TOPIC MODELS FOR IMAGE LOCALIZATION

by

Zheng Wang

A thesis submitted in conformity with the requirements
for the degree of Master of Computer Science
Faculty of Science
University of Ontario Institute of Technology

Supervisor: Dr. Faisal Z. Qureshi

Copyright © 2013 by Zheng Wang

Abstract

Topic Models for Image Localization

Zheng Wang

Master of Computer Science

Faculty of Computer Science

University of Ontario Institute of Technology

2013

We present a new scheme for partitioning geo-tagged reference image database in an effort to speed up query image localization while maintaining acceptable localization accuracy. Our method learns a topic model over the reference database, which in turn is used to divide the reference database into scene groups. Each scene group consists of "visually similar" images as determined by the topic model. Next raw SIFT features are collected from every image in a scene group and a FLANN index is constructed. Given a query image, first its scene group is determined using the topic model and next its SIFT features are matched against the corresponding FLANN index. The query image is localized using the location information associated with the visually similar images in the reference database. We evaluate our approach on Google Map Street View dataset and demonstrate that our method outperforms a competing technique.

Dedication

I dedicate this thesis to my parents, Hailong Wang and Shuying Fu.

Acknowledgements

I would like to express my thanks to my lab colleagues, and in particular W. Starzyk, N. Parvin, L. Zarrabeitia and M. Helala—for their support and friendship. Our discussions were most enjoyable. Special thanks to my supervisor Dr. Faisal Z. Qureshi for his support, guidance and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The ability to geo-localize an image is an important enabling capability [Agarwal et al., 2009]. Of the billions of images stored in the cloud—Flickr[1], Smugmug[2], Facebook[3], etc.—many are already geo-tagged. These geo-tagged images constitute a vital source of data for such applications that require the exact location of an image for subsequent processing. Agarwal *et al.*, for example, construct a 3D model of Rome from a set of geo-tagged images [Agarwal et al., 2009]. Many photo-sharing websites—such as Panaromio[4]—use location information available for geo-tagged images to visualize these images on a map, say, to support virtual tourism. Through virtual tourism anyone can experience a famous bazaar or a historic monument hundreds of miles away from the comfort of their home. Location information embedded in these images may also be used to help decipher business signs, road names, or other textual information that appear in these images. There is also a lot of interest in the ability to automatically curate personal photographs. A common scheme there is to organize pictures in spatially oriented groups, which is a natural way for viewing travel photos.

While it is trivial to find the location at which an image is taken using a Global Positioning System (GPS) device. GPS information is not always available. For example, a GPS device

[1]http://www.flickr.com/. Accessed on 03-July-2013
[2]http://www.smugmug.com/. Accessed on 03-July-2013
[3]https://www.facebook.com/. Accessed on 03-July-2013
[4]http://www.panoramio.com/. Accessed on 03-July-2013

might not be present at the time the image was taken.[5] Furthermore, a GPS device may not work under all circumstances. For example GPS devices do not work indoors or in dense urban centers dominated by tall skyscrapers [Kleusberg and Langley, 1990]. It is fair to say that thousands of images without any location information are taken every day by people all over the world and billions of images lacking any spatial information already exist on the World Wide Web (WWW). Consequently, it is worthwhile to study techniques to geo-localize images without relying upon the existence of GPS data.

1.1 Exploiting Visual Similarity to Localize Images

A promising scheme for geo-localizing images is to leverage existing geo-tagged images. The idea is to exploit visual similarities between the image in question and existing geo-tagged images. The geo-tagged image that appears most "similar" to the query image is used to estimate the location of the query image. Based on this key idea, Zamir and Shah propose an image localization scheme capable of using low-level image features to geo-localize an image using Google Maps Street View data [Google, 2013, Zamir and Shah, 2010].

Finding visually similar geo-tagged images to geo-localize a query image is akin to content based image retrieval. The primary step here is to find the set of visually similar geo-tagged images from the reference set and use these to assign the most likely location to the query image. Given that we are dealing with geo-tagged data comprising hundreds of thousands of images per city, any proposed scheme must scale gracefully. Currently the dominant approach is to organize the low-level features—say Shift Invariant Feature Transform (SIFT) [Lowe, 2004] or some variation thereof—computed from the reference geo-tagged data into an index that supports fast nearest neighbor queries. Such an index, for example, can be constructed using Fast Library for Approximate Nearest Neighbors Search (FLANN) [Muja and Lowe,

[5]We note that more and more GPS-enabled cameras are coming to market. It may be the case that in the future most of the cameras will have a built-in GPS device.

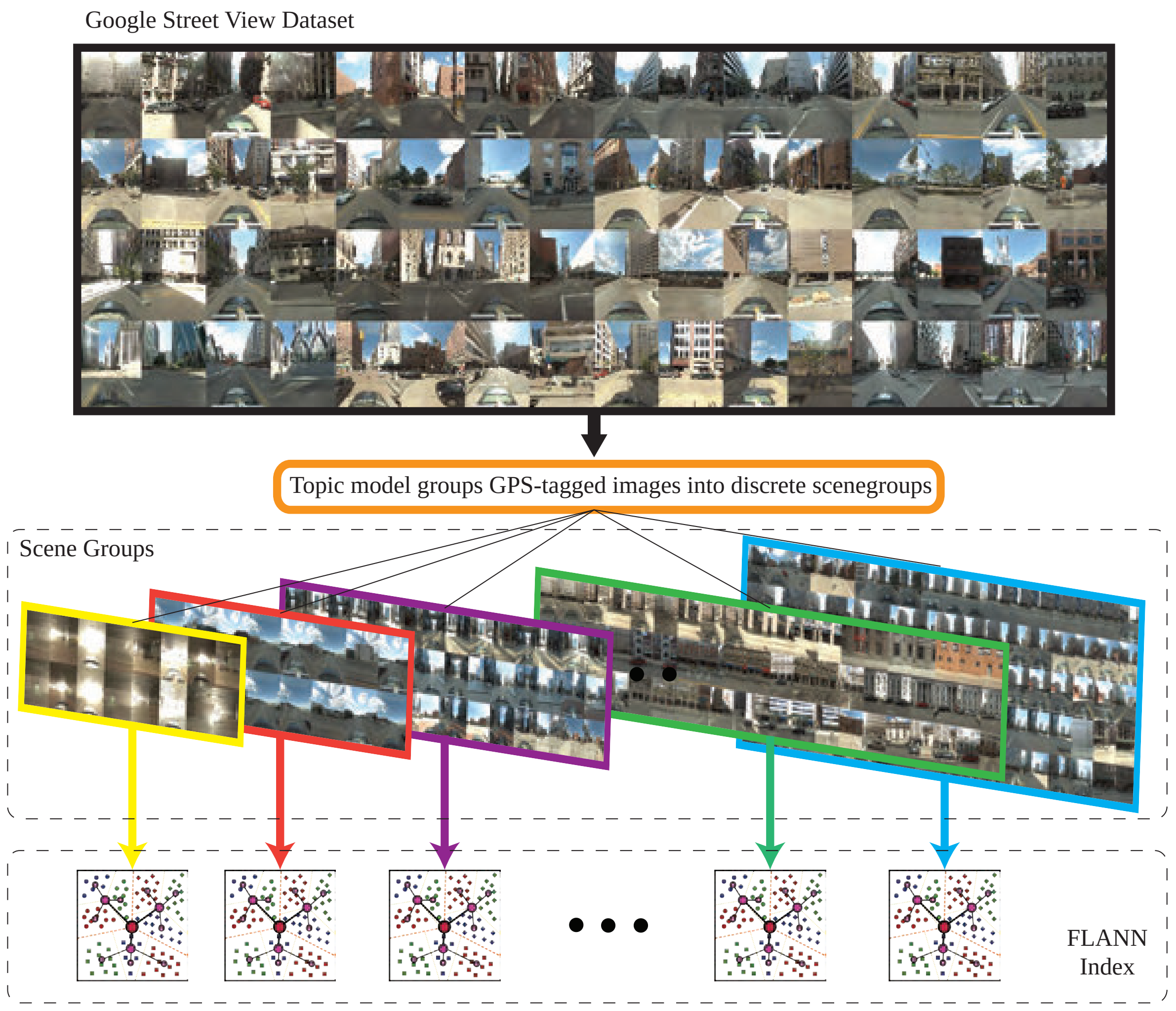

Google Street View Dataset
Topic model groups GPS-tagged images into discrete scenegroups
Scene Groups
FLANN
Index

Figure 1.1: Reference Database Processing. A sampling of images taken from Google Maps Street View Dataset for Pittsburg, PA, which serve as our geo-tagged reference database. The reference database is partitioned into scene groups and a SIFT FLANN index is constructed for each scene group. Partitioning into scene groups is carried out by learning a topic model over the reference database. Each scene group consists of visually similar images as determined by the topic model learnt over the entire reference dataset.

2009]. Features computed from the query image are then matched against those stored in the index to identify a set of images that are "visually similar" to the query image. The locations associated with the set of visually similar images are used to estimate the location of the image in question.

In our experience constructing a single index that contains features from every geo-tagged image in a reference database (say Google Maps Street View) does not scale. An obvious solution to address this scalability limitation is to divide the reference database into different sets and construct an index for each set. Features from the query image can then be processed simultaneously at each index, improving query times and addressing scalability issues. In this paper, however, we have taken a different approach. We begin by learning a topic model for our reference database. Topic model is used to divide the reference dataset into *scene groups*. Images within a scene group have similar visual characteristics as determined by the topic model. Next for each scene group we build a FLANN index over SIFT features extracted from the images belonging to that scene group. Given a new image that needs to be localized, we first infer its scene group and then use the corresponding FLANN index to determine the set of visually similar geo-tagged images. This set is then used to estimate the location of the query image. Our approach is able to achieve a significant speed increase over the technique described in [Zamir and Shah, 2010] while achieving similar localization accuracy.[6]

The reference dataset processing phase of the proposed method is depicted in Figure 1.1, and the query processing is depicted in Figure 1.2. Our reference geo-tagged dataset comprises 50,000 images from the Google Maps Street View Dataset. The images cover the high-lighted region of the city of Pittsburgh, PA shown in Figure 1.3. Topic models initially appeared for statistical analysis of textual documents [Papadimitriou et al., 1998]. Recently these have been applied to analyze images and videos. Specifically topic models have been used for

[6]On a philosophical note, our approach is motivated by our observation about how an individual might geo-localize an image? When presented with an image a human may look for landmarks that might help him narrow the search space. For instance, if presented with an image showing Toronto's CN Tower or Agra's Taj Mahal, the individual will immediately focus his attention to other images of Toronto or Agra, as the case may be. Scene groups, perhaps, mimic this behavior.

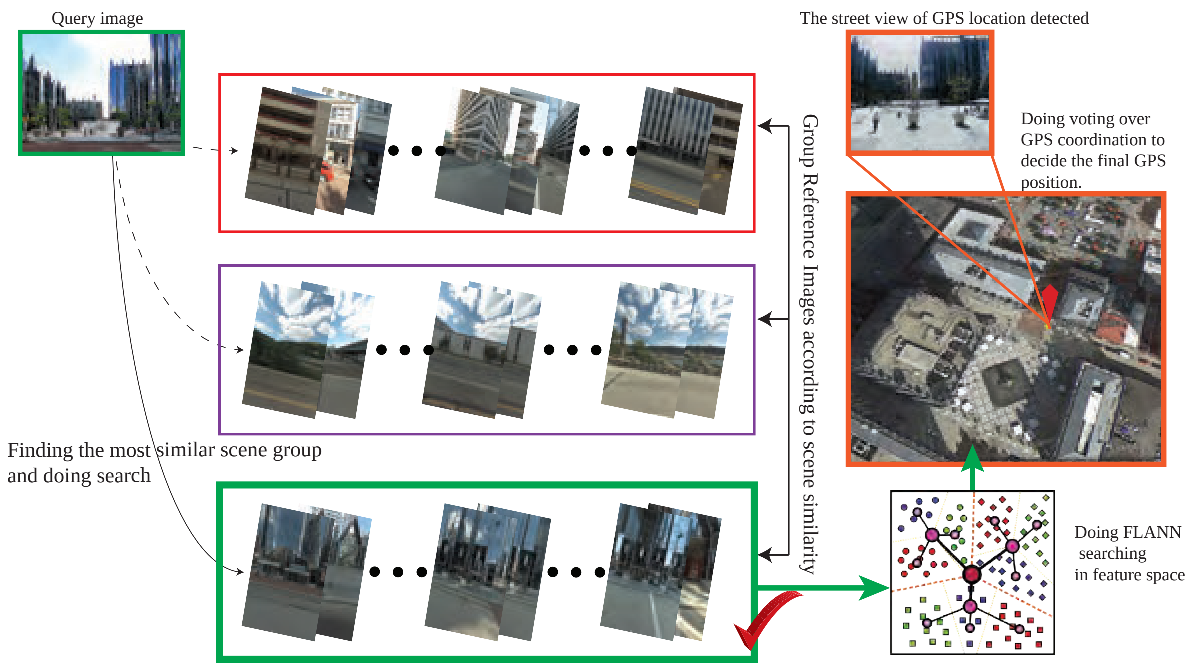

Query image
The street view of GPS location detected
Doing voting over GPS coordination to decide the final GPS position.
Group Reference Images according to scene similarity
Finding the most similar scene group and doing search
Doing FLANN searching in feature space

Figure 1.2: Query image localization. Given a query image, topic model analysis is used to identify the most relevant scene group. Next SIFT features from the query image are matched against the FLANN index for this scene group to identify the set of visually similar geo-tagged images. This set is used to estimate the location of the query image.

image annotation [Feng and Lapata, 2010, Putthividhya et al., 2010], scene classification and modeling [Fei-Fei and Perona, 2005, Bosch et al., 2006, Quelhas et al., 2005a] and image retrieval [Hörster et al., 2009]. To the best of our knowledge, ours is the first application of topic models to image localization. We have compared our approach against the method proposed by Zamir and Shah in [Zamir and Shah, 2010]. Our method outperforms their scheme in terms of query processing times, while achieving similar localization accuracies. We have also compared our method against an image localization method that relies upon vocabulary trees for image matching and our method outperforms the vocabulary tree based localization method.

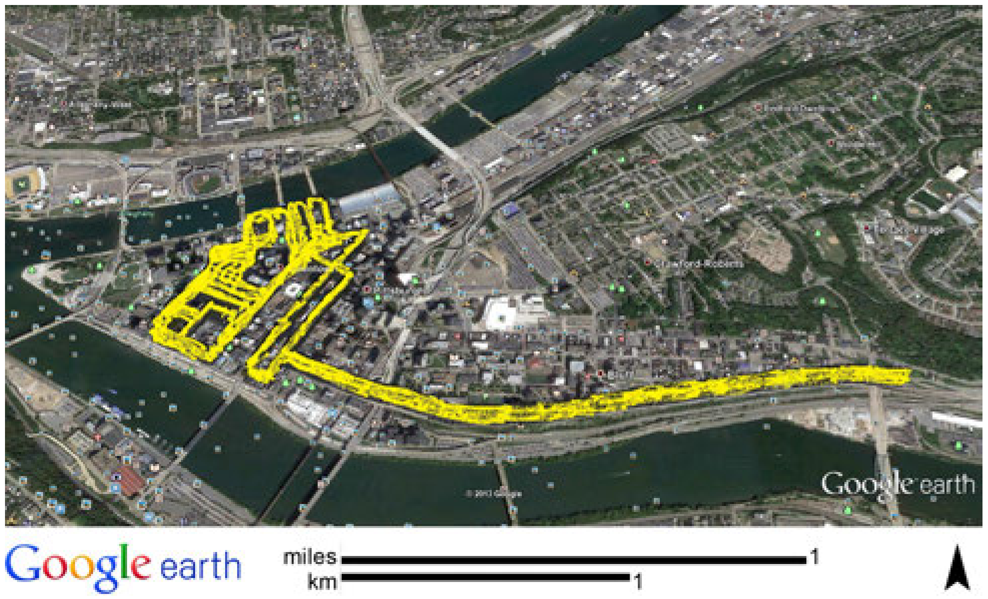


Figure 1.3: Google Maps Street View dataset for Pittsburgh, PA, which serves as our reference dataset, covers the area high-lighted above. Currently we are not able to localize images taken outside of this area. Similarly we cannot localize indoor images taken within this area.

1.2 Contribution

This thesis makes the following contributions:

1: To the best of our knowledge, ours is the first attempt to employ topic models for image localization. Specifically, we use topic models to divided the geo-tagged reference dataset into visually similar groups. When presented with a query image, the topic model analysis is used to select the most promising group for searching for visually similar geo-tagged images. Existing approaches group geo-tagged reference datasets using either location information embedded in the images or low-level visual features. Neither approach is used to restrict visual similarity search to a subset of the geo-tagged reference dataset.

2: Most existing image localization methods are concerned with improving localization accuracy. We have instead focused on performance. Specifically, we have taken a state-of-the-art appearance-based image localization technique proposed by Zamir and Shah and demonstrated how to improve the performance and the scalability of their approach by using topic models. Our method is roughly 4.5 times faster than the competing approach of Zamir and Shah, while achieving comparable accuracy.

3: Vocabulary trees have recently been used for content based image search [Nistr and Stewnius, 2006]. This suggests that we can use vocabulary trees for finding geo-tagged images that are visually similar to the query image. We have compared our approach with a vocabulary tree based image localization scheme. Our method achieves 94% accuracy where as the vocabulary tree based method only achieves 42% accuracy. We also extended the vocabulary tree based method by developing a new scoring mechanism for visual words. Typically vocabulary tree based methods score visual words based upon their occurrence frequency. Instead we propose a new scoring method that also takes into account the spatial extent of a visual word. Using this scoring scheme, the vocabulary tree based localization method achieves 67% accuracy.

4: Part of the work presented in this thesis appeared in the Tenth Conference on Computer and Robot Vision (CRV 2013), Regina, Saskatchewan, Canada, 2013.

1.3 Overview

The remainder of this thesis is organized as follows. Chapter 2 reviews the related work. Chapter 3 describes our scheme of processing the reference database and partitioning it into scene groups. Image localization is discussed in the second half of this chapter. We present experiments and results in the following chapter. Chapter 5 concludes the thesis with a summary and directions for future research.

Chapter 2

Literature Review

In this chapter we review prior work on image localization methods that rely upon visual similarity between the query image and a set of geo-tagged images. In the second half of this chapter we will also briefly discuss topic models and their uses in appearance based image grouping.

2.1 Appearance Based Methods

Earlier attempts of appearance based image localization relied upon visible geometric structures in buildings for matching the query image to the set of geo-tagged images containing these buildings. [Johansson and Cipolla, 2002, Robertson and Cipolla, 2004, Schindler et al., 2008], for example, match building skeletons extracted from a query image to those obtained from the reference geo-tagged images. These methods work well for small image datasets containing images showing frontal views of buildings. However, it is not immediately clear how to construct an index over building skeletons (facades), so these methods do not scale to larger reference databases.

More recent appearance based image localization methods rely upon image features—such as SIFT, GIST [Siagian and Itti, 2007], etc.—to match the query image against geo-tagged

Table 2.1: A comparison of appearance based localization methods

Method	**Has index or not**	**2D/3D**	**Features**	**Front view**
[Johansson and Cipolla, 2002]	No index	2D	Building facade	Yes
[Robertson and Cipolla, 2004]	No index	2D	Building facade	Yes
[Schindler et al., 2008]	No index	2D	Building facade	No
[Zhang and Kosecka, 2006]	FLANN	2D	SIFT	No
[Zamir and Shah, 2010]	FLANN	2D	SIFT	No
[Hays and Efros, 2008]	No index	2D	Multiple features	No
[Zheng et al., 2009]	No index	2D	Visual features + image meta-data	No
[Schindler et al., 2007]	Hierarchical K-means	2D	SIFT	No
[Kalantidis et al., 2011]	Vocabulary tree	2D	SIFT	No
[Li et al., 2010]	FLANN	3D	SIFT	No
[Li et al., 2008]	FLANN	3D	SIFT	No
[Snavely et al., 2006]	ANN	3D	SIFT	No
Our method	FLANN	2D	SIFT	No

reference images [Zhang and Kosecka, 2006, Zamir and Shah, 2010]. These methods first collect all features from reference images. For the sake of clarity, lets refer to the features collected from the reference dataset as *reference features*. Each reference feature is associated with the location information of the image containing this feature. When presented with a query image, these methods use the features computed from the query image to match the query image against the set of geo-tagged images. In this thesis we refer to the features computed from the query image as the *query features*. The subset of matched geo-tagged reference images is used to estimate the location of the query image. Our method belongs to this category.

Content based image retrieval underpins appearance based image localization. Consequently here we also briefly review techniques that have been developed over the last few years for the purposes of content based image retrieval. There have been attempts to combine individual features to construct higher order features to improve the image matching performance. [Hays and Efros, 2008], for example, use multiple visual features; where as, [Zheng et al., 2009] combines visual features with text data embedded in the images for image matching. *Vocabulary Trees*, first introduced in 2006, is now widely used for content based image retrieval. [Schindler et al., 2007, Kalantidis et al., 2011, Torii et al., 2011], for example, rely upon vocabulary trees to match the query image to the set of reference images or to partition reference images into visually similar groups. We compare our method with a vocabulary tree based approach for image localization and note that our method outperforms the competing approach.

In addition to image features, Three Dimensional (3D) point clouds have also been recently adapted for image localization. [Li et al., 2008, Snavely et al., 2006] organize reference features into a 3D point cloud and localize a query image by matching query features against this point cloud. These methods can not only estimate the location of the query image, but also the camera viewpoint used for capturing that image. Table 2.1 provides an overview of the existing methods that perform image localization using geo-tagged reference images.

2.1.1 Building Facades

As stated earlier localizing a query image given a reference image collection is typically done by evaluating visual similarity between the query image and reference images. The four decades of work within the computer vision community suggests that image matching is a non-trivial problem. Transient objects like clouds or pedestrians do not provide any actionable geographical information, while objects that appear everywhere such as trees, are also not useful for localizing an image. Unique architecture, on the other hand, can be very useful when localizing a query image.

Johansson and Cipolla propose an approach of estimating camera pose and location by matching building outlines visible in query and reference images [Johansson and Cipolla, 2002]. Their method first detects building outlines in the reference images and use these to construct a template for that building. A building template encodes the dominant planes of that building. These planes, by definition, contain conspicuous edges (see Figure 2.1, left). Figure 2.1 (right) shows an example of a building template constructed for the building shown in the left. Image matching relies upon matching the lines extracted from the query image against the building templates stored in the reference database.

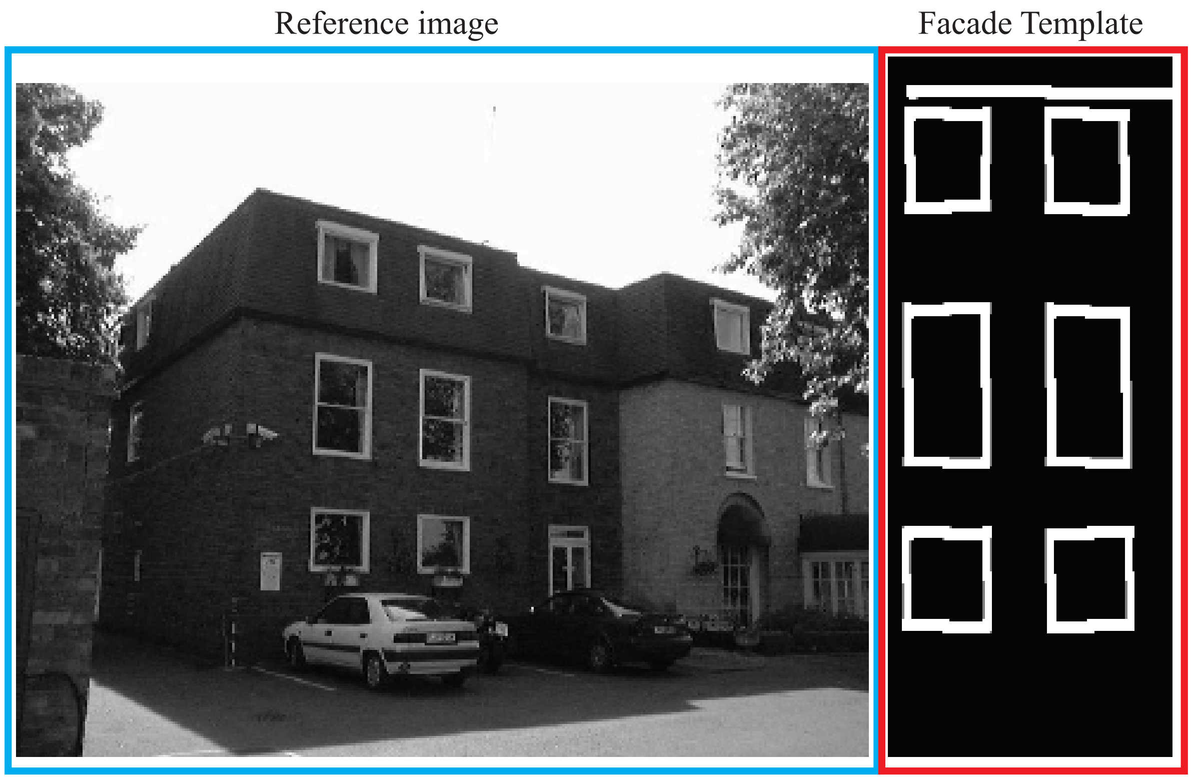


Figure 2.1: The method proposed in [Johansson and Cipolla, 2002]. (Left) the reference image and (right) the corresponding building outline template. *Image courtesy of [Johansson and Cipolla, 2002]*

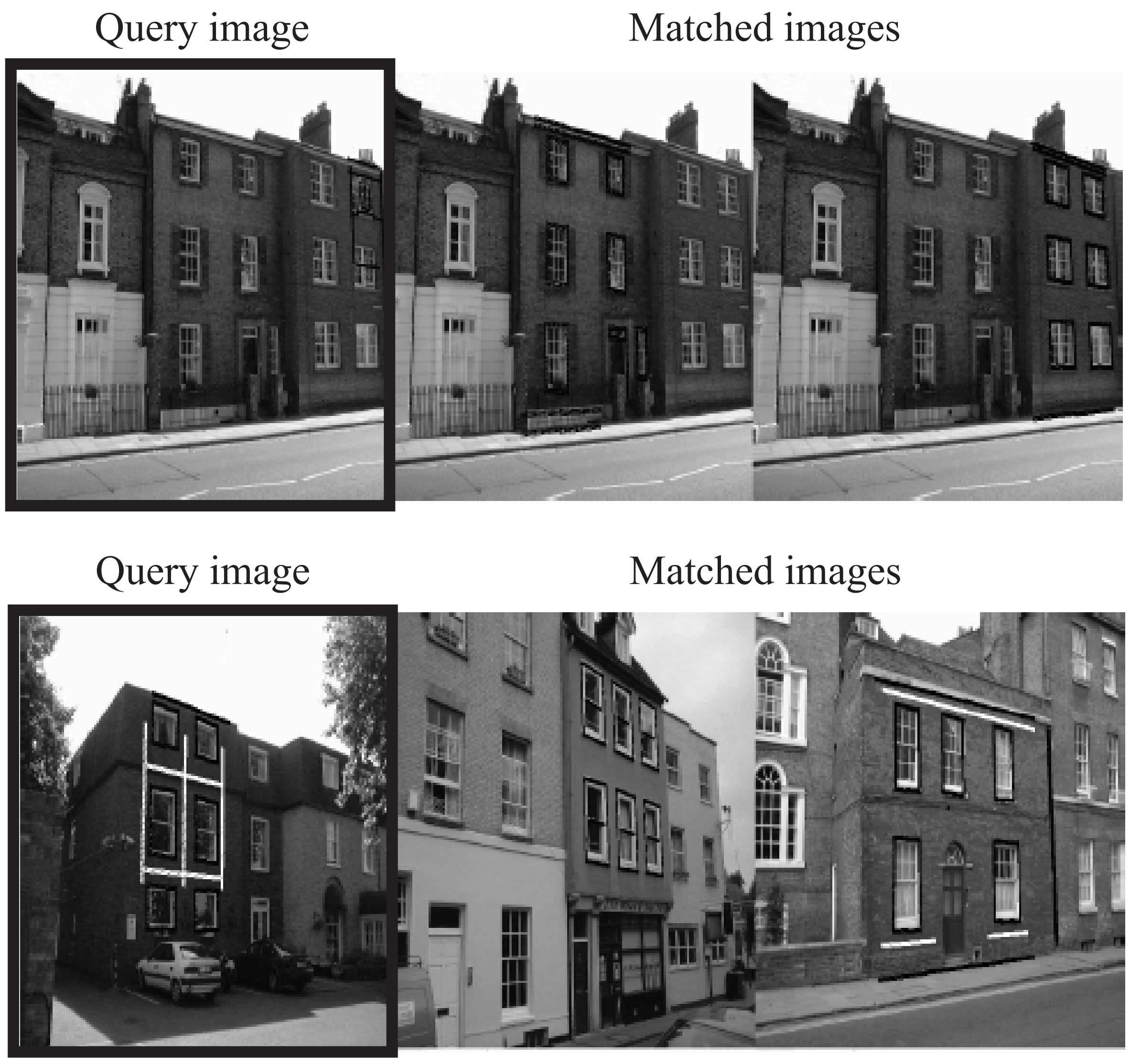


Figure 2.2: Query results in [Johansson and Cipolla, 2002]. Given a query image, the method proposed in [Johansson and Cipolla, 2002] finds the reference images having similar camera viewpoints. *Image courtesy of [Johansson and Cipolla, 2002]*

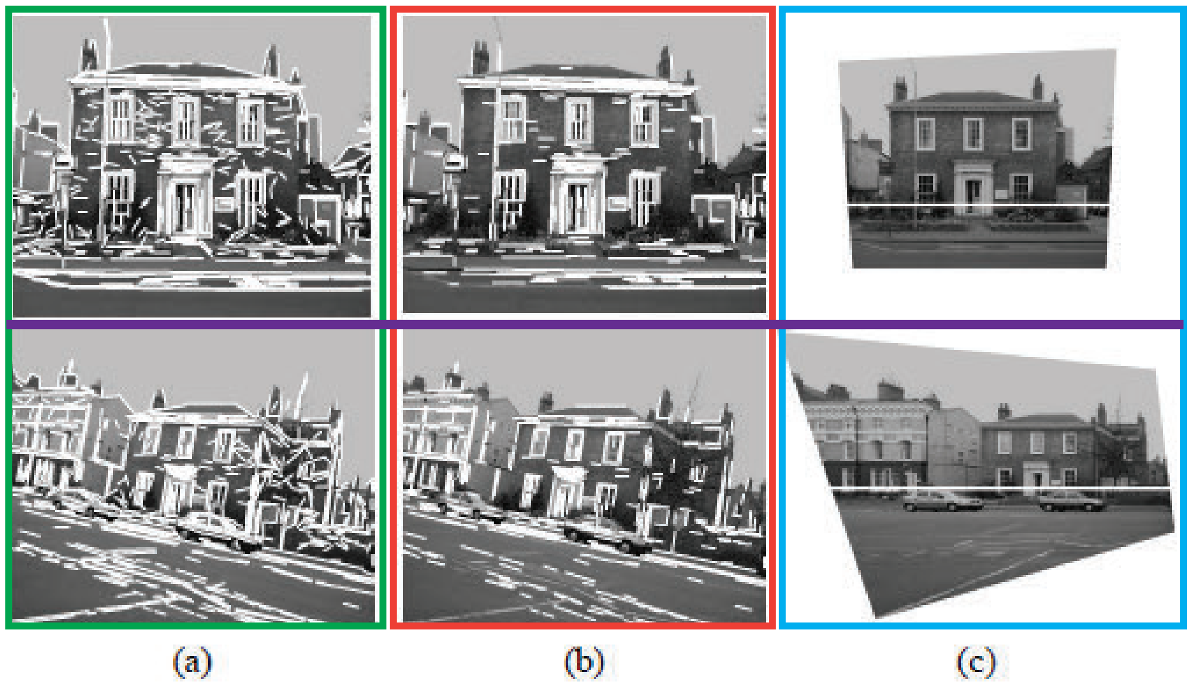


Figure 2.3: Obtaining a canonical view in [Robertson and Cipolla, 2004]. (a) Straight line segments are detected from an image. (b) After filtering, line segments associated with horizontal and vertical vanishing points are kept. (c) The image is transformed into a canonical frame using a rectified view of the dominant plane. *Image courtesy of [Robertson and Cipolla, 2004]*

Given building templates and a query image, Johansson *et al.* estimate the camera calibration matrix that best maps a template to the query image. The template that has the best match is chosen. Figure 2.2 shows some samples of query results. Johansson *et al.*'s method utilizes building facades, including windows and doors, to infer a geometric relationship between a query image and reference images. The geometric relation computed thus defines a sort of visual similarity between the images. Their method is adversely affected by camera rotations. A possible solution to this issue is to eliminate such rotations in reference images before constructing geometric parameters.

Robertson and Cipolla [Robertson and Cipolla, 2004] improve Johansson *et al.*'s method by extracting canonical views from original reference images in order to eliminate the effects of camera rotations. They assume that query images and reference images mostly contain frontal views of the buildings, so the camera orientation can be estimated by vanishing points associated with the horizontal and vertical boundaries of the buildings. Given building outlines and vanishing points in an image, Robertson *et al.* determine the orientation of the camera and further transform the original image into a canonical frame. Figure 2.3 shows an example of obtaining a canonical view from an input image. First line segments in an image are detected. Next the line segments that are not associated with the horizontal or the vertical vanishing points are discarded. The dominant plane of the building is constructed using the line segments that survive the aforementioned filtering step. Lastly the image is transformed to get the canonical view of the building.

These methods work well for smaller datasets containing images that show frontal views of buildings. Furthermore the query image should also contain the (near) frontal views of one or more of the buildings that appear in the reference dataset. In summary, Johansson *et al.*'s method use building facade templates to estimate camera position and orientation. Robertson's method further improves Johansson *et al.*'s method by eliminating image rotations. Both methods have strict image view limitations that query images must contain nearly frontal views of buildings viewed in the reference images for a successful match.

Building facades typically exhibit geometric texture patterns in the form of a grid. Schindler *et al.* [Schindler et al., 2008] exploit building lattice patterns by using "wallpaper patterns" [Liu et al., 2004]. Their method automatically localize an image in man-made environments by matching against texture models constructed from buildings in that environment. To construct a building texture model, they utilize a variation of the RANdom SAmple Consensus (RANSAC) based planar grouping method to detect perspectively distorted lattices of feature points. Given a point dataset extracted from a building image, their method constructs a projection matrix that maps the image points to a lattice space coordinates. Based on the dataset in the lattice space, a building texture model is learned. At the search time, a query image is matched against the database containing building textures. The result of this matching is a set of visually similar textures. The geographic location of the buildings corresponding to these textures are used to localize the query image. Figure 2.4 shows an example of the 3D database of textured building

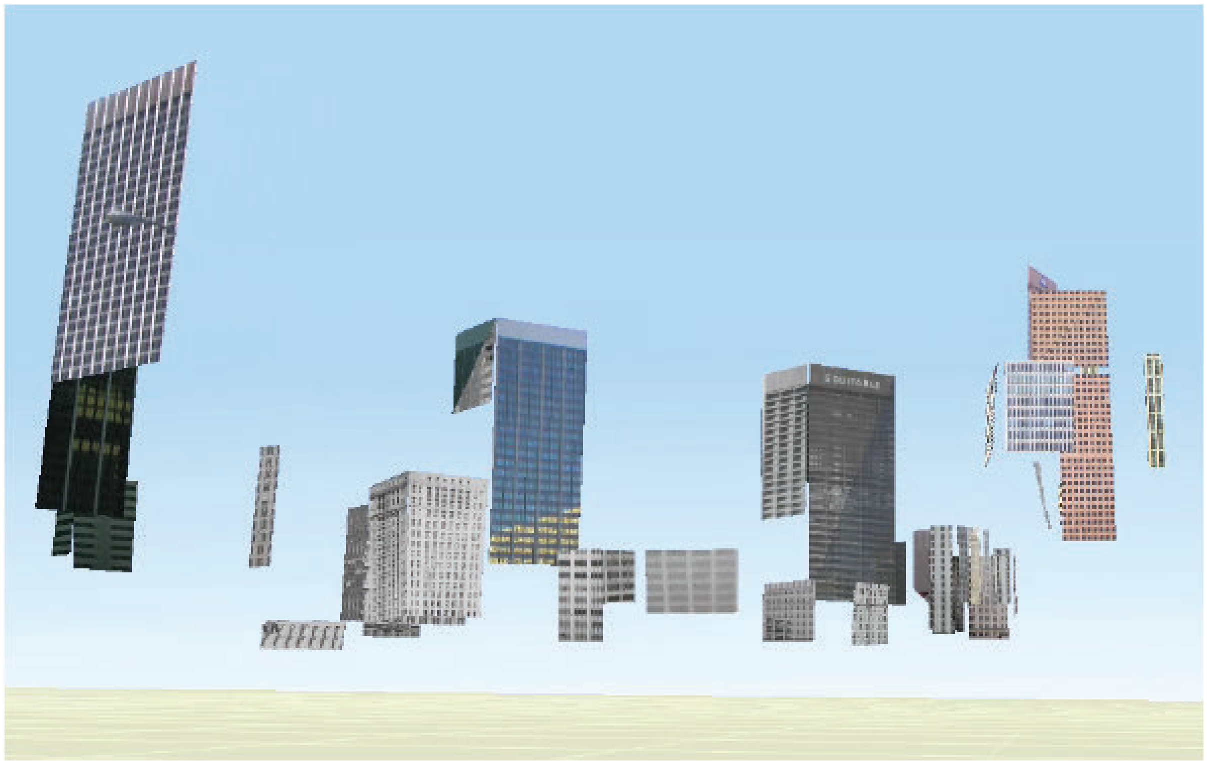


Figure 2.4: 3D database of textured building facade in a Major city used in [Schindler et al., 2008]. Each texture building model is associated with a geographic tag. Given a textured 3D city model database, such method matches a query image against the texture database and finds the most likely 3D textured building model. The query image's location is defined as the location of the corresponding building model. *Image courtesy of [Schindler et al., 2008]*

facade in their work.

The methods mentioned above evaluate image similarity by either comparing camera parameters or using probability models. These methods work well for small image datasets but may not handle a larger dataset. Since without an efficient index, these methods resort to exhaustive search when matching the query image against the geo-tagged reference images.

2.1.2 Visual Features

As stated earlier, we propose an appearance based technique for query image localization. The key ability here is to match the query image against the reference database to identify one or more visually “similar” images. Image features that are invariant to illumination, rotation and scale are routinely employed to determine the degree of similarity between two images. The common intuition being that the features from image 1 will match a larger number of features from image 2 if the two images are more similar. In this work we use Scale Invariant Feature Transform (SIFT) for image matching [Lowe, 2004].

SIFT is perhaps the most widely used image feature for image matching and object recognition tasks. Unlike, for example, Speeded Up Robust Feature (SURF), SIFT exhibits stronger stability under a variety of image transformations—scale, rotation, illumination, etc. [Juan and Gwon, 2009]. Our choice of using SIFT features for calculating visual similarity between two images stems from its robustness to viewpoint changes. Remembering that our query images are taken from very different viewpoints than the geo-tagged images stored in the reference database. We now briefly introduce SIFT features.

Scale Invariant Feature Transform

SIFT feature extraction consists of four steps: 1) scale-space extrema detection, 2) keypoint localization, 3) orientation assignment, and 4) keypoint descriptor construction. We refer the kind reader to [Lowe, 2004] for a detailed description of SIFT. Figure 2.5 illustrates SIFT

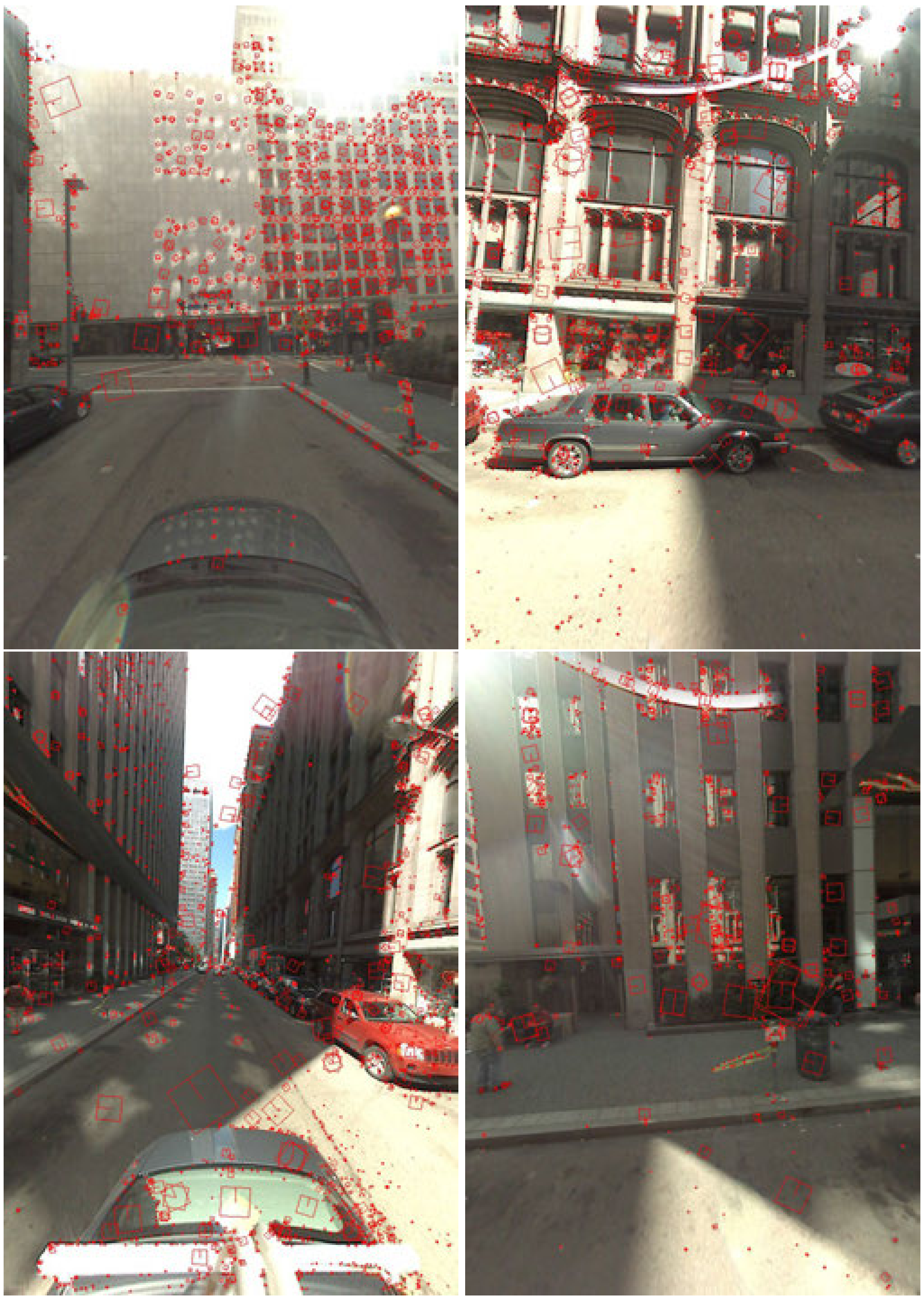


Figure 2.5: SIFT features. Four Street View images taken at the same location in four cardinal different directions.

features extracted from four reference images taken at the same location in four different directions. The centers of the red rectangle indicates the location of the SIFT features; where as, their sizes indicate the scale of these features. The orientations of these rectangles refer to the orientation of the corresponding features.

Each SIFT feature is a 128 dimensional vector. Match score between any two SIFT features f_i and f_j is computed as follows:

$$score(\mathrm{f}_i, \mathrm{f}_j) = \|\mathrm{f}_i - \mathrm{f}_j\|. \tag{2.1}$$

Depending upon the application, a threshold τ is used to decide if the two features match as seen below

$$match(\mathrm{f}_i, \mathrm{f}_j) = \begin{cases} 1 & \text{if } score(\mathrm{f}_i, \mathrm{f}_j) < \tau \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

Fast Library for Approximate Nearest Neighbors

The next step is to store the SIFT features extracted from the reference database into an index that supports fast nearest neighbour queries. Such an index is crucial given the sheer number of SIFT features that we get from our reference dataset. To provide some context, our dataset generates roughly 28 million SIFT features. Clearly, naïve methods of searching and matching SIFT features are not going to work.

k-d tree is a widely used data structure for supporting nearest neighbour queries in high-dimensional spaces. Hierarchical K-means is another technique that can be used to organize features to facilitate efficient searching. In practice the actual search performance of either technique for a given dataset depends upon a number of tunable parameters.

FLANN attempts to construct an efficient index for approximate nearest neighbour queries for a given feature set. FLANN accomplishes this by identifying the algorithm (and the associated parameters) best suited to construct the aforementioned index for the feature set in question. FLANN explores the space of available algorithms—k-d tree and hierarchical K-means—along with their parameters and selects the most suitable algorithm for constructing the index for the given feature set [Muja and Lowe, 2009]. We employ FLANN to construct efficient indices over SIFT features for our purposes. For further details about FLANN we refer the kind reader to [Muja and Lowe, 2009].

Using Single Feature

As stated earlier SIFT features have been employed in the past for image matching. When matching two images, SIFT features extracted from one are matched against those extracted from the second image. Often times using using Equation 2.2 to match a SIFT feature in one image with SIFT features from the second image does not give good results. Essentially Equation 2.2 does not distinguish between the so called *weak* and *strong* features. A feature is considered weak if it matches with a large number of features in the other image. A strong feature, on the other hand, matches with, ideally, only one feature in the other image. Of course we can tune parameter τ in Equation 2.2 to control the number of matches for any given SIFT feature. It does not work in practice, however, since we need a way to automatically select τ. [Lowe, 2004] suggests an alternative that selects a strong feature as follows. Say we want to determine if a feature f is a strong feature for matching. Assume that $NN(\mathrm{f}, 1)$ and $NN(\mathrm{f}, 2)$ are two features in the second image that are closest to f as determined by Equation 2.1. Specifically, $NN(\mathrm{f}, 1)$ is the first nearest neighbour of feature f and $NN(\mathrm{f}, 2)$ is the second nearest neighbour of feature f.[1] Then feature f is strong if

$$\frac{score(\mathrm{f}, NN(\mathrm{f}, 1))}{score(\mathrm{f}, NN(\mathrm{f}, 2))} < \tau_r, \tag{2.3}$$

where τ_r is a user-supplied threshold.

Zhang *et al.* observe that the above scheme for selecting strong features for matching is not useful when matching images for localization purposes. Since this scheme does not fair

[1]This essentially means that $score(\mathrm{f}, NN(\mathrm{f}, 1)) \leq score(\mathrm{f}, NN(\mathrm{f}, 2)) \leq score(\mathrm{f}, NN(\mathrm{f}, i))$ where $i > 2$ and $i \in \mathbf{N}^{+}$. $NN(\mathrm{f}, i)$ refers to the i^{th} nearest neighbour of f.

well when matching images of buildings that contain repeating patterns. Zheng *et al.* instead proposes to use Cosine similarity for matching two features f_i and f_j:

$$cos(\mathrm{f}_i, \mathrm{f}_j) = \frac{\mathrm{f}_i \cdot \mathrm{f}_j}{|\mathrm{f}_i||\mathrm{f}_j|} \geq \tau_c, \tag{2.4}$$

again τ_c is a user-defined threshold. Using this matching scheme, Zhang *et al.* identifies the set of matched images for a given query image. The set of matched images is ranked based upon the number of actual SIFT feature matches between the query image and the individual matched images. The highest ranked match image is used to estimate the location of the query image. This process is akin to voting.

Our method is inspired by the work of Zamir and Shah [Zamir and Shah, 2010] that uses Google Street View dataset for city scale image localization. Their method collects SIFT features from the reference dataset and stores them in a FLANN index. SIFT features from the query image are matched against the SIFT features stored in the index to select a subset of matched geo-tagged images, which are used to estimate the location of the query image. A major drawback of their scheme is that it stores all reference SIFT features in a single index. Consequently their method does not scale. Our method addresses this shortcoming of their scheme.

Zamir and Shah [Zamir and Shah, 2010] note that not all query features are good candidates for image localization. Query features may come from objects—for example, clouds, etc.—that lack any geographical information. Clearly only features related to landmark buildings or objects are useful for localizing an image. The refer to the former type of features as *weak* features and the latter type of features as *strong* features. They propose a pruning scheme to distinguish strong features from weak features given by the following equation.

$$accept(\mathrm{f}_i) = \begin{cases} 1 & \frac{\|\mathrm{f}_i - NN(\mathrm{f}_i,1)\|}{\|\mathrm{f}_i - NN(\mathrm{f}_i,min(j))\|} < 0.8, \quad \forall j \leftarrow \|loc(NN(\mathrm{f}_i,1)) - loc(NN(\mathrm{f}_i,j))\| > \tau_d \\ 0 & \text{otherwise} \end{cases}, \tag{2.5}$$

where f_i is the ith query feature. $NN(\mathrm{f}_i, k)$ is the kth nearest neighbor of f_i, $loc(NN(\mathrm{f}_i, k))$ is the GPS location of the kth nearest neighbor of f_i, τ_d is a GPS distance threshold value selected manually. Equation 2.5 recognizes a strong query feature by first evaluating its geographic distance between its 1st nearest neighbor and other nearest neighbors in the reference features. If there exists a group of nearest neighbor(s) that make the geographic distance bigger than the value τ_d, then the query feature f_i is kept otherwise is eliminated. Next they select the jth reference feature with the minimum index, as the 2nd nearest neighbor and follow Equation 2.3 to compute a distance ratio in feature space, if the ratio is bigger than 0.8, then the matched reference feature is used for geographic voting, otherwise f_i is discarded.

Zamir and Shah evaluate their method by using test images downloaded from public websites. The test images have no overlap with the images present in the reference dataset. Their tests show that around 60% the test images are located within 100 meters of the ground truth. It confirms the effectiveness of their method.

Using Hybrid Features

Single features cannot capture all of the visual information contained in an image. For example, SIFT features neglect color and do not include spatial information. Therefore, there have been attempts at combining multiple features, such as color histograms, texton histograms, etc., to construct hybrid features for image matching purposes. The idea being that combining multiple types of visual features may provide a more complete description of an image, however these also require more sophisticated strategies for indexing and searching.

Hays and Efros [Hays and Efros, 2008] combine different types of features to construct a hybrid feature for representing an image. Specifically, they use tiny images, color histograms, line features, GIST descriptor, plus color and geometric information to represent an image. The hybrid feature comprising all these elements is extracted from the query image and is matched against the features from the reference database. Matching two hybrid features consists of matching their constituents parts. For example, tiny image from one hybrid feature

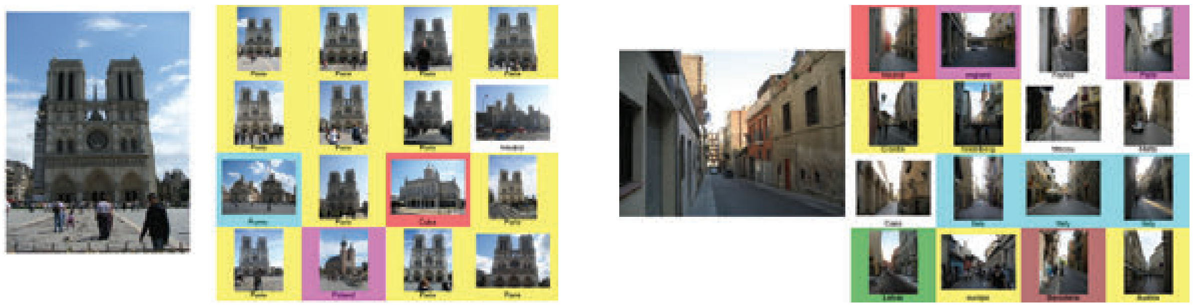


Figure 2.6: Image matching using hybrid features [Hays and Efros, 2008]. The hybrid feature proposed in [Hays and Efros, 2008] matches query images (left) to the set of reference images (right). *Image courtesy of [Hays and Efros, 2008]*

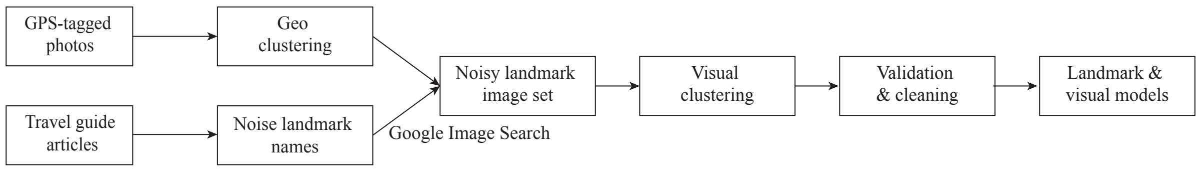


Figure 2.7: Overview of the framework proposed in [Zheng et al., 2009]. This method has two information sources, geo-tagged photos dataset and travel guide articles. The photo dataset is clustered based on photos' geo-tags. The travel articles are used to extract landmark names. Such landmark name dataset is further used to search for related landmark images using Google search engine, this step results in another image landmark dataset. Finally, landmark visual models are learned from both image datasets. *Image courtesy of [Zheng et al., 2009]*

will be matched against the tiny image from the other feature, and so on. In order to normalize the matching scores across the constituents parts, each component of the hybrid feature is assigned a weight that takes into account the variance of that component computed over the entire reference dataset.

Figure 2.6 shows image matching using the hybrid feature proposed in [Hays and Efros, 2008]. Hays and Efros do not address the issue of fast feature matching. There work is focused more on the quality of matches. Specifically, they do not provide a scheme for indexing hybrid features to support efficient nearest neighbour queries. Furthermore, matching hybrid features is also computationally more expensive.

In addition to visual features, metadata associated with images can also be used to estimate the location of an image. Zheng *et al.* use geo-tagged image dataset plus tour guides to identify a list of images containing landmarks [Zheng et al., 2009]. Images containing each landmark are grouped together and visual features for each image group are extracted and stored in a k-d tree. Each k-d tree, in a sense, represents a visual model for the corresponding landmark. Previously unseen images are localized by matching their features against the landmark models. Figure 2.7 presents an overview of their approach. Their approach is noteworthy in that it partitions the reference dataset, albeit around landmarks and using metadata that is not available to our method.

2.1.3 Image Localization using Vocabulary Trees

Visual Words

Content based retrieval systems or image matching systems rarely deal with raw (local) features directly. This is in part because such features, when extracted from a large corpus of images, exhibit redundancy. Also the sheer number of features extracted from such image corpuses easily overwhelm any attempts at searching or matching query features. A solution is to represent the image corpus with a smaller set of features by exploiting any redundancies present in the raw feature set. Visual words is one technique for constructing a (much) smaller set of representative features from the raw features set. These representative features are commonly referred to as the *visual words.*[2] Figure 2.8 illustrates visual words constructed from SIFT features. SIFT features corresponding to the same visual word are drawn in the same color.

Visual words are constructed by clustering the raw features (in our case, SIFT features) and keeping only the cluster centers. One typically has to specify *a priori* the number of visual words desired for a given situation. We use K-means to cluster SIFT features in order to construct visual words. For example, in some of our tests, we construct nearly 2 million visual

[2]Visual words have their roots in text document analysis. Just as text documents are composed of words, images are seen as consists of visual words. Specifically the visual words are defined over the entire image corpus and then any image can be described using these visual words.

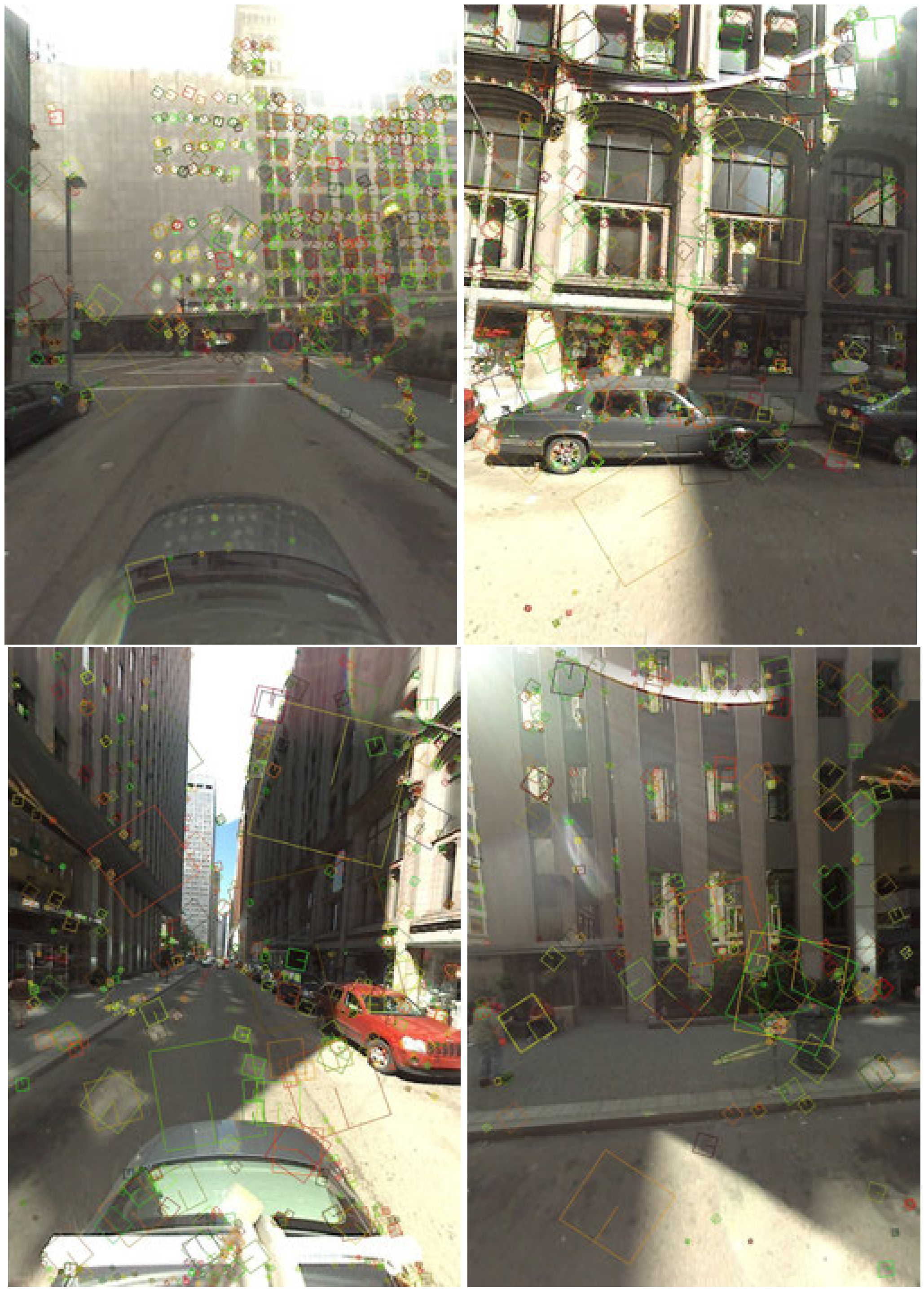


Figure 2.8: Visual words constructed from reference image dataset. SIFT features extracted from the reference dataset are clustered to construct visual words. SIFT features corresponding to different visual words are drawn in different colors, while those that belong to the same visual word are drawn using the same color.

words from 28 million raw features. It is worth noting that visual words are SIFT features representing cluster centers returned by K-means. Consequently, we can match two visual words using the same machinery that we use for matching SIFT features.

Vocabulary Tree

Vocabulary tree is a technique that relies upon visual words to match a query image against a collection of reference images. In this method, visual words are constructed from the reference images and then these words are stored in an efficient search index constructed using hierarchical K-means. Features extracted from the query image are matched against the visual words to identify the list of visually similar images in the reference dataset.

The caveat is that each visual word may be found in multiple images. Consider, for example, a visual word that is present in a large subset of images in the reference dataset. This visual word is not very useful for matching the query image to the images in the reference dataset. The solution to this problem is to use the well-understood technique from text documents where each word is assigned a weight based upon its occurrence frequency. Common words, such as ‘a,’ ‘an,’ and ‘the,’ occur frequently and are assigned low weights. Borrowing this idea, [Nistr and Stewnius, 2006] suggests to assign a weight

$$ln\left(\frac{N}{N_w}\right)$$

to each word $\mathbf{f}'_w$, where N is the total number of images and N_w is the number of images that contain the visual world $\mathbf{f}'_w$. Here higher weight is assigned to words that appear in only a few images in the reference dataset; whereas, a smaller weight is assigned to words that appear in larger number of images.

Image Localization

When presented with a query image, features extracted from the query image are matched against the visual words (from the reference geo-tagged dataset) stored in the hierarchical K-mean structure. The set of matched visual words is used to estimate the location of the query image. The process is as follows. One or more locations are attached to each visual word and each visual word votes for its location(s). As noted before each visual word also has a weight associated with it, which is taken into account during the voting process. The location that gets the highest score after the weighted voting is picked as the location of the query image.

Vocabulary trees do not fair well for image localization tasks. In part because vocabulary tree based approaches, and more specifically visual words, suffer from *polysemy* and *synonymy* [Quelhas et al., 2005b]. Polysemy refers to the fact that any visual word may be present in different scenes and synonymy means that different visual words may describe the same scene.

Grant *et al.* suggests using marginal probability of a specific location given a visual words as a means to improve the performance of vocabulary tree based image localization [Schindler et al., 2007]. Grant *et al.* examine their method using 30,000 street view images and select 278 images from them as query images. There method was able to achieve 70% recognition rates.

[Kalantidis et al., 2011] organize geo-tagged reference images into spatial groups using the location information available for these images. Kalantidis *et al.* observe that some reference images may contain multiple landmarks and should be included in several groups. By necessity these groups will be spatially close to each other. In order to create overlapping and approximately equal size groups, their method use Kernel Vector Quantization (KVQ) [Tipping and Schölkopf, 2001]. Next, for each group, their method identifies one image that contains a rigid object visible in every other image within that group. SIFT features are extracted from images within the group and are filtered based upon the SIFT features found in the representative image. The SIFT features that survive this pruning step are used to construct a vocabulary tree for each scene group. At query time, features extracted from the query image are matched using these vocabulary trees.

Instead of relying upon "voting" to estimate the location of the query image, [Torii et al., 2011] observe that any query image typically contains visual words from nearby geo-tagged reference images. Furthermore, they argue that the location of the query image is best described as a linear combination of locations of the nearby reference images.

2.1.4 3D Point Cloud

More recently Structure from Motion (SfM) techniques have been employed to construct 3D scene structure from the set of geo-tagged images. The visual features extracted from the reference images is then stored as a 3D point cloud; i.e., each visual feature is assigned a unique 3D location in the scene. This is in sharp contrast to the approaches discussed so far where each visual feature is assigned the location(s) of the image(s) containing this visual feature. The primary benefit of this approach is that it can estimate the exact camera location and pose used for capturing the query image.

[Li et al., 2010] use vocabulary trees to group visually similar images into separate groups. SIFT features are extracted from images within each group and matched against each other. Features that appear in only a few images are discarded and the remaining features are used to estimate the 3D structure of the scene using SfM techniques. These features are stored as a 3D point cloud. SIFT features extracted from the query image are matched against the visual features stored as the 3D point cloud.

Li *et al.* combine image features and 3D geometric constraints for scene summarization [Li et al., 2008]. Their method uses K-means to group reference images based on their visual similarities [Oliva and Torralba, 2001]. Their method uses GIST feature for computing image similarities. Next, for each cluster, 8 images that are closest to the cluster centre are selected and then one of these 8 images is selected as the *iconic* image for that cluster. Iconic images that are "visually similar" are connected to form a scene graph that treats individual iconic images as its nodes. The scene graph is used to construct a 3D model of the scene and the query image is matched against this 3D model.

Snavely *et al.* propose a framework for organizing an unstructured collection of images to construct 3D panoramic views from novel views [Snavely et al., 2006]. Their system can be used for content based image retrieval. Their system accepts reference images along with their camera poses, locations and orientations. First reference features are collected and matched with each other by using camera pose information. Then all matched features are further organized into a sparse 3D point cloud using SfM. Based on the 3D point cloud, all images can be registered in a common 3D coordinate system and further stitched together for a panoramic view of each scene spot. During the query time, a user draws a rectangle around target objects in a reference image, and then the system returns all relevant images that contain these objects. Their system can also annotate a reference image by finding visually similar images.

2.2 Topic Models for Image Grouping

Topic models is a class of algorithms for automatically discovering "hidden" themes in a large, unstructured corpus of documents. These first appeared for text document analysis; however, lately topic models are being increasing applied to the problem of automatic image analysis. We refer the reader to Blei *et al.* for an accessible exposition on topic models [Blei et al., 2003]. Here, however, we focus on Latent Dirichlet Allocation (LDA) algorithm for automatically discovering latent themes (referred to as topics) in a collection of images. We will also discuss how to infer the "theme" (or topic distribution) of an image given the latent themes discovered from the image collection.

LDA is easily described by its generative process. It attempts to model the unknown random process through which documents were generated from a given set of words. LDA is a part of generative probabilistic modeling, which assumes that the data is arising from a generative process that includes both hidden and observed entities. Specifically LDA specifies a joint probability over both observed and hidden random variables. Here observed variables are the words present in the documents and the hidden variables refer to the topics or themes of these documents. LDA performs data analysis by using the joint distribution over observed and hidden random variables to compute the conditional distributions of hidden variables given the observed variables. This conditional distribution is referred to as the *posterior distribution*. The problem of document analysis or of inferring the latent topic structure of a particular document is than the problem of computing the posterior distribution.

As stated earlier, topic models are used to analyze textual documents, where each individual document is simply a collection of words. The first step in applying topic models for image analysis is then to represent each image as a list of words. This is typically accomplished by learning visual words from a set of images and then using this vocabulary to describe each image. Specifically, we represent each reference geo-tagged image as list of visual words that appear in this image.

2.2.1 Image Grouping

Li and Pietro deal with image annotation using a variation of topic models [Fei-Fei and Perona, 2005]. Given several classified image groups (a training image dataset), their task is to automatically select the best fitting image group for a query image. To this end, they first construct a vocabulary from the training dataset and represent all training images as word list based on that vocabulary. After that they learn a topic model for each image group. Given a new image, they translate it into a word list using the same vocabulary and use this word list to select the best group.

Hörster *et al.* also test content-based image retrieval using topic model and Support Vector Machines (SVM) on a large-scale image repository [Hörster et al., 2007]. After learning latent topic set, they manually pick up several reference images as positive set and train SVM to distinguish similar theme images from the others. Their method works well for small image datasets but does not work well for a large-scale image archive partly because of manual reference image selection.

Chapter 3

Processing Geo-Tagged Dataset and Localizing an Image

We use Google Maps Street View dataset covering a small area of the city of Pittsburgh, PA as our geo-tagged reference image dataset. This dataset comprises around $50,000$ images. The area covered by this dataset is highlighted in Figure 1.3. The dataset is divided into image squares, a set of four images captured at one location in four cardinal directions. The distance between two adjacent image squares is roughly 12 meters. In order to get a more manageable number of images and to remove unnecessary redundancies in the dataset, we sample the dataset every 3 image squares. Consequently our reference image dataset consists of around 16,000 images covering 4000 geographic locations. These locations are roughly 36 meters apart.

3.1 Geo-tagged Dataset Processing

We process geo-tagged dataset to 1) partition it into scene groups and 2) construct a FLANN index containing SIFT features from each scene group. We now describe different steps involved in this process.

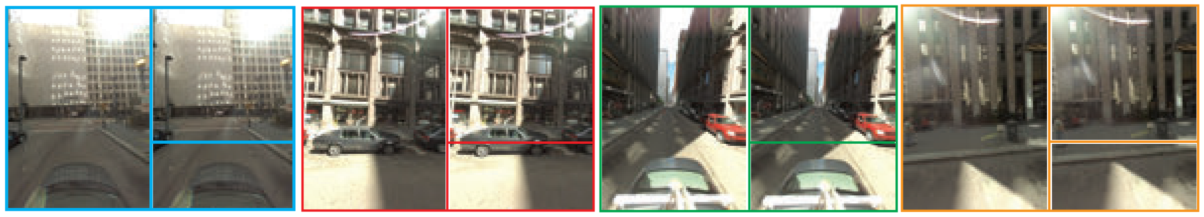
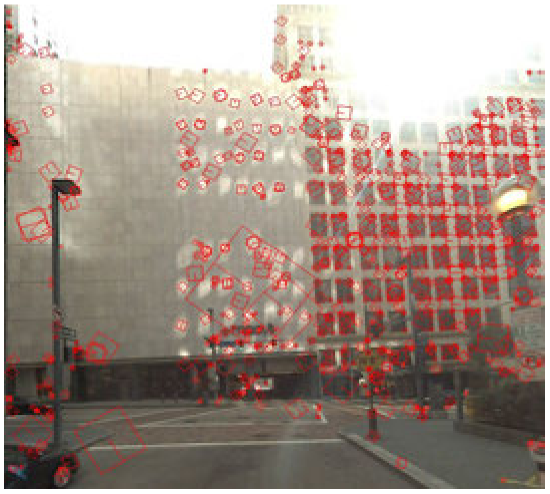


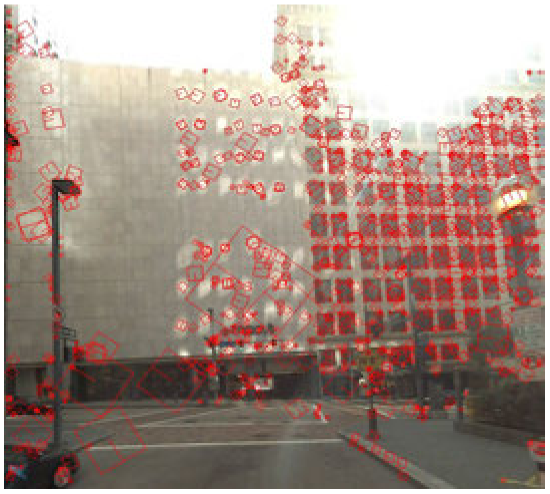
Figure 3.1: Cropping away the bottom one third of portion of the reference images. We noted that this portion rarely contains any geographically relevant information that can be used during image localization.

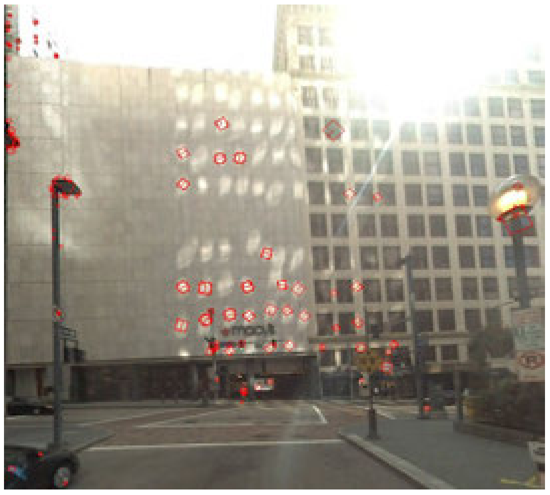
3.1.1 Collecting Visual Features

The first step is to collect SIFT features from the reference geo-tagged images. We noticed that the bottom one-third portion of most of our reference images is dominated by roads (or automobiles), containing little if any useful geographic information. We, therefore, crop our reference images and keep only the top two-thirds portion of these images (see Figure 3.1). We collect SIFT features from the top two-thirds portion of the reference geo-tagged images. It is possible to control the number and the quality of SIFT features extracted from an image by choosing different values for the following four parameters used in the SIFT algorithm:

- Number of layers at each octave (s_l): This value is used to determine the number of layers at each octave. Following Lowe's suggestions, we set this number to 3.
- Contrast threshold (s_c): This value determines whether or not SIFT features will be collected in low-contrast (or homogeneous) regions. A higher value for this parameter filters out a larger number of SIFT features.
- Edge threshold (s_e): This value determines if SIFT features that lie on edges will be kept or not.
- Smoothing (s_σ): This controls the amount of blurring performed on the image during SIFT feature detection. A larger value will result in smoother images and fewer number of SIFT features.


(a) $s_l = 1$, $s_c = 0.04$, $s_e = 5$, $s_\sigma = 1.6$


(b) $s_l = 1$, $s_c = 0.04$, $s_e = 20$, $s_\sigma = 1.6$


(c) $s_l = 2$, $s_c = 0.1$, $s_e = 10$, $s_\sigma = 1.6$

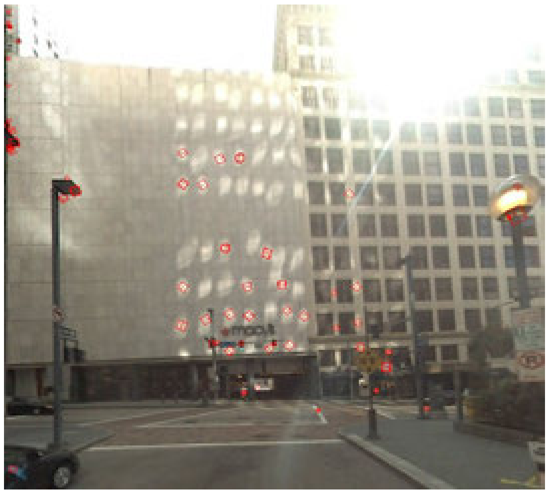

(d) $s_l = 1$, $s_c = 0.1$, $s_e = 10$, $s_\sigma = 1.6$

Figure 3.2: SIFT features extracted from a single cropped reference image for different values of the s_l, s_c, s_e, and s_σ parameters. The total number of SIFT features extracted depends upon the choice of parameter values. For example, for the image shown here, a total number of 915, 1193, 140 and 78 SIFT features were extracted for parameter values shown in (a), (b), (c) and (d), respectively.

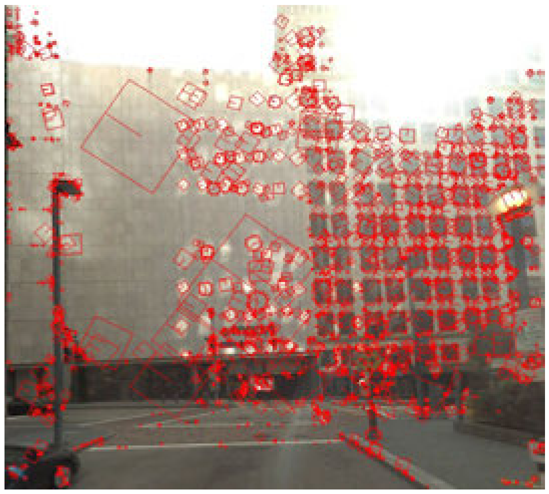


Figure 3.3: SIFT features extracted from a cropped reference image using $s_l = 3$, $s_c = 0.04$, $s_e = 10$ and $s_\sigma = 0.9$. We use these parameter values for extracting SIFT features from our geo-tagged reference dataset. For the image shown here a total of 2336 SIFT features were extracted.

Figure 3.2 shows SIFT features extracted from one reference image using different values of these parameters. Notice that these parameters effect the number of SIFT features extracted from a given image. For the remainder of this work, we will use the following values for extracting SIFT features: $s_l = 3$, $s_c = 0.04$, $s_e = 10$ and $s_\sigma = 0.9$. Figure 3.3 shows the SIFT features extracted (from the reference image used in Figure 3.2) using these values. We arrived at these values empirically. Essentially we chose to extract roughly 2000 features for each cropped reference image. The selected values seem to achieve this for all of our cropped reference images.

Each SIFT feature is associated with the location of the reference image containing it. Say feature $\mathrm{f}_{j,l}$ represents the l^{th} 128 dimensional SIFT feature extracted from image I_j. Then the location of this SIFT feature is the same as the location of image I_j. For the reference dataset used in this thesis, this process collects around 28.3 million SIFT features.

3.1.2 Constructing Vocabulary Tree

The next step consists of constructing a vocabulary tree over the SIFT features extracted from the reference image set. First we construct visual words. As stated earlier, visual words are constructed from SIFT features via K-means clustering. The cluster centers are the visual words that we need. We decided to construct around 2.8 million visual words. This number represents about 10% of the total number of SIFT features extracted from the dataset. Figure 3.4 illustrates examples of visual words constructed through clustering. Here each SIFT feature is colored based upon the visual word that is closest to it as determined by Equation 2.1, remembering that visual words are simply 128 dimensional SIFT features. In order to organize visual words as vocabulary trees, we employ hierarchical K-means clustering that is 5 level deep with a branching factor of 64. For the remainder of this thesis we will use $\mathbf{f}$ to denote SIFT features and $\mathbf{f}'$ to denote visual words.

Each visual word may appear in more than one images. In other words multiple locations are associated with each visual word. We now present a procedure to compute how many times each visual word appears in an image. We will store this information in an $N \times W$ matrix, where N is the number of reference images and W is the number of visual words.

```
Require: Image set $\{I_1, \cdots, I_N\}$.
Require: Visual word set $\{\mathbf{f}'_1, \cdots, \mathbf{f}'_W\}$.
Require: Sets of SIFT features extracted from each image. Say $F_n$ represent the set of features
  extracted from image $I_n$.
Ensure: Matrix $O \in R^{N \times W}$ that stores how many times a visual word appears in an image.
  Set all entries of $O$ to $0$.
  for Each $I_n \in \{I_1, \cdots, I_N\}$ do
    for Each $\mathbf{f} \in F_n$ do
      Find the visual word that is closest to this SIFT feature. Say this visual word is $\mathbf{f}'_w$.
      $O[n][w] \leftarrow 1$.
    end for
  end for
```

Given O, we can easily find out how many images contain a visual word. Specifically, $\sum_n O[n][w]$ is the number of images containing the visual word $\mathbf{f}'_w$. Let us define $count(\mathbf{f}'_w) =$

$\sum_n O[n][w]$. We can then assign a weight to each visual word $\mathbf{f'}_w$ as follows

$$weight(\mathbf{f'}_w) = \frac{N}{count(\mathbf{f'}_w)},$$

where N is total number of images in the reference dataset. Lets also define $occurrence(\mathbf{f'}_w)$ to be the w^{th} column of matrix O. $occurrence(\mathbf{f'}_w)$ is then a binary vector that records if a visual $\mathbf{f'}_w$ appears in a reference image or not. Specifically, visual word $\mathbf{f'}_w$ appears in image I_n iff $occurrence(\mathbf{f'}_w)[n]$ is 1. Square brackets denotes indexing.

3.1.3 Learning Topic Models

Topic models see each image as a random mixture of (latent) topics, which in turn are responsible for generating the visual words associated with that image. The goal of learning a topic model is to find the latent topic set and the associated distributions that best explain the visual words observed in each image. We employ Latent Dirichlet Allocation (LDA) generative model to learn the topic model over the aggregated set of visual words observed in our reference dataset [Blei et al., 2003].

One of the parameters that needs to be chosen *a priori* when learning a topic model given the set of visual words is the dimensionality of the topic vector. A higher dimensional topic vector can explain fine features within an image; whereas, a lower dimensional topic vector ignore finer details, focusing instead on a more abstract features. At the same time, however, a higher dimensional topic vector can suffer from over-fitting, meaning that the topic model is easily distracted by visual clutter present in an image. Such a topic model will not generalize. In this work we present results for 100, 300 and 400 dimensional topic vectors. After topic model analysis, each image is represented as a topic vector, encoding topic distributions inferred for that image. Figure 3.5 plots topic vectors for different images.

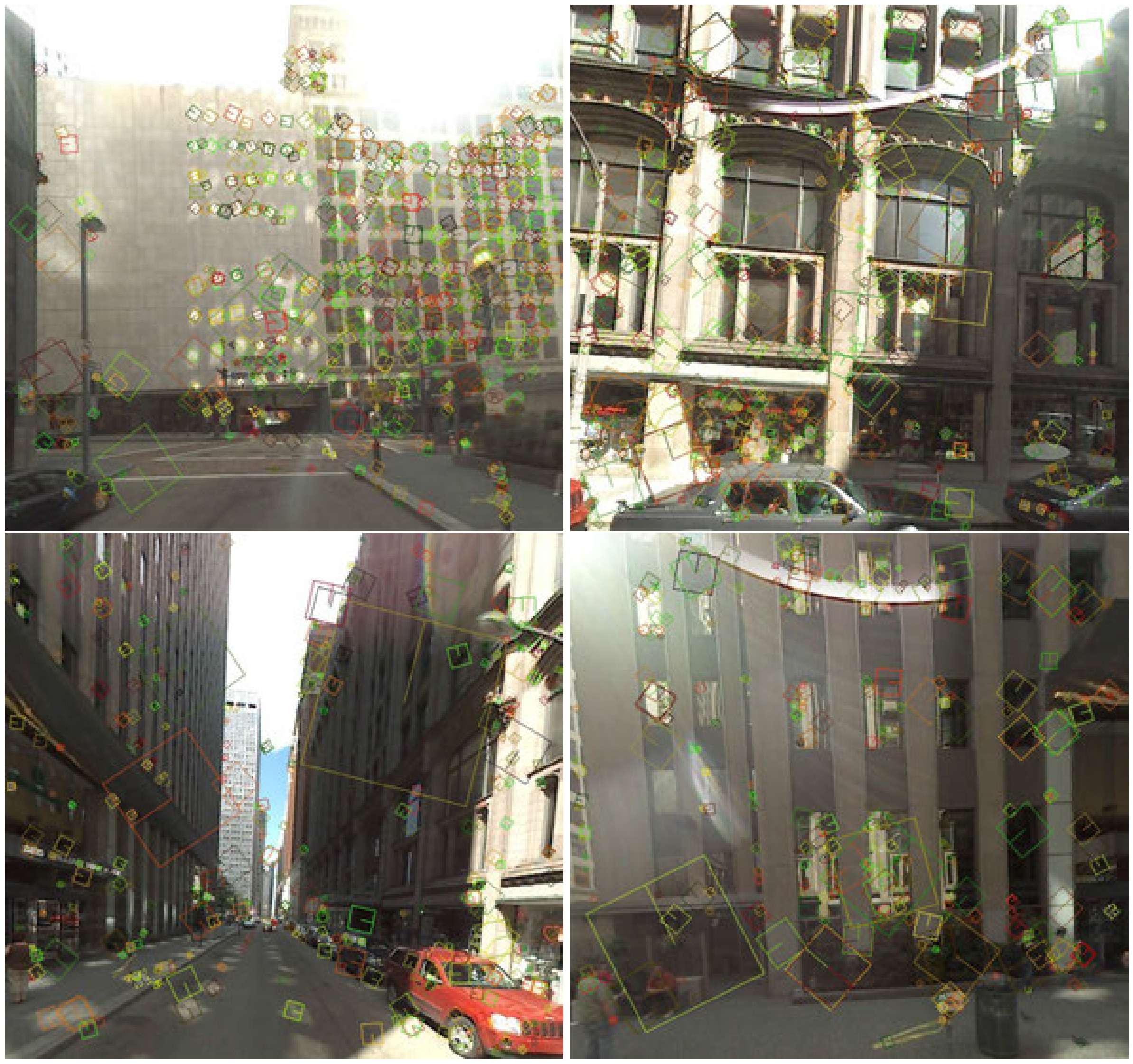


Figure 3.4: Examples of visual words detected in reference images. After collecting SIFT features from an image, we match its SIFT features against a visual word dataset constructed before. Here, we color SIFT features using their corresponding visual words.

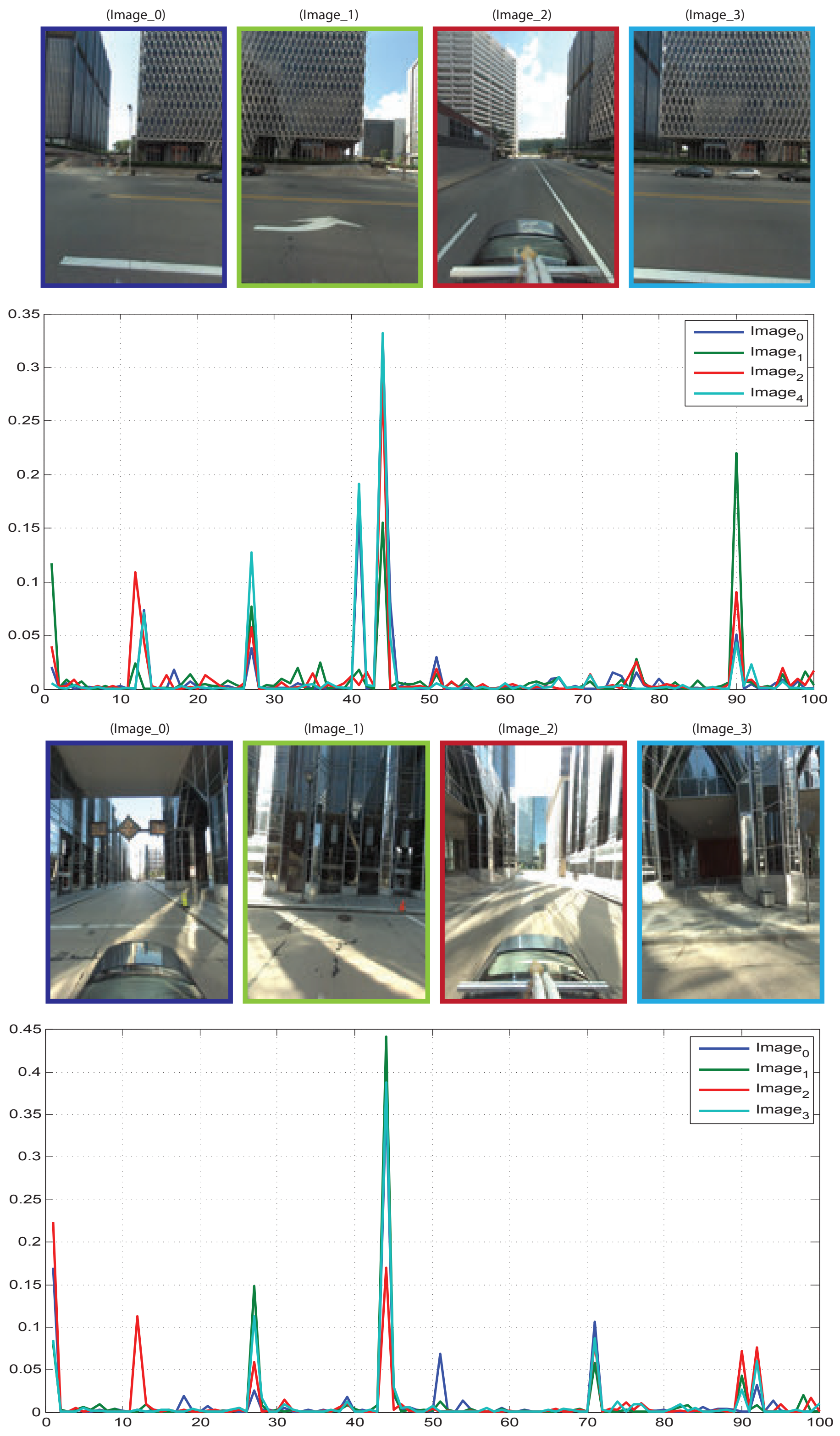


Figure 3.5: Representing images as topic vectors. Each image is represented as a 100-dimensional topic vector. Each topic vector is drawn using a different color, which matches the border of the corresponding image.

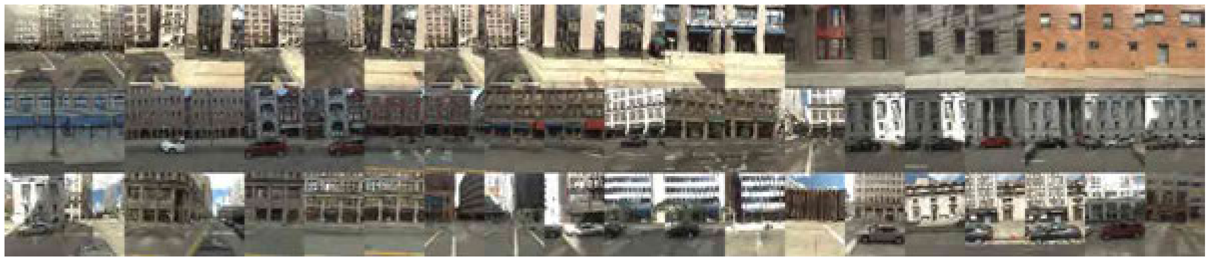
3.1.4 Scene Groups

The next step is to group images that have similar topic distributions. This is accomplished by clustering topic vectors from the geo-tagged reference dataset. Specifically we cluster topic vectors corresponding to each of the reference image into S groups. We refer to these groups as *scene groups*. The premise is that visually similar images will be grouped together since these will exhibit similar topic distributions. Figure 3.6 shows sample images belonging to three different scene groups. Notice how images belonging to a scene group share similar visual features. Notice also how scene groups depicted in Figure 3.6(b) and Figure 3.6(c) are constructed around unique buildings (or landmarks). A possible explanation for this "spatial partitioning" is that the images of a visually distinctive landmark share many visual characteristics, plus that these images are sufficiently different from all other images. Note also that the topic model fails to create a spatial partitioning for images of nondescript buildings (Figure 3.6(a)). This is to be expected.

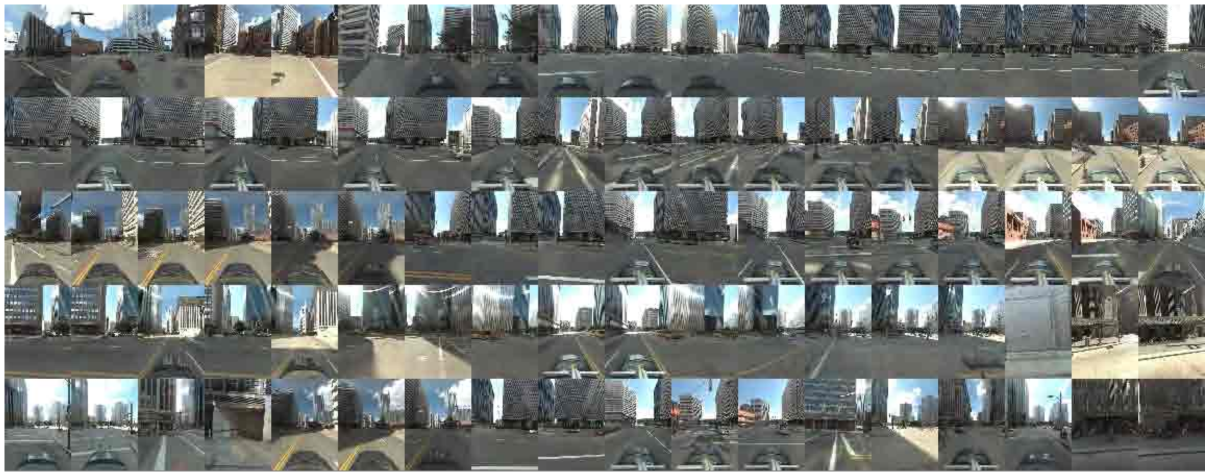
The proposed method then constructs an index for each scene group. For this purpose it collects raw SIFT features from images belonging to a particular scene group and construct a FLANN index over these SIFT features. Specifically the FLANN index for a given scene group is constructed using geo-tagged SIFT features belonging to the images in that scene group. FLANN index currently does not support online modifications; however, it does support fast approximate nearest neighbour queries.

3.2 Localizing a Previously Unseen Image

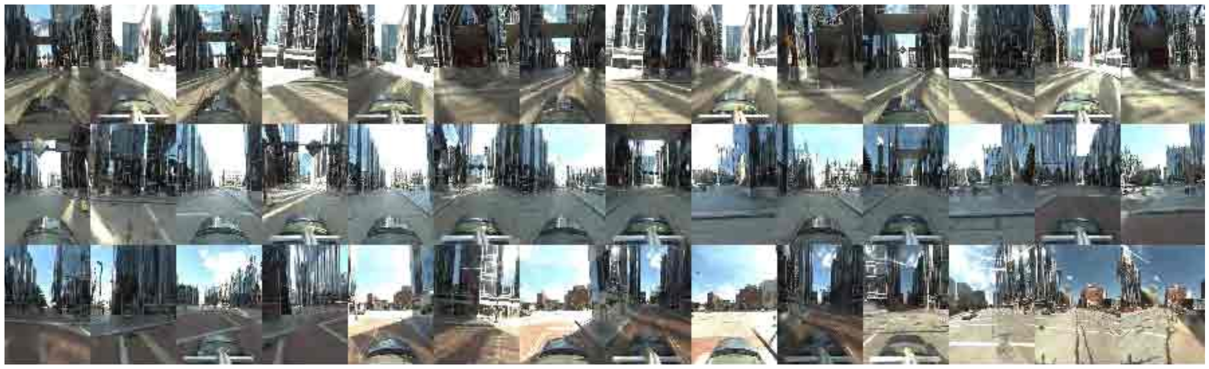
We now turn our attention to the problem of localizing a previously unseen image. We investigate two strategies for localizing an image. First, we discuss how vocabulary trees can be used for localizing an image and then we will introduce our method that uses scene groups for localizing previously unseen images.



(a)



(b)



(c)

Figure 3.6: A sampling of images belonging to different scene groups. Notice how topic model driven partitioning scheme groups visually similar images together. (a) is a generic scene group. (b) is a scene group comprising images that all see a unique building and (c) consists of images that see the famous PPG Place, Pittsburgh, PA. (b) and (c) supports the notion that scene groups are constructed around landmarks.

3.2.1 Localizing an Image using Vocabulary Trees

SIFT features extracted from the query image I_q are matched against the visual words stored in the vocabulary tree. Say $\{\mathbf{f'}_q | q \in [1, Q]\}$ is the set of matched visual words. Each visual word is associated with one or more locations. Remembering that each visual word appears in one or more reference geo-tagged images. Each visual word votes for the locations where it appears (i.e., locations of the images containing this visual word). The location of the query image is

$$locof\left(maxindex\left(\sum_q weight(\mathbf{f'}_q) \times occurrence(\mathbf{f'}_q)\right)\right),$$

where $occurrence(\mathbf{f'}_q)$ is a binary vector that stores the occurrences of the visual word $\mathbf{f'}_q$, $weight(\mathbf{f'}_q)$ is a scalar denoting the weight of the visual word $\mathbf{f'}_q$, function $maxindex$ returns the index corresponding to the largest value of a vector and function $locof$ returns the location of the reference image corresponding to the index passed to it.

We noticed that the scheme described here does not perform well. This is due to the fact that weighting scheme used here fails to distinguish between visual words that appear in a lot of images taken in close proximity to each other and visual words that appear in a lot images that are spread over a large area. The first kind of visual words contain useful geographic information and should be assigned higher weights; where as, the second kind of visual features do not contain useful geographic information and should be assigned lower weights. Say $loc(\mathbf{f'})$ represents the set of locations for visual word $\mathbf{f'}$, we measure its geographical spread by computing standard deviation of $loc(\mathbf{f'})$. We can then estimate the location of the query image using the following equation:

$$locof\left(maxindex\left(\sum_q \frac{weight(\mathbf{f'}_q)}{spread(loc(\mathbf{f'}_q))} \times occurrence(\mathbf{f'}_q)\right)\right),$$

where $spread(loc(\mathbf{f'}_q))$ denote the geographical spread of visual word $\mathbf{f'}_q$. It is computed as follows. Say $sd(loc(\mathbf{f'}))$ denote the standard deviation of vector $loc(\mathbf{f'})$, treating each of its element as a sample of a bivariate distribution. We mention bivariate as each element of $loc(\mathbf{f}')$ represent longitude and latitude values. Then $spread(loc(\mathbf{f}'_q)) = \prod_i sd(loc(\mathbf{f}'_q))[i]$, again using [] to denote indexing.

3.2.2 Localizing an image using Scene Group

When localizing a query image using scene groups, the topic model learned over the reference database is used to infer topic distribution for this new image. The process involves three steps: 1) computing raw SIFT features, 2) projecting the computed SIFT features to the visual word space constructed for the reference database and 3) using the visual words present in the query image to infer its topic distribution.

The topic distribution is then used to identify the most likely scene group for the query image. Say topic distribution inferrence represents the query image as a topic vector $\mathbf{z}_q$. Further assume that the set of topic vectors corresponding to S scene groups is $\{\mathbf{z}_1, \cdots, \mathbf{z}_S\}$. Remembering that scene groups are constructed by clustering topic vectors from the reference dataset, $\mathbf{z}_s$ is the cluster centre corresponding to the s scene group. The scene group s will be chosen for the query image if $|\mathbf{z}_q - \mathbf{z}_s| \leq |\mathbf{z}_q - \mathbf{z}_{s'}|$, $\forall s, s' \in [1, S]$ and $s \neq s'$. Next raw SIFT features from the query image are matched against the FLANN index corresponding to the most likely scene group for that query image. This process identifies the set of matched geo-tagged SIFT features.

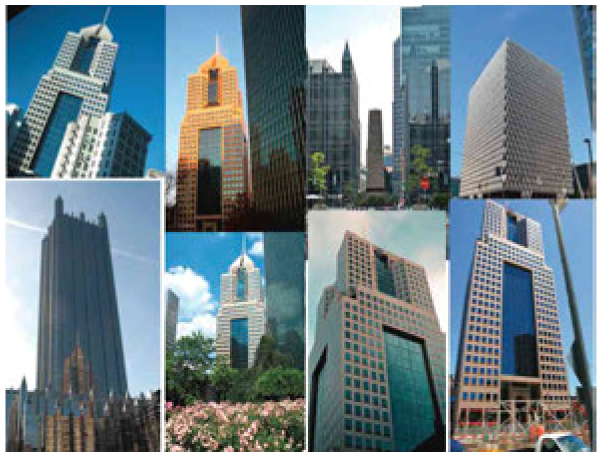
We adopt the scheme proposed in [Zamir and Shah, 2010] to prune the set of SIFT features returned after matching the FLANN index of the most likely scene group (Equation 2.5). The matched geo-tagged SIFT features that pass this test are used to assign a location to the query image through voting. Essentially each of the matched feature votes for its location and the location that has the largest number of votes is assigned to the query image.

Chapter 4

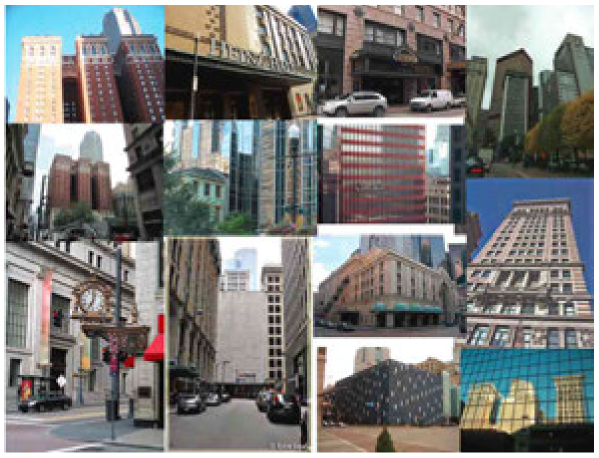
Experiments and Results

In this chapter, we focus on experiments and results conducted in this work. In Section 4.1, we compare image localization using our method that uses scene groups with Zamir and Shah approach that appeared in [Zamir and Shah, 2010]. We use the same geo-tagged reference image dataset that was used in [Zamir and Shah, 2010]. For our query images we selected 121 images with known location from the website Panoramio. These images are from the area covered by our reference datasets and these images have no overlap with our reference dataset. Figure 4.1 shows a subset of our query images. The locations of these images are known; however, these locations are not used during localization. Rather this information serves as the ground truth.

In the second half of this chapter we compare two vocabulary tree based approaches (for image localization) with the proposed approach. We will compare Nistr and Stewnius [Nistr and Stewnius, 2006] scheme for weighing visual words with the approach that we presented in Section 3.1.2. The primary difference between the two approaches is that Nistr and Stewnius approach relies only upon occurrence frequency of a visual word; whereas, the approach presented in Section 3.1.2 also takes into account the geographic extent (or spread) of visual words when assigning them weights. We also compare these approaches with Zamir and Shah scheme and with the proposed method that uses scene groups for image localization. For this set of



(a)



(b)

Figure 4.1: A sampling of test images used to evaluate our method. These images are already geo-tagged, so we have the ground truth available for these images.

tests, we randomly select 100 images from the reference dataset. A similar approach was taken in [Nistr and Stewnius, 2006]. Again the locations of the query images were not used during localization, and served only as ground truth.

For either set of tests we use geo-tagged images from Google Maps Street View dataset for Pittsburgh, PA as our reference dataset. Figure 1.3 shows the region of Pittsburgh covered by this dataset, which consists roughly $50,220$ images. Google Maps Street View dataset consists of an image square (4 images taken in cardinal directions) taken at 12 meters interval. We noticed that this dataset exhibits redundancy, so we decided to subsample the dataset by selecting every fourth image square. The subsampled dataset contains an image square roughly 36 meters apart.

4.1 Scene Groups for Image Localization

Around 28.3 million SIFT features are extracted from $16,740$ geo-tagged reference images. The raw SIFT features when loaded in the memory takes about 13 Gb. K-means clustering is used to construct $2,792,343$ visual words from the raw SIFT features. This represents roughly 10 percent of the total number of raw features. Topic model is learned over these visual words.

Dimensions	Sequential (sec)	Parallel (sec)
100	1.1	0.1487
300	3.1139	0.445
400	4.593	0.656

Table 4.1: Topic distribution inference times for a query image.

We have experimented with topic space dimensions of 100, 300 and 400. Topic distributions aggregated over all the images are then clustered to construct scene groups and FLANN indices are constructed for each of the scene group.

A higher dimensional topic space is able to capture fine features of an image; however, it may suffer from over-fitting. Additionally it takes longer to infer the topic distribution of the query image when topic space has a large number of dimensions.

Table 4.1 lists the average times it take to select the corresponding scene group for a query image. This scene group is most likely to contain the images that are visually similar to the query image. Notice that more time is needed to infer the topic distributions and select the corresponding scene group for higher dimensional topic vectors. The good news is that topic distribution inference times can be drastically reduced by using the algorithm described in [Newman et al., 2006] (Table 4.1, column 2).

Since the primary thrust of this paper is to partition the reference geo-tagged images into groups of visually similar images by employing the topic model learned over all the images. We have evaluated our approach using topic models consisting of 100, 300 and 400 dimensions. We have also experimented with varying the number of scene groups (10, 50 and 100) that we construct from the reference dataset.

We have evaluated our approach on 121 test images. The actual geographic locations of these images are known. These serve as our ground truth. Geographic locations of the test images were not used during the localization procedure. Figure 4.1 shows a subset of our test images. Note that Google Maps Street View data does not contain these images. In other words these images are previously unseen as far as our reference dataset is concerned.

37 landmark images		
Model	Accuracy	Time (s)
Method in [Zamir and Shah, 2010]	97.3%	11.4019
100 Topics / 10 groups	**97.3%**	**2.33134**
100 Topics / 50 groups	86.5%	1.6241
100 Topics / 100 groups	78.4%	1.01103
300 Topics / 10 groups	89.2%	2.05295
300 Topics / 50 groups	89.2%	1.21132
300 Topics / 100 groups	83.8%	1.1616
400 Topics / 10 groups	91.9%	3.9773
400 Topics / 50 groups	78.4%	1.0997
400 Topics /100 groups	78.4%	1.2201

Table 4.2: Localization performance comparison for landmark images.

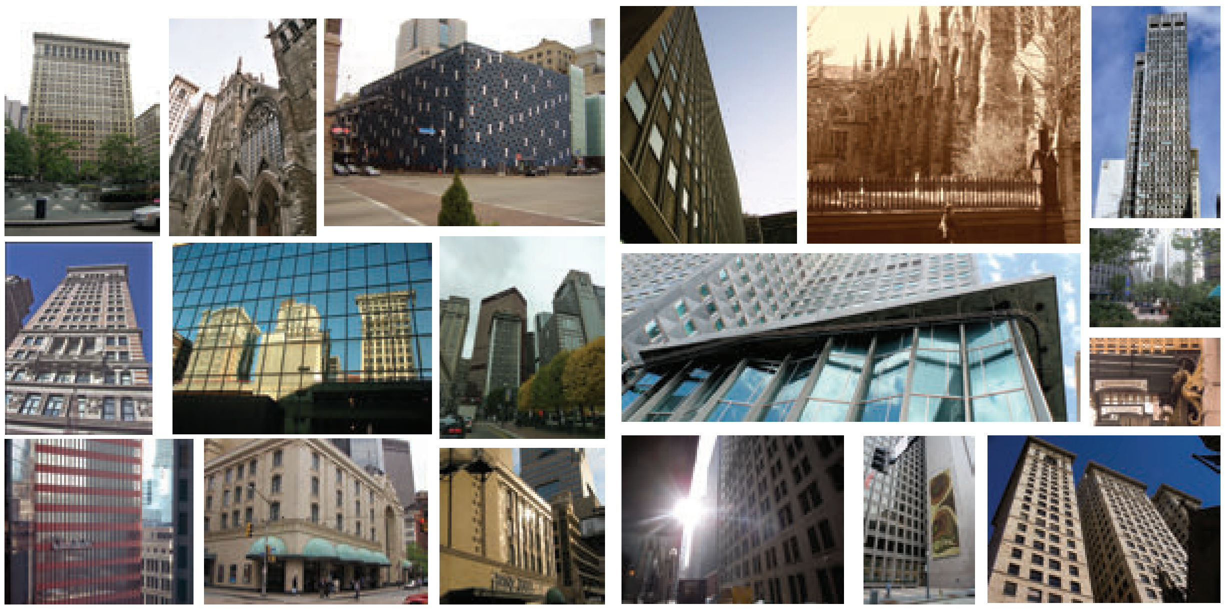
We divided the test images into two groups. The first group consists of those images of Pittsburgh that do not contain any unique (visually distinctive) buildings. We call this set *non-landmark images*. Where as the second group consists of those images of Pittsburgh that see some unique buildings, e.g., the PPG Place that dominates Pittsburgh skyline. We refer to this set as *landmark images*. Remembering that we are dividing the reference dataset into different groups using visual similarity. Our intuition is that reference images that see a particular landmark will be grouped together.

To test whether this is indeed the case and to ascertain if there are any benefits to such a grouping, we evaluate the proposed approach on both non-landmark and landmark images. For the tests presented below we assume that an image is correctly localized if it is within 200 meters of its true location. We used a desktop (Intel Core i5 2.8GHz processor, 6GB RAM) running Windows 7 to carry out the experiments presented here.

Table 4.3 summarizes the results of our approach using 84 non-landmark images. It also compares our approach to that of [Zamir and Shah, 2010]. Note that Zamir and Shah's approach is able to correctly localize 26 images out of 84. Our method is also able to correctly localize 26 images (300 topic dimensions; 10 groups). A reason for such low accuracy is that that

84 non-landmark images		
Model	Accuracy	Time (s)
Method in [Zamir and Shah, 2010]	30.9%	14.4136
100-Topic / 10 groups	28.6%	3.352
100-Topic / 50 groups	23.8%	2.3205
100-Topic / 100 groups	19	1.526
300-Topic / 10 groups	**30.9%**	**2.55567**
300-Topic / 50 groups	28.6%	1.6579
300-Topic / 100 groups	22.6%	1.3893
400-Topic / 10 groups	30.9%	4.969
400-Topic / 50 groups	23.8%	1.3043
400-Topic / 100 groups	20.2%	1.4096

Table 4.3: Localization performance comparison for non-landmark images.



(a) Success (b) Failures

Figure 4.2: (a) Some of the non-landmark images that were localized correctly by our method and (b) a selection of non-landmark images that were localized incorrectly by our method.

reference dataset is impoverished. One needs a lot more than just $50,220$ images to cover a city the size of Pittsburgh. Figure 4.2(b) depicts a sample of non-landmark images that were localized incorrectly by our method. It is worth mentioning that these images exhibit a high visual dissimilarity with the images present in our reference dataset.

Zamir and Shah's method on average takes roughly 14 seconds to process a query (Table 4.3). Our method is able to match the accuracy of Zamir and Shah's method (300 topic dimensions; 10 groups); however, our method processes a query on average in under 3 seconds. Consequently on average our method processes a query roughly 4.5 times faster than that of Zamir and Shah's method.

Table 4.2 summarizes the results of our approach using 37 landmark images. It also compares our approach to that of [Zamir and Shah, 2010]. Note that Zamir and Shah's approach is able to correctly localize 36 images out of 37. Both our method and the method presented in [Zamir and Shah, 2010] are able to achieve an accuracy of 97%. Here too our method matches the accuracy of Zamir and Shah's approach; however, on average our method is spends a little over 2 seconds per query and their method takes around 11 seconds to process a query. This is again a 5 fold increase in query processing speed.

It is worth keeping in mind that the localization accuracy for landmark images is much higher than that of non-landmark images. This is to be expected. Images showing generic items, we think are typically much harder to localize without a very detailed reference dataset. Tables 4.2 and 4.3 also exhibit a trend that we anticipated. Irrespective of the topic dimensions, partitioning the reference dataset into more scene groups adversely affects the accuracy and at the same time increases query processing speed. Increase in the processing speeds is easy to explain—more scene groups mean fewer SIFT features per group and smaller FLANN indices. Decrease in the accuracy; however, is related to the fact that the chance that the topic model will assign the query image to the wrong group increases as we increase the number of scene groups.

Table 4.4 aggregates the results for both non-landmark and landmark images. Our method

121 test images		
Model	Accuracy	Time (s)
Method in [Zamir and Shah, 2010]	51.2%	12.90775
100 Topics / 10 groups	**49.6%**	**2.84167**
100 Topics / 50 groups	42.9%	1.9723
100 Topics / 100 groups	39.7%	1.268515
300 Topics / 10 groups	48.7%	2.30431
300 Topics / 50 groups	47.1%	1.43461
300 Topics / 100 groups	38.8%	1.2755
400 Topics / 10 groups	49.6%	4.47315
400 Topics / 50 groups	40.5%	1.202
400 Topics / 100 groups	38.0%	1.315

Table 4.4: Image localization performance comparison aggregated over landmark and non-landmark images.

compares favorably with the method proposed by Zamir and Shah in terms of accuracy (ours 49.5% compared to theirs 51%). However, on average the proposed method processes a query 4.5 times faster than their method.

4.2 Image Localization using Vocabulary Trees

When using vocabulary trees for image localization, we focus primarily on the schemes used for weighing visual words. We randomly select 100 images from the reference geo-tagged dataset and use these as our query images. For our tests we assume that an image is correctly localized if it is localized to within 50 meters of its true location. We also use this dataset to evaluate the proposed scene groups based approach and the scheme that appeared in [Zamir and Shah, 2010]. Here we represented each reference image as a 100 dimensional topic vector and partitioned our dataset into 10 scene groups. The results are summarized in Table 4.5.

When using vocabulary trees for image localization, our proposal of assigning weights to visual words using both their occurrence frequencies and their geographical spreads outperforms the traditional scheme of assigning weights to visual words using only their occurrence frequencies. Specifically, for our particular dataset, our method of assigning weights to visual

100 test images		
Model	Accuracy	Time(s)
Vocabulary tree [Nistr and Stewnius, 2006]	42 %	0.49
Vocabulary tree (Section 3.1.2)	67%	0.49
Our scene model method	**94%**	**0.34**
[Zamir and Shah, 2010]	98%	1.32

Table 4.5: Image localization using vocabulary trees.

words achieves an accuracy of 67% as opposed to the traditional scheme of assigning weights, which only achieved an accuracy of 47%. It seems that vocabulary trees are perhaps not suitable for image localization applications. Notice that both Zamir and Shah [Zamir and Shah, 2010] and the proposed approaches achieve 98% and 94% accuracies, respectively. Also note that the approach presented in this paper posts better times than all other methods.

Chapter 5

Conclusions and Future Works

In this work, we focus on image localization using a geo-tagged city-scale image dataset. While most existing methods strive for improving accuracy, we focus on the issue of scale. Our method is motivated by Zamir and Shah work on image localization that appeared in [Zamir and Shah, 2010]. Their method constructs a single index for storing SIFT features collected from the reference dataset. Using a single index works reasonably well for smaller datasets, but does not scale to larger datasets. Instead we propose an automatic method for partitioning reference dataset into groups. Our method relies upon topic model analysis to group visually similar images into different groups, called *scene groups*. When presented with a query image, first its corresponding scene group is selected and then SIFT features extracted from this query image are matched against the SIFT features collected from reference images belonging to that group to identify the set of visually similar reference images. This set of visually similar images is then used to estimate the location of the query image.

We have examined the proposed method against Zamir and Shah scheme and show that both methods achieves similar accuracies, however, our method is roughly 4.5 times faster than Zamir and Shah method. We also presented a new scheme for assigning weights to visual words when using vocabulary trees for image matching for the purposes of image localization. Our results suggest that our scheme for assigning weights to the visual words gives better results then the approach used in [Nistr and Stewnius, 2006]. Furthermore our results suggest that vocabulary tree based image matching technique are not suitable for appearance based image localization methods.

It is worth noting that the performance of the proposed methodology depends upon the quality of the reference geo-tagged dataset. For example, our reference dataset do not contain any night images, so our method is unable to localize a query image taken at night. Of course Zamir and Shah [Zamir and Shah, 2010] and Nistr and Stewnius [Nistr and Stewnius, 2006] methods also suffer from the same drawback. In situations where reference dataset contains a lot of "similar" images, topic model analysis may result in unbalanced scene groups, i.e., some of the scene groups contain most of the images from the reference set. The performance of the proposed approach will degrade in such situations.

5.1 Future Works

We realize that we have barely scratched the surface and that many more experiments are needed to fully understand the behavior of the system. There are many avenues for further research, including

- Automatically select the number of visual words that should be constructed from the SIFT features. Currently, we provide this information to the algorithm.
- Dynamic topic model analysis, where topic models are dynamically updated when new images are added to the reference dataset.
- Dynamic FLANN inddex update, where FLANN index is dynamically updated as new SIFT features are added to it.
- Automatic selection of the size of the topic vector. Currently, we provide this information to the algorithm.

Bibliography

[Agarwal et al., 2009] Agarwal, S., Snavely, N., Simon, I., Seitz, S., and Szeliski, R. (2009). Building rome in a day. In *IEEE 12th International Conference on Computer Vision*, volume 2, pages 72–79.

[Blei et al., 2003] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

[Bosch et al., 2006] Bosch, A., Zisserman, A., and Muñoz, X. (2006). Scene classification via plsa. In *European Conference on Computer Vision*, pages 517–530.

[Fei-Fei and Perona, 2005] Fei-Fei, L. and Perona, P. (2005). A bayesian hierarchical model for learning natural scene categories. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 524–531.

[Feng and Lapata, 2010] Feng, Y. and Lapata, M. (2010). Topic models for image annotation and text illustration. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 831–839.

[Google, 2013] Google (2013). `http://maps.google.ca/intl/ALL_ca/help/maps/streetview/index.html/`. last visited: 09 July 2013.

[Hays and Efros, 2008] Hays, J. and Efros, A. A. (2008). im2gps: estimating geographic information from a single image. In *In Proc. of the IEEE on Computer Vision and Pattern Recognition*, pages 1–8.

[Hörster et al., 2009] Hörster, E., Lienhart, R., Effelsberg, W., and Möller, B. (2009). Topic models for image retrieval on large-scale databases. *ACM Sigmultimedia Records*, 1:15–16.

[Hörster et al., 2007] Hörster, E., Lienhart, R., and Slaney, M. (2007). Image retrieval on large-scale image databases. In *In Proc. of the 6th ACM international conference on Image and video retrieval*, pages 17–24.

[Johansson and Cipolla, 2002] Johansson, B. and Cipolla, R. (2002). A system for automatic pose-estimation from a single image in a city scene. In *IN IASTED INT. CONF. SIGNAL PROCESSING, PATTERN RECOGNITION AND APPLICATIONS*.

[Juan and Gwon, 2009] Juan, L. and Gwon, O. (2009). A Comparison of SIFT, PCA-SIFT and SURF. *International Journal of Image Processing (IJIP)*, 3(4):143–152.

[Kalantidis et al., 2011] Kalantidis, Y., Tolias, G., Avrithis, Y. S., Phinikettos, M., Spyrou, E., Mylonas, P., and Kollias, S. D. (2011). Viral: Visual image retrieval and localization. *Multimedia Tools Applications*, 51(2):555–592.

[Kleusberg and Langley, 1990] Kleusberg, A. and Langley, R. B. (1990). The limitations of gps. *GPS World*.

[Li et al., 2008] Li, X., Wu, C., Zach, C., Lazebnik, S., and Frahm, J.-M. (2008). Modeling and recognition of landmark image collections using iconic scene graphs. In *European Conference on Computer Vision*, pages 427–440.

[Li et al., 2010] Li, Y., Snavely, N., and Huttenlocher, D. P. (2010). Location recognition using prioritized feature matching. In *European Conference on Computer Vision*, pages 791–804.

[Liu et al., 2004] Liu, Y., Collins, R., and Tsin, Y. (2004). A computational model for periodic pattern perception based on frieze and wallpaper groups. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):354–371.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[Muja and Lowe, 2009] Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application*, pages 331–340.

[Newman et al., 2006] Newman, D., Smyth, P., and Steyvers, M. (2006). Scalable Parallel Topic Models. *Journal of Intelligence Community Research and Development*.

[Nistr and Stewnius, 2006] Nistr, D. and Stewnius, H. (2006). Scalable recognition with a vocabulary tree. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2161–2168.

[Oliva and Torralba, 2001] Oliva, A. and Torralba, A. (2001). Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *International Journal of Computer Vision*, 42(3):145–175.

[Papadimitriou et al., 1998] Papadimitriou, C. H., Tamaki, H., Raghavan, P., and Vempala, S. (1998). Latent semantic indexing: a probabilistic analysis. In *In Proc. of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 159–168.

[Putthividhya et al., 2010] Putthividhya, D., Attias, H., and Nagarajan, S. (2010). Supervised topic model for automatic image annotation. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 1894–1897.

[Quelhas et al., 2005a] Quelhas, P., Monay, F., Odobez, J.-M., Gatica-Perez, D., Tuytelaars, T., and Gool, L. J. V. (2005a). Modeling scenes with local descriptors and latent aspects. In *International Conference on Computer Vision*, pages 883–890.

[Quelhas et al., 2005b] Quelhas, P., Monay, F., Odobez, J.-M., Gatica-Perez, D., Tuytelaars, T., and Van Gool, L. (2005b). Modeling scenes with local descriptors and latent aspects. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 1, pages 883–890.

[Robertson and Cipolla, 2004] Robertson, D. and Cipolla, R. (2004). An image-based system for urban navigation. In *IN BMVC*, pages 819–828.

[Schindler et al., 2007] Schindler, G., Brown, M., and Szeliski, R. (2007). City-scale location recognition. In *Computer Vision and Pattern Recognition*.

[Schindler et al., 2008] Schindler, G., Krishnamurthy, P., Lublinerman, R., Liu, Y., and Dellaert, F. (2008). Detecting and matching repeated patterns for automatic geo-tagging in urban environments. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7.

[Siagian and Itti, 2007] Siagian, C. and Itti, L. (2007). Rapid biologically-inspired scene classification using features shared with visual attention. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):300–312.

[Snavely et al., 2006] Snavely, N., Seitz, S. M., and Szeliski, R. (2006). Photo tourism: exploring photo collections in 3d. *ACM Transactions on Graphics*, 25:835–846.

[Tipping and Schölkopf, 2001] Tipping, M. and Schölkopf, B. (2001). A kernel approach for vector quantization with guaranteed distortion bounds. *Artificial intelligence and statistics*, pages 129–134.

[Torii et al., 2011] Torii, A., Sivic, J., and Pajdla, T. (2011). Visual localization by linear combination of image descriptors. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 102–109.

[Zamir and Shah, 2010] Zamir, A. R. and Shah, M. (2010). Accurate image localization based on google maps street view. In *In Proc. of the European Conference on Computer Vision*, pages 255–268.

[Zhang and Kosecka, 2006] Zhang, W. and Kosecka, J. (2006). Image based localization in urban environments. In *In Proc. of the Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06)*, 3DPVT '06, pages 33–40.

[Zheng et al., 2009] Zheng, Y.-T., Zhao, M., Song, Y., Adam, H., Buddemeier, U., Bissacco, A., Brucher, F., Chua, T.-S., and Neven, H. (2009). Tour the world: Building a web-scale landmark recognition engine. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1085–1092.