 1 + \mathbf{I}_x^2 & \mathbf{I}_x \mathbf{I}_y \\ \mathbf{I}_y \mathbf{I}_x & 1 + \mathbf{I}_y^2 \end{pmatrix} \tag{2.9}$$

2.4.3 Second Fundamental Form

The normal vector $\mathbf{N}$ to the point $\mathbf{p}$ on an image $\mathbf{I}$ is defined as,

$$\mathbf{N} = \frac{\frac{\partial \mathbf{p}}{\partial x} \times \frac{\partial \mathbf{p}}{\partial y}}{\left| \frac{\partial \mathbf{p}}{\partial x} \times \frac{\partial \mathbf{p}}{\partial y} \right|} = \frac{1}{\sqrt{1 + \mathbf{L}_x^2 + \mathbf{L}_y^2}} \begin{pmatrix} -\mathbf{L}_x \\ -\mathbf{L}_y \\ 1 \end{pmatrix} \tag{2.10}$$

The second order terms of the Taylor expansion conform the Hessian matrix $\mathbf{H}$, which is similar to the second fundamental form $\mathbf{F}_{II}$. The general equation of $\mathbf{F}_{II}$, shown below, is derived in [25] from the normal curvatures and it is calculated using the second order partial derivatives and the normal vector $\mathbf{N}$ for each point $\mathbf{p}$. Specifically, the second fundamental form $\mathbf{F}_{II}$ for images is defined as,

$$\begin{aligned} \mathbf{F}_{II} &= \begin{pmatrix} \frac{\partial^2 \mathbf{p}}{\partial x^2}\mathbf{N} & \frac{\partial^2 \mathbf{p}}{\partial x \partial y}\mathbf{N} \\ \frac{\partial^2 \mathbf{p}}{\partial x \partial y}\mathbf{N} & \frac{\partial^2 \mathbf{p}}{\partial^2 y}\mathbf{N} \end{pmatrix} = \\ &= \begin{pmatrix} \begin{pmatrix} 0 & 0 & \mathbf{L}_{xx} \end{pmatrix} \cdot \mathbf{N} & \begin{pmatrix} 0 & 0 & \mathbf{L}_{xy} \end{pmatrix} \cdot \mathbf{N} \\ \begin{pmatrix} 0 & 0 & \mathbf{L}_{xy} \end{pmatrix} \cdot \mathbf{N} & \begin{pmatrix} 0 & 0 & \mathbf{L}_{yy} \end{pmatrix} \cdot \mathbf{N} \end{pmatrix} = \\ &= \frac{1}{\sqrt{1 + \mathbf{L}_x^2 + \mathbf{L}_y^2}} \begin{pmatrix} \mathbf{L}_{xx} & \mathbf{L}_{xy} \\ \mathbf{L}_{yx} & \mathbf{L}_{yy} \end{pmatrix} = \frac{1}{\sqrt{det(\mathbf{F}_I)}} \mathbf{H} \end{aligned} \tag{2.11}$$

where second-order partial derivatives $\mathbf{L}_{xx}$, $\mathbf{L}_{xy}$ and $\mathbf{L}_{yy}$ are estimated applying Gaussian kernels on the image $\mathbf{I}$ as in equation (2.4) and $det(\mathbf{F}_I)$ is the determinant of the first fundamental form.

2.4.4 Principal Curvatures Estimation

The relations between $\mathbf{F}_I$ and $\mathbf{F}_{II}$ can be seen through the Weingarten equations [25], which provide the partial derivatives $\mathbf{N}_u$ and $\mathbf{N}_v$ of the normal vector $\mathbf{N}$ of $\mathbf{p}$ with respect to all the normal vectors in the neighborhood of $\mathbf{p}$. Weingarten equations relate the factors of $\mathbf{F}_I = \begin{pmatrix} E & F \\ F & G \end{pmatrix}$ and $\mathbf{F}_{II} = \begin{pmatrix} e & f \\ f & g \end{pmatrix}$ in order to estimate $\mathbf{N}_u$ and $\mathbf{N}_v$ on the subspace defined by the tangent plane to $\mathbf{N}$,

$$\nabla \mathbf{N} = \begin{pmatrix} \mathbf{N}_u & \mathbf{N}_v \end{pmatrix} = \frac{1}{EG - F^2} \begin{pmatrix} fF - eG & eF - fE \\ gF - fG & fF - gE \end{pmatrix} \tag{2.12}$$

The eigenvalues of $\nabla \mathbf{N}$ are $-k_1$ and $-k_2$ for a local image region, centered on $\mathbf{p}$. By convention the principal curvatures are the opposite to the eigenvalues of $\nabla \mathbf{N}$ [25], therefore the principal curvatures are k_1 and k_2 .

2.4.5 Gaussian and Mean Curvature

Gaussian curvature K and Mean curvature M are two differential invariants computed from the two principal curvatures k_1 and k_2. Concretely the Gaussian curvature is defined as $K = k_1 k_2$ and can be written in terms of $\nabla \mathbf{N}$ as,

$$K = k_1 k_2 = det(\nabla \mathbf{N}) = \frac{eg - f^2}{EG - F^2} = \frac{\mathbf{L}_{xx}\mathbf{L}_{yy} - \mathbf{L}_{xy}^2}{(1 + \mathbf{L}_x^2 + \mathbf{L}_y^2)^2} \tag{2.13}$$

similarly the mean curvature M can be also written in terms of $\nabla \mathbf{N}$ as,

$$\begin{aligned} M &= \frac{1}{2}(k_1 + k_2) = \frac{1}{2} trace(\nabla \mathbf{N}) = \frac{1}{2}\frac{eG + gE - 2fF}{EG - F^2} = \\ &= \frac{\mathbf{L}_{xx}(1 + \mathbf{L}_y^2) + \mathbf{L}_{yy}(1 + \mathbf{L}_x^2) - 2\mathbf{L}_{xy}\mathbf{L}_x\mathbf{L}_y}{2(1 + \mathbf{L}_x^2 + \mathbf{L}_y^2)^{3/2}} \end{aligned} \tag{2.14}$$

The sign of K and M for a given point $\mathbf{p}$ determine the shape of the surface in its neighbourhood [64]. Although Mean curvature represents an important shape measure, it is in itself not particularly useful. It is important combined with Gaussian curvature in order to perform a more detailed shape classification as is shown in Table 2.1.

	$K < 0$	$K = 0$	$K > 0$
$M > 0$	saddle	contour	dark blob
$M = 0$	or	plane	—
$M < 0$	corner	contour	bright blob

Table 2.1: Curvature shape classification. Combining Gaussian curvature K and Mean curvature M the local shape of an image region is described.

In terms of K, the shape of a given point $\mathbf{p}$ is classified in 3 kinds of local surfaces topologies. When $K > 0$ the surface is locally concave or convex (i.e. blob regions) and the point is called elliptic, while for $K < 0$ the surface is saddle shaped (i.e. corner and saddle regions) and the point is called hyperbolic. Finally, the points with $K = 0$ (i.e. contours or plane regions) are called parabolic. In Figure 2.5 is shown that K is defined positive if its principal curvatures have the same sign, negative if they have different sign and zero if any principal curvature is zero. Finally, the scalar K, apart from giving information about the kind of region, offers also information about the shape strength, which can be understood as saliency.

An special case we think must be mentioned is the behaviour of the Gaussian curvature in a corner structure. In Figure 2.6 is shown the evolution of K from negative values to positive ones, that means the three surface topologies are closely related in this kind of structure. This behaviour was deeply studied in [23, 141, 41], in fact [27] proposes a method to detect the parabolic point located between the elliptical and the hyperbolic extreme since its location is more stable. An interesting study about the stability of hyperbolic points was done in [41] showing that depending on the corner angle the hyperbolic extreme can be inside or outside of

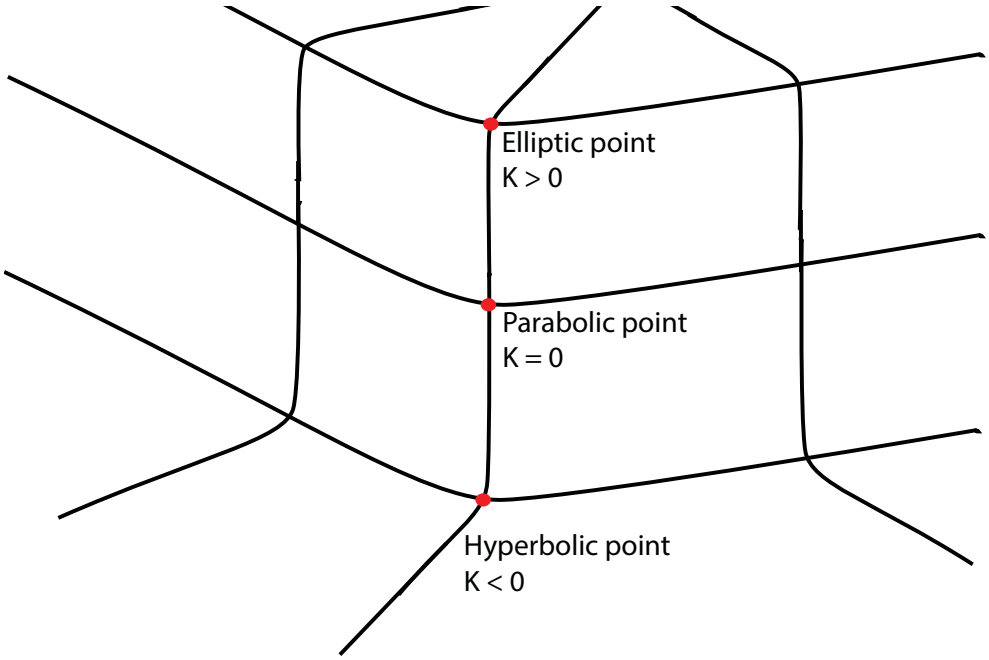


Figure 2.6: Evolution of the Gaussian curvature K in a corner structure. Corner structures can be considered as image regions that can be represented as elliptical, parabolic or hyperbolic regions at the same time.

the corner while the elliptical one is always inside. [4] proposes to estimate the determinant of the Hessian to approximate the Gaussian curvature and detect maxima responses as corners. However, corner detection using this approach in a multi-scale representation is very unstable since the spatial location suffer huge variations along the scales (see Figures 2.8 (top-right) and 2.7 (top-right)). In fact, the Hessian-Laplace detector is considered a multi-scale blob detector [90] instead of a corner detector.

A different kind of corner detectors, which are probably the most known, are based on the computation of the second moment matrix [45, 101, 120, 91] , being the most known the Harris detector [45], which is computed as $det(\mu) - \kappa\, trace^2(\mu)$ where $\kappa = 0.04$ in our experiments. These detectors detect corners inside the corner structures by roughly looking for elliptical points (see Figures 2.8 (bottom-right) and 2.7 (bottom-right)). Recall that elliptical points denotes blob structures, for this reason the multi-scale version of the Harris detector, the Harris-Laplace detector [91], is also partially considered as a blob detector [136].

2.4.6 Relations with Other Common Operators

The denominator in equation (2.13) is always positive, therefore the sign of K coincides with the sign of the numerator, which is the determinant of the Hessian matrix $\mathbf{H}$. Assuming that

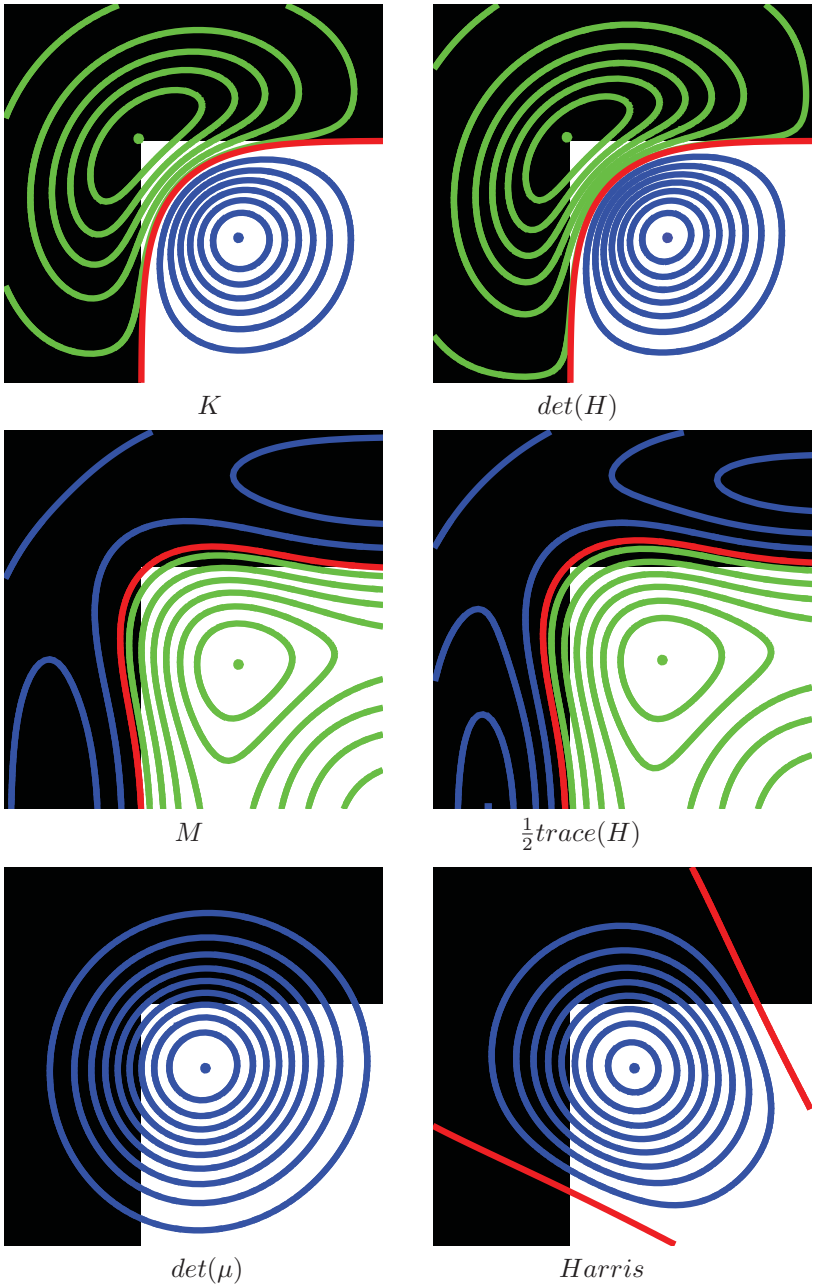


Figure 2.7: Relations between the responses of several operators on a corner structure. Green, red and blue lines denote negative, zero and positive responses. Green and blue dots denote the extreme responses.

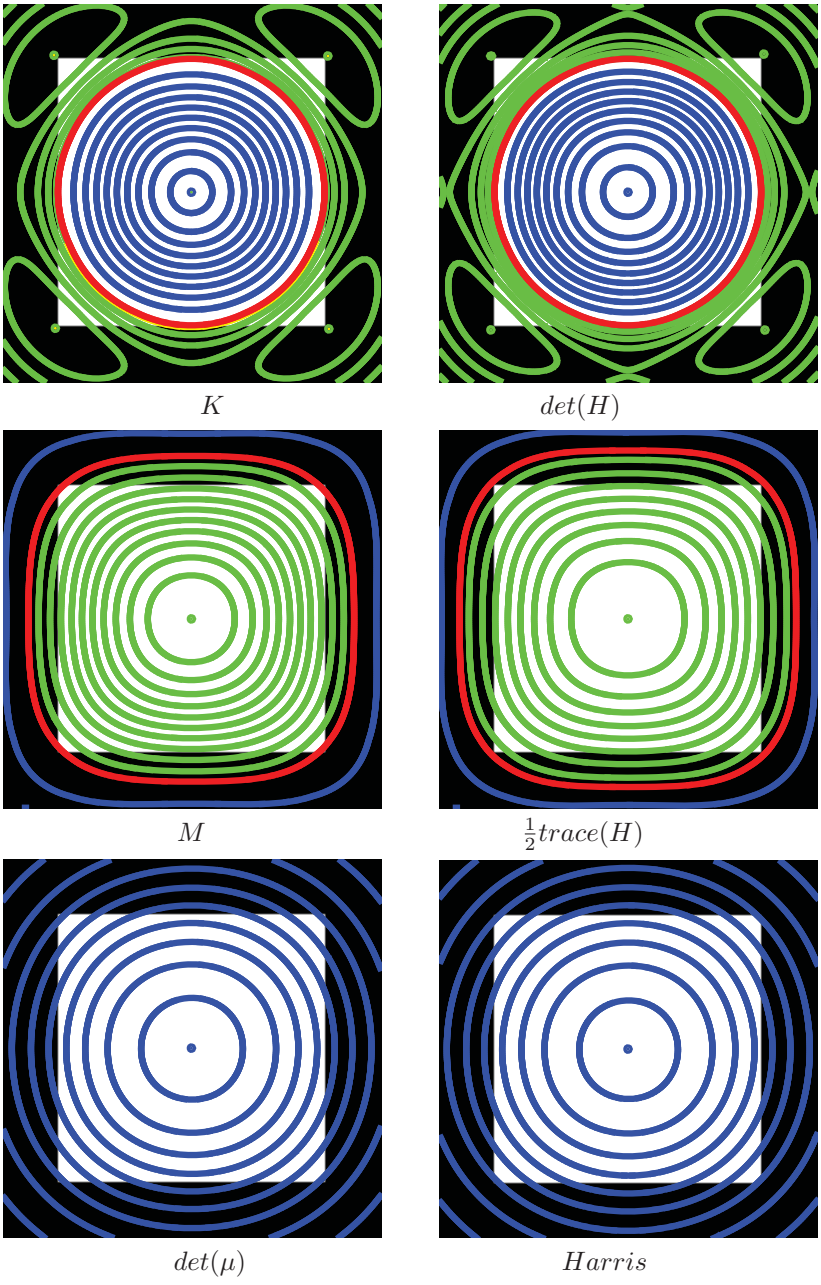


Figure 2.8: Relations between the responses of several operators on a blob structure. Green, red and blue lines denote negative, zero and positive responses. Green and blue dots denote the extreme responses.

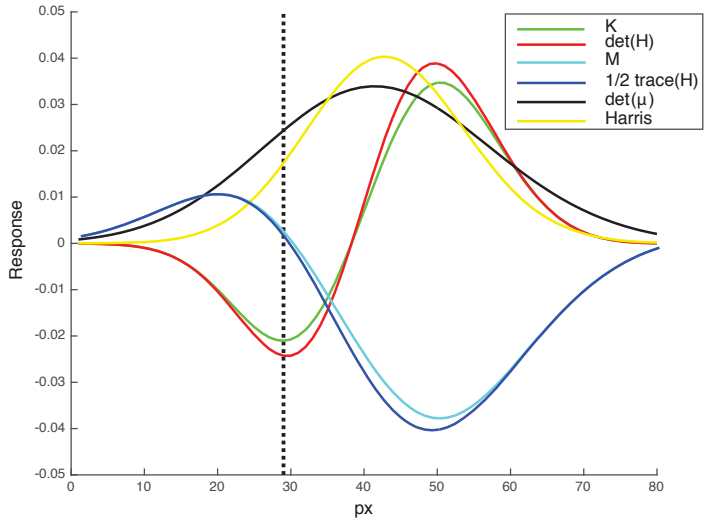


Figure 2.9: Operator responses along the pixels belonging to the diagonal from the top-left corner to the bottom-rigth corner of the image showing the synthetic corner in Figure 2.7. Vertical dashed lines denote the corner location. Responses of M, $trace(H)$, $det(\mu)$ and Harris have been scaled.

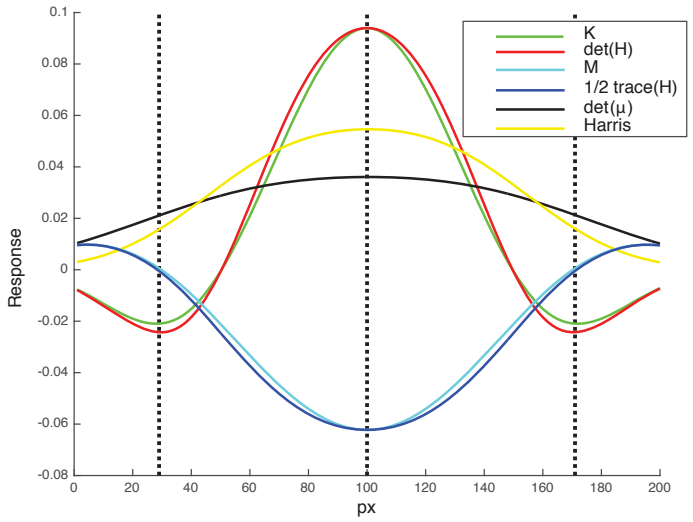


Figure 2.10: Operator responses along the pixels belonging to the diagonal from the top-left corner to the bottom-rigth corner of the image showing the synthetic square in Figure 2.8. Vertical dashed lines denote the spatial location of corners and blob center. Responses of M, $trace(H)$, $det(\mu)$ and Harris have been scaled.

a point $\mathbf{p}$ is critical ($\mathbf{L}_x$ and $\mathbf{L}_y$ vanishes), its Gaussian curvature is the determinant of $\mathbf{H}$. In the other cases, when $\mathbf{L}_x$ and $\mathbf{L}_y$ do not vanish, it can be shown that the value of K is close to $det(\mathbf{H})$ whenever the partial derivatives are estimated using normalized kernels[80]. In Figures 2.8 (top) and 2.7 (top) are shown the similarities in the surfaces generated by both, K and $det(H)$. Moreover in Figure 2.10 is shown that in critical points the responses of both are the same, in all the other cases the responses are slightly different. It is worth to remark that for corner detection, as can be seen in the Figures 2.9 and 2.10, the local extrema in $det(H)$ suffer small perturbations with respect to K. In fact, since first-order derivatives in numerator of equation (2.13) are not taken into account the surface region around the real corner is flatten and then, sometimes $det(H)$ responses cause the creation of spurious local extrema close to the real one.

Similarly, in equation (2.14) when $\mathbf{L}_x$ and $\mathbf{L}_y$ vanishes the Mean curvature is equivalent to the Laplace operator $\frac{1}{2}trace(\mathbf{H})$ (Figure 2.10). As in the previous comparison, the local extreme can suffer small variations in presence of corners. Laplacian operator produces extreme responses for saddles, corners, blobs and even contours, although for blob structures its absolute value is higher than the responses for the other structures (recall that for bright blobs responses are < 0 and for dark blobs are > 0). In Figures 2.9 and 2.10 can be seen slightly maxima around the corner regions and a prominent minimum in the blob region. Similarly, in Figures 2.7 and 2.8 around the edges slightly maxima responses are obtained. The most known detector based on Laplacian filter is [80] and its speed up version proposed by Lowe in [84]. In bibliography this detector is commonly used for scale selection jointly with other operators that create strong positive maxima like the Harris operator in the Harris-Laplace detector [91]).

Finally, in Figures 2.7, 2.8, 2.9 and 2.10 can be seen that the determinant of the second moment matrix is a very rough approximation of the Gaussian curvature and determinant of the Hessian. In fact, in Figures can be seen that both operators, Gaussian curvature and determinant of the Hessian, allows to detect with relative accuracy corner structures even in the huge scale required to detect the blob structure in Figure 2.8.

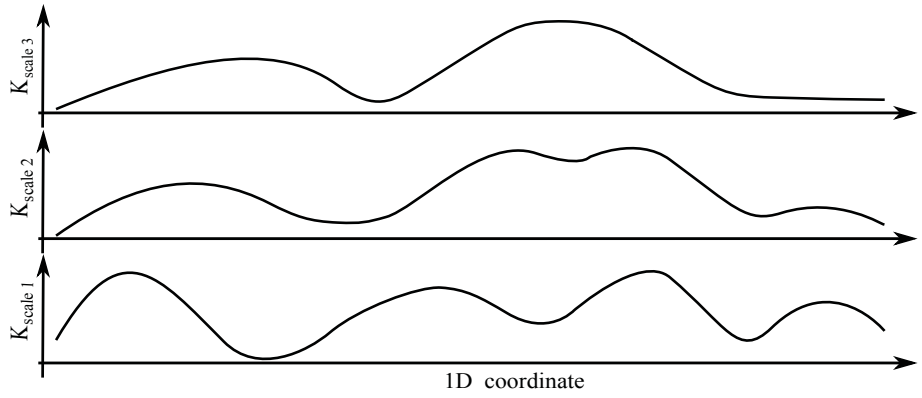
2.5 Scale Invariant Detection of Non-redundant Interest Points

In this section we present three novel scale invariant detectors that are designed thinking in two main goals: First, simplify subsequent tasks, such as matching, improving their performance and reducing their computational complexity by increasing the discriminatory power of the detected points. This can be achieved avoiding redundancy, which increases the distinctiveness and uniqueness. Second, avoid to apply several detectors, reducing the computational complexity to obtain complementary interest points, such as corner and blobs. This goal can be achieved obtaining an accurate description of the image surface that is common to complementary interest points.

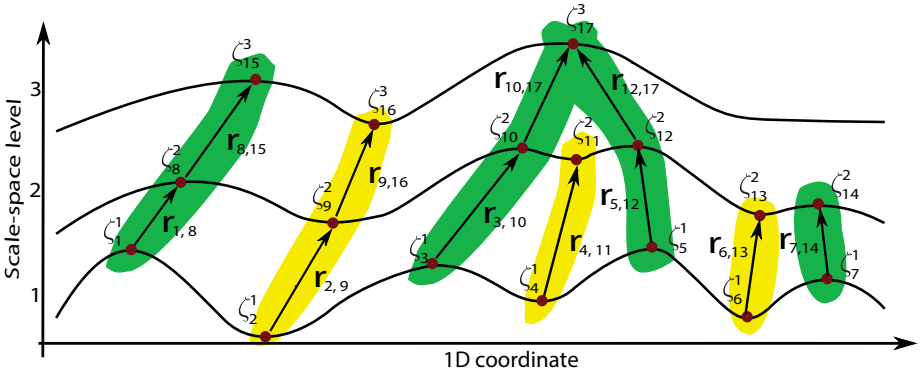
These aims are achieved in the proposed scale invariant detectors taking advantage of the Gaussian curvature properties shown in the previous section. The proposed detectors are able

to detect corners, blobs or both at the same time by means of the evolution along the scales of elliptic and hyperbolic extreme responses.

The evolution of local image structures, through the scale-space levels, is a valuable information allowing to analyse each structure independently, allowing to estimate the most distinctive location and scale for each one. For this purpose, we propose a robust bottom-up method to link elliptic and hyperbolic extreme responses between scale-space levels using a gradient ascent or descent approach depending on the region typology. Afterwards, we propose an automatic scale selection algorithm, which evaluates the responses along each trajectory independently.



(a) Example of 3 scale-space levels



(b) Trajectories along the stacked scale-space levels

Figure 2.11: From scale-space levels to blob trajectories. (a) shows 3 hypothetical consecutive scale-space levels computed using Gaussian curvature on a 1D image (for clarity). (b) shows the trajectories obtained along the stacked scale-space levels shown in (a). Blob and corner trajectories P are represented by the green and yellow regions respectively. Being each trajectory denoted as set of relations $r_{i,j}$ between candidate interest points ζ_i^k.

2.5.1 Proposed Non-redundant Detector for Simultaneous Detection of Corner and Blob Structures

In this section our novel detectors of corners and blobs avoiding redundant detections are proposed. Our approach is based on a bottom-up gradient ascent algorithm for blob detection and a gradient descent algorithm for corner detection using Gaussian curvature as operator to build the multi-scale representation.

Redundancy is avoided from the evolution of the image structures over scale, as Lindeberg in [77] we consider exist 4 events along the evolution: structure creation, structure annihilation, structure merging and structure split. However, we consider only 3 events are required in order to detect the characteristic scale for each interest point: creation, annihilation and merging events. The creation event occurs when a new structure appear in a concrete scale-space level and it is not present in the previous level. Similarly, the annihilation occurs when an structure vanishes. Both events represent the starting and finishing scales where the structures are meaningful. The merging event happens when two or more existing structures join in a single one. In Figure 2.2(right) these three events are shown, we can see four blob creation events, three merging events and one annihilation. The split event consist in the division of one structure in two or more structures. In our approach each split event results in the creation of a new evolution chain. However, if a complete description of the image is required as in [77] we can estimate the split events by using the same strategy but from a top-down point of view.

The initial step of our detectors is the construction of a multi-scale representation $\mathbf{L}$ by using the Gaussian curvature K as operator at n different scales σ_k using equation (2.3). Once $\mathbf{L}$ is constructed, for each scale-space level $\mathbf{L}(x, y, \sigma_k)$ the candidate interest points ζ_i^k are extracted by detecting the local extreme responses in the 8-neighbourhood of each pixel (local maxima for blobs and local minima for corners). The relations $r_{i,j}$ between candidates ζ_i^k at consecutive scale-space levels are extracted in order to determine the trajectory of each blob or corner structure P_l. The final set of interest points X is found looking for local extreme responses of K for each P_l. Figure 2.11 intuitively shows the behavior of the algorithm in a 1D signal where local maxima denote blob structures and local minima denote corner structures. Figures 2.12 and 2.13 show the output for a synthetic and a real image. Concretely, the algorithm to extract the interest points is:

1. For each candidate ζ_i^k
   (a) $\rho^{k+1} \leftarrow$ Project ζ_i^k to level $k+1$.
   (b) Apply a gradient ascent/descent algorithm to ρ^{k+1} in $\mathbf{L}(x, y, \sigma_{k+1})$ in order to maximize/minimize K and obtain the relation $r_{i,j}$ between two candidate points, $r_{i,j} \leftarrow \left(\zeta_i^k, \zeta_j^{k+1}\right)$
   (c) Add $r_{i,j}$ to the existing set of relations, $R \leftarrow \{R, r_{i,j}\}$

   End
2. By concatenation of relations in R, each trajectory P_l is built. $P_l \leftarrow \{\zeta_i^k, \zeta_j^{k+1}, \zeta_m^{k+2}...\} \leftarrow \{r_{i,j}, r_{j,m}...\}$
3. Interest points X are selected as the local maxima/minima of the K along each trajectory P_l.

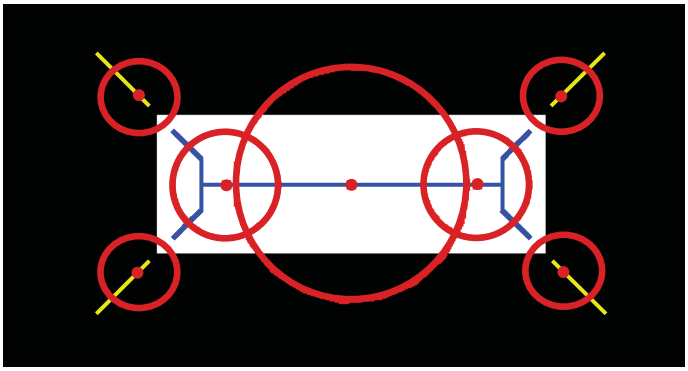


Figure 2.12: Output of our scale invariant blob and corner detector on a synthetic image. The estimated blob and corner trajectories are shown in blue and yellow, respectively. In red are shown the selected interest points.

Therefore, our detector of blobs only looks for local maxima candidates, our corner detector only looks for local minima candidates and our full detector looks for both.

The set of trajectories P contains the creation, merging and annihilation of each blob and corner structure. In order to take into account the split event, the algorithm is used in a top-down manner, projecting the candidates ζ_i^k to an inferior level $k-1$.

An important property of the proposed detectors is related with thresholding. These detectors do not need to use any threshold in contrast with other detectors, which use thresholds to discard not good enough points and select the most reliable points (e.g. [80, 91, 84]).

2.5.2 Computational Complexity

Scale invariant detectors based on a muti-scale representation consume most of the computational cost in the scale-space construction. This cost is related with the convolution of operators at different scales. In order to compare our detectors with the state-of-the-art, we have selected two of the most known blob and corners multi-scale detectors, the Hessian-Laplace [90] and the Harris-Laplace [91] detectors. The Hessian-Laplace detector needs to compute the determinant and trace of the Hessian matrix while the Harris-Laplace needs to compute the laplacian operator and the determinant and trace of the second moment matrix. Our detectors require to compute the Hessian matrix and the Jacobian matrix. In Table 2.2 the detectors complexity is compared.

In our detectors the gradient ascent/descent algorithm can be considered a source of complexity comparing with state-of-the-art detectors, however in our experiments we have seen its complexity is marginal (it is less than 1% of the total cost). From our point of view, the computational cost should not be a problem for real time execution since all the convolutions can be simultaneously done using parallel computing.

Detector (scale invar.)	Operation on image (candidate points)	Operations on patch (point selection)
Ours	$\#s\ K(I)$	$\#m\ \nabla K$
Hessian-Laplace	$\#s\ det(H(I))$	$\#3m\ (d_{xx} + d_{yy})$
Harris-Laplace	$\#s(\ det(\mu(I)) - \alpha\ trace(\mu(I))$	$\#3m\ (d_{xx} + d_{yy})$

Table 2.2: Comparison in terms of computational complexity and cost between our scale invariant detectors and the Hessian-Laplace and Harris-Laplace detectors. $K(I)$ denotes the Gaussian curvature, $H(I)$ the Hessian matrix and $\mu(I)$ the second moment matrix computed for each image point. For each candidate point, ∇K is the gradient ascent/descent applied on the upper scale-space level and $\#3\ (dxx+dyy)$ is a convolution by the Laplacian operator in 3 adjacent scale-space levels. $\#s$ denotes the number of scales and $\#m$ the number of candidate interest points.

2.6 Affine Invariant Blob Detection Fitting Gaussian Functions

In this section we exhibit our shape adaptation algorithm based on the idea of modelling blobs by fitting an anisotropic Gaussian function, in front of the Mikolajczyk and Schmid method [91] based on the second moment matrix. Our algorithm allows simultaneously modelling the spatial location and shape of each blob structure, in contrast with the Mikolajczyk and Schmid algorithm, which is performed in two steps: shape estimation and point re-detection reapplying a scale invariant detector. In order to achieve this simultaneity we propose an optimization method solved applying a weighted non-linear least squares approach.

A Gaussian function has been selected as model because all the values of the domain have a positive image. Thus, the limit when domain values are moving away from the $(0, 0)$ tends to 0, and so the error between the Gaussian function and the image region is controlled. However, parametric functions that do not accomplish this property cannot be estimated accurately. For example, in parabolic functions the error is infinite for far away points.

2.6.1 Weighted Non-linear Least Squares Algorithm in order to Fit Gaussian Functions on Images

Gaussian functions are not linear and do not have a polynomial representation, for this reason a non-linear least squares algorithm is necessary to fit them [24]. Furthermore, to improve the robustness, the Levenberg-Marquardt method [86, 73] and a weighted version of the algorithm have been used.

Given a Gaussian function $\mathbf{G}$ of two variables (x, y) depending on 7 parameters $\boldsymbol{\lambda} = (\beta,$ $\mu_x, \mu_y, \alpha, \sigma_x, \sigma_y, \rho)$,

$$\mathbf{G}(x,y) = \beta + \alpha\ \exp^{-\frac{1}{2}\begin{pmatrix} x-\mu_x & y-\mu_y \end{pmatrix}\begin{pmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_y\sigma_x & \sigma_y^2 \end{pmatrix}^{-1}\begin{pmatrix} x-\mu_x \\ y-\mu_y \end{pmatrix}} \tag{2.15}$$

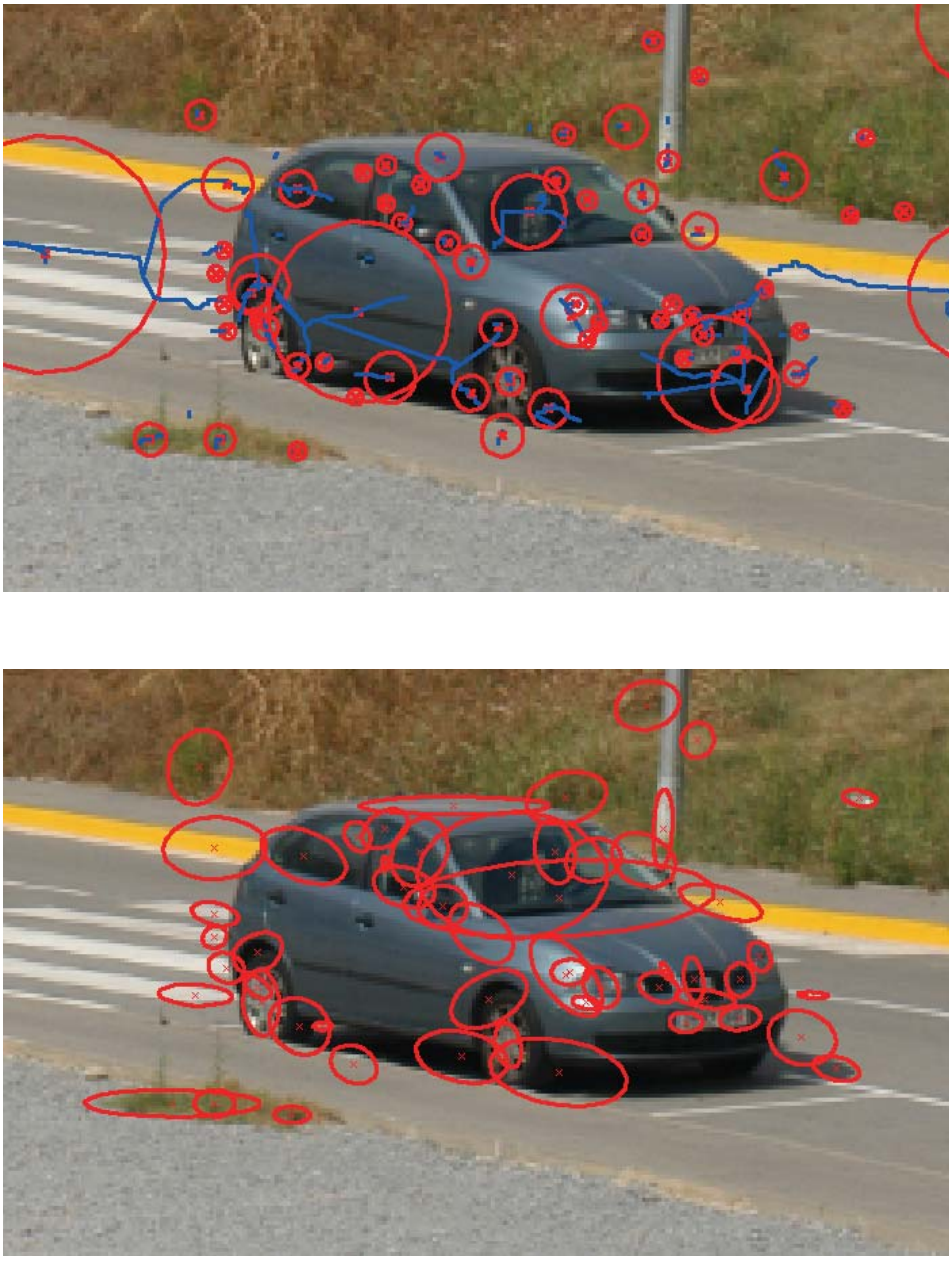


Figure 2.13: Output of our scale invariant blob detector (top image) and output of the proposed shape adaptation algorithm (bottom image). Trajectories appear on the top image in blue and interest points appear in red in both images. Comparing both images is noticeable that some scale invariant blobs do not converge to an affine blob.

The goal is to find the parameters $\boldsymbol{\lambda}$, which best fit the Gaussian function with the given candidate image region, in the least squares sense. Taking m pixel intensity values of the image $\mathbf{I}(x, y)$ and picking an initial guesswork for the parameters, their residuals $\mathbf{r}_i$ are defined as,

$$r_i = \mathbf{I}(x_i, y_i) - \mathbf{G}(x_i, y_i) \quad , \quad i = 1, 2, ..., m. \tag{2.16}$$

The minimum square error $S = \sum_{i=1}^{m} \mathbf{r}_i^2$ occurs when the gradient is zero. Thus, since the Gaussian model contains 7 parameters there are 7 gradient equations. In order to obtain a linearised estimation for the changes $\boldsymbol{\Delta\lambda} = (\Delta\beta, \Delta\mu_x, \Delta\mu_y, \Delta\alpha, \Delta\sigma_x, \Delta\sigma_y, \Delta\rho)$ it is necessary to reduce residuals r_i to 0 by solving,

$$\begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{pmatrix} = \begin{pmatrix} 1 & \frac{\partial f_1}{\partial \mu_x} & \frac{\partial f_1}{\partial \mu_y} & \frac{\partial f_1}{\partial \alpha} & \frac{\partial f_1}{\partial \sigma_x} & \frac{\partial f_1}{\partial \sigma_y} & \frac{\partial f_1}{\partial \rho} \\ 1 & \frac{\partial f_2}{\partial \mu_x} & \frac{\partial f_2}{\partial \mu_y} & \frac{\partial f_2}{\partial \alpha} & \frac{\partial f_2}{\partial \sigma_x} & \frac{\partial f_2}{\partial \sigma_y} & \frac{\partial f_2}{\partial \rho} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ 1 & \frac{\partial f_m}{\partial \mu_x} & \frac{\partial f_m}{\partial \mu_y} & \frac{\partial f_m}{\partial \alpha} & \frac{\partial f_m}{\partial \sigma_x} & \frac{\partial f_m}{\partial \sigma_y} & \frac{\partial f_m}{\partial \rho} \end{pmatrix} \begin{pmatrix} \Delta\beta \\ \Delta\mu_x \\ \Delta\mu_y \\ \Delta\alpha \\ \Delta\sigma_x \\ \Delta\sigma_y \\ \Delta\rho \end{pmatrix} \tag{2.17}$$

where the gradient matrix is the Jacobian $\mathbf{J}$ of the Gaussian function. This equation can be written as $\mathbf{r} = \mathbf{J}\ \boldsymbol{\Delta\lambda}$. Solving the previous equation, $\mathbf{J}^T$ is applied to both sides, obtaining,

$$\mathbf{J}^T\mathbf{r} = (\mathbf{J}^T\mathbf{J})\boldsymbol{\Delta\lambda} \tag{2.18}$$

However, not all the observations $\mathbf{I}(x_i, y_i)$ are equally relevant, because not all the m pixels belong always to the same local structure. Thus, assuming that far pixels are less relevant because they can belong to other local structures, a weighted sum of square errors $S = \sum_{i=1}^{m} \mathbf{W}_{ii}\mathbf{r}_i^2$ should be minimized,

$$\mathbf{J}^T\mathbf{W}\mathbf{r} = (\mathbf{J}^T\mathbf{W}\mathbf{J})\boldsymbol{\Delta\lambda} \tag{2.19}$$

being $\mathbf{W}$ assigned according to a Gaussian function parametrized by $\boldsymbol{\lambda}$.

To increase the robustness of $\boldsymbol{\Delta\lambda}$ estimation, the Marquardt parameter [86] is added to the equation,

$$\mathbf{J}^T\mathbf{W}\mathbf{r} = (\mathbf{J}^T\mathbf{W}\mathbf{J} + \gamma\ diag(\mathbf{J}^T\mathbf{W}\mathbf{J}))\boldsymbol{\Delta\lambda} \tag{2.20}$$

where γ is the Marquardt parameter and $diag(\mathbf{J}^T\mathbf{W}\mathbf{J})$ allows to control its scale.

The Levenberg-Marquardt method [86, 73] can be seen as a combination between the Gauss-Newton algorithm and the steepest descent. Increasing the value of γ has the effect of changing both, the direction (rotating towards the direction of steepest descent) and the norm (making it smaller) of $\boldsymbol{\Delta\lambda}$. So, γ is increased when squared error S increases (making the steepest descent direction stronger) and γ is decreased when squared error S decreases

(making the Gauss-Newton direction stronger). γ is decreased when error decreases because Gauss-Newton method is generally more effective when the squared error S is near to zero. The Levenberg-Marquardt method consists on the iteration of the equation (2.20), modifying γ until the squared error difference between two iterations is lower than a constant τ. Thereby, the equation weights $\mathbf{W}$ are updated on each iteration controlling robustly the addition of new pixels into the model.

Equation (2.20) can be solved for $\mathbf{\Delta\lambda}$ using standard matrix techniques. $\mathbf{\Delta\lambda}$ is then applied to the parameters and new residuals $\mathbf{r}$ are calculated. By iteratively applying this procedure a solution is obtained. As convergence criterium to decide when the solution is obtained we propose,

$$\left| \frac{S^k - S^{k+1}}{S^k} \right| < \tau \tag{2.21}$$

where τ is a threshold, concretely we use $\tau = 0.01$. S^k is the residual for the iteration k.

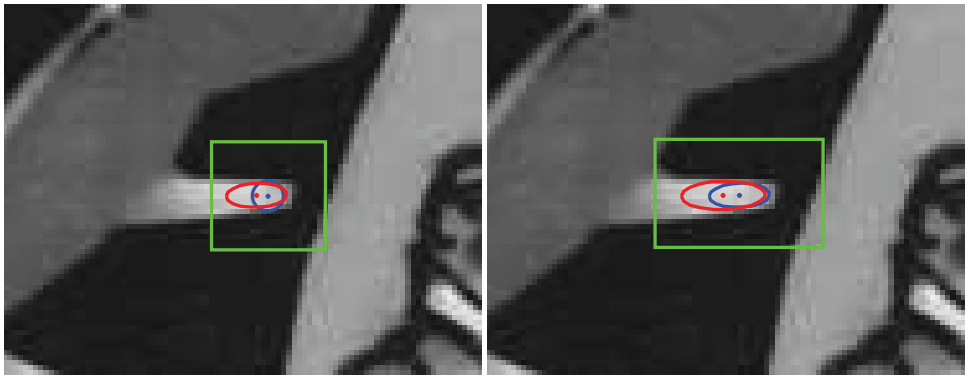
Note that the procedure convergence is often greatly improved by picking initial values close to the best-fit value. Thereby, depending on the goodness of the initial parameters, obtained from a scale invariant blob detector, the convergence will be quicker, slower or even not achieved. Figure 2.13 shows some examples of non-convergence of initial scale invariant blobs to affine blobs, for example the big blob detected on the crosswalk do not converge since the underlying image region is not a clear blob-like structure. In our experiments we discard affine blobs when their center is not found inside the image. However, other measures can be used to quickly discard these kind of regions such as, the intersection with the initial scale invariant blob, huge spatial location changes and/or scale changes. These measures reinforce that the final blob fits on the initially detected local image structure. For example, in [91] to ensure that the shape adaptation process converges fast on the correct image region the spatial location is constrained.

2.6.2 Proposed Shape Adaptation Algorithm

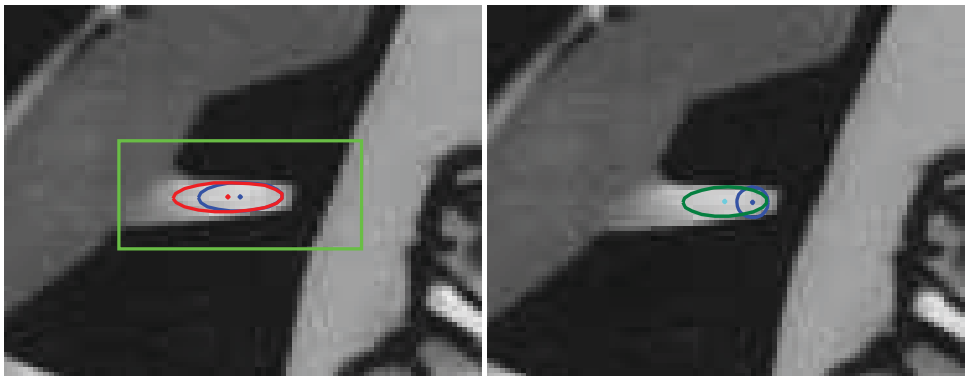
Our shape adaptation algorithm is based on two main ideas. The first one consists in the estimation of the spatial location and shape of blob structures simultaneously using a weighted non-linear least squares approach. The second one is related with the evaluation of the previous estimation, proposing evaluate least squares solutions by means of Gaussian curvature K. Recall that K is maximized when the image region defined by the least squares solution contains a blob structure (see Section 2.4).

It is easy to understand that Gaussian functions can be useful approximating bright blobs because of their similar shapes. That means dark blobs cannot directly be approximated, they must be firstly converted into bright blobs by inverting their image regions. In order to differentiate between these two kinds of blob the mean curvature M plays an essential role, classifying blobs as bright when $M < 0$ and dark when $M > 0$ (see Table 2.1).

The proposed algorithm requires an initial scale invariant blob X_i with $K_i > 0$, initial spatial location $\mathbf{p}_i$ and scale σ_i. The outline of the algorithm for a given X_i blob is:



(a) $K = 0.0330$ (b) $K = 0.0418$



(c) $K = 0.0406$ (d) Final $K = 0.0418$

Figure 2.14: Example of execution of our shape adaptation algorithm. (a) shows the initial isotropic blob in blue ($K = 0.0252$) and the solution (red ellipse) found in the first iteration of our method. (b) and (c) show the two next iterations of the algorithm. The green rectangles are the image regions used in the modeling process. In (d) the final result is shown, where the green ellipse represents the solution that maximizes the *Gaussian curvature* ($K = 0.0418$).

1. Initialization of the covariance matrix for X_i, which defines the shape for $\mathbf{p}_i$ as

$$\mathbf{C}_i = \begin{pmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix}_i = \begin{pmatrix} \sigma_i & 0 \\ 0 & \sigma_i \end{pmatrix}_i \tag{2.22}$$

2. Do

   (a) Extraction of the underlying image region $\mathbf{I}^l$ associated to the parameters $(\mathbf{p}, \sigma_x, \sigma_y, \rho)_i^l$, where $\mathbf{I}^l$ size and orientation is determined by the eigenvectors of $\mathbf{C}_i^l$ multiplied by a constant θ (we use $\theta = 3$).

   (b) Estimate the new parameters $(\mathbf{p}, \sigma_u, \sigma_v, \rho)_i^{l+1}$ on $\mathbf{I}^l$ using the weighted non-linear least squares method shown in the previous section.

(c) Calculate K^{l+1} for $\mathbf{I}^{l+1}$ defined by $(\mathbf{p}, \sigma_x, \sigma_y, \rho)_i^{l+1}$.

While $K^{l+1} > K^l$

3. Finally, the affine invariant blob X_i' is defined by $(\mathbf{p}, \sigma_x, \sigma_y, \rho)_i^l$ maximizing K^l.

In Figure 2.14 a step by step example of this algorithm is shown. Showing how the Gaussian function is fitted in each iteration in order to maximize K. Figures 2.13 and 2.20 show the final results of the adaptation process for a given set of scale invariant blobs.

2.6.3 Computational Complexity

Our shape adaptation algorithm obtains the affine invariance by estimating the parameters of equation (2.15) iteratively, until the Gaussian curvature is maximized. Thereby, since the parameters $\boldsymbol{\lambda}$ for each iteration are estimated by another iterative algorithm, the main computational cost of our approach is due to this nested loop. The number of iterations related to each loop depends on the initialization (better initial parameters imply less iterations). Typically, the number of iterations $\#n_1$ and $\#n_2$ are less than 6 and 4 respectively. On the other hand, for the algorithm proposed by Mikolajczyk and Schmid, $\#n$ usually is less than 10 [91].

In Table 2.3 a complexity comparison of our proposed shape adaptation algorithm with the shape adaptation algorithm used in the Hessian-Affine detector is done and it shows that the main part of the complexity of the Mikolajczyk and Schmid method is due to the re-detection step to refine the location and scale.

Detector	Shape + Location
Ours Affine	$\#n_1\ \#n_2\ Sol.Eq.Syst.$
Hessian-Affine	$\#n\ \boldsymbol{\mu}(\mathbf{x}) \cdot (\#s\ det(\mathbf{H}(\mathbf{x}_{neighbors})) + \#3m\ (d_{xx} + d_{yy}))$

Table 2.3: Comparison in terms of computational complexity and cost between our affine detector and the Hessian-Affine detector. $\boldsymbol{\mu}(x)$ is the second moment matrix computed for a point $\mathbf{x}$ and $\mathbf{H}(\mathbf{x}_{neighbors})$ is the Hessian matrix computed in an image region near to $\mathbf{x}$. $\#3\ (dxx + dyy)$ is a convolution by a Laplacian operator in 3 adjacent scale-space levels. $\#n$ denotes the number of iterations per point, $\#s$ denotes the number of scales for the re-detection step and $\#m$ is the number of scale-invariant interest points detected in the re-detection step.

2.7 Experimental Evaluation

In this section we validate the advantages and disadvantages of the proposed scale and affine invariant detectors in four experimental setups. First, we evaluate the repeatability and matching scores using the evaluation framework provided by Mikolajczyk *et al.* [90]. Secondly, we evaluate the precision and recall scores. Thirdly, a novel measure of over-detection is presented in order to evaluate the redundancy and sparsity of interest points. Finally, from interest points and the RANSAC algorithm [32] a set of homographies are estimated in order

to evaluate the behavior of interest points in a real task. All these experimental setups have been evaluated, by using the images and homographies between image pairs from Mikolajaczyk's evaluation sequences[1], under scale, rotation, viewpoint, image blur, compression and illumination variations. Specifically, there are 8 image sequences composed of 6 images sorted according to the amount of variation.

Scale invariant evaluation : To evaluate our scale invariant detectors we analyse the behaviour of our full detector, where corners and blobs are not treated as different structures, and the behaviour of our corner detector and blob detector individually. We compared our detectors with three of the most known scale invariant detectors in the state of the art: the Derivative of Gaussian (DoG) detector proposed by Lowe in [84], the Hessian-Laplace detector [90][2] and the Harris-Laplace detector [91][2]. The first two detectors are blob detectors and the third one is a corner detector, all of them based on multi-scale representations.

Affine invariant evaluation : The evaluation of our affine invariant blob detector has been done with three affine invariant detectors: the Hessian-Affine detector [90][2], the MSER detector [88][2] and the gradient histogram based detector proposed in [70][3]. All of them have shown good performance in terms of repeatability and matching in [90] and [70].

2.7.1 Repeatability and Matching

Repeatability score measures the constancy of interest points under homographies [90]. Repeatability for a given pair of images is the ratio between the number of point-to-point correspondences and the smaller number of points detected in one of the images inside their common part. Those point-to-point correspondences are computed measuring the overlap error between interest points coming from different images. For example, if we denote two interest points by $\boldsymbol{\mu}_a$ and $\boldsymbol{\mu}_b$, and their corresponding image regions by $\mathbf{S}_{\boldsymbol{\mu}_a}$ and $\mathbf{S}_{\boldsymbol{\mu}_b}$, their overlapping error can be defined as,

$$1 - \frac{\mathbf{S}_{\boldsymbol{\mu}_a} \cap \mathbf{S}_{(\mathbf{H}^\top \boldsymbol{\mu}_b \mathbf{H})}}{\mathbf{S}_{\boldsymbol{\mu}_a} \cup \mathbf{S}_{(\mathbf{H}^\top \boldsymbol{\mu}_b \mathbf{H})}} \tag{2.23}$$

where $\mathbf{H}$ is the known planar homography relating both regions and hence $\mathbf{H}^\top \boldsymbol{\mu}_b \mathbf{H}$ the projection operator from one image to the other one. $\cap$ and $\cup$ denote the intersection and union of the image regions. In this experimental evaluation, as in [90], two interest points are deemed as a correspondence when their supporting image regions have an overlap error $< 40\%$.

Matching score measures the matching ability of interest points under homographies [90]. Matching score is the ratio between the number of correct matches and the smaller number of detected regions in one of the matched images. The best matching for each interest point is found as its nearest neighbour in the SIFT [84] feature space, and its correctness is evaluated by using the repeatability measure as ground truth.

[1] Evaluation sequences can be obtained from `http://www.robots.ox.ac.uk/~vgg/research/affine/`

[2] obtained from `http://www.robots.ox.ac.uk/~vgg/research/affine/`

[3] obtained from `http://740-2.cs.nthu.edu.tw/~htchen/hipd/`

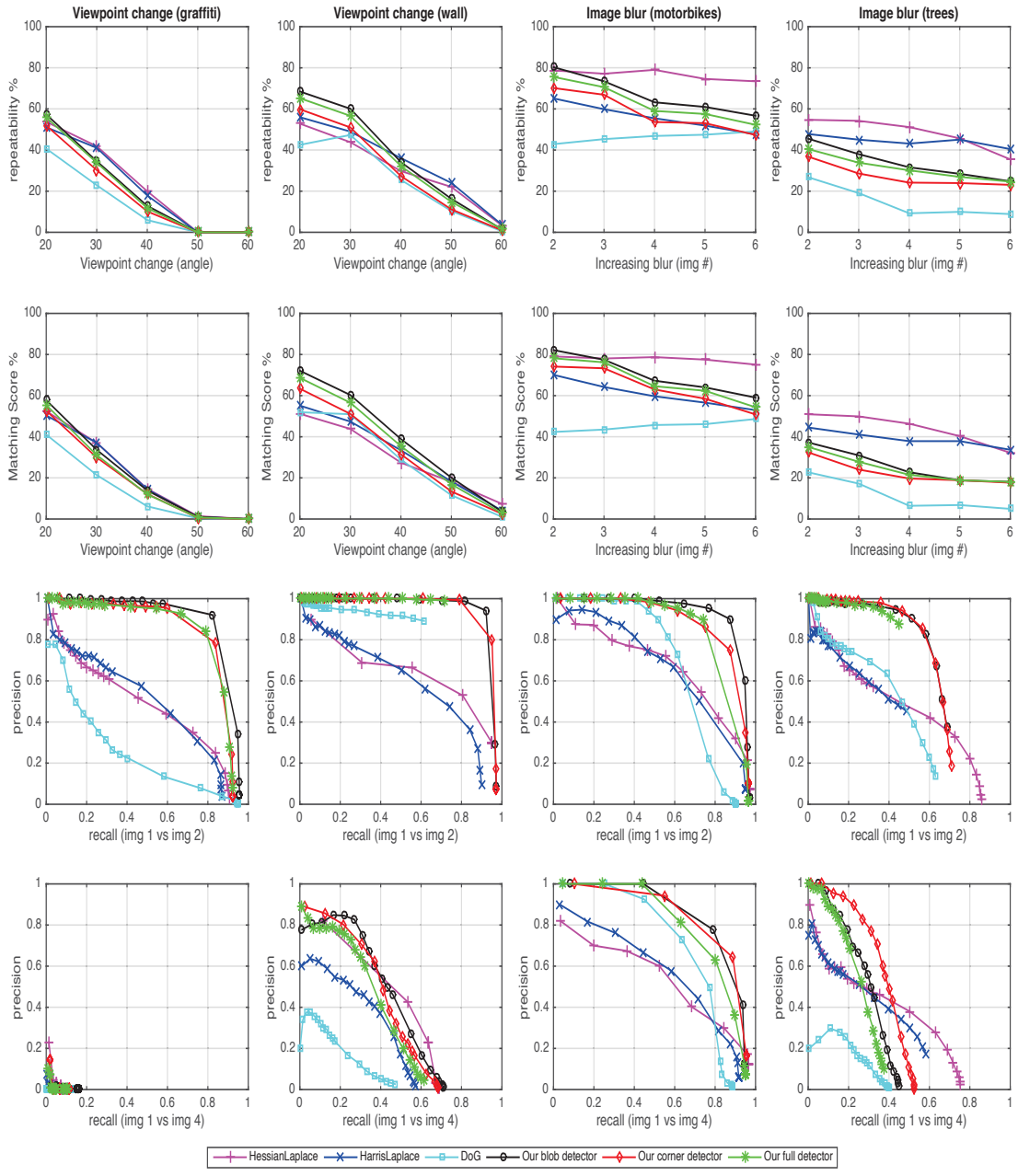


Figure 2.15: Comparison of scale invariant detectors in terms of repeatability, matching and precision/recall from evaluation sequences with viewpoint changes and blurring.

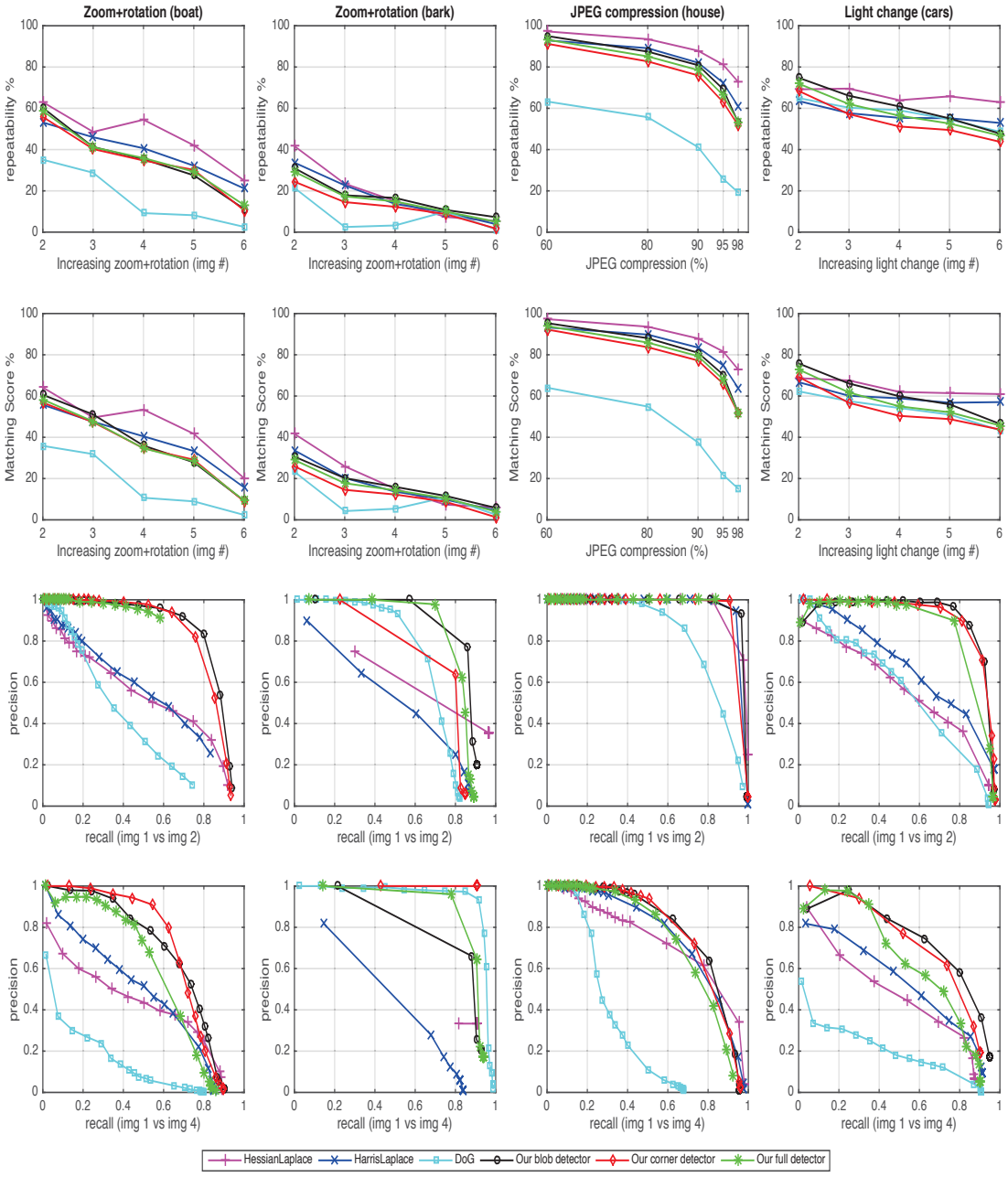


Figure 2.16: Comparison of scale invariant detectors in terms of repeatability, matching and precision/recall from evaluation sequences with zoom+rotation, JPEG compression and illumination changes.

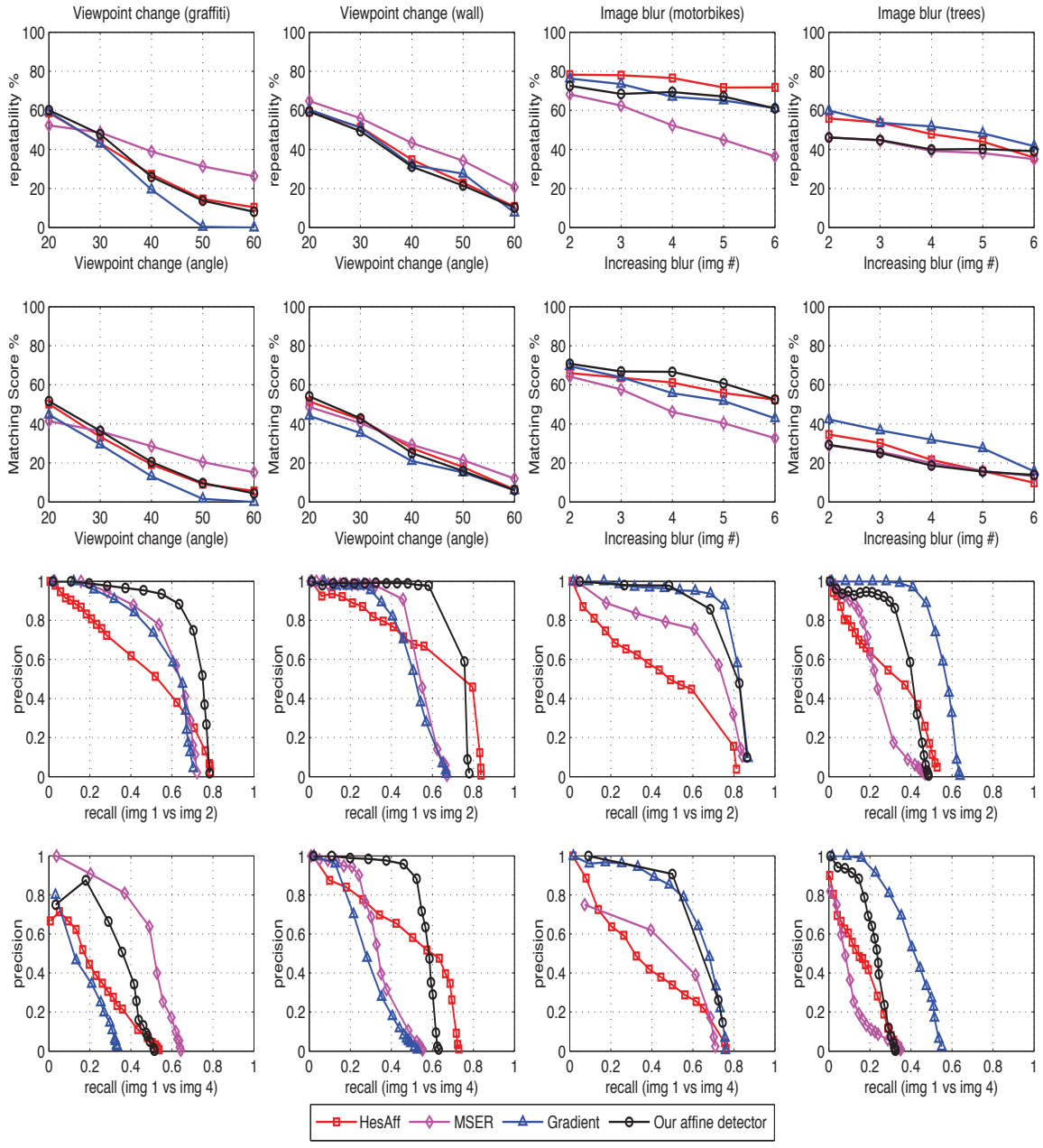


Figure 2.17: Comparison of affine invariant detectors in terms of repeatability, matching and precision/recall from evaluation sequences with viewpoint changes and blurring.

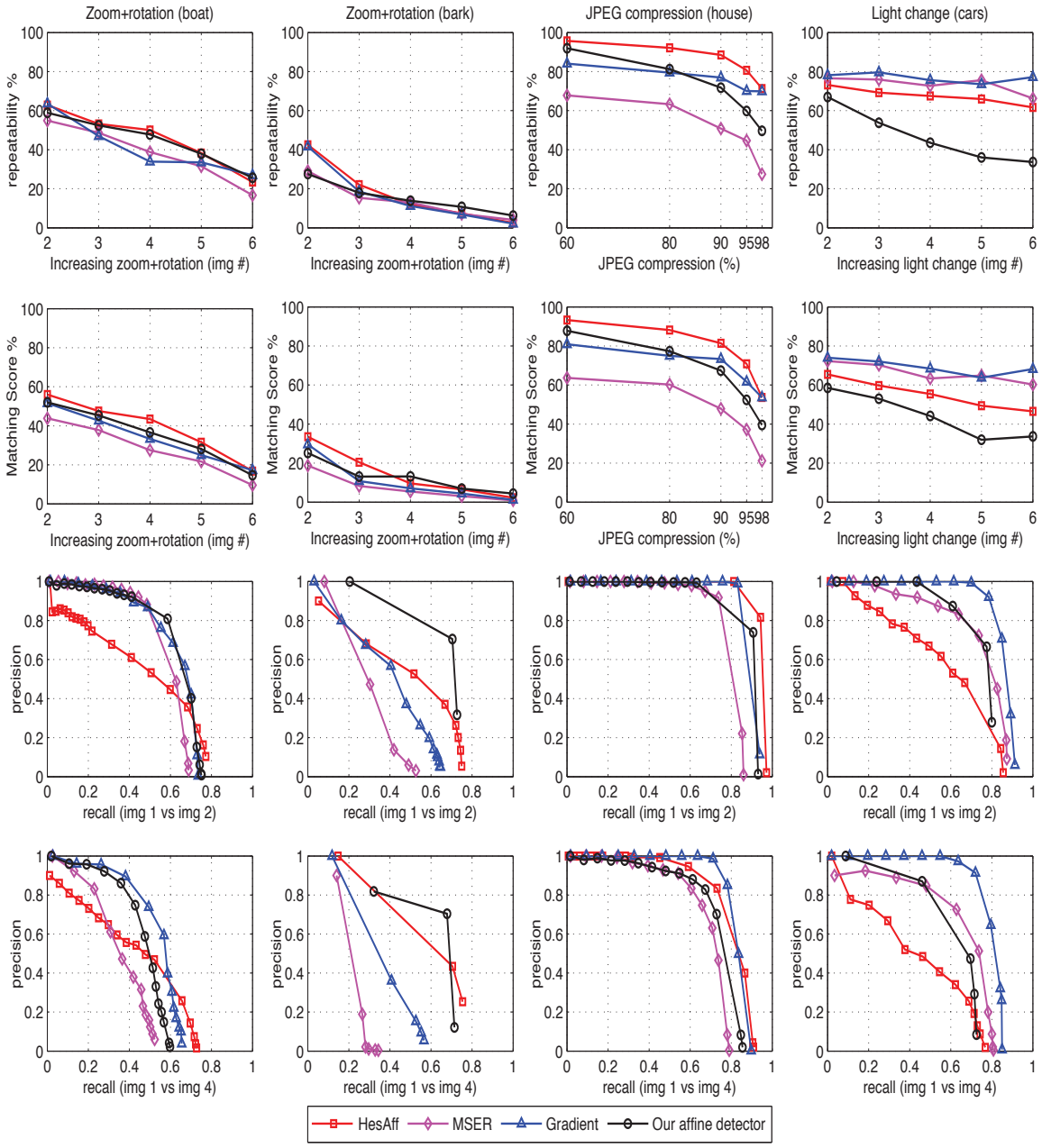


Figure 2.18: Comparison of affine invariant detectors in terms of repeatability, matching and precision/recall from evaluation sequences with zoom+rotation, JPEG compression and illumination changes.

Scale invariant evaluation : Figure 2.15 and Figure 2.16, in terms of repeatability and matching scores, show that there is not any scale invariant detector that clearly outperforms the others, however Hessian-Laplace detector tends to be the one providing better results. Concretely our three scale invariant detectors obtain similar results being the best one the scale invariant blob detector while the full detector, detecting blobs and corners, is the second one.

Affine invariant evaluation : In terms of repeatability and matching scores Figures 2.17 and 2.18 show that there is not an affine invariant detector outperforming all the other ones in all the sequences, as pointed out in [90]. Specifically our affine detector obtains, in terms of repeatability and matching scores, similar results to the best detector in viewpoint, blurring and zoom+rotation sequences. In the JPEG compression sequence performance is good for lower compression rates, but for higher compression rates it declines. Finally, in the light sequence our detector obtains good performance but it does not achieve as good performance as the other detectors.

Scale invariant versus affine invariant detectors : The main difference between scale and affine invariant detectors is their behaviour in sequences with viewpoint changes. In Figure 2.15 is shown how all the scale invariant detectors provides repeatability and matching scores close to zero for high viewpoint changes. On the other hand in Figure 2.17 is shown how the repeatability and matching scores improves for high viewpoint changes. In all the other sequences the repeatability and matching scores are similar in both, scale and affine detectors.

2.7.2 Precision and Recall

The precision measure allows to evaluate the distinctiveness of descriptors associated to interest points. Showing how the correct matches decline as the total number of matches (correct + false positive matches) is increased. Thus, precision score is defined as,

$$Precision = \frac{\#correct\ matches_{thr}}{\#total\ matches_{thr}} \tag{2.24}$$

where thr denotes a threshold on the matching measure.

In contrast, the recall measure is the number of correct matches (thresholded as in the precision measure) with respect to the absolute number of correspondences (overlapping error $< 40\%$). Thus, recall score is defined as,

$$Recall = \frac{\#correct\ matches_{thr}}{\#total\ correspondences} \tag{2.25}$$

Precision/recall curve evaluates the performance of detectors looking for high precision scores with high recall scores. Thus, when precision and recall are high, a high percentage of correspondences are identified as correct matches containing few matching errors (false positives).

The third and fourth rows in Figures 2.15, 2.16, 2.17 and 2.18 show the behaviour of all the evaluated detectors, in terms of precision and recall. The third row denotes the behaviour under slight variations between images while the fourth row denotes the behaviour under heavy variations.

Scale invariant evaluation : Our three scale invariant detectors for slight variations clearly outperforms all the other detectors in all the sequences. The only one sequence where the other detectors provide comparable results is the JPEG compression sequence. In the case of heavy variations our detectors also outperform the other ones in 5 sequences, and provides similar results in 2 sequences. Finally, for heavy variations in one of the viewpoint change sequences all the scale invariant detector fails.

Affine invariant evaluation : In case of slight variations, our affine invariant detector clearly outperforms the other detectors in viewpoint sequences, while in zoom+rotation sequences our detector outperforms or obtains a similar performance to the best detector. Specifically, with slight variations between images, we can see that our detector provides the best recall score with a precision of 80% in 4 sequences and the second best in the other 4 sequences. On the other hand, for heavier variations the performance is similar, in 3 sequences is the best and in other 3 sequences is the second best.

Comparing detectors one by one with our affine blob detector in terms of precision and recall scores, Figures 2.17 and 2.18 show that Hessian-Affine detector is clearly outperformed by our detector. This fact is remarkable because both, Hessian-Affine detector and our affine detector, are based on geometry operators and try to detect the same kind of structures. The MSER detector is outperformed in almost all the sequences as well, this fact is very important from our point of view because MSER has proved its performance to locate very stable gray-level regions [90]. Finally, the detector based on gradient histograms shows to be a powerful detector, obtaining better results than our detector in 3 sequences, drawing in 2 and worse results in 3.

Concretely for viewpoint, scale, rotation or blurring variations the overall behaviour of our affine detector is better than the overall behaviour of other affine detectors. In spite of this, our affine detector obtains similar scores in terms of repeatability and matching. However, for the illumination variations, where our repeatability and matching scores are worse, the precision and recall scores are on the average respect to the other detectors. Thereby, we can assume that our affine detector finds image regions with higher distinctiveness, showing that precision declines more slowly than the other detectors for similar recall scores.

Scale invariant versus affine invariant detectors : The main remarkable difference between scale and affine invariant detectors is shown in viewpoint change sequences with heavy variations, in those sequences the precision and recall scores are significantly improved. For the other sequences with heavy variations scale and affine detectors provide similar scores.

However in the case of slight variations the precision and recall scores tend to be worst for affine detectors, we think that this is because of shape adaptation processes are more sensitive to noise than multi-scale approaches [84].

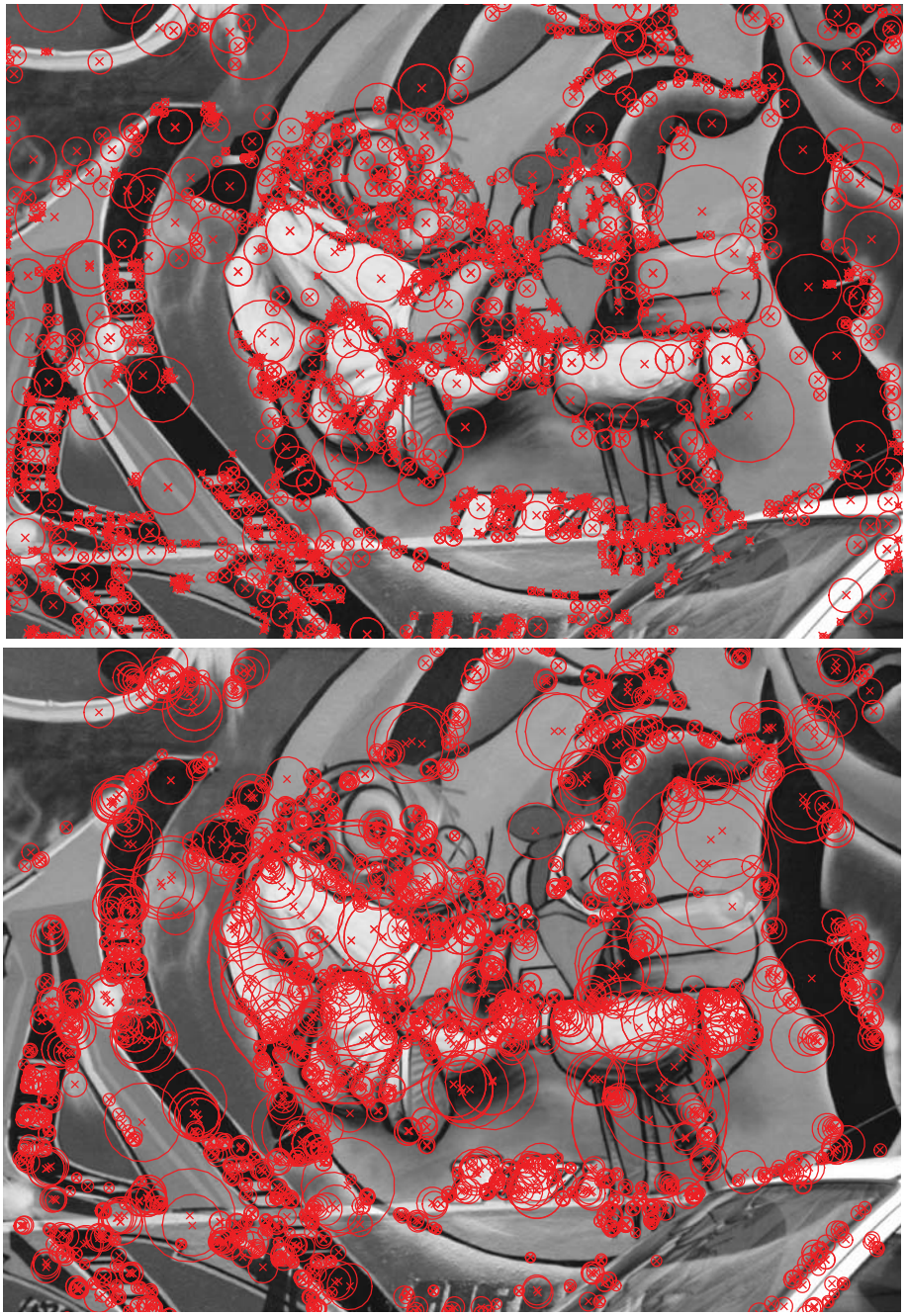


Figure 2.19: Example of interest points detected using our blob and corner detector (top) and Hessian-Laplace detector (bottom) on an image of the graffiti sequence. Our detector locates 2582 interest points ($\approx 50\%$ corners) and the Hessian-Laplace detector 2477 interest points. In pictures interest point locations are depicted as red crosses and treir scales as red circles.

2.7.3 Over-representation

The over-representation measures whether each local image structure is represented by more than one interest points. We propose the self-repeatability measure, which provides a measure of the redundancy and sparsity of the detected points. This measure is computed as the repeatability measure (2.23), but now between interest points detected on the same image, preventing one interest point can be related with itself. We consider that local image structures are over-represented when the overlapping error of near interest points is lower than 30%.

Scale invariant evaluation : In Figure 2.19 the results of our full scale invariant detector, including corners and blobs, and the results of Hessian-Laplace detector are depicted. In these images is easy to verify that Hessian-Laplace over-represent local image structures while our detector tends to avoid representing image structures with more than one interest point. However, the self-repeatability measure of both detectors on this image is similar, ours 21% and Hessian-Laplace 30%. This is due to many corners are detected close to blobs and the self-repeatability considers them as redundant. In Table 2.4 the self-repeatability for all the image sequences and detectors is provided, showing that our corner and blob detectors clearly avoid over-represent image structures providing non-redundant interest points. In the case of our full detector, as in the Figure 2.19, the self-repeatability is significantly increased because of corner and blob structures are highly related as we explain in Section 2.4.5.

	viewpoint		Image blur		zoom+rotation		JPEG	light	overall
	graffiti	wall	bikes	trees	boat	bark	house	cars	
Our Full	20.88	23.71	8.46	24.62	31.29	14.90	31.40	23.81	22.38
Our Blob	**8.02**	10.54	2.34	11.61	17.07	**3.45**	15.39	**9.90**	9.79
Our Corner	8.31	**9.65**	**2.11**	**10.02**	**16.73**	3.65	**15.12**	11.21	**9.60**
Hess-Lap	28.63	20.47	37.18	38.23	30.27	24.13	32.09	36.87	30.98
Harr-Lap	26.44	23.19	26.32	30.65	29.43	23.97	32.04	31.43	27.93
DoG	35.83	26.53	24.05	35.28	36.93	26.94	32.40	40.32	32.28

Table 2.4: Ratio of image regions over-represented by the scale invariant detectors for each evaluation sequence. The minimum ratio for each sequence is shown in bold. Self-repeatability has been computed with an overlapping error $< 30\%$.

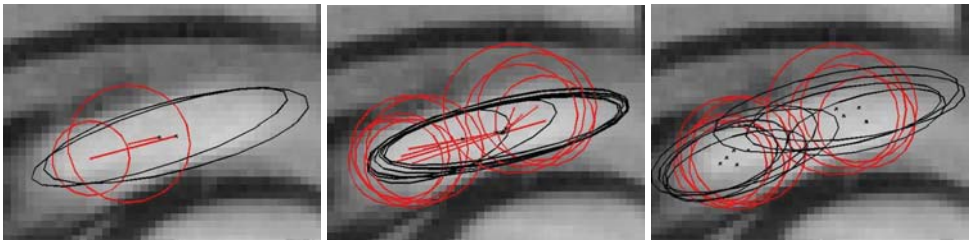
Affine invariant evaluation : In Table 2.5 the self-repeatability measure for the affine invariant detectors is shown. Our affine blob detector and the MSER detector provide the lowest over-representation rates. The overall evaluation show that our blob detector provides the best results being 45% lower than the MSER rate. Thus, our detector allows to improve the spatial distribution of the interest points and increasing their distinctiveness.

Figure 2.20 shows how our shape adaptation algorithm fits blobs on real images. Moreover, Figure also shows that our detector perceptually converges better on real blob structures than the Hessian-Affine detector, avoiding to detect several interest points in order to represent the same local image structure.

Scale invariant versus affine invariant detectors : In both evaluations our non-redundant interest point detectors provide the lowest rates in terms of self-repeatability. In the case of

	viewpoint		Image blur		zoom+rotation		JPEG	light	overall
	graffiti	wall	bikes	trees	boat	bark	house	cars	
Our Affine	**7.86**	4.47	**1.88**	**5.72**	8.48	4.02	7.71	**5.86**	**5.75**
MSER	13.34	**3.44**	34.84	11.91	**3.55**	**3.72**	**3.55**	9.60	10.49
Hes-Aff	50.60	41.12	53.63	52.01	48.03	49.06	49.75	48.42	49.08
Gradient	17.05	17.91	15.00	17.39	20.46	16.83	22.67	19.44	18.34

Table 2.5: Ratio of image regions over-represented by the affine invariant detectors for each evaluation sequence. The minimum ratio for each sequence is shown in bold. Self-repeatability has been computed with an overlapping error $< 30\%$.



(a) (b) (c)

Figure 2.20: Example of accuracy on a real image. In (a) and (b) affine invariant blobs found by our shape adaptation algorithm converge to the real blob: (a) initialized by our proposed scale invariant blob detector and (b) initialized by the Hessian-Laplace detector. In (c), blobs found by the Hessian-Affine detector are shown, which do not converge to real blobs. Red circles are the initially scale invariant detected blobs and black ellipses are the shape-adapted blobs.

scale invariant detectors, in the overall evaluation our rate is up to three times lower than the other detectors. For the affine invariant blob detectors our method provides 2 times lower rate than MSER detector. Specifically, our affine blob detector, which is initialized with the interest points detected by our scale invariant blob detector, reduces the overall self-repeatability from 9.79 to 5.75. This reduction is because of convergence is not achieved, as we explain at the end of Section 2.6.1, or because of more than one affine blob exactly represent the same image region, then only one of them is maintained as interest point, suppressing the others. In contrast, although Hessian-Affine detector uses Hessian-Laplace as initial detector the self-repeatability between both cannot be compared. The parameters used inside the Hessian-Affine detector are not explicitly explained so we have used standard parametrizations[1].

2.7.4 Homography Estimation

In order to evaluate the accuracy and computational cost of the matching process in a real task, we have registered the 8 image sequences provided by the dataset. Evaluation has been done by comparing the ground truth homographies H, which are also provided by the

[1]parameters were obtained from `http://www.robots.ox.ac.uk/~vgg/research/affine/`

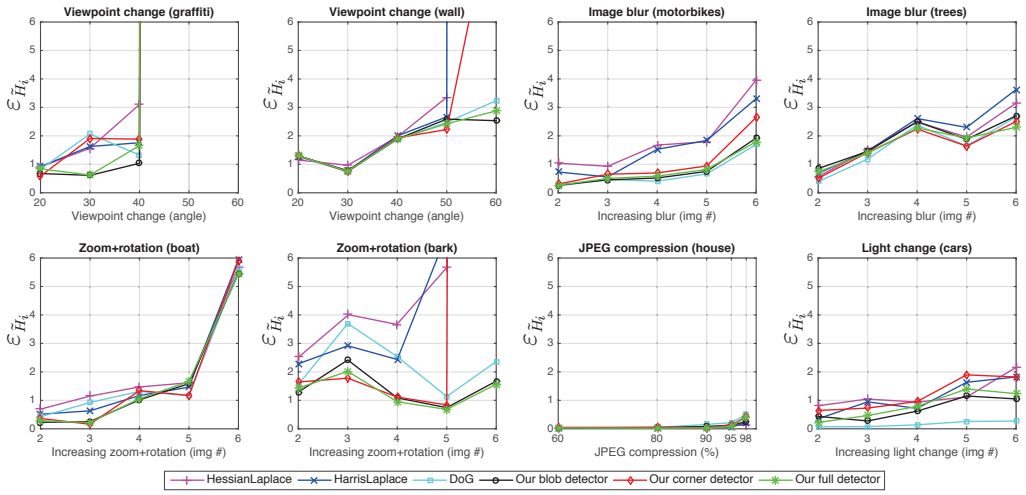


Figure 2.21: Comparison of scale invariant detectors in terms of $\varepsilon_{\tilde{H}_i}$ from evaluation sequences. $\varepsilon_{\tilde{H}_i}$ for each detector and sequence has been computed as the $median(\varepsilon_{\tilde{H}_i})$ of 10 Ransac-based homography estimations.

dataset, with the set of estimated homographies $\tilde{H}$. $\tilde{H}$ are estimated by applying 10 times the Ransac-based image registration algorithm provided by Hartley and Zisserman in [46] (using a distance threshold $t = 0.001$). In order to increase the robustness of the registration we only consider matches in which the nearest neighbor is less than 0.8 times the distance (computed in an euclidean sense in the SIFT feature space) to the best matching, similarly to Lowe in [84].

Image registration is evaluated by means of the error, denoted by $\varepsilon_{\tilde{H}}$, between each pair of homographies H and $\tilde{H}$. $\varepsilon_{\tilde{H}}$ is measured for each image pair and for each detector i as,

$$\varepsilon_{\tilde{H}_i} = \frac{1}{n}\sqrt{\sum_{j}^{n}\left(x_j - \tilde{H}_i(x'_j)\right)^2} \quad , \quad x_j = H_i(x'_j) \tag{2.26}$$

where x_j are the pixel coordinates belonging to the first image, which correspond with x'_j in the second image. Finally, from the set of 10 homographies estimated using Ransac for each image pair and detector we have selected only one $\tilde{H}_i$. Being $\tilde{H}_i$ selected in function of the median error ($median(\varepsilon_{\tilde{H}_i})$).

Scale invariant evaluation : Figure 2.21 shows a comparison in terms of $median(\varepsilon_{\tilde{H}_i})$ after 10 Ransac-based homography estimations for each image pair and scale invariant detector. All the evaluated scale invariant detectors provide similar errors, however our detectors tend to be between the ones providing the lowest $\varepsilon_{\tilde{H}_i}$. In the wall and the bark sequences our full and blob detectors allow to find a solution in all the variations while Hessian-Laplace, Harris-Laplace and our corner detectors do not find a solution for large variations.

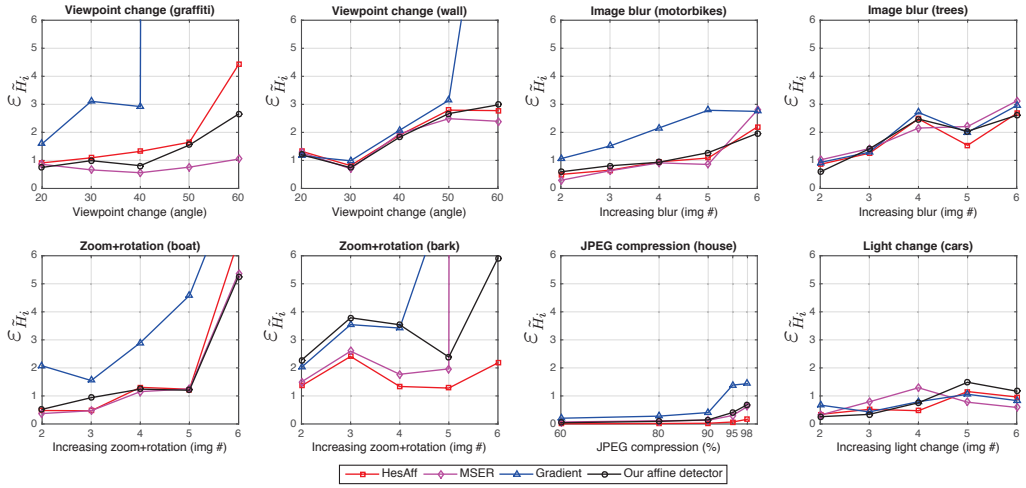


Figure 2.22: Comparison of scale invariant detectors in terms of $\varepsilon_{\tilde{H}_i}$ from evaluation sequences. $\varepsilon_{\tilde{H}_i}$ for each detector and sequence has been computed as the $median(\varepsilon_{\tilde{H}_i})$ of 10 Ransac-based homography estimations.

First column in Table 2.6 shows the mean number of features detected by each scale invariant detector, showing that our blob and corner detectors with less interest points provide comparable or even better results than the others with a fraction of their cost. The number of interest points detected by our full detector is the sum of the interest points found using our blob and corner detectors and its median cost is computed assuming that interest point typology is unknown. Even so its $\varepsilon_{\tilde{H}_i}$ are not worse than the errors for our blob and corner detectors, that means that our corner interest points do not tend to be matched with our blob interest points and conversely. On the other case, when blobs are only matched with blobs and corners with corners the computational cost of our full detector decreases up to ≈ 10 seconds, dividing its cost by 3.

	mean number of features x image	median cost x image pair (in secs)
Our blob	1489	5,39
Our corner	1283	4,93
Our full	2773	33,28
Hessian-Laplace	2018	18,46
Harris-Laplace	2241	20,12
DoG	3464	57,08

Table 2.6: Comparison of scale invariant detectors in terms of computational cost of the matching process for the registration task assuming interest points were previously extracted. The first column shows the mean number of features detected on each dataset image. The second column shows the computational cost required to match interest points and estimate each homography for each image pair.

Affine invariant evaluation : As in the scale invariant case, Figure 2.22 shows a comparison in terms of $median(\varepsilon_{\tilde{H}_i})$ for each affine invariant detector. In the Figure can be seen that our affine blob detector, the MSER and the Hessian-Affine detectors provide quite similar estimations for $\tilde{H}$, while the gradient based detector provides the worst results, even not finding a solution with acceptable error in some cases.

	mean number of features x image	median cost x image pair (in secs)
Our affine	1023	2,97
MSER	1270	5,11
Hessian-Affine	1987	17,56
Gradient	793	1,93

Table 2.7: Comparison of affine invariant detectors in terms of computational cost of the matching process for the registration task assuming interest points were previously extracted. The first column shows the mean number of features detected on each dataset image. The second column shows the computational cost required to match interest points and estimate each homography for each image pair.

Table 2.7 shows that with less interest points and computational cost our affine blob detector allows to compute homographies with similar $\varepsilon_{\tilde{H}_i}$. Qualitatively, Figure 2.23 shows some examples of registered image pairs using our affine blob detector under very high variations.

Scale invariant versus affine invariant detectors : Comparing Figures 2.21 and 2.22 can be seen that the main difference between them is related with the error in the viewpoint change sequences, where affine interest points show its ability to deal with large viewpoint changes. Surprisingly, while Hessian-Affine tends to reduce its error with respect to Hessian-Laplace detector in almost all the sequences, our affine blob detector tends to slightly increase the errors except in the viewpoint change sequences. We think that this behaviour is due to the reduction, in terms of quantity, of the interest points carried out by our affine detector. Anyway our affine detector has comparable results to the Hessian-Affine detector.

Tables 2.6 and 2.7 show that the computational cost is superlinear in front of the number of interest points, so detectors providing less and more discriminative interest points allow to speed up the homography estimation. Combining these detectors with approximated nearest neighbours matching methods, such as [99], could speed up the process in exchange for a small reduction in accuracy.

2.8 Summary

In this chapter we have presented a powerful set of tools allowing to find simultaneously scale invariant corners and blobs, and also affine invariant blobs. All these detectors are specifically designed to maximize the geometrical invariance and distinctiveness properties. Distinctiveness property is satisfied avoiding the over-detection of local image structures and so, obtaining non-redundant and sparse interest points. On the other hand, in the case of scale

Figure 2.23: Examples of our affine blob detector for image registration tasks on image pairs with large variations. Green rectangles represent the initial image registered using the correct homography H and yellow rectangles using $\tilde{H}$). In the first row there are two examples of image pairs with high variations in scale and rotation $\varepsilon_{\tilde{H}} = 0.67$ and $\varepsilon_{\tilde{H}} = 2.70$. In the second row, the first image pair suffers large variations in the point of view ($\varepsilon_{\tilde{H}} = 4.35$) and the second image pair suffers a large degradation due to compression ($\varepsilon_{\tilde{H}} = 0.55$) .

invariant detectors geometrical invariance is satisfied extracting interest points from a multi-scale representation using Gaussian curvature as operator. In the case of our affine invariant blob detector geometrical invariance is satisfied by the proposed shape adaptation algorithm, which simultaneously fits the shape and the spatial location of local image structures.

The experimental evaluation shows for all our detectors a comparable performance to the state of the art in terms of repeatability and matching scores, while in terms of precision and recall our detectors outperform or obtain similar scores. In addition, experiments show our detectors provide sparse interest points, avoiding redundant detections that over-represent specific image regions. Therefore, our detector improves the spatial distribution of the interest points and increases their distinctiveness, allowing to decrease the computational cost of future matching tasks. Finally, we have shown our scale invariant detectors obtain slightly better accuracy in image registration tasks than other detectors while our affine invariant detector obtain similar accuracy.

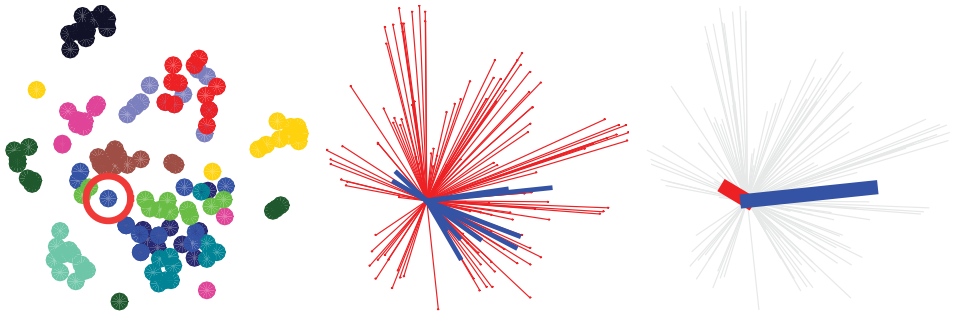
Chapter 3

Learning of Discriminative Local Image Descriptors

Many computer vision applications still rely on hand-crafted local image descriptors such as SIFT. However, for image-level tasks such as classification, Deep learning has become the dominant paradigm in detriment of hand-crafted ones. In this chapter we propose a novel framework based on learning of discriminant descriptors from local image regions using Convolutional Neural Networks (CNNs). In particular we propose training a Siamese network architecture using pairs of corresponding and non-corresponding image regions. We deal with the large number of potential pairs with the combination of a stochastic sampling of the training set and an aggressive mining strategy biased towards regions that are hard to classify.
We thoroughly evaluate multiple architectures, configurations and hyper-parameters over different datasets. Our models are fully convolutional, can be computed densely in an efficient manner, and are amenable to modern GPUs. By using the L_2 distance during both, training and testing, we develop 128-D descriptors whose euclidean distances reflect region similarity, and which can be used as a drop-in replacement for any task involving SIFT. We demonstrate consistent performance gains over the state of the art on various tasks, and generalize well against scaling and rotation, perspective transformation, non-rigid deformation, and illumination changes.

3.1 Introduction

The previous chapter is dedicated to find interest local image regions, in this one we study invariant and discriminative representations for those local image regions in order to improve their ability to be recognized. Representing local image regions in an invariant and discriminative manner is a major research topic in Computer Vision. The aim of these representations consist in obtain robust descriptors to image perturbations and changes in the viewing conditions, such as noise, blurring, illumination changes or perspective changes.



(a) Data (b) All pairs (c) "Hard" pairs

Figure 3.1: To train models with Siamese networks, we need pairs of corresponding and non-corresponding samples. (a) We use t-SNE [137] to display $\sim$ 100 64×64 patches of 12 3D points from different images (see Figure 3.5 for examples). Corresponding regions are drawn with the same color. (b) We single out the red-circled patch, belonging to the blue point cloud, and consider all of its potential pairings. The line length encodes the closeness between this region and the rest: positive matches in blue, negative in red. Most pairs are easy to discriminate and ineffectual for training. (c) We mine the samples to obtain the closest negative (shortest red line) and the most distant positive (longest blue line). This simple strategy allows us to train discriminative networks over large datasets.

The field reached maturity with SIFT [84], and has since become the cornerstone of a wide range of applications of Computer Vision as object recognition [84, 135], image registration [91] or 3D reconstruction [1]. Similarly to SIFT, which is based on spatially distributed histograms of gradients, most descriptors rely on hand-crafted features [3, 65, 84, 121, 131, 133, 142] trying with more or less success provide simultaneously invariance and discriminatory power.

There has recently been interest in using machine learning techniques to learn descriptors from large datasets [113, 122, 134]. In this case, descriptors are learnt from sets of local image regions, which represent under different viewing conditions specific image regions. The goal is to automatically capture the similarities and differences between those local image regions in order to maximize the invariance and discriminative power. These descriptors clearly outperform the hand-crafted ones [122], since they are able to deal with very difficult issues such as: spatial location errors, in-plane perspective variations or even those perspective variations produced by image regions that are not planar in the real 3D scene and can even be partially affected by self-occlusions.

In this chapter we draw inspiration from the recent success of Deep Convolutional Neural Networks (CNNs) in large-scale image classification problems [66, 129] to learn discriminative descriptors for local image regions. However, in our case discriminative training does not rely on labels of individual regions, but rather on pairs of corresponding, or non-corresponding regions. For this reason we propose a Siamese network architecture [8] that employs two CNNs with identical parameters to compare pairs of image regions. Each pair of image regions is propagated forward through the network giving two different CNN outputs, which are treated as region descriptors. Then we minimize a loss function that enforces the L_2 norm between both descriptors to be small for corresponding regions and large otherwise.

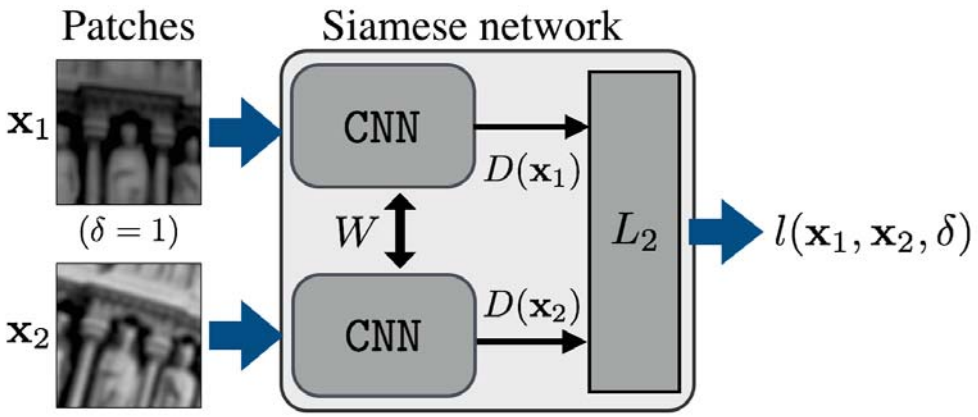


Figure 3.2: Schematic of a Siamese network, where pairs of input patches are processed by two copies of the same CNN, sharing all the internal parameters.

In Figure 3.2 a schematic view of the proposed approach is shown.

To train this network we rely on the multi-view stereo dataset (MVS) [9], which contains over 1.5M grayscale 64×64 image regions from different views of 500K 3D points. The difficulty with such a large dataset is that it becomes impossible to exhaustively explore all corresponding and non-corresponding pairs, so we must resort to some form of random sampling. Based on the observation that after a certain point of learning most pairs are correctly classified, and using them no longer improves the learned embedding, we propose a strategy of aggressive mining of "hard" positives and negatives. During the learning stage we enforce the back-propagation of samples with a large loss, i.e. corresponding pairs that match poorly and non-corresponding pairs that are hard to discriminate (see Figure 3.1 for a graphical interpretation). This fact proves to be most useful for efficiently learning discriminative descriptors.

Since we use the L_2 distance to compare descriptors, rather than some nonlinear, task-specific metric, as e.g. in [42, 147], we believe that the resulting descriptors can be used as drop-in replacement for popular representations such as SIFT, in a manner agnostic to the application. Furthermore, as our descriptors are primarily built from convolutions they are very efficient to compute and can be easily parallelized, taking advantage of modern GPUs to greatly speed up their extraction.

3.1.1 Overview

This chapter is organized as follows. In the next section recent CNN-based descriptors are introduced. Section 3.3 explains the fundamentals required to understand Neural Networks and CNNs. In Section 3.4 we present our CNN-based descriptors taking advantage of Siamese networks and large sets of correspondences. Finally, in the last section we perform in-depth comparisons against both traditional, hand-crafted descriptors [84, 131, 121] as well

as learned, state-of-the-art descriptors [122, 134], using Precision/Recall (PR) and its Area Under the Curve (AUC) as a metric, and demonstrate consistent gains in performance. We also show that our descriptors generalize very well to applications for which they were not specifically trained, demonstrating remarkable robustness against scaling, rotation, viewpoint changes, non-rigid deformations, and varying illumination.

3.2 Related CNN-Based Approaches

Recent developments in the design of local image repesentations are moving away from carefully-engineered descriptors [3, 84, 131] and towards learning descriptors from large volumes of data. This line of works includes unsupervised techniques based on hashing as well as supervised approaches using Linear Discriminant Analysis [9, 39, 125], boosting [134], and convex optimization [122].

In this section we explore solutions based on Convolutional Neural Networks (CNNs), which currently are the dominant paradigm in tasks involving semantic information, e.g. image classification [66, 129] or semantic segmentation [83, 12]. Even though it may be unclear whether CNNs are equally appropriate for applications based on local image regions where semantic information may be missing, we argue that for our particular problem this is indeed the case.

Descriptor learning using CNNs was addressed early in [50, 104], but the experimental results in these works left open questions regarding several practical aspects, such as the most appropriate network architectures and application-dependent training schemes. We aim to provide a rigorous analysis of several of these topics. More recently, the use of Siamese networks for descriptor learning was exploited by concurrent works on joint descriptor and metric learning [42, 146, 147]. Han et al. [42] use a Deep Convolutional Neural Network in a Siamese architecture followed by a fully-connected network that learns a comparison function. Zagoruyko et al. [146] rely on a similar architecture but add a network that only focuses on the center of the image, which they show increases performance, at a computational cost. Zbontar & LeCun [147] trained CNNs for narrow-baseline stereo and obtained the top results on the KITTI benchmark [36]. These approaches rely on larger networks and do not necessarily learn compact, discriminative representations, like ours. In contrast, we show how to exploit discriminative training strategies to build small but powerful models.

One key distinction between [42, 146] and our work is that we aim at using the CNN outputs of our Siamese networks in place of traditional descriptors in existing pipelines. Unlike [42, 146, 147] there is no non-linear 'metric network' following the Siamese network application, but rather we simply use the L_2 distance to compare patches. In [146] a limited evaluation of L_2-based similarity shows promising results, which however is not entirely clearly outperforming [122]. Instead we show substantial gains, which can be also attributed to using the L_2 distance during training. Finally, using descriptors that can be compared with the L_2 distance facilitates the use of efficient methods for nearest neighbour computations, such as KD-trees, which we believe opens up the path to large-scale retrieval applications.

Another deviation of our work from common practice is that we observe that during de-

scriptor training the majority of non-corresponding region pairs eventually become easy to discern, which hinders the learning of discriminative models. Mining hard negatives is a well-known procedure in the context of sliding-window detectors [30], where the number of negative samples (windows) is virtually unlimited and yet most negatives are easily discriminated once we have already used a certain number of negative samples for training. Similar techniques have been applied to CNNs for object detection [37, 129]. In this chapter we demonstrate that aggressive mining of both "hard" positive and negative samples greatly enhances the learning process: as we detail in the Section 3.4, we sample a large number of matches and use the subset with the largest loss to update the network.

3.3 Introduction to Artificial Neural Networks and CNNs

Artificial Neural Networks (ANNs) were inspired by biological neural networks during the 40s. From that moment ANNs have evolved up to the current concepts of Deep Network where many layers are added in order to improve the learning of unknown functions described by input data, or Deep Convolutional Network where convolutional layers are used in order to reduce the number of parameters. During the last decade thanks to a radical increase in the available computational power, allowing to train networks with millions of parameters, their usage has exploded in the computer vision field showing very good performance in tasks as object or image classification.

An ANN consist in a set of inter-connected layers containing nodes, so called "neurons". In this section we consider ANNs defined by a directed acyclic graph where nodes are connected with all the nodes in the previous and next layers, these ANNs are commonly called feedforward neural networks. The output a_i^l of the node i in the layer l is related to the outputs in the layer $l-1$ by,

$$a_i^l = \sigma\left(\sum_k w_{ik}^l a_k^{l-1} + b_i^l\right) \tag{3.1}$$

where σ denotes a non-linear activation function, b^l a bias term and w_{ik}^l a weight for the connection with the node k of the layer $l-1$. This equation can be written in a matrix form denoting all the nodes in each layer as,

$$\mathbf{a}^l = \sigma\left(\left(\mathbf{W}^l\right)^T \mathbf{a}^{l-1} + \mathbf{b}^l\right) \tag{3.2}$$

where each column in $\mathbf{W}$ encodes all the input weights for a given node, $\mathbf{a}^{l-1}$ all the outputs of the previous layer and $\mathbf{b}^l$ all the biases for the nodes in layer l. Typically, the activation function σ was defined by the sigmoid function or hyperbolic tangent, however the Rectified Linear Unit (ReLU), $f(z) = \max(0, z)$, is the most popular activation function for Deep Networks [68].

3.3.1 Back-propagation Procedure

During the network training a large number of weights w_{ik}^l and biases b_i^l must be learnt. The back-propagation procedure [114] allows to iteratively adjust the weights and biases in the network so as to minimize a cost function C, which is an average over a loss function $L(\mathbf{a}^L, \mathbf{y})$ relating the output vector of the last layer $\mathbf{a}^L$ with the desired output $\mathbf{y}$. Therefore, the goal of back-propagation is to compute the partial derivatives $\frac{\partial C}{\partial \mathbf{w}}$ and $\frac{\partial C}{\partial \mathbf{b}}$ as an averaging of $\frac{\partial L}{\delta \mathbf{w}}$ and $\frac{\partial L}{\partial \mathbf{b}}$ over training samples. To compute those partial derivatives back-propagation provides a procedure to compute the error δ_i^l for each node in the network. First of all the errors $\boldsymbol{\delta}^L$ in the output layer are computed,

$$\boldsymbol{\delta}^L = \frac{\partial C}{\partial \mathbf{a}^L} \odot \sigma' \left(\left(\mathbf{W}^L \right)^T \mathbf{a}^{L-1} + \mathbf{b}^L \right) \tag{3.3}$$

where $\odot$ denotes the Hadamard product and σ' the derivative of the activation function. Afterwards, the errors $\boldsymbol{\delta}^l$ can be computed in terms of the errors in the next layer $\boldsymbol{\delta}^{l+1}$,

$$\boldsymbol{\delta}^l = \left(\left(\mathbf{W}^{l+1} \right)^T \boldsymbol{\delta}^{l+1} \right) \odot \sigma' \left(\left(\mathbf{W}^l \right)^T \mathbf{a}^{l-1} + \mathbf{b}^l \right) \tag{3.4}$$

Thus, reapplying this equation for all the layers from layer $L-1$ to layer 2 all the errors are estimated and partial derivatives for each node can be computed,

$$\frac{\partial C}{\partial w_{ik}^l} = a_k^{l-1} \delta_i^l \tag{3.5}$$

$$\frac{\partial C}{\partial b_i^l} = \delta_i^l \tag{3.6}$$

3.3.2 Learning Procedure

In summary, back-propagation first of all given a set of training samples apply a forward pass using the equation (3.2) obtaining an output vector $\mathbf{a}^L$ for each sample. Afterwards, the output error $L(\mathbf{a}^L, \mathbf{y})$ for each sample is computed and used to perform a backward pass propagating the error through the network using equations (3.3) and (3.4). Finally, all the partial derivatives are computed using equations (3.5) and (3.6) in order to update the weights and biases terms using the gradient descent learning rule,

$$w_{ik}^l \rightarrow w_{ik}^l - \eta \frac{\partial C}{\partial w_{ik}^l} \tag{3.7}$$

$$b_i^l \rightarrow b_i^l - \eta \frac{\partial C}{\partial b_i^l} \tag{3.8}$$

where η is the learning rate. All this procedure must be iterated until convergence is achieved.

A common method to accelerate the gradient descent convergence is the momentum method [109], this method accumulates the gradient directions looking for directions reducing persistently the error across iterations,

$$\mathbf{v} \rightarrow \mu\mathbf{v} - \eta\frac{\partial C}{\partial \boldsymbol{\theta}} \tag{3.9}$$

$$\boldsymbol{\theta} \rightarrow \boldsymbol{\theta} + \mathbf{v} \tag{3.10}$$

where μ denotes the momentum coefficient, $\mathbf{v}$ the accumulated gradients and $\boldsymbol{\theta}$ all the weights and biases.

3.3.3 Minibatch Stochastic Gradient Descent

Exploiting that C is an average over $L(\mathbf{a}^L, \mathbf{y})$, a fast method to update the weights and biases in equations (3.7) and (3.8) relies in using stochastic gradient descent approaches to compute partial derivatives in equations (3.5) and (3.5). These approaches are specially useful for large datasets. Concretely, the usual approach to optimize networks is the minibatch stochastic gradient descent method [5], which updates the weights and biases by taking the average over a small batch of training samples,

$$w_{ik}^l \rightarrow w_{ik}^l - \frac{\eta}{m}\sum_x \frac{\partial L(\mathbf{a}_x^L, \mathbf{y}_x)}{\partial w_{ik}^l} \tag{3.11}$$

$$b_i^l \rightarrow b_i^l - \frac{\eta}{m}\sum_x \frac{\partial L(\mathbf{a}_x^L, \mathbf{y}_x)}{\partial b_i^l} \tag{3.12}$$

where m is the number if samples in the batch and $L(\mathbf{a}_x^L, \mathbf{y}_x)$ denotes the error with respect to the desired output for the sample x.

3.3.4 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a specialized type of neural network for processing data with a known grid-like topology, such us images. CNNs architecture allows fast training with fewer parameters, in addition from a Computer Vision point of view allows some degree of shift, scale and distortion invariance on the image regions [69]. These networks combine three architectural ideas:

- **Local receptive fields:** Convolutional layers instead of using fully connected layers make connections in small and local regions, i.e. one node in a layer l will be only connected with a reduced set of nodes in layer $l-1$. These regions are called local receptive fields and they are highly overlapped along neighbouring nodes in layer l.

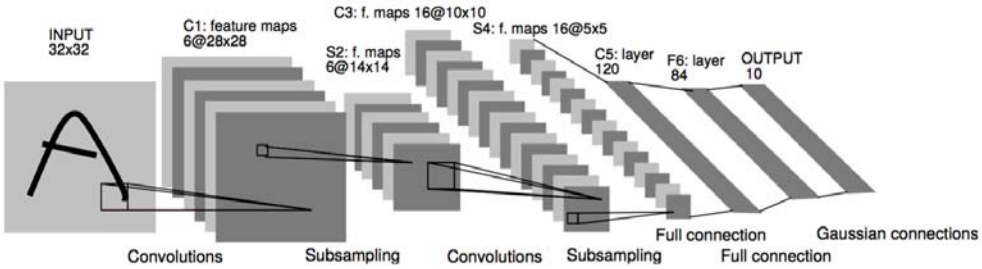


Figure 3.3: Example of a Convolutional Neural Network (CNN) for digits recognition. Convolutional and subsampling layers have associated a label, $a@b \times c$, where a denotes the number of feature maps and $b \times c$ the number of nodes for each feature map. The local receptive fields are shown as small black squares. Figure reproduced from [69].

- **Shared weights and biases:** Each node in a convolutional layer l has a bias and weigths relating the node with its underlying local receptive field. Unlike in the ANNs where all the weights and biases are independent in the convolutional layers those parameters are shared with all the nodes in the layer. Therefore, assuming two dimensional fields the output $a^l_{i,j}$ for a node with coordinates (i, j) can be written as,

$$a^l_{i,j} = \sigma\left(b^l + \sum_u \sum_v w^l_{u,v} a^{l-1}_{i+u,j+v}\right) \tag{3.13}$$

  where σ denotes the activation function, (u, v) the local receptive field coordinates and $w^l_{u,v}$ and b^l the shared weights and bias, respectively. The name of convolutional layers come from the fact that the operation in this equation is also known as convolution. In addition, each local receptive field can be related with more than one set of shared weights and bias, being their outputs named as feature maps.

- **Subsampling:** Besides of convolutional layers there are also subsampling layers, also known as pooling layers. These layers are usually placed after convolutional layers and they are useful to simplify the outputs of convolutional layers, summarizing local region of outputs to a single output. A common procedure is known as max-pooling where each local region is summarized by its maximum output. Another common approach is the L_2 pooling where each local region is summarized by the square root of the sum of the squares of their outputs.

After the convolutional and subsampling layers, fully connected layers can be used losing the spatial information maintained by the previous layers. Figure 3.3 shows an example combining these three architectural ideas with fully-connected layers.

Since CNNs are not fully-connected and use subsampling layers their training requires straightforward modifications to the backpropagation procedure [100]. CNNs tend to have many layers and parameters, for this reason they require a large amount of samples for the training. In order to automatically increase the training dataset the most common method consists on data augmentation, which is based on apply synthetic deformations on the training

samples, such as adding noise, cropping, or rotating. Data augmentation has shown to help the networks to reduce overfitting on training samples and generalize better [66].

3.4 Learning Deep Descriptors

Given an intensity image region $\mathbf{x} \in \mathbb{R}^d$, the descriptor of $\mathbf{x}$ is a non-linear mapping $D(\mathbf{x})$ that is expected to be discriminative, i.e. descriptors for image patches corresponding to the same point should be similar, and dissimilar otherwise.

In the context of multiple-view geometry, descriptors are typically computed for relevant points where scale and orientation can be reliably estimated, for invariance. Image regions around those relevant points then capture local projections of 3D scenes. Let us consider that each image patch $\mathbf{x}_i$ has an index p_i that uniquely identifies the 3D point which roughly projects onto the 2D patch, from a specific viewpoint. Therefore, taking the L_2 norm as a similarity metric between descriptors, for an ideal descriptor we would wish that

$$d_D(\mathbf{x}_1, \mathbf{x}_2) = \|D(\mathbf{x}_1) - D(\mathbf{x}_2)\|_2 = \begin{cases} 0 & \text{if } p_1 = p_2 \\ \infty & \text{if } p_1 \neq p_2 \end{cases} \tag{3.14}$$

We propose learning descriptors with a Siamese network [8], where a non-linear mapping is represented by a CNN that is optimized for pairs of corresponding or non-corresponding image regions, as shown in Figure 3.2. We propagate the regions through the model to extract the descriptors and then compute their L_2 norm, which is a standard similarity measure for image descriptors. We then compute the loss function on this distance. Given a pair of regions $\mathbf{x}_1$ and $\mathbf{x}_2$ we define a loss function of the form

$$l(\mathbf{x}_1, \mathbf{x}_2, \delta) = \delta l_P(d_D(\mathbf{x}_1, \mathbf{x}_2)) + (1 - \delta) l_N(d_D(\mathbf{x}_1, \mathbf{x}_2)) \tag{3.15}$$

where δ is the indicator function, which is 1 if $p_1 = p_2$, and 0 otherwise. l_P and l_N are the partial loss functions for patches corresponding to the same 3D point and to different points, respectively. When performing back-propagation [114], the gradients are independently accumulated for both descriptors, but jointly applied to the weights, as they are shared.

We relax the ideal loss for equation (3.14) by using a margin m for $l_N(\cdot)$. In particular, we consider the hinge embedding criterion [93],

$$l_P(d_D(\mathbf{x}_1, \mathbf{x}_2)) = d_D(\mathbf{x}_1, \mathbf{x}_2) \tag{3.16}$$

$$l_N(d_D(\mathbf{x}_1, \mathbf{x}_2)) = \max(0, m - d_D(\mathbf{x}_1, \mathbf{x}_2)) \tag{3.17}$$

3.4.1 CNN-based Descriptors

When designing the structure of the CNN we are limited by the size of the input data: in our case 64×64 regions, from the MVS dataset [9], while we extract descriptors of the same size as SIFT [84], i.e. 128 dimensions. Note that larger regions and/or output spaces would allow

Name	Layer 1	Layer 2	Layer 3	Layer 4
CNN3_NN1	32x7x7	64x6x6	128x5x5	128
	x2 pool	x3 pool	x4 pool	-
CNN3	32x7x7	64x6x6	128x5x5	-
	x2 pool	x3 pool	x4 pool	-
CNN2a_NN1	32x5x5	64x5x5	128	-
	x3 pool	x4 pool	-	-
CNN2b_NN1	32x9x9	64x5x5	128	-
	x4 pool	x5 pool	-	-
CNN2	64x5x5	128x5x5	-	-
	x4 pool	x11 pool	-	-
CNN1_NN1	32x9x9	128	-	-
	x14 pool	-	-	-

Figure 3.4: Different CNN configurations evaluated to learn descriptors. Each layer is described as $axbxcxdpool$ denoting the number of feature maps a, the size of the local receptive fields bxc and the size of the pooling and normalization region d.

us to consider possibly more informative descriptors, but at the same time they would be also more susceptible to occlusions and slower to train and compute.

In our approach each convolutional layer comprises four steps: convolution by the shared weights, apply the non-linear activation function, pooling and normalization. Since sparser connectivity has been shown to improve performance while lowering parameters and increasing speed [18], except for the first layer, the shared weights are not densely connected to the previous layers. Instead, they are sparsely connected at random, so that the mean number of connections each input layer has is constant. Each set of shared weights of the second and third layer are also connected randomly to 8 feature maps of the previous layer so that the mean number of connections stays roughly equal to 16 connections per output.

Regarding the non-linear activation function, we use hyperbolic tangent units (Tanh), as we found it to perform better than Rectified Linear Units (ReLU). We use L_2 pooling, which has been shown to outperform the more standard max-pooling [118]. Normalization is also important for deep networks [51] and fundamental for descriptors [89]. We use subtractive normalization, i.e. subtract the weighted average over a 5×5 neighbourhood with a Gaussian kernel after the first and second layers.

An overview of the architectures we consider is given in Figure 3.4. The network depth is constrained by the size of the image region (64x64). We consider only up to 3 convolutional layers. Additionally, we consider adding a single fully-connected layer at the end. We choose a set of six networks, from 2 up to 4 layers. In Section 3.5 we will show that deeper networks outperform shallower ones, and architectures with a fully-connected layer at the end do worse than fully convolutional architectures.

3.4.2 Stochastic Sampling Strategy and Mining

Our goal is to optimize the network parameters from an arbitrarily large set of training image regions. Let us consider a dataset with k regions and $m \leq k$ unique 3D region indices, each with c_i corresponding image regions. Then, the number of matching image regions, $\mathcal{P}$ (positives) and the number of non-matching images regions, $\mathcal{N}$ (negatives) is,

$$\mathcal{P} = \sum_{i=1}^{m} \frac{c_i(c_i - 1)}{2} \quad \text{and} \quad \mathcal{N} = \sum_{i=1}^{m} c_i \left(k - c_i\right). \tag{3.18}$$

Since both $\mathcal{P}$ and $\mathcal{N}$ are intractably large, we resort to stochastic gradient descent, using random subsets of our training set to estimate the gradient of our loss function. For positives we can randomly sample a set of s_p 3D point indices from the set $\{p_1, \ldots, p_m\}$, and for each chosen 3D index p_i we randomly pick two 2D regions with corresponding 3D point indices.

For negatives one simple idea would be to randomly choose s_n random pairs with non-matching indices; but once the network has reached a reasonable level of performance, most non-corresponding points will already have a distance above C, contributing nothing to the loss and the gradient. This can result in a very small and noisy estimate of the gradient, effectively stalling the learning process.

Instead, we iterate over non-corresponding region pairs to search for "hard" negatives, i.e. pairs that are close in descriptor space and incur a high loss. In this manner it becomes feasible to train discriminative models faster while also increasing performance.

In particular, at each epoch we generate a set of s_n randomly chosen region pairs, and after forward-propagation through the network and computing their loss we keep only a subset of the s_n^H "hardest" negatives, which are back-propagated through the network in order to update the weights. Additionally, the same procedure can be used over the positive samples, i.e. we can sample s_p corresponding region pairs and prune them down to the s_p^H "hardest" positives. Our experimental results clearly show that the combination of aggressive mining for both positive and negative region pairs allows us to greatly improve the discriminative capability of our learned descriptors.

3.4.3 Learning

We normalize the dataset by the mean and standard deviation of the training patches. We then learn the weights with stochastic gradient descent. We use a learning rate that decreases by an order of magnitude every fixed number of iterations. Additionally, we use standard momentum in order to accelerate the learning process. We use a subset of the data for validation, and stop training when the metric we use to evaluate the learned models converges. Due to the exponentially large pool of positives and negatives available for training and the small number of parameters of our relatively small architectures, no techniques to cope with overfitting are used. The particulars of the learning procedure are detailed in the following section.

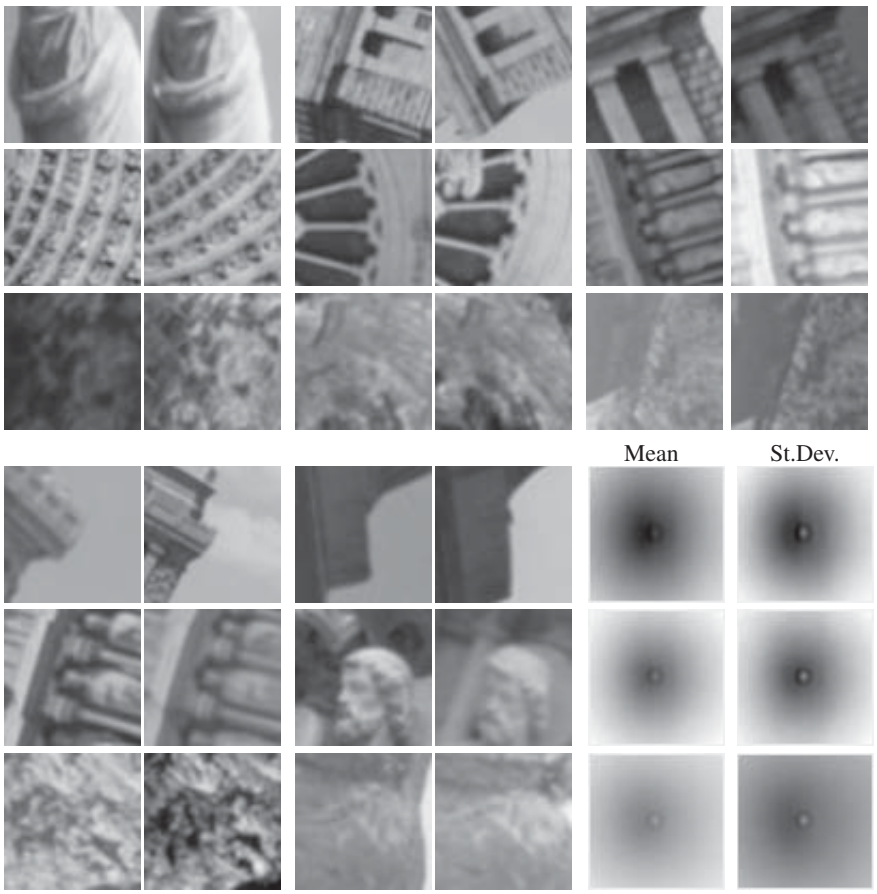


Figure 3.5: Pairs of corresponding samples from the MVS dataset. Top: 'Liberty' (LY). Middle: 'Notre Dame' (ND). Bottom: 'Yosemite' (YO). Right: we compute the absolute value of the pixel difference between corresponding regions on each set and show their mean/std (black corresponds to small values).

Note that extracting descriptors with the learned models does not further require the Siamese network, and does not incur the computational costs related to mining.

3.5 Experimental Evaluation

For training we use the Multi-view Stereo Correspondence dataset (MVS) [9], which consists of 64×64 grayscale image regions sampled from 3D reconstructions of the Statue of Liberty (LY), Notre Dame (ND) and Half Dome in Yosemite (YO). Regions are extracted using the Difference of Gaussians detector [84], and determined as a valid correspondence if they are within 5 pixels in position, 0.25 octaves in scale and $\pi/8$ radians in angle. Fig. 3.5 shows some samples from each set, which contain significant changes in position, rotation

and illumination conditions, and often exhibit very noticeable perspective changes.

We join the data from LY and YO to form a training set with over a million regions. Out of these we reserve a subset of 10,000 unique 3D points for validation ($\sim 30,000$ regions). The resulting training set contains 1,133,525 possible positive combinations and 1.117×10^{12} possible negative combinations. This skew is common in correspondence problems such as stereo or structure from motion; as stated before, we address it with aggressive mining. We use this split to evaluate different architectures and configurations, and then train the top-performing model over the two remaining splits.

A popular metric for classification systems is the Receiving Operator Characteristic (ROC), used e.g. in [9], which can be summarized by its Area Under the Curve (AUC). However, ROC curves can be misleading when the number of positive and negative samples are very different [19], and are already nearly saturated for the SIFT baseline. A richer performance indicator is the Precision/Recall curve (PR). We benchmark our models with PR curves and their AUC. In particular, we simulate the 'needle in a haystack' setting of retrieval by having a thousandfold more negative than positive pairs: for each of the 10,000 unique points in our validation set we generate a single positive pair, by randomly sampling two corresponding regions, and 1,000 non-corresponding regions, chosen from the remaining points.

Results outline: We explored multiple architectures and configurations in the two next sections. We study the effect of mining for "hard" samples in Sec. 3.5.3. We then evaluate our top-performing models over the test set in Sec. 3.5.4. To build a test set we follow the same procedure as for validation, evaluating 10,000 points with 1,000 negatives each, over 10 different folds (see Sec. 3.5.4 for details). We consider four splits: LY+YO (tested on ND), LY+ND (tested on YO), and YO+ND (tested on LY), plus a final split with training data from all three sets.

Finally, we apply the models learned over the MVS dataset to different applications. In Sec. 3.5.6 we study the robustness of our descriptors to patch rotation. In Sec. 3.5.7 we use our models to match wide-baseline images from a different stereo dataset. In Sec. 3.5.8 we benchmark our descriptors on a recent dataset with very challenging non-rigid deformations and drastic changes in illumination. Our models outperform state-of-the-art baselines in every case, without fine-tuning over new data, and over considerably different application domains.

We will consider all hyperparameters to be the same unless otherwise mentioned, i.e. a learning rate of 0.01 that decreases every $10,000$ iterations by a factor of 10, and a momentum of 0.9. Our implementation is based on Torch7 [16].

3.5.1 Depth and Fully Convolutional Architectures

An overview of the six architectures we consider is given in Table 3.4. The network depth is constrained by the size of the image regions. We consider only up to 3 convolutional layers (CNN$\{$1-3$\}$). Additionally, we consider adding a single fully-connected layer at the end (NN1). Fully-connected layers increase the number of parameters by a large factor, which increases the difficulty of learning and can lead to overfitting.

In this section we show the results of the six architectures in the Figure 3.6. Results show

Architecture	Parameters	PR AUC
SIFT	—	.361
CNN1_NN1	68,352	.032
CNN2	27,776	.379
CNN2a_NN1	145,088	.370
CNN2b_NN1	48,576	.439
CNN3_NN1	62,784	.289
CNN3	46,272	**.558**

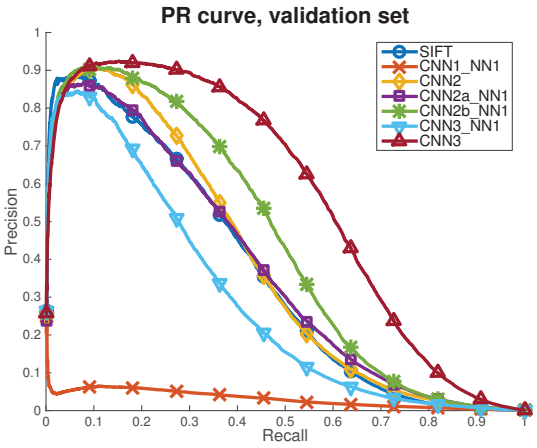


Figure 3.6: Precision-Recall results for the evaluated architectures. Effect of network depth (up to 3 CNN layers), and fully convolutional networks vs networks with final fully-connected layer (NN1). PR AUC on the validation set for the top-performing iteration.

that deeper networks outperform shallower ones, and architectures with a fully-connected layer at the end do worse than fully convolutional architectures. In the following experiments we consider only models with 3 convolutional layers.

3.5.2 Hidden Units Mapping, Normalization, and Pooling

It is generally accepted that Rectified Linear Units (ReLU) perform better in classification tasks [66] than other non-linear functions. They are, however, ill-suited for tasks with continuous outputs such as regression or the problem at hand, as they can only output positive values. We consider both the standard Tanh and ReLU. For the ReLU case we still use Tanh for the last layer. We also consider not using the normalization sublayer for each of the convolutional layers. Finally, we consider using max pooling rather than L_2 pooling. We show results for the fully-convolutional CNN3 architecture in Figure 3.7. The best results are obtained with Tanh, normalization and L_2 pooling ('CNN3' in the table/plot). We will use this configuration in the following experiments.

3.5.3 Mining

We analyze the effect of both positive and negative mining by training different models in which a large, initial pool of s_p positives and s_n negatives are pruned down to a smaller number of "hard" positive and negative matches, which are used to update the parameters of the network. We observe that increasing the batch size does not offer benefits in training: see Table 3.1. We thus keep the batch size fixed to $s_n^H = 128$ and $s_p^H = 128$, and increase the ratio of both negative mining $r_n = s_n/s_n^H$ and positive mining $r_p = s_p/s_p^H$. We keep all other parameters constant. In the following, we use the notation r_p/r_n, for brevity.

Architecture	PR AUC
SIFT	.361
CNN3	**.558**
CNN3 ReLU	.442
CNN3 No Norm	.511
CNN3 MaxPool	.420

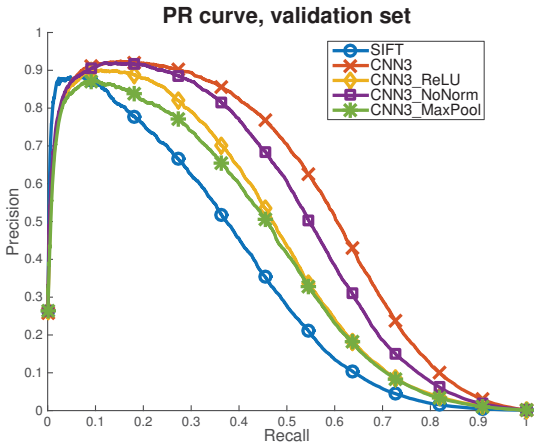


Figure 3.7: Precision-Recall results for experiments on hidden units, normalization, pooling. Fully convolutional CNN3 models with Tanh and ReLU, without normalization, and with max pooling instead of L2 pooling. The best results are obtained with Tanh, normalization, and L2 pooling (i.e. CNN3). PR AUC on the validation set for the top-performing iteration.

Large mining factors have a high computational cost, up to 80% of the total computational cost, which includes mining (i.e. forward propagation of all s_p and s_n samples) and learning (i.e. backpropagating the "hard" positive and negative samples). Note that this is only applicable to the learning stage—once the model is deployed, we discard the Siamese network and do not incur the computational costs related to mining. In order to speed up the learning process we initialize the CNN3 models with positive mining, i.e. 2/2, 4/4, 8/8 and 16/16, with an early iteration of a model trained only with negative mining (1/2).

s_p	s_n	r_p	r_n	Cost	PR AUC
128	128	1	1	—	0.366
256	256	1	1	—	0.374
512	512	1	1	—	0.369
1024	1024	1	1	—	0.325
128	256	1	2	20%	0.558
256	256	2	2	35%	0.596
512	512	4	4	48%	0.703
1024	1024	8	8	67%	**0.746**
2048	2048	16	16	80%	0.538

Table 3.1: Four top rows: effect of increasing batch size, without mining. Four bottom rows: with mining. Mining factors indicate the samples considered (s_p, s_n), the hardest 128 of which are used for training. Column 5 indicates the fraction of the computational cost spent mining hard samples. These experiments correspond to the validation set.

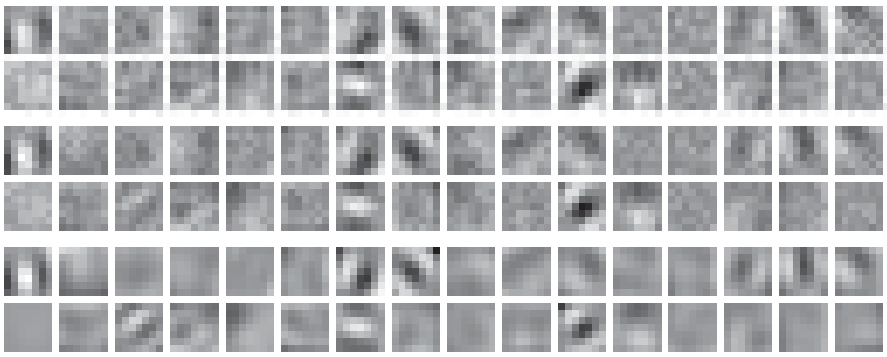


Figure 3.8: Effect of mining on filters. *Top:* first layer filters, before mining. *Middle:* same, after aggressive mining, filters are initialized with the filters in the top row. *Bottom:* difference between top and middle row. For visualization purposes all values are normalized such that the minimum value is displayed in black and the maximum in white for all the values in the group. Note that mining accentuates the maxima and minima of the filters.

Results are shown in Table 3.1. We see that for this particular problem, aggressive "hard" mining is fundamental. This is due to the extremely large number of both negatives and positives in the dataset, in combination with models with a relatively low number of parameters. We observe a drastic increase in performance up to 8/8 mining factors.

We show a qualitative example of how the mining changes the weights in Figure 3.8. While in both cases the filters look similar, the performance doubles when aggressive mining is performed (0.366 vs. 0.746 in PR AUC). In this case the first layer filters are being accentuated by mining, i.e. positive areas are made more positive and negative areas more negative.

3.5.4 Generalization & Comparison to State of the Art

In this section we consider the three splits for the MVS dataset of [9]. We train the top-performing model (i.e. CNN3), with different mining ratios (1/2, 2/2, 4/4 and 8/8), on a combination of two sets, and test it on the remaining set. We select the training iteration that performs best over the corresponding validation set. The test datasets are very large (up to 633K patches) and we use the same procedure as for validation: we consider 10,000 unique points, each with 1,000 random non-corresponding matches. We repeat this process over 10 folds, thus considering 100,000 sets of one corresponding patch vs 1,000 non-corresponding patches. We show results in terms of PR AUC in Table 3.2, and the corresponding PR curves are pictured in Figure 3.9.

We report consistent improvements over SIFT, a hand-crafted descriptor which nevertheless remains the most popular among its brethren. The performance varies significantly from split to split; this is due to the nature of the different sets. 'Yosemite' contains mostly frontoparallel translations with illumination changes and no occlusions (Figure 3.5, row 3);

Train	Test	SIFT	CNN3 mine-1/2	CNN3 mine-2/2	CNN3 mine-4/4	CNN3 mine-8/8
LY+YO	ND	0.349	0.535	0.555	0.630	**0.667**
LY+ND	YO	0.425	0.383	0.390	0.502	**0.545**
YO+ND	LY	0.226	0.460	0.483	0.564	**0.608**

Table 3.2: PR AUC for the generalized results over the three MVS dataset splits, for different mining factors.

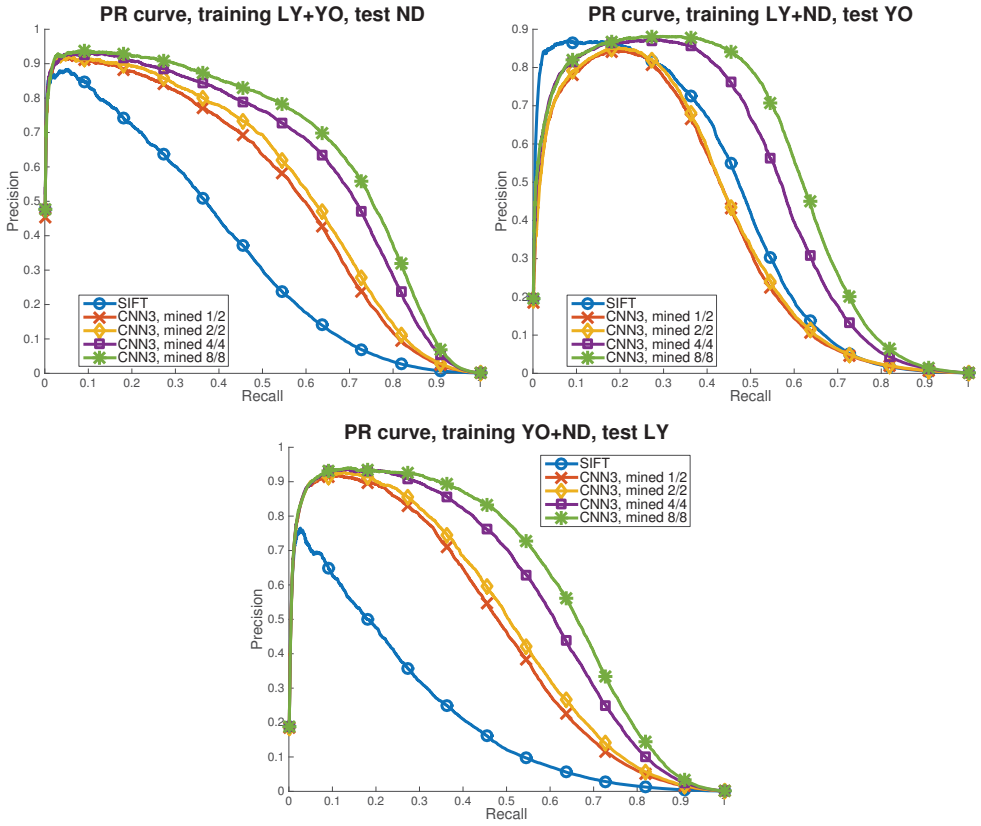


Figure 3.9: PR curves for the generalized results over the three MVS dataset splits, for different mining factors.

SIFT performs well on this type of data. Our learned descriptors outperform SIFT on the high-recall regime (over 20% of the samples; see Figure 3.9), and is 28% better overall in terms of PR AUC. The effect is much more dramatic on 'Notredame' and 'Liberty', which contain significant patch translation and rotation, as well as viewpoint changes around outcropping, non-convex objects, which result in occlusions (Figure 3.5, rows 1-2). Our learned descriptors outperform SIFT by 91% and 169% over ND and LY, respectively.

Test	SIFT (128f)	BGM (256b)	L-BGM (64f)	BinBoost-{64,128,256} (64b)	(128b)	(256b)	VGG (80f)	Ours (128f)
ND	0.349	0.487	0.495	0.267	0.451	0.549	0.663	**0.667**
YO	0.425	0.495	0.517	0.283	0.457	0.533	**0.709**	0.545
LY	0.226	0.268	0.355	0.202	0.346	0.410	0.558	**0.608**
All	0.370	0.440	0.508	0.291	0.469	0.550	0.693	**0.756**

Table 3.3: Generalized results: PR AUC over the three MVS splits, and a new split with data from all three sets, against SIFT, BinBoost [134], and VGG [122]. We re-train VGG with data from two sets (rows 1-3) and all sets (row 4).

Additionally, we pit our approach against the state of the art descriptors of [134] and [122]. For [134] we consider 4 binary descriptor variants (BGM, BinBoost-64, BinBoost-128, and BinBoost-256) and a floating-point variant (L-BGM); for the binary descriptors we use the Hamming distance, instead of the Euclidean distance. For VGG [122] we re-train their models over two sets at a time, to provide a fair comparison with ours. We consider only their top-performing variant, i.e. the largest descriptor. The VGG descriptor considers multiple compression settings—we show the results for the best model (i.e. floating point, size 80).

The results are summarized in Table 3.3 and shown in Figure 3.10. Due to the binary nature of the Hamming distance, the curves for the binary descriptors can be seen to have a sawtooth shape where each tooth corresponds to a 1-bit difference. Our approach outperforms the baselines on 'Notredame' and 'Liberty'. On 'Yosemite' VGG obtains the best results, and our approach outperforms the other baselines by a smaller margin. We argue that this is due to the fact that ND/LY are not representative of YO. We illustrate this in Figure 3.5 (bottom-right), where we compute the pixel difference over every corresponding pair of regions in each set, and plot its mean and std. deviation: YO exhibits a much smoother mean and a smaller variance, which corresponds with our observation that unlike ND/LY, it contains mostly lighting changes and small displacements. This hurts our approach more than VGG, which builds on traditional grid-based descriptors [122]. To illustrate this point, we re-train both our models and VGG [122] over a new split ('All') with data from all three sets, following the methodology of Sec. 3.5. The results in Figure 3.10 (bottom-right) and in the last row of Table 3.3 show a 9.1% relative improvement over VGG, demonstrating the strong generalization properties of our models and training strategy (with relevant training data). This confirms the findings of the deep learning community.

	Ours (GPU)	Ours (CPU)	SIFT	VGG [122]
Time (ms)	0.76	4.81	0.14	4.21

Table 3.4: Computational cost for one descriptor (in batch).

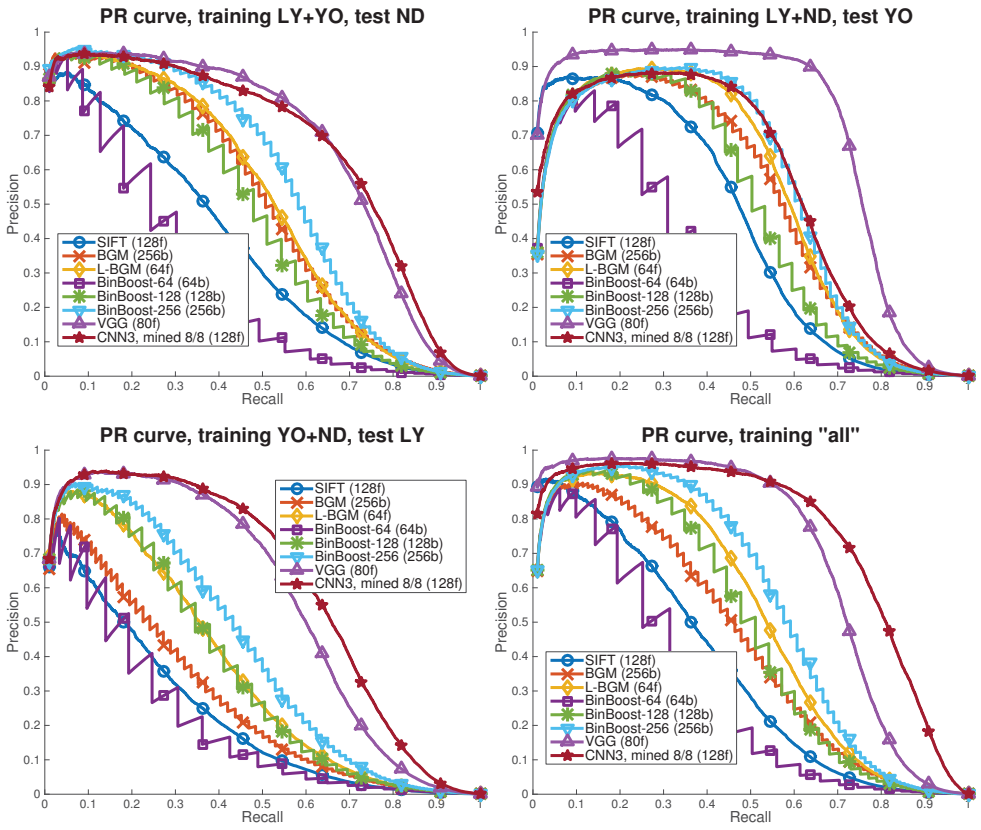


Figure 3.10: Generalized results: PR curves over the three MVS splits, and a new split with data from all three sets, compared to SIFT, Binboost [134], and VGG [122]. We re-train VGG with data from two sets (columns 1-3) and all sets (column 4).

Finally, we provide the computational cost in Table 3.4. The CPU descriptors run on a 12-core 3.47GHz Xeon CPU, multi-threaded. Our GPU variant runs on a Titan Black. SIFT and VGG rely on VLFeat [138], while our approach can still be optimized, particularly for dense computation.

3.5.5 Qualitative Analysis

Figure 3.11 shows samples of matches retrieved with our CNN3-mined-4/4 network, over the validation set for the first split. In this experiment the corresponding patches were ranked in the first position in 76.5% of cases; a remarkable result, considering that every true match had to be chosen from a pool of 1,000 false correspondences. The right-hand image shows cases where the ground truth match was not ranked first; notice that most of these image regions exhibit significant changes of appearance. We include a failure case (highlighted in red), caused by a combination of large viewpoint and illumination changes; these misdetections are however very uncommon.

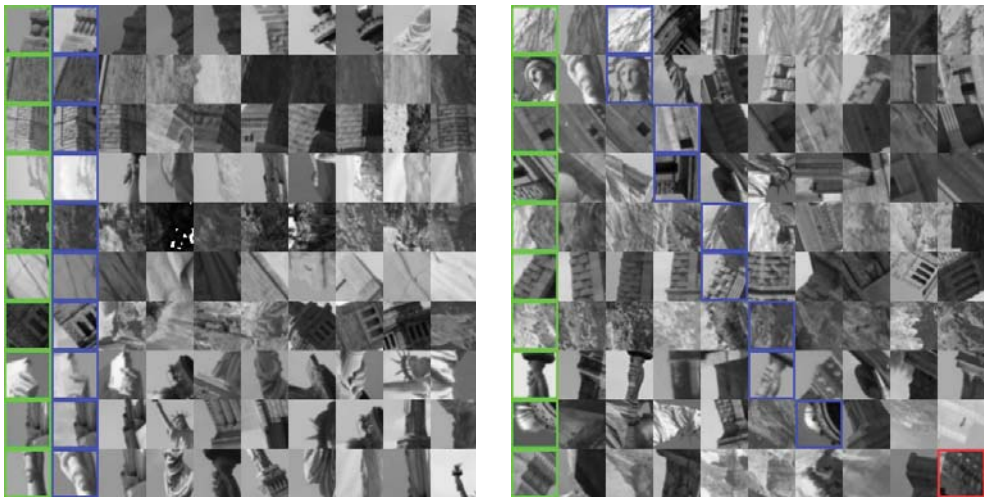


Figure 3.11: Samples of matches retrieved with our descriptor. Each row depicts the reference patch (in green) and the top matching candidates, sorted from left to right by decreasing similarity. The ground truth match is highlighted in blue. Left: Examples where the ground truth match is retrieved in the first position. Right: Examples in which the ground truth match is ranked at positions 2-6. The last row shows a failure case, highlighted in red, where the correct match is ranked 632/1000.

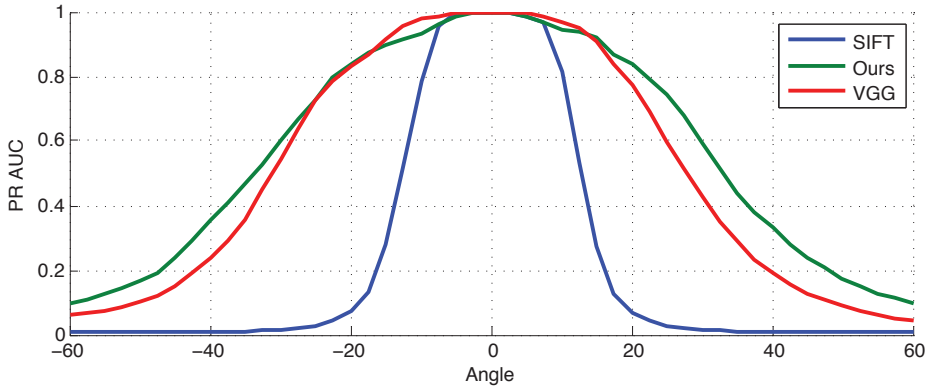
3.5.6 Robustness to Rotation

Robustness to rotation is crucial to many applications, as most rotation-invariant detectors can incur in significant errors when estimating the orientation of an image region. For this purpose we evaluate the performance of our descriptor under rotation errors, in a synthetic scenario. To do this we extract interest points with a Difference of Gaussians detector, and extract their correspondent descriptors. We then increase the rotation of each image region in a systematic manner, and compute descriptors for new features. We match the descriptors and calculate the PR AUC, for increasing values of the rotation error.

We evaluate SIFT and the learned, state-of-the-art VGG descriptor [122] in addition to ours, and show results in Figure 3.12. In particular we use an image of Santiago de Chile and randomly extract 147 image regions (shown in Figure 3.12-(a)), constrained to the center of the image to avoid border artefacts. We observe that while all descriptors perform well below 10 degrees of rotation, SIFT's performance begins to deteriorate by that point. Our descriptor proves the most robust in this scenario, with a 11.2% relative improvement over VGG, using the top-performing model in either case.

This robustness against rotation is particularly valuable when computing dense descriptors, where rotating each patch independently would incur in a considerable computational overhead.

(a) Feature points.



(b) PR AUC results under increasing rotation.

	SIFT	VGG [122]	Ours
Area under the curve	0.223	0.507	**0.564**

(c) Area under the curve of (b).

Figure 3.12: Robustness to Rotation.

3.5.7 Wide-baseline Matching

In this section we apply our models to the wide-baseline stereo dataset of [126], which consists of two multi-view sets of high-resolution images with ground truth depth maps. This allows us to further evaluate the generality of our models across different datasets, and to study how robust the descriptors are against perspective transformations.

We pit our descriptor against SIFT, Daisy [131] and the VGG descriptor [122]. We consider the 'fountain' set, which contains much wider baselines in terms of angular variation and provides a harder challenge. Fig. 3.13 shows the images used; we match '3' (the top-rigth view) against '4'-'8'. We sample 1000 (non-occluded) points randomly and use the ground truth depth maps to determine their correspondence over the opposite camera. We match every point in one camera with every possible correspondence, and compute PR curves. The difference in viewpoint across increasing baselines creates perpective transformations, which include scaling, rotation, and partial occlusions. We explore different image region sizes, from 8×8 up to 64×64. Note that our models were trained with regions of size 64×64, and we upscale the regions if required; we expect that better performance can be obtained training filters of a size commensurate to the region. The results are shown in Tables 3.5-3.9; the top perfomer for every setting is highlighted in bold, and the top performer for a whole baseline is marked with †. As expected, large regions are more informative across narrow baselines, whereas small regions perform better across wide baselines. Our descriptors outperform the baseline in just about every setting, proving that they generalize well across datasets. Note that both our models and VGG are trained with the MVS dataset [9].

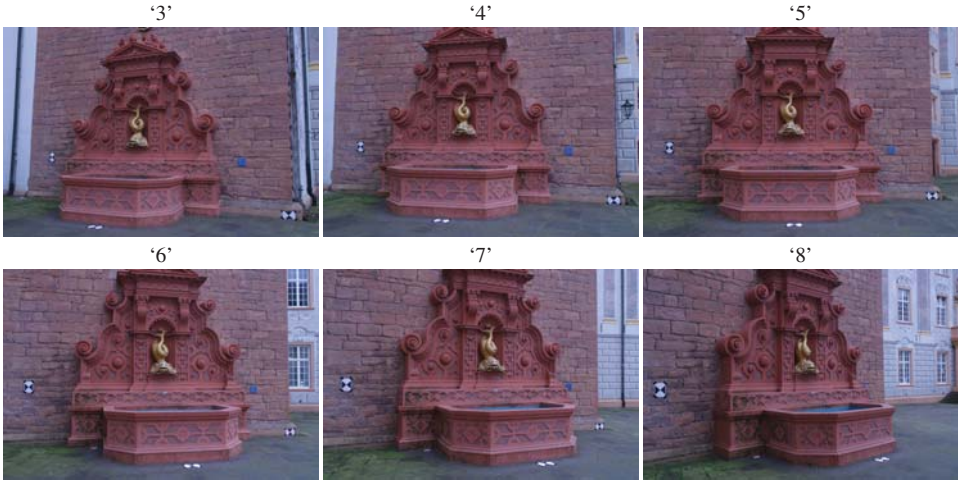


Figure 3.13: Samples of the wide-baseline stereo dataset of [126].

3.5.8 Deformation and Varying Illumination Dataset

Lastly, we evaluate our descriptors on a recent, publicly available dataset featuring challenging non-rigid deformations and very severe illumination changes [121]. The dataset consists

Descriptor	Training	8×8	16×16	24×24	32×32	48×48	64×64
Ours	LY+YO	0.743	**0.912**	0.915	0.916	0.918	**0.923**†
Ours	LY+ND	0.627	0.910	**0.917**	0.916	0.912	0.919
Ours	YO+ND	0.754	0.911	**0.917**	**0.921**	**0.922**	0.922
VGG [122]	YO	0.597	0.850	0.876	0.889	0.897	0.894
VGG [122]	ND	0.598	0.840	0.872	0.877	0.891	0.880
VGG [122]	LY	0.586	0.839	0.875	0.874	0.887	0.879
Daisy [131]	–	**0.796**	0.875	0.878	0.873	0.862	0.835
SIFT [84]	–	0.677	0.837	0.846	0.841	0.798	0.772

Table 3.5: Stereo matching, '3' vs '4'.

Descriptor	Training	8×8	16×16	24×24	32×32	48×48	64×64
Ours	LY+YO	0.481	**0.763**	0.762	0.755	0.713	**0.690**
Ours	LY+ND	0.368	0.757	**0.780**†	0.765	0.703	0.677
Ours	YO+ND	0.504	0.759	0.770	**0.777**	**0.716**	0.685
VGG [122]	YO	0.338	0.633	0.669	0.687	0.672	0.632
VGG [122]	ND	0.330	0.617	0.641	0.657	0.628	0.590
VGG [122]	LY	0.316	0.604	0.641	0.660	0.630	0.582
Daisy [131]	–	**0.526**	0.719	0.735	0.714	0.660	0.594
SIFT [84]	–	0.357	0.551	0.563	0.587	0.540	0.532

Table 3.6: Stereo matching, '3' vs '5'.

Descriptor	Training	8×8	16×16	24×24	32×32	48×48	64×64
Ours	LY+YO	**0.283**	**0.575**†	**0.564**	0.540	0.478	**0.456**
Ours	LY+ND	0.181	0.543	0.561	0.543	0.468	0.424
Ours	YO+ND	0.271	0.547	0.561	**0.556**	**0.490**	0.439
VGG [122]	YO	0.232	0.414	0.466	0.456	0.420	0.400
VGG [122]	ND	0.234	0.402	0.441	0.440	0.381	0.372
VGG [122]	LY	0.223	0.389	0.424	0.423	0.388	0.365
Daisy [131]	–	0.278	0.482	0.510	0.500	0.440	0.363
SIFT [84]	–	0.143	0.340	0.328	0.333	0.300	0.308

Table 3.7: Stereo matching, '3' vs '6'.

Descriptor	Training	8×8	16×16	24×24	32×32	48×48	64×64
Ours	LY+YO	**0.138**	0.337	0.331	0.301	0.240	0.218
Ours	LY+ND	0.088	0.319	**0.336**	0.339	0.253	0.197
Ours	YO+ND	0.121	**0.341**†	0.333	**0.340**	**0.275**	**0.228**
VGG [122]	YO	0.109	0.226	0.250	0.239	0.220	0.174
VGG [122]	ND	0.115	0.229	0.242	0.228	0.198	0.182
VGG [122]	LY	0.107	0.215	0.233	0.220	0.192	0.166
Daisy [131]	–	0.131	0.283	0.323	0.315	0.252	0.172
SIFT [84]	–	0.066	0.158	0.149	0.152	0.125	0.138

Table 3.8: Stereo matching, '3' vs '7'.

Descriptor	Training	8×8	16×16	24×24	32×32	48×48	64×64
Ours	LY+YO	**0.080**	**0.188**†	0.180	0.156	0.**110**	**0.088**
Ours	LY+ND	0.058	0.173	0.158	0.153	0.087	0.058
Ours	YO+ND	0.078	0.178	**0.183**	**0.159**	0.107	0.082
VGG [122]	YO	0.062	0.125	0.107	0.086	0.080	0.067
VGG [122]	ND	0.062	0.121	0.100	0.075	0.083	0.068
VGG [122]	LY	0.062	0.107	0.094	0.076	0.083	0.064
Daisy [131]	–	0.049	0.098	0.113	0.104	0.060	0.032
SIFT [84]	–	0.028	0.051	0.049	0.045	0.044	0.053

Table 3.9: Stereo matching, '3' vs '8'.

Figure 3.14: Samples of the deformation and varying illumination dataset of [121].

of a series of photographs of 12 deformable objects, such as clothes and newspapers, which are subjected to four different deformation levels and four different illumination levels, i.e. 16 images per object, for a total of 192 grayscale 640×480 images. Feature points, extracted with Difference-of-Gaussians detectors, are provided for each image. Some examples of the kind of transformations featured in the dataset are shown in Fig. 3.14.

We pit our descriptor against DaLI, SIFT, Daisy and the VGG descriptor, and show the results in Table 3.10. We evaluate our model trained on three different splits of the MVS dataset, and observe that they all obtain a similar performance. We outperform the current state of the art in the deformation (Def.) and deformation with illumination (Def.+Ill.) settings. This is despite having to upscale the image regions from 41×41 pixels to 64×64 pixels, the fact that the image regions are cropped to be circular while our descriptor relies on square regions, and that we trained our descriptors on datasets of rigid, non-deformable objects. In the case of only illumination changes (Ill.), we obtain a performance very close to the DaLI descriptor [121], explicitly designed to deal with these kind of transformations. We also compare favorably to the VGG descriptor [122], which we outperform in every scenario.

Descriptor	Training	Def.	Ill.	Def.+Ill.
Ours	LY+YO	76.568	88.434	75.933
Ours	LY+ND	75.702	87.521	75.606
Ours	YO+ND	**76.731**	88.898	**76.591**
VGG [122]	YO	74.120	87.342	74.765
VGG [122]	ND	72.629	84.690	72.599
VGG [122]	LY	72.602	84.848	72.565
DaLI [121]	-	70.577	**89.895**	72.912
Daisy [131]	-	67.373	75.402	66.197
SIFT [84]	-	55.822	60.760	53.431

Table 3.10: Results on the dataset of [121]. We evaluate over three different settings, corresponding to deformation changes only (Def.), illumination changes only (Ill.), and both simultaneously (Def.+Ill.). We show the mean accuracy of descriptor matches and highlight the top-performing descriptor for each of setting, in bold.

3.6 Summary

In this chapter we have presented an interest point descriptor, which provides discriminative representations of local image regions. In contrast with hand-crafted descriptors, we propose a data-driven approach which is able to learn descriptors from large datasets of local image regions. Our descriptor is learnt by using corresponding and non-corresponding pairs of image regions under different viewing conditions, maximizing the invariance and discriminative power.

We propose learning a deep convolutional network using a Siamese network architecture, which employs two CNNs with identical parameters fed with pairs of image regions. Each region is propagated forward through the network, providing two CNN outputs that must be similar in terms of L_2 error for corresponding pairs and dissimilar otherwise. In addition we propose a novel training scheme, which yields large performance gains in image region retrieval, based on aggressive mining for both positive and negative region pairs.

We carried out a thoroughly evaluation of multiple architectures, configurations and hyper-parameters. Afterwards, we demonstrate consistent performance gains over the state of the art. Our models generalize well across different datasets and applications, including wide-baseline matching, non-rigid deformations and extreme illumination changes. Moreover, our descriptors can be used as a drop-in replacement for existing descriptors, such us SIFT.

Chapter 4

Robust Monocular Camera Pose Estimation

In this chapter we propose our novel algorithms to deal with the camera pose estimation problem using monocular images. The main advantages of our initial approach are twofold: first, it integrates the outlier rejection within the pose estimation pipeline with a negligible computational overhead; and second, its scalability to arbitrarily large number of correspondences. Given a set of 3D-to-2D correspondences, we formulate pose estimation problem as a low-rank homogeneous system where the solution lies on its 1D null space. Outlier correspondences are those rows of the linear system which perturb the null space and are progressively detected by projecting them on an iteratively estimated solution of the null space. Since our outlier removal process is based on an algebraic criterion which does not require computing the full-pose and reprojecting back all 3D points on the image plane at each step, we achieve speed gains of more than $100\times$ compared to RANSAC strategies. An extensive experimental evaluation will show that our solution yields accurate pose estimation results in situations with up to 50% of outliers, and can process more than 1000 correspondences in less than $5ms$.
Afterwards, we propose a real-time and accurate algorithm that is the first one exploiting the fact that in practice the 2D position of not all 2D features is estimated with the same accuracy. Assuming a model of such feature uncertainties is known in advance, we reformulate the camera pose estimation problem as a maximum likelihood estimation approximated by an unconstrained Sampson error function, which naturally penalizes the most noisy correspondences. The advantages of this approach are clearly demonstrated in synthetic experiments where feature uncertainties are exactly known. Pre-estimating the feature uncertainties in real experiments is, though, not easy. Concretely, we propose to model feature uncertainty as 2D Gaussian distributions representing the sensitivity of the 2D feature detectors to different camera viewpoints. When using these noise models with our formulation we still obtain promising pose estimation results that outperform the most recent approaches.

4.1 Introduction

The camera pose estimation problem from monocular images consist in the estimation of the extrinsic parameters –location and orientation– and sometimes also the intrinsic parameters of a camera with respect to a 3D scene from a single perspective image with n 3D-to-2D point correspondences. Since knowing the internal geometry of the cameras permits to increase the accuracy most of the solvers require a calibrated camera.

Therefore, the fundamental information sources used during the camera pose estimation process are the intrinsic camera parameters –focal lengths, optical center, skew and radial distortion– and the 3D-to-2D correspondences between the 3D scene and the 2D image. These 3D-to-2D correspondences are typically computed by means of a matching task between a set of image features and a set of previously known 3D scene features, which will be considered from now on the 3D scene model.

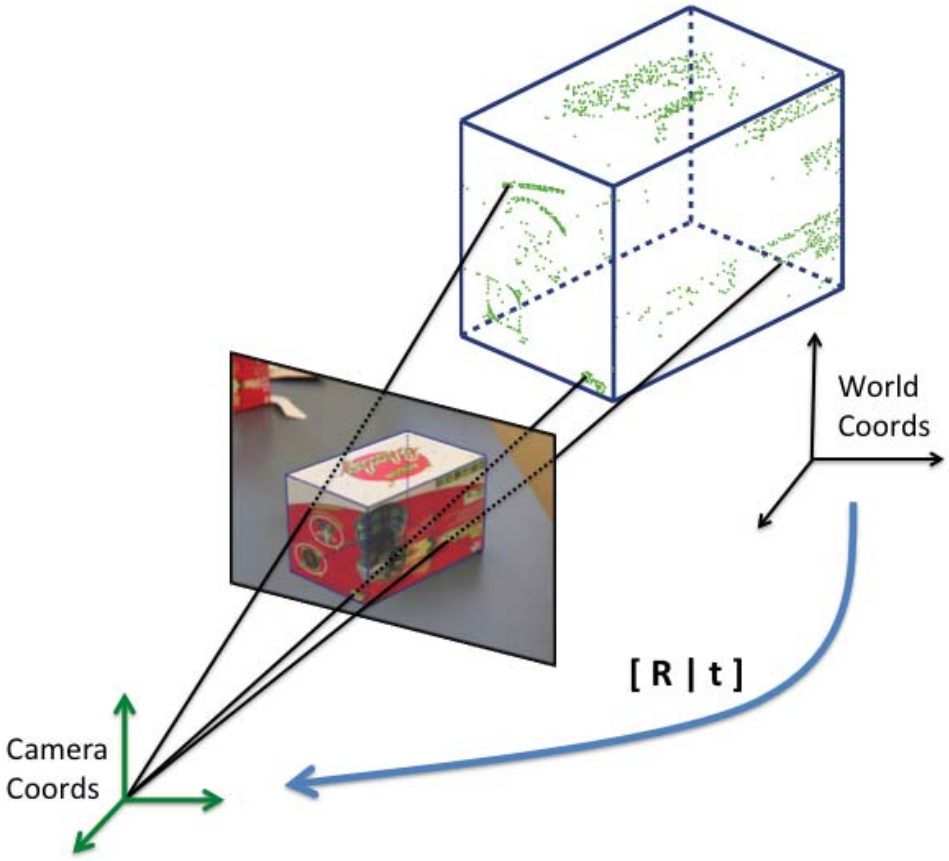


Figure 4.1: PnP problem definition: given an input image and a known textured 3D model, the problem is to estimate the camera pose w.r.t. the 3D model in terms of $[\mathbf{R}|\mathbf{t}]$. Valid 3D-to-2D correspondences are denoted as black lines from the camera center (origin of the camera coordinates) trougth interest points on the 2D image up to 3D points on the model.

In the previous chapters interest point detectors and descriptors have been introduced, in this chapter these kind of algorithms are used to get the image features to build 3D-to-2D correspondences. Depending on the number of 3D-to-2D correspondences the camera pose estimation problem can be also called the Perspective-n-Point (PnP) problem, where n represents the number of correspondences. This problem can be solved for very small and concrete numbers of correspondences, e.g. P3P or P4P solvers, or for an undetermined number of correspondences, in this last case the solvers are simply called PnP solvers. In Figure 4.1 the PnP problem is graphically defined showing some correct correspondences between the image and the 3D model.

The PnP is at the core of many Computer Vision tasks with applications in other areas such as Robotics, Photogrammetry or Augmented Reality. Despite being a topic studied for more than a century (the first solution dates back to 1841 [40]) the last decade has witnessed a wide body of literature in which the primary goal has been to build efficient and accurate solutions.

The Efficient PnP (EPnP) [97] was the first closed-form solution to the problem with $O(n)$ complexity and little loss of accuracy with respect to the most accurate iterative methods [85]. The main contribution of EPnP was to introduce a fixed number of virtual control points to represent whatever number of 3D points. The problem was then reduced to that of estimating the position of the control points, which was efficiently done using linearization techniques. Subsequent works dealing with an arbitrary number of correspondences, have improved the accuracy of the EPnP specially for the minimal cases with $n = \{3, 4, 5\}$ correspondences. This has been essentially achieved by replacing the linearization approaches with more sophisticate polynomial solvers [75] along with new reformulations of the PnP as a least squares problem [47, 149, 148].

However, some issues and challenges remain to be studied or have been partially shelved. We deal with three of them in this chapter:

- **The fundamental problem of mismatched correspondences** has not been directly handled by PnP solvers. They assume that all correspondences are at most corrupted by noise, but not by mismatches. These outlier correspondences are typically rejected in a preliminary stage using P3P solvers [62] in combination with RANSAC-based schemes [32, 46]. Afterwards, PnP problem is solved with the remaining inlier correspondences. This initial rejection stage significantly lessens the efficiency properties of the whole pose estimation process, especially for large number of outliers.
- **The particular structure of the noise associated to each correspondence.** Although all previous PnP solvers assume that correspondences may be at most corrupted by noise and show robustness against large amounts of it, none of them considers that the particular structure of the noise associated to each correspondence, could indeed to be used to further improve the accuracy of the estimated pose. Specifically, existing solutions assume all 2D correspondences to be affected by the same model of noise, a zero mean Gaussian distribution, and consider all correspondences to equally contribute to the estimated pose, independently of the precision of their actual location.
- **Another recurrent problem is the scalability** provided by the PnP solvers. Commonly, the efficiency of PnP solvers decreases linearly with respect to the number of

correspondences. Therefore, a large number of correspondences reduces significantly the efficiency of the solver. This problem has been recently studied in [149, 148] where scalable PnP solvers to thousands of correspondences have been proposed.

4.1.1 Overview

This chapter is organized as follows. Next section introduces the most relevant state-of-the-art approaches to solve the PnP problem. In Section 4.3 the EPnP solver [97] is revisited and reformulated in order to increase its accuracy and efficiency providing almost constant computational cost for any number of correspondences, even thousands. Afterwards, the problem of mismatched correspondences is addressed in Section 4.4, where we propose our outlier rejection mechanism that is fully integrated in our pose estimation pipeline requiring a negligible computational overhead. In this section a deep comparative evaluation against state of the art is performed, showing the robustness to outlier correspondences and scalability abilities of our proposed PnP methods. In Section 4.5 in order to further improve the accuracy of the estimated pose we model the particular structure of the noise for each correspondence and we reformulate the PnP problem as a maximum likelihood estimation. In addition we also propose a strategy to compute a specific uncertainty model per correspondence in real experiments, by modelling the sensitivities of 2D interest point detectors to different viewpoints. We also show our approach outperforms the most recent techniques in terms of accuracy while keeping a running time still linear with respect to the number of correspondences.

4.2 Related PnP Approaches

The PnP problem has been primarily addressed for a fixed and small number of correspondences. Several closed-form solutions have been proposed to solve the P3P [40, 21, 43, 62, 34, 74], the P4P [32], and the problem with $n = \{4, 5\}$ points [132]. Yet, while these minimal problems can be very efficiently solved, they can produce multiple solutions [32] and they have the drawback of being very sensitive to noise. This noise is attenuated by exploiting data redundancy of larger point sets. The most straight-forward algorithm for doing so, is the $O(n)$ Direct Linear Transformation (DLT) [46]. However, DLT does not make use of the fact that in the PnP problem the calibration parameters are assumed to be known in advance, and generally gives poorer results than methods that explicitly consider these parameters. The most "specialized" PnP approaches for an arbitrary number of points can be roughly split into iterative and non-iterative methods.

Iterative PnP approaches optimize an objective function involving all correspondences. Standard objective functions are based on geometric errors (e.g. 2D reprojection) [103]. This method, though, is computationally expensive as it performs a demanding exploration of the solution space to avoid local minima solutions. In the Procrustes PnP (PPnP) [35], the error between the 3D points and their estimated locations is iteratively minimized in a least squares sense. Yet, since this approach is not combined with a global search strategy, it is prone to fall into local minima. An alternative to geometric errors, is to optimize algebraic errors, which are faster to compute [7]. This is explored in [85], that minimizes an algebraic error resulting

from deviations in the line of sight of the 3D-to-2D correspondences. Again, this iterative approach is sensitive to local minima.

Most of the limitations undergone by iterative methods (i.e, local minima solutions and/or high computational cost) may be overcome by non-iterative approaches. Paradoxically, early solutions were not computationally tractable, as they considered all n points as unknowns of the problem. Retrieving their locations, after imposing inter-point distances, lead to $O(n^5)$ [110] and $O(n^8)$ [2] solutions. More recently, there have been several closed form and $O(n)$ formulations that can afford arbitrarily large point sets. The first of these techniques was the EPnP [97, 72], that reduced the PnP to retrieving the position of a fixed number of control points spanning any number of 3D points, three control points for the planar case and four for the non-planar case. Thus, in the non-planar case 3x4 parameters must be estimated, entailing that $n \geqslant 6$ 3D-to-2D correspondences are needed to obtain accurate solutions. The inter-point distance constraints were only considered between these control points, and the resulting quadratic polynomials were solved with simple linearisation techniques. This reformulation of the problem permitted dealing with hundreds of correspondences in real time.

Subsequent works have improved the accuracy of the EPnP, still in $O(n)$, by replacing the linearisation with polynomial solvers. For instance, the Robust PnP (RPnP) method [75] replaces the linearisation scheme of [110] by retrieving the roots of a seventh degree polynomial that results from the least square minimization of multiple P3P problems solved using [74]. The Direct-Least-Squares (DLS) method [47] formulates a non-linear cost function, that produces a fourth order polynomial system and is solved using the Macauley matrix method. The main drawback in DLS is related with its rotation parametrization using the Cayley representation, which is degenerate at 180 degrees. A new unpublished version of the DLS that avoids this problem is provided in the author's webpage[1], their solution consists in solving the equation system three times under known rotations. In order to address the Cayley drawback, [149, 148] propose two direct minimization methods using a quaternion parameterization solved by means of a Gröbner basis solver, resulting in the so-called Accurate and Scalable PnP (ASPnP) [149] and the Optimal PnP (OPnP) [148], two of the most competitive techniques in the state of the art. In addition, both methods obtain accurate solutions with an almost constant computational cost, independently of n.

4.3 Real-Time and Scalable PnP Solution

We next describe the main ingredients of our real-time and scalable solver for the PnP problem. Our proposed method takes advantage of the linear formulation of the EPnP method, which allows to describe all the 3D points belonging to a known 3D model using their barycentric coordinates with respect to a set of control points. Therefore, as well as the EPnP method our goal is to retrieve the position in camera coordinates of those control points. These coordinates cannot be directly estimated, however it is straightforward to estimate the subspace where those control points lie in camera coordinates.

Our proposed method, as the EPnP method, is a two stages method. In the initial stage the subspace containing the 3D control points in camera coordinates is estimated. Afterwards, in

[1] `http://www-users.cs.umn.edu/~joel/index.php?page=software`

the second stage control points and camera pose are determined. In the EPnP method they are determined by solving four linear equation systems and in our case applying an alignment based on Procrustes.

We first review the linear formulation of the problem that results from the EPnP algorithm [97], which have linear computational cost with respect to n. Afterwards, we rewrite the EPnP equations in order to estimate the subspace with almost constant computational cost, independently of n. Finally, our Procrustes-based alignment is proposed.

4.3.1 Problem Statement

Let us assume we are given a set of 3D-to-2D correspondences between n 3D reference points $\mathbf{p}_i^w = [x_i^w, y_i^w, z_i^w]^\top$ expressed in a world coordinate system w and their 2D projections $\mathbf{u}_i = [u_i, v_i]^\top$, for $i = 1, \ldots, n$. Let $\mathbf{A}$ be the camera internal calibration matrix, also assumed to be known in advance. Given these assumptions, the goal of the PnP is to estimate the rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ that align the camera and world coordinate frames. As it is standard, this is addressed through the minimization of a cost function based on the accumulated reprojection errors. For each point i, we have the following perspective constraint with three equations,

$$d_i \begin{bmatrix} \mathbf{u}_i \\ 1 \end{bmatrix} = \mathbf{A} \, [\mathbf{R}|\mathbf{t}] \begin{bmatrix} \mathbf{p}_i^w \\ 1 \end{bmatrix} , \tag{4.1}$$

where d_i is the depth of the point.

4.3.2 Revisiting the EPnP Linear Formulation

Following the EPnP formulation [97], $\mathbf{p}_i^w$ can be rewritten in terms of the barycentric coordinates of four control points $\mathbf{c}_j^w$, $j = 1, \ldots, 4$, chosen so as to define an orthonormal basis centred at the origin of the world coordinate system. Every reference point, can therefore be expressed as a weighted sum of the control points,

$$\mathbf{p}_i^w = \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^w \tag{4.2}$$

where α_{ij} denotes the barycentric coordinate for a given control point j and a reference point i. In addition, barycentric coordinates are constrained for each reference point $\mathbf{p}_i^w$ as,

$$\sum_{j=1}^{4} \alpha_{ij} = 1. \tag{4.3}$$

Note that the barycentric coordinates α_{ij} are independent on the coordinate system, and specifically they remain the same when writing the reference points in the camera coordinate

system c. That is,

$$\mathbf{p}_i^c = \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^c. \tag{4.4}$$

If we replace the term $\mathbf{R}\mathbf{p}_i^w + \mathbf{t}$ by $\mathbf{p}_i^c$ in equation (4.1), the three equations becomes,

$$\begin{aligned} d_i \begin{bmatrix} \mathbf{u}_i \\ 1 \end{bmatrix} &= \mathbf{A} \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^c \\ d_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} &= \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4} \alpha_{ij} \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix} \end{aligned} \tag{4.5}$$

where the unknown parameters are the depths d_i and the control points $\mathbf{c}_j^c$ instead of $[\mathbf{R}|\mathbf{t}]$. Since, A is an upper triangular matrix where its last row is $[0 \quad 0 \quad 1]$, equation (4.5) implies that,

$$d_i = \sum_{j=1}^{4} \alpha_{ij} z_j^c \tag{4.6}$$

where z_j^c denotes the third coordinate of each control point. Substituting equation (4.6) in equation (4.5) the perspective projection constraint for one single correspondence can be rewritten as the following linear system of two equations,

$$\sum_{j=1}^{4} \alpha_{ij} f_u x_j^c + \sum_{j=1}^{4} \alpha_{ij} \gamma y_j^c + (u_0 - u_i) z_j^c = 0 \tag{4.7}$$

$$\sum_{j=1}^{4} \alpha_{ij} f_v y_j^c + (v_0 - v_i) z_j^c = 0 \tag{4.8}$$

Concatenating these equations for all n correspondences can be expressed as a linear system,

$$\mathbf{M}\mathbf{x} = \mathbf{0} \tag{4.9}$$

where M is a $2n \times 12$ matrix and $\mathbf{x} = [\mathbf{c}_1^{c\top}, \ldots, \mathbf{c}_4^{c\top}]^\top$ is the unique unknown, a 12-dimensional vector made of the control point coordinates in the camera reference system.

The solution $\mathbf{x}$ belongs to the null space of $\mathbf{M}$, that can be expressed as,

$$\mathbf{x} = \sum_{i=0}^{N} \beta_i \mathbf{v}_i \tag{4.10}$$

where $\mathbf{v}_i$ denotes the singular vectors of $\mathbf{M}^T\mathbf{M}$ corresponding to the N null singular values. Therefore, the final solution $\mathbf{x}$ is determined by the coefficients β_i. In [97] is shown that the null space can be spanned up to 4 singular vectors with lower singular values.

In order to estimate the appropriate values for β_i the known distances between the control points must be preserved, that means an small number of quadratic equations must be solved using linearisation techniques for the value of $N = 1, 2, 3$ and 4. Since the N value is not easy to estimate, [97] proposes to solve the equation system for each feasible value of N and keep the one that yields the smallest reprojection error. Finally, in order to improve the accuracy an iterative Gauss-Newton minimization on the inter-point distances is proposed in [72] to refine the β_i values.

4.3.3 Reformulating the EPnP Linear Formulation

In the EPnP method [97], the most expensive computation in terms of computational cost is related with the computation of the matrix $\mathbf{M}$ and its product $\mathbf{M}^T\mathbf{M}$. We propose reduce drastically the computational cost by precomputing some parts of the equations (4.7) and (4.9).

These precomputations can be done because of the properties of the matrix A defining the intrinsic parameters: the inverse matrix of an invertible upper triangular matrix is upper triangular, and when the last row is equal to $[0 \quad 0 \quad 1]$ the row in the inverse matrix remains equal.

Thus, equation (4.5) can be rewritten by multiplying both sides by A^{-1},

$$\mathbf{A}^{-1}\begin{bmatrix} d_i\mathbf{u}_i \\ d_i \end{bmatrix} = \sum_{j=1}^{4} \alpha_{ij}\mathbf{c}_j^c \tag{4.11}$$

therefore, equation (4.6) is still valid and by substituing equation (4.6) in equation (4.11) the perspective projection constraint can be rewritten as the following linear system of two equations,

$$\sum_{j=1}^{4} \alpha_{ij}x_j^c - \sum_{j=1}^{4} \alpha_{ij}u_i^c z_j^c = 0 \tag{4.12}$$

$$\sum_{j=1}^{4} \alpha_{ij}y_j^c - \sum_{j=1}^{4} \alpha_{ij}v_i^c z_j^c = 0 \tag{4.13}$$

where $[u_i^c, v_i^c, 1]^\top = \mathbf{A}^{-1}[u_i, v_i, 1]^\top$ are the normalized 2D coordinates. The previous equation system can be rewritten in matrix form as,

$$\begin{bmatrix} 1 & 0 & -u_i^c \\ 0 & 1 & -v_i^c \end{bmatrix} \mathbf{B}_i\mathbf{x} = \mathbf{0} \tag{4.14}$$

where $\mathbf{B}_i$ is a sparse 3×12 matrix built from the barycentric coordinates α_{ij}, and $\mathbf{x} = [\mathbf{c}_1^{c\top}, \dots, \mathbf{c}_4^{c\top}]^\top$ is our unique unknown, a 12-dimensional vector made of the control point coordinates in the camera reference system. At this point we can exploit the particular sparsity pattern of $\mathbf{B}_i$ to write the equation (4.14) as,

$$\begin{bmatrix} \alpha_{i1} & \alpha_{i2} & \alpha_{i3} & \alpha_{i4} \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & -u_i^c \\ 0 & 1 & -v_i^c \end{bmatrix} \mathbf{x} = \mathbf{0} \tag{4.15}$$

where $\otimes$ denotes the Kronecker product.

Finally, concatenating these equations for all n correspondences can be expressed as a linear system $\mathbf{M}\mathbf{x} = \mathbf{0}$, where $\mathbf{M}$ contains exactly the same coefficients as the matrix $\mathbf{M}$ in equation (4.9).

It is worth to point that in equation (4.15) the products between barycentrics and 1 can be understood by substitutions in $\mathbf{M}$ and the number of real operations is reduced to four per equation. As we will see in the experimental results in Section 4.4.3 this reformulation results in significant speed-ups when building the correspondence matrix $\mathbf{M}$.

4.3.4 Estimating Scale and Absolute Pose

So far we have estimated a solution $\mathbf{x}$ of the system $\mathbf{M}\mathbf{x} = \mathbf{0}$ up to scale, i.e, any scaled version $\gamma\mathbf{x}$ would also be a solution of that linear system. In addition, $\mathbf{x}$ is an estimation of the position of the control points $\mathbf{c}_j^c$ in camera referencial, but our ultimate goal is to estimate the $\mathbf{R}$ and $\mathbf{t}$ that yield to absolute camera pose in the world coordinate system. This is linearly calculated using the generalization of the Orthogonal Procrustes problem [116], which solves, in closed-form, the following minimization problem,

$$\begin{aligned} &\arg\min_{\gamma,\mathbf{R},\mathbf{t}} \sum_{j=1}^{4} \|\mathbf{R}\mathbf{c}_j^w + \mathbf{t} - \gamma\mathbf{c}_j^c\|^2 \\ &\text{subject to } \mathbf{R}^\top\mathbf{R} = \mathbf{I}_3 \end{aligned} \tag{4.16}$$

where I_3 denotes the 3×3 identity matrix.

In equation (4.16) a global optimum alignment between $\mathbf{c}_j^c$ and $\mathbf{c}_j^w$ is found, however this alignment tends to contain errors because of slightly errors in the null space estimation. Therefore, in a similar manner as [72] extends the original EPnP with a final Gauss-Newton optimization, we also propose an iterative refinement of $\mathbf{x}$. For this purpose we consider the estimation of the control points positions from equation (4.16) as,

$$\hat{\mathbf{c}}_j^c = \mathbf{R}\mathbf{c}_j^w + \mathbf{t} \tag{4.17}$$

and we re-estimate the null-space vector as $\hat{\mathbf{x}} = [(\hat{\mathbf{c}}_1^c)^\top, \dots, (\hat{\mathbf{c}}_4^c)^\top]^\top$. This vector is then projected on an extended null space of $\mathbf{M}$, which is spanned by the four singular vectors with

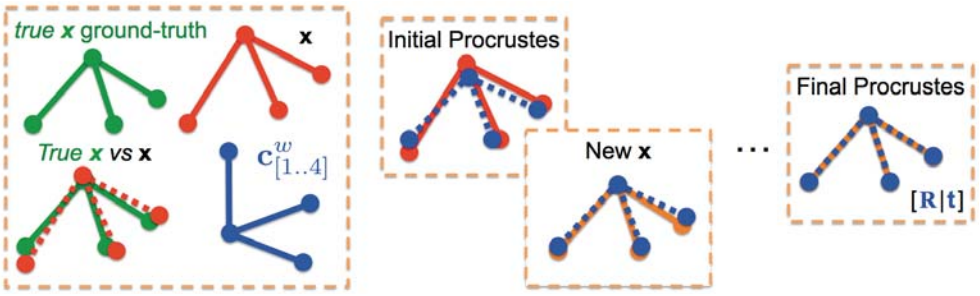


Figure 4.2: Example of the proposed Procrustes-based alignment. Left square contains, a comparison between the ground-truth $\mathbf{x}$ estimated without noise in the correspondences and the $\mathbf{x}$ estimated with noise. The blue control points are expresed in world coordinates. In the rigth side an example showing the evolution of $\mathbf{x}$ iterating our proposed method. First the control points in world coordinates (blue dashed) are aligned with the noisy $\mathbf{x}$ solution using equation (4.16), afterwards in orange is shown the new $\mathbf{x}$ solution computed using equation (4.18).

lower singular values. Recall we know the final solution must lie inside these subspace [72]. Thus $\mathbf{x}$ is recomputed as,

$$\mathbf{x} \leftarrow \mathbf{N}\boldsymbol{\beta} : \arg\min_{\boldsymbol{\beta}} \|\mathbf{N}\boldsymbol{\beta} - \hat{\mathbf{x}}\|^2 \tag{4.18}$$

where $\mathbf{N}$ is a 12×4 matrix representing the 4 singular vectors in the assumed null space of $\mathbf{M}$ and $\boldsymbol{\beta}$ is a 4-dimensional vector of weights. This minimization is solved in closed form, and yields updated values $\mathbf{x}$. These new estimates are in turn fed again to equation (4.16) to recompute γ, $\mathbf{R}$ and $\mathbf{t}$ until the convergence of both equations. An example of this process can be seen in Figure 4.2.

Since this optimization is computed just over the four control points, its computational overhead is negligible.

4.3.5 The Planar Case

Just as with the EPnP [97], the planar case requires a slight modification of the method. Since in this case only three control points are necessary to span the reference points onto the plane, the dimensionality of our vector of unknowns $\mathbf{x}$ drops to 9, and $\mathbf{M}$ becomes a $2n \times 9$ matrix of correspondences. Besides these changes the rest of the algorithm remains completely unchanged.

4.4 Algebraic Outlier Rejection in the PnP Problem

In this section we propose a novel solution to the PnP problem that integrates an outlier-rejection mechanism and does not need to resort to an independent and separate strategy. Like in RANSAC [32], our approach also iterates to remove outliers. However, instead of using a geometric criterion to reject them, where at each step one needs to compute the full pose and project back all points, we use much direct criterion that results from exploring the algebraic error of a linear system inherent of our PnP formulation (See Fig. 4.3). In addition, this process typically converges in less than 5 iterations, even in situations where up to 50% of the correspondences are outliers. As we will show in the experimental section, this results in speed-ups of up to $100\times$ compared to the standard P3P+RANSAC+PnP strategies, while yielding similar or even better accuracies.

We next briefly introduce different outlier rejection schemes proposed in the state of the art, showing that in the PnP problem outliers have been always discarded by preprocessing the correspondences. Afterwards, we describe our robust to outliers PnP method, which is based on our scalable PnP solver proposed in the previous section. We show that based on the previous linear formulation, we can easily formulate a robust outlier rejection procedure based on the minimization of the algebraic error of the linear system. Once an initial outlier-free solution is obtained, the pose can be estimated applying our Procrustes-based alignment introduced in Section 4.3.4.

4.4.1 Related Outlier Rejection Schemes

Yet, as we have pointed out above, the essential problem of dealing with outliers has not been directly handled by previous PnP solutions. To the best of our knowledge this task is set aside, as an independent preliminary step based on RANSAC-like strategies [75, 149, 148], where a series of P3P problems are solved and their solutions evaluated on all points. When a large consensus is found it is finally evaluated on a PnP method. The main drawback of this two stage strategy is that the efficiency of the PnP algorithm is not fully exploited, as most of the time is spent in the multiple evaluations of the P3P.

There are related geometric problems proposing alternative outlier-rejection schemes that are intrinsically integrated within the problem that is being solved. For instance, to robustify PCA [20, 49] use influence functions and [11, 152] use L_1-norm. In [61, 55] the L_1-norm is used within the optimization scheme to handle certain amount of outliers in matrix factorization for multiview geometry problems.

Another way to handle the outlier rejection problem, is to focus on maximizing the number of inliers that satisfy the model that is being optimized. Dual problem [102] or the minimax formulations [145] are used for this purpose, although they are limited to relatively a small number of outliers ($< 35\%$). Larger amounts have been recently handled using a truncated L_2-norm [28]. However, since this approach has a $O(n^d)$ complexity, where d is the dimensionality of the model, it is constrained to problems with relative small dimensionalities, like estimating planar motions or stereo triangulations.

In this section we draw inspiration in these outlier rejection schemes and integrate them

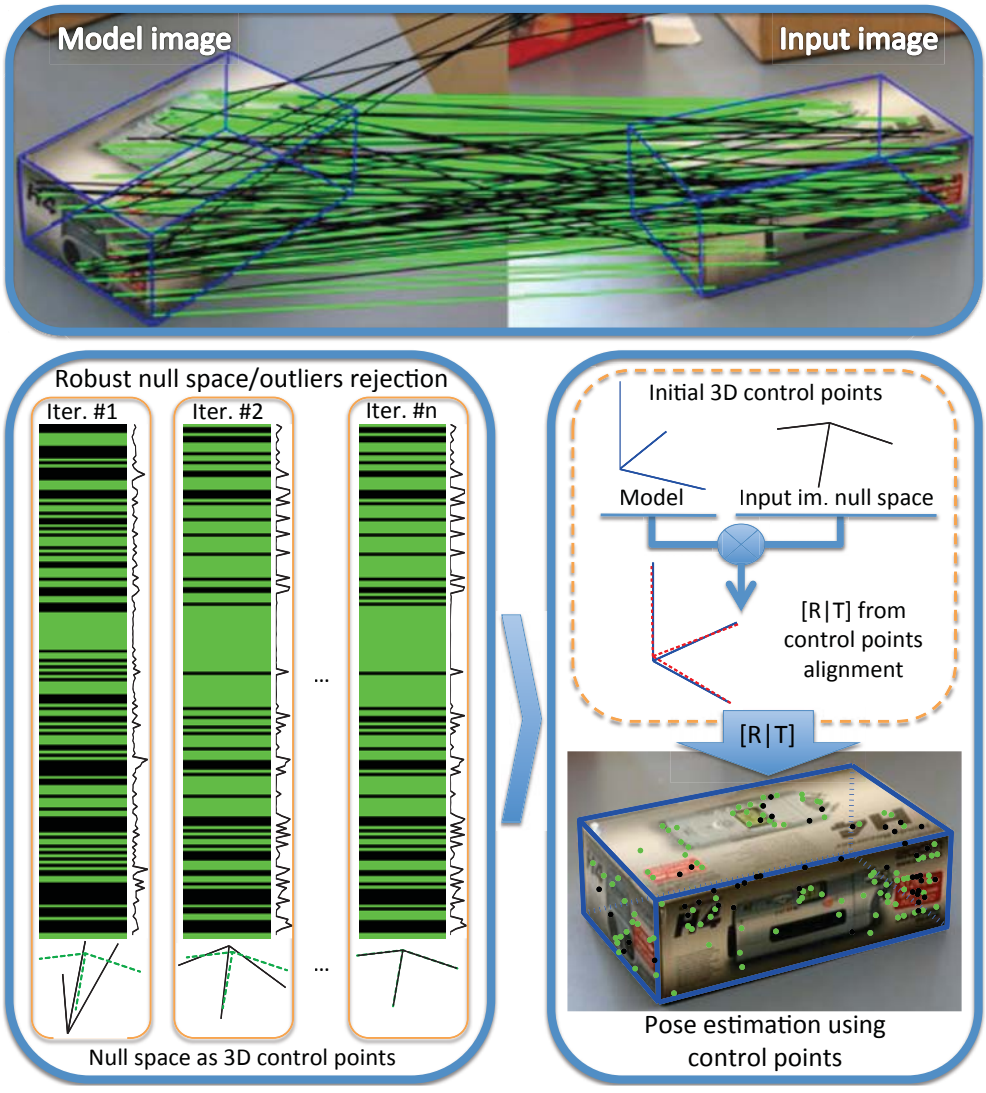


Figure 4.3: Our approach to simultaneously estimate pose and reject outliers. Top: Sample result, where green and black lines indicate the inlier and outlier matches, respectively. The 3D model is overlaid in blue in the input image, using the estimated pose. The overall process (outlier rejection and pose estimation) is done in less than 5 ms. Bottom-left: Our PnP solution iteratively computes the null-space of a linear system. Here we plot the matrix of this system, where rows represent correspondences. The error per match of the estimated null-space is plotted on the side of each matrix, and is used to detect outliers (black rows). The convergence of the null-space is plotted at the bottom of each matrix, where dotted lines depict the true null-space. Bottom-right: The null-space gives an estimate of the 3D model in camera coordinates. We finally compute the pose using the method proposed in Section 4.3.4.

within the linear formulation of the PnP solution proposed in the previous section. The combination of both ingredients let us to progressively discard outliers with a simple step-like loss function and estimate the null-space of the linear system in just a few iterations, each executed in $O(n)$ time.

4.4.2 Robust Null-Space Estimation

As discussed in the original EPnP formulation [97] in the noise-free case the rank of the null-space of $\mathbf{M}$ is one, but due to noise this rank can grow up to four. The algorithm proceeds by separately solving each of the cases and retaining the one that minimizes the reprojection error of all points. However, as it will be shown in the results section, this strategy is sensitive to the presence of outliers.

Instead, we propose a robust approach for solving the system $\mathbf{Mx} = \mathbf{0}$ – built as in Section 4.3.3 –, which is intended to remove outliers, and also those correspondences with large amounts of noise. As a result, we can initially safely assume that the rank of the null space of $\mathbf{M}$ will be one (this assumption is relaxed in the final pose refinement stage explained in Section 4.3.4). Thus, like in the Robust PCA paradigm [11], our robust estimation can be considered as an approximation of a full rank matrix $\mathbf{M}$ because of noise and outliers by a low rank matrix $\mathbf{L}$. However, in contrast to [11], we known in advance the rank of $\mathbf{L}$ and we write our problem as,

$$\begin{aligned} &\underset{\mathbf{L},\mathbf{W}}{\arg\min} \|\mathbf{W}(\mathbf{M}-\mathbf{L})\|^2 \\ &\text{subject to } \text{rank}(\mathbf{L}) = \text{rank}(\mathbf{M}) - 1 \end{aligned} \tag{4.19}$$

where $\mathbf{L}$ is a $2n \times 12$ low rank approximation of $\mathbf{M}$, and $\mathbf{W} = \text{diag}(w_1, w_1, \ldots, w_n, w_n)$ is a $2n \times 2n$ diagonal matrix with binary entries indicating if each particular correspondence is considered an inlier ($w_i = 1$) or an outlier ($w_i = 0$).

Since we are interested in the null space of $\mathbf{M}$, (i.e. the vector $\mathbf{x}$), we rewrite the problem by multiplying the previous minimization by $\mathbf{x}$. Given that $\mathbf{x}$ and the rows in $\mathbf{L}$ lie into complementary subspaces, $\mathbf{Lx} = \mathbf{0}$, the constrained minimization of equation (4.19) can be turned into a non-linear minimization of an algebraic error with unknowns $\mathbf{x}$ and $\mathbf{W}$,

$$\underset{\mathbf{x},\mathbf{W}}{\arg\min} \|\mathbf{WMx}\|^2 \ . \tag{4.20}$$

In order to optimize equation (4.20) we follow the iterative approach detailed in Algorithm 1 in which $\mathbf{x}$ and $\mathbf{W}$ are sequentially estimated.

Initially, all correspondences are assumed to be inliers, and thus $\mathbf{W}$ is initialized to the identity matrix. We then compute the singular value decomposition of $\mathbf{WM}$, and take $\mathbf{x}$ to be the singular vector associated to the smallest singular value. Let $\boldsymbol{\epsilon} = \mathbf{Mx}$ be the residual vector, and $\varepsilon_i = \|[\epsilon_{2i-1}, \epsilon_{2i}]\|$ the algebraic error associated to the i-th correspondence. All

Algorithm 1 Robust Null Space Estimation

Input: $\mathbf{M}$: $2n\times12$ corresp. matrix; δ_{max}: max. algebraic error
Output: $\mathbf{W}$, $\mathbf{x}$
Initialize: $\mathbf{W} = \mathbf{I}_{2n}$, $\xi = \text{Inf}$

loop
  $[\mathbf{U}, \mathbf{S}, \mathbf{V}]$=svd($\mathbf{WM}$);
  where $\mathbf{V}$=$[\mathbf{v}_1, \ldots, \mathbf{v}_{12}]$; $\mathbf{S}$=diag$(s_1, \ldots, s_{12})$
  $\mathbf{x} \leftarrow \mathbf{v}_k$; where $k : s_k = \min(s_1, \ldots, s_{12})$
  $\boldsymbol{\epsilon} = \mathbf{Mx}$
  $\varepsilon_i = \|[\epsilon_{2i-1}, \epsilon_{2i}]\|$
  $\varepsilon_{\text{max}} = \mathrm{Q}_{25\%}(\varepsilon_1, \ldots, \varepsilon_n)$
  if $\varepsilon_{\text{max}} > \xi$ **then**
    return $\mathbf{W}$,$\mathbf{x}$
  else
    $\xi = \varepsilon_{\text{max}}$
  end if
  Update w_i using Eq. 4.21
end loop

of the entries of the matrix $\mathbf{W}$ are updated according to a simple step function,

$$w_i = \begin{cases} 1 & \text{if } \varepsilon_i \leq \max(\varepsilon_{\text{max}}, \delta_{\text{max}}) \\ 0 & \text{otherwise} \end{cases} \tag{4.21}$$

where $\varepsilon_{\text{max}} = \mathrm{Q}_{25\%}(\varepsilon_1, \ldots, \varepsilon_n)$ returns the algebraic error of the correspondence that is at the boundary of the lowest 25% quartile. This function is used as a robust estimator to reject those correspondences with largest algebraic errors. For this purpose, we also evaluated using the median operator. Yet, the median has a breakdown point (maximum number of supported outliers) of 50%, while, as we will show in the results section, the breakdown point of the $\mathrm{Q}_{25\%}$ reaches the 60%.

Furthermore, note that if we solely used this criterion, in an outlier-free case we would be unnecessarily rejecting inlier correspondences. In order to avoid this situation and achieve faster convergence rates we introduce a minimal algebraic error threshold δ_{max} that needs to be reached in order to consider a specific correspondence as outlier, similar to the minimum reprojection error τ used in standard RANSAC to classify matches as outliers. Yet, since in our algebraic error we have normalized the image coordinates by the focal length (when multiplying $[u_i, v_i]$ by $\mathbf{A}^{-1}$) we set this threshold to $\delta_{\text{max}} = 1.4\tau/f$, where f is the camera focal length, and τ is the maximum reprojection error (in pixels) permitted to the inliers. The constant 1.4 accounts for the correlation factor between the f and τ and has been experimentally obtained from a large number of synthetic experiments where we have swept the joint space of both variables.

This process is repeated until the convergence of the null-space $\mathbf{x}$ which usually happens

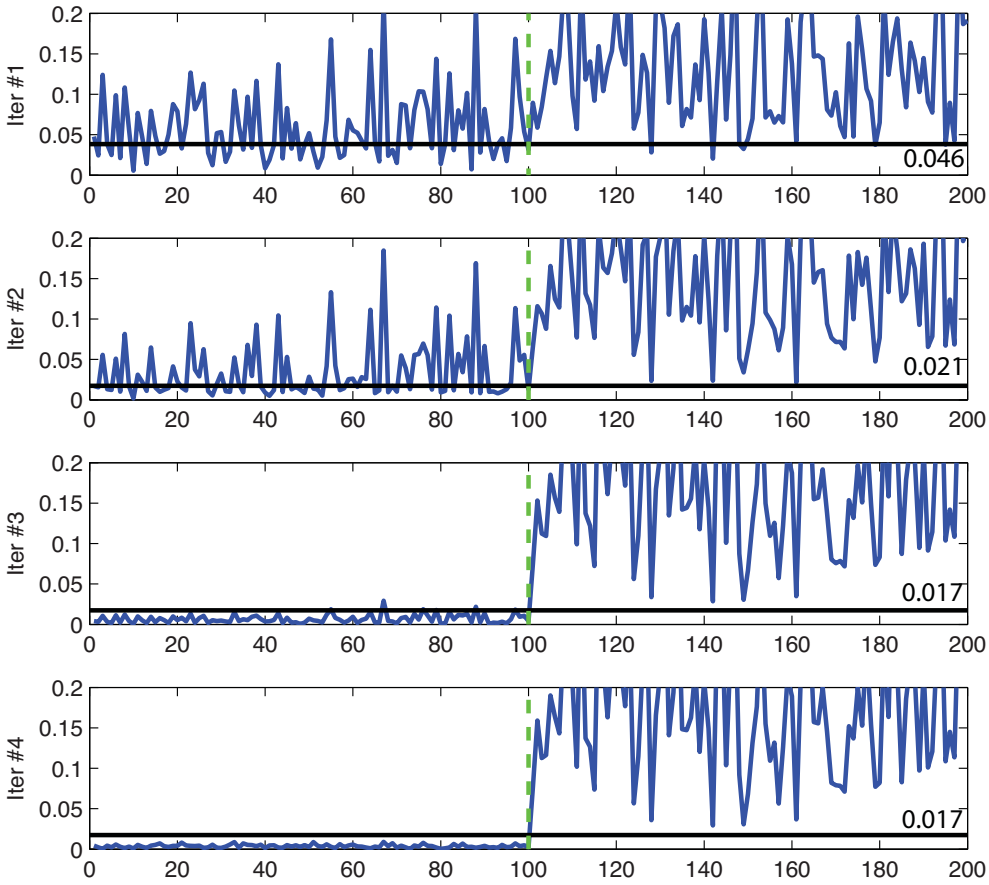


Figure 4.4: Example of outlier rejection with 100 inliers (shown on the left-most part of the graphs) and 100 outliers. The blue line represents the algebraic error of each correspondence, and the horizontal black line is the threshold $\max(\varepsilon_{\max}, \delta_{\max})$ used in Eq. 4.21 to classify each match as inlier or outlier. δ_{max} is the only threshold we need to provide, and we set it to 0.017, which approximately represents an image noise of $\tau = 10$ pixels for a focal length $f = 800$.

in less than 5 iterations. Fig. 4.4 shows an example of the execution of our algorithm with 50% of outliers, which converged after 4 iterations. It is worth mentioning that all entries of $\mathbf{W}$ are reestimated at each iteration, using the updated vector $\mathbf{x}$. By doing this, we are able to re-accept in our inlier set those correspondences that in initial iterations were wrongly labeled as outliers.

4.4.3 Experimental Results

We compare the accuracy and scalability of our methods, with and without the outlier rejection mechanism, against state of the art on synthetic and real data. Our PnP methods

are written in MATLAB and the source code is publicly available[1] within a toolbox, which contains all the evaluated PnP methods and the code to replicate the experimental results.

In these experiments we consider the two proposed PnP methods referred to as Efficient Procrustes PnP (EPPnP) for the scalable one proposed in Section 4.3 without the outlier-rejection scheme and the Robust Efficient Procrustes PnP (REPPnP) for the outlier rejection case proposed in Section 4.4.2.

Synthetic Experiments

We assume a virtual calibrated camera with image size of 640×480 pixels, focal length of 800 and principal point in the image center. We randomly generated 3D-to-2D correspondences, where 3D reference points were distributed into the x, y, z interval $[-2, 2] \times [-2, 2] \times [4, 8]$. We also added Gaussian noise to the 2D image coordinates and different percentages of outliers, produced by randomizing the 2D position of the projections. Finally, we chose the ground-truth translation $\mathbf{t}_{\text{true}}$ as the centroid of the 3D reference points and we randomly generated a ground truth rotation matrix $\mathbf{R}_{\text{true}}$. As a metric errors we used the same as in [75, 148]. The absolute error is measured in degrees between the $\mathbf{R}_{\text{true}}$ and the estimated $\mathbf{R}$ as $\text{e}_{\text{rot}}(\text{deg}) = \max_{k=1}^{3}\{\text{acos}(\mathbf{r}_{\text{k,true}}^{\top} \cdot \mathbf{r}_k) \times 180/\pi\}$ where $\mathbf{r}_{\text{k,true}}$ and $\mathbf{r}_k$ are the k-th column of $\mathbf{R}_{\text{true}}$ and $\mathbf{R}$. The translation error is computed as $e_{\text{trans}}(\%) = \|\mathbf{t}_{\text{true}} - \mathbf{t}\|/\|\mathbf{t}\| \times 100$.

All the plots discussed in this section were created by running 500 independent MATLAB simulations and report the average and median rotation and translation errors.

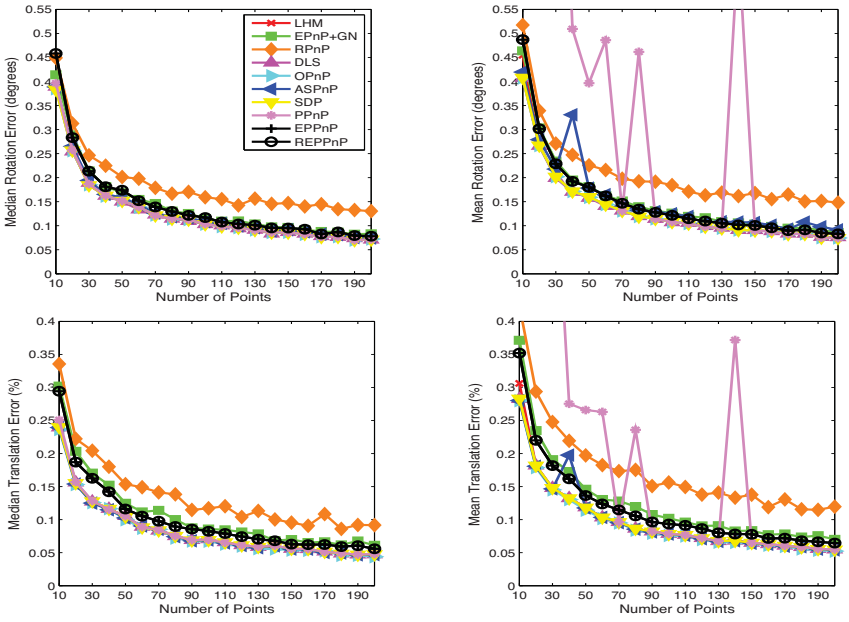
Number of Correspondences and Noise

In these experiments we compared the accuracy and running time of our PnP methods assuming there are no outlier correspondences. However, since we want to show that the outlier rejection process does not affect the final results, we consider our two PnP methods, EPPnP and REPPnP.

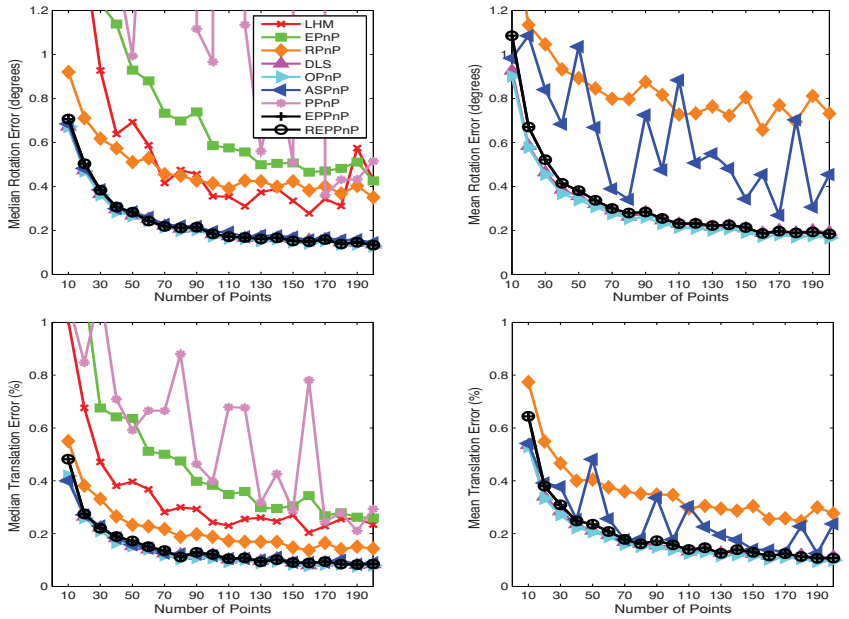
We have compared our formulations against the most recent PnP approaches: the robust version of DLS [47], ASPnP [149], OPnP [148], RPnP [75], PPnP [35], EPnP + GN [72], SDP [117], and the LHM [85].

Figure 4.5 plots the accuracy for increasing number of correspondences, from $n = 10$ to 200, with constant Gaussian noise of $\sigma = 2$ pixels. In addition, Figure 4.6 depicts the errors for increasing amounts of noise, from $\sigma = 0$ to 20, and a constant number of correspondences $n = 30$. Both experiments for planar and non-planar configurations, show that the proposed approach yields accurate solutions, comparable to the best state-of-the-art solutions. Just to give significance of the difference between the errors in our method and the most accurate approach (OPnP), in the worst case it is equivalent to a mean deviation of about 0.5 mm in estimating the position of points randomly distributed within a cube of 40 cm side located at a 60 cm of the camera. For the planar case, it is interesting to note how our approach significantly improves the performance of the EPnP, mostly due to the use of the Procrustes

[1] `http://cmtech.upf.edu/system/files/pdf/REPPnP_Toolbox.zip`

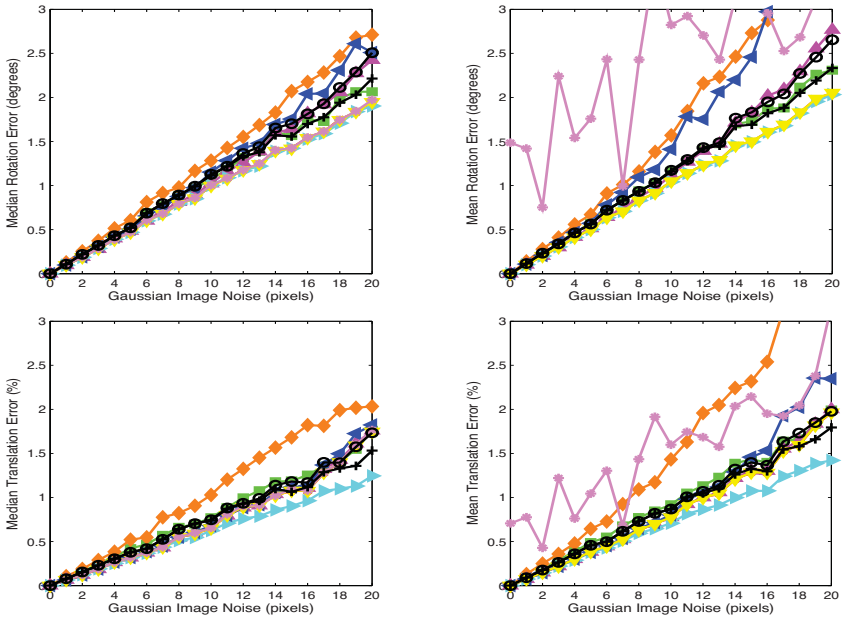


(a) Non-planar case

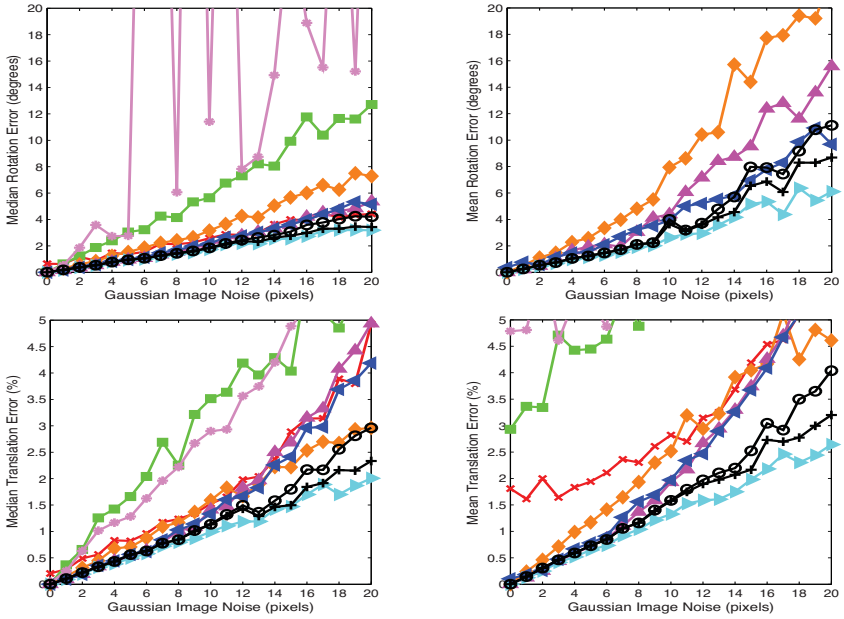


(b) Planar case

Figure 4.5: Synthetic experiments for (a) non-planar objects and (b) planar objects. Varying the number of correspondences.



(a) Non-planar case



(b) Planar case

Figure 4.6: Synthetic experiments for (a) non-planar objects and (b) planar objects. Varying the amount of Gaussian noise. The color codes and line styles are the same as those used in Figure 4.5

refinement stage.

In addition, as we have discussed before, we want to make clear that our approach is not intended to work in minimal cases. In fact, similar to EPnP, the performance of our approach drops for $n \leq 6$. Instead, we seek to exploit the consistency of large number of correspondences. This is indeed a realistic situation, as current interest point detectors are able to extract hundreds of reliable feature points in standard images. And what is most important, we can process all this amount of points very efficiently, and as we will show below, even under the presence of outliers.

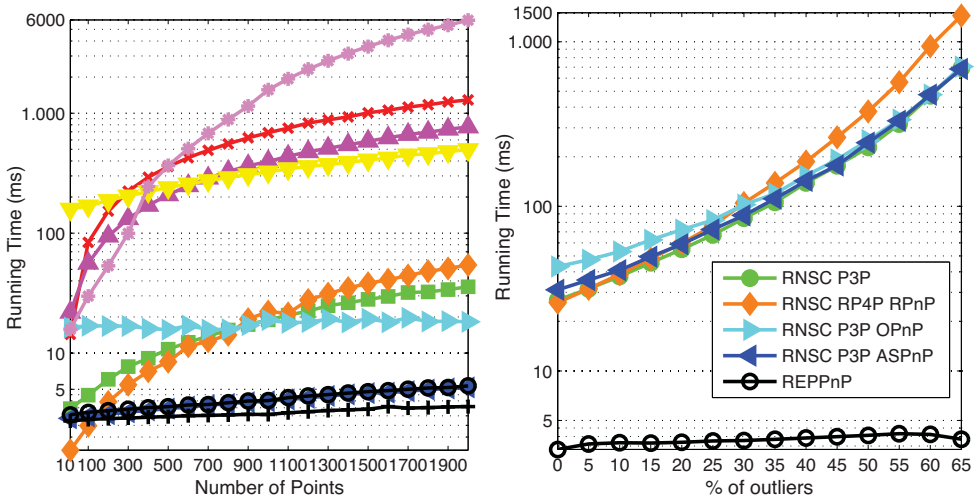


Figure 4.7: Running time. Varying the number of outlier-free correspondences (left) and varying the % of outliers (right). In the left, the color codes and line styles are the same as those used in Figure 4.5.

Figure 4.7(left) shows the computation time of all methods in the outlier free case, for an increasing number of correspondences, from $n = 10$ to 2000, and with a fixed $\sigma = 2$. This experiment was done on an Intel Core i7 CPU to 2.7Ghz and all methods are implemented in MATLAB. Note that our EPPnP method is the fastest one with almost constant computational cost fixed to ≈ 3 ms whatever number of correspondences. Closely followed by ASPnP and our robust implementation REPPnP, which despite being computed in a situation without outliers, it tries to discard them executing several iterations. Figure shows that the cost of those iterations to discard outliers is negligible, increasing slightly the computational time of our EPPnP method.

Robustness to Outliers

The main contribution in this section is to embed the outlier rejection scheme within the pose estimation pipeline. The success of our approach holds in part on the fact that the null-space $\mathbf{x}_{\text{ini}}$ computed initially (i.e., the solution of $\mathbf{M}\mathbf{x} = \mathbf{0}$ before removing any outlier) "only" deteriorates linearly with the presence of outliers. We show this in Figure 4.8(left) where we

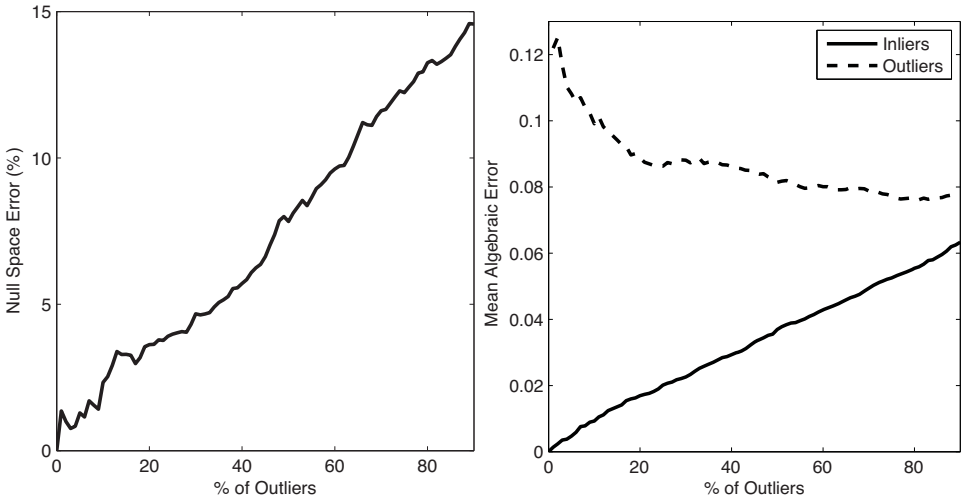
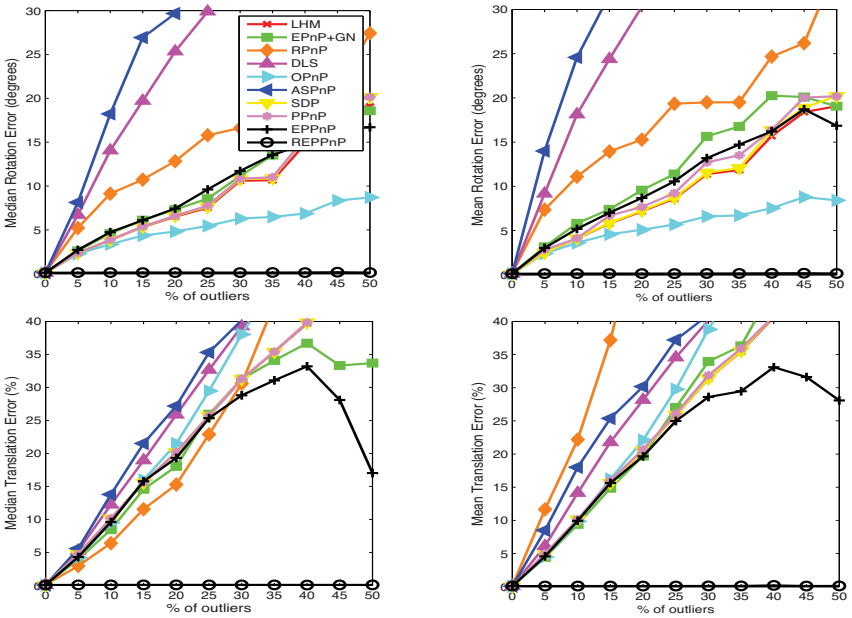


Figure 4.8: Synthetic experiments for different levels of outliers in terms of relative error of the null-space of $\mathbf{M}\mathbf{x} = \mathbf{0}$ computed without removing outliers (left); and algebraic error for the inlier and outliers correspondences projected onto this null space (right). The deviation levels of the initially estimated null-space let to clearly identify inliers and outliers, based on their algebraic error.

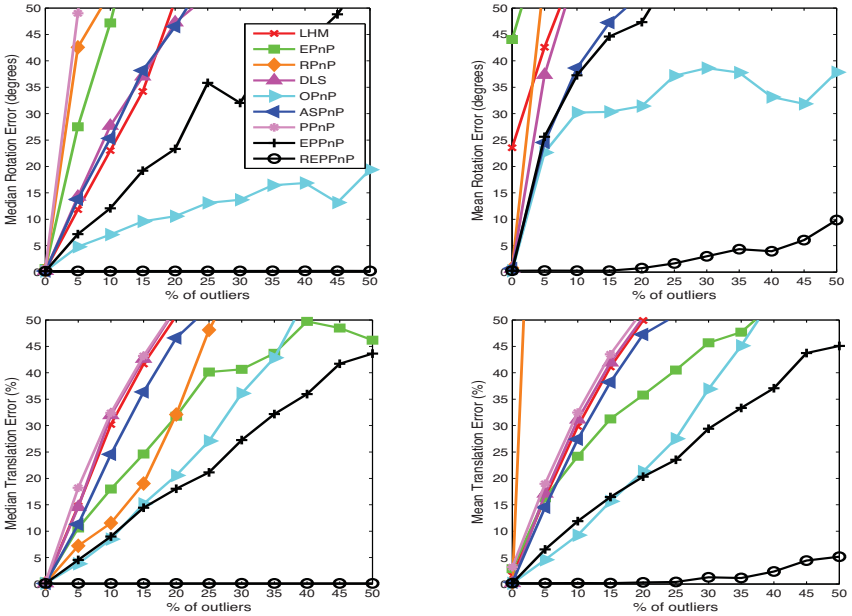
plot $\|\mathbf{x}_{\text{true}} - \mathbf{x}_{\text{ini}}\|/\|\mathbf{x}_{\text{ini}}\| \times 100$ for increasing amounts of outliers. Note that the difference between the true and the initially estimated null-space is kept within reasonable bounds for large percentages of outliers. This lets us to easily detect the outlier correspondences, as they generally have larger algebraic errors than inlier correspondences when projected onto this initial null-space. We show this in Figure 4.8(right), where we plot the algebraic error $\mathbf{M}\mathbf{x}_{\text{ini}}$.

This robustness to the presence of outliers, contrasts with the sensitivity undergone by the geometric error. If we use the same experimental setup, and compute the 2D reprojection using the state-of-the-art PnP methods described above, we see that the obtained solutions are so corrupted that inliers and outliers cannot be distinguished from geometric error alone. Obviously this results in total failures of these methods, if outliers are not initially removed (Figure 4.9).

In Figure 4.10 we therefore compare the performance of our robust approach, and other PnP methods when used in a RANSAC strategy, concretely [46], with the following combinations of minimal and general approaches: (RANSAC+P3P [62]); (RANSAC + RP4P + RPnP [75]); (RANSAC + P3P [62] + ASPnP [149]); and (RANSAC + P3P [62] + OPnP [148]). Note that our approach tend to yield slightly better results than other approches, we suspect it comes from the fact of using a different criterion to choose the inliers correspondences – equation (4.21)–. As mentioned above, the breakdown point of our approach is between 50% and 60% of outliers for the non-planar case and $\approx 5\%$ smaller for the planar case. Note that RANSAC based strategies have higher breakdown points, however as seen in Figure 4.7(right) this is at the expense of a significant increase in the computation time, which is

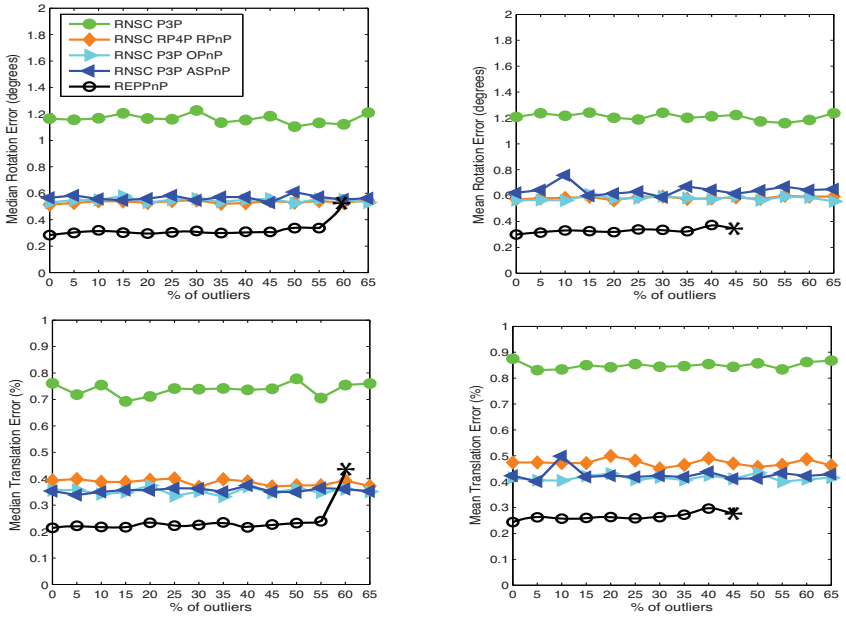


(a) Non-planar case

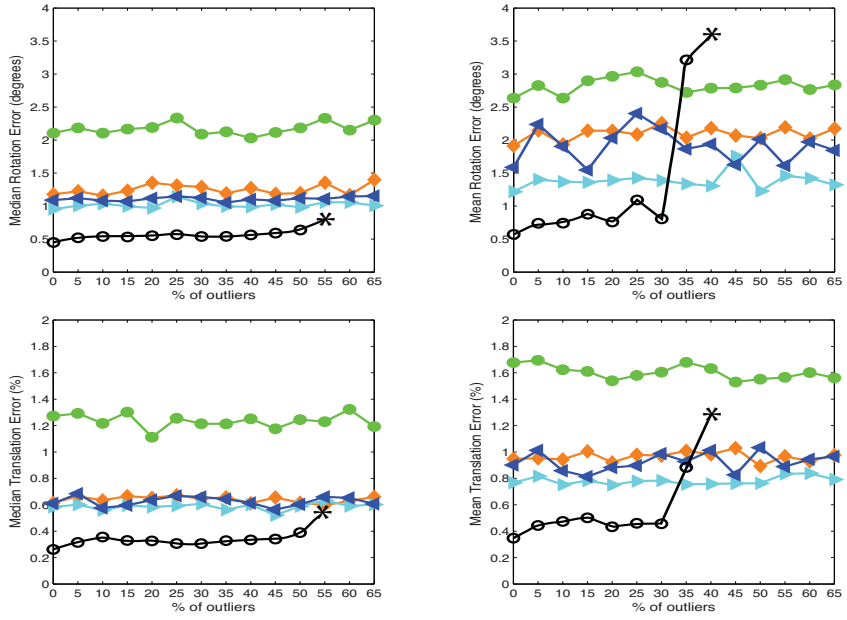


(b) Planar case

Figure 4.9: Synthetic experiments for different levels of outliers for (a) non-planar and (b) planar objects. Varying the % of outlier correspondences. The real inliers are fixed to 100 with $\sigma = 2$. Note that even for small percentage of outliers the obtained solutions are completely wrong.



(a) Non-planar case



(b) Planar case

Figure 4.10: Synthetic experiments, varying the % of outlier correspondences. The real inliers are fixed to 100 with $\sigma = 5$. The symbol $*$ represents the last value before the breakdown point.

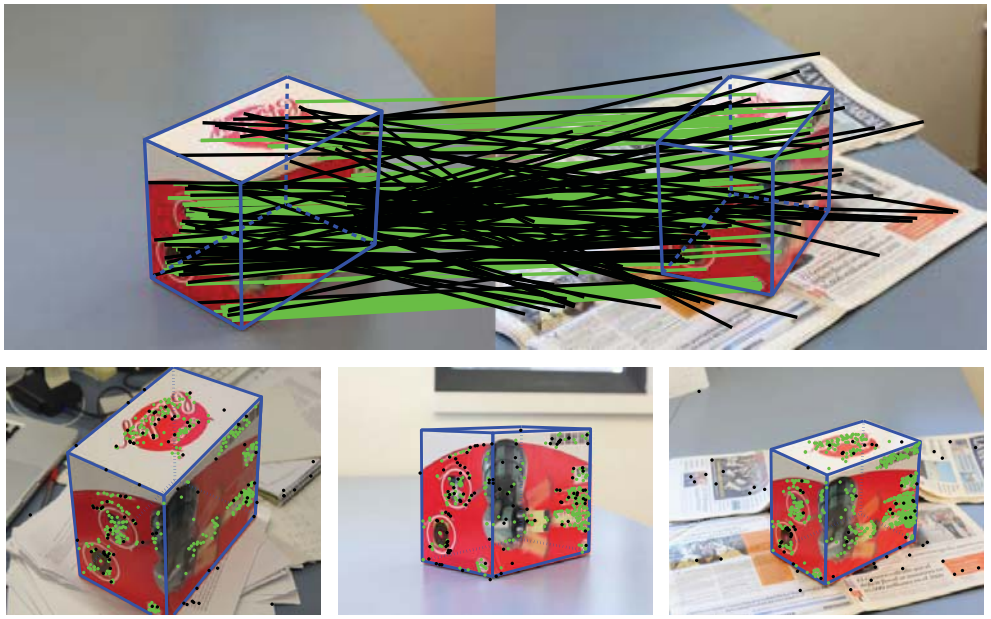


Figure 4.11: Real image examples. Top: Inlier and outlier correspondences found by our method between a model (left) and a test image (rigth). Bottom: Examples of fitting using our method. Green and black dots represent the inliers and outliers respectively.

more than 100x larger than the time required by our approach[1].

Finally, we would like to comment that we are aware that exist more efficient versions of RANSAC [111]. For instance PROSAC [15] exploits priors on the confidence of the detector . Note, however, that we could also incorporate this kind of knowledge in our approach, e.g. by weighting each correspondence based on its confidence. However, for the clarity of the experiments we did not further explore these situations.

Real Images

We also tested our approach in real image sequences corrupted by large amounts of outliers. Interest points, descriptors and initial correspondences are obtained using SIFT [84], which yields around $200-400$ correspondences per image. Figure 4.11(top) shows one sample input image and its set of 248 correspondences, from which 96 are outliers. Our REPPnP can detect these outliers and compute the pose in less than 4ms, $25\times$ faster than any other RANSAC-based method. The results of other detections are plotted on the bottom of Figure 4.11. In Figure 4.12 an example for a planar object is shown, showing in the left the estimated camera pose with respect to the 3D object model. Note that an additional benefit of our approach, is that the outlier removal process is completely transparent to the user, simplifying thus its

[1]The stopping criterion for the RANSAC is based on the number of iterations that guarantee, with a probability 0.99 that at least one minimal solution is outlier free (See. [46], page 119).

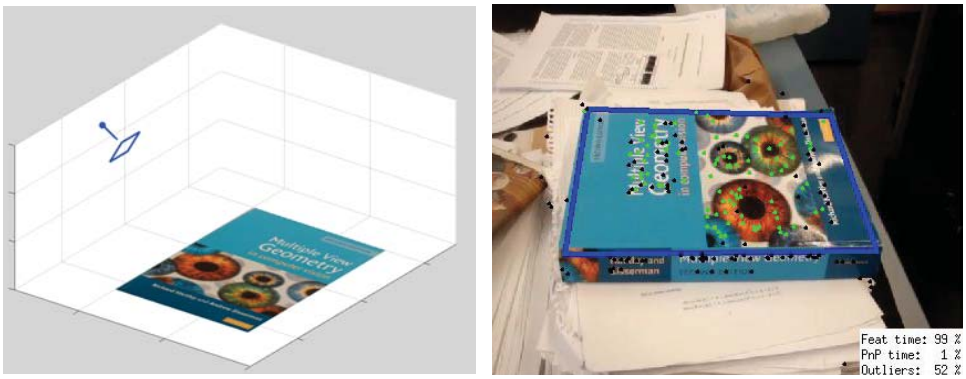


Figure 4.12: Example with a planar object. Estimated camera pose with respect to the 3D object model (left) for the book shown in the image (right). Green and black dots represent the inliers and outliers respectively and blue rectangle the object model projected on the image.

practical utilization.

4.5 Leveraging Feature Uncertainty in the PnP Problem

In this section we propose a solution to the PnP problem which, to the best of our knowledge, is the first one that inherently incorporates into its formulation feature uncertainties. Although all previous PnP solutions assume that correspondences may be corrupted by noise and show robustness against large amounts of it, none of these works considers that the particular structure of the uncertainty associated to each correspondence could indeed to be used to further improve the accuracy of the estimated pose (see Figure 4.13). Specifically, existing solutions assume all 2D correspondences to be affected by the same model of noise, a zero mean Gaussian distribution, and consider all correspondences to equally contribute to the estimated pose, independently of the precision of their actual location.

In order to incorporate into our formulation the feature uncertainties we propose to iteratively minimize an unconstrained Sampson error function [46], which approximates the Maximum Likelihood solution. Furthermore, we also propose a strategy to compute a specific uncertainty model per correspondence in real experiments, by modeling the sensitivities of 2D interest point detectors to different viewpoints. As we will show in both synthetic and real experiments, our approach outperforms the most recent techniques in terms of accuracy while keeping a running time still linear with respect to the number of correspondences.

We next introduce different methods to deal with uncertainties. Afterwards, we reformulate our scalable PnP solution (EPPnP) proposed in Section 4.3 in order to integrate feature uncertainties and estimate the camera pose based on an approximated Maximum Likelihood procedure. Finally, we describe a methodology to model viewpoint-independent 2D feature uncertainties using anisotropic Gaussian distributions.

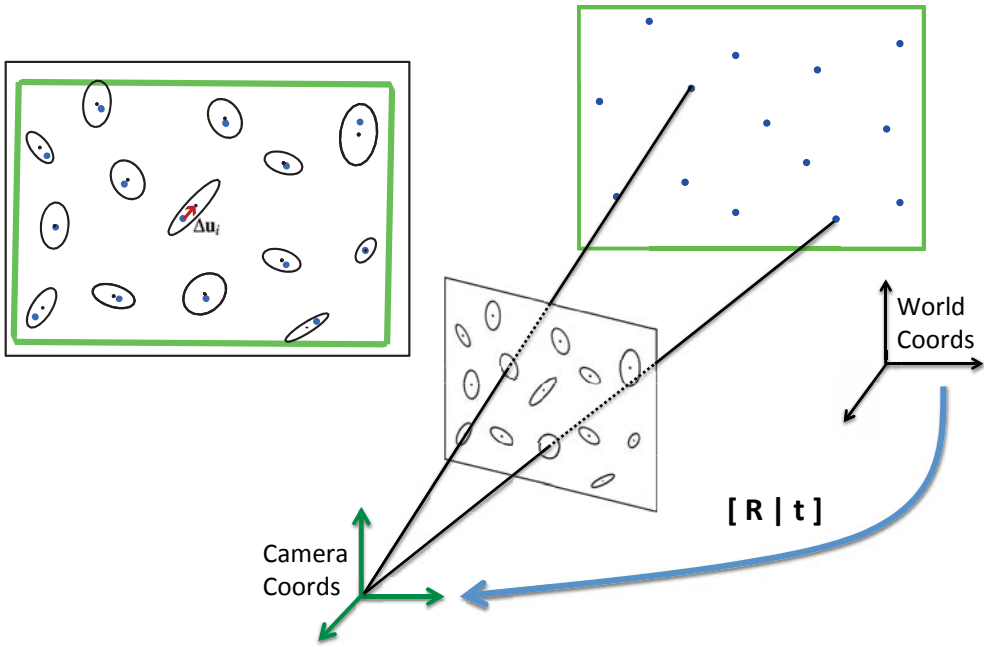


Figure 4.13: PnP problem with noisy correspondences. We assume a set of 2D feature points is given, with a particular noise model for each feature point, which is represented by one 2D ellipse. We also assume the correspondences (black lines) and 3D points (blue points) with respect to a 3D model are known. Our approach estimates a solution of the PnP problem that minimizes the Mahalanobis distances $\Delta \mathbf{u}_i$ shown in the left (red arrow). In the left the green rectangle and blue dots are the true projection of the 3D model.

4.5.1 Related Methods to Deal with Uncertainty

Traditionally, the PnP problem has been applied to small subsets of correspondences yielding closed form solutions to the P3P [40, 21, 62, 34], P4P [32], and P5P [132] problems. Yet, these solutions to the minimal case are prone to be sensitive to noise, being therefore typically used within RANSAC schemes. Noise robustness can be achieved by considering larger sizes of the correspondence set.

Yet, as we have pointed out above, the noise problem has not been directly handled by previous PnP solutions, which simply attenuate its effect by exploiting data redundancy. In contrast, other problems in geometric computer vision, such as Simultaneous Pose and Correspondence [98, 119, 115], Fundamental matrix computation [13, 60] , ellipse fitting [13, 71, 59, 58], do take into account specific models of uncertainty per observed point. In most these approaches, the uncertainty is modeled by a covariance matrix, and Maximum Likelihood strategies are proposed to minimize the Mahalanobis distances between the noisy and the true locations of the point observations. As discussed in [14], estimating the global minima for this kind of problems is impractical. A feasible alternative is to minimize approximated Sampson error functions, for instance by means of iterative approaches such as the Fundamental Numerical Scheme (FNS) [13], the Heterocedastic Errors-in-Variables (HEIV) [71] or projective Gauss Newton [60]. These minimization approaches can be considered as a solution refinement and they need to be fed with an initial solution. Other methods as the Renormalization [57] or the recent Hyper-Renormalization [59] do not need an initial solution, and find a solution by solving a set of *estimating equations* which need not to be derivatives of some cost function.

All these previous approaches, though, are focused on theoretical derivations which are only evaluated over synthetic data where the uncertainty models per point are assumed to be known in advance. The problem of estimating these input models in real data is completely obviated. In this section we will propose a strategy for this in the case of estimating the pose of planar objects.

4.5.2 Integration of Feature Uncertainties in Our Linear Formulation

Let us assume that $\mathbf{u}_i = [u_i, v_i]^\top$ in equation (4.1) represents an observed 2D feature location obtained using an interest point detector. This observed value can be regarded as a perturbation from its true 2D projection $\overline{\mathbf{u}}_i$ by a random variable $\Delta\mathbf{u}_i$. We write this as,

$$\mathbf{u}_i = \overline{\mathbf{u}}_i + \Delta\mathbf{u}_i \tag{4.22}$$

We assume that $\Delta\mathbf{u}_i$ is small, independent and unbiased allowing to model the uncertainty in statistical terms, with expectation

$$E[\Delta\mathbf{u}_i] = \mathbf{0} \tag{4.23}$$

and covariance

$$E[\Delta\mathbf{u}_i\Delta\mathbf{u}_i^\top] = \sigma^2\mathbf{C}_{\mathbf{u}_i} \tag{4.24}$$

where $\mathbf{C}_{\mathbf{u}_i}$ is the known 2×2 uncertainty covariance matrix and σ is an unknown global constant specifying the global uncertainty in the image. Splitting the uncertainty term into two components is motivated because the optimal solution can be obtained ignoring σ [58], making the known uncertainties to be independent of the object size in the image.

From these assumptions, the likelihood of each observed 2D feature location $\mathbf{u}_i$ from its true 2D projection $\overline{\mathbf{u}}_i$ can be expressed as,

$$\begin{aligned} P(\mathbf{u}_i) &= k\cdot\exp(-\frac{1}{2}(\mathbf{u}_i-\overline{\mathbf{u}}_i)^\top\mathbf{C}_{\mathbf{u}_i}^{-1}(\mathbf{u}_i-\overline{\mathbf{u}}_i)) \\ &= k\cdot\exp(-\frac{1}{2}\Delta\mathbf{u}_i^\top\mathbf{C}_{\mathbf{u}_i}^{-1}\Delta\mathbf{u}_i) \end{aligned} \tag{4.25}$$

where k is a normalization constant.

Thus, the Maximum Likelihood solution for the PnP problem is equivalent to minimizing the Mahalanobis distance in equation (4.25) for all n correspondences,

$$\begin{aligned} &\underset{\Delta\mathbf{u}_i,\mathbf{x}}{\arg\min}\ \sum_{i=1}^{n}\|\Delta\mathbf{u}_i\|^2_{\mathbf{C}_{\mathbf{u}_i}^{-1}} \\ &\text{subject to}\quad \mathbf{M}_{\overline{\mathbf{u}}_i}\mathbf{x}=\mathbf{0} \end{aligned} \tag{4.26}$$

where $\mathbf{M}_{\overline{\mathbf{u}}_i}\mathbf{x}=\mathbf{0}$ enforce the 3D-to-2D projective constraints in terms of the noise-free correspondences. Assuming the uncertainty $\Delta\mathbf{u}_i = [\Delta u_i \ \ \Delta v_i]^\top$ to be small, a first order perturbation analysis allows to approximate the projective constraint as,

$$\mathbf{M}_{\overline{\mathbf{u}}_i}\mathbf{x} = \mathbf{M}_{\mathbf{u}_i}\mathbf{x} - \Delta u_i\nabla_u\mathbf{M}_{\mathbf{u}_i}\mathbf{x} - \Delta v_i\nabla_v\mathbf{M}_{\mathbf{u}_i}\mathbf{x} = \mathbf{0} \tag{4.27}$$

where $\nabla_u\mathbf{M}_{\mathbf{u}_i}$ and $\nabla_v\mathbf{M}_{\mathbf{u}_i}$ are the partial derivatives of $\mathbf{M}_{\mathbf{u}_i}$ in equation (4.15) with respect to u and v,

$$\begin{aligned} \nabla_u\mathbf{M}_{\mathbf{u}_i} &= \begin{bmatrix} \alpha_{i1} & \alpha_{i2} & \alpha_{i3} & \alpha_{i4} \end{bmatrix} \otimes \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} \\ \nabla_v\mathbf{M}_{\mathbf{u}_i} &= \begin{bmatrix} \alpha_{i1} & \alpha_{i2} & \alpha_{i3} & \alpha_{i4} \end{bmatrix} \otimes \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \end{aligned} \tag{4.28}$$

Using Lagrange multipliers we can further eliminate the constraints in equation (4.26), and write the problem as an unconstrained minimization of the Sampson error function $J(\mathbf{x})$, namely

$$\arg\min_{\mathbf{x}} \; J(\mathbf{x}) = \arg\min_{\mathbf{x}} \; \sum_{i=1}^{n} \frac{\mathbf{x}^\top \mathbf{M}_{\mathbf{u}_i}^\top \mathbf{M}_{\mathbf{u}_i} \mathbf{x}}{\mathbf{x}^\top \mathbf{C}_{\mathbf{M}_i} \mathbf{x}} \; . \tag{4.29}$$

$\mathbf{C}_{\mathbf{M}_i}$ is the following 12×12 covariance matrix

$$\mathbf{C}_{\mathbf{M}_i} = (\nabla_u \mathbf{M}_{\mathbf{u}_i} + \nabla_v \mathbf{M}_{\mathbf{u}_i})^\top \mathbf{C}_{\mathbf{u}_i} (\nabla_u \mathbf{M}_{\mathbf{u}_i} + \nabla_v \mathbf{M}_{\mathbf{u}_i}) \; , \tag{4.30}$$

which can be interpreted as the uncertainty $\mathbf{C}_{\mathbf{u}_i}$ propagated to the $\mathbf{M}$-space [58].

In order to minimize equation (4.29) we take the derivative of $J(\mathbf{x})$ with respect to $\mathbf{x}$,

$$\frac{\partial J}{\partial \mathbf{x}} = 2 \sum_{i=1}^{n} \frac{\mathbf{M}_{\mathbf{u}_i}^\top \mathbf{M}_{\mathbf{u}_i}}{\mathbf{x}^\top \mathbf{C}_{\mathbf{M}_i} \mathbf{x}} \mathbf{x} - 2 \sum_{i=1}^{n} \frac{\mathbf{x}^\top \mathbf{M}_{\mathbf{u}_i}^\top \mathbf{M}_{\mathbf{u}_i} \mathbf{x} \mathbf{C}_{\mathbf{M}_i}}{(\mathbf{x}^\top \mathbf{C}_{\mathbf{M}_i} \mathbf{x})^2} \mathbf{x} = \mathbf{N}\mathbf{x} - \mathbf{L}\mathbf{x} \tag{4.31}$$

where $\mathbf{N}$ and $\mathbf{L}$ are 12×12 matrices which also depend on $\mathbf{x}$.

Setting the previous equation to zero we obtain $\mathbf{N}\mathbf{x} - \mathbf{L}\mathbf{x} = \mathbf{0}$. As we mentioned in Section 4.5.1, there are several techniques to compute $\mathbf{x}$ from this equation, e.g [13, 71, 60]. We chose the Fundamental Numerical Scheme (FNS) approach [13], which solves iteratively the following eigenvalue problem,

$$(\mathbf{N} - \mathbf{L})\mathbf{x} = \boldsymbol{\lambda}\mathbf{x} \tag{4.32}$$

where the eigenvector with smaller eigenvalue is used to update the solution $\mathbf{x}$. Note that at each iteration, the matrices $\mathbf{N}$ and $\mathbf{L}$ need to be updated with the new $\mathbf{x}$ estimate up to the convergence. For the first iteration, $\mathbf{x}$ is initialized solving the equation system $\mathbf{M}\mathbf{x} = \mathbf{0}$ as is done in Section 4.3.

Finally, once $\mathbf{x}$ is estimated, the PnP problem is solved using the proposed method in Section 4.3.4, which is based on solving the Orthogonal Procrustes problem in equation (4.16) and iteratively refining its solution using the equation (4.18).

4.5.3 Dealing with Feature Uncertainties on Real Images

Estimating 2D feature uncertainties $\mathbf{C}_{\mathbf{u}_i}$ in real images is still an open problem. Most of previous approaches dealing with geometric estimation under noise, just address the problem in synthetic situations where 2D uncertainties are perfectly modeled using Gaussian distributions. In this section, we propose an approach to model stochastically the behaviour of an interest point detector under real camera pose changes.

Our approach starts by detecting features on a given reference view $\mathbf{V}_r$ of the object of interest. Then, we synthesize m novel views $\{\mathbf{I}_1, \ldots, \mathbf{I}_m\}$ of the object, which sample poses around $\mathbf{V}_r$. Each view $\mathbf{I}_j$ is generated by projecting the 3D object model onto the image plane assuming a known projection matrix $\mathbf{P}_{\mathbf{I}_j} = \mathbf{A}[\mathbf{R}_{\mathbf{I}_j} | \mathbf{t}_{\mathbf{I}_j}]$ (without changing the distance between the 3D object center and the camera to avoid changes in the global scale of

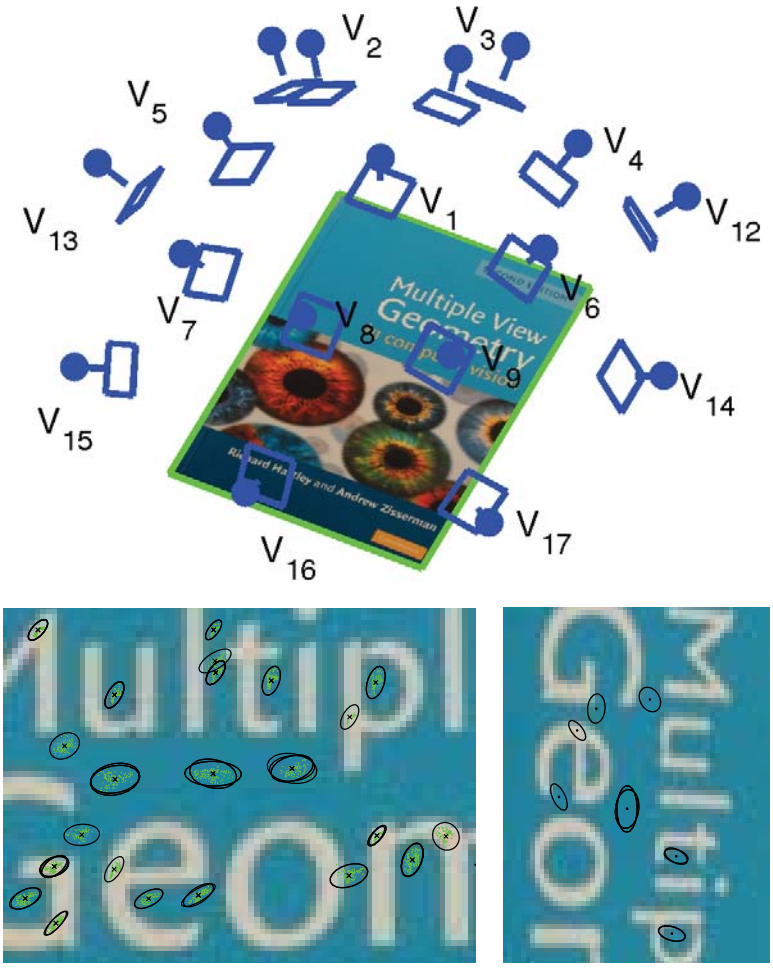


Figure 4.14: Feature uncertainties on real images. Top: Example of a grid of reference views where uncertainties must be estimated. Bottom-Left: Example of feature point clouds (in green) and their Gaussian models (black ellipses) for V_1. Bottom-Right: Results of feature matching and uncertainties alignment against a test image.

the uncertainty σ). We then extract 2D features for each $\mathbf{I}_j$, and reproject them back to $\mathbf{V}_r$, creating feature point clouds (see Figure 4.14 Bottom-Left).

Note that to compute these point clouds onto the reference view we have not performed any feature matching process,i.e., there might be points within one cluster that do not belong to the same feature. In order to resolve this issue and perform a correct clustering of features we make use of the repeatability measure proposed in [90], which measures to what extend do the area associated to each feature overlap. This area could correspond, for instance, to the scale ellipse defined by the SIFT detector [84]. Specifically, if we denote by $\boldsymbol{\mu}_a$ and $\boldsymbol{\mu}_b$ the two feature points, and $\mathbf{S}_{\boldsymbol{\mu}_a}$ and $\mathbf{S}_{\boldsymbol{\mu}_b}$ their corresponding image areas, we compute the

following intersection over union measure,

$$\text{IoU} = \frac{\mathbf{S}_{\boldsymbol{\mu}_a} \cap \mathbf{S}_{(\mathbf{H}^\top \boldsymbol{\mu}_b \mathbf{H})}}{\mathbf{S}_{\boldsymbol{\mu}_a} \cup \mathbf{S}_{(\mathbf{H}^\top \boldsymbol{\mu}_b \mathbf{H})}} \tag{4.33}$$

where $\mathbf{H}$ is the known planar homography relating both regions, and $\cap$ and $\cup$ represent the intersection and union of the regions. Then, the two feature points are deemed to correspond if $1 - \text{IoU} < 0.4$, following the same criterion as in [90].

Once features are grouped we model each cluster i using a Gaussian distribution, with associated covariance matrix $\mathbf{C}_{\mathbf{u}_i}$. Note that this covariance tends to be anisotropic, which means that it is not rotationally invariant with respect to the roll angle. To achieve this invariance we use the angles of the main gradients of the feature regions, similarly as is done by the SIFT detector [84]. Figure 4.14(Bottom-Left and Bottom-Right) show how each $\mathbf{C}_{\mathbf{u}_i}$ is rotated respect to the main feature gradients.

In practice, we found that $\mathbf{C}_{\mathbf{u}_i}$ describes with accuracy the uncertainties when the camera pose of $\mathbf{I}_j$ is near to the camera pose of the reference $\mathbf{V}_r$. This accuracy drops when camera pose moves away. This is motivated because each $\mathbf{C}_{\mathbf{u}_i}$ is computed for $\mathbf{V}_r$, remind that uncertainties are not on the 3D model. In order to handle this, we defined a set of l reference images $\{\mathbf{V}_1, ..., \mathbf{V}_l\}$ under different camera poses and each one with its own uncertainty models. We experimentally found that taking a grid of reference images all around the 3D object every 20° in yaw and pitch angles (see Figure 4.14(Top)), we obtained precise uncertainty models. Before running the proposed PnP method we had to choose an initial reference image to start with. For this, we used our scalable PnP method referred to as EPPnP.

In summary, the algorithm for real images can be split into the following three main steps:

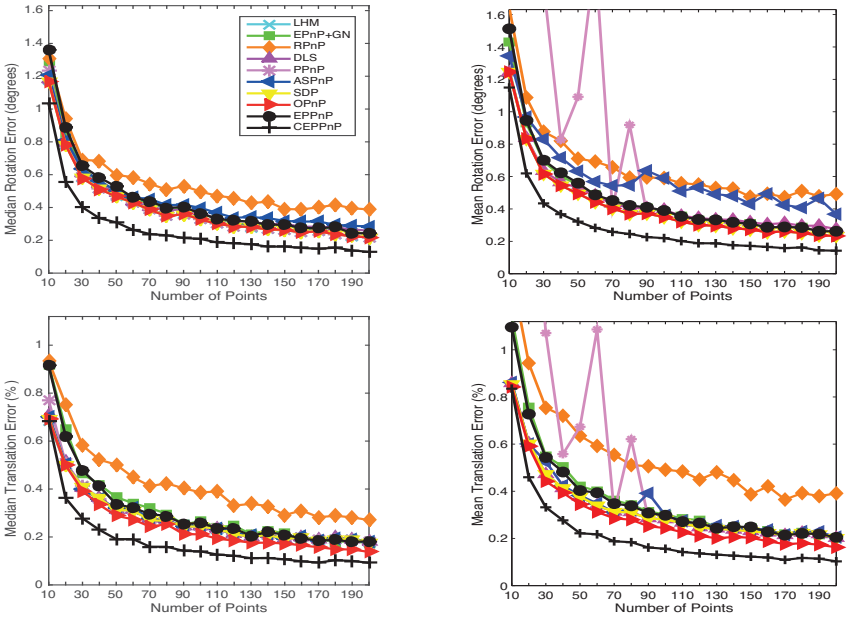
1. Estimate an initial camera pose without considering feature uncertainties using EPPnP. Let $[\mathbf{R}|\mathbf{t}]_{EPPnP}$ be this initial pose.
2. Pick the nearest reference view $\mathbf{V}_k$ taking into account that roll angle is not used to compute the grid of reference images. Find $\mathbf{V}_k$ such that

$$\arg\max_k \left(\frac{\mathbf{c}_k^\top}{\|\mathbf{c}_k\|} \cdot \frac{\mathbf{c}_{EPPnP}}{\|\mathbf{c}_{EPPnP}\|} \right) \tag{4.34}$$

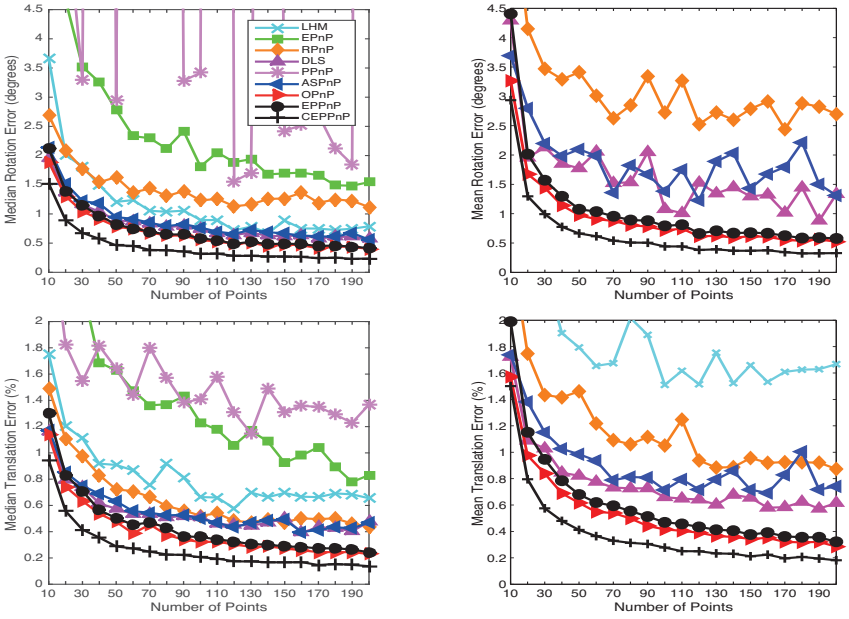
   where $\mathbf{c}_k/\|\mathbf{c}_k\|$ and $\mathbf{c}_{EPPnP}/\|\mathbf{c}_{EPPnP}\|$ are the normalized camera centers in world coordinates, being $\mathbf{c}_k = -\mathbf{R}_k^\top \mathbf{t}_k$ and $\mathbf{c}_{EPPnP} = -\mathbf{R}_{EPPnP}^\top \mathbf{t}_{EPPnP}$.
3. Solve equation (4.32) using the covariances $\mathbf{C}_{\mathbf{u}_i}$ of the reference image $\mathbf{V}_k$, and $[\mathbf{R}|\mathbf{t}]_{EPPnP}$ for initializing the iterative process. The final pose $[\mathbf{R}|\mathbf{t}]_{CEPPnP}$ is obtained using Procrustes, equation (4.16), and its refinement.

4.5.4 Experimental Results

In this section we compare the accuracy and scalability of our novel PnP method against the EPPnP method introduced in Section 4.3 and the state of the art on synthetic and real data.

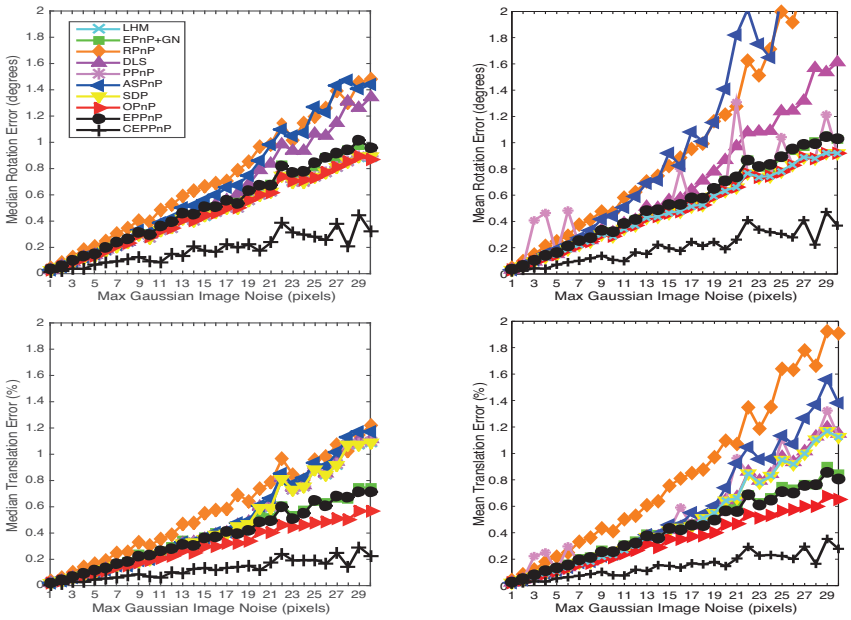


(a) Non-planar case

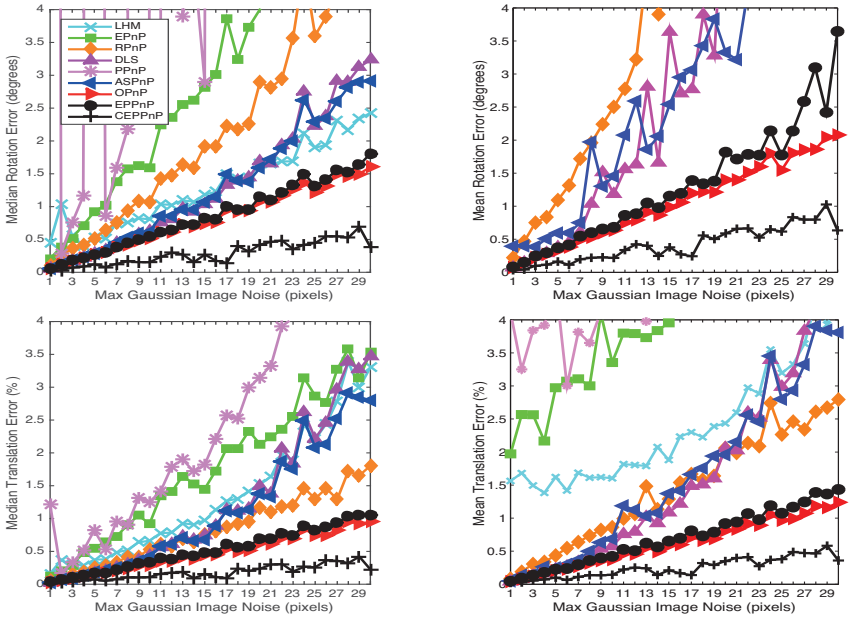


(b) Planar case

Figure 4.15: Synthetic experiments for (a) non-planar objects and (b) planar objects. Varying the number of correspondences.



(a) Non-planar case



(b) Planar case

Figure 4.16: Synthetic experiments for (a) non-planar objects and (b) planar objects. Varying the amount of uncertainty.

Our novel PnP method is written in MATLAB and the source code is publicly available[1] within a toolbox, which contains all the evaluated PnP methods and the code to replicate the experimental results.

Synthetic Experiments

In these experiments we compared the accuracy and running time of our proposed method assuming each uncertainty has a known Gaussian distribution. In experiments we refer to our method as Covariant Efficient Procrustes PnP (CEPPnP) method and we have compared it against the most recent PnP approaches: the robust version of DLS [47], ASPnP [149], OPnP [148], RPnP [75], PPnP [35], EPnP + GN [72], SDP [117], LHM [85] and the EPPnP described in SectionSection 4.3.

We assume a virtual calibrated camera with image size of 640×480 pixels, focal length of 800 and principal point in the image center. We randomly generated 3D-to-2D correspondences, where 3D reference points were distributed into the interval $[-2, 2] \times [-2, 2] \times [4, 8]$. We also added Gaussian noise to the 2D image coordinates. Finally, we chose the ground-truth translation $\mathbf{t}_{\text{true}}$ as the centroid of the 3D reference points and we randomly generated a ground truth rotation matrix $\mathbf{R}_{\text{true}}$. As a metric errors we used the same as in Section 4.4.3 or in [75, 148]. The absolute error is measured in degrees between the $\mathbf{R}_{\text{true}}$ and the estimated $\mathbf{R}$ as $\text{e}_{\text{rot}}(\text{deg}) = \max_{k=1}^{3}\{\text{acos}(\mathbf{r}_{\text{k,true}}^{\top} \cdot \mathbf{r}_k) \times 180/\pi\}$ where $\mathbf{r}_{\text{k,true}}$ and $\mathbf{r}_k$ are the k-th column of $\mathbf{R}_{\text{true}}$ and $\mathbf{R}$. The translation error is computed as $e_{\text{trans}}(\%) = \|\mathbf{t}_{\text{true}} - \mathbf{t}\|/\|\mathbf{t}\| \times 100$. All the plots discussed in this section were created by running 500 independent MATLAB simulations and report the average and median rotation and translation errors.

In order to evaluate the accuracy we propose two different experiments. The first one having into account an increasing number of correspondences and in the second one having into account an increasing level of noise. For the first experiment we evaluate the accuracy from $n = 10$ to 200 3D-to-2D correspondences. We determine 10 different known isotropic Gaussian distributions, with standard deviations $\sigma = [1, ..., 10]$, being each one used to corrupt 10% of the correspondences.

Figure 4.15 shows the results in terms of accuracy for the first experiment in the case of planar and non-planar configurations. Keeping in mind that some correspondences are corrupted by huge noise, σ up to 10, Figure shows that all the methods provide relatively accurate results. However, Figure 4.15 also shows that our CEPPnP method clearly outperforms in all the cases all the other methods.

In the second experiment, depicted in Figure 4.16, for a constant number of correspondences, $n = 100$, the amounts of noise are increased. For each correspondence a uniform random σ is computed between 0 and the maximum image noise value, from 1 to 30. Therefore, the noise corrupting each correspondence follows a different known Gaussian distribution. In the Figure is shown that solutions estimated using CEPPnP method are clearly less sensitive to noise yielding very accurate solutions.

Finally, in Figure 4.17 we show the computation time of all methods, for an increas-

[1] `http://cmtech.upf.edu/system/files/pdf/CEPPnP_Toolbox.zip`

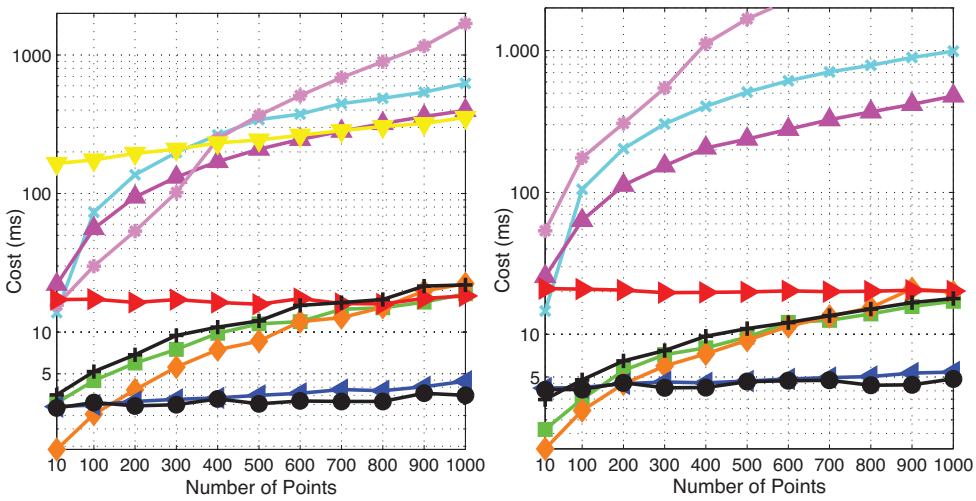


Figure 4.17: Computation time for the non-planar case (left) and the planar case (right) in synthetic experiments varying the number of correspondences. The color codes and line styles are the same as those used in Figure 4.15.

ing number of correspondences, from $n = 10$ to 1000 with known distributions with $\sigma = [1, ..., 10]$. This experiment was done on an Intel Core i7 CPU to 2.7Ghz and all methods are implemented in MATLAB. Note that our CEPPnP method, although does not inherit the constant running time of the EPPnP method, is quite fast and has a linear running time with respect to the number of correspondences.

Real Images

We also tested our approach in real images. Interest points are obtained using SIFT [84] and correspondences are estimated using the repeteability measure [90], which yield around $200 - 400$ correspondences per image. The ground truth used to assess our approach is obtained by randomly generating 3,200 views using homographies with known camera poses. The camera pose for each view is generated with respect to an orthogonal reference view of a planar surface (similar to V_1 in Fig. 4.14 Top) by changing the camera orientations in pitch, yaw and roll angles. As the camera pose of test views moves away from the reference image, we estimate the error with respect to the maximum absolute value of the pitch and yaw angles, from $\pm 10^\circ$ to $\pm 40^\circ$. The roll angle is not restricted since it does not affect directly to the perspective.

We compare CEPPnP against the methods showing the best results in the synthetic experiments of the planar case, namely DLS [47], ASPnP [149], OPnP [148] and our EPPnP method.

Figure 4.18 shows that the pose results (in rotation) we obtain using CEPPnP are remarkably more consistent than all other approaches when using the feature uncertainties modelled

by reference images close from the input image, while the accuracy progressively fall when input image is far. Specifically, our method using the modelled uncertainties clearly outperform state-of-the-art methods when absolute values of pitch and yaw are lower than $\pm 30°$, showing that our uncertainties non-degenerate up to significant projective deformations. This justifies the need of the three-step process we have described in Section 4.5.3. From the results shown in the Figure 4.18 we observe that reference images should be distributed for modelling the uncertainties around the object at every $20°$ in pitch and yaw angles to clearly take advantage of the method on real images. In terms of translation error, all approaches yield almost the same accuracy. This is most probably due to the fact that we have only generated testing images by constraining the camera to be on the surface of an hemisphere on top of the object.

Finally, in Figure 4.19 the camera pose solutions provided by the EPPnP, the OPnP and the proposed CEPPnP methods are shown. In the Figure, the top-left image is the closest one in terms of camera pose with respect to the picture used to compute the 3D model. In the images we observe that the results provided by CEPPnP are quite similar to the others, this is because of although CEPPnP clearly outperform numerically the other methods the improvements in accuracy are difficult to be perceived. However, for extreme camera poses with few correspondences our method provides more accurate solutions as can be seen in the bottom-rigth image in Figure 4.19.

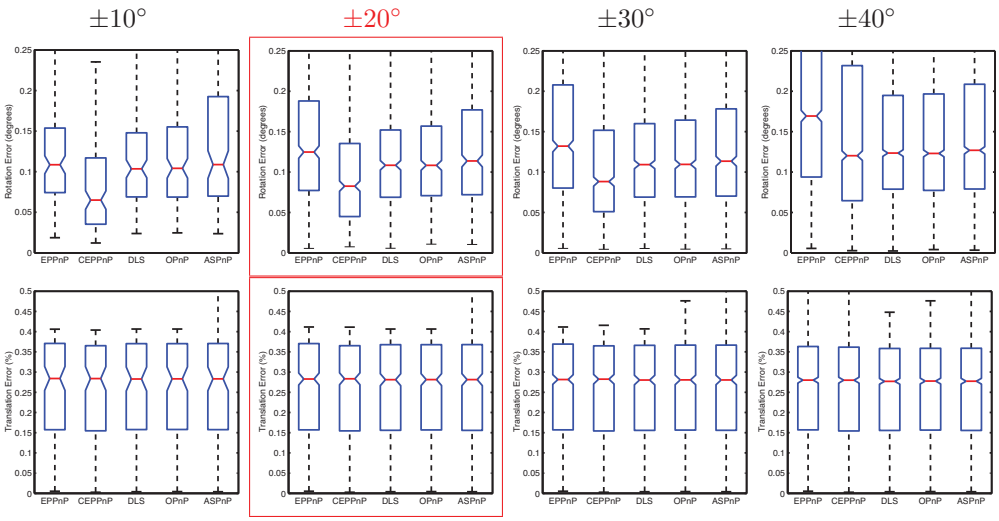


Figure 4.18: Experiments with real images of a planar object. Each column depicts the errors (median and quartiles) as the camera pose move away from the reference view, in rotation (first row) and translation (second row) terms. In red we remark the angular distance we have selected to generate the grid of reference images.

Figure 4.19: Experiments with real images of a planar object. In these images the results provided by the EPPnP (red rectangle), the OPnP (yellow rectangle) and the CEPPnP (blue rectangle) are shown.

4.6 Summary

In this chapter we have dealt with the Perspective-n-Point problem, proposing three novel solutions for three challenging problems. Concretely, we have dealt with the scalability, the robustness to outliers and the uncertainty analysis to provide fast and accurate solutions to the PnP problem.

First of all we have reformulated the EPnP method and we have proposed the EPPnP method, which is very fast and scalable to thousands of correspondences with comparable accuracy to the state-of-the-art methods. Afterwards, we have introduced our novel REPPnP method which is, to the best of our knowledge, the very first PnP method addressing the hard problem with outlier correspondences, avoiding RANSAC-based preprocessing steps. The outlier rejection scheme is integrated within the pose estimation pipeline with a negligible overhead, resulting in an approach completely scalable and with almost constant computational cost for an arbitrary large number of correspondences. Experimental results show that our method compared with RANSAC-based state-of-the-art methods provides speed-ups of up to $100\times$ yielding similar or even better accuracies.

In addition, we have proposed a real-time and very accurate solution to the PnP problem that incorporates into its formulation the fact that in practice the 2D position of not all 2D features is estimated with the same accuracy. Our method approximates the Maximum Likelihood solution by minimizing an unconstrained Sampson error function. Furthermore we have proposed an effective strategy to model feature uncertainties on real images analysing the sensitivity of feature detectors under viewpoints changes. Finally, experimental results show that our approach outperforms the accuracy of state-of-the-art methods in both, synthetic and real experiments.

Chapter 5

Monocular Camera Pose Estimation in Complex Scenarios

In this chapter we propose a robust and efficient method to estimate the pose of a camera with respect to complex 3D textured models of the environment that can potentially contain more than $100,000$ 3D model points. To tackle this problem we follow a top down approach where we combine high-level deep network classifiers with low level geometric approaches to come up with a solution that is fast, robust and accurate. Given an input image, we initially use a pre-trained deep network to compute a rough estimation of the camera pose. This initial estimate constrains the number of 3D model points that can be seen from the camera viewpoint. We then establish 3D-to-2D correspondences between these potentially visible points of the model and the 2D detected image features. Accurate pose estimation is finally obtained from the 3D-to-2D correspondences using our PnP method proposed in the previous chapter that rejects outliers without the need to use a RANSAC strategy, and which is up to 100 times faster than other methods that use it. Two real experiments dealing with very large and complex 3D models demonstrate the effectiveness of the approach.

5.1 Introduction

Robust camera localization is a fundamental problem in a wide range of robotics applications, going from precise object manipulation to autonomous vehicle navigation. Despite being a topic researched for decades it is still an open challenge. There exist approaches based on infrared cameras and high-frequency systems such as Vicon[1], which have shown excellent results for localization and navigation of robots. However, these systems are limited to indoor environments where lighting conditions are controlled.

Another alternative to robustly localize the robot is to equip it with multiple sensors, such

[1] www.vicon.com

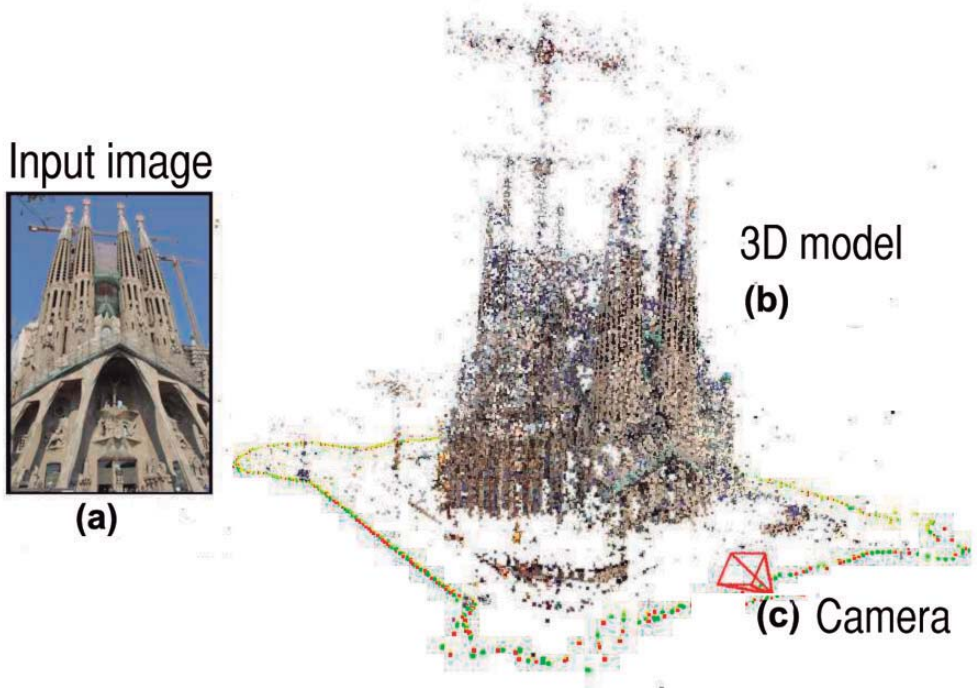


Figure 5.1: Problem definition: Given an input image (a) and a known textured 3D model of the environment (b), the problem is to estimate the pose of the camera that captured the image with respect to the model (c). The main challenge addressed in this chapter is to perform the correspondence of points efficiently and reliably for complex 3D that contain a large number of points. In this example, the model of the Sagrada Familia has over 100,000 points.

as lasers or stereo cameras, and then fusing the data from each of them. Although these systems increase the reliability of the robot for self-localization, they have a negative impact on the computational cost and payload. This is specially critical in some robotic tasks where small and low-cost robots (e.g. aerial robots) are frequently used.

In contrast to these multi-sensor approaches, in this chapter we propose an efficient and robust system based uniquely on a monocular camera and a known pre-computed 3D textured model of the environment, as shown in Figure 5.1. Indeed, the proposed method is able to efficiently estimate the full pose (rotation and translation) of the camera within the 3D map, given solely one single input image. No temporal information is used about the previous poses that can constrain the region of the 3D model where the camera is pointing to. While this makes our problem significantly more complex, it makes the resulting pose estimation robust to issues such as drifting, occlusions of the model during short periods of time, or sudden camera motions.

More precisely, let us assume our 3D model is made of n 3D points, each associated to a visual descriptor representing its appearance. Figure 5.2 shows the overall scheme of the proposed method. In our case these visual descriptors correspond to SIFT features [84]

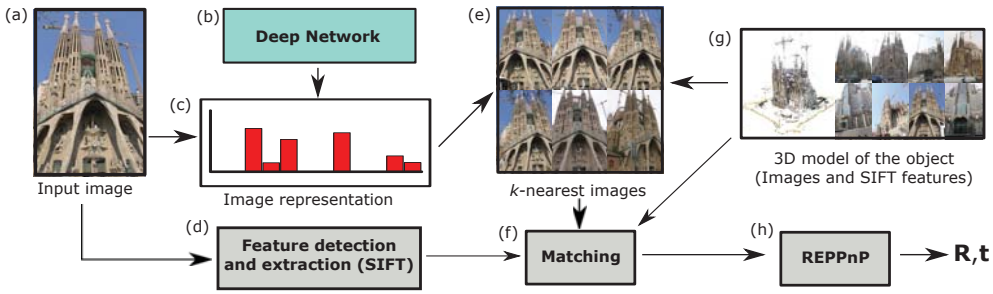


Figure 5.2: Overall scheme of the proposed method for accurate camera pose estimation using highly complex models.

obtained from a set of training images previously used, in an off-line step, to build the model (Figure 5.2(g)). At runtime, the descriptors of the 3D model are compared against the m descriptors extracted from the input image (Figure 5.2(d)) in order to determine a first set of 3D-to-2D correspondences candidates, to then estimate the pose (Figure 5.1(c)). However, solving this correspondence problem has an $\mathcal{O}(n \cdot m)$ complexity, which is extremely costly if we consider that n can be very large (e.g. 100,000 points).

In order to alleviate the computational load, we propose including a preliminary step based on a deep-learning network that yields an approximate initial pose, without the need to explicitly compute correspondences. This method provides the k most similar images (among the training images used to build the 3D model) to the input image (Figure 5.2(e)). Then, the descriptors of these images are used to obtain the 3D-to-2D correspondences by matching (Figure 5.2(f)). Therefore, the correspondence is done with a small part of the model but not with the complete model. And most importantly, this initial pre-selection of most similar images is done in a matter of milliseconds. Since these correspondences may still contain false matches, we get rid of them by using REPPnP, our novel RANSAC-less algorithm for rejecting outliers described in Chapter 4. The experiments have shown that the proposed method not only reduces the computation cost but that it also achieves high accuracy rates in highly complex models.

5.1.1 Overview

The rest of the chapter describes each component of the proposed method. In the next section we split methods for camera pose estimation in two categories, geometric approaches as the ones shown in Chapter 4 and appearance-based methods based on the computation of global descriptors. Afterwards, Section 5.3 explains the methodology we use to build 3D models. Section 5.4 describes the proposed method, where the initial pose estimation is computed using Convolutional Neural Networks (CNNs) and the subsequent pose refinement is done using our REPPnP. In Section 5.5 the method is extensively evaluated over two different models. Finally, Section 5.6 summarizes the main contributions.

5.2 Related Methods for Camera Pose Estimation

Methods for 3D pose estimation from monocular images can be roughly split into two main categories: *Geometric approaches* that rely on local image features (e.g. interest points) and use geometric relations to compute the camera pose parameters; and *Appearance-based methods* that compute global descriptors of the image, and then use machine learning approaches in order to estimate which image within the training set is the closest one to a given input image.

Geometric approaches use local interest points and descriptors to estimate 3D-to-2D correspondences between one input image and one or several reference images registered to a 3D model. PnP solvers such as the ones shown in Chapter 4 are then used to enforce geometric constraints and solve for the pose parameters. On top of that, robust RANSAC-based strategies [15, 98] can be used both to speed up the matching process and to filter outlier correspondences. Yet, while these methods provide very accurate results, they require both the reference and input images to be of high quality, such that local features can be reliably and repetitively extracted. Additionally, if the number of points is very large, the outlier rejection scheme can become extremely slow. Recently, [128] has shown that priors about the orientation of the camera relative to the ground plane can speed up this process. We do not consider these kind of priors for our method, though.

On the other hand, approaches relying on global descriptions of the image are less sensitive to a precise localization of individual features. These methods typically use a set of training images acquired from different viewpoints to statistically model the spatial relationship of the local features, either using one single detector for all poses [48, 76, 127] or a combination of various pose-specific detectors [105, 106, 130, 139, 140]. Another alternative is to bind image features with poses during training and have them vote in the camera pose space [38]. The limitation of these approaches is that the estimated camera pose parameters tend to be inaccurate, and highly depends on the spatial resolution at which the training images have been acquired. The highest granularity of the training set, the more precise can be the estimated camera pose, although the resulting detector is more prone to give false positives.

In this chapter we combine the best of both worlds. On the one hand, we will use a global descriptor based on a deep Convolutional Neural Network (CNN) to get a first estimation of the pose. Deep networks have recently shown impressive results in image classification tasks [66]. This initial estimate will reduce the number of potential 3D-to-2D correspondences, and make geometric approaches applicable. On the geometric side, we will make use of our recent PnP method proposed in the previous chapter, which is scalable to large number of correspondences and inherently incorporates an outlier rejection scheme without the need to run a preprocessing RANSAC-based step. The combinations of both ingredients will result in a powerful and accurate camera pose estimation strategy capable of dealing with very large models.

5.3 Building the 3D Model

Our approach assumes a 3D textured model of the scene to be available. To build these models we used Bundler [123], a structure-from-motion (SfM) system for unordered image collections. Given a set of N images of the scene we seek to model, this package initially extracts SIFT interest points and descriptors of all images, and then simultaneously estimates the N camera poses and 3D structure using bundle adjustment. This is usually a time consuming process that can take a few hours.

For each located 3D point of the model, Bundler provides its SIFT descriptor, its 3D position, its RGB color and the number of images where the point appears. Also, for each image Bundler provides the intrinsic and extrinsic camera parameters, i.e. the calibration matrix, the rotation matrix, $\mathbf{R}$, and the translation vector, $\mathbf{t}$. This data will be used as "ground truth" when evaluating the precision of the algorithm.

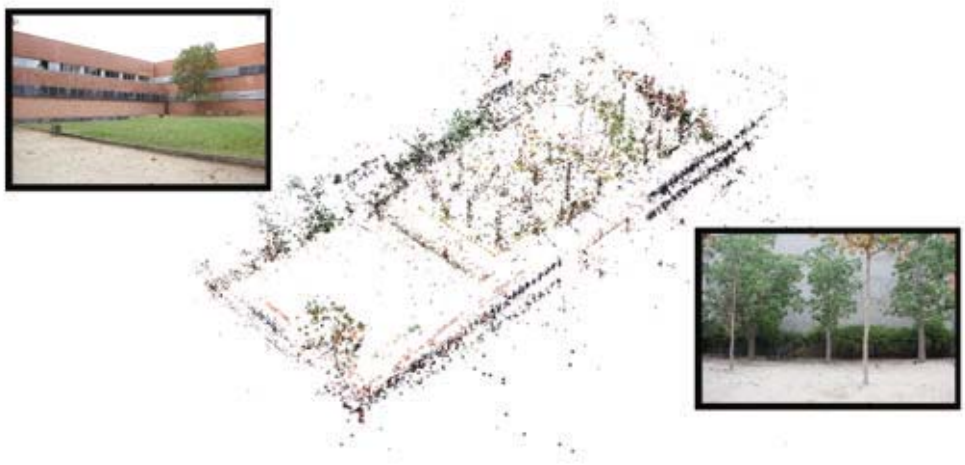


Figure 5.3: Courtyard 3D model built and two sample images.

In this chapter, we used two 3D models: The *Sagrada Familia* dataset (from [107]), composed of 478 images of this church in Barcelona. The resulting 3D model contains $100,532$ 3D points. Figure 5.1(b) shows the model, and plots the retrieved camera pose for each one of the N training set images. The *Courtyard* of the mathematics school dataset is composed of 265 images, which resulted in a 3D model with $30,196$ points. The model and two sample images are shown in Figure 5.3. Note that while the amount of points in the first model is larger, its images present less difference in terms of visual appearance.

5.4 Merging Appearance and Geometric Methods

We next describe how appearance and geometric methods are combined in a top-down coarse-to-fine approach, to tackle the problem of pose estimation from very large models. Let us first formulate our problem:

Assume we are given a 3D model with n points $\{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ and N training images $\{T_1, \ldots, T_N\}$ which have been used to compute the 3D model. Each of these images has an associated pose $\{\mathbf{R}_i, \mathbf{t}_i\}$. Each point $\mathbf{p}_i$ has an associated SIFT descriptor and a visibility list $\mathbf{v}_i$ with the indexes of the training images from where it is seen. Given an input image I, our goal is to accurately compute the pose from where it was acquired.

One straightforward solution would be to extract m 2D features $\{\mathbf{u}_1, \ldots, \mathbf{u}_m\}$ from the input image I and compute 3D-to-2D correspondences $\{\mathbf{p}_i \leftrightarrow \mathbf{u}_i\}$ by just comparing SIFT descriptors. This set of correspondences could then be filtered using a RANSAC+PnP scheme to get rid of outliers and accurately estimate the pose. Nonetheless, since we are considering cases where $n >> m$ those correspondences are prone to contain a very large percentage of outliers, which might dramatically slow down the process.

In this work we propose a two stage strategy that combines the so-called appearance and geometric methods. The former will compute the subset of the training images which is more similar to our input image. This will constrain the set of candidate poses. The latter will use this subset of poses to limit the number of 3D points of the model that are potentially visible, and refine the pose using a geometric approach. These two steps are next discussed.

5.4.1 Coarse Pose Estimation

Given our input image I and training images $\{T_1, \ldots, T_N\}$ we seek to design a fast and robust strategy to get the k training images which are more similar to I. For obtaining this subset we could represent the images using any global image descriptor, e.g, bag of words, GIST, and simply compare such descriptors.

In this chapter, though, we have used the recent generic image-level features obtained from a deep network for image representation. In particular, we use the 4,096-dimensional second to last layer of a Convolutional Neural Network (CNN) as our high-level image representation. The full network has 5 convolutional layers followed by 3 fully connected layers, and obtained the best performance in the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC-2012). The network is trained on a subset of ImageNet [22] to classify 1,000 different classes. We use the publicly available implementation and pre-trained model provided by [52]. The features obtained with this procedure have been shown to generalize well and outperform traditional hand-crafted features, thus they are already being used in a wide diversity of tasks [124].

Comparing the resulting vectors of this representation we obtain, for each input image I, a subset of the k most "similar" training images $\{T_1, \ldots, T_k\}$. These images may or may not be clustered in a similar region of the space. Indeed, the value of k is chosen sufficiently large to ensure that the subset of "similar" images contains at least one image which is actually close

to I, i.e, the set of poses associated to these images represents just a very rough estimation of the ground truth pose. We next explore them and refine the pose using a geometric approach.

5.4.2 Fine Pose Estimation

The k closest training images provide a coarse estimate of the camera pose. To increase the accuracy we adapted the REPPnP method proposed in Chapter 4, which simultaneously allows to discard outlier correspondences while estimates the camera pose.

Standard PnP approaches assume the 3D-to-2D correspondences to be free of outliers. Therefore, when dealing with real images these methods need an outlier rejection preprocessing step (e.g. RANSAC + P3P), which may significantly reduce the overall computational efficiency, as we show in Chapter 4.

Given c 3D-to-2D correspondences $\{\mathbf{p}_i \leftrightarrow \mathbf{u}_i\}$ and the camera internal calibration matrix $\mathbf{A}$, PnP methods build upon the perspective constraint for each 2D feature point i,

$$d_i \begin{bmatrix} \mathbf{u}_i \\ 1 \end{bmatrix} = \mathbf{A} \, [\mathbf{R}|\mathbf{t}] \begin{bmatrix} \mathbf{p}_i \\ 1 \end{bmatrix} \ , \tag{5.1}$$

where d_i is the depth of the feature point.

REPPnP reformulates the previous equations as a low-rank homogeneous set of equations $\mathbf{M}\mathbf{x} = 0$, where $\mathbf{M}$ is a $2c \times 12$ matrix representing the perspective constraints for all c correspondences. In Chapter 4 is shown that the solutions $\mathbf{x}$ can be estimated assuming that the rank of the null-space of $\mathbf{M}$ is equal to 1. Once $\mathbf{x}$ is estimated, the camera pose $[\mathbf{R}|\mathbf{t}]$ is solved using a generalization of the Orthogonal Procrustes problem (see equation (4.16)), combined with a projected gradients optimization. In REPPnP, the outlier rejection is done by iterating the estimation of $\mathbf{x}$. At each iteration, those equations in $\mathbf{M}$ that after being projected onto $\mathbf{x}$ provide the lowest algebraic errors are chosen. This procedure shows convergence with up to 50% of outliers and requiring up to two orders of magnitude less computational time than standard RANSAC + P3P + PnP algorithms.

In order to adapt REPPnP to the case of this chapter with large scenarios, we propose building the $\mathbf{M}$ matrix as k different parts, each coming from one of the k similar images retrieved in the previous stage. In cases where the outlier rate is above 50% a standard RANSAC + P3P approach is used before applying REPPnP to refine the camera pose, therefore sometimes the computational cost is increased since both methods must be executed. In fact, according as k is increased the probability of getting more than 50% of outliers also increases, mainly because the k images tend to represent disjoint parts of the model. Thus, in order to speed up the proposed method we could detect these disjoint image sets and apply REPPnP in each one, however we do not consider crucial this issue to validate our coarse-to-fine approach and we leave this work for the future.

k	**Time (s)**	**# 3D points**	**# Correspondences**	e_{rot}	e_{trans}
		Sagrada Familia model			
All	45.6694	100532	470	0.0124	0.0231
6	12.3344	24855	424	0.0178	0.0305
8	14.1634	29046	436	0.0176	0.0306
12	16.9470	35422	447	0.0177	0.0303
16	20.2644	43005	452	0.0178	0.0306
		Courtyard model			
All	50.2912	30196	379	0.0279	0.0449
3	15.5296	6189	296	0.0255	0.0441
6	21.2358	10038	322	0.0267	0.0476
8	24.3889	12134	328	0.0212	0.0343
12	28.9528	15152	334	0.0206	0.0360
16	33.5211	18145	337	0.0303	0.0538

Table 5.1: Performance of the proposed approach in the two models for different values of the parameter k. We also consider the case when *all* the model points are considered.

Concretely, we propose to solve,

$$\left[\mathbf{M}_1^\top \mathbf{M}_2^\top ... \, \mathbf{M}_k^\top\right]^\top \mathbf{x} = 0 \tag{5.2}$$

where $\mathbf{M}_j$ for $j = [1, 2, ..., k]$ represent the homogeneous equations obtained as in Chapter 4 by matching the 2D points $\mathbf{u}_i$ with the 3D points $\mathbf{p}_i^j$ which are visible (according to $\mathbf{v}_, \ldots, \mathbf{v}_k$) in each of the k nearest images.

With the proposed method, taking $k = 6$, we decrease the number of 3D points to be matched to one quarter of the total in the Sagrada Familia scenario and to one third in the Courtyard scenario, increasing the number of inlier correspondences. Table 5.1 shows the performance of our approach for both models and different values of k.

5.5 Experimental Results

In this section, the efficiency and accuracy of the proposed method will be evaluated using the two models presented in Section 5.3. Previously, we will show how our image retrieval algorithm performs on the test datasets.

For the experiments, both datasets have been evenly split into two disjoint sets. One of them is used as training set for the construction of the model, and the other one as testing set for the evaluation of the method. In the case of the *Sagrada Familia* dataset we use 239

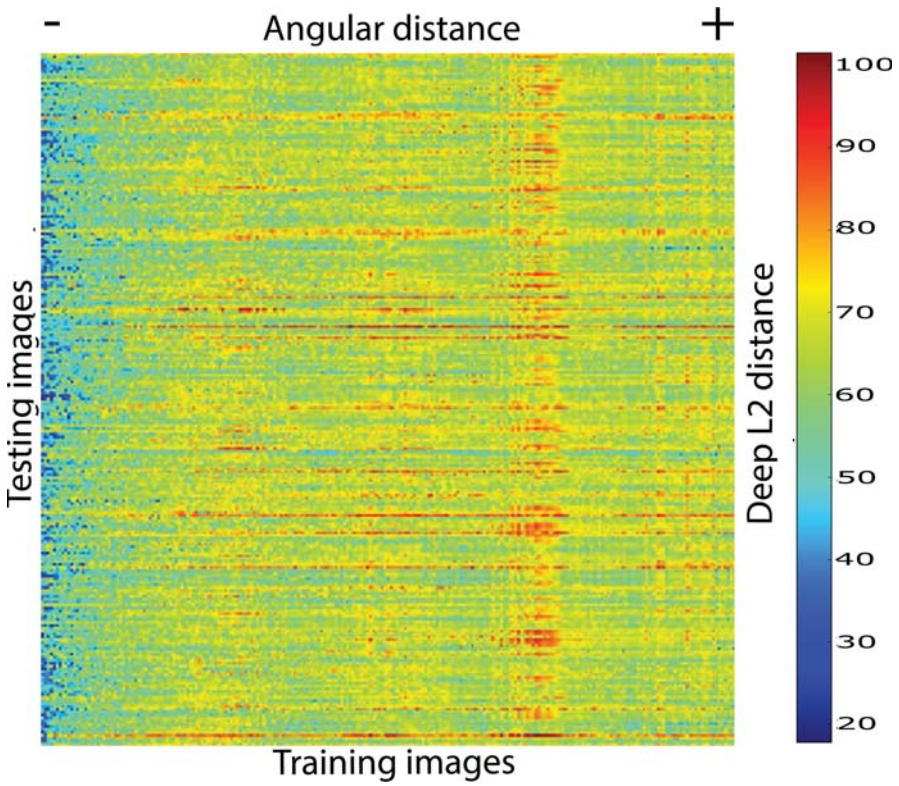


Figure 5.4: Comparison of the L2 distances between distance deep network features and the known angular distance between the test images and training images. Each row is sorted according to the angular distance. We can see a strong correlation between images that are close to each other and their descriptors obtained from the deep network.

randomly selected images as training set and 239 images as evaluation set. On the other hand, in the *Courtyard* dataset we use 132 images as training set and 133 as evaluation set. The Sagrada Familia dataset was captured in two different days (one sunny day and one cloudy day), so the images contained in the dataset represent two full turns around the church. In the case of the Courtyard the dataset represents one single turn.

5.5.1 Image Retrieval for Coarse Pose Estimation

In order to evaluate the quality of the image retrieval using the deep network descriptors, we have plotted in Figure 5.4 the Euclidean distance between the descriptors for each pair of test/train images in the Sagrada Familia dataset. For each test image (row) the training images are ordered by angular distance of the rotation matrix. The accumulation of bluish regions on the left hand side of the graph indicates that the distance between descriptors increases as we move away from the original viewpoint, and thus confirms a high confidence in the image

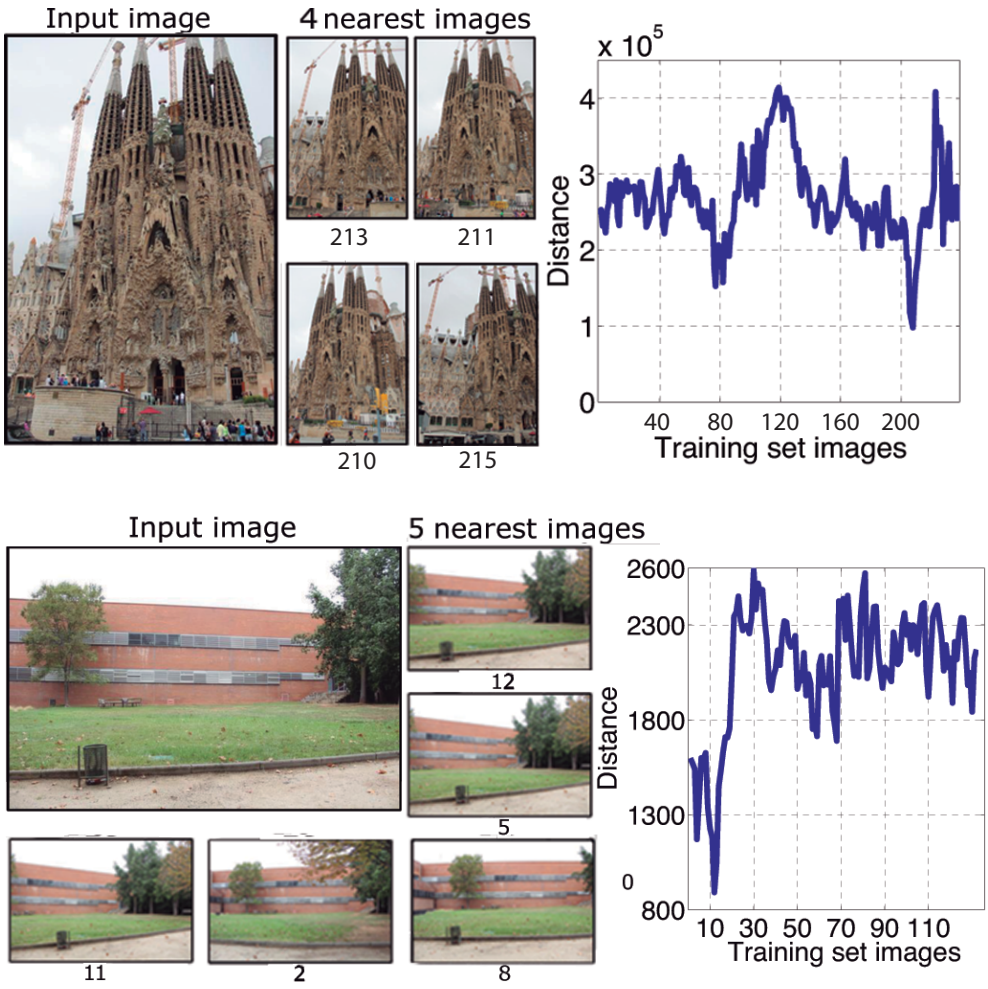


Figure 5.5: Example of one input image of the Sagrada Familia (top) and one input image of the Courtyard (bottom) in the validation sets. Next to the input images similar images in the training set selected by our image retrieval algorithm ($k = 4$ and $k = 5$) and a plot showing the distance to the training images.

retrieval process we use.

As illustrative examples, in Figure 5.5 we show one sample query for each dataset. We plot both the input image and the closest images in the training set according to the distances between the deep network descriptors. In both cases nearly all selected images present similar poses to the test image. We want to remark that the plots representing the distances clearly show the performance of the method, in the top plot the pick around the training sample 80 is due to the images taken from a similar camera pose but in a sunny day.

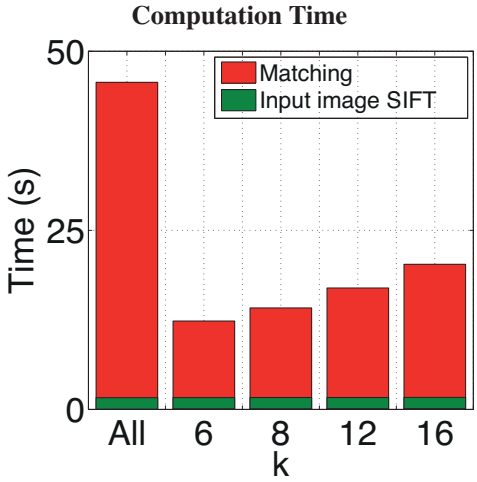


Figure 5.6: Analysis of computation time. Left: Computation time of the approaches with different values of k. The case *All* is equivalent to not doing any appearance based pre-computation over the 3D model.

5.5.2 Efficiency Assessment

In order to have a clear picture of the cost of our method, we have measured the computational time for each of the different stages: SIFT feature extraction on the input image, image retrieval through our deep learning approach, descriptor matching and outlier removal along with pose estimation done by our fine pose estimation method. To compare the computation time of the approach against a typical baseline, we have replaced our fine camera pose estimation in our pipeline, by RANSAC + P3P combined with OPnP [148], one of the fastest and most accurate approaches in the state of the art as we show in the experimental section in Chapter 4.

Average times over the complete test set are given in Figure 5.6. As can be observed, the coarse filtering of model images reduces significantly the computation time of the geometrical estimation of the camera pose. It can also be seen in the figure that the most time consuming step of the whole pipeline is by far the SIFT descriptor matching. This step is performed using the MATLAB implementation of the VLFeat open source library [138].

It is, therefore, critical to obtain a good set of neighbouring images: the better it is, the less model descriptors will be required in the geometrical estimation step. In our experiments, this coarse to fine approach led to 75% reduction of the computational time, but it could be even more significant on larger models. This is made possible by the robustness of the deep learning approach. It can be seen that the other stages of the pipeline are negligible.

Finally, in Table 5.2 we show the average time required to compute the deep network descriptors and the bag of visual words from a new image (top), and the REPPnP and RANSAC+P3P+OPnP (bottom). While a deep network descriptor can be extracted directly

Model	Bag of Words	CNN Features	Gain (%)
Sag. Familia	1.63	0.0208	98.72
Courtyard	5.48	0.0207	99.62
k	**RANSAC+OPnP**	**REPPnP**	**Gain (%)**
6	0.0376	0.0050	86.70
8	0.0421	0.0057	86.46

Table 5.2: Time values in seconds for the different methods evaluated. The upper part of the table compares the two methods evaluated for the coarse estimation of the pose (Bag of words vs. Deep network features). The lower part reports the computation time of the methods used for fine pose estimation (RANSAC + OPnP vs. REPPnP). These results are obtained by computing the mean computation time for all the evalaution images.

from an input image in a matter of milliseconds, the bag of visual words requires first computing SIFT descriptors of that image (on the order of seconds, depending on the size of the image), and then finding the corresponding visual word for each descriptor with a precomputed dictionary. Regarding the geometrical estimation part, by using REPPnP we are able to reduce the time required by a factor of 7.5. We want to remark that in this experiment we are comparing the cases where the outlier rate is below the 50%, in the other cases the time required for our fine camera pose estimation method is similar as for RANSAC+P3P+OPnP.

5.5.3 Accuracy

The accuracy is computed as the rotation and the translation errors in the calculated pose. The rotation error is estimated using quaternions as $e_{\text{rot}} = \|quat(\mathbf{R}) - quat(\mathbf{R}_{\text{true}})\| / \|quat(\mathbf{R}_{\text{true}})\|$, and the translation error as $e_{\text{trans}} = \|\mathbf{t} - \mathbf{t}_{\text{true}}\| / \|\mathbf{t}_{\text{true}}\|$. The estimated pose is $\{\mathbf{R}, \mathbf{t}\}$, and $\{\mathbf{R}_{\text{true}}, \mathbf{t}_{\text{true}}\}$, corresponds to the groundtruth given by the Bundler algorithm, as mentioned in Section 5.3. As observed in Table 5.1, the errors found using the coarse to fine approach are comparable to the ones obtained with the complete model. The rotation and translation errors using only the deep network descriptors is ≈ 10 times bigger than with the coarse to fine approach.

Figure 5.7 shows a qualitative comparison of the reprojection of the interest point locations obtained using SIFT on four input images using the $\mathbf{R}$ and $\mathbf{t}$ provided by REPPnP and RANSAC+OPnP, along with the groundtruth reprojection (given by Bundler). In Table 5.3 the rotation and translation errors for these four images are shown. In all experiments, we obtain similar results when comparing with RANSAC+OPnP. We want to remark that when there are more than 50% of outliers, REPPnP is combined with RANSAC+P3P.

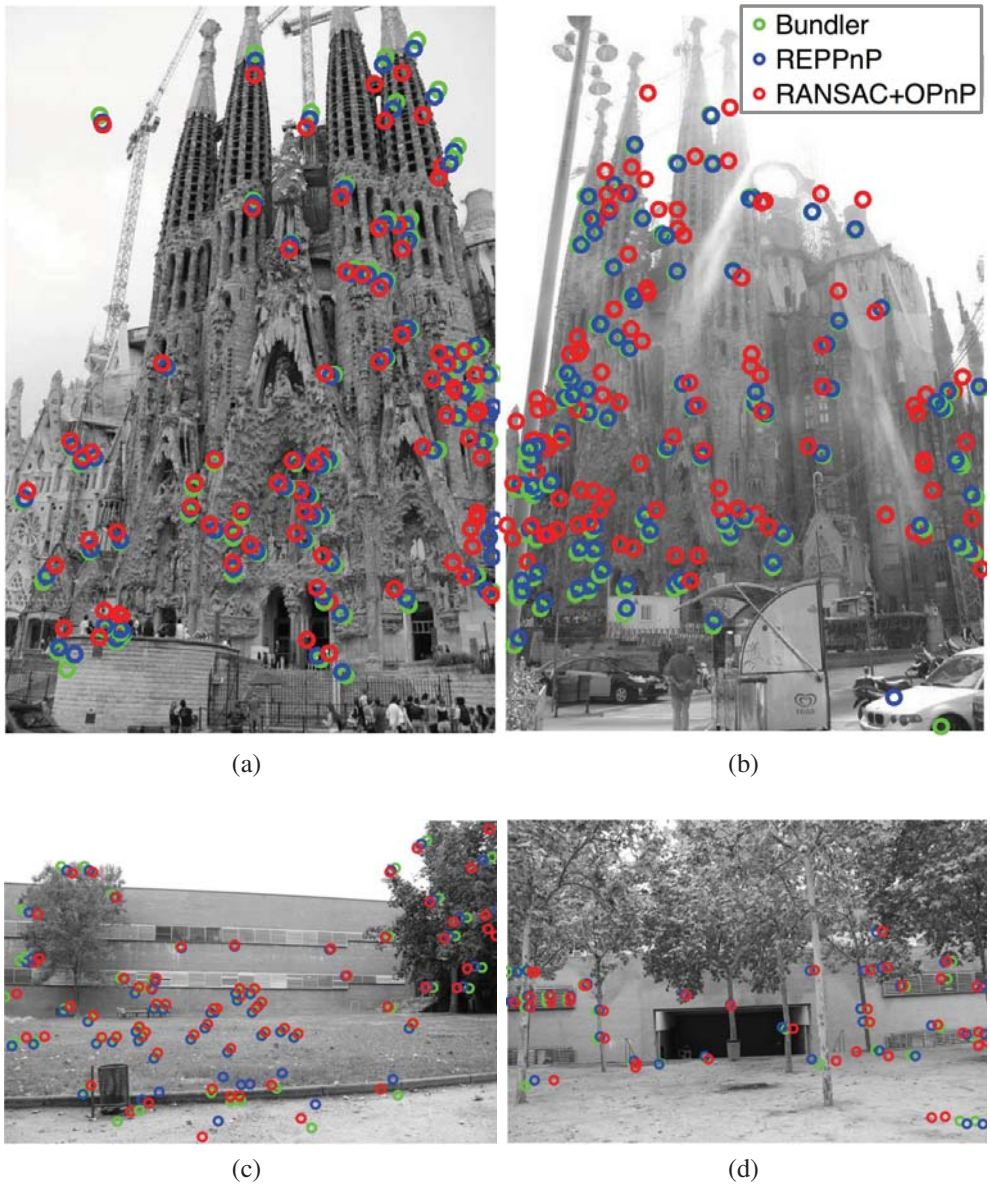


Figure 5.7: Examples of pose estimation results for the Sagrada Familia (top) and Courtyard (bottom) models. For each image we show the reprojected 3D coordinates of the SIFT points using the groundtruth **R** and **t** (from Bundler) and the ones estimated by both REPPnP and RANSAC+OPnP with $k = 6$.

Rotation errors

	REPP*n*P	**RANSAC+OP*n*P**
(a)	0.0068	0.0007
(b)	0.0042	0.0812
(c)	0.0033	0.0024
(d)	0.0012	0.0079

Translation errors

	REPP*n*P	**RANSAC+OP*n*P**
(a)	0.0306	0.0383
(b)	0.0109	0.1497
(c)	0.0091	0.0091
(d)	0.0109	0.0267

Table 5.3: Rotation and translation errors for the images shown in Figure 5.7 using REPPnP and RANSAC+OPnP with $k = 6$.

5.6 Summary

In this chapter we have proposed a new method to estimate the camera pose on large scale 3D models combining a purely appearance based technique with a geometrical approach. The proposed coarse-to-fine approach allows to our method reduce significantly the time required to estimate the camera pose. By using first global image appearance we reduce the number of matches to test but at the same time by applying a PnP solver over the candidate images the error in our camera pose estimation becomes nearly negligible.

In this work we have shown how it is possible to leverage deep learning techniques to improve appearance based approaches for the robotic community. It is remarkable that we are able to perform accurate pose estimation over hundreds of thousands of points in a few seconds per image, especially considering that we are using a MATLAB implementation.

Chapter 6

Conclusions and Future Work

In this thesis we have dealt with the effect of perspective in three main tasks, proposing robust algorithms under different viewing conditions from a low-level to high-level point of view. From a high-level point of view we propose algorithms to recover in real-time accurate camera pose parameters from a single perspective image. These algorithms rely on low-level tasks, which analyse the perspective from a local point of view. Concretely we tackle two low-level tasks, interest point detection and interest point description, enhancing their invariance under different viewing conditions.

In this chapter we summarize the main contributions of each chapter with a discussion of the proposed key features, starting from the low-level tasks. In addition, for each chapter we present open research lines for future work. Finally, we discuss how all the algorithms proposed can be combined.

6.1 Interest Point Detection

This thesis starts dealing with multi-scale interest point detection. We have shown that state-of-the-art detectors tend to over-represent local image structures associating several interest points for each one. This fact promotes not completely distinguishable interest points, losing the ability to clearly describe the uniqueness of each local image structure. In addition, we have shown that common operators used to build multi-scale representations are approximations of a more generic framework, which analyses the image curvatures from a differential geometry point of view.

We have focussed on improving the geometrical invariance, discriminative power and uniqueness of the detected interest points. To this end we have proposed sparse detectors of blobs and corners, which try to describe each image structure with a single interest point. The cornerstones of our detectors are two, an accurate multi-scale representation obtained using Gaussian curvature as operator and our novel bottom-up algorithm to analyse the movement and evolution of local image structures over scale. Similarly to [79], for each local image

structure we have proposed to estimate the spatial location and scale where it was created, annihilated and merged with other image structures.

We have proposed two scale invariant detectors using this approach, in one side our blob detector based on finding elliptic image regions and in the other side our corner detector based on finding hyperbolic image regions. Since both detectors are extracted from the same multi-scale representation with a minimal difference in the evolution algorithm, we have also considered a full detector, which is able to detect blobs and corners simultaneously with negligible computational overhead.

We have also proposed an affine invariant detector of blobs, which captures the intrinsic geometry of the blob structures. The shape and location of each scale invariant blob is refined by fitting an anisotropic Gaussian function, which minimizes the error with respect to the underlying image and simultaneously estimates both the shape and location, by means of a non-linear least squares approach.

The experimental evaluation shows for all our detectors a comparable performance to the state-of-the-art detectors in terms of repeatability and matching scores. Even having into account that in the experimentals for our full detector comparisons are performed assuming the typology of the interest points is unknown. In terms of precision and recall our scale invariant detectors tend to clearly outperform the state-of-the-art detectors while our affine invariant detector obtain better scores in 3 of 8 sequences, similar to the best scores in 3 sequences and worse results in 2 sequences. For image registration we have shown our scale invariant detectors obtain slightly better accuracy than the other detectors while our affine invariant detector obtains similar accuracy.

In addition, experiments show our detectors provide sparse interest points, which clearly avoid redundant detections that over-represent specific image regions. Therefore, our detectors improve the spatial distribution of interest points and increases their distinctiveness, allowing to decrease the computational cost of future matching tasks.

Comparing scale and affine invariant detectors we have concluded that affine detectors only outperform scale detectors in image pairs with heavy viewpoint changes. Thus, in general their usefulness is focused in image sequences with extreme viewpoint changes. This point confirms the appreciation of Lowe in [84].

Future Work : Our full detector of corners and blobs can over-represent some corner-like local image structures since both detectors are computed independently on the same multi-scale representation, Figure 2.6 show that corner structures can be considered as elliptical or hyperbolic regions. Our detector does not take into account that one corner can be described simultaneously as an hyperbolic and an elliptic region. This issue has been tackled in bibliography, [27] proposes to detect the parabolic point located between the elliptic and the hyperbolic one, however this strategy seems to be unstable on real images because is not easy to know when a blob detection represents a corner and even knowing that, it is hard to know its corresponding corner detection.

6.2 Interest Point Description

Once the interest points are detected they are described, typically as a vector, to be matched with other points. We have shown that our machine learning based descriptors provide consistent performance gains over the state of the art, particularly our approach clearly outperforms hand-crafted descriptors such us SIFT, which have difficulties to deal with severe viewing conditions such as small spatial location errors, in-plane perspective variations or even those perspective variations produced by image regions that are not planar in the real 3D scene and can even be partially affected by self-occlusions.

Similarly to the situation for image-level tasks such as classification, where deep learning has become the dominant paradigm in detriment of hand-crafted ones. At local image region level, we have proposed a novel framework to learn a deep Convolutional Neural Network (CNN) using a Siamese network architecture, which employs two CNNs with identical parameters fed with corresponding and non-corresponding pairs of image regions. Each region is propagated forward through the network, providing two CNN outputs that must be similar in terms of L_2 error for corresponding pairs and dissimilar otherwise. In addition we have proposed a novel training scheme, which yields large performance gains in image region retrieval, based on aggressive mining for both positive and negative region pairs. Since in our datasets there are a huge number of corresponding pairs positive mining is needed, biasing the training towards corresponding pairs that are hard to classify.

We have carried out a thoroughly evaluation of multiple network architectures, configurations and hyper-parameters. We have shown that the convolutional network with best performance is fully-convolutional containing three convolutional layers, positive and negative aggressive mining, hyperbolic tangent as activation function, L_2 pooling and subtractive normalization. During the training and testing we have evaluated our descriptors using L_2 distance in order to reflect region similarity, therefore our descriptor can be used as a drop-in replacement for any task involving SIFT.

Results on several datasets show that the descriptors we learn yield remarkable performance improvements compared to state of the art, and a great ability to deal with severe viewing conditions such us scaling and rotation, perspective transformation, non-rigid deformation and illumination changes. It is remarkable that some of these viewing conditions have been evaluated on very specific datasets, without a specific training of our descriptor. Concretely, our descriptor, which is trained on datasets without non-rigid deformations, outperforms DaLI [121], which is a very specific and powerful descriptor for non-rigid deformations and illumination changes.

Future Work : We identify multiple directions for future research. Our descriptors are learnt using grayscale regions with a restricted size (64×64 pixels), we think is interesting to study the descriptor performance using color cues and deeper networks assuming bigger image regions. On the other hand, in our experimentals all the image regions are extracted using DoG detector, therefore our descriptor could learn in some way specific models for specific detectors. we should analyse the generalization ability under different detectors. In addition, if we find strong relations between both, we could also try to learn a regressor in order to refine the spatial location of interest points.

6.3 Robust Monocular Camera Pose Estimation

In camera pose estimation from monocular images some issues and challenges remain to be studied or have been partially shelved. Concretely, we have dealt with three of those challenges –the scalability to thousands of correspondences, the robustness to outlier correspondences and the noisy estimations induced by feature uncertainty– to provide fast and accurate solutions to the so called Perspective-n-Point (PnP) problem.

First of all we have proposed our fast and scalable PnP method. Our approach is based on reformulating the EPnP [97] method, converting its linear cost in an approach completely scalable and with almost constant computational cost for an arbitrary large number of correspondences (e.g. 1000 correspondences are processed in 3ms in our experiments). We have also improved the accuracy of EPnP by iteratively computing the camera pose having into account that the true solution must lie in a known subspace. We have achieved comparable accuracy to the best state-of-the-art methods.

Afterwards, we have introduced our PnP method which is, to the best of our knowledge, the very first PnP method addressing the hard problem with outlier correspondences, avoiding RANSAC-based preprocessing steps. In our approach the outlier rejection scheme is integrated within the pose estimation pipeline of our previous method with a negligible overhead. We formulate our method as a low-rank homogeneous system where the solution lies on its 1D null space, therefore outlier correspondences can be determined as those equations in our linear system which perturb the null space. Experimental results show that our method compared with RANSAC-based state-of-the-art methods provides speed-ups of up to $100\times$ yielding similar or even better accuracies.

Finally, we have proposed a real-time and very accurate solution to the PnP problem that incorporates into its formulation the fact that in practice the 2D position of not all 2D features is estimated with the same accuracy. Our method approximates the Maximum Likelihood solution by minimizing an unconstrained Sampson error function. Furthermore we have proposed an effective strategy to model feature uncertainties on real images analysing the sensitivity of feature detectors under viewpoint changes. We have shown that our approach outperforms the accuracy of state-of-the-art methods in both, synthetic and real experiments.

Future Work : There are multiple directions for the future works. Now, our robust to outliers method has a breakdown point between 50% and 60%, to make robust our method to larger percentage of outliers we consider two alternatives. We plan to integrate additional priors (e.g. detector confidence levels) to compute the null-space of our proposed equation system. We also plan to include a RANSAC-based approach to get the best subset of equations. In addition, we plan to apply our outlier rejection method to other low rank problems where we can safely assume that the null space rank is one, a clear example is homography estimation.

Another direction consists in making robust our approaches to the camera intrinsic parameters, allowing to take advantage of our methods when these parameters are unknown. Interesting approaches based in the EPnP formulation are shown in [108, 56]. We plan to reformulate our methods to add the focal length inside the set of unknowns.

We also plan to transfer the 2D feature uncertainties to the 3D model, in order to make unnecessary to have a set of reference images with different 2D uncertainties. Moreover we plan to propagate the feature uncertainties towards the camera pose in order to compute its uncertainty using a 6 dimensional covariance matrix and then obtain a reliability measure of the solution. Finally, we also plan to extend the features used to compute the camera pose, at this moment we are only using interest points, however in recent publications line features are gaining relevance since allows to deal with texture-less objects.

6.4 Monocular Camera Pose Estimation in Complex Scenarios

We have proposed an early work in order to accurately estimate the camera pose dealing with complex and large 3D scenarios combining a purely appearance based technique with a geometrical approach. The proposed coarse-to-fine approach allows to our method reduce significantly the time required to estimate accurate camera poses in comparison with the time required if accurate methods, such us PnP, are directly applied. By first using global image appearance we select a set of candidate images in the training set and we reduce the number of feasible matches to be evaluated by our PnP solver, then the error in our camera pose solution becomes nearly negligible.

In this work we have shown how it is possible to leverage deep learning techniques to improve appearance-based approaches for the robotic community. It is remarkable that we are able to perform accurate pose estimation over hundreds of thousands of points in a few seconds per image, especially considering that we are using a MATLAB implementation.

Future Work : Since this is an early work, there are several directions for future research. First of all, as we mention in Section 5.4, according as the number of candidate images in the training set is increased the probability of getting more than 50% of outliers also increases, mainly because those candidate images tend to represent disjoint parts of the model. Thus, we plan speed up the proposed method detecting these disjoint image sets in order to apply REPPnP in each one.

Another very interesting work consist in take advantage of the rough camera pose, estimated using the appearance-based method, to spatially constrain the searching of 3D-to-2D correspondences in specific regions of the query image. Therefore, the feature matching cost, which at this time it is by far the largest of our approach, will be drastically reduced since each 3D feature on the model will be only matched with 2D features in an expected region on the query image.

Finally, in this work we have only compared for the coarse camera pose estimation the proposed CNN approach with one bag of visual words approach. In the future, we plan to compare several image classification methods in terms of computational time and accuracy.

6.5 Future Work Combining Proposed Algorithms

In this thesis we have dealt with three main areas of konwledge, interest points detection, interest points description and camera pose estimation. We have proposed novel algorithms in each one of those areas, however we have not taken advantage of all of them together. For example, we think it is interesting to evaluate the performance of our interest point descriptor on our interest point detector, since our detectors improve the distinctiveness of the image regions our descriptor can provide even better results. Combining these detectors and descriptors with camera pose estimation can improve drastically the performance of our PnP approaches, reducing the number of outliers because of their distinctiveness abilities under different viewing conditions, specially the ones related with viewpoint changes.

Bibliography

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. Building rome in a day. *Commun. ACM*, 54(10):105–112, 2011.

[2] Adnan Ansar and Konstantinos Daniilidis. Linear pose estimation from points or lines. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 25(5):578–589, 2003.

[3] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In *European Conference on Computer Vision*, 2006.

[4] P. R. Beaudet. Rotationally invariant image operators. In *Proceedings of the 4th International Joint Conference on Pattern Recognition*, pages 579–583, 1978.

[5] Yoshua Bengio, Ian J. Goodfellow, and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2015.

[6] Josef Bigun. A structure feature for some image processing applications based on spiral functions. *Computer Vision, Graphics, and Image Processing*, 51(2):166 – 194, 1990.

[7] Fred L Bookstein. Fitting conic sections to scattered data. *Computer Graphics and Image Processing*, 9(1):56–71, 1979.

[8] J. Bromley, I. Guyon, Y. Lecun, E. S?ckinger, and R. Shah. Signature verification using a "siamese" time delay neural network. In *Neural Information Processing Systems*, 1994.

[9] M. Brown, Gang Hua, and S. Winder. Discriminative learning of local image descriptors. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 33(1):43–57, 2011.

[10] M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale oriented patches. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 510–517, 2005.

[11] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):11, 2011.

[12] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015.

[13] Wojciech Chojnacki, Michael J. Brooks, Anton Van Den Hengel, and Darren Gawley. On the fitting of surfaces to data with covariances. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 22(11):1294–1303, 2000.

[14] Wojciech Chojnacki, Michael J Brooks, Anton van den Hengel, and Darren Gawley. FNS,CFNS and HEIV: A unifying approach. *Journal of Mathematical Imaging and Vision*, 23(2):175–183, 2005.

[15] Ondrej Chum and Jiri Matas. Matching with prosac-progressive sample consensus. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 220–226, 2005.

[16] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011.

[17] James L. Crowley. *A representation for visual information with application to machine vision*. PhD thesis, 1982.

[18] E. Culurciello, J. Jin, A. Dundar, and J. Bates. An analysis of the connections between layers of deep neural networks. *CoRR*, abs/1306.0152, 2013.

[19] J. Davis and M. Goadrich. The relationship between PR and ROC curves. In *International Conference in Machine Learning*, 2006.

[20] Fernando De La Torre and Michael J Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1-3):117–142, 2003.

[21] D. DeMenthon and L.S. Davis. Exact and approximate solutions of the perspective-three-point problem. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 14(11):1100–1105, 1992.

[22] J. Deng, W. Dong, R Socher, L.-J. Li, K. Li, and L Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[23] Rachid Deriche and Gerard Giraudon. Accurate corner detection: An analytical study. In *International Conference on Computer Vision*, pages 66–70. IEEE, 1990.

[24] D.G. Watts D.M. Bates. *Nonlinear regression analysis and its applications*. John Wiley and Sons. New York, 1988.

[25] Manfredo P. DoCarmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976.

[26] Gyuri Dorkó and Cordelia Schmid. Maximally stable local description for scale selection. In *European Conference on Computer Vision*, pages 504–516, 2006.

[27] L.S. Dreschler and H.-H. Nagel. On the selection of critical points and local curvature extrema of region boundaries for interframe matching. In ThomasS. Huang, editor, *Image Sequence Processing and Dynamic Scene Analysis*, volume 2 of *NATO ASI Series*, pages 457–470. Springer Berlin Heidelberg, 1983.

[28] Olof Enqvist, Erik Ask, Fredrik Kahl, and Kalle Åström. Robust fitting for multiple view geometry. In *European Conference on Computer Vision*, pages 738–751, 2012.

[29] Sergio Escalera, Petia Radeva, and Oriol Pujol. Complex salient regions for computer vision problems. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[30] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[31] M. Fidrich and J.P. Thirion. Stability of corner points in scale space: The effects of small nonrigid deformations. *Computer Vision and Image Understanding*, 72(1):72–83, 1998.

[32] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. of the ACM*, 24(6):381–395, 1981.

[33] W. Förstner, T. Dickscheid, and F. Schindler. Detecting interpretable and accurate scale-invariant keypoints. In *International Conference on Computer Vision*, 2009.

[34] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 25(8):930–943, 2003.

[35] Valeria Garro, Fabio Crosilla, and Andrea Fusiello. Solving the pnp problem with anisotropic orthogonal procrustes analysis. In *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pages 262–269, 2012.

[36] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research*, 2013.

[37] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[38] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich. Viewpoint-aware object detection and pose estimation. In *International Conference on Computer Vision*, 2011.

[39] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2013.

[40] J. A. Grunert. Das pothenotische problem in erweiterter gestalt nebst über seine anwendungen in geodäsie. *In Grunerts Archiv für Mathematik und Physik*, 1841.

[41] Antonio Guiducci. Corner characterization by differential geometry techniques. *Pattern Recognition Letters*, 8(5):311–318, 1988.

[42] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C. Berg. MatchNet: Unifying feature and metric learning for patch-based matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[43] Bert M Haralick, Chung-Nan Lee, Karsten Ottenberg, and Michael Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3):331–356, 1994.

[44] R.M. Haralick and L.G. Shapiro. *Computer and robot vision*. Prentice Hall, 1993.

[45] C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[46] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge Univ Press, 2000.

[47] Joel A Hesch and Stergios I Roumeliotis. A direct least-squares (dls) method for pnp. In *International Conference on Computer Vision*, pages 383–390, 2011.

[48] W. Hu and S.-C. Zhu. Learning a probabilistic model mixing 3D and 2D primitives for view invariant object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[49] Peter J Huber. *Robust Statistics*. Wiley, 1981.

[50] M. Jahrer, M. Grabner, and H. Bischof. Learned local descriptors for recognition and matching. In *Computer Vision Winter Workshop*, 2008.

[51] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *International Conference on Computer Vision*, 2009.

[52] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *CoRR*, 2014.

[53] Timor Kadir and Michael Brady. Saliency, scale and image description. *International Journal of Computer Vision*, V45(2):83–105, November 2001.

[54] Timor Kadir, Andrew Zisserman, and Michael Brady. An affine invariant salient region detector. In *European Conference on Computer Vision*, pages 228–241, 2004.

[55] Fredrik Kahl, Sameer Agarwal, Manmohan Krishna Chandraker, David Kriegman, and Serge Belongie. Practical global optimization for multiview geometry. *International Journal of Computer Vision*, 79(3):271–284, 2008.

[56] Ekaterina Kanaeva, Lev Gurevich, and Alexander Vakhitov. Camera pose and focal length estimation using regularized distance constraints. In *British Machine Vision Conference*, 2015.

[57] Kenichi Kanatani. Renormalization for unbiased estimation. In *International Conference on Computer Vision*, pages 599–606, 1993.

[58] Kenichi Kanatani. Optimization techniques for geometric estimation: Beyond minimization. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 11–30, 2012.

[59] Kenichi Kanatani, Ali Al-Sharadqah, Nikolai Chernov, and Yasuyuki Sugaya. Renormalization returns: Hyper-renormalization and its applications. In *European Conference on Computer Vision*, pages 384–397, 2012.

[60] Kenichi Kanatani and Yasuyuki Sugaya. High accuracy fundamental matrix computation and its performance evaluation. In *British Machine Vision Conference*, pages 217–226, 2006.

[61] Qifa Ke and Takeo Kanade. Quasiconvex optimization for robust geometric reconstruction. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 29(10):1834–1847, 2007.

[62] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2969–2976, 2011.

[63] Jan J Koenderink. The structure of images. *Biological cybernetics*, 50(5):363–370, 1984.

[64] Jan J. Koenderink. *Solid shape*. MIT Press, 1990.

[65] I. Kokkinos, M. Bronstein, and A. Yuille. Dense scale-invariant descriptors for images and surfaces. In *INRIA Research Report 7914*, 2012.

[66] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Neural Information Processing Systems*, 2012.

[67] Ruan Lakemond, Clinton Fookes, and Sridha Sridharan. Affine adaptation of local image features using the hessian matrix. *IEEE Conference on Advanced Video and Signal Based Surveillance*, 0:496–501, 2009.

[68] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[69] Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[70] Wei-Ting Lee and Hwann-Tzong Chen. Histogram-based interest point detectors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1590–1596, 2009.

[71] Yoram Leedan and Peter Meer. Heteroscedastic regression in computer vision: Problems with bilinear constraint. *International Journal of Computer Vision*, 37:127–150, 2000.

[72] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155–166, 2009.

[73] K. Levenberg. A method for the solution of certain problems in least squares. In *The Quarterly of Applied Mathematics*, volume 2, 1944.

[74] Shiqi Li and Chi Xu. A stable direct solution of perspective-three-point problem. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(05):627–642, 2011.

[75] Shiqi Li, Chi Xu, and Ming Xie. A robust o(n) solution to the perspective-n-point problem. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 34(7):1444–1450, 2012.

[76] J. Liebelt and C. Schmid. Multi-view object class detection with a 3d geometric model. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[77] Tony Lindeberg. Discrete derivative approximations with scale-space properties: A basis for low-level feature extraction. *Journal of Mathematical Imaging and Vision*, pages 349–373, 1993.

[78] Tony Lindeberg. On scale selection for differential operators. In *Scandinavian Conference on Image Analysis, Tromsö, Norway, May 1993*, pages 857–866, 1993.

[79] Tony Lindeberg. *Scale-Space Theory in Computer Vision (The International Series in Engineering and Computer Science)*. Springer, December 1993.

[80] Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30:77–116, 1998.

[81] Tony Lindeberg. A computational theory of visual receptive fields. *Biological Cybernetics*, 107(6):589–635, 2013.

[82] Tony Lindeberg and Jonas Gårding. Shape-adapted smoothing in estimation of 3-d shape cues from affine deformations of local 2-d brightness structure. *Image and Vision Computing*, 15(6):415–434, 1997.

[83] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[84] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[85] C-P Lu, Gregory D Hager, and Eric Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 22(6):610–622, 2000.

[86] Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11(2):431–441, 1963.

[87] Brais Martinez, Luis Ferraz, Xavier Binefa, and Jose Diaz-Caro. Multiple kernel two-step tracking. In *International Conference on Image Processing*, pages 2785–2788. IEEE, 2006.

[88] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference*, pages 384–393, 2002.

[89] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[90] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65:43–72, 2005.

[91] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60:63–86, 2004.

[92] Anlong Ming and Huadong Ma. A blob detector in color images. In *Conference on Image and Video Retrieval*, pages 364–370, 2007.

[93] H. Mobahi, R. Collobert, and J. Weston. Deep learning from temporal coherence in video. In *International Conference in Machine Learning*, 2009.

[94] F. Mokhtarian and F. Mohanna. Performance evaluation of corner detectors using consistency and accuracy measures. *Computer Vision and Image Understanding*, 102(1):81–94, 2006.

[95] F. Mokhtarian and R. Suomela. Robust image corner detection through curvature scale space. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 20(12):1376–1381, 1998.

[96] J.M. Morel and G. Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009.

[97] Francesc Moreno-Noguer, Vincent Lepetit, and Pascal Fua. Accurate non-iterative o(n) solution to the pnp problem. In *International Conference on Computer Vision*, pages 1–8, 2007.

[98] Francesc Moreno-Noguer, Vincent Lepetit, and Pascal Fua. Pose priors for simultaneously solving alignment and correspondence. In *European Conference on Computer Vision*, volume 2, pages 405–418, 2008.

[99] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application*, pages 331–340, 2009.

[100] Michael A. Nielsen. *Neural networks and deep learning*. Determination Press, 2015.

[101] J. Alison Noble. Finding corners. *Image and Vision Computing*, pages 2–121, 1988.

[102] Carl Olsson, Anders Eriksson, and Richard Hartley. Outlier removal using duality. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1450–1457, 2010.

[103] Carl Olsson, Fredrik Kahl, and Magnus Oskarsson. Branch-and-bound methods for euclidean registration problems. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 31(5):783–794, 2009.

[104] C. Osendorfer, J. Bayer, S. Urban, and P. van der Smagt. Convolutional neural networks learn compact local image descriptors. In *Neural Information Processing*, volume 8228, pages 624–630. Springer, 2013.

[105] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[106] N. Payet and S. Todorovic. From contours to 3D object detection and pose estimation. In *International Conference on Computer Vision*, 2011.

[107] A. Penate-Sanchez, F. Moreno-Noguer, J. Andrade-Cetto, and F. Fleuret. LETHA: Learning from high quality inputs for 3D pose estimation in low quality images. In *3D Vision*, 2014.

[108] Adrian Penate-Sanchez, Juan Andrade-Cetto, and Francesc Moreno-Noguer. Exhaustive linearization for robust camera pose and focal length estimation. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 35(10):2387–2400, 2013.

[109] B.T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1 – 17, 1964.

[110] Long Quan and Zhongdan Lan. Linear n-point camera pose determination. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 21(8):774–780, 1999.

[111] Rahul Raguram, Jan-Michael Frahm, and Marc Pollefeys. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. In *European Conference on Computer Vision*, pages 500–513, 2008.

[112] V. Rodehorst and A. Koschan. Comparison and evaluation of feature point detectors. In *International Symposium Turkish-German Joint Geodetic Day*, 2006.

[113] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: an efficient alternative to SIFT or SURF. In *International Conference on Computer Vision*, 2011.

[114] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[115] Jordi Sánchez-Riera, Jonas Ostlund, Pascal Fua, and Francesc Moreno-Noguer. Simultaneous pose, correspondence and non-rigid shape. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1189–1196, 2010.

[116] Peter H Schönemann and Robert M Carroll. Fitting one matrix to another under choice of a central dilation and a rigid motion. *Psychometrika*, 35(2):245–255, 1970.

[117] Gerald Schweighofer and Axel Pinz. Globally optimal o (n) solution to the pnp problem for general camera models. In *British Machine Vision Conference*, pages 1–10, 2008.

[118] P. Sermanet, S. Chintala, and Y. LeCun. Convolutional neural networks applied to house numbers digit classification. In *International Conference on Pattern Recognition*, 2012.

[119] Eduard Serradell, Mustafa Özuysal, Vincent Lepetit, Pascal Fua, and Francesc Moreno-Noguer. Combining geometric and appearance priors for robust homography estimation. In *European Conference on Computer Vision*, pages 58–72, September 2010.

[120] Jianbo Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[121] E. Simo-Serra, C. Torras, and F. Moreno-Noguer. DaLI: deformation and light invariant descriptor. *International Journal of Computer Vision*, 2015.

[122] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning local feature descriptors using convex optimisation. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 2014.

[123] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring image collections in 3d. In *ACM SIGGRAPH*, 2006.

[124] R. Socher, A. Karpathy, Q.V. Le, C.D. Manning, and A.Y. Ng. Grounded compositional semantics for finding and describing images with sentences. In *Transactions of the Association for Computational Linguistics*, 2014.

[125] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua. Lda-hash: Improved matching with smaller descriptors. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 34(1):66–78, 2012.

[126] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[127] H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *International Conference on Computer Vision*, 2009.

[128] L. Svarm, O. Enqvist, M. Oskarsson, and F. Kahl. Accurate localization and pose estimation for large 3d models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[129] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In *Neural Information Processing Systems*, 2013.

[130] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiel, and L. Van Gool. Towards multi-view object class detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[131] E. Tola, V. Lepetit, and P. Fua. DAISY: An Efficient Dense Descriptor Applied to Wide Baseline Stereo. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 32(5):815–830, May 2010.

[132] Bill Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *International Conference on Computer Vision*, volume 1, pages 278–284, 1999.

[133] E. Trulls, I. Kokkinos, A. Sanfeliu, and F. Moreno-Noguer. Dense segmentation-aware descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[134] T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit. Boosting binary keypoint descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[135] T. Tuytelaars. Dense interest points. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2281–2288, 2010.

[136] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: a survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.

[137] L. van der Maaten and G. Hinton. Visualizing data using t-SNE. In *Journal of Machine Learning Research*, 2008.

[138] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org`, 2008.

[139] M. Villamizar, A. Garrell, A. Sanfeliu, and F. Moreno-Noguer. Online human-assisted learning using random ferns. In *International Conference on Pattern Recognition*, 2012.

[140] M. Villamizar, A. Sanfeliu, and F. Moreno-Noguer. Fast online learning and detection of natural landmarks for autonomous aerial robots. In *International Conference on Robotics and Automation*, 2014.

[141] H. Wang and M. Brady. Real-time corner detection algorithm for motion estimation. *Image and Vision Computing*, 13(9):695–703, 1995.

[142] Zhenhua Wang, Bin Fan, and Fuchao Wu. Local intensity order pattern for feature description. In *International Conference on Computer Vision*, 2011.

[143] A. P. Witkin. Scale-space filtering. In *8th International Conference on Artificial Intelligence*, pages 1019–1022, Karlsruhe, Germany, 1983.

[144] Richard A Young. The gaussian derivative model for spatial vision: I. retinal mechanisms. *Spatial vision*, 2(4):273–293, 1987.

[145] Jin Yu, Anders Eriksson, Tat-Jun Chin, and David Suter. An adversarial optimization approach to efficient outlier removal. In *International Conference on Computer Vision*, pages 399–406, 2011.

[146] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[147] Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[148] Yinqiang Zheng, Yubin Kuang, Shigeki Sugimoto, Kalle Aström, and Masatoshi Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *International Conference on Computer Vision*, pages 4321–4328, 2013.

[149] Yinqiang Zheng, Shigeki Sugimoto, and Masatoshi Okutomi. Aspnp: An accurate and scalable solution to the perspective-n-point problem. *IEICE Transactions on Information and Systems*, 96(7):1525–1535, 2013.

[150] Z. Zheng, H. Wang, and E. Khwang Teoh. Analysis of gray level corner detection1. *Pattern Recognition Letters*, 20(2):149–162, 1999.

[151] Huiyu Zhou, Yuan Yuan, and Chunmei Shi. Object tracking using sift features and mean shift. *Computer Vision and Image Understanding*, 113(3):345–352, 2009.

[152] Xiaowei Zhou, Can Yang, and Weichuan Yu. Automatic mitral leaflet tracking in echocardiography by outlier detection in the low-rank representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 972–979, 2012.

Publications

- E. Simo-Serra, Trulls, E., **Ferraz, L.**, Kokkinos, I., Fua, P., and Moreno-Noguer, F., Discriminative Learning of Deep Convolutional Feature Point Descriptors. In *International Conference in Computer Vision (ICCV)*, 2015.

- A. Rubio, Villamizar, M., **Ferraz, L.**, Peñate-Sánchez, A., Ramisa, A., Simo-Serra, E., Sanfeliu, A., and Moreno-Noguer, F., Efficient Monocular Pose Estimation for Complex 3D Models. In *International Conference on Robotics and Automation (ICRA)*, 2015.

- **L. Ferraz**, Binefa, X., and Moreno-Noguer, F., Very Fast Solution to the PnP Problem with Algebraic Outlier Rejection. In *Conference in Computer Vision and Pattern Recognition (CVPR)*, 2014.

- **L. Ferraz**, Binefa, X., and Moreno-Noguer, F., Leveraging Feature Uncertainty in the PnP Problem. In *British Machine Vision Conference (BMVC)*, 2014.

- E. Simo-Serra, Trulls, E., **Ferraz, L.**, Kokkinos, I., and Moreno-Noguer, F., Fracking deep convolutional image descriptors. In *Computing Research Repository (CoRR)*, 2014.

- A. Rubio, Villamizar, M., **Ferraz, L.**, Peñate-Sánchez, A., Sanfeliu, A., and Moreno-Noguer, F., Estimacin monocular y eficiente de la pose usando modelos 3D complejos. In *Jornadas de Automtica*, 2014.

- **L. Ferraz** and Binefa, X., Fast and robust monocular 3D deformable shape estimation for inextensible and smooth surfaces. In *International Conference in Pattern Recognition (ICPR)*, 2012.

- O. Martinez, Cirujeda, P., **Ferraz, L.**, and Binefa, X., Dissociating rigid and articulated motion for hand tracking. In *International Conference in Pattern Recognition (ICPR)*, 2012.

- **L. Ferraz** and Binefa, X., A sparse curvature-based detector of affine invariant blobs. In *Computer Vision and Image Understanding (CVIU)*, 2012.

- O. Martnez, **Ferraz, L.**, Binefa, X., Gmez, I., and Dorronsoro, C., Concealed object detection and segmentation over millimetric waves images. In *Conference in Computer Vision and Pattern Recognition Workshop (CVPRW)*, 2010.

- **L. Ferraz** and Binefa, X., A Scale Invariant Interest Point Detector for Discriminative Blob Detection. In *Iberian Conference on Pattern Recognition and Image Analysis (IbPria)*, 2009.

- **L. Ferraz** and Binefa, X., A New Non-redundant Scale Invariant Interest Point Detector. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2009.

- **L. Ferraz**, R. Felip, B. Martinez, X. Binefa, A Density-Based Data Reduction Algorithm for Robust Estimators. In *Iberian Conference on Pattern Recognition and Image Analysis (IbPria)*, 2007.

- B. Martinez, Perez, A., **Ferraz, L.**, and Binefa, X., Structure Restriction for Tracking Through Multiple Views and Occlusions. In *Iberian Conference on Pattern Recognition and Image Analysis (IbPria)*, 2007.

- B. Martinez, **Ferraz, L.**, Binefa, X., and Diaz-Caro, J., Multiple Kernel Two-Step Tracking. In *International Conference on Image Processing (ICIP)*, 2006.

- B. Martnez, **Ferraz, L.**, and Binefa, X., Two-Step Tracking by Parts Using Multiple Kernels. In *International Conference of the Catalan Association for Artificial Intelligence (CCIA)*, 2006.

- B. Martnez, **Ferraz, L.**, Diaz-Caro, J., and Binefa, X., Optimization of the SSD multiple kernel tracking applied to IR video sequences. In *European Symposium on Optics and Photonics for Defence and Security (SPIE)*, 2005.