Disparate View Matching

Mayank Bansal

University of Pennsylvania, mayban@gmail.com

Follow this and additional works at: http://repository.upenn.edu/edissertations

Part of the Computer Sciences Commons

Recommended Citation

http://repository.upenn.edu/edissertations/1012
Disparate View Matching

Abstract
Matching of disparate views has gained significance in computer vision due to its role in many novel application areas. Being able to match images of the same scene captured during day and night, between a historic and contemporary picture of a scene, and between aerial and ground-level views of a building facade all enable novel applications ranging from loop-closure detection for structure-from-motion and re-photography to geo-localization of a street-level image using reference imagery captured from the air. The goal of this work is to develop novel features and methods that address matching problems where direct appearance-based correspondences are either difficult to obtain or infeasible because of the lack of appearance similarity altogether. To address these problems, we propose methods that span the appearance-geometry spectrum in terms of both the use of these cues as well as the ability of each method to handle variations in appearance and geometry. First, we consider the problem of geo-localization of a query street-level image using a reference database of building facades captured from a bird's eye view. To address this wide-baseline facade matching problem, a novel scale-selective self-similarity feature that avoids direct comparison of appearance between disparate facade images is presented. Next, to address image matching problems with more extreme appearance variation, a novel representation for matchable images expressed in terms of the eigen-functions of the joint graph of the two images is presented. This representation is used to derive features that are persistent across wide variations in appearance. Next, the problem setting of matching between a street-level image and a digital elevation map (DEM) is considered. Given the limited appearance information available in this scenario, the matching approach has to rely more significantly on geometric cues. Therefore, a purely geometric method to establish correspondences between building corners in the DEM and the visible corners in the query image is presented. Finally, to generalize this problem setting we address the problem of establishing correspondences between 3D and 2D point clouds using geometric means alone. A novel framework for incorporating purely geometric constraints into a higher-order graph matching framework is presented with specific formulations for the three-point calibrated absolute camera pose problem (P3P), two-point upright camera pose problem (Up2p) and the three-plus-one relative camera pose problem.

Degree Type
Dissertation

Degree Name
Doctor of Philosophy (PhD)

Graduate Group
Computer and Information Science

First Advisor
Kostas Daniilidis

Keywords
Computer Vision, Correspondenceless, Geo-localization, Graph Matching, Image Matching, Spectral Correspondence

This dissertation is available at ScholarlyCommons: http://repository.upenn.edu/edissertations/1012
Subject Categories
Computer Sciences

This dissertation is available at ScholarlyCommons: http://repository.upenn.edu/edissertations/1012
DISPARATE VIEW MATCHING

Mayank Bansal

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

2015

Kostas Daniilidis, Professor
CIS
Supervisor of Dissertation

Lyle Ungar, Professor
CIS
Graduate Group Chairperson

Dissertation committee

Jianbo Shi, Professor
CIS
University of Pennsylvania

Camillo J. Taylor, Professor
CIS
University of Pennsylvania

Jean Gallier, Professor
CIS
University of Pennsylvania

Fernando De la Torre, Associate Professor
Robotics Institute
Carnegie Mellon University

Harpreet S. Sawhney, CTO
Center for Vision Technologies
SRI International
DISPARATE VIEW MATCHING

COPYRIGHT

2015

Mayank Bansal
Acknowledgments

I am profoundly grateful to Prof. Kostas Daniilidis for his resolute confidence in me and for lending me unstinting support in every facet of my graduate school life. Thanks to the members of my thesis committee for their invaluable feedback that has helped improve the technical rigor and presentation of my work.

I would like to acknowledge Prof. Jianbo Shi and Prof. Kostas Daniilidis for providing me the opportunity to learn from the enriching TA-ship experience. I am thankful to Prof. Ben Taskar, Prof. Jean Gallier, Prof. Ali Jadbabaie and Prof. Sanjeev Khanna for imparting me unique technical insights through their courses.

A special thanks to Harpreet Sawhney, Rakesh (Teddy) Kumar, Jayan Eledath and Supun Samarasekara at SRI/Sarnoff for supporting my decision to pursue a PhD while letting me work on exciting new research projects at SRI. Jayan Eledath deserves an exclusive mention for furnishing me full cooperation through numerous conference deadlines and otherwise. Thanks to Oleg Naroditsky for setting an example for me by going back to school, to Ajay Divakaran for providing the reinforcement I needed to follow this dream, Aveek Das for keeping me motivated and Amir Tamrakar for always extending his insights into the reviewers’ feedback.

To my GRASP lab colleagues: Kosta Derpanis, Menglong Zhu, Cody Phillips, Matthieu Lecce and Weiyu Zhang, for helping me out on every occasion that I needed help with. Thanks to Mike Felker and Charity Payne for always being there to answer any and all of my questions!

Thanks to both my families and my lovely wife Kanika for believing in me. Without Kanika’s whole-hearted and relentless support, love and affection, I would have never accomplished this!
Matching of disparate views has gained significance in computer vision due to its role in many novel application areas. Being able to match images of the same scene captured during day and night, between a historic and contemporary picture of a scene, and between aerial and ground-level views of a building facade all enable novel applications ranging from loop-closure detection for structure-from-motion and re-photography to geo-localization of a street-level image using reference imagery captured from the air. The goal of this work is to develop novel features and methods that address matching problems where direct appearance-based correspondences are either difficult to obtain or infeasible because of the lack of appearance similarity altogether. To address these problems, we propose methods that span the appearance-geometry spectrum in terms of both the use of these cues as well as the ability of each method to handle variations in appearance and geometry. First, we consider the problem of geo-localization of a query street-level image using a reference database of building facades captured from a bird’s eye view. To address this wide-baseline facade matching problem, a novel scale-selective self-similarity feature that avoids direct comparison of appearance between disparate facade images is presented. Next, to address image matching problems with more extreme appearance variation, a novel representation for matchable images expressed in terms of the eigen-functions of the joint graph of the two images is presented. This representation is used to derive features that are persistent across wide variations in appearance. Next, the problem setting of matching between a street-level image and
a digital elevation map (DEM) is considered. Given the limited appearance information available in this scenario, the matching approach has to rely more significantly on geometric cues. Therefore, a purely geometric method to establish correspondences between building corners in the DEM and the visible corners in the query image is presented. Finally, to generalize this problem setting we address the problem of establishing correspondences between 3D and 2D point clouds using geometric means alone. A novel framework for incorporating purely geometric constraints into a higher-order graph matching framework is presented with specific formulations for the three-point calibrated absolute camera pose problem (P3P), two-point upright camera pose problem (Up2p) and the three-plus-one relative camera pose problem.
Contents

Acknowledgments iii

Contents vi

List of Tables ix

List of Figures x

List of Algorithms xvii

1 Introduction 1
  1.1 Thesis Contributions ........................................ 6
  1.2 Thesis Outline .................................................. 9

2 Background on Image Matching 10
  2.1 Features for Image Matching .................................... 11
  2.2 Wide-baseline Image Matching .................................. 12
  2.3 Disparate Appearance Matching ................................. 13
    2.3.1 Features for Disparate Appearance Matching .......... 14
    2.3.2 Disparate Image Matching by Co-segmentation ......... 14
  2.4 Geometry Guided Disparate View Matching .................. 16
    2.4.1 Graph Matching based Techniques ........................ 16
    2.4.2 Voting-based Techniques .................................. 17
  2.5 Image-based Geo-Localization ................................ 17
    2.5.1 Geo-localization using Street-level Image Data ....... 18
    2.5.2 Geo-localization using Aerial Image Data ............... 20
    2.5.3 Geo-localization using 3D Reference Data ............... 20

3 Ultra-wide Baseline Facade Matching for Geo-Localization 22
  3.1 Scale-Selective Self-Similarity Features .................... 24
  3.2 Facade Extraction and Segmentation ......................... 29
  3.3 Facade Matching ............................................... 32
  3.4 Experiments and Results ...................................... 33
## 3.4.1 Scale Selection Results ........................................... 36
3.4.2 Facade Detection Evaluation ...................................... 36
3.4.3 SV to BEV Matching ............................................... 37
3.4.4 Street-View to Street-View Matching ............................... 40
3.4.5 Comparison with SIFT Features ................................... 40
3.4.6 Camera Pose Estimation .......................................... 41

### 3.5 Facade Extraction using Satellite Imagery ........................... 44

### 3.6 Related Work .................................................... 50

### 3.7 Conclusion ....................................................... 51

## 4 Joint Spectral Correspondence for Disparate Image Matching 53

| 4.1 Similarity in the Eigen Space .................................. 56 |
| 4.1.1 Image Graph .................................................. 58 |
| 4.1.2 Image Features and the Joint Spectrum ......................... 59 |
| 4.1.3 Characterization of Persistent Regions ........................ 62 |
| 4.1.4 Eigen-function Feature Matching .............................. 63 |

### 4.2 Experiments ................................................... 64

| 4.2.1 Detector Repeatability ....................................... 65 |
| 4.2.2 Descriptor Evaluation ........................................ 66 |
| 4.2.3 Qualitative Results ........................................... 69 |

### 4.3 Robustness to Geometric Transformations .......................... 72

| 4.3.1 Rotation ....................................................... 73 |
| 4.3.2 Scale .......................................................... 73 |
| 4.3.3 Perspective ................................................... 74 |

### 4.4 Impact of Algorithm Parameters .................................. 74

| 4.4.1 Intra-vs-Inter Affinity ...................................... 78 |
| 4.4.2 Spatial Affinity ............................................... 79 |
| 4.4.3 Feature Affinity .............................................. 80 |
| 4.4.4 Spatial Sampling .............................................. 80 |

### 4.5 Related Work ................................................... 81

### 4.6 Conclusion ....................................................... 88

## 5 Geometric Urban Geo-Localization 89

| 5.1 With Correspondence Information ................................ 93 |
| 5.1.1 Preliminaries .................................................. 93 |
| 5.1.2 Correspondence of two points ................................ 94 |
| 5.1.3 Correspondence of one line ................................... 97 |
| 5.1.4 Correspondence of a point and a line .......................... 99 |

### 5.2 Correspondenceless Geolocalization ............................... 100

| 5.2.1 Stratified Geo-localization Algorithm ........................ 102 |
| 5.2.2 Feature Extraction ............................................ 107 |

### 5.3 Experiments ..................................................... 108
6 Geometric Polynomial Constraints in Higher-Order Graph Matching

6.1 Tensor Matching for Geometric Problems
   6.1.1 Tensor formulation for higher-order graph matching
   6.1.2 Geometric Constraints for Higher-Order Graphs
   6.1.3 Polynomial Resultant as Edge Affinities

6.2 Formulation for Specific Problems
   6.2.1 Three-point Calibrated Absolute Pose Problem (P3P)
   6.2.2 Three-plus-One Calibrated Relative Pose Problem (3P1)
   6.2.3 Two-point Calibrated Absolute Pose for an Upright Camera (up2p)

6.3 Experiments
   6.3.1 P3P with Higher-order Geometric Constraints
   6.3.2 P3P using Higher-order Angle Constraints
   6.3.3 Noise Robustness of the Edge Affinity Measure
   6.3.4 Comparison with RANSAC
   6.3.5 Employing High-confidence Matches
   6.3.6 3P1 with Higher-order Geometric Constraints
   6.3.7 Up2p with Higher-order Geometric Constraints
   6.3.8 Comparison with Voting
   6.3.9 Experiments with Real Data

6.4 Related Work

6.5 Conclusions

7 Discussion and Conclusions

Bibliography
List of Tables

1.1 Unification of Disparate Matching Approaches in this thesis . . . . . 6
3.1 Parameter settings . . . . . . . . . . . . . . . . . . . . . . . . . . . 33
3.2 Facade detection performance . . . . . . . . . . . . . . . . . . . . . 37
4.1 Detector repeatability compared with [Hauagge and Snavely, 2012]. . 66
4.2 Descriptor mean average precision (mAP) evaluation and comparison with [Hauagge and Snavely, 2012]. . . . . . . . . . . . . . . . . . . . . 66
6.1 Accuracy of a voting-based method for the Up2p problem across 100 simulated instances with $n = 10$ 3D points each. . . . . . . . . . . . . . . 148
7.1 Unification of Disparate Matching Approaches in this thesis . . . . . 155
List of Figures

1.1 Disparate View Matching problems in this thesis ............................. 2
1.2 Where different parts of this thesis fall within the appearance-geometry spectrum. ......................................................... 3
3.1 Disparate appearance of the same building facade between an oblique aerial view (left) and a view captured at street-level (right). ...... 23
3.2 Comparison of local self-similarity and SIFT descriptors on facade regions. (a) Rectified facade images from SV (top) and BEV (bottom), (b) corresponding SIFT descriptors for the blue region, (c) correspon-
ding self-similarity descriptors for the blue region obtained by correlating the green region. The self-similarity descriptors are noticeabley more similar than the SIFT descriptors. ......................... 25
3.3 Scale selection. To determine the scale $\lambda_x$ of the (black) 1D signal in the second row, if we autocorrelate a patch of width $w$, we get one of the profiles shown in rows 3-7 depending on the starting offset. However, for a poor offset choice (green and blue curves), one can get comparable peaks in the correlation profile for scale values $< \lambda_x$ making it difficult to extract the correct scale. Integrating across these profiles, however, resolves this issue and results in a well defined profile $p_{avg}(r)$ shown in the first row. The high peaks now correspond to the correct wavelength $\lambda_x$. ......................................................... 28
3.4 Facade Extraction and Segmentation. Rectified BEV images showing, left: the selected horizontal scales with red dots at the locations assigned zero scale value and, right: cluster assignments after K-means. 30
3.5 Scale Extraction and over-segmentation of facades from rectified BEV image. ................................................................. 31
3.6 Pittsburgh dataset. ................................................................. 34
3.7 Evaluation of scale estimation accuracy for a test set of 10 building facades from the BEV imagery. Densely estimated scale values from each facade are used to compute a normalized histogram which is plotted as blue circles (with radii proportional to the histogram values); the red pluses denote ground-truth scale values. ......................... 36
3.8 Example Street-view processing. ................................. 38

3.9 (a) ROC curve for BEV-to-SV matching on Pittsburgh dataset, (b) SV-to-SV matching performance on the “Pankrac+Marseilles” public dataset. ................................. 39

3.10 SV-to-SV matching examples on the “Pankrac+Marseilles” public dataset. First column shows the query followed by the top three retrieved results. The red, green and blue points denote the subset of features in the query which match best with each of three top contenders respectively. The black points match some other images from the database. The database had only one matching candidate in the case of the second example and the other retrieval results have the closest matching facade structure to the query. ......................... 41

3.11 Qualitative Matching Results. The main tiles show rectified BEV images. The insets show the original and rectified query street-view facades. On the rectified inset, the colored points are a subset of the words $w_1, w_2, \ldots, w_n$ with the top three most frequent recovered labels $\mathcal{L}(w_i)$ shown as red, green and blue points respectively; similarly colored points in the BEV image are words $v_j$ which belong to these three clusters. Top two rows: some examples of correct retrieval. Bottom row: Mismatched result (left) due to missing fine structure in the street-view image that was seen in the BEV image. Correct matching result for another street-view image which shows the fine structure is shown in the middle. The bottom right example shows a problem scenario where the non-fronto parallel facade in the BEV causes it to be mismatched due to the difference in the self-similarity structure presented by the descriptors. ......................... 42

3.12 Localization Results. The top row shows the SV images, and the bottom row shows the estimated and ground-truth camera locations. 43

3.13 Sample SAT tiles (left) and BEV imagery (right) from Ottawa, Canada. 44

3.14 SAT image rotated to align city-block direction with the x-axis (left) and the corresponding BEV image automatically aligned to the SAT image w.r.t the ground-plane using the geo-coordinate information (right). ................................. 45

3.15 Example of the facade extraction process. a) detected building tops and bottoms, and b) extracted facade tiles. ......................... 47

3.16 Building top search. Line segments extracted from the SAT imagery are projected to the canny edge map of BEV where a sweep along the gravity direction is expected to give a maximal point at the top edge of the building. ......................... 47
3.17 Effect of Graph-Cuts Optimization. The green edges are the SAT edges directly projected to this view and they lie in the ground plane. The red edges are the estimated building top edges. The top row shows the estimates obtained by picking the maximum score for each edge pixel independently; the bottom row shows these estimates refined by the GC optimization.

4.1 Day to night persistence matching. The spectrum of the joint image graphs is computed. The first row shows a day-time query-image (blue box) which is matched pair-wise against the pre-dusk, dusk and night images respectively from left to right. The second rows shows SIFT features detected on each image of the sequence. The third and fourth rows show the second eigen-vectors $J^{(2)}_1$ and $J^{(2)}_2$ for each pair of images (pre-dusk : day), (dusk : day), (night : day). The eigen-vectors corresponding to the query have a blue box to ease visualization. The plot compares the repeatability (bars) and average-precision (AP) (polyline) of the SIFT detector (blue) with the spectral method (red).

4.2 First column shows an image-pair from the dataset in [Hauagge and Snavely, 2012]. Second through fifth columns show eigen-function pairs $(J^{(2)}_1, J^{(2)}_2), \ldots, (J^{(5)}_1, J^{(5)}_2)$ along with the detected MSER feature-ellipses. The green and magenta colors denote whether the features correspond to maxima or minima.

4.3 For the image pair in the first column, the successive columns show the second-through-fifth eigen-function pairs obtained using a pixel-color based joint image graph. In this case, the eigen-functions do not suggest any significant correlation with the region correspondence in the original images.

4.4 For the image pair in the first column, the successive columns show the second-through-fifth eigen-functions obtained using dense-SIFT-based image graphs. The eigen analysis is performed on each image graph independently. In this case, the eigen-functions show correlations but the correlated regions are distributed across several different eigenfunctions.

4.5 Precision-Recall curves comparing performance of the spectral approach (JSPEC) with the features evaluated in [Hauagge and Snavely, 2012]. Each column shows plots for the image pair in the top row. For each image pair, the JSPEC curve is repeated in the four rows to show comparison with the four different detectors in [Hauagge and Snavely, 2012].
4.6 Comparison of SIFT matches and JSPEC matches on a day-night image pair. We also show the affinity matrix for the joint graph and the eigen-function pairs from which the JSPEC feature matches were obtained. In (e), we show eigen-function pairs \((J_1^{(2)}, J_2^{(2)}), \ldots, (J_1^{(5)}, J_2^{(5)})\) along with the MSER feature-ellipses that have been matched using the SIFT-bidirectional matching criterion \((\tau = 0.8)\).  

4.7 Characterization of persistent regions detected by JSPEC matching. JSPEC features capture more elongated structures compared to SIFT features.  

4.8 Painting to image matching. The painting images (top-row) have been taken from the dataset in [Shrivastava et al., 2011].  

4.9 The first column shows an image-pair from the dataset in [Hauagge and Snavely, 2012] along with the correspondences assembled from the individual eigen-functions. Second through fifth columns show eigen-function pairs \((J_1^{(2)}, J_2^{(2)}), \ldots, (J_1^{(5)}, J_2^{(5)})\) along with the MSER feature-ellipses that have been matched using the SIFT-bidirectional matching criterion \((\tau = 0.8)\).  

4.10 The first column shows an image-pair from the dataset in [Szeliski, 2005] along with the correspondences assembled from the individual eigen-functions. Second through fifth columns show eigen-function pairs \((J_1^{(2)}, J_2^{(2)}), \ldots, (J_1^{(5)}, J_2^{(5)})\) along with the MSER feature-ellipses that have been matched using the SIFT-bidirectional matching criterion \((\tau = 0.8)\).  

4.11 Performance evaluation of JSPEC features with varying in-plane rotation angle \(\theta\).  

4.12 Performance evaluation of JSPEC features with varying scale parameter \(s\).  

4.13 Performance evaluation of JSPEC features with varying horizontal perspective parameter \(\tau_x\).  

4.14 Performance evaluation of JSPEC features with varying intra-image similarity weight \(\alpha\).  

4.15 Performance evaluation of JSPEC features with varying spatial affinity parameter \(\sigma_s\) for the intra-image similarity.  

4.16 Impact of variation of feature affinity parameter \(\sigma_f\) on the performance of JSPEC features.  

4.17 Performance evaluation of JSPEC features with varying spatial sampling parameter \(\delta\). Also shown is the impact of varying this parameter on the computation time.  

5.1 Geometric geo-localization of a night image using PointRay features.  

5.2 Degenerate cases for the two point method.
5.3 Query processing: a) Each red-dot and green-arrow pair make up one PointRay feature \((p_i, l_i)\). b) The PointRay feature \((p_i, l_i)\) shown in red when corresponded with the correct 3D PointRay feature from the database results in a camera pose locus. By corollary-(1), the camera in this locus maps the remaining database corners to line segments shown in yellow and cyan. The cyan segments represent the inlier set \(C\) because of their proximity to the query corners (shown as green circles). We show two cases here where different locii are created by using a different (but correct) reference PointRay feature correspondence. c) Green circles denote the inliers after the two-point method is applied to \(C\); the green lines depict projected 3D edges using the computed camera pose. d) DEM contours rendered using the hypothesis camera pose generate the green skyline which matches correctly with the perceived skyline. e) DEM showing the reference corners (blue +), recovered camera pose (yellow), ground-truth camera pose (red) and the recovered inliers (green).

5.4 DEM from Ottawa with query locations overlaid. Each location is shown as a red circle with green arrows depicting the look-at vectors corresponding to each query image. The inset shows the elevation distribution for the query set.

5.5 Example queries where sky-detection failure disrupts detection of PointRay features.

5.6 Example queries where geo-localization within 20m of the ground-truth could not be achieved. This is due to the missing PointRay features leading to only a single valid PointRay feature in each image, thus making the geo-localization problem unconstrained. The last panel shows a zoomed-in view of the third example and exhibits the poor quality of the query images.

5.7 Geo-localization performance evaluation.

5.8 Geometric geo-localization of a night image using PointRay features.

5.9 Samples from the 50 image query set along with the PointRay features detected on each. Notice the variety of viewpoints as well as the complexity of building placement captured in this dataset.

6.1 Matching between feature points in images \(I\) and \(I'\) can be expressed using higher-order relationships between 3-tuples of feature points by defining the affinity as a function of the triangle angles; for example as
\[
H(i, i'; j, j'; k, k') = \exp \left(-\left(||\alpha - \alpha'|| + ||\beta - \beta'|| + ||\gamma - \gamma'||\right)/\sigma^2\right).
\]

6.2 Correspondences in sets \(S\) and \(S'\) are the hyper-edges corresponding to the minimal configuration and generate constraints in the form of polynomial equations \(q_S = 0\) and \(q_{S'} = 0\) respectively. These sets are combined to form a new hyper-edge with weight given by the resultant of the Sylvester matrix \(M(q_S, q_{S'})\) of the two polynomials.

6.3 Setup of the Three-point Calibrated Absolute Pose Problem (P3P).
6.4 **Simulation results for the P3P Problem using 3D geometry constraints.** Average matching accuracy across 100 randomly generated instances of the P3P problem ($n = 10$) for different amounts of Gaussian noise. The *solid* and *dashed* curves represent the performance of tensor power-iteration algorithm [Duchenne et al., 2011] with and without the $\ell^1$-norm constraint on the assignment matrix respectively. In (a) no outliers were added; in (b) 50% and 100% additional random image points were added as outliers to the inlier set of 10 points.

6.5 **Simulation results for the P3P Problem using angle constraints.** Average matching accuracy across 100 randomly generated instances of the P3P problem ($n = 10$) for different amounts of Gaussian noise. The *solid* and *dashed* curves represent the performance of tensor power-iteration algorithm [Duchenne et al., 2011] with and without the $\ell^1$-norm constraint on the assignment matrix respectively. In (a) no outliers were added; in (b) 50% and 100% additional random image points were added as outliers to the inlier set of 10 points.

6.6 **Robustness of the Edge Affinity Measure for the P3P Problem.** Error bars ($\mu \pm \sigma$) for the affinity measure (6.9) across 1000 randomly generated hyper-edges from the P3P problem for different amounts of Gaussian noise. The *red* and *blue* curves represent performance on sets of valid vs. invalid correspondences respectively indicating the discriminability of the affinity measure as well as the increase in confusion between valid and invalid configurations at higher noise levels.

6.7 **Simulation Results for the P3P problem using RANSAC.** Average matching accuracy achieved by RANSAC across 100 randomly generated instances of the P3P problem ($n = 10$) for different amounts of Gaussian noise. The poor performance can be attributed to incorrect matches that lead to lower reprojection errors but an increased number of inliers.

6.8 **Simulation Results for the P3P problem given some high-confidence matches.** Average matching accuracy across 100 randomly generated instances of the P3P problem ($n = 10$) for different number of initially specified high-confidence matches. The accuracy achieved by the tensor-matching algorithm steadily improves even with $\sigma = 1.0$ pixel image noise as more high-confidence seed correspondences are specified.
6.9 **Simulation results for the 3P1 Problem.** Average matching accuracy across 100 randomly generated instances of the 3P1 problem ($n = 10$) for different amounts of Gaussian noise. The *solid* and *dashed* curves represent the *sparse* and *dense* tensor power-iteration algorithms [Duchenne et al., 2011]. The left and right columns compare performance for two different values of the baseline between the two cameras. The top and bottom rows show results without and with added outliers respectively.

6.10 **Simulation results for the Up2p Problem.** Average matching accuracy across 100 randomly generated instances of the Up2p problem ($n = 10$) for different amounts of Gaussian noise. The *solid* and *dashed* curves represent the *sparse* and *dense* tensor power-iteration algorithms [Duchenne et al., 2011]. In (a) and (b) the data points all followed a single motion; in (c) the 10 points were split in two sets of 5 points each following a different motion. In (b), we also show performance with 50% and 100% additional 2D points as outliers for the single motion case.

6.11 **Up2p on Real Data.**

6.12 Solution assignment matrix $X$ of the Up2p problem for hypergraphs with increasing number of sampled edges.

6.13 Solution assignment matrix $X$ of the Up2p problem for hypergraphs with increasing number of sampled edges. The first five 2D points are added as outliers leading to a $25 \times 20$ matrix $X$.

6.14 Solution assignment matrix $X$ of the Up2p problem for hypergraphs with increasing number of sampled edges. The first 10 2D points are added as outliers leading to a $25 \times 15$ matrix $X$. Note that the algorithm converges to a good solution after sampling a larger number of hyper-edges relative to Fig. 6.13 due to the larger proportion of outliers.
## List of Algorithms

<table>
<thead>
<tr>
<th>Section</th>
<th>Algorithm</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>BEV processing</td>
<td>33</td>
</tr>
<tr>
<td>3.2</td>
<td>SV processing</td>
<td>33</td>
</tr>
<tr>
<td>4.1</td>
<td>JSPEC Algorithm</td>
<td>64</td>
</tr>
<tr>
<td>5.1</td>
<td>Proposed Geo-localization Algorithm</td>
<td>105</td>
</tr>
<tr>
<td>6.1</td>
<td>Rank-3 Tensor Power Iteration Algorithm</td>
<td>122</td>
</tr>
<tr>
<td>6.2</td>
<td>Mixed Order Tensor Power Iteration Algorithm</td>
<td>123</td>
</tr>
<tr>
<td>6.3</td>
<td>Preparing a geometric problem for higher-order graph matching</td>
<td>128</td>
</tr>
<tr>
<td>6.4</td>
<td>Higher-order Geometric Graph Matching Algorithm</td>
<td>128</td>
</tr>
<tr>
<td>6.5</td>
<td>RANSAC Algorithm for P3P problems with given correspondences</td>
<td>139</td>
</tr>
<tr>
<td>6.6</td>
<td>RANSAC Algorithm for P3P problems without correspondences</td>
<td>140</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Image matching lies at the core of computer vision, its importance underscored by the sheer number of features and matching techniques in the literature today. A number of these features (SIFT, MSER, BRIEF, etc.) and matching techniques (graph matching, geometry estimation etc.) have been devised to work extremely well for a given set of invariants or for a given set of restrictions on the kinds of variations that the given image pair might have. These techniques thus support matching of images that exhibit appearance variation within the prescribed limits and thus do not handle larger appearance variations.

In this thesis, we focus on matching problems where direct appearance-based correspondences are either difficult to obtain or infeasible because of the lack of appearance similarity altogether. Fig. 1.1 shows several examples of disparate matching problems. In the top row of this figure, we include problems where image-based appearance information is available but difficult to match directly due to stark differences arising from illumination, viewpoint, or rendering style variation etc. The bottom row of Fig. 1.1 shows an example of matching a digital elevation map (DEM) model of an urban environment against a query street level image of the scene with
an aim to geo-locate the query image. In this case, appearance information is available only in the form of sparse building edges and corners and is not sufficient to solve the matching problem without exploiting additional geometric cues. Finally, the problem of matching sets of 3D points and their 2D projections without apriori correspondence information can only be solved by exploiting geometric constraints if no appearance information is available. Thus, each of these matching problems exhibit a different mix of challenges due to appearance and geometric variations. Correspondingly, approaches to address these challenges also need to utilize features and matching methods that span the appearance and geometry spectrum.

Fig. 1.2 lays out the contributions of this thesis on the appearance-geometry spectrum in terms of both the use of these cues as well as the ability of each approach
Figure 1.2: Where different parts of this thesis fall within the appearance-geometry spectrum.

to handle variations in appearance and geometry. We have also included SIFT [Lowe, 2004] matching as a reference point to aid comparisons.

The air-ground matching problem depicted in Fig. 1.1 is essentially a wide-baseline image matching problem where the candidate images have been acquired from a ground-level camera and from an oblique-angled aerial camera. The geometry and availability of geographic metadata in this problem setup allows pre-processing of the input imagery such that both the aerial and ground-level facade images can be pre-rectified to be fronto-parallel. This reduces the problem to that of matching rectified facade imagery (depicted as the Facade Matching bullet in Fig. 1.2) where the key challenge is to deal with disparate appearance and ambiguous repetitive structure. Even in this simplified setting, the local gradient structure exploited by SIFT features is incapable of dealing with the large appearance changes resulting from the viewpoint and temporal differences between the image pair. Due to the infeasibility of direct appearance matching between such image pairs, we have to consider features which can encode the appearance of individual building facades
visible in the street-level and aerial imagery. We show how we can exploit the repetitive pattern geometry in facade regions to create a richer feature by adapting the self-similarity descriptors [Shechtman and Irani, 2007a]. This feature can then be used in a Bayesian framework to allow retrieval of matching facades given a query street-level facade image.

The problem of pair-wise matching of images exhibiting more extreme variations like large illumination differences (day-night), rendering style differences (painting-picture) etc. forces even the self-similarity features to be insufficient. Since there is no local repetitive structure to exploit in this case, we need to employ geometric cues provided by short and long-range similarity structure of local features. Thus, as compared to local self-similarities, we need to look at long range similarities without the knowledge of a fixed pattern in which the features might repeat. We show how this can be achieved in a joint spectral (JSPEC) representation of the image-pair where starting from local dense SIFT features, we can derive a representation where matchable structures stand out. Since we operate on a spatial representation of the eigen-functions, long range spatial geometric relationships are instrumental in supplying matchable information. This framework is depicted as the JSPEC Matching bullet in Fig. 1.2.

The problem of matching the digital elevation map (DEM) of an urban area to a street-level image is inherently geometric since the only structure that is common are the edges and corners of buildings and other static elements in the scene. Therefore, we propose a geometric approach that exploits the 3D geometry of projection of building corners and edges into the camera view to aid geo-localization. This approach, thus relies even more heavily on geometry and further less on appearance information. To make the problem tractable, we propose a stratified geometric
framework of associating 3D and 2D features that produces a ranked list of potential solutions of the street-view camera pose, thus facilitating geo-localization in an urban setting. This is depicted as the Geometric Urban Geo-localization bullet in Fig. 1.2.

Finally, the problem of 3D-2D point set matching does not provide any appearance information to start with. Thus, the focus of a solution method is to utilize geometry while allowing an optimization-based framework. Graph matching in general, and higher-order graph matching in particular provides such an optimization framework. However, these methods have only been utilized in geometric matching setups where some invariants can be established across the two sets. To handle 3D-to-2D setups (e.g. the three point calibrated absolute camera pose (P3P) problem), where no such invariants exist, we define a novel framework that allows formulation of polynomial constraints in geometric problems as edge weights of a higher-order graph. This is the Higher-order Geometry Matching bullet in Fig. 1.2.

All of the approaches in the discussion above allow handling of varying amounts of disparate appearance while making certain assumptions about the geometric setting in which they can operate (see Fig. 1.2(b)). While all of them handle more appearance variation than SIFT features, Facade matching and JSPEC matching allow for more limited geometric variations. Facade matching assumes rectified facade images and thus lies at the lowest point on the geometric handling axis. We show empirically that the JSPEC matching framework can handle a range of geometric transformations but this range is still limited compared to SIFT features. The setting in geometric urban geo-localization handles significantly more appearance variation since it only relies on the detection of edges and corners of buildings visible in the street-view image which are more robust to appearance changes. Since we employ a general 3D-2D geometric framework for matching these line and corner features,
Table 1.1: Unification of Disparate Matching Approaches in this thesis

<table>
<thead>
<tr>
<th></th>
<th>Facade Matching</th>
<th>JSPEC Matching</th>
<th>Geometric Urban Geo-localization</th>
<th>Higher-Order Geometry Matching</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assumed Geometry</td>
<td>Rectified Plane</td>
<td>Small Transformation</td>
<td>Upright Camera</td>
<td>General</td>
</tr>
<tr>
<td>Features Exploited</td>
<td>Repetitive Texture; not-locally matchable</td>
<td>Short and long range interactions of local gradients</td>
<td>Specialized line and corner features</td>
<td>Geometric corners</td>
</tr>
<tr>
<td>Matching</td>
<td>Bag of features</td>
<td>Eigen structure of joint graph</td>
<td>Brute-force (stratified)</td>
<td>Higher-order graph with geometric constraints</td>
</tr>
</tbody>
</table>

the approach can also handle significantly more geometric variations. Higher-order geometry matching does not use appearance at all, and uses general 3D geometry constraints and thus has the best appearance and geometry handling ability.

1.1 Thesis Contributions

This thesis addresses the problem of matching disparate views by introducing methods that span the appearance-geometry spectrum. Table 1.1 captures the relationship between these techniques with respect to the assumed geometry, features exploited and the matching algorithms. Text in bold highlights general areas of contribution of this thesis – we include more specific contributions below and in the chapters corresponding to each method.

Facade Matching: We address the problem of geo-localization using aerial image data as that of matching facade imagery between street-level and aerial views. The key idea is to avoid direct matching of features to solve this extreme case of wide-baseline matching. Thus, we formulate the problem as “embeddings” within each respective dataset (SV and BEV) so that large variations are incorporated within
the structure of embeddings. This idea has not been explored before especially in the context of air-ground matching. We make the following contributions to the state of the art:

- We introduce an approach for matching image regions with significant appearance, scale, and viewpoint variations based on a novel Scale-Selective Self-Similarity ($S^4$) feature that combines intrinsic scale selection with self-similarity descriptors.

- We demonstrate a novel system for matching street-level queries to a database of bird’s eye views. We show experimental results on the retrieval accuracy from our technique and compare our performance with standard SIFT-descriptors.

**JSPEC Matching:** We address the problem of matching disparate appearance between painting-picture, day-night, etc. image pairs and make the following contributions:

- We introduce a new representation between two images: the joint image graph defined only based on the affinity between image structures in the dense set of pixels from both images without considering the proximity between two image positions.

- We propose a new definition of persistent regions as the stable regions of the eigen-functions of that graph considered in their “soft” form without any discretization.

- We show that such persistent features are both repeatable across images and similar in terms of SIFT descriptors computed in the eigen space itself, in a variety of cross domain setups.
**Geometric Urban Geo-localization:** We address the problem of geo-localization of a street-level image using 3D Reference Data. Without any appearance information, we rely on geometric constraints between building corners in the street-level query image and the 3D corners from the DEM reference data. To this end, we make the following contributions:

- We propose a novel formulation for upright pose estimation using a point-line pair and derive its degeneracy conditions.

- We propose a novel framework for stratified pose estimation using point-line and 2-point algorithms.

- We introduce an application of this framework to geo-localization without appearance correspondences while avoiding any visibility information or rendering.

**Higher-order Geometry Matching:** We address the problem of matching 3D-2D point sets for which only geometric constraints are available. We operate in a higher-order graph matching framework and make the following contributions:

- We solve the correspondence problem based purely on geometric polynomial constraints within an existing graph matching optimization framework.

- We introduce the novel idea of using the magnitude of the resultant of a pair of polynomial equations as a measure of agreement between the models represented by the equations that can represent a higher-order affinity across graphs.

- As opposed to RANSAC which filters initial correspondences, we start with complete lack of any feature matches.
• As opposed to EM, we enable correspondence finding without computing the underlying geometric transformations, thus enabling matching in the presence of multiple transformations like articulated motion.

1.2 Thesis Outline

This thesis is structured as follows. In Chapter 2, we review some background material on image matching with respect to features as well as matching techniques. In addition, we review recent work on image-based geo-localization since this is the application of our focus in both facade matching and geometric urban geo-localization. Chapters 3, 4, 5 and 6 describe our frameworks and algorithms for facade matching, JSPEC matching, geometric urban geo-localization and higher-order geometry matching, respectively. We conclude with a discussion about the unification of various techniques as well as potential future directions in Chapter 7.
Chapter 2

Background on Image Matching

There is a tremendous amount of literature addressing the problem of image matching. Numerous features and matching techniques have been devised to allow matching between two images of the same scene captured from different viewpoints or with varying amount of appearance difference.

The usual pipeline for image matching works as follows. First, a set of interest points are detected independently in the two images. A feature descriptor vector is usually associated with each interest point to allow comparison of the image appearance in the neighborhood of the detected interest point. Next, the two sets of features are compared with respect to their distance in the feature space and the nearest-neighbor of each feature is determined. To keep only the most distinctive matches, a criterion like the ratio of distance to the nearest and second-nearest neighbor is typically employed [Lowe, 2004] and matches for which the ratio is smaller than a threshold are rejected. The set of matches that remain after this (and other potential) filtering step(s) are known as the “putative” correspondences between the given image pair. Finally, a geometric verification step is applied to the correspondences if the given images are known to satisfy a particular geometric constraint.
For example, if the two images are known to be images of a plane, then a projective transformation describes the mapping of the image coordinates of features in one image to the coordinates of the features in the second image. For a general 3D scene, a Fundamental matrix would describe the geometric constraint between the feature coordinates. These constraints are enforced through either voting-based methods or through the widely used Random Sample Consensus (RANSAC) method [Fischler and Bolles, 1981].

2.1 Features for Image Matching

Given the dependence of most image matching algorithms on good image features, a large body of work in the literature has focused on defining interest point detectors that are invariant to certain image variations as well as on defining descriptors that can be matched (distinctively) between images with significant viewpoint and appearance variation. The influential work by [Lowe, 2004] introduced the Scale Invariant Feature Transform (SIFT) approach where interest points exhibiting scale and orientation-invariance were detected and were shown to exhibit good repeatability across different views of an object or scene. These interest points were then associated with a descriptor vector based on the gradient distribution around each detected interest point and this representation was shown to be quite discriminative for the image matching task. [Mikolajczyk and Schmid, 2004] proposed a novel approach for detecting interest points invariant to scale and affine transformations. This detector, along with a number of other affine covariant region detectors, were analyzed by [Mikolajczyk et al., 2005] in terms of their performance against changes in viewpoint, scale, illumination, defocus and image compression.

Like in the case of region detectors, a number of descriptors for local interest
regions have also been proposed in the literature. These include SIFT [Lowe, 2004], shape-context [Belongie et al., 2002], steerable filters [Freeman and Adelson, 1991], PCA-SIFT [Ke and Sukthankar, 2004], self similarity [Shechtman and Irani, 2007a] etc. The idea behind these descriptors is to either capture a different characteristic of the images being matched or to handle specific kinds of variations in the image. For example, the shape-context descriptor encodes the binary contours of a shape around an interest point to allow matching of shape signatures between two images. The SIFT descriptor encodes the local image gradients in a scale-invariant manner without binarizing the edge map. The self-similarity descriptor encodes the auto-correlation surface of an image patch in its neighborhood and is thus useful for matching between images where the larger appearance differences make direct comparison infeasible. A comprehensive performance evaluation of a number of these descriptors can be found in [Mikolajczyk and Schmid, 2005] where the GLOH descriptor (introduced by the authors) is shown to perform the best, closely followed by SIFT.

2.2 Wide-baseline Image Matching

The features (interest points and descriptors) described above can handle a large variety of image matching problems. However, the performance of these features degrades rapidly as the viewpoint difference between the captured images increases. Therefore, novel features and matching methods are required to deal with this special case of image matching which is commonly referred to as the wide-baseline matching problem and is important for applications like wide-baseline stereo, geo-localization using air-ground matching etc.

[Matas et al., 2004] addressed this problem by defining new features called the
maximally stable extremal regions (MSER) that were shown to be repeatable under wide camera pose changes. The affinely-invariant property of these regions allows them to deal with significant scale changes ($3.5\times$), out-of-plane rotation, 3D camera translation and change in illumination conditions. To allow the SIFT algorithm to handle larger camera pose changes, a generalization of the SIFT algorithm, called Affine-SIFT (ASIFT) was presented by [Morel and Yu, 2009b]. This method simulates all six parameters of an affine transformation to generate synthesized versions of the given image pair and then applies the SIFT algorithm to obtain higher quality matches. This allows this method to handle a much higher viewpoint variation than other methods. [Wu et al., 2008b] proposed the Viewpoint-Invariant Patches (VIP) features for 3D scene alignment and large scale scene reconstruction. These features are obtained by rectifying the image texture at each point into an orthographic view using the local scene geometry at the point, thus leading to a viewpoint independent representation which is shown to be stable (but not invariant) to projective transformations.

2.3 Disparate Appearance Matching

Matching between images with hard to match appearance can be addressed from two perspectives. The first is by considering features that are invariant to the nature of appearance differences at hand, the second is by considering the matching problem directly and using techniques that directly model matching of regions between the two images as a co-segmentation or co-recognition problem.
2.3.1 Features for Disparate Appearance Matching

Not many approaches exist that can handle the discrepancy between two images at the level that we address in this thesis. [Hauagge and Snavely, 2012] have focused on the task of matching such images by defining “local-symmetry” features which capture various symmetries like bilateral, rotational etc. at the local level. [Shechtman and Irani, 2007b] proposed an approach for matching disparate images using patterns of local self-similarity encoded into a shape-context like descriptor. This descriptor encodes the local self-similarity around each interest point and thus allows matching of corresponding descriptors from images with very different appearance since the absolute appearance is never directly compared across the two images. [Shrivastava et al., 2011] proposed an approach for cross-domain image matching using data-driven learning techniques. Using a linear classifier, they learn the relative importance of different features (specifically, components of the global image HoG descriptor in their paper) for a given query image and then use the weight vector to define a matching score.

2.3.2 Disparate Image Matching by Co-segmentation

Significant amount of work in the literature has focused on the problem of “Co-segmentation” where the objective is to segment the same object or foreground region from a pair of images. This is different from the objective of finding correspondence between two images of the same scene with hard to match appearance differences. [Chang et al., 2011] proposed an MRF optimization model that uses co-saliency as a prior to solve the co-segmentation problem in an unsupervised manner. Image saliency is estimated using the approach of [Goferman et al., 2012] and is converted to co-saliency by reducing the weight of more repeated image features.
The repeatedness is measured by considering the distance between sampled SIFT descriptors (in the salient regions) across multiple images. [Cho et al., 2008] proposed a generative model for “co-recognition” of an object between arbitrary image pairs. [Mukherjee et al., 2012] address the co-segmentation problem at a more general level.

[Li and Ngan, 2011] proposed a co-saliency model specifically for a pair of images. They combine single image saliency models with a pair-wise saliency term that decomposes each image into a spatial pyramid composed of super-pixels. The affinity between the super-pixels in the two images is then measured by using the SimRank algorithm on a combination of texture and color descriptors. [Jacobs et al., 2010] present an idea of co-saliency that is trained using a collection of user-annotated image pairs. However, they focus on pairs of similar images, like those captured within the same burst of shots.

[Glasner et al., 2011] proposed a shape-based approach to jointly segment multiple closely-related images by combining contour and region information. They show examples of image pairs with illumination differences where their joint segmentation approach achieves better co-clustering than what is possible by using intra-image constraints alone. They start from a super-pixel segmentation of the images and then use contour-based constraints to drive their intra-image affinities. The inter-image constraints are derived from a comparison of HoG-like features only on contour segments.

The spectral analysis of the joint matrix between two images appeared first in [Yu et al., 2002] where the affinity matrices of object model patches and the input image are combined with a non-diagonal matrix associating object patches and image pixels. [Toshev et al., 2007] proposed an approach to determine co-salient regions between two images using a spectral technique on the joint image graph constructed from the images. Their joint image graph was constructed with all the pixels in the
two images by defining separate affinity functions for intra and inter image terms. The intra image affinity was defined using the intervening contour cue while the inter image term was based entirely on the initial set of feature correspondences between the images.

2.4 Geometry Guided Disparate View Matching

When only geometric entities (like sets of 3D or 2D points, lines etc.) are being matched, then the matching problem has to be formulated in a correspondenceless setting. In such cases, graph matching has been employed as the formal means to express the matching function as a combination of unary and higher-order potentials that measure similarity between single and sets of correspondences respectively. Voting-based schemes have also been employed in cases where the unary terms have sufficient power to guide the voting process.

2.4.1 Graph Matching based Techniques

[Leordeanu and Hebert, 2005] consider the quadratic assignment problem where distances between pairs of features from two images are used to create an affinity matrix and an efficient spectral solution to solving this problem is proposed. [Cour et al., 2006] generalize their approach to allow incorporation of additional affine constraints. [Schellewald and Schnörr, 2005] address the same problem in a convex optimization framework by relaxing the discrete problem into a semidefinite program (SDP). More recently, [Zhou and De la Torre, 2013, Zhou and De la Torre, 2012] proposed deformable graph matching (DGM) for matching graphs subject to global rigid and non-rigid geometric constraints. However, they also restrict the choices of transformation to certain classes like similarity, affine and RBF non-rigid and work
in the context of image to image matching.

The use of higher-order matching in the computer vision literature has focused on inclusion of constraints derived from higher-order geometric invariants like angles of triangles, cross-ratio along lines etc. This allows more robust matching between features in two images (under affine or plane projective assumptions) or between 3D point clouds. However, the case of matching between 3D and 2D features has not been addressed due to the lack of any geometric invariants between them. [Ochs and Brox, 2012] apply spectral clustering on a projected hypergraph computed from higher-order tuples of motion trajectories. Using affinities beyond just pairs of trajectories allows them to handle non-translational motion like rotation and scaling.

2.4.2 Voting-based Techniques

Correspondenceless estimation of pose or scene structure has been addressed in [Makadia et al., 2007] where the approach employs a voting based setup instead of a graph-matching framework. However, their method relies on appearance matches between SIFT descriptors and thus does not address the problem setting where no appearance features are available.

2.5 Image-based Geo-Localization

As an important application of matching between images with disparate appearance, we consider the image-based geo-localization problem. Geo-localization of street-level query images using a reference image/3D database has been a research focus of a number of varied approaches in the past decade. In the case of the reference image data, the imagery might be captured either from street-level as well or from the air. In both of these cases, issues related to disparate appearance matching pose
an important challenge. In case of reference 3D data, issues related to incorporation of geometric constraints in the matching problem need to be addressed as there is no direct appearance information that can be matched.

2.5.1 Geo-localization using Street-level Image Data

In case the reference data consists of imagery captured at the street-level, the image matching problem is made difficult by the appearance variation due to viewpoint differences and matching ambiguities because of repeated structure on building facades. Thus, techniques to handle this problem focus on more discriminative features as well as on explicit detection and handling of facade regions.

Matching of facades. [Doubek et al., 2010] match the similarity of repetitive patterns by comparing the grayscale tiles, the peaks in color histogram, and the sizes of the two lattices. In [Schaffalitzky and Zisserman, 1999], corners are extracted and grouped according to consistency with the geometric transformations corresponding to the generators of the lattice. [Kosecka and Zhang, 2005] extract rectangle projections by grouping line segments according to vanishing point consistency. Using [Zhang and Kosecka, 2006] they match a query street-view image to a database of geo-tagged street-view images using wide-baseline matching. [Schindler et al., 2008] detect lattices by mapping quadruples of SIFT features to the projective basis and checking the consistency of the rest of the points with respect to this basis. They combine multiple 2D-to-3D pattern correspondences and recover the camera orientation and location as an intersection of the family of solutions obtained using each correspondence.
Feature indexing methods. [Zamir and Shah, 2010] employed a structured dataset of 360° panoramic imagery from Google Street-view to create an index of SIFT features which is used to geolocate a query image. Their method requires an extensive dataset of street-level reference imagery to be available and be indexed. SIFT feature matching was also employed for urban localization by [Zhang and Kosecka, 2006]. [Hays and Efros, 2008] propose a data-driven approach to single-image localization which also uses scene features from a large dataset.

In [Cipolla et al., 2004] and [Robertson and Cipolla, 2004], a query street-view image is again matched to a database of street-view images and then used to compute the camera pose. They assume the query image camera internal parameters to be known and use a pyramid to match at multiple scales using geometric consistency. In [Chekhlov et al., 2008], quantized descriptors based on Haar coefficients are used to build an index for relocalization in SLAM. In [Molton et al., 2004], normals of local planar patches are estimated along time and used to enable wide-baseline matching through warping of the patches. Morel and Yu introduced the affine SIFT (ASIFT, [Morel and Yu, 2009a]) to account for significant viewpoint variation. A-SIFT simulates affine distortions and uses plain SIFT to compare warped patches differing only by roll, rotation and scale. In [Wu et al., 2008a], a viewpoint normalization of planar patches is followed by SIFT computation of the rectified patch. The use of derived geometric features for geo-localization was shown by [Cham et al., 2010] where omnidirectional views are matched to building outline maps by detecting the tallest vertical corners of the buildings which are matched through 2D to 1D projection.
2.5.2 Geo-localization using Aerial Image Data

More recent work on image-based geolocation has also looked at using non-ground-level database imagery. Although direct feature correspondence is not employed in these approaches, appearance information is still used either in a bag-of-words or a feature learning framework. Examples of these two frameworks include work on using self-similarity bag-of-words features for matching to oblique aerial imagery [Bansal et al., 2012], multiple feature learning for matching to satellite and land cover attribute imagery [Lin et al., 2013] and static camera localization by correlating with satellite imagery [Jacobs et al., 2007].

Recently, [Bansal et al., 2011a] established the feasibility of matching highly disparate street view images to aerial image databases to precisely geo-localize SV images without the need for GPS or camera metadata.

2.5.3 Geo-localization using 3D Reference Data

Image-based geo-localization has largely been approached as an appearance matching problem between a query image and a database of geo-tagged images. However, more recent works have looked at using only 3D reference data because of the ease with which it can be acquired. In particular, Digital elevation models (DEM) and 3D models of the environment have shown promise for the geo-localization problem by relying on rendering-based techniques [Ramalingam et al., 2009, Matei et al., 2013, Baatz et al., 2012]. Typically, these techniques render the 3D model on a uniform grid in the ground-plane, compute features for each rendering and then match these against the query features to retrieve candidate locations. While these techniques have proven efficient and effective for geolocating in a mountainous terrain [Baatz et al., 2012], their adaptation to urban environments has not had the same
level of success. The reason for this is the computational overhead of rendering the building models at a fine enough resolution such that the rendering can closely match the query image. [Baatz et al., 2012] described a framework for geolocating queries in a mountainous terrain by matching against skylines pre-rendered from digital elevation models. Rendering based techniques have also been employed in an urban geo-localization setup. [Matei et al., 2013] used a LIDAR scan of the environment to create a DEM which is rendered exhaustively from multiple locations and viewpoints. Features extracted from these renderings are matched against query features to generate candidate camera locations.

[Ramalingam et al., 2011] present a formulation for computing camera pose using minimal configurations of points and lines, and use this to geolocate a query using the 3D model of a city. However, their approach demands the availability of an initial correspondence between one query image and the 3D model. This correspondence is used to setup 3D-to-2D constraints which are then propagated to a new query image using image-to-image appearance matching. Thus, they do not address the geo-localization problem in the traditional sense and implicitly use image appearance. Skylines precomputed from a 3D model have been used for urban geolocalization of an omni-camera in [Ramalingam et al., 2009]. However, the approach has shown more promise for keeping track of the camera location rather than for initialization.
Chapter 3

Ultra-wide Baseline Facade Matching for Geo-Localization

Consider the problem of matching facade imagery from very different viewpoints – like from a low flying aircraft and from a street-level camera (see Fig. 3.1). Such images are characterized by unmitigated differences in local appearance which render any comparison of bags of visual words infeasible. A visual comparison of this imagery even after rectification testifies to the hardness of the problem. Moreover, a vast majority of facades contain repetitive patterns which make correspondence estimation highly ambiguous. We rather have to rely on comparing the structures of the facade patterns and still account for any transformations between such structures.

In this chapter, we address the facade matching problem in a geo-localization setup where street-view (SV) query images need to be matched against a database of pre-processed bird’s-eye-view (BEV) images. The key idea is to avoid direct matching of features to solve this extreme case of wide-baseline matching. Thus, we formulate the problem as “embeddings” within each respective dataset (SV and BEV) so that large variations are incorporated within the structure of embeddings.
This idea has not been explored before especially in the context of air-ground matching. We make the following contributions to the state of the art: (a) we introduce an approach for matching image regions with significant appearance, scale, and viewpoint variations based on a novel Scale-Selective Self-Similarity ($S^4$) feature that combines intrinsic scale selection with self-similarity descriptors, and (b) we demonstrate a novel system for matching street-level queries to a database of birds-eye views. We show experimental results on the retrieval accuracy from our technique and compare our performance with standard SIFT-descriptors.

We approach the facade detection and matching problem from a combined statistical and structural viewpoint. While other approaches model the lattice structure explicitly [Park et al., 2009], we capture the statistical self-similarity (or dissimilarity) of a local patch to its neighbors. By avoiding using a specific feature like SIFT, MSER, or line segments, we can capture this structure at any point – in implementation we do it on a randomly jittered grid. In addition, the self-similarity descriptor also captures the dis-similarity between neighboring elements ignored in lattice approaches but still observed e.g. in [Chung et al., 2010]. The challenge with self-similarity is to capture the intrinsic local scale governed by the periodicity/generator group of a lattice. We estimate the scale by discovering the closest
most salient repetition of a patch which can be centered anywhere. With the exception of [Hays et al., 2006], other approaches rely on the robustness of interest point or line segment detectors. Having obtained the intrinsic scale enables us to compute the scale-invariant $S^4$ descriptor and also allows us to detect facades as clusters of such points in space that have similar scale and descriptors. Similar descriptors are obtained from the query street-level image as well. At this point, instead of lattice or graph matching [Hays et al., 2006, Chung et al., 2010], we apply a labeling approach that labels each query descriptor with the most probable facade label (cluster) in a naive-Bayes sense. This way, we match local lattice structures rather than global ones and the most likely closest database facade is obtained.

3.1 Scale-Selective Self-Similarity Features

The viewpoint and appearance difference between oblique Bird’s-Eye-View (BEV) and street-view (SV) imagery is too large to be captured by direct matching of descriptors like SIFT and MSER. Therefore, we propose to create a descriptor that captures the structure of repetition of patterns or more generally the relative similarity between local patches within facades. Instead of modeling the structure with a graph or lattice and relying on the robustness of the detection of their nodes, we define a new feature which we call the Scale-Selective Self-Similarity or $S^4$ feature. This feature improves upon the well-known self-similarity descriptor from [Shechtman and Irani, 2007a] by adding a SIFT-like scale-normalization to allow characterization of the self-similar structure in a scale-invariant manner.

Using the same notation as [Shechtman and Irani, 2007a], for a given pixel $q$, the local self-similarity descriptor $d_q$ is computed as follows. A local image patch of width $w_{ss}$ (e.g., 5 pixels) centered at $q$ is correlated with a larger surrounding image
Figure 3.2: Comparison of local self-similarity and SIFT descriptors on facade regions. (a) Rectified facade images from SV (top) and BEV (bottom), (b) corresponding SIFT descriptors for the blue region, (c) corresponding self-similarity descriptors for the blue region obtained by correlating the green region. The self-similarity descriptors are noticeably more similar than the SIFT descriptors.

region of radius $r_{ss}$ (e.g., 40 pixels), resulting in a local internal ‘correlation surface’. The correlation surface is then transformed into a binned log-polar representation which accounts for increasing positional uncertainty with distance from the pixel $q$, accounting, thus, for small local variations in scale, orientation and shear.

Fig. 3.2(a) shows a pair of ortho-rectified SV and BEV images of the facade in Fig. 3.1, after manual normalization to the same scale. Fig. 3.2(b) and (c) compare how well their self-similarity descriptors match relative to their SIFT descriptors. The self-similarity descriptor at the center of the green ROI (local patch) is computed by correlating within the surrounding support region (blue ROI). The computed descriptors are noticeably quite similar even with the large appearance difference between the images themselves. In comparison, the SIFT descriptors computed using the same support region are dissimilar due to the noisy gradients they employ.

**Scale-Selection.** While it is clear that the inherent self-similar structure in
building facades can serve as a good matching criterion, it is not clear how that structure can be matched if the building is seen at different scales. The basic self-similarity descriptor discussed above assumes a distance binning which is not scale invariant. To account for feature scale differences, [Shechtman and Irani, 2007a] suggest computing the self-similarity descriptors on a Gaussian image pyramid representation and then searching for the template object across all scales. For the purposes of retrieval, however, such an approach would not work. In particular, for building facades, capturing the self-similar structure at all scales will reduce the discriminability evident at the fundamental scale of the facade. Instead, we would like a SIFT like normalization so that the descriptors between differently scaled buildings can still be matched. The repetitive structure of building facades provides one such normalization scale. However, building facades typically also exhibit local periodicity. While recovering this scale will serve the purpose of a valid normalizing scale, it may compromise on the overall discriminability of the computed descriptor by (a) being too local, and (b) by being too dependent on the inherent image scale (the smallest scale structure will be lost first in a noisy query image).

In this work, we focus on recovering the motif scale. We define the motif scale at a pixel in the facade as the smallest wavelength at which any patch in this pixel’s local neighborhood repeats. Defined this way, a local window scale would be ignored if it is not consistent with a few other window pixels in its neighborhood – thus making this scale robust against local pattern noise. This motif scale can be measured independently in both horizontal and vertical directions; in our implementation, we have only used the horizontal scale (denoted as \( \lambda_x \)), but the approach is symmetric with respect to using either of the two. Given the motif-scale \( \lambda_x \) value at any pixel, the \( S^4 \) descriptor is defined as the self-similarity descriptor computed by setting the patch size \( w_{ss} \) to the estimated motif scale \( \lambda_x \) and the correlation radius to \( r_{ss} = 2\lambda_x \).
Our approach for motif scale-selection is based on the peaks in the autocorrelation surface in a local neighborhood surrounding a pixel. Consider a pixel \((x, y)\) inside an image \(I\) exhibiting periodic structure and let \(\lambda_x\) be its scale along the x-direction. Now consider a small \(w \times h\) patch of pixels around this pixel and correlate it with patches extracted at various offsets \((r, \theta)\) in a polar representation. To capture the correlations most relevant to the self-similarity descriptor, we measure the correlation profile using the following SSD measure. Let \(J(s, t) = I(x + s, y + t)\), then:

\[
q(r, \theta) = \sum_{t_y = \frac{w}{2}}^{\frac{h}{2}} \sum_{t_x = \frac{w}{2}}^{\frac{w}{2}} \left( J(t_x, t_y) - J(t_x + r \cos(\theta), t_y + r \sin(\theta)) \right)^2
\]  

(3.1)

Then, the correlation profile \(p_{(x,y)}(r)\) is computed by integrating the scores \(q(r, \theta)\) in a 20° lobe \((\theta_0 = 10°)\) around the horizontal direction:

\[
p_{(x,y)}(r) = \exp \left( -\frac{1}{2\theta_0 + 1} \sum_{\theta = -\theta_0}^{\theta_0} q(r, \theta) \right)
\]  

(3.2)

where the subscript \((x, y)\) makes explicit the fact that the profile was obtained by correlating the patch around pixel \((x, y)\). The angular integration provides robustness against image distortions and ortho-rectification errors. The value of \(r\) is varied such that \(r \in \{1, \ldots, S_{\text{max}}\}\), where \(S_{\text{max}}\) is a pre-defined maximum scale value we expect the structure in the input image to exhibit. The correlation profile thus obtained captures the periodicity of the structure by producing the highest correlation for \(r \in \{\lambda_x, 2\lambda_x, \ldots\}\). However, depending on the starting location \((x, y)\), the correlation profile can exhibit peaks at \(r\) values which are non-integral multiples of \(\lambda_x\). This will be the case if the patch contains a sub-motif of the facade which is
Figure 3.3: Scale selection. To determine the scale $\lambda_x$ of the (black) 1D signal in the second row, if we autocorrelate a patch of width $w$, we get one of the profiles shown in rows 3-7 depending on the starting offset. However, for a poor offset choice (green and blue curves), one can get comparable peaks in the correlation profile for scale values $\lambda_x^\prime < \lambda_x$ making it difficult to extract the correct scale. Integrating across these profiles, however, resolves this issue and results in a well defined profile $p_{avg}(r)$ shown in the first row. The high peaks now correspond to the correct wavelength $\lambda_x$.

locally periodic at a higher frequency. The illustration in Fig. 3.3 depicts this happening for the green and blue profiles obtained from the (black) 1-D signal. The wavelength of both these curves is smaller than the motif scale $\lambda_x$ by our definition above. To alleviate this issue, we compute multiple correlation profiles by varying the starting offset in an interval $\mathcal{O} = \{(x, y), (x + 1, y), (x + 2, y), \ldots, (x + m, y)\}$. The maximum offset $(x+m, y)$ is set so that the patch around it covers the structure at the maximum scale $S_{max}$ from the starting position i.e. $m + w/2 \geq S_{max}$. The correlation profiles are combined into a single profile $p_{avg}(r)$ by integrating across the offsets, i.e. $p_{avg}(r) = \sum_{o \in \mathcal{O}} p_{o}(r)$. This removes the higher-frequency peaks in the individual profiles, leaving only the peaks corresponding to the actual wavelength $\lambda_x$ as depicted in Fig. 3.3. Furthermore, the scale estimation becomes independent of the choice of the patch dimensions $w$ and $h$. The idea of summing across multiple correlation profiles to resolve ambiguities within individual profiles and to obtain a clear minimum was also used in the framework of a multi-baseline stereo algorithm.
in [Okutomi and Kanade, 1993].

To be robust against shallow peak responses, we measure a peakness measure around each peak in the profile $p_{avg}(r)$ and prune peaks which are shallower than a threshold $t_{peak}$. This threshold is set empirically by running the scale-estimator on textureless and non-repetetive structures. From the locations of the remaining peaks, the scale value $\lambda_x$ can be readily obtained by a discrete Fourier transform. In the absence of any peaks the underlying structure is labeled aperiodic (assigned scale zero) – this removes most of the non-facade pixels and serves as an effective building detection mechanism.

### 3.2 Facade Extraction and Segmentation

We now describe our general approach for extracting building facade regions which is applicable to both BEV and SV images. The key idea is to exploit the self-similar structure of building facades: ortho-rectify the image, compute motif scales at sampled locations in the given image, compute $S^4$ descriptors at the computed scales and then cluster the descriptors to group similar structures together.

**Motif scale computation.** In the rectified image, we sample a grid of pixel locations every $\sigma_f = 5$ pixels apart and add uniformly random spatial jitter of amplitude $\sigma_f/2$ at each sample location. This jitter allows us to capture a good sampling of the feature distributions expected from this facade structure at the matching stage. At each sample location, we compute the motif scales using the approach discussed in section 3.1. An example result at this stage is shown in the left half of Fig. 3.4. Note that the scale selection has removed the non-building areas almost completely by labeling them with a zero scale value (shown as red dots in the figures). Also note the wide range of motif scales seen across buildings stressing
Facade Segmentation. At each sample location, we compute the $S^4$ descriptor ($n_\theta = 20$ angular bins and $n_r = 4$ distance bins) by setting the patch size $w_{ss}$ to the estimated motif scale $\lambda_x$ and the correlation radius to $r_{ss} = 2\lambda_x$. Now, we perform K-means clustering in this $S^4$ feature space using $L_1$ norm as our distance measure. To avoid descriptor grouping across different buildings, we penalize clustering of descriptors which were sampled from far off locations. The desired number of clusters $N$ is set as follows. We manually mark the boundaries of a small number of buildings (5 in our case) in the BEV image and initialize $N = N_0$. Now, we iteratively run K-means with decreasing value for $N$ as long as the following invariant is maintained: clusters on the marked buildings are contained within the marked boundaries. At the end of this process, we obtain a clustering that has the fewest number of clusters within each building and does not merge two different buildings into a single cluster (note that this is not guaranteed for unmarked buildings in general, but due to the descriptor-based grouping, we have not seen any merging of separate buildings into
Figure 3.5: Scale Extraction and over-segmentation of facades from rectified BEV image.

(a) BEV Image  (b) Horizontal Scales  
(c) Vertical Scales  (d) K-Means Clusters

Notation. In the following, we will denote the $S^4$ descriptor vectors obtained from the entire set of BEV imagery by words $\mathcal{V} = \{v_1, v_2, \ldots, v_m\}$, the cluster labels as $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$ and the labeling function mapping each word to its cluster...
3.3 Facade Matching

Given a query street-view image, we would like to retrieve facades from our BEV database that match the dominant facade(s) in the query. Sec. 3.4.3 and Fig. 3.8 illustrate the key steps in our SV-to-BEV matching pipeline. After ortho-rectification, motif scale selection and $S^t$ descriptor computation, we obtain a set of descriptor vectors $\mathcal{W} = \{w_1, w_2, \ldots, w_n\}$ from the query. For each of these words, we would like to estimate the probability $p(C = c_k|w_i)$ of being assigned to one of the clusters $c_k$ in $\mathcal{C}$. The problem of finding the closest cluster label for each word $w_i$ can be formulated in a Bayesian settings as follows. By Bayes’ theorem,

$$p(C = c_k|w_i) = \frac{p(w_i|C = c_k)p(C = c_k)}{\sum_{j=1}^{N} p(w_i|C = c_j)p(C = c_j)}$$

For each word $w_i$, we estimate the likelihoods $p(w_i|C = c_k)$ by kernel density estimation using a Gaussian kernel $\mathcal{K}(w_i, v_j)$ with wavelength parameter $\sigma_K$. The likelihood is then computed as:

$$p(w_i|C = c_k) = \frac{1}{|c_k|} \sum_{\mathcal{L}(v_j) = c_k} \mathcal{K}(w_i, v_j)$$

where $|c_k|$ denotes the cardinality of cluster $k$. The prior probability $p(C = c_k)$ is simply set from the sample proportions: $p(C = c_k) = \frac{|c_k|}{m}$. For each word $w_i$, we estimate the MAP estimate of the label by choosing the label $k$ with the maximum a-posteriori probability: $\mathcal{L}(w_i) = \arg \max_k p(C = c_k|w_i)$. Given the above word assignments, we can now compute the most probable label for the entire query facade.
Algorithm 3.1: BEV processing
1. Ortho-rectify BEV image using vanishing points.
2. Compute motif-scale $\lambda_x$ at a jittered grid of pixel-locations on the BEV.
3. Compute $S^4$ descriptors $v_i$ at locations with non-zero scales.
4. Cluster $S^4$ descriptors $v_i$ using K-means to obtain label-set $\mathcal{C}$ and labeling function $\mathcal{L}$.

Algorithm 3.2: SV processing
1. Ortho-rectify SV image using vanishing points.
2. Compute motif-scale $\lambda_x$ at a jittered grid of pixel-locations on the SV.
3. Compute $S^4$ descriptor-set $\mathcal{W} = \{w_j\}$ at locations with non-zero scales.
4. Compute labels $\mathcal{L}(w_j)$ using Eqn.3.3.
5. Best matching BEV facade: Facade containing cluster $\mathcal{L}(\mathcal{W})$ (Eqn.3.6).
6. Top matching facade set: For threshold $t$, return facades containing clusters $k$ s.t. $f(k) > t$ (Eqn.3.5).

Table 3.1: Parameter settings

<table>
<thead>
<tr>
<th>$w$</th>
<th>$h$</th>
<th>$S_{\text{max}}$</th>
<th>$\sigma_f$</th>
<th>$w_{ss}$</th>
<th>$r_{ss}$</th>
<th>$n_{\theta}$</th>
<th>$n_r$</th>
<th>$N_0$</th>
<th>$\sigma_K$</th>
</tr>
</thead>
<tbody>
<tr>
<td>13 px</td>
<td>13 px</td>
<td>48 px</td>
<td>5 px</td>
<td>$\lambda_x$</td>
<td>$2\lambda_x$</td>
<td>20</td>
<td>4</td>
<td>100</td>
<td>2.5</td>
</tr>
</tbody>
</table>

by accumulating the word assignments from each word:

$$f(k) = \sum_{i} \delta(\mathcal{L}(w_i) = c_k)$$ (3.5)

$$\mathcal{L}(\mathcal{W}) = \arg \max_k \{ f(k) \mid k = 1, \ldots, N \}$$ (3.6)

where $\delta(.)$ is the indicator function. The label $\mathcal{L}(\mathcal{W})$ identifies a cluster $c^* \in \mathcal{C}$ which, by construction of the clustering algorithm, identifies a single BEV facade.

### 3.4 Experiments and Results

**Algorithm Parameters.** In Table-3.1, we list all the parameter settings we used in our implementation. The scale estimation process was found robust against different
Figure 3.6: Pittsburgh dataset.
choices of patch-size parameters $w$ and $h$. $S_{\text{max}}$ was set to a number greater than the maximum horizontal building scale for our BEV dataset (manually eyeballed). The $S^d$ values for $n_\theta$ and $n_r$ were set the same as in [Shechtman and Irani, 2007a].

**BEV and SV Imagery Datasets.** Our dataset comprises of BEV imagery (2000 $\times$ 1500 pixels) downloaded using Microsoft’s Bing service for an area approximately 2 Km$\times$1.2 Km in size (Fig. 3.6(a)) in downtown Pittsburgh, PA, USA. This dataset is challenging due to a large number (approx. 40) of buildings and very similar facade patterns. This dataset also covers a much larger area than used in related works in air-ground-based localization e.g. 440m $\times$ 440m in [Cham et al., 2010]. Street-view images downloaded using Panoramio, Flickr, Google Street-View(screenshots), and Microsoft Bing’s Streetside(screenshots) were used as queries. For ground-truth purposes, only the SV imagery with geo-tags or visually identifiable facade correspondence (with the BEV) was retained.

**Imagery Rectification.** We rectify BEV to an orthographic view aligned with the dominant city-block direction. Similarly, the SV imagery is rectified to an orthographic view of the dominant facade in the scene using the Geometric Parsing based vanishing point estimation approach and code [Barinova et al., 2010, Tardif, 2009]. For the BEV imagery, we adapted the approach to handle the high density of lines detected in these images. Using the estimated vanishing points corresponding to the two facade axes, we obtain pairs of extremal rays in the horizontal and vertical directions. Intersecting these gives us four corners of a quadrilateral which are then used to estimate a rectification homography using the approach in [Kosecka and Zhang, 2005]. This transformation warps the facade to be fronto-parallel and also corrects the aspect ratio between the horizontal and vertical directions.
3.4.1 Scale Selection Results

To characterize our scale selection algorithm, we selected a test set of 10 building facades extracted from the Pittsburgh BEV dataset. We manually measured the ground-truth horizontal scale(s) for each facade and compared them to those estimated by our approach. Since we densely estimate these scale values over the facade, we computed a histogram of the estimated scale values and the normalized histogram values are shown as the blue circles (with radii proportional to the histogram values) in the bubble plot of Fig. 3.7. The red pluses denote the ground-truth scale values – multiple in cases where the facade exhibits more than one motif scale. The comparison shows the accuracy of our scale estimation and the presence of very few outliers.

3.4.2 Facade Detection Evaluation

Table-3.2 shows results from our facade detection algorithm. For each BEV scene, we looked at the computed horizontal scales – points with non-zero scale values
Table 3.2: Facade detection performance

<table>
<thead>
<tr>
<th>Scene</th>
<th>TP Rate</th>
<th># Buildings</th>
<th># FPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEV-1</td>
<td>86%</td>
<td>29</td>
<td>8</td>
</tr>
<tr>
<td>BEV-2</td>
<td>91%</td>
<td>33</td>
<td>3</td>
</tr>
<tr>
<td>BEV-3</td>
<td>86%</td>
<td>21</td>
<td>5</td>
</tr>
</tbody>
</table>

are treated as potential facades. We quantify the performance as follows: for each building facade, if at least 50% of its visible area was assigned a non-zero scale, then we count it as a true detection. If in any $4 \times 4$ sub-grid of sampled locations not on a building facade, at least 25% are assigned a non-zero scale, then we count it as a false-positive.

### 3.4.3 SV to BEV Matching

Fig. 3.8 illustrates our typical query SV processing pipeline. The algorithmic steps are outlined in Algorithm-2. First, the query SV image is ortho-rectified [Barinova et al., 2010, Tardif, 2009]. Next, the motif scale computation algorithm described in Sec. 3.2 is employed to compute the horizontal scales on a uniform grid. $S^4$ descriptors are then computed at locations with non-zero scale producing the query word set $\mathcal{W} = \{w_1, w_2, \ldots, w_n\}$. The facade matching algorithm in Sec. 3.3 is used to label each word with a cluster label.

Fig. 3.9(a) shows the retrieval performance of our approach (along with a comparison with SIFT – details in Sec. 3.4.5) with a query set of 79 images including 33 true negatives i.e. buildings which were either not part of the BEV database or were significantly occluded. The query set contains challenging images with significant uncorrected image distortions, urban clutter and varied zoom range. A third of these images are high-resolution pictures from Flickr and Panoramio and the remaining are low-resolution screenshots from Google Street-View and Bing Streetside. A few
(a) Query Street-view

(b) Ortho-rectified Street-view

(c) Street-view Horizontal Scales

(d) Street-view Vertical Scales

(e) Descriptor Sampling Grid. Red crosses – jittered locations from blue grid points

(f) Matching result with BEV with correspondingly matching clusters shown in same colors.

Figure 3.8: Example Street-view processing.
Figure 3.9: (a) ROC curve for BEV-to-SV matching on Pittsburgh dataset, (b) SV-to-SV matching performance on the “Pankrac+Marseilles” public dataset.

samples from the query set are shown in Fig. 3.6(b). For generating the ROC curves, instead of using the most probable label from Eqn.3.6 directly, we treat the vector of frequency of each label \( f(k) = \sum_i \delta(L(w_i) = c_k) \) as a probability distribution. Then, to get a point on the ROC curve, we pick a value between 0.0 and 1.0 and select all the labels with probabilities higher than this value. This becomes our retrieval set which is compared with the ground-truth facade set to compute the TP and FP rates in the usual manner.

Fig. 3.11 shows few examples of the top three retrieval matches on representative (screen-captured) Google street-view queries. From the amount of perspective (and distortion) in the SV imagery, it is clear that features like MSER and SIFT would hardly find any correspondences. The bottom row shows some problem cases due to the scale difference between the SV and BEV images or due to the global rectification of the BEV image – as expected, the descriptors from non-fronto-parallel facades do not match well to fronto-parallel rectified SV imagery.
3.4.4 Street-View to Street-View Matching

We present results of our approach applied to the public dataset Pankrac-Marseille\textsuperscript{1} of SV-only images. This dataset contains 106 images of approx. 30 buildings from Pankrac, Prague and Marseille appearing in more than one image, number of appearances ranges from 2 to 6. Fig. 3.9(b) shows the performance of our approach on this dataset and compares it with the best performance shown by [Doubek et al., 2010]. Their approach uses detection of repeated lattice tiles followed by appearance features on the detected tile pattern as a means to match between facades in SV imagery and it is not surprising that we come close to their results using the self-similarity descriptor. In fact, the SV-to-SV matching problem does not entail the same challenges as the SV-to-BEV matching we address here. In the former, once the projective distortions are removed, one can still achieve good performance using direct feature matching across images because the appearance does not look as dissimilar as in the case of SV vs. BEV. A qualitative snapshot of our results is shown in Fig. 3.10.

3.4.5 Comparison with SIFT Features

Given the prevalence of SIFT features in wide-baseline matching literature, we present experimental comparison of its performance with our approach. To avoid any bias against SIFT due to perspective distortions (and to preclude comparison with SIFT variants like A-SIFT), we extract SIFT features on ortho-rectified BEV and ortho-rectified SV imagery. Next, we use the building clusters found using our $S^4$-based algorithm and perform an assignment of the SIFT features to these clusters.

\textsuperscript{1}http://cmp.felk.cvut.cz/data/repetitive
using a nearest-neighbor association on pixel coordinates thus discarding any features on non-building background clutter. The Bayesian classification from Sec. 3.3 is used on the SIFT clusters to retrieve matching facades for the query images and the quantitative results are shown in the ROC in Fig. 3.9(a) which illustrates that we achieve significant improvement in performance using $S^4$ features instead of SIFT features.

3.4.6 Camera Pose Estimation

Facade matching is in itself good enough to localize the SV image within a constrained visibility zone defined by the facade. However, for precise localization of the SV camera we compute the 6 DOF pose of the camera to establish the efficacy of our method. We have developed two different algorithms for this task and we briefly describe them below.

**Manual Correspondence Algorithm.** In this method, we manually identify 7
Figure 3.11: Qualitative Matching Results. The main tiles show rectified BEV images. The insets show the original and rectified query street-view facades. On the rectified inset, the colored points are a subset of the words $w_1, w_2, \ldots, w_n$ with the top three most frequent recovered labels $L(w_i)$ shown as red, green and blue points respectively; similarly colored points in the BEV image are words $v_j$ which belong to these three clusters. Top two rows: some examples of correct retrieval. Bottom row: Mismatched result (left) due to missing fine structure in the street-view image that was seen in the BEV image. Correct matching result for another street-view image which shows the fine structure is shown in the middle. The bottom right example shows a problem scenario where the non-fronto parallel facade in the BEV causes it to be mismatched due to the difference in the self-similarity structure presented by the descriptors.
Figure 3.12: Localization Results. The top row shows the SV images, and the bottom row shows the estimated and ground-truth camera locations.

Point correspondences between the SV and BEV image in the structure surrounding the matched facade. These correspondences are used to estimate the fundamental matrix \( F \) [Hartley and Zisserman, 2003] between the SV and BEV images. The epipole of the BEV image, as computed from \( F \), then corresponds to the SV camera location in the BEV coordinate system.

**Automatic RANSAC Algorithm.** In this method, we use purely geometric constraints to simultaneously estimate both the correspondences as well as the camera pose. We use the plane+parallax methodology for this problem. We use the recovered matching facade to estimate a homography between the BEV and SV imagery and then use this homography to measure the parallax corresponding to any point correspondence between the SV and BEV imagery. We employ constraints from the SAT imagery to make this problem more tractable. We use a few corners from the SV image and building top corner correspondence (between BEV and SAT imagery) to enforce the parallax geometry and recover the SV camera location.
The top corner correspondence itself, can be extracted from the facade extraction phase where we explicitly detected the top edge of each building when using the stereo-based algorithm.

The SV camera location in the BEV image is mapped to absolute lat-long coordinates using the ground plane correspondence with the SAT imagery. Finally, the metric (cms/pixel) information in the SAT image is used to estimate the camera focal length which can be used in conjunction with any knowledge about the CCD array dimensions to establish the camera field-of-view as well. The camera look-at direction is also estimated using the metric information available from the SAT imagery by a simple trigonometric calculation. Fig. 3.12 shows the localization results obtained for three query images using the manual correspondence algorithm.

3.5 Facade Extraction using Satellite Imagery

In Sec. 3.2, we described an approach to extract facade regions from BEV imagery by clustering $S^4$ descriptors. However, this technique would not be able to distinguish between nearby buildings if they exhibit similar facade pattern. We have developed a geometric technique [Bansal et al., 2011b] to extract building facades from BEV
imagery by aligning it with satellite (SAT) imagery when available.

Given a BEV image and a SAT image for the same region of interest, first, we align the ground plane between the SAT and BEV imagery. Thus, BEV images can be rectified with respect to the ground plane with canonical axes (N-S, E-W) aligned. Then, we match building outlines extracted from SAT imagery with the corresponding outlines in the rectified BEV images. Subsequently, we use the identified building outlines to find the roofs of buildings thus identifying the facades. This allows extraction of ortho-rectified building facades from the BEV from which $S^4$ features can then be extracted and matched using the matching algorithms described before. In the following, we describe in sequence the above algorithmic steps in more detail.

**Imagery Alignment:** Given the set of SAT and BEV image tiles (see examples in Fig. 3.13) and the mapping of their pixel coordinates to lat-long coordinates, we can warp the BEV images to the SAT coordinate system. To compute the warping transformation, we approximate it as a projective transformation between pixels in SAT and BEV – thus approximating the Earth’s surface within each tile as a flat plane. Using the computed transformations, we warp each of the images to the
SAT image coordinate system. As a result, the ground plane gets aligned well in all the images as shown in Fig. 3.14. To aid further processing, we also compute the dominant city block direction in the SAT imagery and rotate this image before warping the other images to its coordinate system. This renders most of the buildings parallel to the scan-lines in the image – a feature which will be exploited in further processing.

After initial imagery rectification, we extract regions from the BEV imagery corresponding to building facades. To ensure least distortion, we concentrate only on the facade planes which face the heading direction of the particular BEV image. Since the SAT imagery was previously rotated to align the city blocks with the image scan-lines, we can now restrict our attention to facade planes whose 2D projections are horizontal in the SAT images.

**Vertical Vanishing Point Estimation:** In the ground-aligned BEV imagery, lines along the vertical (gravity) vanishing direction can be seen to be convergent. Before extracting affine corrected facades, we first rectify the BEV imagery so that these lines are rendered parallel. We detect canny edges in the BEV image and then group these edges into line segments. Lines along horizontal and vertical directions correspond to city block axes and can hence be rejected. From the remaining line segments, a RANSAC-based process then determines the inlier set of lines that intersect at the required vanishing point.

**Image Rectification:** Given the computed vanishing point, we now rectify the BEV image by mapping this vanishing point to a point at infinity (in particular to \(v_x = [1, 0, 0]^t\)), thus making the building edges parallel. This rectifying transformation is a projective warp which is computed by a method similar to the epipolar rectification method described in [Hartley and Zisserman, 2003]. Due to the choice of \(v_x\), the building facade edges in the rectified BEV become parallel to the image.
Figure 3.15: Example of the facade extraction process. a) detected building tops and bottoms, and b) extracted facade tiles.

Figure 3.16: Building top search. Line segments extracted from the SAT imagery are projected to the canny edge map of BEV where a sweep along the gravity direction is expected to give a maximal point at the top edge of the building.

scan-lines.

**SAT Edge Extraction:** To extract building facades from BEV, we start by detecting building contours in the overhead SAT imagery. The contours need to be detected as chains of line-segments, each segment corresponding to one face of a building. We developed an iterative algorithm to extract these line segments from an initial canny edge-detector processed SAT image. Briefly, the algorithm links edges into edge-chains based on proximity and then fits line segments to these edge-chains, splitting wherever the deviation of the edges from the fitted line segment
Figure 3.17: Effect of Graph-Cuts Optimization. The green edges are the SAT edges directly projected to this view and they lie in the ground plane. The red edges are the estimated building top edges. The top row shows the estimates obtained by picking the maximum score for each edge pixel independently; the bottom row shows these estimates refined by the GC optimization.

becomes greater than a threshold. Consistent line segments are merged into longer line segments and the overall process is iterated a few times.

**Facade ROI Search:** From the line segments extracted in the SAT imagery, we keep only the segments along the dominant facade direction in the BEV. Using the ground plane homography between SAT and BEV, we warp these segments into the rectified BEV image coordinate system. These segments then map to approximately the bottom of the buildings in the BEV image because the transformation corresponds to the ground plane. In the rectified BEV imagery, the gravity vanishing direction is aligned with the scan lines and therefore the tops of these buildings can be found by sliding the mapped line segments horizontally (Fig. 3.16). Our algorithm
to determine the building tops is described below. Once the building tops are determined, we obtain the coordinates of the four corners of each facade which can then be mapped back to the original (unrectified) BEV imagery for high-resolution texture retrieval. For each facade, we crop the texture from the original BEV imagery and then warp it into a rectilinear coordinate system. Fig. 3.15 shows an example of this process where a few of the facades are extracted and rectified to their orthographic representations.

**Computation of Building Tops using Graph-Cuts:** Given the nature of the rectified BEV imagery, the top of each building can be determined as a translation $\delta(s)$ for each segment $s$ projected to the building bottom. We formulate this problem as a Graph Cut (GC) optimization of an objective function that consists of the usual data and smoothness costs. The data cost for a line segment is strictly a function of the hypothesized translation and is computed by measuring the average edge strength in the rectified BEV image when the line segment is translated by this value. Thus, when the segment lands on the top of a building, we incur a lower cost due to the high edge strength. To ensure smoothness in the translation values for connected line segments, we add a smoothness cost that penalizes difference in translation values for line segments that are spatially close to each other at their endpoints. For the typical polygonal chains of line segments that we detect for each building, the smoothness cost enforces a strong constraint that the entire building top be at a single translation and avoids the problem of local optima occurring at the numerous edges in the middle of the building facade. Fig. 3.17 shows an example of how this optimization approach helps the building extraction process.
3.6 Related Work

In the discussion of related work, we emphasize two main aspects: detection of facades/lattices and matching. [Chung et al., 2010] extract MSER regions in multiple scales which are then clustered w.r.t similarity. Local histograms of gradient similarity, area ratio, and configuration entropy are used to build adjacency matrices which are matched by using a spectral approach comparing only the graph structure. The commonality with our approach is that we never use any direct comparison of appearance across images. On the other hand, their query and model graph structures have to match globally while our approach uses the statistics of the edges of these graphs represented by the self-similarity descriptor and hence exploits the redundancy in features better. Moreover, the self-similarity descriptor is more general and implicit than the concatenation of several neighborhood descriptions (HoG, area ratio, entropy). [Park et al., 2009] model the lattice discovery as a multi-target tracking problem using Mean-Shift Belief Propagation. Candidates for lattice vertices are interest points that are obtained through clustering. [Hays et al., 2006] randomly select regions and search for their repetition in two directions in their immediate neighborhood. Lattice discovery is formulated as a graph matching problem with higher-order constraints that model the lattice structure of the region repetitions. The advantage of [Park et al., 2009, Hays et al., 2006] is that they can deal with deformed lattices in the detection step while almost all other approaches including ours remove projective and sometimes affine distortions using vanishing points and ratio constraints. [Schindler et al., 2008] detect lattices by mapping quadruples of SIFT features to the projective basis and checking the consistency of the rest of the points with respect to this basis. They combine multiple 2D-to-3D pattern correspondences and recover the camera orientation and location as an intersection of the
family of solutions obtained using each correspondence.

Recently, [Bansal et al., 2011a] established the feasibility of matching highly disparate street view images to aerial image databases to precisely geo-localize SV images without the need for GPS or camera metadata. [Doubek et al., 2010] match the similarity of repetitive patterns by comparing the grayscale tiles, the peaks in color histogram, and the sizes of the two lattices. In [Schaffalitzky and Zisserman, 1999], corners are extracted and grouped according to consistency with the geometric transformations corresponding to the generators of the lattice. [Kosecka and Zhang, 2005] extract rectangle projections by grouping line segments according to vanishing point consistency. Using [Zhang and Kosecka, 2006] they match a query street-view image to a database of geo-tagged street-view images using wide-baseline matching. In [Cipolla et al., 2004] and [Robertson and Cipolla, 2004], a query street-view image is again matched to a database of street-view images and then used to compute the camera pose. They assume the query image camera internal parameters to be known and use a pyramid to match at multiple scales using geometric consistency. In [Wu et al., 2008a], a viewpoint normalization of planar patches is followed by SIFT computation of the rectified patch. We close our discussion with [Cham et al., 2010] where omnidirectional views are matched to building outline maps by detecting the tallest vertical corners of the buildings which are matched through 2D to 1D projection.

### 3.7 Conclusion

We have been able to match query street-level facades to airborne imagery under challenging viewpoint and illumination variation by introducing a novel approach of selecting the intrinsic facade motif scale and modeling facade structure through
self-similarity. Using the motif scale, we extract and segment lattice-like facades and construct scale-invariant $S^4$ descriptors. We localize queries by classifying descriptors, thus matching to facades with semi-local lattice consistency.
Chapter 4

Joint Spectral Correspondence for Disparate Image Matching

In this chapter, we focus on matching images with disparate appearance that do not necessarily exhibit a repetitive building facade-like texture pattern. Such images might be taken during day and night or in different times in history, and they differ at the local pixel level in the sense that neither intensity nor gradient distributions are locally comparable. Thus, we cannot rely on either pixel-level feature descriptors like SIFT or on local repetitive texture features like the $S^4$ features proposed in chapter 3. Instead, we need to look at both short and long range interactions of local gradient structures without apriori knowledge of the spatial extents of these interactions. Thus, the matching approach has to rely not just on novel local features but also on a suitable matching method that can exploit these interactions. In this chapter, we propose a novel spectral representation for the images that allows us to detect and match persistent features which robustly encode the appearance similarity we perceive when we look at such images.

Consider the images in Fig. 4.1 where we have the same scene captured at different
Figure 4.1: Day to night persistence matching. The spectrum of the joint image graphs is computed. The first row shows a day-time query-image (blue box) which is matched pair-wise against the pre-dusk, dusk and night images respectively from left to right. The second rows shows SIFT features detected on each image of the sequence. The third and fourth rows show the second eigen-vectors $J_1^{(2)}$ and $J_2^{(2)}$ for each pair of images (pre-dusk : day), (dusk : day), (night : day). The eigen-vectors corresponding to the query have a blue box to ease visualization. The plot compares the repeatability (bars) and average-precision (AP)(polyline) of the SIFT detector (blue) with the spectral method (red).

times of day\textsuperscript{1}. Visual comparison reveals the large amount of appearance change that

\textsuperscript{1}Frames extracted from the time-lapse sequence at: http://www.youtube.com/watch?v=00K4CdQ-haU
occurs in the scene due to the illumination variation. Numerous SIFT features are detected in these images and they show good repeatability (blue bars in the plot) as well. However, the average precision (AP) of the SIFT descriptors computed directly from these images significantly degrades as the illumination difference between the matched image pairs is increased as is visible from the blue polyline in the plot. In contrast, the spectral features we propose in this chapter are comparable in their repeatability (red bars) and they behave significantly better in the Average Precision (red polyline) even for the most challenging pair: night vs. day.

Spectral methods on the image graph laplacian have been used extensively in the literature for applications like clustering, segmentation [Arbelaez et al., 2011, Yu et al., 2002] etc. The extracted eigen-functions are either discretized to obtain the desired number of clusters or segments in the image or they are used directly as the spectral space coordinates of the pixels in an embedded space representation. These coordinates are then further clustered using K-means to obtain discrete clustering solutions. In contrast, we propose to use the individual eigen-functions themselves as a feature representation of the image pair from which interesting and useful feature correspondence can be derived. We show how such a representation captures persistent regions in the image pair even when the appearance difference between them is substantial (day-night, historic-new etc.). Moreover, we propose a new definition of the joint image graph: all pixels of both images are nodes and the corresponding edge weights depend only on the difference of the local image structures and not on the proximity between the pixels. Although a partitioning of such a graph might cluster together distant regions, these regions even though disconnected in the image space are persistent across images.

Our most significant contributions in this chapter are: (1) a new representation between two images: the joint image graph defined only based on the affinity between
image structures in the dense set of pixels from both images without considering the proximity between two image positions; (2) a new definition of persistent regions as the stable regions of the eigen-functions of that graph considered in their “soft” form without any discretization. We show that such persistent features are both repeatable across images and similar in terms of SIFT descriptors computed in the eigen space itself, in a variety of cross domain setups.

We show experimental results of our approach on the challenging dataset from [Hauagge and Snavely, 2012] which contains image pairs exhibiting dramatic illumination, age and rendering style differences. Our results clearly indicate the substantial matching improvement possible by looking at features derived from a joint image spectrum rather than relying on features detected individually in the two images to match in their descriptors. Unlike standard local-features approaches which detect features on each image independently, our method relies on computing features using both images simultaneously. However, we believe that the global information encoded in the joint image graph and its eigen-functions is the new cue that enables a better performance than approaches relying only on local neighborhoods.

4.1 Similarity in the Eigen Space

Consider the image pair in the first column of Fig. 4.2 depicting the same monument under significantly different illumination conditions. Each local facet of the monument is illuminated differently leading to dissimilar contrast and color characteristics. It is evident that finding features that are repetitive between the two pictures is a daunting task; in fact, the problem of finding descriptors that can account for the appearance differences at geometrically matching locations is itself quite challenging. On the other hand, we humans find it quite easy to ascertain by visual inspection
that these two images correspond to the same monument. The kind of features we use to make such a judgment are the more inherent “persistent” features in the scene like the contours, salient regions, the local shapes, patterns of contrast etc. One can argue, then, that shape-based image matching techniques should be applicable for matching these images. However, the contrast variations make it very difficult to detect the image contours robustly. Most of the dominant contours in the scene are very low energy and the intensity at which corresponding contours would get detected varies between different regions in the two images. Therefore, we propose a spectral approach that detects these persistent image features using the eigen-spectrum of the joint image graph computed from appropriate local gradients in the two images.

Before going into the details of the way the graph is constructed, let us focus on the images in the second to fifth columns of Fig. 4.2. In each column, the top and bottom images correspond to one of the eigen-functions of the joint graph reshaped back to the size of the images. The red and blue shades represent the maxima
and minima respectively, of the eigen-function. It is clear that, for each eigen-
function pair (i.e. images in a single column) the distribution and shapes of these
eigen extrema correspond well between the two images and the image regions where
this correspondence is strong is in agreement with the actually corresponding image
regions. Thus, by computing features that encode these extrema (both in their shape
and the eigen-energy profile), we can more robustly match these images without
relying on descriptors computed directly from the images.

The technical approach is organized as follows. First, we will review basic funda-
mentals of the image graph construction and its spectrum, followed by a look at the
actual features we use to build the joint image graph. Then, we will characterize the
eigen-function extrema as persistent regions and discuss algorithms to detect and
match these extrema.

4.1.1 Image Graph

The spectral analysis of the content of an image is carried out on a weighted image
graph \(G(V, E, W)\) which contains all the image-pixels as vertices in the vertex-set
\(V\) of cardinality \(n\). The edge-set \(E\) contains all pair-wise relationships between
every pair of vertices (pixels) in the set \(V\) thus making \(G\) a complete graph. The
weight \(w_{ij} \geq 0\) associated with an edge \((v_i, v_j) \in E\) encodes the affinity between
the pixels represented by vertices \(v_i\) and \(v_j\). We can collect these weights into an
\(n \times n\) affinity matrix \(W = (w_{ij})_{i,j=1,...,n}\). The degree matrix \(D\) of this graph is
defined as a diagonal matrix \(D\) with \(D(i, i) = \sum_{j=1}^{n} w_{ij}\). Using \(W\) and \(D\), we can
now compute the normalized graph laplacian \(\bar{L}\) as \(\bar{L} = I - \frac{1}{2}WD^{-1/2}\). We are
interested in the eigen-spectra \(U\) of this laplacian matrix which can be computed by
eigen-value decomposition \(\bar{L}U = \lambda U\) and setting \(U = D^{-1/2} \bar{U}\). The eigen-vectors
The formulation above can be easily extended from a single image to a pair of images as follows. Let $G_1(V_1, E_1, W_1)$ and $G_2(V_2, E_2, W_2)$ be the image graphs for images $\mathcal{I}_1$ and $\mathcal{I}_2$. Then the joint image graph $G(V, E, W)$ is defined such that $V = V_1 \cup V_2$, $E = E_1 \cup E_2 \cup V_1 \times V_2$ where $V_1 \times V_2$ is the set of edges connecting every pair of vertices in $(V_1, V_2)$. The affinity matrix $W$ is given by:

$$W = \begin{pmatrix} W_1 & C \\ C^T & W_2 \end{pmatrix}_{(n_1+n_2) \times (n_1+n_2)} (4.1)$$

where $|V_1| = n_1$, $|V_2| = n_2$ and $C$ is the $n_1 \times n_2$ matrix encoding the affinities of edges in $V_1 \times V_2$. The eigen-spectra for the joint graph can be computed exactly as before by defining the normalized laplacian $\bar{L}$ and carrying out its eigen-value decomposition.

### 4.1.2 Image Features and the Joint Spectrum

Consider first an experiment where we perform spectral analysis of the joint image graph $G(V, E, W)$ with the matrix $W$ defined directly in terms of the pixel color values in the two images, i.e. both the intra-image weights $W_1$, $W_2$ as well as the inter-image weights $C$ are defined by a function of the perceptually uniform Lab-space color difference between the pixel pair. Now, we compute the eigen-spectra of this graph’s laplacian to see if the corresponding eigen-functions show any
Figure 4.3: For the image pair in the first column, the successive columns show the second-through-fifth eigen-function pairs obtained using a pixel-color based joint image graph. In this case, the eigen-functions do not suggest any significant correlation with the region correspondence in the original images.

Figure 4.4: For the image pair in the first column, the successive columns show the second-through-fifth eigen-functions obtained using dense-SIFT-based image graphs. The eigen analysis is performed on each image graph independently. In this case, the eigen-functions show correlations but the correlated regions are distributed across several different eigen-functions.

patterns of correspondence. Fig. 4.3 shows the second through fifth eigen-function pairs (reshaped back into a matrix) for the same image pair as in Fig. 4.2 obtained using the above Lab-based graph. It is clear that we do not see much correspondence between the eigen-functions in this case – this motivates the need for features stronger than just the individual pixel colors. Features that encode local image gradients are good candidates as they provide a soft metric on the salient regions without needing a hard-thresholding step needed by edge and contour detectors. In this thesis, we propose to use SIFT [Lowe, 2004] descriptors computed densely on the image at a fixed spatial sampling $\delta$ (we use $\delta = 5$ pixels for experiments in this chapter).
To capture local image gradients at multiple scales, at each location we compute the SIFT descriptors at two different scales (size of the SIFT spatial bin) $s_1$ and $s_2$. The resulting feature vectors are concatenated to result in a 256-D feature vector $f_i(x)$ at each location $x$ in image $I_i$. Taking into account the spatial sampling of the features $\delta$, let $n_1$ and $n_2$ be the number of feature vectors obtained from images $I_1$ and $I_2$ respectively. Then the affinity matrices $W_1$, $W_2$ and $C$ are defined as follows:

$$
(W_i)_{x,y} = \exp \left( -\frac{\|f_i(x) - f_i(y)\|^2}{\sigma_f^2} \right) \tag{4.2}
$$

$$
(C)_{x,y} = \exp \left( -\frac{\|f_i(x) - f_j(y)\|^2}{\sigma_f^2} \right) \tag{4.3}
$$

where $f_i(x)$ and $f_i(y)$ are features at locations $x$ and $y$ in image $I_i$. We use the cosine-distance as the feature distance function $\|\cdot\|$ with tuning parameter $\sigma_f$ set to 1.0 in all our experiments. The scales $s_1$ and $s_2$ were set to 10 and 6 respectively for all the experiments. Note that unlike most image-domain spectral approaches in the literature, we do not use a spatial affinity term to reduce the influence of spatially separated pixels. In fact, supporting long range interactions is a key component of our approach as this allows us to obtain more distinctive profiles in the computed eigen-functions. With a spatial proximity term in the affinity matrix, we run the risk of artificially limiting the spatial extent of an eigen-function extrema and thus rendering the derived features less distinctive.

Given the joint graph affinity matrix $W$ from eqns-(4.1), (4.2) and (4.3), it is straightforward to compute the eigen-spectra. But before we do that, let us see if we can determine any correspondence information between image regions by extracting the spectra from each image graph separately. Fig. 4.4 shows the eigen-functions obtained by spectral analysis of the image graphs of the top and bottom row images.
independently. Even though the eigen-functions correctly represent the grouping of gradient information as is expected from our gradient features, one cannot infer useful correspondence information between image regions from the corresponding pair of eigen-functions directly.

Now, we can go back and look at the eigen-function pairs in Fig. 4.2. These were obtained as the eigen-vectors $u_1, \ldots, u_5$ corresponding to the smallest $K = 5$ eigen-values of the eigen-value decomposition $LU = \lambda U$. From each $n_1 + n_2$ dimensional eigen-vector $u_k$, we compute an eigen-function pair $(J^{(k)}_1, J^{(k)}_2)$ as follows. The first $n_1$ entries of $u_k$ are reshaped to the dimensions of $I_1$ by assigning its component values to the sampling locations where the features were extracted from and then interpolating the values in between. Similarly, the next $n_2$ entries of $u_k$ are reshaped to the dimensions of $I_2$ leading to the eigen-function $J^{(k)}_2$.

### 4.1.3 Characterization of Persistent Regions

As discussed before, the extrema of the eigen-function pairs $(J^{(2)}_1, J^{(2)}_2), \ldots, (J^{(5)}_1, J^{(5)}_2)$ represent persistent features that can serve well as means of finding correspondences across these difficult pairs of images. We want to characterize these extrema in terms of their location, their region of support as well as the variation of the eigen-energy in the vicinity of each extrema. Since the extrema can commonly exhibit elongated ridge-like shapes, an isotropic blob-detector would not work well. The continuous nature of the eigen-functions suggests that a water-shed like algorithm would serve as a good detector that might find both the location as well as the support region for these extrema. Therefore, we found the well known feature detection algorithm – the Maximally Stable Extremal Region (MSER) detector [Matas et al., 2004] – to be suitable for this purpose.
The intensity-based MSER detector is typically used to find affine-covariant regions in an image by looking for water-shed areas that remain stable as an image intensity threshold is varied. Each detected region is a set of connected pixels to which an ellipse is typically fit to represent the support region. To apply the MSER detector, we first normalize each eigen-function $J_{1}^{(k)}$ (and $J_{2}^{(k)}$) to a range of $[0, 255]$ by scale and offset correction. Then, we run intensity-based MSER along with ellipse fitting to detect stable affine regions. All the eigen-function figures in this chapter depict these regions as green or magenta ellipses corresponding to maxima and minima respectively.

To represent the eigen-energy variation around each detected MSER region, a number of different descriptors can be used. Through empirical evaluation, we have found the SIFT [Lowe, 2004] descriptor to work well. Each detected MSER ellipse is affine corrected to a circular region and a SIFT descriptor is computed for a region five times the ellipse size by computing gradients on the eigen-function. The large spatial extent of the SIFT window allows us to capture the eigen-energy profile more distinctly while still finding corresponding features between the eigen-function pairs. We will use the term JSPEC to refer to this feature which combines MSER ellipse keypoint with the eigen-space SIFT descriptor.

### 4.1.4 Eigen-function Feature Matching

The centroids of the MSER ellipses along with their associated SIFT descriptors can be treated as image features in a traditional sense. Therefore, we adopt a simple approach to matching these features by using the nearest-neighbor criterion coupled with the ratio-test [Lowe, 2004]. However, we match the descriptors from each pair of eigen-functions ($J_{1}^{(k)}, J_{2}^{(k)}$) independently i.e. for each descriptor in $J_{1}^{(k)}$, the nearest
Algorithm 4.1: JSPEC Algorithm

1. Compute features $f_1(x)$ and $f_2(x)$ at a spatial sampling $\delta$ for $I_1$ and $I_2$.
2. Compute affinity matrix $W$ using eqns-(4.1), (4.2) and (4.3).
3. Compute the $K$ smallest eigen-vectors $u_1, \ldots, u_K$ for $W$.
4. Extract eigen-function pairs $(J_1^{(k)}, J_2^{(k)})$ from each $u_k$.
5. Detect MSER features and compute SIFT descriptors for each $(J_1^{(k)}, J_2^{(k)})$.
6. Match features from each $(J_1^{(k)}, J_2^{(k)})$ by bi-directional SIFT matching.
7. Collect matches from all $K$ eigen-functions to get the final match set.

and second-nearest descriptors are searched only in $J_2^{(k)}$ and the association to the nearest descriptor is accepted only if its euclidean descriptor distance is less than $\tau$ times the distance to the second-nearest descriptor. To enforce a stronger match criterion, we perform matching from $J_1^{(k)}$ to $J_2^{(k)}$ and from $J_2^{(k)}$ to $J_1^{(k)}$ and keep the matches which are mutually consistent. This gives us a set of correspondences $C_k$ from the eigen-function pair $(J_1^{(k)}, J_2^{(k)})$. It should be noted that unlike traditional SIFT feature matching, our constraint on being able to match between individual eigen-function pairs results in a much stronger match criterion.

Algorithm. We present our method in a reproducible algorithmic form in Alg. 4.1.

4.2 Experiments

We evaluate our approach on the dataset of challenging image pairs from [Hauagge and Snavely, 2012]. This dataset contains 46 pairs of images exhibiting dramatic illumination, age and rendering style differences. Some image pairs are pre-registered with a homography to focus on appearance differences, while others exhibit both geometric and photometric variation. For each image pair, a manually extracted ground-truth homography $H_{12}$ is included with the dataset.

Hauagge and Snavely [Hauagge and Snavely, 2012] evaluated their local symmetry
features first, in terms of the detector repeatability and second, in terms of descriptor mean-average-precision performance. In our evaluation, we follow their methodology and evaluation metrics closely and provide a thorough comparison of our JSPEC features with their SYM-I and SYM-G features.

4.2.1 Detector Repeatability

To evaluate the repeatability of the eigen-space MSER features for a given image pair, we consider all the detections before the SIFT matching step. We collect all the features from across all eigen-functions into two sets of keypoints $K_1$ and $K_2$ for images $I_1$ and $I_2$ respectively. Each keypoint has a centroid and an ellipse associated with it. Therefore, we can directly apply the repeatability metric from [Mikolajczyk et al., 2005] which we briefly review next. Each keypoint $k_1 \in K_1$ is warped into $I_2$’s coordinate frame using the ground-truth homography $H_{12}$ and its (warped) support region is compared with the support region of each keypoint $k_2 \in K_2$ to obtain an overlap score. If the overlap score is more than 0.6 (i.e. less than 40% overlap error), then we count the associated keypoint-pair as a correct detection. The ratio of the correct detections to the smaller of the number of keypoints in either image is used as the measure of detector “repeatability”. To be invariant to absolute keypoint scales, the keypoints in $I_1$ are rescaled by a factor $s$ to a fixed area $A$ before applying the homography. The same scale $s$ is applied to the keypoints in $I_2$ before determining the overlap score.

Hauagge and Snavely [Hauagge and Snavely, 2012] computed the repeatability scores of their features by considering subsets of top-100 and top-200 detections ordered by either feature scale or score. This was done to avoid a bias when comparing to detectors that produce a large number of keypoints. Our MSER detector does
Table 4.1: Detector repeatability compared with [Hauagge and Snavely, 2012].

<table>
<thead>
<tr>
<th>Detector</th>
<th>Scale</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100</td>
<td>200</td>
</tr>
<tr>
<td>MSER</td>
<td>0.087</td>
<td>0.103</td>
</tr>
<tr>
<td>SIFT (DoG)</td>
<td>0.144</td>
<td>0.153</td>
</tr>
<tr>
<td>SYM-I</td>
<td>0.135</td>
<td>0.184</td>
</tr>
<tr>
<td>SYM-G</td>
<td>0.173</td>
<td>0.228</td>
</tr>
<tr>
<td>JSPEC</td>
<td>0.287</td>
<td>0.292</td>
</tr>
</tbody>
</table>

Table 4.2: Descriptor mean average precision (mAP) evaluation and comparison with [Hauagge and Snavely, 2012].

<table>
<thead>
<tr>
<th>Description</th>
<th>GRID</th>
<th>SIFT</th>
<th>SYM-I</th>
<th>SYM-G</th>
<th>JSPEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Sim.</td>
<td>0.29</td>
<td>0.14</td>
<td>0.12</td>
<td>0.16</td>
<td></td>
</tr>
<tr>
<td>SIFT</td>
<td>0.49</td>
<td>0.21</td>
<td>0.28</td>
<td>0.25</td>
<td>0.61</td>
</tr>
<tr>
<td>SYMD</td>
<td>0.41</td>
<td>0.22</td>
<td>0.20</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>SIFT-SYMD</td>
<td>0.58</td>
<td>0.28</td>
<td>0.35</td>
<td>0.36</td>
<td></td>
</tr>
</tbody>
</table>

not output a detection score and so we only present repeatability numbers based on ordering by scale. These are shown in Table-4.1 where we have also reproduced numbers from [Hauagge and Snavely, 2012] for ease of comparison. We observe that our JSPEC features achieve slightly better repeatability than what SYM-G achieved using the top scoring 200 detections.

### 4.2.2 Descriptor Evaluation

We evaluate the discriminative power of our eigen-space SIFT descriptors in a manner similar to [Hauagge and Snavely, 2012]. Here again, we consider the descriptors associated with all the features (i.e. before the matching step) collected together from all the eigen-functions. Then, we match these descriptors using the standard ratio test [Lowe, 2004] on the top two nearest neighbor distances. For a given choice of the ratio threshold, we get a set of candidate correspondences which are evaluated
Figure 4.5: Precision-Recall curves comparing performance of the spectral approach (JSPEC) with the features evaluated in [Hauagge and Snavely, 2012]. Each column shows plots for the image pair in the top row. For each image pair, the JSPEC curve is repeated in the four rows to show comparison with the four different detectors in [Hauagge and Snavely, 2012].

with the ellipse overlap criterion of [Mikolajczyk et al., 2005] using the ground-truth homography $H_{12}$ to compute a point on the precision-recall curve. By varying the ratio-test threshold, we can trace the full precision-recall curve.

Fig. 4.5 compares the performance of our JSPEC features against each of the features studied by [Hauagge and Snavely, 2012]$^2$. Note that the JSPEC plots across each column are exactly the same since we do not vary the detector. We have plotted them four times to allow comparison with the individual plots from [Hauagge and Snavely, 2012].

$^2$The precision-recall data plotted here was obtained from the authors directly.
The first four image pairs show a substantial improvement in performance over other competing methods. The first row of plots (“Grid”) represents a synthetic detector experiment in [Hauagge and Snavely, 2012] where locations on a uniform grid in \( \mathcal{I}_1 \) are chosen as \( K_1 \) and these locations warped by \( H_{12} \) into \( \mathcal{I}_2 \) are chosen as \( K_2 \). This is meant to test how well the descriptor matches appearance of perfect geometrically matching locations. Even though we do not use the grid-detector, a comparison of the JSPEC PR-curves with other curves in the “Grid” row clearly indicate that SIFT features computed on the eigen-functions match better across the extreme day-night appearance changes. The graffiti image pair (fifth column) shows that we perform similar to the SYMD descriptor on SIFT features but, as expected, worse than the SIFT detector-descriptor pair. Finally, the Taj example (last column) shows a failure case where our method fails because large parts of the scene have changed completely. In this case, a combination with SIFT features is likely to give better performance. Also note that we have not applied either the bi-directional matching criterion or the “match only within each eigen-function pair” criterion to obtain these precision-recall curves for a fair comparison with other methods (which also do not apply the bi-directional constraint). The performance is expected to be higher with these additional criteria.

Table-4.2 compares our mean average precision with [Hauagge and Snavely, 2012] on the entire dataset. We achieve an overall mAP of 0.61 which is higher than the synthetic grid-detector (combined with SIFT-SYMD descriptor) based mAP of 0.58 achieved by [Hauagge and Snavely, 2012].
Figure 4.6: Comparison of SIFT matches and JSPEC matches on a day-night image pair. We also show the affinity matrix for the joint graph and the eigen-function pairs from which the JSPEC feature matches were obtained. In (e), we show eigen-function pairs \((J_1^{(2)}, J_2^{(2)}), \ldots, (J_1^{(5)}, J_2^{(5)})\) along with the MSER feature-ellipses that have been matched using the SIFT-bidirectional matching criterion \((\tau = 0.8)\).

### 4.2.3 Qualitative Results

In Figs. 4.6-4.10, we show some qualitative outputs of our algorithm. The matches overlaid on the images are the final matches obtained after bi-directional JSPEC matching at a ratio-threshold of 0.8. Fig. 4.6 shows a day-night image pair and compares the SIFT and JSPEC matches obtained. It is clear that the SIFT matches are sparse, poorly distributed and mismatched a number of times. The JSPEC matches
Figure 4.7: Characterization of persistent regions detected by JSPEC matching. JSPEC features capture more elongated structures compared to SIFT features.

Figure 4.8: Painting to image matching. The painting images (top-row) have been taken from the dataset in [Shrivastava et al., 2011].

are more evenly distributed and capture regions at a greater range of scales including more elongated structures. To ease visualization of JSPEC features, Fig. 4.7 shows three image pairs with the JSPEC matches without the connecting lines. As compared to SIFT features, the JSPEC ellipses capture more elongated edge-like structures with fewer features on corner-like regions. In addition, there are blob-like regions with repetitive texture that are also captured by the JSPEC features.
Figure 4.9: The first column shows an image-pair from the dataset in [Haugage and Snavely, 2012] along with the correspondences assembled from the individual eigen-functions. Second through fifth columns show eigen-function pairs $(J^{(2)}_1, J^{(2)}_2), \ldots, (J^{(5)}_1, J^{(5)}_2)$ along with the MSER feature-ellipses that have been matched using the SIFT-bidirectional matching criterion ($\tau = 0.8$).

Fig. 4.8 shows two examples of paintings from the dataset shared by [Shrivastava et al., 2011] in the top row. We downloaded similar looking images from the web and tested our algorithm on this difficult painting to image matching task. The examples show the quality of our matches. Fig. 4.9 shows three more different kinds of examples with the correspondences detected in each of the four eigen-function pairs collected together and overlaid in the first column. In the first row, we have a difficult day-night pair where we find valid matches despite the poor contrast. The second row shows matching between a historic picture and a drawing. The third row shows the standard graffiti image pair with uncorrected perspective distortion that
Figure 4.10: The first column shows an image-pair from the dataset in [Szeliski, 2005] along with the correspondences assembled from the individual eigen-functions. Second through fifth columns show eigen-function pairs \((J_1^{(2)}, J_2^{(2)})\), \(\ldots, (J_1^{(5)}, J_2^{(5)})\) along with the MSER feature-ellipses that have been matched using the SIFT-bidirectional matching criterion \((\tau = 0.8)\).

our method can easily cope with. Note the accuracy of the feature detections on the original images. Fig. 4.10 shows two examples from the [Szeliski, 2005] dataset where we show JSPEC matches between two images from a translated camera. This shows that the method is robust to camera pose changes and does not assume that the field-of-views of the two cameras are fully overlapping.

### 4.3 Robustness to Geometric Transformations

In this section, we present qualitative and quantitative results from a systematic evaluation of the JSPEC algorithm across a range of image transformations. In particular, we evaluate the repeatability of the detected features and the mAP of
the matched descriptors on the complete [Hauagge and Snavely, 2012] dataset by applying synthetic image warps to the second image for each image pair from this dataset. We include qualitative results for image pair riga (shown in Fig. 4.6(a)) to showcase the impact of these transformations on individual eigen-function pairs.

4.3.1 Rotation

We apply an in-plane image rotation (about the image center) of magnitudes $\theta \in \{-30, -15, 0, 15, 30\}$ to image $I_2$ for each image pair ($I_1, I_2$), and run the JSPEC matching and evaluation pipeline as described before. Fig. 4.11 shows the influence of this warp on the joint image graph eigen-functions for one example image pair. The local DSIFT features employed in our method are not rotation invariant and are thus not expected to match well for these large rotations. Thus, the quantitative plot shows a significant drop in the matching performance even though the detector repeatability still remains high. This can be seen from the qualitative examples where we find that the eigen-functions still capture the similarity signature of corresponding regions quite well.

4.3.2 Scale

We apply an isotropic scaling transformation to image $I_2$ with scale magnitudes $s \in \{0.75, 0.90, 1.0, 1.10, 1.25\}$ and evaluate the JSPEC algorithm. Fig. 4.12 shows the qualitative and quantitative results. We note that the method is more robust to scale changes than it is to image rotations even though the DSIFT features are not scale-invariant. This is due to the explicit sampling of DSIFT descriptors at two different scales in our approach. Also note that the method performs more poorly when $I_2$ is zoomed-out than when it is zoomed-in. This is due to the artificial black
4.3.3 Perspective

We apply a horizontal perspective warp to image $I_2$ about the image center:

$$H = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \tau_x & 0 & 1 \end{pmatrix} \quad (4.4)$$

with $\tau_x \in \{-1.0, -0.5, 0.0, 0.5, 1.0\}$. Fig. 4.13 shows the robustness of the JSPEC approach to perspective warps between image pairs. The eigen-structure of the image-pair remains largely intact and the mAP drop is less significant than seen in the case of image rotation and scale change.

4.4 Impact of Algorithm Parameters

In this section, we analyze the impact of the various algorithm parameters to the nature of the eigen-functions and the matchability of the derived JSPEC features. In particular, we evaluate the repeatability of the detected features and the mAP of the matched descriptors on the [Hauagge and Snavely, 2012] dataset by individually varying each parameter through a range of discrete values and re-running the full algorithm and evaluation pipeline. We include qualitative results for image pair *riga* (shown in Fig. 4.6(a)) to showcase the impact of these parameter variations on individual eigen-function pairs.
<table>
<thead>
<tr>
<th>Rotation Angle, $\theta$</th>
<th>Large Warp</th>
<th>Original Dataset</th>
<th>Large Warp</th>
</tr>
</thead>
<tbody>
<tr>
<td>$-30$</td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
</tr>
<tr>
<td>$-15$</td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
</tr>
<tr>
<td>$0$</td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
</tr>
<tr>
<td>$15$</td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
</tr>
<tr>
<td>$30$</td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
<td><img src="#" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 4.11: Performance evaluation of JSPEC features with varying in-plane rotation angle $\theta$. 
Figure 4.12: Performance evaluation of JSPEC features with varying scale parameter $s$. 
<table>
<thead>
<tr>
<th>$\tau_x\rightarrow$</th>
<th>Large Warp</th>
<th>Original Dataset</th>
<th>Large Warp</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1.0</td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
<tr>
<td>-0.5</td>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
</tr>
<tr>
<td>0.0</td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
<td><img src="image9.png" alt="Image" /></td>
</tr>
<tr>
<td>0.5</td>
<td><img src="image10.png" alt="Image" /></td>
<td><img src="image11.png" alt="Image" /></td>
<td><img src="image12.png" alt="Image" /></td>
</tr>
<tr>
<td>1.0</td>
<td><img src="image13.png" alt="Image" /></td>
<td><img src="image14.png" alt="Image" /></td>
<td><img src="image15.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 4.13: Performance evaluation of JSPEC features with varying horizontal perspective parameter $\tau_x$. 

Mean Average Precision (mAP) vs. Repeatability vs. Horiz. Perspective Parameter, $\tau_x$. 

77
4.4.1 Intra-vs-Inter Affinity

Our approach employs affinities from a complete graph that is composed of all the sampled pixel locations from both images. It is instructive to see the relative impact of affinities obtained from each graph (intra affinities) and the affinities between the two images (inter affinities). For this experiment, we re-write the full graph affinity matrix $W$ as follows:

$$W = \begin{pmatrix} \alpha W_1 & (1 - \alpha)C \\ (1 - \alpha)C^T & \alpha W_2 \end{pmatrix}$$ (4.5)

where $\alpha \in [0, 1]$ determines the relative influence of the intra graph affinities $W_i$ and the inter graph affinity matrix $C$. Fig. 4.14 shows the impact of varying $\alpha$ from $\alpha = 0$ corresponding to using only the inter-image affinity matrix $C$ to $\alpha = 1$ corresponding to using only the intra-image affinity matrices $W_i$. The case $\alpha = 0.5$ in the center column corresponds to equal weights for the two affinities which is the algorithm that has been presented and evaluated in this chapter so far. We see that, as expected, as the influence of the inter-image affinities is reduced ($\alpha \rightarrow 1.0$), the eigen-functions also exhibit reduced matchable regions thus leading to a drop in the mAP and repeatability numbers shown in the plot. However, surprisingly, the quantitative performance drops a very small amount even when the intra-image term is reduced to zero! The qualitative results show that removing the intra-image affinities do lead to noisier eigen-functions – particularly in the higher-orders. However, the JSPEC algorithm is robust to this noise and can maintain a higher mAP. The reason that the eigen-structure remains intact even without the intra-image terms can be understood as follows. Even when only the $C$ matrix is supplied, each pixel in $I_1$ gets connected to every pixel in $I_2$ and vice-versa.
Therefore, if two pixels $p$ and $q$ in $I_1$ get connected to a particular pixel $r$ in $I_2$ by high-affinity edges, then this implicitly encodes a high-affinity edge between pixels $p$ and $q$ by transitivity via pixel $r$. Since we allow all possible connections, the overall joint graph still encodes long range interactions within each graph via the other graph. Thus, explicitly adding the intra-affinity terms only help reduce noise by a small amount.

### 4.4.2 Spatial Affinity

Even though the above analysis indicates that the intra-image affinities have little influence on the joint eigen-functions, a spatial term within the intra-image affinity functions $W_i$ can still impact the joint image graph. To model this, we modify the intra-image affinity function $W_i$ as follows:

$$
(W_i)_{x,y} = \exp \left( - \frac{\| f_i(x) - f_i(y) \|^2}{\sigma_f^2} \right) \exp \left( - \frac{\| x - y \|^2}{\sigma_s^2} \right)
$$

(4.6)

where the second exponential penalizes the separation between pixels locations $x$ and $y$. Fig. 4.15 shows the joint image graph eigen-functions when the parameter $\sigma_s$ is varied from $10^2$ to $10^6$. At the lowest value ($\sigma_s = 10^2$), the spatial terms goes to zero which removes the contribution of the intra-image affinity term $W_i$ and thus corresponds to the case we saw in the intra-vs-inter affinity analysis. At the highest value ($\sigma_s = 10^6$), the exponential term goes to one leading to a fully connected graph which is the case we have looked at throughout the chapter. In between these two extremes, the spatial affinity term only allows a local neighborhood of pixels to contribute. The middle three columns show that this does have a profound negative impact on the eigen-structure and correspondingly on the quantitative performance.
This can be understood by considering that the joint graph structure has been artificially modified here to include only specific local edges (within each $W_i$) while still allowing all many-to-many connections in the $C$ matrix. This artificial distortion leads to a loss in the fine-scale correspondence leaving behind the coarse bi-level correspondence visible in the center column.

4.4.3 Feature Affinity

We also evaluate the influence of the feature affinity parameter $\sigma_f$ without the spatial affinity term. Fig. 4.16 compares the eigen-structure at three different values of $\sigma_f$. At $\sigma_f = 0.1$, we are very strict about the feature distance and hence only DSIFT descriptors that are very similar show up as connected segments in the eigen-structure. As $\sigma_f$ is increased, we relax this constraint thereby leading to a richer eigen-structure which improves the achievable mAP. Note that since the feature affinity parameter $\sigma_f$ impacts both the intra ($W_i$) and the inter ($C$) matrices in a similar manner, the eigen-structure is not completely distorted. This is unlike the spatial affinity parameter $\sigma_s$ which only impacts the intra terms and hence artificially distorts the eigen-structure.

4.4.4 Spatial Sampling

The spatial sampling parameter $\delta$ in our formulation governs the sampling of the grid where the DSIFT descriptors (and correspondingly the graph vertices) are evaluated (positioned). Therefore, given a fixed image resolution it determines both the size of the graph as well as the accuracy of the computed eigen-structure. In Fig. 4.17, we evaluate five different values for the sampling parameter $\delta$. At the finest sampling of $\delta = 5$ (the default setting in the rest of the chapter), we capture the eigen-structure
at a fine-level but this results in a higher computation cost because of the higher number of DSIFT features computed as well as the larger graph size for the eigen-solver. At $\delta = 13$, the coarser sampling reduces the computation time significantly while the performance is only marginally impacted. This shows that the sampling parameter can be used as an effective knob to trade-off between computation cost and algorithm accuracy.

### 4.5 Related Work

Not many approaches exist that can handle the discrepancy between two images at the level that we address in this chapter. [Shechtman and Irani, 2007b] proposed an approach for matching disparate images using patterns of local self-similarity encoded into a shape-context like descriptor. However, for the kind of disparate images we consider, the local self-similarity pattern itself can be significantly different between corresponding points in the image pair. [Shrivastava et al., 2011] recently proposed an approach for cross-domain image matching using data-driven learning techniques. Using a linear classifier, they learn the relative importance of different features (specifically, components of the global image HoG descriptor in the paper) for a given query image and then use the weight vector to define a matching score. In contrast, our focus is on extracting local features that are persistent between a pair of images instead of deriving a global image descriptor that can be used for retrieval. Recently, [Hauagge and Snavely, 2012] have focused on the task of matching such images by defining “local-symmetry” features which capture various symmetries like bilateral, rotational etc. at the local level. This approach addresses matching rather than retrieval and the discrepancy level is similar to the level our approach handles, hence, we decided to use the dataset [Hauagge and Snavely, 2012] as our main test
<table>
<thead>
<tr>
<th>$\alpha \rightarrow$</th>
<th>No intra-image $W_i$</th>
<th>No inter-image $C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td><img src="image1" alt="Image" /></td>
<td><img src="image2" alt="Image" /></td>
</tr>
<tr>
<td>0.25</td>
<td><img src="image3" alt="Image" /></td>
<td><img src="image4" alt="Image" /></td>
</tr>
<tr>
<td>0.50</td>
<td><img src="image5" alt="Image" /></td>
<td><img src="image6" alt="Image" /></td>
</tr>
<tr>
<td>0.75</td>
<td><img src="image7" alt="Image" /></td>
<td><img src="image8" alt="Image" /></td>
</tr>
<tr>
<td>1.00</td>
<td><img src="image9" alt="Image" /></td>
<td><img src="image10" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 4.14: Performance evaluation of JSPEC features with varying intra-image similarity weight $\alpha$. 

82
Figure 4.15: Performance evaluation of JSPEC features with varying spatial affinity parameter $\sigma_s$ for the intra-image similarity.
Figure 4.16: Impact of variation of feature affinity parameter $\sigma_f$ on the performance of JSPEC features.
Figure 4.17: Performance evaluation of JSPEC features with varying spatial sampling parameter $\delta$. Also shown is the impact of varying this parameter on the computation time.
Our methodology is most related to the works of [Yu et al., 2002] and [Toshev et al., 2007]. The spectral analysis of the joint matrix between two images appeared first in [Yu et al., 2002] where the affinity matrices of object model patches and the input image are combined with a non-diagonal matrix associating object patches and image pixels. [Toshev et al., 2007] proposed an approach to determine co-salient regions between two images using a spectral technique on the joint image graph constructed from the images. Their joint image graph was constructed with all the pixels in the two images by defining separate affinity functions for intra and inter image terms. The intra image affinity was defined using the intervening contour cue while the inter image term was based entirely on the initial set of feature correspondences between the images. In this thesis, we also use a joint image graph but we differ in (i) using no proximity information in the affinity matrix, (ii) using SIFT descriptor information instead of intensity differences, and (iii) defining both the intra and inter image terms densely using all the image pixels with a uniform affinity function that successfully captures the persistent regions shared between the images. The density of the inter-image term allows us to apply spectral decomposition directly instead of requiring us to use the subspace technique in [Toshev et al., 2007]. Thereafter, we show how we can use the extracted eigen-functions to construct features that are invariant to the large appearance changes between the input images.

Our method is also related to an algorithm for associating points from two images proposed by [Scott and Longuet-Higgins, 1991]. Much like the feature weight matrix \( C \) in our method, this algorithm used a cross-image affinity matrix constructed from the Gaussian weighted spatial distances between the feature points and then proposed an approach to maximize the inner product between this matrix and an assignment matrix by computing the Singular Value Decomposition (SVD) of the
distance matrix. The assignment matrix revealed correspondence information between the feature points. [Scott and Longuet-Higgins, 1991] also showed that their method recovers feature correspondences resulting from affine transformations in the image space. In our approach, the feature weight matrix $C$ does not employ any spatial coordinates, and is not seeded from repeatable feature locations. Thus, our method does not aim to directly minimize a spatial affine deformation cost, nor does it aim to find an assignment matrix between the feature locations in the two images directly. Instead, we are interested in the spatial arrangement of the eigen-functions as a representation for the images from which matchable features can be derived.

[Glasner et al., 2011] proposed a shape-based approach to jointly segment multiple closely-related images by combining contour and region information. They show examples of image pairs with illumination differences where their joint segmentation approach achieves better co-clustering than what is possible by using intra-image constraints alone. They start from a super-pixel segmentation of the images and then use contour-based constraints to drive their intra-image affinities. The inter-image constraints are derived from a comparison of HoG-like features only on contour segments. Our approach also uses gradient-based descriptors to enforce inter-image constraints – however, we do so densely between every pair of pixels in the input images. Thus, we do not need any prior segmentation and we are not prone to errors due to misdetection of contours, particularly since contour detection would be very challenging for the kind of disparate images we focus on. Contours of the soft version of the eigenvectors of a single image affinity matrix computed following the Normalized Cuts criterion have also been used in [Arbelaez et al., 2011] to include global relationships into the probabilistic boundary feature vector.

Laplacian representation created from a dense sampling of image pixels has recently been employed as a structure descriptor in [Eynard et al., 2014]. The eigen
structure of the Laplacian is then employed to derive structure-preserving color transformations between image pairs with applications like color-to-gray conversion, gamut mapping, multispectral image fusion etc.

4.6 Conclusion

Image matching across different illumination conditions and capture times has been addressed in the past by comparing descriptors of local neighborhoods or employing discriminative learning of local patches. In this chapter we introduced global image information into the matching process by computing the spectrum of the graph of all pixels in both images associated only by the similarity of their neighborhoods. Significantly, the eigen-functions of this joint graph exhibit persistent regions across disparate images which can be captured with the MSER characteristic point detector and represented with the SIFT descriptor in the resulting stable regions. Such characteristic points exhibit surprisingly high repeatability and local similarity.
Chapter 5

Geometric Urban Geo-Localization

In the facade matching and JSPEC matching approaches, we focused on novel appearance-level features ($S^4$ features) and novel matching techniques derived from local appearance features (joint spectral representation based on dense SIFT features) under the assumption of limited geometric variation (rectified facade planes or small transformations). In this chapter, we move further on the appearance-geometry spectrum to explore how views as dissimilar as an overhead orthographic elevation map and a street-level image can be matched. We study the challenging problem of geo-locating a street-level image using only the corners and roof-line edges of the buildings visible in the image and matching them geometrically to a database of 3D corners and direction vectors extracted from an elevation map. Since the appearance constraints in this case are limited to information from building edges, corners and other static scene structure, matching techniques to handle this problem have to rely heavily on the 3D scene geometry.

Recent work on geo-localization using a model database has relied largely on rendering-based techniques [Ramalingam et al., 2009, Matei et al., 2013, Baatz et al., 2012]. Typically, these techniques render the 3D model on a uniform grid in the
ground-plane, compute features for each rendering and then match these against the query features to retrieve candidate locations. While these techniques have proven efficient and effective for geolocating in a mountainous terrain [Baatz et al., 2012], their adaptation to urban environments has not had the same level of success. The reason for this is the computational overhead of rendering the building models at a fine enough resolution such that the rendering can closely match the query image.

In this chapter, we take a different approach to the urban geolocalization problem. Instead of pre-rendering the model, we extract purely geometric entities from the model and the query image and then use them directly to solve for the camera pose. To handle this combinatorial problem in a tractable manner, we propose a novel framework for correspondenceless pose-estimation in a 3D-to-2D setup that employs a minimal solver without suffering from the combinatorial explosion typical in such setups. In particular, we employ the two-point method for estimating the pose of a calibrated camera with known vertical direction in the image. Without a set of extracted 3D-to-2D correspondences, employing even a 2-point algorithm for pose estimation is prohibitively expensive. In a geolocation setup, assuming we have identified \( m \) building corners in the query image and have \( n \) 3D-building corner points in a database, the cost of testing all minimal configurations is \( O(m^2n^2) \). In addition, the number of correct correspondences is at most \( m \) which makes any direct voting-based method infeasible.

Therefore, we propose a stratified approach that uses a lower number of constraints to compute a partial solution which is then used to generate putative correspondences on which the 2-point method can be applied. Specifically, first we use a single point and line direction (ray) correspondence to solve for the camera pose partially. In this partial solution, the camera rotation (pan) is recovered and the translation is expressed as a locus along a 3D-line segment. We show that the
projection of 3D points using this parametric camera pose generates line-segments in the image. The perpendicular distance of the image points from these segments can be used to identify putative correspondence of the 2D-points with the 3D-points corresponding to these line segments. This is a novel insight that uses a partial solution to establish putative correspondences without using any appearance information. The standard 2-point algorithm can now be applied in a RANSAC setting to this putative set to generate a hypothesis camera pose. In this framework, we test at most $O(m^2n)$ minimal configurations which is a substantial cost saving in
typical problem instances where $n >> m$. In the geolocalization setting we address here, $n$ corresponds to the total number of building corners in a database which is substantially larger than the number of building corners detected in a single query image ($m$).

Our work distinguishes from the state of the art in the following contributions:

1. A novel formulation for upright pose estimation using a point-line pair.

2. Degeneracy conditions for the point-line problem.


4. An application of this framework to geo-localization without appearance correspondences.

5. The fact that we avoid any visibility information or rendering.

The chapter is organized as follows. In section 5.1, we formulate the geometric problem for estimating the pose of an upright calibrated camera in terms of minimal set of points and direction correspondences. Section 5.2 presents details of our proposed algorithm for geo-localization including procedures for detecting query and database features. Section 5.3 presents experimental results.
5.1 With Correspondence Information

5.1.1 Preliminaries

Let \( \bar{p} = (\bar{u}, \bar{v}, 1)^\top \) be the (homogeneous) image projection of a 3D point \( P = (X, Y, Z)^\top \). Then the projection equation is given by:

\[
K(RP + t) \cong \bar{p} \tag{5.1}
\]

where \( K \) is the camera internal matrix and \( R \) and \( t = (t_x, t_y, t_z)^\top \) are the camera rotation and translation respectively relative to the world. Defining \( p = K^{-1}\bar{p} \) and converting the projective equivalence to equality by taking a cross product, we get:

\[
[p]_x (RP + t) = 0 \tag{5.2}
\]

where \( p = (u, v, 1)^\top \) are the normalized coordinates of the image point and \([p]_x \) is the skew-symmetric matrix representing cross-product with the vector \( p \).

If the Y-axis of the camera is aligned with the Y-axis (and the gravity vector in the geo-localization setup) of the world coordinate system, the rotation matrix \( R \) can be simplified to:

\[
R = \begin{pmatrix}
\frac{1-q^2}{1+q^2} & 0 & \frac{-2q}{1+q^2} \\
\frac{2q}{1+q^2} & 0 & \frac{1-q^2}{1+q^2} \\
0 & 1 & 0
\end{pmatrix} \tag{5.3}
\]

where \( q = \tan(\theta/2) \) and \( \theta \) is the unknown camera pan angle. Substituting in equation
and denoting $q = (q^2, q, 1)^\top$, we get:

\begin{align*}
    f(u, P)^\top q &= (u\bar{t}_z - \bar{t}_x) \\  
    g(v, P)^\top q &= (\bar{t}_y - v\bar{t}_z)
\end{align*}

(5.4)

(5.5)

where,

\begin{align*}
    f(u, P) &= \begin{pmatrix} uZ - X \\ -2(uX + Z) \\ X - uZ \end{pmatrix}, &
    g(v, P) &= \begin{pmatrix} -Y - vZ \\ 2vX \\ vZ - Y \end{pmatrix}
\end{align*}

(5.6)

and $\bar{t} = (\bar{t}_x, \bar{t}_y, \bar{t}_z)^\top = (1 + q^2)t$.

### 5.1.2 Correspondence of two points

**Proposition 1.** Given correspondence of any two points, not in a degenerate configuration, there are two possible solutions for the pose of an upright camera[\textit{Kukelova et al., 2011}].

**Proof.** Let $P_1 = (X_1, Y_1, Z_1)^\top$ and $P_2 = (X_2, Y_2, Z_2)^\top$ be the two 3D points and $(u_1, v_1)^\top$ and $(u_2, v_2)^\top$ be the corresponding image points. Then, from equation (5.4), we get the following pair of equations:

\begin{align*}
    f(u_1, P_1)^\top q &= (u_1\bar{t}_z - \bar{t}_x) \\  
    f(u_2, P_2)^\top q &= (u_2\bar{t}_z - \bar{t}_x)
\end{align*}

(5.7)

(5.8)
Eliminating $t_x$ between equations (5.8) and (5.7), we get:

$$(f(u_1, P_1) - f(u_2, P_2))^\top q = (u_1 - u_2)\bar{t}_x$$  \hspace{1cm} (5.9)

Similarly, from equation (5.5), we get:

$$g(v_1, P_1)^\top q = (\bar{t}_y - v_1\bar{t}_z)$$  \hspace{1cm} (5.10)
$$g(v_2, P_2)^\top q = (\bar{t}_y - v_2\bar{t}_z)$$  \hspace{1cm} (5.11)

which, after eliminating $\bar{t}_y$ gives:

$$(g(v_1, P_1) - g(v_2, P_2))^\top q = (v_2 - v_1)\bar{t}_z$$  \hspace{1cm} (5.12)

Eliminating $\bar{t}_z$ from equations (5.9) and (5.12), we get a quadratic in $q$ which can be solved to get two solutions for the camera rotation parameter $q$. Substituting this value in either equation (5.9) or (5.12), we can solve for $\bar{t}_z$ and then for $\bar{t}_x$ and $\bar{t}_y$ from the remaining equations leading to two solutions for the camera pose.  \hspace{1cm}  \Box

**Degeneracies.** The two-point method is degenerate if and only if either a) the two points lie in the XZ-plane passing through the camera center, or b) the two points lie on the same vertical line.

**Proof.** Assume two points $A$ and $B$ at known positions in a 3D world frame. It is well known that the locus of camera centers $O$ which view these points at the same relative non-zero angle (relative bearing) is a toroid. This is easy to see since in the 2D case the locus is a circle which when rotated about axis $AB$ becomes a toroid. It is also known that with one more point (P3P), three toroids are created which in the general case intersect at 8 points.
If the third point is at infinity then we have the case of a known direction $g$ w.r.t. the world (like gravity). Assume that the angle between the known direction and the ray to $A$ is $\alpha$. Then the locus of camera centers that see $A$ under angle $\alpha$ w.r.t. the known direction is a cone $((X - A)^T g)^2 = (X - A)^2 \cos^2 \alpha$. A second cone is created for the constraint that point $B$ is seen under angle $\beta$ w.r.t. the known direction $g$. In the general case, the toroid intersects with the two cones in two points.

This is not the case when the cone degenerates to a plane (case 1) or when the intersections of each cone with the toroid are identical (case 2), yielding in both cases a one-parameter family of solutions (Fig. 5.2).

Case 1: When the angle $\alpha$ is $90^\circ$, the cone degenerates into a plane. This is not a problem when this happens with one of the points since the number of intersections of a toroid, a cone, and a plane, is finite. In the case of gravity, this is the case when one of the two points is in the middle row of the image. However, if this happens to both points $\alpha = \beta = 90^\circ$ then the two planes coincide and the intersection with the toroid is a circle. This is the only case when the intersection of the two “cones” is a two-parameter solution.

Case 2: If the line $AB$ happens to be parallel to the direction of reference, then
the two cones intersect at a circle. This circle is contained in the toroid because at each position of this circle the relative bearing $A\hat{O}B$ is constant and equal to $|\beta - \alpha|$. This is not the case with points in the general intersection between two cones because in this case the angles $\alpha$ and $\beta$ are spanned in different planes. This is also the only case (except case 1) where the intersection is a curve because this is the only case when the intersection between the cones is a circle. In any other case the intersection between two cones is a conic section and the only conic section satisfying the constancy of relative bearing is the circle.

**Proposition 2.** Given correspondence of two points at the same elevation from the ground, the camera location can be determined without the knowledge of the elevation of the points up to an unknown $t_y$ i.e. the camera elevation will remain undetermined.

**Proof.** Let the unknown elevation be $Y = Y_1 = Y_2$. Then, from equation (5.6), $g(v_1, P_1) - g(v_2, P_2)$ is independent of $Y$. Since equation (5.9) is also independent of $Y$, we can now solve for $q$ and then $\bar{t}_z$ and $\bar{t}_x$ using the same steps as in proposition (1). From equation (5.10) (or (5.11)), $t_y$ can be expressed as follows:

$$t_y = v_1t_z + \frac{q^2(-v_1Z_1) + q(2v_1X_1) + v_1Z_1}{1 + q^2} - Y$$  \hspace{1cm} (5.13)

This places the camera vertical translation $t_y$ at a fixed offset relative to the unknown elevation $Y$ of the 3D points.

5.1.3 Correspondence of one line

**Proposition 3.** For a camera with its vertical axis aligned with the world vertical, a single line correspondence is sufficient to determine two possible solutions for the unknown camera rotation (pan angle).
Proof. Let $L = (L_x, L_y, L_z)^\top$ be the direction vector of a 3D line and $n = (n_1, n_2, n_3)^\top$ be the homogeneous representation of the corresponding line observed in the image. Then, the following result expresses the relationship between $n$, $L$ and the rotation matrix $R$ of the camera in the world coordinate system:

$$n^\top RL = 0$$ (5.14)

For an upright camera, the rotation matrix $R$ can be set from equation (5.3) leading to the following quadratic equation in the unknown camera rotation $q$:

$$l^\top q = 0$$ (5.15)

where,

$$l = (-L_x n_1 + L_y n_2 - L_z n_3, 2L_x n_3 - 2L_z n_1, L^\top n)^\top$$ (5.16)

The above equation leads to two different solutions for the unknown $q$.  

**Degeneracies.** For a camera with its vertical axis aligned with the world vertical, estimation of the camera rotation (pan) using a line correspondence is degenerate if and only if either a) the line is vertical in the world or b) the line is in the XZ-plane passing through the camera center.

Proof. The equation (5.15) is degenerate when the quadratic coefficients $l$ are all zero. Setting $l = 0$ leads to the following four conditions: a) $L = 0$, b) $n = 0$, c) $\{n_2 = 0, L_z = 0, L_x = 0\}$, and d) $\{n_1 = 0, n_3 = 0, L_y = 0\}$.

In (c), $n_2 = 0 \Rightarrow n = (\alpha, 0, \beta)$, which implies that the line is vertical in the image. Also, $L_z = L_x = 0$ implies that it is vertical in the world as well.

98
In (d), \( n_1 = n_3 = 0 \) implies the image line \( v = 0 \) which is a horizontal line passing through the image center. This implies that \( Y = -t_y = c_y \) from the camera projection equation. Coupled with the condition \( L_y = 0 \), this is a line in the XZ-plane passing through the camera center.

### 5.1.4 Correspondence of a point and a line

**Proposition 4.** Given the correspondence of a point and 3D direction vector, the camera pose can be determined up to an unknown location on a 3D line.

**Proof.** From the line correspondence, first we estimate the unknown camera pan angle using proposition (3). Then, we use the point correspondence in equations (5.7) and (5.10) to establish the locus of the camera center. Substituting the values of the known point coordinates and the estimated value of \( q \), we get equations of the form:

\[
\begin{align*}
t_x &= u_1 t_z + \alpha \\
t_y &= v_1 t_z + \beta
\end{align*}
\]

where \( \alpha \) and \( \beta \) are functions of \((u_1, v_1)\) and \((X_1, Y_1, Z_1)\) and are known. Thus, the locus of the camera translation \( t \) is a 3D line.

\[
t = \frac{t_y}{v_1} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} + \frac{1}{v_1} \begin{pmatrix} \alpha v_1 - \beta u_1 \\ 0 \\ -\beta \end{pmatrix}
\]

with orientation given by the vector \((u_1, v_1, 1)^T\).
Degeneracy. The above formulation is degenerate when \( v_1 = 0 \) (which is when the point \( P_1 \) is on the XZ-plane passing through the camera center). In this case, we can solve for \( t_y \) directly as \( t_y = \beta \) and the locus of the translation is given by equation (5.17). This locus is a line on the XZ-plane passing through the camera center. Note that in this case the camera location is not constrained even with the knowledge of the camera height.

**Corollary 1.** A camera with known rotation and location on a 3D line projects another 3D point to a line in the image which passes through the first point.

**Proof.** Let \( P_2 = (X_2, Y_2, Z_2)^\top \) be the second 3D point (first point being the reference point used to determine the locus of the camera location). Its projection in the image is then given by:

\[
p_2 \cong R P_2 + t
\]

Substituting the locus from equation (5.19), we get:

\[
p_2 \cong \begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} + \frac{t_y}{v_1} \begin{bmatrix} u_1 \\ v_1 \end{bmatrix} + \frac{1}{v_1} \begin{bmatrix} \alpha v_1 - \beta u_1 \\ \alpha u_1 - \beta v_1 \end{bmatrix}
\]

The point \( p_2 \) traces a line in the image as \( t_y \) is varied. As \( t_y \to \infty \), \( p_2 \to (u_1, v_1, 1)^\top \).

This corollary is depicted on a real example in Fig. 16.

\[\square\]

### 5.2 Correspondenceless Geolocalization

With the machinery established in the previous section, we are in a position to describe an algorithm for geolocating a street-level image using a database of 3D...
building corners extracted from a Digital Elevation Map (DEM) of the environment.

Let $P_j \in \mathbb{R}^3$ with $j \in \{1, \ldots, n\}$ be the set of $n$ 3D building corners extracted from the DEM. Each corner is also associated with the 3D direction vector $L_j \in \mathbb{R}^3$ along the roofline of the building\(^1\). The pairing of each point and the direction vector is referred to as a *PointRay* feature. The set of *PointRay* features extracted from the DEM is represented by the database feature set $D = \{(P_j, L_j), j = 1, \ldots, n\}$. An algorithm for creating this feature set from a DEM will be discussed in section-5.2.2.

Similarly, we process the query image to detect a number of building corners and associate them with roofline edges (in the image) to generate a set of 2D *PointRay* features $(p_i, l_i)$ where $p_i \in \mathbb{R}^2$ is the point coordinate and $l_i \in \mathbb{R}^3$ are the line coefficients. This leads to the query feature set $Q = \{(p_i, l_i), i = 1, \ldots, m\}$. An algorithm for creating this feature set from a query image will be discussed in section-5.2.2.

We begin by describing an algorithm that naively uses the 2-point algorithm in Proposition-1 to recover camera pose using $D$ and $Q$.

**Algorithm-0.** We select a pair of features from $Q$ and a pair of features from $D$ and employ the 2-point algorithm to compute pose hypothesis $(R, t)$. Then, we project all 3D-points $P_j$ into the image using the computed camera $(R, t)$, and count the number of image features $p_i$ for which a 3D-point projects within a threshold $\epsilon$. The number of inliers can be used as a score for this pose. Since there is no apriori correspondence information, the above process has to be repeated for all potential minimal (2-point) configurations i.e. $\binom{m}{2} \binom{n}{2} \approx O(m^2n^2)$ times. Clearly, a skyline rendering-based verification step is not feasible with such a large candidate pose space.

\(^1\)Each corner is associated with two roofline edges and hence two different direction vectors. Thus, each corner is represented twice in the set $\{P_j\}$ once for each direction vector.
5.2.1 Stratified Geo-localization Algorithm

We select a single feature from $Q$, associate it with a single feature from $D$ and employ the point-line algorithm in Proposition-4 to compute a pose hypothesis $(R, t_\alpha)$ ($\alpha$ parameterizes the 3D camera height). Then, similar to Algorithm-0, we project all 3D-points $P_j$ into the image using the computed camera $(R, t_\alpha)$. However, in this case, by corrolary-1, each point projects to a line-segment in the image. For each point $p_i$, we search for the closest line-segment (within a reprojection error threshold $\epsilon = 10$ pixels) and associate the corresponding 3D point with this query feature. This set of putative correspondences is now used to solve the 2-point problem (Proposition-1) in a RANSAC-based setting to generate a refined pose. Similar to Algorithm-0, a score can be associated with this pose by counting the number of inliers or by a separate scoring function. The above process is repeated for each pair of features from $Q$ and $D$ i.e. $mn$ times.

Algorithm 5.1 describes the proposed geo-localization algorithm. The function $\text{PosePointRay}(p_i, l_i, P_j, L_j)$ applies Proposition-4 to recover the camera rotation $R$ and translation locus $t_\alpha$ from a single point and direction correspondence. The function $\text{Project}(P_k, K, R, t_{\alpha_1}, t_{\alpha_2})$ projects the 3D-point $P_k$ using the camera matrices $K[R|t_{\alpha_1}]$ and $K[R|t_{\alpha_2}]$ to a line segment in the image $s_k$.

The parameters $\alpha_1$ and $\alpha_2$ specify an interval for allowable camera height. In our implementation, we keep this fairly loose to compensate for image noise and variable ground elevation at each location. Our implementation uses $\alpha_1 = Y_{ground} - 5m$ and $\alpha_2 = Y_{ground} + 20m$ where $Y_{ground}$ is the estimated average ground elevation obtained from the DEM.

The for loop on line-number-7 builds a set of correspondences $C$ by looking for the nearest line-segment from the set $\{s_i\}$ for each query point $p_k$. The function
Nearest($p_k, \{s_j\}, \epsilon$) returns the index of the 3D-point that is closest.

The function PoseTwoPoints($p_i, P_j, C$) uses the two-point algorithm in Proposition-1 using $(p_i, P_j)$ as a fixed correspondence and successively trying out each candidate correspondence from $C$. For each choice, the algorithm produces two solutions from which the rotation which is closer to the original estimate $R$ is selected. Since $C$ can have at most $m$ elements, each call to PoseTwoPoints() invokes Proposition-1 at most $m$ times. However, it outputs only one solution pose ($R', t'$) – the pose for which maximum number of elements of $C$ have a low image projection error.

The run-time complexity of this algorithm is $O(m^2 n)$ and it can produce an output list $P_0$ of length at most $2mn$. Next, we describe the functions Filter and ScoreAndRank.

**Candidate Filtering**

The candidate pose list $P_0$, in the worst case, can be of length $2mn$. Since the camera height was restricted to a fairly loose interval using the parameters $\alpha_1$ and $\alpha_2$, there are potentially a large number of recovered poses which do not agree with the ground-level at the lat-long location of the pose. However, the function Filter removes such poses and generates the output list $P_1$ as follows. First, we filter out all camera poses that land outside the extent of the DEM. Next, we look at the $XZ$-location of each camera pose and look-up the ground elevation from the DEM. If the camera height $c_y$ is within a threshold $h_{cam} \text{m}$ of the ground elevation then we keep this pose, otherwise we filter it out. The threshold $h_{cam}$ allows us to seamlessly model specific geo-localization scenarios. For example, when the query imagery is only taken by walking pedestrians, we can set this threshold to say $h_{cam} = 2.5 \text{m}$ to filter out poses with camera placed higher than 2.5m.
Scoring and Ranking

For geo-localization purposes, each pose \((R, t)\) in the list \(P_1\) needs to be associated with a score so that a ranked list can be created and further verified either using appearance or using a specialized rendering-based matcher. We propose two algorithms for scoring each pose and present experimental comparison of these two methods in section-5.3.

Inlier edge scoring. For this scoring step, we look at the set of inlier correspondences that generated each pose \((R, t)\). Since each building corner is associated with 3 building edge directions, we project these lines for each inlier into the image and score the correspondence based on the edge-strength accumulated by the lines in the query gradient map. The sum of scores over all inlier correspondences constitutes the score for the pose. Fig. 5.3(c) shows an example of the line projection from the inlier correspondences. The intuition behind this scoring strategy is to penalize poor features detected on the query which may not have enough edge-support when measured using projected building edges.

Skyline match verification. The skyline-based verification step is similar to the use of urban-skylines for pose estimation by [Ramalingam et al., 2009]. However, instead of pre-rendering the urban skylines from the continuous pose space, we only render the skyline at the camera poses in the filtered list. In addition, this rendering step is very efficient as it can be carried out using linear algebra alone. We take the 3D point set corresponding to the DEM edge map \(E\) (see section-5.2.2) and project it to the image using the camera matrix. Keeping only the points which fall into the query image, we determine the highest point that projects at each image column by a simple linear search. This gives us the \(v\) coordinate for the rendered skyline.
Algorithm 5.1: Proposed Geo-localization Algorithm

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\mathcal{P}_0 \leftarrow \emptyset$;</td>
</tr>
<tr>
<td>2</td>
<td>for $i \leftarrow 1$ to $m$ do</td>
</tr>
<tr>
<td>3</td>
<td>for $j \leftarrow 1$ to $n$ do</td>
</tr>
<tr>
<td>4</td>
<td>$(R, t_a) \leftarrow \text{PosePointRay}(p_i, l_i, P_j, L_j)$;</td>
</tr>
<tr>
<td>5</td>
<td>$s_k \leftarrow \text{Project}(P_k, K, R, t_{a1}, t_{a2}) \ \forall k \in {1, \ldots, n}$;</td>
</tr>
<tr>
<td>6</td>
<td>$C \leftarrow \emptyset$;</td>
</tr>
<tr>
<td>7</td>
<td>for $k \leftarrow 1$ to $m$ do</td>
</tr>
<tr>
<td>8</td>
<td>$f_k \leftarrow \text{Nearest}(p_k, {s_i}, \epsilon)$;</td>
</tr>
<tr>
<td>9</td>
<td>$C \leftarrow C \cup (p_k, P_f_k)$;</td>
</tr>
<tr>
<td>10</td>
<td>end</td>
</tr>
<tr>
<td>11</td>
<td>$(R', t') \leftarrow \text{PoseTwoPoints}(p_i, P_j, C)$;</td>
</tr>
<tr>
<td>12</td>
<td>$\mathcal{P}_0 \leftarrow \mathcal{P}_0 \cup {(R', t')}$;</td>
</tr>
<tr>
<td>13</td>
<td>end</td>
</tr>
<tr>
<td>14</td>
<td>end</td>
</tr>
<tr>
<td>15</td>
<td>$\mathcal{P}_1 \leftarrow \text{Filter}(\mathcal{P}_0)$;</td>
</tr>
<tr>
<td>16</td>
<td>$\mathcal{P} \leftarrow \text{ScoreAndRank}(\mathcal{P}_1)$;</td>
</tr>
</tbody>
</table>

at each column of the image. Fig. 5.3(d) shows an example of the skyline rendered from a correctly computed pose. For the query image, we directly use the sky-mask from the Geometric Context [Hoiem et al., 2005] algorithm to generate a skyline. At this stage, we can use the two skylines to refine the estimated pose. However, in this work, we only use the skylines to generate a score for each pose – the scoring function measures the area overlap between the sky regions from the rendered and the query image using an intersection over union criterion.

Fig. 5.3 walks through the entire algorithm on a query from our evaluation set.
Figure 5.3: Query processing: a) Each red-dot and green-arrow pair make up one PointRay feature \((p_i, l_i)\). b) The PointRay feature \((p_i, l_i)\) shown in red when corresponded with the correct 3D PointRay feature from the database results in a camera pose locus. By corollary-(1), the camera in this locus maps the remaining database corners to line segments shown in yellow and cyan. The cyan segments represent the inlier set \(C\) because of their proximity to the query corners (shown as green circles). We show two cases here where different locii are created by using a different (but correct) reference PointRay feature correspondence. c) Green circles denote the inliers after the two-point method is applied to \(C\); the green lines depict projected 3D edges using the computed camera pose. d) DEM contours rendered using the hypothesis camera pose generate the green skyline which matches correctly with the perceived skyline. e) DEM showing the reference corners (blue +), recovered camera pose (yellow), ground-truth camera pose (red) and the recovered inliers (green).
5.2.2 Feature Extraction

Database Features. The database feature set $\mathcal{D}$ is generated from a 3D-model or a LIDAR scan as follows. First, we project the data using an orthographic camera onto the ground-plane and keep the height information of the highest point at each pixel. This generates a digital elevation map (DEM) with the elevation information represented by the pixel intensity. We use the median elevation within the DEM as an approximate ground elevation estimate $Y_{ground}$ and use it to mask out the ground regions. This generates building regions as disconnected components which are further processed one at a time. Within each component, we use the video-compass [Košecká and Zhang, 2006] algorithm to detect long line-segments, associate each line-segment with the elevation information from the DEM underneath, and then intersect them pairwise to generate corner locations in the DEM image. Thus, each corner is associated with a 3D location $\mathbf{P}_j$ in the world coordinate system. The line-segments that intersect at this corner are converted to direction vectors $\mathbf{L}_j$ thus generating pairs $(\mathbf{P}_j, \mathbf{L}_j)$. The set of pairs from all building components is collected into the set $\mathcal{D}$ of database features. The canny edge map used for line extraction is also associated with elevation information from the DEM and is stored as a set of 3D coordinates $\mathcal{E}$ for use in the skyline rendering step of our algorithm. Note that the above algorithm is much simpler than the typical processing that is required to extract full building models from a DEM [Matei et al., 2013]. This is a distinct advantage of our PointRay feature which does not need full building outlines to be extracted.

Query Features. Before computing the query features, we rectify the query image (to an upright camera orientation) using vanishing-points. We use the video-compass
algorithm from [Košecká and Zhang, 2006] to fit line segments to the canny edge-map of the query image. This gives us a set of candidate line-segments from which we select candidate “sky-hugging” segments by scoring each segment using a sky-probability map computed by the Geometric-Context algorithm [Hoiem et al., 2005]. The resulting segments are then intersected pairwise to generate candidate building corners. For each corner $p_i$, we thus also obtain the line segment(s) $l_i$ that generated this corner and are along building rooflines by construction. In practice, for each corner we also verify its proximity to the end-points of the intersecting line-segments, and remove multiple corners by non-maximal suppression. The pairs $(p_i, l_i)$ that pass the verification are collected into the final feature set $Q$. Fig. 5.3(a) shows an example of the extracted $PointRay$ features.

5.3 Experiments

We use the publicly available dataset of aerial LIDAR scans of Ottawa, Ontario, Canada. We created a DEM at a ground resolution of 0.25m per pixel for a 1Km × 0.5Km area from this data (see Fig. 5.4) and ran the feature extraction algorithm (section-5.2.2) resulting in approximately $n = 600$ 3D $PointRay$ features.

5.3.1 Query Images

For query data, we use Google Street-view imagery downloadable at specified latitude-longitude locations. 50 images from several locations within the extent of the DEM map at several different camera pan angles were downloaded (see Fig. 5.4 for the query pose variety). The camera tilt was set at zero degrees to simulate an upright camera. The camera internal parameters were fixed at a field-of-view of 90° and a resolution of 640 × 640. This is the maximum available resolution from Google for
Figure 5.4: DEM from Ottawa with query locations overlaid. Each location is shown as a red circle with green arrows depicting the look-at vectors corresponding to each query image. The inset shows the elevation distribution for the query set.

Figure 5.5: Example queries where sky-detection failure disrupts detection of \textit{PointRay} features.

general users and the poor quality (pixellation, ringing, blur) of this imagery poses numerous challenges for line and corner extraction algorithms. For each query, we automatically extract the query features using the algorithm in section-5.2.2. We found that detection of \textit{PointRay} features is quite robust to image quality issues and is most severely impacted by poor sky segmentation from the Geometric Context algorithm when it gets confused by tall construction equipment in the scene.

In Fig. 5.9, we present a visual snapshot of our query set of 50 images to impress upon the reader the complexity of this dataset. We cover several viewpoints, building arrangements, variation in camera-to-building range and scene outliers (like traffic
Figure 5.6: Example queries where geo-localization within 20m of the ground-truth could not be achieved. This is due to the missing PointRay features leading to only a single valid PointRay feature in each image, thus making the geo-localization problem unconstrained. The last panel shows a zoomed-in view of the third example and exhibits the poor quality of the query images.

signals, construction equipment etc.) in this dataset. On each image, we also show the result of our query feature detection stage which shows that we can deal with varying amounts of scene complexity. In addition, we show some images that have a large number of spurious feature detections (on buildings as well as on non-building entities); the performance curves shown later demonstrate that we are robust to this issue.

In Fig. 5.5, we specifically showcase the scenario where the detection of PointRay features gets disrupted due to poor sky-detection. In the two examples shown, we miss a number of building corners due to the incorrect sky segmentation.

In Fig. 5.6, we show three example queries for which geo-localization within 20m of the ground-truth could not be achieved. In all these cases, only a single valid building corner was detected (and a number of spurious non-building corners were detected as PointRay features as well). In this case, the problem becomes unconstrained and we cannot achieve a unique geo-localization. The last panel shows a zoomed-in view of the third query depicting the poor quality of the image leading to failure in PointRay detection. In future work, we plan to improve our PointRay detection
algorithm as well as experiment with a dataset of higher resolution queries.

5.3.2 Results

For each query image, we run our geo-localization algorithm and generate the list of candidate poses $\mathcal{P}$ sorted by decreasing scores. Given the precise ground-truth location and orientation for each query, we label as true-positive each returned pose that is within a distance threshold $\tau$ of the ground-truth location and has a look-at vector within $25^\circ$ of the ground-truth rotation. Fig. 5.7 presents the results for three different distance thresholds, $\tau$: 5m, 10m and 20m. For each value of top-K, the precision is measured as the fraction of queries that returned a true-positive pose within the first top-K elements of the output list $\mathcal{P}$. The two curves within each plot compare the results of using the two scoring strategies outlined before. Using geometry alone, we achieve significant performance considering that for our typical
Figure 5.8: Geometric geo-localization of a night image using *PointRay* features.

case of \( m = 15 \) and \( n = 600 \), we potentially create a ranked list of \( 2mn \) i.e. 18000 poses and still obtain within 20\( m \) localization in the top-100 ranks.

The inset in Fig. 5.4 shows the distribution of ground elevation at the ground-truth locations of the 50 queries in our test set\(^2\). The variation clearly indicates the importance of being able to operate with a flexible camera height assumption in our algorithm. Any algorithm that assumes a fixed camera height (at some nominal ground-plane elevation) would not be able to deal with this variation.

**Geo-Localization of Night Imagery.** We tested our geo-localization algorithm for a night image downloaded from Panoramio (http://www.panoramio.com/photo/)

\(^2\)Since the images were taken by the Google street-view car, it is fair to assume that the camera elevation for the query images would be at an approximately constant offset from the ground elevation.
We used the focal-length estimate from the image EXIF data and estimated the camera tilt using the vertical vanishing point. Fig. 5.8 shows the intermediate stages of our algorithm as well as the localization result on the DEM. The red circle on the DEM shows the ground-truth location obtained from Panoramio – we were able to localize this query to within 3.4m of the ground-truth using the inlier edge scoring method alone and this result was obtained at the first rank of the returned pose list. This demonstrates the usefulness of our approach for difficult scenarios like this where appearance matching can be very challenging.

5.4 Related Work

Image-based geo-localization has largely been approached as an appearance matching problem between a query image and a database of geo-tagged images. [Zamir and Shah, 2010] employed a structured dataset of 360° panoramic imagery from Google Street-view to create an index of SIFT features which is used to geolocate a query image. However, their method requires an extensive dataset to be available and be indexed. SIFT feature matching was also employed for urban localization by [Zhang and Kosecka, 2006]. [Hays and Efros, 2008] propose a data-driven approach to single-image localization which also uses scene features a large dataset. More recent work on image-based geolocalization has also looked at using non-ground-level database imagery. Although direct feature correspondence is not employed in these approaches, appearance information is still used either in a bag-of-words or a feature learning framework. Examples of these two frameworks include work on using self-similarity bag-of-words features for matching to oblique aerial imagery [Bansal et al., 2012], multiple feature learning for matching to satellite and land cover attribute imagery [Lin et al., 2013] and static camera localization by correlating with
Figure 5.9: Samples from the 50 image query set along with the *PointRay* features detected on each. Notice the variety of viewpoints as well as the complexity of building placement captured in this dataset.
satellite imagery [Jacobs et al., 2007].

Digital elevation models (DEM) and 3D models of the environment have shown promise for the geo-localization problem as well. [Baatz et al., 2012] described a framework for geolocating queries in a mountainous terrain by matching against skylines pre-rendered from digital elevation models. [Ramalingam et al., 2011] present a formulation for computing camera pose using minimal configurations of points and lines, and use this to geolocate a query using the 3D model of a city. However, their approach demands the availability of an initial correspondence between one query image and the 3D model. This correspondence is used to setup 3D-to-2D constraints which are then propagated to a new query image using image-to-image appearance matching. Thus, they do not address the geo-localization problem in the traditional sense and implicitly use image appearance. Skylines precomputed from a 3D model have been used for urban geolocalization of an omni-camera in [Ramalingam et al., 2009]. However, the approach has shown more promise for keeping track of the camera location rather than for initialization.

In terms of the problem setup, our work is most closely related to the urban geo-localization setup of [Matei et al., 2013]. They used a LIDAR scan of the environment to create a DEM which is rendered exhaustively from multiple locations and viewpoints. Features extracted from these renderings are matched against query features to generate candidate camera locations. In this work, we employ a DEM as the starting point of our database as well. However, instead of rendering apriori in a quantized pose space, we extract sparse PointRay features which are purely geometric and allow us to compute candidate query poses without any appearance matching. We verify each candidate pose by comparing the building skyline visible in the query with the skyline rendered from the candidate viewpoint. This rendering step is very efficiently performed by linear algebraic means and involves projection of
building contours from the DEM and does not need any depth culling computation.

Closed-form minimal solutions to the absolute pose problem for an upright camera were first proposed by [Kukelova et al., 2011]. We propose a novel formulation for the geo-localization problem using this minimal solver and derive a stratified approach that can work in a correspondence and appearance-free setup by solving for the partial pose using a point-ray correspondence. The 2-point absolute pose framework is also employed by [Saurer et al., 2012] for visual odometry under an upright camera assumption. They also confirm that the vertical direction measurement from an off-the-shelf IMU is accurate enough for the 2-pt pose estimation algorithm.

Correspondenceless estimation of pose or scene structure has been addressed in [Makadia et al., 2007]. However, their method relies on appearance matches between SIFT descriptors while our approach works purely with geometric entities.
Chapter 6

Geometric Polynomial Constraints
in Higher-Order Graph Matching

In the previous chapter, we looked at the combinatorial problem of associating corner and line features between 3D and 2D data. We showed that under the restriction of an upright camera, we can improve upon the combinatorial brute-force solution by a stratification approach that relies heavily on the specific geometry of the problem. Since, such a specialized approach might not be possible in all geometric matching problem setups, in this chapter we explore graph matching as an avenue to pose these problems in an optimization framework.

Graph Matching has been the standard way to formalize correspondence finding as an optimization problem. It has been employed for matching features between image pairs using appearance and geometric constraints, and for matching 3D point sets using geometric constraints. More recent work has extended the graph matching framework to higher-order graphs where affinities representing a match between tuples of features can easily be written as differences of lengths, angles, or cross-ratios.

As an example, for the problem of matching features between two images, one
can directly use a higher-order (hyper) edge of degree 3 to represent the similarity between a pair of triangles as depicted in Fig. 6.1. However, expression of these higher order constraints relies on invariant features that can be computed on each image independently. Thus, one computes a feature vector \( f_{i,j,k} \) representing some invariant property measured from the triangle \( ijk \) in image \( I \) and feature vector \( f_{i',j',k'} \) representing the same property for triangle \( i'j'k' \) in image \( I' \). Then, the affinity of the hyper-edge \((i,i';j,j';k,k')\) is measured as a function of the distance \( ||f_{i,j,k} - f_{i',j',k'}|| \). In this particular example, one may construct the feature vector as \( f_{i,j,k} = (\alpha, \beta, \gamma)^\top \), and the corresponding affinity measure as \( H(i,i';j,j';k,k') = \exp \left( - \frac{(||\alpha - \alpha'|| + ||\beta - \beta'|| + ||\gamma - \gamma'||)}{\sigma^2} \right) \) which implies that the measure would be high only if the corresponding angles of the two triangles \((i,j,k)\) and \((i',j',k')\) match. Several examples of such constraints were shown by [Duchenne et al., 2011] like the perspective-invariant feature vector composed of the three cross-ratios measured from each triangle or difference of angles in [Park et al., 2013, Chertok and Keller, 2010]. These measures are thus limited to intra-image features which can be computed for each triangle in an image independently and then compared against similar features computed for a triangle in the second image.
This limits the applicability of the higher-order matching framework to setups in which such invariant features can be designed.

In this chapter, we propose to extend the idea of higher-order constraints to inter-image constraints of the kind encountered in geometric correspondence problems. As a concrete example, consider the Perspective-n-Point (PnP) problem where we are given a set of 3D points and their 2D projections (with unknown correspondence) and we are interested in solving for the correspondence using purely geometric constraints. It is well known that there are no geometric invariants between a set of 3D points and their 2D projections. Therefore, one cannot employ higher-order constraints for this problem using the framework proposed in the literature. However, the constraints in this problem setup occur in the form of geometric constraints between corresponding points when one considers a minimal configuration of three point correspondences. Given a 3D triangle to 2D triangle correspondence, one can solve for up to four possible solutions for the P3P problem. However, the existence of a solution is not sufficient to say if the chosen triangle correspondence is correct – a fourth correspondence is required to verify the solution. The key idea in this chapter is to estimate an affinity measure for a 4th order hyper-edge directly without first solving the minimal geometric problem completely. To achieve this, we express the geometric problem for each subset of points in a minimal configuration using a univariate polynomial equation. Then, two such minimal sets are consistent only if their polynomial equations share a common root. Thus, the affinity measure can be expressed as a resultant of the polynomial pair (see section 6.1.3).

We believe that with this chapter, we advance the state of the art in the following directions:

- We introduce the novel idea of using the magnitude of the resultant of a pair
of polynomial equations as a measure of agreement between the models represented by the equations that can represent a higher-order affinity across graphs.

- We solve the correspondence problem based purely on geometric polynomial constraints within an existing graph matching optimization framework.

- As opposed to RANSAC which filters initial correspondences, we start with complete lack of any feature matches.

- As opposed to EM, we enable correspondence finding without computing the underlying geometric transformations, thus enabling matching in the presence of multiple transformations like articulated motion.

6.1 Tensor Matching for Geometric Problems

6.1.1 Tensor formulation for higher-order graph matching

[Duchenne et al., 2011] introduced the idea of using a tensor representation for higher-order graph matching problems and we present a brief review of their algorithm in this section.

Consider the problem of matching $N$ points in image $I$ against $N'$ points in image $I'$. This problem is equivalent to determining an $N \times N'$ assignment matrix $X$ such that $X_{i,i'}$ is 1 when point $P_i$ is matched to the point $P_{i'}$ and 0 otherwise. There are additional constraints on the matrix $X$ in the form of unit row or column sums depending on whether we allow many-to-one and one-to-many matching. We can assume that we are given similarity values between pairs of points $P_i$ and $P_{i'}$, which can be collected together into a $N \times N'$ affinity matrix $H$. Thus, each component $H_{i,i'}$ of this matrix corresponds to the similarity between points $P_i$ and $P_{i'}$. The
matching problem can now be formulated as the maximization of the cost function
given by:

$$\text{score}(X) = \sum_{i,i'} H_{i,i'} X_{i,i'}$$  \hspace{1cm} (6.1)

subject to the row/column stochasticity constraints on $X$. The $N \times N'$ matrix $H$
can be re-written as a $NN'$-dimensional vector $\tilde{H}$ by treating each pairing of the
indexes $(i,i')$ as a single index $\tilde{i}$ in the range $\{0,\ldots,NN'-1\}$. Similarly, we can rewrite $X$ as a $NN'$-dimensional vector $\tilde{X}$ which is obtained by concatenating the
columns of $X$. This allows the score in (6.1) to be specified as:

$$\text{score}(\tilde{X}) = \sum_{\tilde{i}} \tilde{H}_{\tilde{i}} \tilde{X}_{\tilde{i}} = \tilde{H} \otimes \tilde{X} \hspace{1cm} (6.2)$$

where the second expression was obtained by recognizing $\tilde{H}$ as a rank-1 tensor and
using $\otimes$ to denote the tensor inner product between the two tensors.

The above formulation considered only first order matching constraints i.e. con-
straints specified between a single pair of nodes. We can generalize this to incorporate
similarity between tuples of points (with $d$ points per tuple) by using a higher-rank
tensor. For example, for $d = 3$, the similarity between tuples of points $(P_i, P_j, P_k)$
and $(P_{i'}, P_{j'}, P_{k'})$ can be specified as $H_{i,i';j,j';k,k'}$. In this case the matching cost
function can be written as:

$$\text{score}(X) = \sum_{i,i',j,j',k,k'} H_{i,i';j,j';k,k'} X_{i,i'} X_{j,j'} X_{k,k'}$$  \hspace{1cm} (6.3)

which adds the similarity value $H_{i,i';j,j';k,k'}$ to the score function if and only if each
of the assignments $X_{i,i'}$, $X_{j,j'}$ and $X_{k,k'}$ are set to 1. Once again, by considering
Algorithm 6.1: Rank-3 Tensor Power Iteration Algorithm

input : Super-symmetric tensor $\tilde{H}$ of rank-3.
output : $V$, the first eigen-vector of $\tilde{H}$.

1. Initialize $V$ with random non-negative values;
2. while not converged do
   3. $V \leftarrow ((\tilde{H} \otimes V) \otimes V)$ ;
   4. $V \leftarrow \frac{V}{||V||_2}$ ;
5. end

each pairing $(i, i')$ as a single index $i$, we can rewrite $H$ as a rank-3 tensor $\tilde{H}$ of size $(NN')^3$. Note that $\tilde{H}$ is a rank-3 super-symmetric tensor which means that its elements are invariant under any permutation of its indices.

The score function in (6.3) can now be written in the tensor notation as:

$$
\text{score}(X) = \sum_{i,j,k} \tilde{H}_{i,j,k} \tilde{X}_i \tilde{X}_j \tilde{X}_k
= (((\tilde{H} \otimes \tilde{X}) \otimes \tilde{X}) \otimes \tilde{X})
$$

The score function in (6.4) can be optimized using the tensor power iteration algorithm proposed by [Duchenne et al., 2011] which generalizes the idea of power iterations for eigenvalue problems. They also show a version of the algorithm that uses $\ell^1$-norm constraints on the rows of the assignment matrix $X$ to generate a close to boolean output matrix. Algorithms-6.1 and 6.2 outline the power iteration algorithms with a single tensor and for a mix of different rank tensors. For further details, the reader is referred to [Duchenne et al., 2011].
Algorithm 6.2: Mixed Order Tensor Power Iteration Algorithm

| input | : Super-symmetric tensors $\tilde{H}_3$ of rank-3, $\tilde{H}_2$ of rank-2 and $\tilde{H}_1$ of rank-1. |
| output | : $V$, the first eigen-vector of $\tilde{H}_i$. |

1. Initialize $V$ with random non-negative values;
2. while not converged do
3. $V \leftarrow ((\tilde{H}_3 \otimes V) \otimes V) + (\tilde{H}_2 \otimes V) + \tilde{H}_1$;
4. $V \leftarrow \frac{V}{\|V\|_2}$;
5. end

6.1.2 Geometric Constraints for Higher-Order Graphs

Consider a geometric matching problem where the objective is to match a set $\mathcal{P}$ of $n$ points in instance I against a set $\mathcal{P}'$ of $n'$ points in instance II, subject to some geometric constraints. We will assume that there are no appearance constraints that can be used to aid correspondence. Additionally, we will assume that there are no geometric invariants that can be computed and matched between subsets of points in $\mathcal{P}$ or $\mathcal{P}'$. This is the case, for example, in the “Perspective-n-Point” (PnP) problem where $\mathcal{P}$ will specify a set of 3D points and $\mathcal{P}'$ will specify the corresponding set of 2D projections of these points into a calibrated camera. In a problem like this, the geometric constraints are typically specified for a “minimal configuration” of $m$ point correspondences from the sets $\mathcal{P}$ and $\mathcal{P}'$. For concreteness, let $S = \{(p_1, p'_1), (p_2, p'_2), \ldots, (p_m, p'_m)\}$ be a set of $m$ correspondences from the set $\mathcal{P} \times \mathcal{P}'$. Without loss of generality, let $F_{\tau}(p_i, p'_i) = 0$ be a (polynomial) function specifying the geometric constraint between points $p_i$ and $p'_i$ with the unknown parameter vector $\tau$ providing a parametrization of the global geometric configuration. The set of constraints $\mathcal{C} \equiv \{F_{\tau}(p_i, p'_i) = 0\}$ for $i = 1, \ldots, m$ can be used jointly to solve for the parameter vector $\tau$ by successive elimination of variables to reduce the set $\mathcal{C}$ to
Figure 6.2: Correspondences in sets $S$ and $S'$ are the hyper-edges corresponding to the minimal configuration and generate constraints in the form of polynomial equations $q_S = 0$ and $q_{S'} = 0$ respectively. These sets are combined to form a new hyper-edge with weight given by the resultant of the Sylvester matrix $M(q_S, q_{S'})$ of the two polynomials.

A single univariate polynomial equation of (say) degree $k$:

$$q_S(x) \equiv a_{S,k}x^k + a_{S,k-1}x^{k-1} + \ldots + a_{S,0} = 0$$  \hspace{1cm} (6.5)

where the subscript $S$ reflects the dependence of the polynomial on the chosen minimal set $S$. In this thesis, we focus on the problems for which such a reduction to a univariate polynomial is possible and we explicitly derive the formulation for three common problems in computer vision.

The polynomial equation (6.5) can be solved to obtain solution(s) for the parameter $x$ and then back-substituted into the set $C$ to determine the full parameter vector $\tau$. However, this is not sufficient to associate a cost with the set $S$. Typically, a new point correspondence $(p_{m+1}, p'_{m+1})$ is used to test and validate the solutions of (6.5). However, this approach needs one to solve for the full parameter vector $\tau$ and then evaluate the new correspondence. Instead, we propose to use the $m+1^{th}$ correspondence to define another minimal set $S'$ that shares $m-1$ correspondences

\footnote{For certain problems, the non-existence of real solutions for the equation $q_S(x)$ can be used to associate a cost value. However, there are robustness issues with this approach if the data points are noisy.}

124
with the set $S$ and define a cost directly using the polynomial equations $q_S$ and $q_{S'}$. This set $S' = \{(p_2, p'_2), (p_3, p'_3), \ldots, (p_{m+1}, p'_{m+1})\}$ is obtained from $S$ by replacing the first correspondence $(p_1, p'_1)$ by another correspondence $(p_{m+1}, p'_{m+1})$. Since this is a minimal set, we can derive a polynomial equation $q_{S'}(x)$ where $x$ is the same variable\(^2\) as in the polynomial for set $S$. The set $S \cup S'$ consists of $m+1$ correspondences and is geometrically consistent if the polynomials $q_S$ and $q_{S'}$ share a common root.

In section 6.1.3, we describe an approach to quantify the existence of a common root between the two univariate polynomials using the Sylvester resultant. Using the resultant, we directly measure the distance of the two polynomials from co-primeness without needing to evaluate their roots and solving for the full parameter vector $\tau$. Fig. 6.2 illustrates this construction for $m = 3$ which is the minimal configuration for the $P3P$ problem discussed in section 6.2.1.

### 6.1.3 Polynomial Resultant as Edge Affinities

Consider the family of univariate degree-$n$ polynomial equations $\{p_i(x)\}$ defined as:

$$p_i(x) \equiv a_{i,n}x^n + a_{i,n-1}x^{n-1} + \ldots + a_{i,0} = 0 \quad (6.6)$$

The polynomials are assumed to have a unitary 2-norm i.e.

$$\| (a_{i,n}, a_{i,n-1}, \ldots, a_{i,0})^\top \|_2 = 1 \quad (6.7)$$

Given two polynomials $p_i$ and $p_j$ from this family, we are interested in the problem of determining if they have a common root. This problem can be approached by considering the resultant matrix of the polynomials. A resultant matrix of two

\(^2\)Note that problem-specific tricks might be required to ensure that the variable $x$ is indeed a variable that can be shared between the polynomials $q_S$ and $q_{S'}$.\)
polynomials is a matrix obtained from the polynomial coefficients with the property that the polynomials have a common root if and only if the matrix has a zero determinant. Two of the most common resultant matrices often employed are the Sylvester and the Bézout matrices. In this work, we employ the resultant of the Sylvester matrix corresponding to the polynomial pair \((p_i, p_j)\) because of its simpler form in terms of the polynomial coefficients in comparison to the Bézout matrix. The Sylvester matrix \(M(p_i, p_j)\) is defined as follows:

\[
M(p_i, p_j) = \begin{pmatrix}
   a_{i,n} & a_{i,n-1} & \cdots & a_{i,0} \\
   \vdots & \ddots & \ddots & \ddots \\
   a_{j,n} & a_{j,n-1} & \cdots & a_{j,0}
\end{pmatrix}_{2n \times 2n}
\]  

It is well known that the Sylvester matrix is a resultant matrix [Laidacker, 1969]. Given a vector \(y = (x^{2n-1}, x^{2n-2}, \ldots, x, 1)^\top\), the system \(My = 0\) has a solution if and only if the polynomials \(p_i(x)\) and \(p_j(x)\) have a common root. This implies that the square matrix \(M\) has to be rank deficient for the polynomials to share a root. Thus, we can use the magnitude of the smallest singular value \(\sigma_{\text{min}}(M)\) of \(M\) as a measure of co-primeness of the polynomials \(p_i\) and \(p_j\). In addition, the last non-zero row of the \(R\) matrix obtained by a QR-factorization of the matrix \(M\) represents the coefficients of the GCD polynomial for \(p_i\) and \(p_j\) (see [Corless et al., 2004]). Because of this property, we can also use the absolute value of the last element \(R_{2n,2n}\) of \(R\) to measure the co-primeness of \(p_i\) and \(p_j\).
The above formulation allows us to assign an affinity value to the hyper-edge $S \cup S'$ defined in section 6.1.2 as follows:

$$H_{S \cup S'} = e^{-|M(q_S, q_{S'})|/\rho}$$

(6.9)

where $|M(q_S, q_{S'})|$ represents the resultant value estimated either from the SVD as $\sigma_{\text{min}}(M(q_S, q_{S'}))$ or from the QR factorization as $|R_{2n,2n}|$, and $\rho$ is a parameter that determines the spread of the exponential function in (6.9).

For the simulations in this chapter, we experimented with both methods for computing the resultant value and found them to perform equally well. Therefore, we picked the QR factorization based approach for all the experimental results because of its lower computational complexity as compared to SVD. The parameter $\rho$ was set empirically in our experiments and kept constant for all the instances and problems. We present an empirical analysis of the robustness and discriminability of this resultant-based edge affinity measure in Sec. 6.3.3 where we show how for the three-point calibrated absolute camera pose problem (P3P), this measure allows us to separate between valid and invalid correspondences while exhibiting robustness to Gaussian noise in the image coordinates.

**Overall Algorithm**

Collecting the ideas presented in the previous sections, the general framework to approach graph matching for geometric problems is outlined in Algorithms-6.3 and 6.4. In Algorithm-6.3, we describe the necessary steps to prepare a given geometric problem for higher-order graph matching. In the next section, we discuss formulations for specific geometric problems that follow this outline. In Algorithm-6.4, we describe the solution method to estimate the correspondence matrix given a particular...
Algorithm 6.3: Preparing a geometric problem for higher-order graph matching.

1. Consider the correspondence problem in its minimal configuration (of size say \(m\)) and analytically derive the minimal geometry equations relating the \(m\) pairs of observations (points) to the unknown geometric variables.
2. **Analytically** combine the constraint equations over the minimal set into a univariate polynomial with coefficients dependent on the \(m\) pairs of observations in the minimal set.

Algorithm 6.4: Higher-order Geometric Graph Matching Algorithm

1. Sample hyper-edge tuples of \(m + 1^{th}\) order from the space of all possible \(m + 1^{th}\) order tuples of correspondences.
2. For each sample, **numerically** compute the coefficients of the two univariate polynomials derived in Algorithm-6.3. Compute the affinity value for this hyper-edge by plugging-in these coefficients into equation (6.9).
3. Apply the tensor power iteration method from [Duchenne et al., 2011] to the computed affinity tensor to compute an assignment matrix.

6.2 Formulation for Specific Problems

In the following, we describe novel formulations for three specific geometric problems. Two of the problems deal with absolute camera pose recovery given 3D points as \(\mathcal{P}\) and their 2D projections as \(\mathcal{P}'\). The third problem deals with the relative camera pose problem (visual odometry) in a setting where the camera rotation axis is known (or can be estimated by a directional correspondence). In this case, only three point correspondences are required and we show how we can formulate this in our graph-matching framework. The sets \(\mathcal{P}\) and \(\mathcal{P}'\) in this case are both 2D image points.
6.2.1 Three-point Calibrated Absolute Pose Problem (P3P)

There are several algorithms for this classic problem in the literature. However, in this chapter we will work with the formulation proposed by [Fischler and Bolles, 1981] in their classic RANSAC paper.

Minimal Setup

The minimal setup consists of three 3D points $X_a$, $X_b$ and $X_c$ being observed by a camera at a 3D location $O$ as shown in Fig. 6.3. Given image projections $u_a$, $u_b$ and $u_c$ of these points and the camera calibration matrix $K$, the problem is to estimate the absolute camera pose. Given the pairwise distances between the 3D points i.e. $||X_a - X_b|| = R_{ab}$, $||X_b - X_c|| = R_{bc}$, $||X_c - X_a|| = R_{ca}$ and the angle subtended by each pair of image points at the camera center i.e. $\cos(\angle X_aOX_b) = C_{ab}$, $\cos(\angle X_bOX_c) = C_{bc}$, $\cos(\angle X_cOX_a) = C_{ca}$, the problem is to estimate the distances of the points from the camera center. Let these unknowns be denoted by $a$, $b$ and $c$, i.e. $||X_a - O|| = a$, $||X_b - O|| = b$, $||X_c - O|| = c$.

Each of the triangles $OX_aX_b$, $OX_bX_c$ and $OX_cX_a$ provides a quadratic constraint.
per the cosine law:

\[ R_{ab}^2 = a^2 + b^2 - 2abC_{ab} \]  
(6.10)

\[ R_{bc}^2 = b^2 + c^2 - 2bcC_{bc} \]  
(6.11)

\[ R_{ca}^2 = c^2 + a^2 - 2caC_{ca} \]  
(6.12)

By introducing new variables \( x \) and \( y \) such that \( x = b/a \) and \( y = c/a \), algebraic manipulation of the system (6.10)-(6.12) leads to the following quartic polynomial equation in \( x \):

\[ G_4^{abc} x^4 + G_3^{abc} x^3 + G_2^{abc} x^2 + G_1^{abc} x + G_0^{abc} = 0 \]  
(6.13)

Solving this polynomial equation leads to four possible solutions for the variable \( x \) which is the ratio of the depths of the points \( X_b \) and \( X_a \) from the camera center \( O \). The depths \( a, b \) and \( c \) can then be solved for using the system (6.10)-(6.12).

**Hyper-edge**

As outlined in section 6.1.2, since the minimal solution may not provide any constraints in the general case, we add another 3D point \( X_d \) (and the corresponding image point \( u_d \)) and derive another quartic polynomial by considering the tetrahedron \( OX_aX_bX_d \). With \( ||X_b - X_d|| = R_{bd}, ||X_d - X_a|| = R_{da}, \cos(\angle X_bOX_d) = C_{bd}, \cos(\angle X_dOX_a) = C_{da} \) and additional variable \( ||X_d - O|| = d \), we get the following quartic polynomial:

\[ G_4^{abd} x^4 + G_3^{abd} x^3 + G_2^{abd} x^2 + G_1^{abd} x + G_0^{abd} = 0 \]  
(6.14)

where \( x \) is defined again as \( x = b/a \) and \( z \) is defined as \( z = d/a \).

We can now define a 4\(^{th}\) order hyper-edge \( e \equiv (X_a, u_a; X_b, u_b; X_c, u_c; X_d, u_d) \) with
the edge affinity $H_e$ defined by (6.9) with $S = (X_a, u_a, X_b, u_b, X_c, u_c)$ and $S' = (X_a, u_a, X_b, u_b, X_d, u_d)$; $M(q_S, q'_S)$ is the $8 \times 8$ Sylvester matrix of the two polynomials (6.13) and (6.14). Note that this formulation works since the variable $x = b/a$ is shared between the two equations and hence the problem of looking for a common root for the polynomial pair is well defined.

### 6.2.2 Three-plus-One Calibrated Relative Pose Problem (3P1)

In this problem setup, recently proposed by [Naroditsky et al., 2012], we are given three image correspondences $q_i \leftrightarrow q'_i, i = 1, 2, 3$ in two calibrated views along with a single directional correspondence $d \leftrightarrow d'$. The problem is to determine the essential matrix $E$ relating the two cameras and thus estimate the relative pose between the cameras. It was shown in [Naroditsky et al., 2012] that by taking into account the directional constraint, the degrees of freedom in the essential matrix can be reduced from 5 to 3 and the problem can be formulated in closed-form as the solution of a univariate quartic polynomial $\sum_{i=0}^{4} h_i x^i = 0$, where the variable $x = \cos(\theta)$ corresponds to the cosine of the one-parameter rotation angle between the two cameras. This allows us to formulate this problem in the same manner as the P3P problem using a 4th order affinity tensor derived from the resultant of a pair of these quartic polynomials. However, in this case we have multiple choices for the polynomial pairs for each hyper-edge $(q_1, q'_1, q_2, q'_2, q_3, q'_3, q_4, q'_4)$ since the shared variable $x$ is a camera parameter and is not dependent on the points used (unlike the P3P case where it was dependent on the depths of two of the points). Thus, we can derive $\binom{4}{2} = 4$ different polynomial equations from these 4 correspondences which can be paired to result in $\binom{4}{2} = 6$ different resultant values for the hyper-edge affinity. For the simulations in
this chapter, we have used one of these 6 values as the hyper-edge affinity although more complex values derived from multiple of these is also possible.

6.2.3 Two-point Calibrated Absolute Pose for an Upright Camera (up2p)

[Kukelova et al., 2011] proposed a closed-form solution to the absolute pose problem for an upright camera from two 2D-3D correspondences. Since the camera is parameterized by a single angle $\theta$ about the vertical axis, the problem has a total of four unknowns – the three components of the translation vector and the camera rotation $\theta$. Thus, unlike the P3P case, only two correspondences are required to estimate the camera pose. It was shown in [Kukelova et al., 2011] that the constraints from the minimal set of two correspondences can be combined to derive a univariate quadratic equation $\sum_{i=0}^{2} h_i x^i = 0$, where the variable $x = \tan(\frac{\theta}{2})$ is a function of the camera rotation and is thus not dependent on the points chosen. Therefore, we can formulate this problem using a 3rd order affinity tensor derived from the resultant of a pair of these quadratic equations. In this case, we will get a $4 \times 4$ Sylvester matrix from which we can compute the tensor affinity value for each hyper-edge like before. Note that, in this case, we again have $\binom{3}{2} = 3$ different quadratics from the minimal (3 correspondence) set from which we can compute $\binom{3}{2} = 3$ different resultant values. The simulations in this chapter use only one of these 3 values to define the hyper-edge affinity.
6.3 Experiments

We performed simulations for each of the three problems to characterize the robustness of our geometric tensor formulation under varying noise and outlier configurations. For each problem simulation, we generated 100 instances of the problem randomly and used the proposed method to compute an assignment matrix $X$ for each instance. To characterize the dependence of the algorithm accuracy on the number of sampled hyper-edges, we keep track of the affinity tensor $\tilde{H}_i$ for each sample size $s_i$ and compute the corresponding assignment matrix $X_i$ using the tensor $\tilde{H}_i$. The matching accuracy for each sample size $s_i$ is measured as the number of good matches in the computed $X_i$ divided by $n$ where $n$ is number of points used for the specific simulation. In the following, we report results on the accuracy value averaged over the 100 instances under different simulation conditions and provide details about the specific simulation setup for each problem. We used the authors’ [Duchenne et al., 2011] implementation of the tensor power-iteration algorithm and extended it to allow inclusion of 4th order affinities. In all cases, we assume projection to a $640 \times 480$ image from a camera with an effective focal-length of $f_u = f_v = 1000$.

6.3.1 P3P with Higher-order Geometric Constraints

Simulation Setup

For each problem instance, $n = 10$ 3D points were generated uniformly at random in a $4 \times 4 \times 4$ cube centered at the origin. For each instance, the camera center was also chosen uniformly at random on the surface of a sphere with radius 12 centered at the origin. The camera rotation was set so that its optical axis passes through the origin ensuring that all the points are always within the camera FOV.
Noise Sensitivity

Fig. 6.4 compares the accuracy achieved by our algorithm for different amounts of Gaussian noise added to the image points as a function of the number of random tensor edges sampled from the sample space. The solid curves represent the accuracy using the sparse tensor power-iteration algorithm (i.e. with the $\ell^1$-norm constraint on $X$) while the dashed curves were obtained for the dense formulation. It is clear that the sparse solver performs significantly better than the dense solver. In Fig. 6.4 (a), we show results without any outliers added to the image point set. For the noise-free case ($\sigma = 0$), we note that the algorithm achieves 100% accuracy in a very small number of samples indicating that our polynomial-based cost is very effective in enforcing the geometric constraints. The other plots correspond to increasing noise level from $\sigma = 0.1$ to $\sigma = 1.0$ pixel. As expected, noise in the image coordinates interferes with the geometric constraints and hence many more samples are required for the graph-matching to be able to deduce the correct correspondence. Also note that the curves asymptote at less than 100% accuracy because of the inability of purely geometric constraints derived from noisy observations being able to guide the matching process.

Performance with Outliers

In Fig. 6.4 (b), we show results for simulations with random outlier points added to the image. Thus, for the 5 outlier scenario, the simulation generated $n = 10$ uniformly random 3D points as before, projected them to the image to generate 10 2D points, and then added 5 additional 2D points uniformly at random to simulate outliers. Then, Gaussian noise with $\sigma$ specified by the experiment was added to all the 2D points. The plots show that the approach can handle a large number of
Figure 6.4: Simulation results for the P3P Problem using 3D geometry constraints. Average matching accuracy across 100 randomly generated instances of the P3P problem ($n = 10$) for different amounts of Gaussian noise. The solid and dashed curves represent the performance of tensor power-iteration algorithm [Duchenne et al., 2011] with and without the $\ell^1$-norm constraint on the assignment matrix respectively. In (a) no outliers were added; in (b) 50% and 100% additional random image points were added as outliers to the inlier set of 10 points.

outliers in noise-free case but the performance degrades rapidly when combined with image noise. Also note that in this case the assignment matrix $X$ is non-square and we are therefore solving a subgraph matching problem here.

### 6.3.2 P3P using Higher-order Angle Constraints

We ran the P3P simulation using the angle difference measure used by [Chertok and Keller, 2010] (also shown in Fig. 6.1), as the only higher-order constraint. Note that while this measure is useful for scale invariance in an image-to-image setup, the angles will not be preserved from 3D-to-2D thus validating if constraints derived in our general geometric framework provide any benefit. Fig. 6.5 shows the performance achieved across 100 randomly generated instances of the P3P problem. Without outliers, the angle based constraints perform worse than the geometric constraints for all noise levels. This shows that using 3D-2D geometric constraints, we can exploit
Figure 6.5: **Simulation results for the P3P Problem using angle constraints.** Average matching accuracy across 100 randomly generated instances of the P3P problem ($n = 10$) for different amounts of Gaussian noise. The *solid* and *dashed* curves represent the performance of tensor power-iteration algorithm [Duchenne et al., 2011] with and without the $\ell^1$-norm constraint on the assignment matrix respectively. In (a) no outliers were added; in (b) 50% and 100% additional random image points were added as outliers to the inlier set of 10 points.

more meaningful information about the true correspondence structure. The angle constraints are able to remove some incorrect correspondences but are still unable to achieve high accuracy even in the noise-free case. A similar trend is observed when outliers and image noise are simultaneously included as shown in Fig. 6.5(b). Our method performs significantly better than angle constraints, has better robustness to outliers and captures the geometric constraints more completely.

### 6.3.3 Noise Robustness of the Edge Affinity Measure

In Fig. 6.6, we present an analysis of the robustness of the edge affinity measure (6.9) in the P3P problem setup. Keeping the simulation setup the same as that for the P3P experiments above, 1000 sets of 4 3D-2D point correspondences were generated. The edge affinity measure (6.9) was evaluated for each such valid set and the mean and standard-deviation across the 1000 trials was plotted as the error-bars in Fig. 6.6. To
Figure 6.6: **Robustness of the Edge Affinity Measure for the P3P Problem.** Error bars ($\mu \pm \sigma$) for the affinity measure (6.9) across 1000 randomly generated hyper-edges from the P3P problem for different amounts of Gaussian noise. The red and blue curves represent performance on sets of valid vs. invalid correspondences respectively indicating the discriminability of the affinity measure as well as the increase in confusion between valid and invalid configurations at higher noise levels.

To evaluate noise-sensitivity, Gaussian noise with varying $\sigma$ was added to the projected 2D points before computing the affinity measure. The red plot in the figure shows the variation of the mean affinity value (along with the standard-deviation) with increasing noise level for these sets of valid correspondences.

Evaluate the discriminability of the affinity measure in separating valid and invalid correspondences, we generated another 1000 sets of 4 3D-2D point correspondences where the 3D-2D points did not correspond to a valid configuration. Gaussian noise was again added to the image points as before. The blue plot in Fig. 6.6 shows the variation of the affinity measure on these sets of invalid correspondences. We can observe that the mean affinity measure is well separated between valid and invalid configurations in the noise-free case and as expected, the gap is reduced as the noise
level is increased. A similar observation can be made for the standard-deviation which shows no overlap in the noise-free case but a significant overlap with a 5-pixel image noise. The large overlap explains why significantly more samples (hyper-edges) are needed to obtain a valid solution in Fig. 6.4(a) in the presence of added noise – a higher percentage of invalid hyper-edges score in the higher-affinity range and similarly a higher percentage of valid hyper-edges score in the lower-affinity range leading to more confusion for the tensor matching algorithm; sampling more edges increases the probability of sampling from the correct distribution.

### 6.3.4 Comparison with RANSAC

The Random Sample Consensus (RANSAC) algorithm [Fischler and Bolles, 1981] is a hypothesize-and-test framework which is used to estimate the parameters of a given model on a set of input data corrupted with outliers. The parameters of the given model are estimated on a minimal hypothesis set and then tested against the remaining data to estimate the number of inliers. This process is repeated for random choices of the minimal hypothesis set while keeping track of the best model (defined as the one which leads to either the maximum number of inliers or another criterion on the inlier set). RANSAC terminates when the probability of finding a better model drops below a certain threshold.

RANSAC and its several variants have been applied successfully to geometric estimation problems where a set of correspondences (with outliers) is already specified. Algorithm-6.5 outlines the basic version of RANSAC for the P3P problem, given a set of 3D-2D correspondences $C$. The number of trials $N$ is chosen high enough to ensure with a high probability $p$ that at least one of the random hypothesis sets chosen is free of outliers. Let $w$ be the probability of picking a single inlier. Then
**Algorithm 6.5:** RANSAC Algorithm for P3P problems with given correspondences

1. Pick a random hypothesis subset $c \in C$ containing three 3D-2D correspondences.
2. Employ any three-point calibrated absolute pose algorithm to compute a camera pose $P$ using the hypothesis set $c$.
3. Test each 3D-2D correspondence $(X, x)$ against the computed pose $P$ by projecting the 3D point $X$ and computing the projection error (pixel distance) between the projection $PX$ and the 2D point $x$. Count the number of correspondences (inliers) for which the projection error is smaller than a threshold $\delta$.
4. Repeat steps 1-3 $N$ times. On each iteration, keep track of the pose $P^*$ and the hypothesis set $c^*$ that gives the maximum number of inliers.
5. Compute a refined camera pose by fitting a least square model to the inlier set for the pose $P^*$ obtained at the conclusion of the iterative steps above.

The probability of picking a hypothesis with at least one outlier is given by $1 - w^3$ (in the P3P case). Thus, after $N$ trials, the probability $p$ can be computed as:

$$1 - p = (1 - w^3)^N$$  \hspace{1cm} (6.15)

which leads to:

$$N = \frac{\log(1 - p)}{\log(1 - w^3)}$$  \hspace{1cm} (6.16)

For Algorithm-6.5, the value $w$ can be determined directly from the fraction of inliers in the initial correspondence set $C$ (if known).

To allow the application of RANSAC to the correspondenceless setting, we need to modify the hypothesis testing framework to work without a given set of correspondences. We propose the adaptation in Algorithm-6.6 which starts from unordered sets of 3D points and 2D points $Q_W$ and $Q_I$ respectively. Since there are no correspondences, step-3 of the algorithm has to compare each projected 3D point against all given 2D points (unlike the with-correspondence case, where the comparison is
Algorithm 6.6: RANSAC Algorithm for P3P problems without correspondences

1. Pick a random ordered subset \( \{i_1, i_2, i_3\} \) of 3D point indices from the set \( Q_W \) and a random ordered subset \( \{j_1, j_2, j_3\} \) of 2D point indices from the set \( Q_I \).
2. Employ any three-point calibrated absolute pose algorithm to compute a camera pose \( P \) using the hypothesis set \( c \) formed by assuming correspondence of 3D-2D points in the selected subset above.
3. Project all 3D points in the set \( Q_W \) using the computed pose \( P \) and for each projection, find the closest 2D point in the set \( Q_I \). Count the number of 3D points for which the projection is within a threshold \( \delta \).
4. Repeat steps 1-3 \( N \) times. On each iteration, keep track of the pose \( P^* \) and the hypothesis set \( c^* \) that gives the maximum number of inliers.

only to the 2D point specified as the correspondence). In addition, the number of trials \( N \) has to take into account the revised probability \( w \) of picking a single inlier correspondence. Given \( N \) 3D-points and \( n \) 2D points, this probability is given by
\[
w = \frac{k}{Nn}
\]
where \( k \) is the number of inlier correspondences. Consider the setting with no outliers and equal number of 3D and 2D points i.e. \( k = N = n \). Even in this case, \( w = 1/n \) is quite small and thus the required number of trials \( N \) in (6.16) is high (for \( n = 10 \) and \( p = 0.99 \), \( N \approx 5000 \)).

We compare the performance of the proposed tensor matching framework against RANSAC by implementing Algorithm-6.6. The simulation setup is the same as that described in Sec. 6.3.1 where 100 random instances of the P3P problem (with \( n = 10 \)) are generated and varying amounts of Gaussian noise is added to the image points. Fig. 6.7 shows the matching accuracy results obtained with increasing number of trials \( N \in [100, 5000] \) averaged across the 100 problem instances. Observe that the curves are not monotonically increasing even though the number of inliers always increases as more trials are conducted. This is because of the inability of the hypothesis verification step in distinguishing an incorrect match from an inlier given the lack of correspondences. A solution that leads to a higher number of 3D points
6.3.5 Employing High-confidence Matches

Consider the geometric matching problem as described above but with an additional constraint that a few high-confidence correspondences are already known. We can incorporate these constraints directly as first order constraints in the higher-order
Figure 6.8: **Simulation Results for the P3P problem given some high-confidence matches.** Average matching accuracy across 100 randomly generated instances of the P3P problem \((n = 10)\) for different number of initially specified high-confidence matches. The accuracy achieved by the tensor-matching algorithm steadily improves even with \(\sigma = 1.0\) pixel image noise as more high-confidence seed correspondences are specified.

We evaluate the impact of adding high-confidence correspondences to the accuracy achieved by the tensor matching algorithm for the P3P problem. For the same simulation setup as before, we vary the number of pre-specified high-confidence correspondences from 1 to 3 and measure the average accuracy of the tensor matching...
algorithm where the high-confidence correspondences are added as first order constraints. Fig. 6.8 shows that adding even a single point correspondence leads to a significant boost in performance and the algorithm can deal with $\sigma = 1.0$ pixel image noise very robustly.

### 6.3.6 3P1 with Higher-order Geometric Constraints

**Simulation Setup**

For each problem instance, $n = 10$ 3D points were generated uniformly at random in a $4 \times 4 \times 4$ cube centered at the origin. For each instance, the first camera center was chosen uniformly at random on the equator of a sphere with radius 12 centered at the origin. The second camera center was chosen uniformly at random in the equatorial plane at a displacement of $b$ units (*stereo baseline*) from the first center. We report results for two values of $b$ in Fig. 6.9. The rotation for both cameras were set so that their optical axis passes through the origin ensuring that all the points are always within the camera FOV. Thus, instead of simulating the directional correspondence, we assume that the relative camera rotation is around the vertical axis only.

**Noise Sensitivity**

Fig. 6.9 compares the accuracy achieved by our algorithm for different amounts of Gaussian noise added to the image points (*in both images*) as a function of the number of random tensor edges sampled from the sample space. Like in the P3P plots, the solid and dashed curves represent the accuracy using the *sparse* and *dense* tensor power-iteration algorithms respectively with the sparse solver again performing significantly better. Panels (a) and (b) compare the performance for two different baseline settings in the simulation with $b = 1$ and $b = 5$. We note that for the
Figure 6.9: Simulation results for the 3P1 Problem. Average matching accuracy across 100 randomly generated instances of the 3P1 problem ($n = 10$) for different amounts of Gaussian noise. The solid and dashed curves represent the sparse and dense tensor power-iteration algorithms [Duchenne et al., 2011]. The left and right columns compare performance for two different values of the baseline between the two cameras. The top and bottom rows show results without and with added outliers respectively.

smaller baseline, the points are less likely to confuse with each other and are thus more robust to noise. For the larger baseline, the performance degrades rapidly with
added noise and does not reach 100% accuracy even in the noise-free case indicating likely degenerate conditions for this problem.

**Performance with Outliers**

Fig. 6.9 (c) and Fig. 6.9 (d) show results with random outliers added to the second image. The method is quite robust against both outliers and image noise for the shorter baseline but the performance degrades rapidly for the larger baseline.

### 6.3.7 Up2p with Higher-order Geometric Constraints

**Simulation Setup**

The camera setup in this case was similar to the 3P1 case to ensure an upright camera with rotation only about the vertical axis. For this problem, we also simulated multiple motions for the results shown in Fig. 6.10 (c). To simulate two motions, for each problem instance, a random rotation about the vertical axis and a random unit translation vector were generated and applied to half of the 3D points before projecting them using the global camera model. Gaussian noise was then added to all the projected points as usual.

**Noise Sensitivity**

Fig. 6.10 (a) shows that for the single motion case, this problem has much better noise handling characteristics than the P3P model. For the dual-motion case in Fig. 6.10 (c), the noise robustness is poorer but the algorithm still shows good performance even with the combination of two motions and image noise.
Figure 6.10: Simulation results for the Up2p Problem. Average matching accuracy across 100 randomly generated instances of the Up2p problem ($n = 10$) for different amounts of Gaussian noise. The solid and dashed curves represent the sparse and dense tensor power-iteration algorithms [Duchenne et al., 2011]. In (a) and (b) the data points all followed a single motion; in (c) the 10 points were split in two sets of 5 points each following a different motion. In (b), we also show performance with 50% and 100% additional 2D points as outliers for the single motion case.

Performance with Outliers

Fig. 6.10 (b) shows results with random outliers added to the image and we note that this method is more robust than the P3P problem in this evaluation as well.
6.3.8 Comparison with Voting

Voting directly in the parameter space of the model being fit is another approach to address geometry problems and is typically employed for problems with fewer than 4 parameters e.g. for detection of lines (2-parameter) or circles (3-parameter) in an image. Therefore, we discuss the performance of a voting-based approach in the context of the Up2p problem which has 4 parameters. We generated 100 instances of the Up2p problem with the same simulation settings as above (but with a single motion) and varied the noise as $\sigma = 0.0$, $\sigma = 1.0$ and $\sigma = 5.0$ pixels. We also added 5 and 10 outliers to the image in combination with the above noise settings. Next, we used the closed-form method of [Kukelova et al., 2011] to solve for two solutions of the camera pose for each choice of a pair of points in 3D and a pair of points in 2D. Since our setting is correspondenceless, we have to pick all possible pairings of such 3D-2D two-point configurations. All the estimated poses across these samples were collected together and quantized into a 4-dimensional $(\cos(\theta), t_x, t_y, t_z)$ parameter space to generate a vote space. The bin with the maximum number of votes determines the winning camera pose. This pose is compared with the ground-truth, and the number of problem instances (out of 100) for which the correct pose was recovered was recorded as the accuracy in Table 6.1. We note that the performance of this algorithm quickly degrades with added noise or outliers. This is because by voting directly in the camera pose space, the algorithm is fragile with respect to the quantization boundaries. In addition, in a correspondenceless setting the number of votes that are expected to contribute to the correct solution are a small fraction of the total number of votes cast. For example, in the Up2p setting with $n$ 3D and 2D points (and no outliers), we can pick $\binom{n}{2} \binom{n}{2} 2!$ unique two-point hypothesis, each of which generates two pose solutions because of the quadratic equation in [Kukelova et al.,
Table 6.1: Accuracy of a voting-based method for the Up2p problem across 100 simulated instances with $n = 10$ 3D points each.

<table>
<thead>
<tr>
<th># Image Outliers</th>
<th>$\sigma = 0.0$</th>
<th>$\sigma = 1.0$</th>
<th>$\sigma = 5.0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Outliers</td>
<td>1.0</td>
<td>0.76</td>
<td>0.08</td>
</tr>
<tr>
<td>5 Outliers</td>
<td>0.95</td>
<td>0.30</td>
<td>0.03</td>
</tr>
<tr>
<td>10 Outliers</td>
<td>0.85</td>
<td>0.10</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Of these, only $\binom{n}{2}$ pairings generate a vote for the correct camera pose. Thus, only the fraction $\frac{1}{2n(n-1)}$ of the total votes contribute to the correct pose and hence any noise can lead to a dilution of this vote contribution even further. In contrast, the higher-order method does not directly compute the pose space and thus does not depend on the quantization of the parameter space.

### 6.3.9 Experiments with Real Data

We evaluate the ability of our algorithm in finding correspondences between 3D and 2D points on real data captured in the Up2p setting. The experimental setup is shown in Fig. 6.11. A random dot pattern was projected from a fixed projector on a fixed chair in a lab environment. A calibrated stereo camera pair, with camera optical axes parallel to the ground, was used to capture the images in Fig. 6.11(a) which are then rectified and processed to identify the image features corresponding to the projected pattern as shown in Fig. 6.11(b). This allowed 3D reconstruction of the pattern points leading to $n = 25$ 3D points shown in Fig. 6.11(c). The query image shown in Fig. 6.11(d) was captured after moving the camera (keeping the optical axis parallel to the ground), keeping the scene and the projector fixed. The pattern was manually detected to compute the 2D locations of the feature points leading to $n = 25$ 2D points.
Figure 6.11: **Up2p on Real Data.**
Fig. 6.12 shows the resulting $25 \times 25$ assignment matrix $X$ after the tensor matching algorithm has been run on a hypergraph with 100$m$, 200$m$, $\ldots$, 500$m$ sampled hyperedges, where $m = \frac{n!}{(n-3)!}$ is the number of permutations of the image features picking 3 at a time. We can see that with sufficient samples, the algorithm is able to estimate the correct assignment matrix. Figs. 6.13 and 6.14 demonstrate the performance of the algorithm on the same data where the first 5 and 10 3D points, respectively, have been removed. This means that the first 5 (respectively 10) feature points in the image are outliers. The final assignment matrices show that the algorithm is robust to these outliers and that it requires a larger sampling of hyperedges to achieve similar accuracy when a larger proportion of outliers is added.
Figure 6.14: Solution assignment matrix $X$ of the Up2p problem for hypergraphs with increasing number of sampled edges. The first 10 2D points are added as outliers leading to a $25 \times 15$ matrix $X$. Note that the algorithm converges to a good solution after sampling a larger number of hyper-edges relative to Fig. 6.13 due to the larger proportion of outliers.

### 6.4 Related Work

[Leordeanu and Hebert, 2005] consider the quadratic assignment problem where distances between pairs of features from two images are used to create an affinity matrix and an efficient spectral solution to solving this problem is proposed. [Cour et al., 2006] generalize their approach to allow incorporation of additional affine constraints. [Schellewald and Schnörr, 2005] address the same problem in a convex optimization framework by relaxing the discrete problem into a semidefinite program (SDP). More recently, [Zhou and De la Torre, 2013, Zhou and De la Torre, 2012] proposed deformable graph matching (DGM) for matching graphs subject to global rigid and non-rigid geometric constraints. However, they also restrict the choices of transformation to certain classes like similarity, affine and RBF non-rigid and work in the context of image to image matching.
The use of higher-order matching in the computer vision literature has focused on inclusion of constraints derived from higher-order geometric invariants like angles of triangles, cross-ratio along lines etc. This allows more robust matching between features in two images (under affine or plane projective assumptions) or between 3D point clouds. However, the case of matching between 3D and 2D features has not been addressed due to the lack of any geometric invariants between them. [Ochs and Brox, 2012] apply spectral clustering on a projected hypergraph computed from higher-order tuples of motion trajectories. Using affinities beyond just pairs of trajectories allows them to handle non-translational motion like rotation and scaling.

Many recent approaches have proposed algorithms for computing an assignment matrix given a higher-order graph encapsulating relations between tuples of features. [Zass and Shashua, 2008] approached the hyper-graph matching problem in a probabilistic setting but used certain independence assumptions to factor the model into first-order interactions. [Lee et al., 2011] proposed a random-walk approach for higher-order graph matching. [Duchenne et al., 2011] proposed an extension of the spectral power iteration method for matrix eigen-value problems to tensors and show how it can be used to solve assignment problems on higher-order graphs by expressing the hyper-edge affinities as a tensor.

A number of recent approaches have focused on the computational aspect of the higher-order matching problem. [Park et al., 2013] recently proposed the Higher Order FAst Spectral graph Matching (HOFASM) algorithm that approximates the affinity tensor used for higher-order graph matching resulting in lower memory and computational requirements. However, to exploit the redundancy in the affinity tensor, they require the existence of many tuples of feature points whose corresponding angles are very close to each other. Thus, their approach doesn’t directly apply to problems where the same kind of features are not being matched (e.g. 3D to 2D).
[Cheng et al., 2013] also focus on the computational aspects of higher-order matching by defining a compact affinity tensor, devising a sampling strategy to reduce redundancy and optimizing the power iteration method for computational efficiency. While some aspects of their approach are applicable to our formulation, in this thesis our focus is on proposing a theoretical framework to allow inclusion of geometric constraints into higher-order problems which lack geometric invariants.

6.5 Conclusions

In this chapter, we have presented a novel approach to incorporate geometric constraints modeled as polynomial equation systems into the higher-order graph matching framework. We have shown example formulations for three important geometric problems in computer vision and have shown the robustness of the approach to handle noise and outliers through extensive simulations. Finally, we have also shown that this framework allows us to handle correspondence problems with multiple motions using the same geometric constraints.

A potential direction for future work is to look at the practical aspects of applying this higher-order formulation to geometric matching problems. In our formulation, building the affinity tensor involves computation of the resultant of a large number of small fixed size matrices (e.g. $8 \times 8$ for the P3P problem). This is the most computationally intensive part of the algorithm but it is infinitely parallelizable. Therefore, we believe that distributing this computation using parallel GPU cores [Agullo et al., 2011, Krüger and Westermann, 2003] should tremendously scale down the tensor build time allowing practical applications with higher-order tensors.
Chapter 7

Discussion and Conclusions

In this thesis, we have outlined four methods for disparate view matching across the gamut of appearance and geometry. Table 7.1 revisits the relationship between these four methods in reference to the assumed geometry, features exploited and matching techniques. In relation to the assumed geometry, the first three methods operate with some restrictions on the assumed geometry. Future work can focus on relaxing some of these strict geometric assumptions by adapting the features or the matching methods. In our approach for facade matching, we assumed rectified plane geometry to facilitate exploitation of the repetitive patterns on building facades using self-similarity features. By considering features that capture texture patterns with more invariance to planar transformations, this assumption can be potentially relaxed. Similarly, there is potential to improve the geometric variation handling capability of the JSPEC approach by considering local features with better invariance properties than the DSIFT features explored in this thesis. Additionally, the matching method can also be made more invariant by exploring ways to match the spectral representation other than the MSER/SIFT-based approach outlined here.
### Table 7.1: Unification of Disparate Matching Approaches in this thesis

<table>
<thead>
<tr>
<th></th>
<th>Facade Matching</th>
<th>JSPEC Matching</th>
<th>Geometric Urban Geo-localization</th>
<th>Higher-Order Geometry Matching</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assumed Geometry</td>
<td>Rectified Plane</td>
<td>Small Transformation</td>
<td>Upright Camera</td>
<td>General</td>
</tr>
<tr>
<td>Features Exploited</td>
<td>Repetitive Texture; not-locally matchable</td>
<td>Short and long range interactions of local gradients</td>
<td>Specialized line and corner features</td>
<td>Geometric corners</td>
</tr>
<tr>
<td>Matching</td>
<td>Bag of features</td>
<td>Eigen structure of joint graph</td>
<td>Brute-force (stratified)</td>
<td>Higher-order graph with geometric constraints</td>
</tr>
</tbody>
</table>

A significant point of congruency between the facade matching and JSPEC matching approaches is in their use of dense pixel data. In facade matching, we extract $S^4$ features at a dense pixel sampling to capture individual facade signatures while also facilitating detection and segmentation of individual facade regions. Similarly, in JSPEC matching, we extract SIFT features on a dense pixel grid on both images of the input image pair and then construct a joint graph to derive the spectral representation of the images. Both of these approaches highlight the potential of exploiting dense pixel data for other matching tasks like object detection, segmentation, scene classification etc.

In relation to the matching techniques, all four methods are similar in that they all utilize information from all features from both images in one matching function. In facade matching, we employ Bayesian classification to label all query image features using all features from the reference data. In JSPEC matching, we employ DSIFT features from all locations in the two images simultaneously to derive the novel spectral representation for the images. In geometric urban geo-localization, we exploit all point-ray features in the reference 3D data and in the 2D image simultaneously to generate a potential camera pose list. Finally, in higher-order geometry
matching, we embed all 3D and 2D points in a single higher-order graph to simultaneously optimize for a valid correspondence set.

We have also explored two different aspects of graph matching in this thesis. In the higher-order geometry matching approach, we construct a graph using all potential correspondences between 3D and 2D points, compute the eigen-vectors for the affinity tensor of this graph and then quantize these eigen-vectors to derive a solution for the correspondence problem. While this is the usual manner in which graph matching has been explored in the literature, in the JSPEC matching approach, we have also examined the unquantized eigen-functions of a joint graph constructed from all the pixels in two images. By using these eigen-functions as a representation of the individual images, we have shown how useful information about matching regions can be derived. We believe that such a representation might be useful for other problems where graph matching has traditionally been used to directly estimate matching information from quantized eigen-vectors.

In Chapters 3 and 5, we have shown how disparate view matching techniques can lead to practical solutions for the important problem of image-based geo-localization. In particular, we looked at two approaches to this problem – the first using aerial imagery as reference data and the second using 3D Digital Elevation Map (DEM) as reference data. We believe that a combination of the two approaches can lead to practically usable large-scale geo-localization systems where the DEM and image-level reference data can together help resolve matching ambiguities while providing a computational benefit by filtering out potential outlier regions.

In conclusion, we believe that disparate view matching is an open and exciting area where techniques spanning novel features, geometric cues and matching methods can all work together to address practical and challenging problems while also allowing a richer application of existing computer vision techniques. In addition,
there is a need for specific datasets that allow advancement of the state-of-the-art in this area by facilitating quantitative comparisons of new matching algorithms across the appearance-geometry spectrum.
Bibliography


[Cham et al., 2010] Cham, T., Ciptadi, A., Tan, W., Pham, M., and Chia, L. (2010). Estimating camera pose from a single urban ground-view omnidirectional image and a 2D building outline map. In CVPR.


160


161


165


166


