UNIVERSITY OF CALIFORNIA,
IRVINE

Strong Geometric Context for Scene Understanding

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Electrical Engineering and Computer Science

by

Raúl Díaz García

Dissertation Committee:
Charless Fowlkes, Chair
Deva Ramanan
Athina Markopoulou

2016

DEDICATION

To Aida, my parents and brother, and my *yayos*.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

I want to thank my advisor, Prof. Charless Fowlkes, for taking me as his student. I joined the EECS department in 2011 and could not find an advisor that would take me to do research on image processing or computer vision. Right after meeting Prof. Fowlkes, he offered his help in anything that was on his hands so I could join the vision group at the ICS department. I am grateful for the patience that he has had with me for the past 5 years. I started a PhD program after 5 years working for the industry in Barcelona, with little approach to the research world. I realized my studying skills were rusty after that time, but he was always willing to explain any doubts that I had, no matter how basic they were.

I would also like to thank my committee members, Profs. Deva Ramanan and Athina Markopoulou to help me improve this thesis. I am especially grateful for Prof. Ramanan's questions during our group meetings. His honest, insatiable curiosity for knowing every detail of the research I was presenting helped me improve my presentation skills and the interaction with the audience.

I am also happy to have met my vision lab mates. From the "veterans" Chaitanya, Dennis, and Yi, to the "juveniles" Phuc, Peiyun, Minhaeng, and Shu. Thank you also to Xiangxing, my third-year carpool mate; to Mohsen and his worry-free mentality; to James and our conversations about the meaning of "freedom"; and to Bailey for all the time shared in the small lab. Thank you also to Golnaz, Maryam, Shaofei, and the morning chats with Greg. Last but not least, with Dr. Sam Hallman, partner in research and travel through these years.

Thank you to my family and their support. From my parents who were unsure about the step I was taking in 2011 but always supported me when I started the program, to my brother Robert and his partner Anna for bringing my niece Rita to this world. Thank you to my childhood friends who I still see when I go back home and who always say they are proud of me. They do not know they are the ones I miss the most. I also feel grateful for the wonderful people I met during these years: the friends from San Diego's "la Vecindad" and the long dinner parties and asados; also, the "Catalan mafia" of Irvine, a never-ending stream of folks that always makes you feel at home.

Last but not least, I want to thank my wonderful partner in life Aida. Thank you for all the days lived and the ones to come, for being patient and supporting, and for helping me take this wonderful opportunity.

Finally, I would like to thank the Balsells Foundation for supporting me during the first two years of the program, as well as the long term financial support provided by NFS grants IIS-1618806 and IIS-1253538.

CURRICULUM VITAE

Raúl Díaz García

EDUCATION

Doctor of Philosophy in Computer Science **2016**
University of California, Irvine *Irvine, California*

Master of Science in Computer Vision and Artificial Intelligence **2009**
Universitat Autonòma de Barcelona *Cerdanyola del Vallès, Spain*

Bachelor of Science in Computer Engineering **2007**
Universitat Autònoma de Barcelona *Cerdanyola del Vallès, Spain*

RESEARCH EXPERIENCE

Graduate Research Assistant **2011–2016**
University of California, Irvine *Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant **2009–2011**
Universitat Autonòma de Barcelona *Cerdanyola del Vallès, Spain*

REFEREED CONFERENCE PUBLICATIONS

Multi-View Background Subtraction for Object Detection **Jun 2013**
CVPR Scene Understanding Workshop (SUNw)

Detecting Dynamic Objects with Multi-View Background Subtraction **Dec 2013**
International Conference on Computer Vision (ICCV), p. 273-280

Lifting GIS Maps into Strong Geometric Context for Scene Understanding **Mar 2016**
Winter Conference on Applications in Computer Vision (WACV)

ABSTRACT OF THE DISSERTATION

Strong Geometric Context for Scene Understanding

By

Raúl Díaz García

Doctor of Philosophy in Electrical Engineering and Computer Science

University of California, Irvine, 2016

Charless Fowlkes, Chair

Humans are able to recognize objects in a scene almost effortlessly. Our visual system can easily handle ambiguous settings, like partial occlusions or large variations in viewpoint. One hypothesis that explains this ability is that we process the scene as a global instance. Using global contextual reasoning (*e.g.*, a car sits on a road, but not on a building facade) can constrain interpretations of objects to plausible, coherent precepts. This type of reasoning has been explored in Computer Vision using weak 2D context, mostly extracted from monocular cues. In this thesis, we explore the benefits of strong 3D context extracted from multiple-view geometry. We demonstrate strong ties between geometric reasoning and object recognition, effectively bridging the gap between them to improve scene understanding.

In the first part of this thesis, we describe the basic principles of structure from motion, which provide strong and reliable geometric models that can be used for contextual scene understanding. We present a novel algorithm for camera localization that leverages search space partitioning to allow a more aggressive filtering of potential correspondences. We exploit image covisibility using a coarse-to-fine, prioritized search approach that can recognize scene landmarks rapidly. This system achieves state of the art results in large-scale camera localization, especially in difficult scenes with frequently repeated structures.

In the second part of this thesis, we study how to exploit these strong geometric models and

localized cameras to improve recognition. We introduce an unsupervised training pipeline to generate scene-specific object detectors. These classifiers outperform state of the art and can be used when the rough camera location is known. When precise camera pose is available, we can inject additional geometric cues into novel re-scoring framework to further improve detection. We demonstrate the utility of background scene models for false positive pruning, akin to video-surveillance background subtraction strategies. Finally, we observe that the increasing availability of mapping data stored in Geographic Information Systems (GIS) provides strong geo-semantic information that can be used when cameras are located in world coordinates. We propose a novel contextual reasoning pipeline that uses lifted 2D GIS models to quickly retrieve precise geo-semantic priors. We use these cues to to improve object detection and image semantic segmentation, providing a successful trade-off of false positives that boosts average precision over baseline detection models.

Chapter 1

Introduction

The problems of object recognition and estimation of 3D geometry in Computer Vision have largely been pursued separately. However, these two traditional fields can be seen as versions of *scene understanding* even though they look very different. There seems to be many good arguments for why these two sub-disciplines should join forces. Accurate recognition and segmentation of objects in a scene should constrain matching of features to hypothesized surfaces, aiding reconstruction and camera localization. Similarly, geometric information should provide useful features and contextual constraints for object detection and recognition. This thesis builds bridges between these two main blocks of Computer Vision in order to improve scene understanding. Figure 1.1 lays out the relation between these concepts and their corresponding chapters in this thesis.

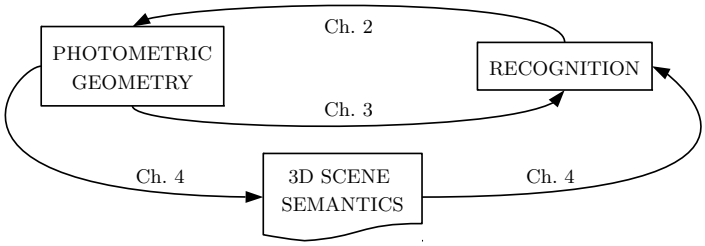


Figure 1.1: Recognition and Geometry are two disciplines of Computer Vision that can be joined to benefit from each other. We explore these ties and exploit external geo-semantic resources with the goal of improve scene understanding.

1.1 Recognition cues for improved multi-view geometry

Multi-view geometry can be seen as a form of recognition that detects the underlying geometric structure of a scene. Firstly, 2D points from a set of unstructured images are matched in a pairwise fashion. This step can be regarded as a detection phase, in which 2D points in each image are described using robust, invariant transforms (*e.g.*, SIFT [46], SURF [5], BRIEF [9], etc.) and matched against another image using a discriminative similarity measure. A second step consists in finding the subset of those correspondences that explain the whole scene depicted by these images. This is done by using a geometric verification that turns these 2D observations into 3D points. This process, named Structure from Motion (or SfM), is well developed and allows us to build robust models of a scene in the form of 3D point clouds and relative camera poses. There exists a handful of well known implementations with minor differences and improvements that have proved to be successful in this task [73, 52, 66]. Additionally, dense multi-view stereo (or MVS) can find and add new 3D points to the scene by using a photometric measure along the epipolar geometry of the images registered in the SfM model [24]. We can use these models to make strong predictions about where a novel test image was taken, including its camera location and scene points within

such image.

In this thesis, we analyze the difficulties of large-scale camera localization using Structure-from-Motion models in Chapter 2. The motivations are twofold. First, geometric camera localization is a major requirement to study object recognition from a multi-view perspective, since an image that is not well localized is useless given a geo-semantic context. Second, there are major problems widely unexplored when finding the pose of an image in the presence of heavily repeated structure, which is the case of most man-made environments like cities. Large-scale repetitive structure clutters the feature space used to find correspondences between the query image and the model. This results in both the rejection of many good features due to the commonly used ratio test [46], and the mismatching of correspondences due to approximate nearest neighbor search. We propose to solve such problems by using a fast, linear-time algorithm that quickly finds the spatial distribution where the query image is most likely taken from. Once we have such hypothesis, we perform 3D-to-2D matching, as opposed to the more popular 2D-to-3D matching, following a landmark recognition criteria while exploiting co-visibility relations between model images. We show that this method outperforms state of the art for both retrieval problems (finding model images close to the query) and for full 6DOF camera pose estimation.

1.2 Improving recognition using geometric constraints

The classic approach in object recognition consists in training a classifier with little or no contextual reasoning at all. The classifier essentially solves a binary problem, where image features from positive instances (the object/class of interest) are distinguished from negative ones (all other parts of the image). These features are typically extracted from a generic, unstructured, and unrelated training set of images. Then, a state of the art classifier is trained to discern between these positive and negative instances.

However, this process is far from being similar to the one happening in humans. For us, the task of finding objects in images is a very simple task that we do with little effort. A hypothesis for this ability is that our brain uses semantic context to enforce the understanding of the scene as a global instance. Consequently, we will easily discard the statues when looking for pedestrians in Figure 1.2. Cognitive sciences have studied this human behavior and defined a list of *violations* that allows us to detect quickly inconsistent scenes. As stated by Biederman [7], there are four out the five of such violations that can be considered of a geometric nature: size, support, interposition, and position. Hence, while the appearance and shape of the statues may seem to look like people, they are a clear example of a position violation. This indicates that geometry is a crucial part of scene understanding that should be incorporated in machine learning techniques.

General efforts in object recognition using geometric cues have focused on the role of context in a single image, operating under the assumption that the camera and scene geometry are unknown and must largely be inferred based on the analysis of monocular cues. This problem is quite difficult in general although some progress has been made [38, 64, 17], particularly on indoor scenes of buildings where the geometry is highly regular [60, 34, 35]. The use of detailed stereo depth has already proven to be incredibly helpful in the world of object detection, particularly for RGB-D sensors in indoor scenes [30, 45]. More general formulations have attempted to jointly integrate geometric reconstruction from multiple images with scene and object recognition [4, 11, 40].

However, multi-view geometry has shown to be extremely robust and able to generate precise camera pose estimation, as well as strong geometric models of the scene depicted by large sets of unstructured images [1, 73]. To this end, we argue that scene understanding should shift to a multi-view framework that provides accurate geometric and semantic priors to study the influence of context in the world of object recognition and semantic segmentation.

In essence, Structure from Motion and dense multi-view stereo can be seen as the detection

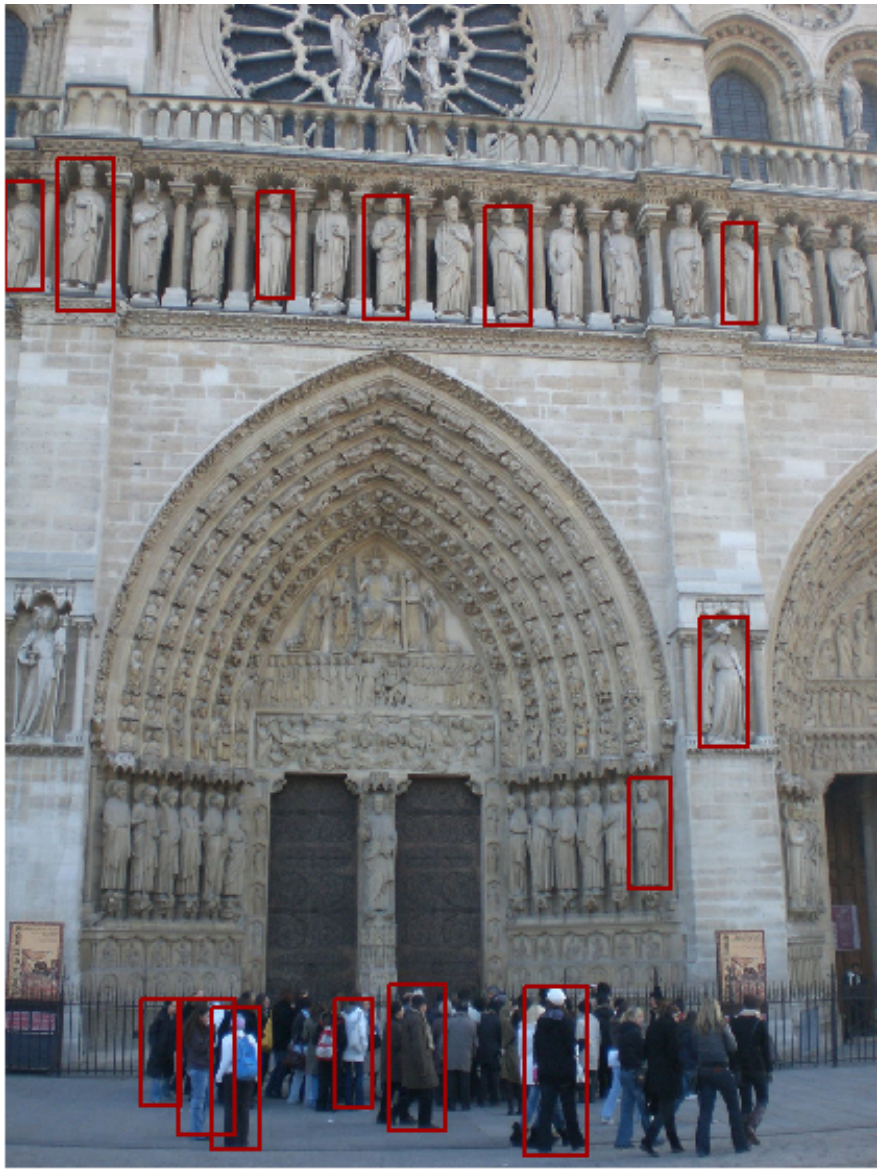


Figure 1.2: A common error produced by traditional object detectors is considering all areas of an image equally likely to contain the object of interest. This produces false positive detections where there is a human-shaped object (the statues in the facade). However, humans quickly infer the context of the scene to discard such figures as pedestrians. We present ideas to add this type of reasoning to machines in order to improve object detection and segmentation.

and scene understanding of the static (rigid) background of the scene: the final structure model only contains those components that were common and constant across all images (*e.g.*, buildings, roads, street lights). For dynamic objects, the problem is less constrained. However, past images of a scene can still provide general information about where objects are likely to appear in the future. For example, we might expect *a priori* to see pedestrians on a sidewalk and cars on the road (and not vice versa). From the perspective of multi-view geometry, dynamic objects are a nuisance and must be treated as outliers during matching. In this thesis we explore how to combine these two ideas, namely: *How can strong models of static backgrounds improve detection of dynamic objects?*

The confluence of these robust algorithms for Structure from Motion along with high-coverage mapping and imaging of the world around us suggests that it will soon be feasible to accurately estimate camera pose for a large class photographs taken in outdoor, urban environments. We investigate how such information can be used to improve the detection of dynamic objects such as pedestrians and cars. First, we show that when rough camera location is known, we can utilize detectors that have been trained with a scene-specific background model in order to improve detection accuracy. Second, when precise camera pose is available, dense matching to a database of existing images using multi-view stereo provides a way to eliminate static backgrounds such as building facades, akin to background-subtraction often used in video analysis. We evaluate these ideas using a dataset of tourist photos with estimated camera pose. For template-based pedestrian detection, we achieve a 50 percent boost in average precision over baseline. This work was published in [15], and is described in detail in Chapter 3.

Recently, another important source of geo-spatial information has become available. Data stored in Geographic Information Systems (GIS) offers a rich source of contextual information that has been largely untapped by Computer Vision. While detailed GIS map data is used extensively in analysis of aerial images [81, 82, 59], only a handful of papers have exploited

this resource to study scene understanding from a ground-level perspective. Recently, a few groups have tried using GIS data and multi-view geometry for improving object recognition. [49] introduced a geographic context re-scoring scheme for car detection based on street maps. Ardeshir *et al.* used GIS data as a prior for static object detection and camera localization [3] and for performing segmentation of street scenes [2].

We propose to leverage rich 3D GIS models for scene understanding by combining them with large sets of unorganized photographs using Structure from Motion techniques. We present a pipeline to quickly generate strong 3D geometric priors from 2D GIS data using SfM models aligned with minimal user input. Given an image registered against this model using full 6DOF estimation, we generate robust predictions of depth, surface normals, and semantic labels. Despite the lack of detail in the model, we show that the precision of the predicted geometry is substantially more accurate than other single-image depth estimation methods. We then demonstrate the utility of these contextual constraints for re-scoring pedestrian detections, and use these GIS contextual features alongside object detection score maps to improve a CRF-based semantic segmentation framework, boosting accuracy over baseline models. These ideas, presented in Chapter 4, were reflected in [16].

Chapter 2

Large-Scale Structure from Motion and Camera Localization

2.1 Introduction

Structure-from-Motion (SfM) is the collection of techniques that allows both the reconstruction of scenes into 3D models, and the discovery of the relative camera poses of a set of unstructured images taken from different viewpoints. The initial steps in this direction started with self-calibration methods [6, 50] and the development of the first pipelines applied in unordered pictures from the Internet [70]. Recently, SfM has exploded into a wide variety of implementations for numerous purposes: from systems that can handle a few hundred images to many thousands of them. To approach these larger scale problems, the Computer Vision community has focused on different strategies like incremental [1, 84, 52, 66], hierarchical [26], or global [53, 12] SfM. In this thesis, we always consider the case of incremental SfM, as it accounts to solve the eventual problem of constantly adding new images to the model, targeting the goal of world-sized reconstructions [36].

Different methods for geometric verification have been proposed to find the 6 degrees of freedom of a camera pose (roll, pitch, yaw, and 3D translation). These methods are commonly known as Perspective-n-Point (PnP), and can be used when correspondences between 3D points in the world and 2D points in the image are available. To remove potential outliers, robust techniques like RANSAC [21], MLESAC [78], or AC-RANSAC [51] have been introduced. These iterative techniques evaluate the fitness of a randomly sampled minimal solution in the observed data, until a better solution is unlikely to be found.

The importance of where a novel query image was taken from is also crucial. This method, named camera localization or resection, is an important step in many applications of Computer Vision beyond Structure-from-Motion (*e.g.*, autonomous driving, robotics, etc.). It also plays an important role in the world of scene understanding, as it allows for precise backprojection of strong geosemantic features [49, 83]. Many of the challenges that camera localization faces are related to the big data era, where social networks have provided a quasi-infinite stream of visual information that is hard to handle by current localization strategies.

In this chapter, we review the basic building blocks of incremental SfM and its related work and implementations. We also review the state of the art in camera localization, and propose a strategy that can handle large-scale datasets with heavily repeated structure.

2.2 Incremental Structure from Motion

SfM is composed of several steps targeted to guarantee the most precise, reliable, and complete representation of the underlying scene depicted by a provided set of unordered images. As mentioned in the introduction of this thesis, SfM can be seen as a detection and a scene understanding technique. First, features from the set of images are extracted and matched,

followed by a geometric verification. The resulting scene graph is then traversed greedily to incrementally grow a 3D point cloud while relative camera poses are being computed. This section lays the basic concepts of SfM as well as the related work in this topic, dividing the reconstruction in two main blocks. Figure 2.1 illustrates these different steps taken in SfM.

2.2.1 Feature Matching and Geometric Verification

Keypoint Detection: The first step in any SfM pipeline is to find points in the images that are salient and likely to be found in other images depicting the same scene. For every image I in the image set $\mathcal{I}$, these keypoints are extracted and stored alongside a feature vector (commonly known as descriptor). These descriptors are generated in a way such that they are invariant to geometric and photometric distortions (*e.g.*, rotation, scale, lightning, etc.). Many implementations have emerged in the past years providing different benefits such as speed [5] or wide-baseline robustness [77]. Among them, SIFT [46] is widely used as a gold standard for SfM.

Matching: The next stage tries to find correspondences between descriptors of every possible pair of images $I_a, I_b \in \mathcal{I}$. To do so, the traditional approach consists of creating a kd-tree search index in one of the images of the pair and query the other image's descriptors to find the closest and second closest nearest neighbor (NN) keypoints. The second NN serves as an upper bound to evaluate how discriminative the first NN is. Due to the similarity between descriptors in scenes with repeated structures, Lowe's test [46] computes the ratio of the distances in feature space of the first and second nearest neighbors f, f' with respect to a query feature q, $\alpha = \frac{\|f-q\|}{\|f'-q\|}$. If α is smaller than some threshold τ, the first NN is considered to be a potential good match. By filtering out features that appear visually similar to more than one candidate location, the ratio test has proved to be effective at automatically selecting reliable candidates.

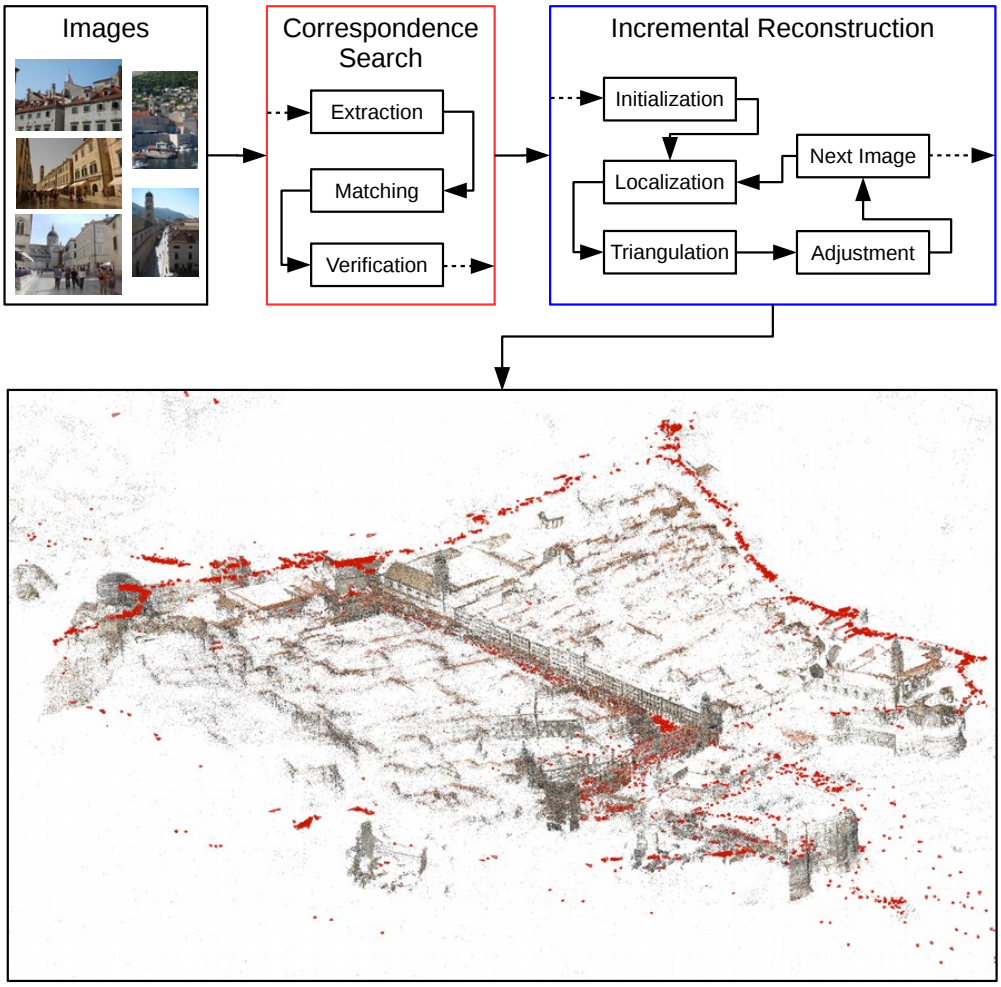


Figure 2.1: Structure-from-Motion is able to reconstruct the scene depicted by large sets of images, and recover the relative camera poses. Section 2.2.1 explains the different steps of denoted in the red box for pairwise correspondence search. The blue box contains the processes involved in the incremental reconstruction of the scene, defined in section 2.2.2. In this example, we show a model obtained from 6844 images from the city of Dubrovnik.

While this matching step works well in practice for small datasets, larger datasets become intractable since we need to match exhaustively every possible pair of images. Hence, the computation complexity is quadratic over the number of images. To accelerate this step, the most common solution in SfM implementations is the use of an approximate nearest neighbor search [55, 56], instead of a slower, exhaustive search. This speeds up the pairwise matching with a small tradeoff in correspondence precision, although we still need to evaluate all possible pairs of images. To avoid this, preemptive matching [84] proposes to perform an initial search only in the top h largest descriptors (since descriptors like SIFT provide a scale measure). If the number of matches obtained is greater than some threshold t_h, the pair is considered good to perform full matching using all keypoints. Other approaches make use of a vocabulary tree to identify good image pairs [32]. Vocabularies consist of a relatively small set of generic descriptors, in which image descriptors are matched only once (as opposed to compute all possible pairs). Pairs of images with enough common vocabulary words are used for full matching. Recently, [65] proposed the use of learning-based methods that are able to classify good and bad pairs based on hash functions over the distribution of descriptors in every pair of images.

Geometric Verification: The last step of the first building block of SfM tries to verify that the appearance-based matching of an image pairs is coherent also geometrically. Different projective mappings are used depending on the configuration of the dataset. When scenes are mostly planar, homographies are usually the choice for geometric verification. A more general fitting is acquired using epipolar geometry mappings, like the fundamental or essential matrices [31]. The computation of these mappings are subject to noise in the form of outliers. Robust fitting methods like RANSAC are used to identify the inliers of the pair that geometrically verify the fitness of such pair.

2.2.2 Incremental Reconstruction

The result of the previous matching and verification steps consists of a *scene graph* whose nodes are the images, and whose edges link geometrically verified pairs of such images. For every pair, the list of inlier correspondences is also stored for further use. Initially, a first pair of images is chosen to perform two-view reconstruction. When the relative camera pose between this pair is estimated, a set of initial 3D points is triangulated from the inlier correspondences. The rest of the incremental reconstruction stage involves a greedy loop over the scene graph, until no more cameras can be localized and SfM stops. The initial pair selection is critical, as one of the major problems of incremental SfM is bad initialization. When the initial pair does not return a good reconstruction, errors accumulate throughout the pipeline and the model may not be able to recover from such state. This produces a drifting effect that leads to poor reconstructions and the inability of closing a loop (*e.g.*, turning around a whole block in a street). If the initialization is satisfactory, SfM generally loops smoothly over the scene graph, providing accurate geometry of the underlying scene. We now describe the three basic steps within such loop.

Next image localization: After the initial reconstruction, the scene graph is traversed to find the next image that shares more connected correspondences with the currently reconstructed 3D points. This yields a set of 2D-3D correspondences that allow the use of PnP solvers. The result of such solvers include extrinsic parameters (3 for camera position + 3 for camera orientation), and intrinsic parameters (focal length and radial distortion) if the latter were unknown a priori. RANSAC is used to guarantee that the camera location is found with no outliers under some reprojection error threshold ϵ.

Triangulation: Once a new image has been added to the model, SfM uses the scene graph to find new 3D points to augment the coverage of the scene. For every image already

included in the model that shares points with the recently added image, SfM looks for the correspondences that were found in the initial matching stage. Those matches which had not been triangulated yet are estimated and added to the model, thus increasing the likelihood of those cameras not yet localized to have more potential correspondences.

Bundle adjustment: BA is the last piece of the incremental loop of SfM. After an image has been added and new 3D points have been triangulated, the model tends to accumulate errors with respect to the initial reconstruction and the model can drift. To prevent this effect, BA [31, 80] provides a refinement that minimizes errors between camera poses and 3D points reconstructed so far. The sum of the reprojection errors is modeled as a non-linear least squares minimization problem using Levenberg-Margquardt. In order to speed up the overall time of the reconstruction, BA is usually performed after a certain number of images have been added to the model, rather than every single time a new image is localized.

Model evaluation: When SfM finishes, a collection of camera poses containing intrinsic and extrinsic parameters, 3D points, and *tracks* (list of image observations linked to a 3D point) are returned. Performance of the recovered model can be measured in several ways. For small datasets, there exists ground truth data in the form of laser-precision models [75] where the camera pose errors can be measured up to a scale factor. With the appropriate settings, SfM can yield impressive results of only a few centimeters in localization errors [52]. For larger datasets, benchmarks typically evaluate the completeness of the model like the number of cameras successfully localized, the number of 3D points, the average reprojection error, and the time spent in the reconstruction [66].

2.3 Large Scale Camera Localization

Camera localization (also commonly referred as *resectioning*) is the task of finding the full 6DOF camera pose of a query image when given a 3D model. This method is approached in a similar way as SfM adds new images in the reconstruction. However, the set of points where the correspondences need to be found belong to two different geometric spaces, as the 2D points of the query image need to be matched against a cloud of 3D points obtained from an SfM reconstruction. The matching is performed in descriptor space (*e.g.*, SIFT) using an approximate nearest neighbor search function $kNN(q, D, k)$. This function returns the closest k descriptors of the query feature q in a database D. Algorithm 1 shows the outline of this standard approach. For every query feature in the test image, a first and second nearest neighbor model descriptors v_1 and v_2 are requested and a ratio test is performed to eliminate matches that are not considered discriminative. A PnP solver is later used in a robust model fitting pipeline like RANSAC to find the camera pose (and intrinsic parameters if missing) from the set of 2D-3D correspondences, alongside a vector δ of reprojection errors. The resulting camera pose is considered valid if it has more than 12 inlier correspondences whose reprojection error is smaller than some threshold ϵ.

There are several strategies to describe a 3D point in SIFT space prior to perform nearest neighbor search. A 3D point is the triangulation of n 2D observations from n different images depicting the same point (where $n > 1$). Hence, there are at least 2 descriptors defining such point. Typical representations in the search space are the mean or median feature descriptor of the 3D point observations (also named *views* in this thesis). Another alternative is to simply use all possible descriptors from all views of the SfM model. [61] evaluated in depth these different strategies in a metric scaled dataset of tourist images [44].

The main problems in this standard localization pipeline arise when the search space becomes denser. As the SfM model becomes large, finding good correspondences becomes more

difficult due to two factors. First, the standard 2D-to-3D *forward* matching is likely to accept bad correspondences of a query feature with the model since the feature space is cluttered with similar descriptors from completely different locations. This is often common in man-made environments such as cities, where repeated structures are heavily present. For the same reason, a nearest neighbor that is matched correctly might fail the ratio test since its second nearest neighbor might be too close in feature space. Second, a noisier set of correspondences obtained from the matching stage increases the runtime of the fine pose RANSAC estimation step, as its complexity grows exponentially with the number of outliers.

Algorithm 1 Standard Camera Localization

Require: Query features Q, Model features $\mathcal{V}$, ratio threshold τ, error threshold ϵ
  $\mathcal{M} \leftarrow \emptyset$
  for all $q \in Q$ **do**
    $v_1, v_2 = \text{kNN}(q, \mathcal{V}, 2)$
    $\alpha_q = \frac{\|q - v_1\|}{\|q - v2\|}$
    if $\alpha_q \leq \tau$ **then**
      $\mathcal{M} = \mathcal{M} \cup (q, v_1)$
    end if
  end for
  $I_P, \delta = \text{ROBUSTFITTING}(\mathcal{M}, \epsilon)$
  if $|\delta \leq \epsilon| \geq 12$ **then**
    return Camera Pose I_P
  else
    return Error - Pose not found
  end if

2.3.1 Related work

These problems are well known by the community and generally approached in several ways. Works such as [44] are focused on a simplification of the 3D model where the query image is matched. Given the bipartite visibility graph between training images and 3D points in the model, they solve a set cover problem, where they search for the smallest subset of points that cover all cameras as a measure of representativeness. Rather than the traditional approach of forward matching, they implement a 3D-to-2D *back* matching step, allowing the

ratio test to be performed in a sparser feature space. To prevent having to match all model points against the query image, they prioritize the search given 3D point visibility across images. Additionally, [43] apply co-occurrence priors in the RANSAC loop to reduce the number of sampling rounds in the fine pose estimation step.

An alternative of model compactness are vocabulary trees. [61] make use of them to speed up forward matching by assigning each model point and each query feature to a vocabulary word. The main advantage for this is that the initial nearest neighbor search is performed against a database of *only* 100 thousand words, rather than against millions of point descriptors from the original model. A linear search for the first and second nearest neighbors is performed within each word in ascending order of the number of model descriptors assigned in the word. When a desirable number of matches is obtained, the search stops. On top of this work, [62] use active search in the vocabulary tree to prioritize the back matching of 3D points close to those that have already been matched.

All of the mentioned works above try to overcome two main problems of large-scale image localization: speed and repetitive structure. While the former problem is approached by model compactness or vocabulary trees, the latter is still mainly unexplored. The common approach for finding a candidate match (either forward or backward) is still looking for the first and second nearest neighbor in the feature space, and applying the widely used Lowe's ratio test. This method is very restrictive and works well in similar query and search spaces like pairs of images or relatively small 3D models, where the points are sufficiently different from each other so that a significant number of matches pass the ratio test. In denser spaces, the likelihood of two points to have a similar appearance is high, hence the ratio test rejects too many matches and fails in the final fine pose estimation step.

A radical different approach is found in the works of [88, 76]. They frame camera localization as a Hough voting procedure. To do so, they take advantage of the little used geometric properties of SIFT (scale and orientation). Alongside focal length and camera gravity direc-

tion priors, they hypothesize ground planes where each 2D-to-3D match might cast a voting shape based on the cone projected from the matched 3D point and SIFT scale. They use orientation and model co-visibility to filter out unlikely matches. The main advantage is that the voting step is done in linear time in the number of correspondences, while handling a large proportion of outliers. However, they match all query features against all descriptors in the model, resulting in slower localization times.

In this chapter, we propose a fast yet simple method for camera localization that tries to accommodate for large amounts of matches with high outlier ratios while getting rid of the annoyances of complex model simplifications or hard prior requirements (*e.g.*, gravity direction of the camera). We tackle this problem by a coarse-to-fine approach that tries to divide the search space in smaller bins that can handle Lowe's ratio test with higher success, as opposed to a global, more restrictive approach. We formulate a linear time voting step at the match stage that can successfully find locally discriminative correspondences. We then exploit co-visibility to constrain the search for newer matches given a location recognition criteria. Figure 2.2 defines the main pipeline of our approach. Our method is still generic, in the sense that it is applicable in any situation without further priors needed. This is especially useful in cases where the focal length prior might be unreliable due to corrupt EXIF metadata and posterior image manipulation.

2.4 Multiple Nearest Neighbors for Local Ratio Test

In the previous sections, we laid out the main pieces of incremental SfM and standard camera localization. A key observation for the latter task is that the main efforts for improving localization are targeted to *sparsify* the search space in order to obtain more matches passing the ratio test. This can be practical in scenarios where, despite the area covered by the images is large, the features of the different landmarks are relatively unique. However, there are

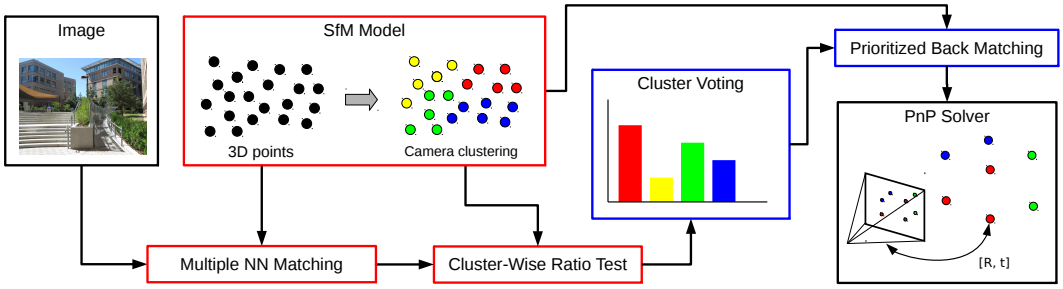


Figure 2.2: Overview of our camera localization pipeline. We exploit multiple nearest neighbor search and leverage camera pose clustering to identify potential landmarks where the query image was taken. The red colored boxes denote the steps defined in Section 2.4. We explore cluster co-visibility to prioritize back matching of model features. This strategy, shown in the blue boxes, is explained in Section 2.5.

Algorithm 2 Proposed Image Localization

Require: Query features Q, Model features $\mathcal{V}$, ratio threshold τ, error threshold ϵ, clusters $\mathcal{C}$, co-visibility graph G, number of NN k, minimum matches N_F, N_B

$\mathcal{M} \leftarrow \emptyset$
while $|\mathcal{M}| < N_F$ **do**
  q = random sample from Q
  $\{(q, v_1), ..., (q, v_{k+1})\} = \text{kNN}(q, \mathcal{V}, k+1)$
  $\alpha_q = \frac{\|q - v_1\|}{\|q - v_{k+1}\|}$
  if $\alpha_q \leq \tau$ **then**
    $\mathcal{M} = \mathcal{M} \cup \{(q, v_1), ..., (q, v_k)\}$
  end if
end while
$\mathcal{M}_\mathcal{F} = \text{CLUSTER-WISE-RATIO-TEST}(\mathcal{M}, \mathcal{C}, \tau)$
$\mathcal{M}_\mathcal{B} = \text{PRIORITIZED-BACK-MATCH}(\mathcal{M}_\mathcal{F}, N_B, G, \tau)$
$I_P, \delta = \text{ROBUSTFITTING}(\mathcal{M}_\mathcal{B}, \epsilon)$
if $|\delta \leq \epsilon| \geq 12$ **then**
  return Camera Pose I_P
else
  return Error - Pose not found
end if

cases where structures are constantly repeated in different locations of the scene (*e.g.*, several buildings of our university look essentially the same). In this case, the efforts for simplifying the search space do not provide a great advantage, as global matching can still lead to the wrong correspondence, far away from where the query image was actually taken from.

2.4.1 Global versus Exhaustive Local Matching

A naive, gold standard way to solve the problem of large-scale ratio testing is simple. We can divide the reconstructed model into several clusters, using any arbitrary technique (*i.e.*, k-means on the camera locations, spectral clustering on the scene graph, etc.). Then, we can perform image matching and localization in each of the clusters exhaustively. While this local matching should work better in practice, it comes at a cost: rather than building one global kd-tree search index, we need to build as many as the number of local clusters we create. This implies that not only we are using more memory resources, but also that the computational cost of matching a query feature becomes bounded linearly by the number of clusters. Without loss of generality, the complexity of a global kd-tree search in a model with N descriptors is logarithmic: $O(log(N))$. If we divide the model into $|\mathcal{C}|$ clusters, its complexity becomes $O(|\mathcal{C}|log(\frac{N}{|\mathcal{C}|}))$. Hence, execution time is dominated by the number of clusters since $O(|\mathcal{C}|log(\frac{N}{|\mathcal{C}|})) > O(log(N))$.

We empirically tested camera localization following this gold standard approach, exhaustively performing matching and localization in different model clusters. We used the Eng-Quad dataset that we collected for [16], and later defined in Chapter 4. Figure 2.3 shows the model reconstructed using COLMAP [66]. We built two different models: one containing only the training images, and a second one containing both training and test images. We geo-registered the resulting reconstructions with a GIS model provided by the university's building management office, so that the scale of the SfM model is approximately metric. We

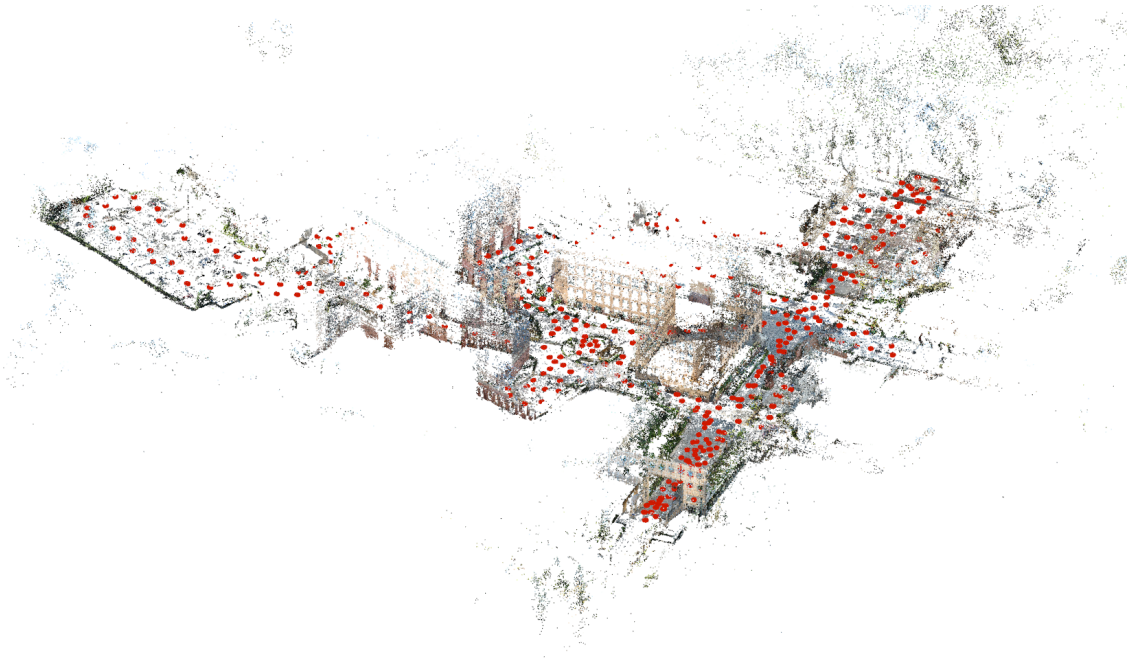


Figure 2.3: Reconstruction of the Engineering Quad dataset, using COLMAP [66]

used the training+test model only to obtain ground truth camera poses for the set of test images. Out of the 570 test images, 520 were resectioned. The model with only training images is used to perform camera localization. Out of the 6402 training images, 5129 of them were resectioned. The resulting point cloud has 579,859 3D points and 2,901,885 views. This dataset has the desirable property that it contains heavily repeated structure due to its brutalist architecture style. This poses a hard challenge not only for the SfM reconstruction itself, but also for large scale camera localization.

Given the scene graph of the SfM model, we perform spectral clustering [68] using the 50 largest eigenvectors of the scene matrix S, which contains at every element $S_{i,j}$ the number of points every image pair I_i, I_j share in the model. We chose three different granularities: no clustering at all (purely global), 50 clusters, and 500 clusters. To evaluate the performance, we use two metrics. First, an image is considered to be correctly registered as long as it has at least 12 inlier correspondences under a reprojection error $\epsilon = 6px$. The second metric measures the actual localization error given the ground truth position of the image obtained

#Clusters	#images	#inl	ratio	med.	fwd [s]	RNSC [s]	Total [s]
1 (global)	463	94	0.57	0.64	0.833	0.129	0.962
50 (exhaustive)	512	66	0.54	0.45	13.10	43.62	56.822
500 (exhaustive)	517	51	0.49	0.29	80.23	523.69	603.52
50 (all)	451	145	0.16	0.63	13.10	3.40	16.50
500 (all)	383	162	0.02	8.20	80.23	8.53	88.76

Table 2.1: Quantitative results on the 520 test image set using the standard localization framework of algorithm 1. We divided the SfM model using spectral clustering. We account for the number of localized images, the inlier ratio and the median error in meters. We also report average timings, in seconds, of forward matching, RANSAC, and total time per image. We show two different approaches: exhaustively run RANSAC on each of the clusters, and running RANSAC once over all accumulated correspondences.

in the training+test SfM reconstruction. We used a P3P solver [41] and RANSAC on each set of cluster correspondences, using as focal length prior the EXIF metadata of the image file.

We show in table 2.1 that this exhaustive strategy performs well, but takes too much time as it needs to loop across all clusters to perform matching and/or localization. We evaluated two possible approaches. First, we exhaustively run RANSAC on each of the clusters correspondences and select the solution with the smallest median localization error across clusters. Second, we accumulate the correspondences of each cluster and run a single instance of RANSAC. As expected, we acquire smaller median errors and localize more cameras under this exhaustive framework, but execution time grows roughly linear with respect to the number of clusters. It is also interesting to see how running RANSAC on the accumulated correspondences performs worse, indicating that some clusters only provide outliers that degenerate the solution.

2.4.2 Local Ratio Test using Global Correspondences

We argue that this exhaustive local matching can be roughly approximated by global matching at its $O(log(N))$ cost. The main reason in favor of this idea comes by the way approxi-

mate nearest neighbor search is performed in a kd-tree structure. When looking for a nearest neighbor, the kd-tree is traversed until we reach a leaf expected to contain it. Then, a back-tracing procedure is performed and several other leaves which are close are also explored to guarantee a minimum precision in the search. This implies that, when looking for a nearest neighbor, we are also visiting the leaves of the nearest second, third, fourth, and subsequent neighbors. When requesting k-NN ($k > 1$), the additional computational effort performed by the search is negligible compared to requesting a single NN, since we already visited those leaves. This particular property of the kd-tree search seems to have had very little attention from the community when performing image localization. While there are some exceptions [88, 87], the vast majority of works still focus on ratio testing the first and second nearest neighbors in a global search.

To this point, exhaustive local matching and ratio test is a gold standard that can lead to better results by paying a high cost in time. However, we know that it is computationally equivalent to search one single nearest neighbor or a few of them. We study how we can take advantage of this NN search trick to approximate global matching to exhaustive local matching with faster execution times. In particular, we want to answer the question: *how informative are the non-first nearest neighbors for large scale camera localization?*. To do so, we define every component of an SfM model that will be used for such purpose. We consider a *view* $v \in \mathcal{V}$ as the 2D point observation of a 3D point $p \in \mathcal{P}$ in a particular model image $I \in \mathcal{I}$. Given a clustering $\mathcal{C}$ of the SfM model images, we assign the view descriptors of each image to their corresponding cluster $c \in \mathcal{C}$. Note that these clusters divide images in disjoint groups, but they do share common points, as a 3D point can have multiple views belonging to images assigned to different clusters.

For a query image I_Q with query features Q, we search for k nearest neighbors using a global kd-tree structure built from all model views $\mathcal{V}$. A global prune of non discriminative correspondences is applied if the ratio test between the first and $k+1$ nearest neighbors

is greater than some threshold τ, as proposed by [87]. Otherwise, all k nearest neighbors are included in the set of putative correspondences $\mathcal{M}$, where each pair match $(q, v) \in \mathcal{M}$ links each query feature with a model view. Note that this *global* ratio test is much more conservative than the standard first vs second NN test. In the remainder of this chapter, we will refer to this global test as *k-ratio* test, defined formally as $\frac{\|q-v_1\|}{\|q-v_{k+1}\|} \leq \tau$. The standard first versus second NN test will be referred as *1-ratio* test.

The k-ratio test allows for more matches to be included in $\mathcal{M}$, especially those that are not first NN. In this scenario, the size of $\mathcal{M}$ is larger than usual, as it accounts for more than just first nearest neighbor matches. On one hand, these extra correspondences can provide good matches since the NN search is approximate and it implies that the third NN could be a better match than the first or second NN. On the other hand, it is also a source of outliers because the more nearest neighbors we search for, the noise-to-signal ratio is likely to increase. We propose to solve this problem by using the model clustering $\mathcal{C}$. For each match (q, v) assigned to a cluster c, we have an immediate *re-ranking* of the nearest neighbor order within such cluster. For instance, two global matches (q, v_2) and (q, v_4) corresponding to a second and fourth NN of the same query feature q can in turn be locally the first (q, v_{c1}) and second (q, v_{c2}) NN in the particular cluster they are assigned to. Figure 2.4 illustrates this re-organization of correspondences.

Lemma 1. *If a candidate match fails the global k-ratio test, it also fails the local 1-ratio test.*

Proof Let $\{v_{c1}, v_{c2}\} \in \{v_1, ..., v_k\}$ be the first and second local nearest neighbors of a particular query feature q. Since the global set $\{v_1, ..., v_k\}$ is sorted by ascending distance, it implies that $\|q - v_{c2}\| \leq \|q - v_{k+1}\|$, and $\|q - v_{c1}\| \geq \|q - v_1\|$. Formally,

$$\frac{\|q - v_{c1}\|}{\|q - v_{c2}\|} \geq \frac{\|q - v_{c1}\|}{\|q - v_{k+1}\|} \geq \frac{\|q - v_1\|}{\|q - v_{k+1}\|} \tag{2.1}$$

Hence, the local 1-ratio will always be equal or greater than the global k-ratio. This guar-

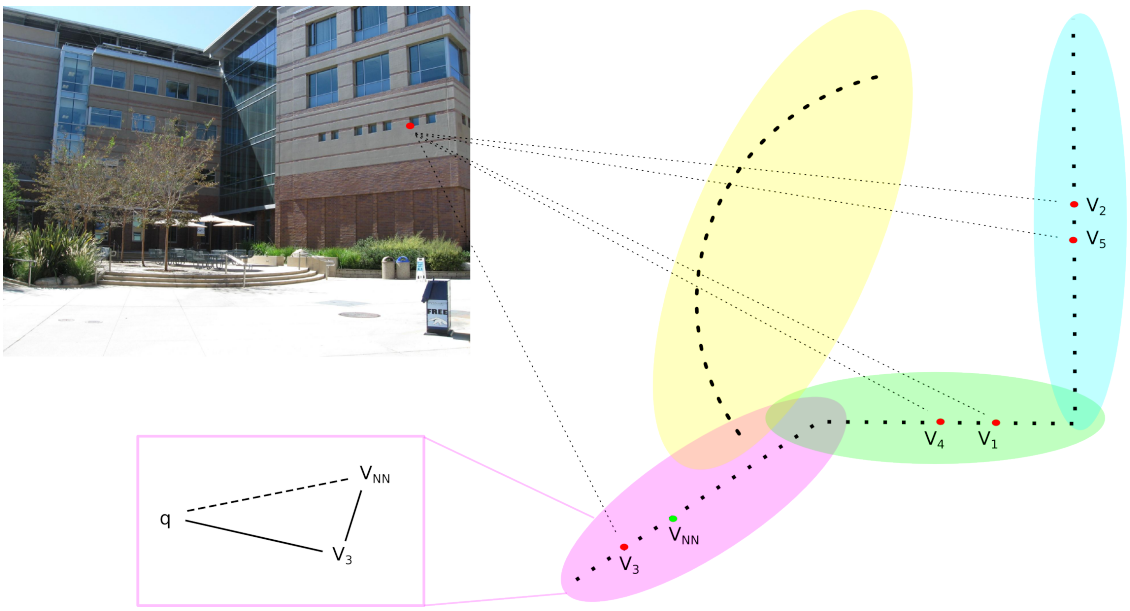


Figure 2.4: Example of our cluster-wise ratio test. Model views are divided into clusters. For a query feature q in the image (red dot), we search up to 5 nearest neighbors. The matches are immediately assigned to their corresponding clusters, where they are re-ranked: second and fifth NN are the blue cluster's first and second, and fourth NN is the green cluster's second. The third nearest neighbor, isolated in the pink cluster, is evaluated following a triangle inequality with respect to the matched view's nearest neighbor v_{NN} of v_3. Ratio test is performed using first and second NN in each cluster whenever possible. The isolated match is ratio tested using equation 2.2.

antees that a correspondence failing the k-ratio test will always fail the local 1-ratio test. A correspondence passing the k-ratio test might not pass the local 1-ratio test, yielding more stringent results. □

Re-ranking correspondences allows us to perform standard 1-ratio test as long as two local views v_{c1}, v_{c2} of a query feature q are found in the same cluster. We ensure that the views belong to different 3D points p_1, p_2. If the number of assigned views is greater than two, the third and following ones are ignored. An ambiguous situation might happen if a single match falls uniquely in a cluster. This correspondence might not be locally discriminative but a second NN local correspondence was not found the same cluster. In this case, we use the training data to generate distance bounds that allow for reliable local testing of these singleton correspondences. In particular, we take advantage of triangle inequality properties to define a ratio test for such cases, which we name *t-ratio* test.

Ambiguous correspondences assigned uniquely to a cluster can be bounded using the model descriptors belonging to such cluster. Given a local correspondence $v_{c1} \in c$, we define $v_{NN} = kNN(v_{c1}, c, 1)$ as the nearest neighbor of such view in the feature space defined by cluster c. Since v_{NN} is obtained purely from training data, we can compute it offline and access it at test time. The distance $\|v_{c1} - v_{NN}\|$ can give an idea of how discriminative the match (q, v_{c1}) is. We define the t-ratio test as

$$\frac{\|q - v_{c1}\|}{\|q - v_{c1}\| + \|v_{c1} - v_{NN}\|} \leq \tau \tag{2.2}$$

Lemma 2. *If a candidate match fails the t-ratio test, it also fails the local 1-ratio test.*

Proof Let q be a query point, v_{c1} and v_{c2} the fist and second local nearest neighbors in a cluster c, and $v_{NN} = kNN(v_{c1}, c, 1)$. We can bound the distance to the second nearest

neighbor by triangle inequalities:

$$\|q - v_{c2}\| \leq \|q - v_{NN}\| \leq \|q - v_{c1}\| + \|v_{c1} - v_{NN}\| \tag{2.3}$$

where the first inequality holds since $\|q - v_{c2}\| \leq \|q - v\| \, \forall \, v \in c \setminus v_{c1}$, and the second holds by the triangle inequality. Thus,

$$\frac{\|q - v_{c1}\|}{\|q - v_{c2}\|} \geq \frac{\|q - v_{c1}\|}{\|q - v_{c1}\| + \|v_{c1} - v_{NN}\|} \tag{2.4}$$

Consequently, an ambiguous match that fails the t-ratio test will always fail the local 1-ratio test, successfully filtering correspondences that would have passed the local ratio test if v_{c2} was available. □

2.4.3 Back matching and fitting

The cluster-wise ratio test defined in algorithm 3 allows pruning a big portion of non-discriminative correspondences while still maintaining the locally unique matches. The complexity of this algorithm is linear in the number of correspondences N_F. For every local NN v_{c1}, we simply look for its second NN pair v_{c2} within the list of k nearest neighbors. The list of intra-cluster nearest neighbors v_{NN} can be pre-computed offline and accessed at constant time. Hence, at most N_F ratio tests will be performed. To additionally enforce robustness, we back match the views of the forward pass correspondences against the features of the query image. We search for the first and second nearest neighbor and apply 1-ratio test. We then select as the final set of correspondences the intersection of pairs (q, v) that passed the forward and back matching process. This implies that these pairs are *best buddies* [14], since each q and v of a pair are both discriminative features in the query and model feature space. The final camera pose is found using RANSAC on this set of correspondences using a P3P solver that takes as prior focal length the one stored in the EXIF metadata of the image file.

Algorithm 3 Cluster-Wise Forward Matching

Require: matches $\mathcal{M}$, clusters $\mathcal{C}$, threshold τ
  $\mathcal{M}_{\mathcal{F}} = \emptyset$
  for all $c \in \mathcal{C}$ with at least one $(q, v) \in \mathcal{M}$ **do**
    for all $(q, v_{c1}) \in c$ **do**
      if $(q, v_{c2}) \in \mathcal{M}$ **then**
        $\alpha_{(q,c)} = \frac{\|q - v_{c1}\|}{\|q - v_{c2}\|}$
      else
        $v_{NN} = kNN(v_{c1}, c, 1)$
        $\alpha_{(q,c)} = \frac{\|q - v_{c1}\|}{\|q - v_{c1}\| + \|v_{c1} - v_{NN}\|}$
      end if
      if $\alpha_{(q,c)} \leq \tau$ **then**
        $\mathcal{M}_{\mathcal{F}} \leftarrow \mathcal{M}_{\mathcal{F}} \cup (q, v_{c1})$
      end if
    end for
  end for
  return $\mathcal{M}_{\mathcal{F}}$

#Clusters	#images	#inl	ratio	med.	fwd [s]	RT [s]	bck [s]	RNSC [s]	Total [s]
1 (global)	481	115	0.74	0.69	0.821	0.008	0.021	0.046	0.895
50	477	127	0.59	0.66	0.818	0.008	0.028	0.061	0.915
500	480	133	0.56	0.61	0.821	0.009	0.038	0.066	0.934
1 im/clus	**482**	136	0.55	0.62	0.833	0.009	0.048	0.070	0.961

Table 2.2: Quantitative results on the 520 test image set using the proposed localization framework of algorithm 3 and best-buddy filtering. We used 5 nearest neighbors in the k-NN search. We account for the number of resectioned images, the inlier ratio and the median error in meters. We also report timings, in seconds, of every part of the pipeline: forward matching, cluster-wise ratio test (RT), back matching, RANSAC and the total time. Back matching time includes the time for building the kd-tree on the query image features.

We evaluated this cluster-wise approach using the same settings as our gold standard baseline experiment. However, we added a finer division of the model, consisting in atomic clusters of a single image each. Table 2.2 shows the localization performance on these different granularities. A single global cluster gives surprisingly good results in the number of localized cameras, although it provides worse camera position results. This is due to the restrictiveness of the ratio test in denser search spaces, yielding fewer inliers and missing some discriminative correspondences that would improve results. As we increase the number of clusters, the localization errors are reduced thanks to the cluster-wise ratio test, providing higher confidence matches. We acquire best results using the finest clustering (a single image per cluster), successfully localizing 482 images. Compared to the gold standard in table 2.1, our localization results are competitive. More cameras are localized under a global approach, and while we do not resection as many cameras using multiple clusters, our strategy is orders of magnitude faster than exhaustive matching.

While these results differ by small amounts on the number of resectioned images, they clearly indicate the benefit of the cluster-wise ratio test by providing more and higher quality matches, improving the localization error up to 8 centimeters over a global matching scheme. However, more clusters increase the time of back matching/RANSAC because the local ratio test is accepting more correspondences. The forward matching step constitutes the main speed bottleneck, as the kd-tree is built on millions of descriptors.

2.5 Faster matching by cluster recognition

If we could figure out beforehand what clusters provide the best correspondences for camera localization, we could speed the pipeline considerably. This can be framed as a location recognition problem. Location recognition can be considered a broader definition of camera localization. It essentially focuses on finding the landmarks where the image was roughly

taken from, without using a precise PnP solver. Typically, the output of a location recognition approach is presented as a short ranked list of model images that should depict the same landmarks as the query image (*e.g.*, same buildings, statues, etc.). From this perspective, location recognition can be used as and additional metric to evaluate our proposed matching pipeline. Clusters can be viewed as a partitioning of the camera pose space, and matches assigned to such clusters are thus votes for the camera location in the world. We study the benefits of treating our cluster-wise algorithm as a location recognition problem in order to speedup the pipeline and improve localization errors.

2.5.1 Location recognition using cluster-wise forward matching

At the finest granularity of 1 image per cluster, correspondences found are more discriminative. In this framework, we define the space covered by the SfM model as a voting space (*e.g.*, a histogram) in which matches are the votes we cast in every cluster. Intuitively, the model clusters containing more votes should be those depicting the same landmarks as the query image. To evaluate this intuition, we perform an experiment similar to [10]. For every query image, we rank the model images in descending order of the number of correspondences that passed the cluster-wise ratio test of our approach. We then measure if there exists at least one image among the top k images in the ranked list that is considered to depict the same landmark as the query image. To determine if two images have overlap, and hence were taken roughly at the same position, we compute the epipolar geometry of the pair by fitting a fundamental matrix. If this fitting returns more than 12 inlier points, the image pair is considered to depict the same landmark. We benchmark our experiment in two different datasets: Eng-Quad and Dubrovnik [44].

The results in table 2.3 are inspiring. Algorithm 3 is able to recognize the location of all 800 test images in the Dubrovnik dataset using 200 random features passing the k-ratio

Dubrovnik - 800 test images

Method	top-1	top-2	top-5	top-10	Time [s]
$N_F = 50$	99.00%	99.38%	99.62%	99.88%	0.048
$N_F = 100$	99.62%	99.75%	99.88%	99.88%	0.085
$N_F = 200$	100%	100%	100%	100%	0.157

Eng-Quad - 520 test images

Method	top-1	top-2	top-5	top-10	Time [s]
$N_F = 50$	83.85%	85.96%	88.46%	88.65%	0.064
$N_F = 100$	84.62%	86.54%	89.62%	90.96%	0.125
$N_F = 200$	85.77%	87.69%	90.38%	91.35%	0.242
$N_F = 300$	85.38%	87.88%	90.38%	91.54%	0.345
$N_F = 400$	85.77%	88.08%	90.58%	91.35%	0.432
$N_F = 500$	86.15%	88.27%	90.96%	91.35%	0.502
All features	86.92%	89.23%	91.15%	91.92%	0.833

Table 2.3: We acquire perfect results on location recognition in the Dubrovnik dataset using just a random subset of 200 features that pass the k-ratio test, suggesting that our cluster-wise ratio test successfully finds local discriminative correspondences for all 800 test images. We also obtain good results in the more challenging Eng-Quad dataset. Out of the 520 test images, 478 (91.92%) contain at least a good image in the top 10 ranked list. This agrees with the baseline results obtained in table 2.2. The performance of using a random subset of query features is slightly worse.

test. Results on the more challenging Eng-Quad dataset provide almost 92% accuracy on recognizing the landmarks of the 520 query images for which we have a ground truth pose. It is also interesting to see that a random subset of the query features acquires almost as good recognition results as using all image features (a query image usually has between 5,000 and 10,000 features). This suggests that the forward matching can be significantly faster while still maintaining good recognition performance. However, using a random subset in the whole pipeline will lead to few correspondences, and it would be likely to localize less query images or have a larger position error.

2.5.2 Prioritized Back Matching

In order to maintain the speed benefits of subsampling, we shift our strategy to a framework where forward matching is used for recognition purposes, and where back matching takes the lead on finding the correspondences that will be used for fine camera pose estimation. We propose a greedy algorithm that exploits SfM co-visibility and landmark location recognition

to constrain back matching to the local subspace of the scene where the image was taken. Motivated by the results of table 2.3, we believe that the fine pose estimation step (RANSAC) should be applied only to those correspondences that belong to the top ranked images. This is, in fact, a very intuitive reasoning: recognizing the landmark where the query image was taken from should geometrically constrain the set of model points used for fine pose estimation. This strategy also raises questions about the correspondences used at back matching. When forward matching, the correspondences found do not cover all views of the most voted images (especially when we randomly subsample the query features). That does not imply that those views are not discriminative, but that they were simply not matched due to subsampling. Instead, back matching all views of a top ranked image against the query feature space should help discover these correspondences that forward matching could not find. At the same time, we can exploit the model's bipartite visibility graph to discover additional nearby images depicting the same query points.

Algorithm 4 describes our greedy approach. It requires a list of forward matched correspondences using our cluster-wise ratio test from algorithm 3. Since there is no need to use all query features, we randomly sample a sufficient amount such that at least N_F correspondences pass the global k-ratio test. We start by selecting the most voted cluster (or image if we use maximum granularity) c^* and back match all of its views against the query space using standard 1-ratio test with threshold τ. The correspondences M_{c^*} found are added to the pool of back matched pairs $\mathcal{M}_{\mathcal{B}}$ and will be used for the fine pose estimation. These matches are in turn also used as votes. Let G be the scene graph relating the clusters/images that observe common points in the SfM model. We use G to add votes to other clusters that observe the same views as in M_{c^*}. These new votes help identify and increase the likelihood of neighboring clusters to be selected next in the greedy loop. To avoid degenerate solutions based on a wrong set of top ranked images, we only update the histogram of votes H using the same criteria used for location recognition. Since the model views have already been geometrically validated in the SfM reconstruction, we cast new votes if and only if a back

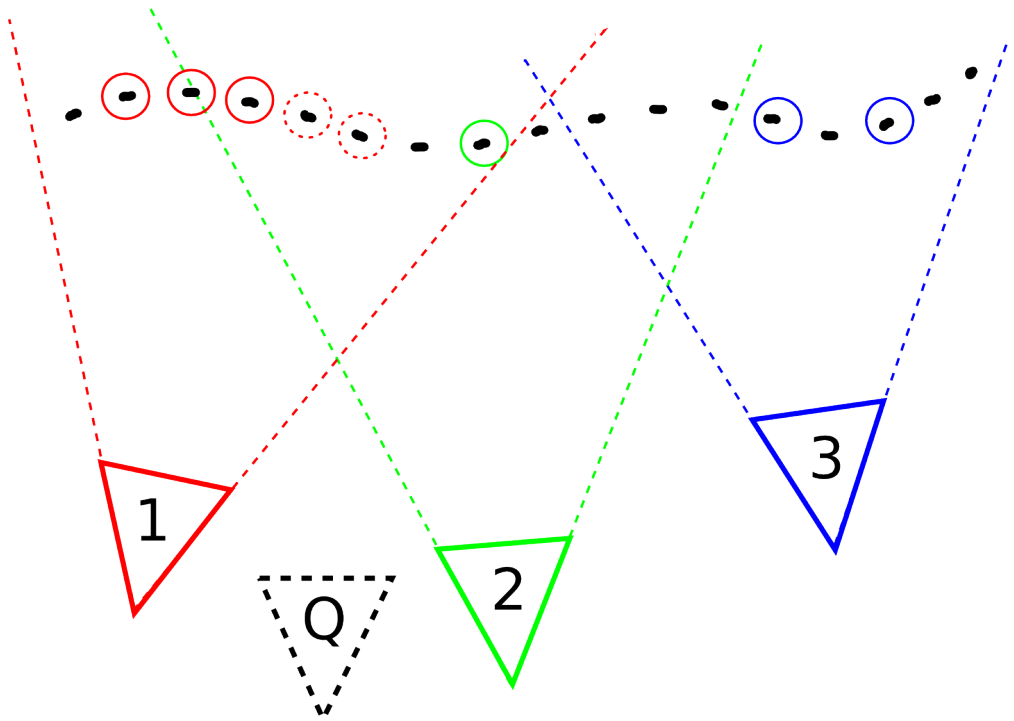


Figure 2.5: Toy example of our prioritized back matching. We greedily select image 1 (red) as it has more forward matches (solid circles) than image 2 (green) and image 3 (blue). After back matching the 3D points visible in image 1, two new matches are discovered (dotted circles) that are linked with image 2 in the scene graph. These new matches are accounted in the priority queue. The priority of image 2 increases from one vote to three, so it will chosen next in our greedy loop instead of image 3, which initially had two votes. This back matching prioritization quickly identifies the landmark location of the query image (dashed in black) to find better discriminative correspondences.

matched cluster returns more than 12 matches. The algorithm stops when $\mathcal{M}_\mathcal{B}$ is sufficiently large to guarantee a good camera localization, or a certain number of clusters have been back matched to prevent slow runtimes. Figure 2.6 shows qualitative benefits of our prioritized approach, successfully identifying similar images to the query and discarding those which depict a totally different landmark in the scene.

Algorithm 4 Prioritized Back Matching

Require: list of forward matches $\mathcal{M}_\mathcal{F}$, clustering $\mathcal{C}$, query features Q, threshold τ, minimum number of matches N_B, scene graph G

```
H = Histogram of votes such that H_c = |(q, v) ∈ M_F : v ∈ c|
M_B ← ∅
VC ← ∅ /* Visited Clusters */
while (|M_B| < N_B) ∧ (|VC| ≤ 20) do
  c* = argmax_{c∉VC} H
  M_{c*} ← ∅
  for all v ∈ c* do
    q_1, q_2 = kNN(v, Q, 2) /* Back match model views in query space */
    α_v = ||v−q_1|| / ||v−q_2||
    if α_v ≤ τ then
      M_{c*} = M_{c*} ∪ (q_1, v)
    end if
  end for
  M_B = M_B ∪ M_{c*}
  if |M_{c*}| ≥ 12 then
    for all (q, v) ∈ M_{c*} do
      for all c' = {c ∈ C : v ∈ c} do
        H_{c'} = H_{c'} + 1
      end for
    end for
  end if
end while
return M_B
```

2.6 Experimental results

We use two different PnP solvers in all experiments. We use P3P [41] when using a prior focal length extracted from EXIF metadata or the SfM model itself. Alternatively, we also

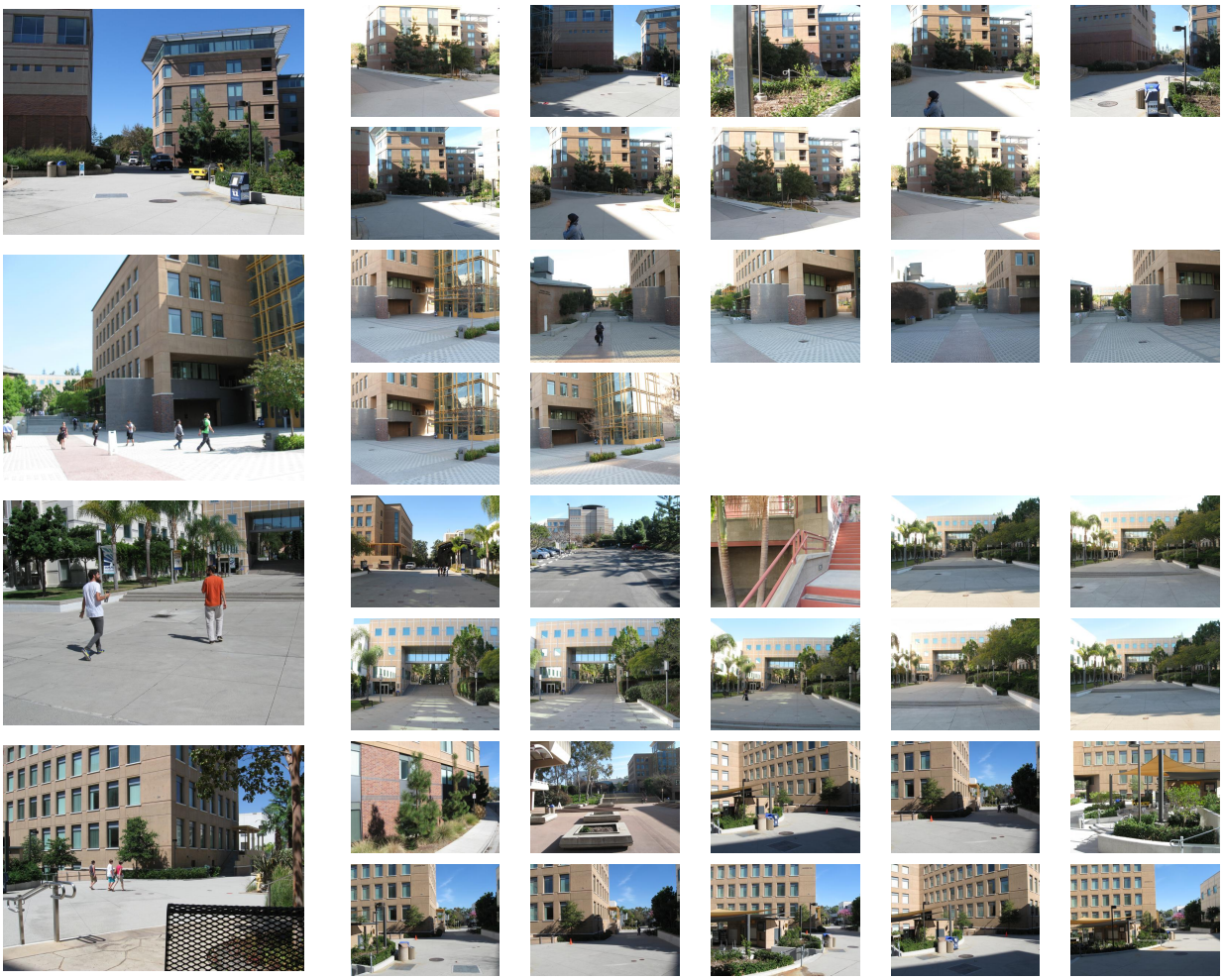


Figure 2.6: For every query image, we show two lists of the most voted model images. Top row shows the images with most votes after using our cluster-wise ratio test. Bottom row shows the most voted images after our prioritized back matching. Images that do not depict the correct landmark are quickly moved down the list using our visibility-based update. We successfully identify the correct images that produce high quality correspondences for fine pose estimation, often needing just a couple of images to successfully find a good camera pose.

use P4Pf [8], which tries to additionally estimate the focal length of the query image. We evaluated three different datasets: Eng-Quad, Dubrovnik, and Rome. When using P3P, we selected EXIF focal priors for Enq-Quad. Focal distances from the SfM models were used in Dubrovnik and Rome, as not all query images had EXIF metadata. Eng-Quad is perhaps the most challenging of the three despite having fewer images and 3D points. Its repetitive structure suppose a hard challenge for image localization. Rome is a large dataset of 16,000 training images, 1,000 of which are used for test. The Rome model is not reconstructed in a single piece, but rather in several components depicting different landmarks of the city. Finally, Dubrovnik is a popular 6,844 image dataset whose SfM model is roughly aligned to geographic coordinates, which makes all measures metric and poses a gold-standard for camera localization.

Dubrovnik correctness : After carefully analyzing the original model provided by the authors, we found out several inaccuracies in the Dubrovnik dataset. Both models (training+test and training only) reported misleading focal lengths, orders of magnitude larger than what they should, and incorrect camera positions. In addition, the test image features reconstructed in the training+test model did not correspond to the actual SIFT keypoints provided, which in turn degrades the co-visibility graph. For all of this, it was hard to evaluate our localization approach, as the dataset does not truly reflect the ground truth structure of Dubrovnik. Works like [63] illustrate this misleading data, as they report better results using P4Pf than using P3P with these wrong focal length priors. Their results are contrary to what it should be expected: knowing the ground truth focal and using P3P should always be better than having to estimate it using P4Pf.

For this reason, we reconstructed a new version of Dubrovnik using [66]. We used the same original SIFT keypoints provided by the authors. While we could not reconstruct all images, the models are sound and all images correctly report focal lengths and co-visibility. We

aligned this new model with the original one using a RANSAC-based Procrustes analysis to have a metric reference for our evaluations. After the alignment, only 3853 out of the 6844 images were located within 3 meters or less from their corresponding position in the original model. A total of 777 of the 800 query images obtained a camera pose in the training+test model, which we use to evaluate our localization pipeline. The Rome dataset also suffers from the same inaccuracies as Dubrovnik. However, evaluations performed in this dataset account for the number of cameras localized rather than precise camera pose, since Rome is not geo-registered and split in disjoint models of different landmarks.

Anytime performance: Our algorithm for camera localization depends on two main speedup factors: the number of minimum forward matches N_F to perform location recognition, and the minimum number of back matches sufficient to apply fine pose estimation N_B. These two parameters trade off localization performance with faster execution times. Figure 2.7 shows the influence of these variables using the Eng-Quad dataset. We benchmarked forward matching times by randomly sampling query features until a desired number N_F pass the global ratio test under three different values for N_B. Similarly, we fix N_F on these same three values and evaluate different values for N_B in algorithm 4. In both cases, the range of values tested vary from 50 up to 500 matched features. Figure 2.7 shows the number of registered images under these different configurations, and the time spent to achieve such performance.

Image localization: We tested our localization pipeline in the aforementioned datasets using the following settings. For every dataset, we built a global kd-tree search index using all model views. 128 leaves are checked when requesting for $k = 5$ nearest neighbors. We chose a threshold of $\tau = 0.7$ across all of our ratio tests (global, cluster-wise, and back match). We set $N_F = 200$ and $N_B = 200$ to provide a good balance between camera localization and speed performance. To prevent algorithm 4 to loop excessively across training images,

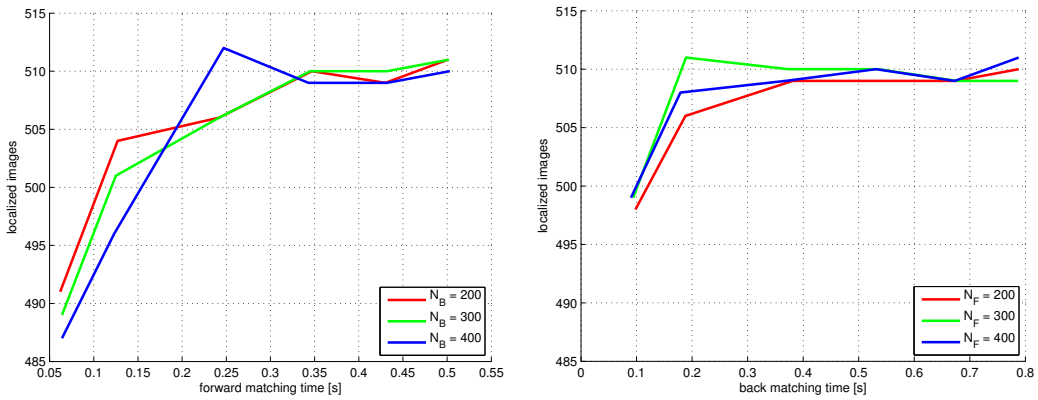


Figure 2.7: Anytime performance: querying a small number of features dramatically reduces runtime without a major loss in localization performance. The forward subsampling does not affect location recognition significantly, and stabilizes after 0.3 seconds (left) regardless of the N_B value. Similarly, localization quickly plateaus after 0.2 seconds at back matching time for different values of B_F (right).

we stop after at most 20 images have been evaluated. Experiments were performed using a single thread on an Intel i7-5930 CPU at 3.50GHz. We compare our results with [61, 62], as they provide an implementation of their localization procedure. Their code was executed in a single thread on an Intel i7-3770 CPU at 3.40GHz on the Eng-Quad and corrected Dubrovnik datasets. We used the generic vocabulary tree available with their code.

Results in table 2.4 show the performance of our approach. We acquire state of the art results in all three datasets evaluated. We successfully localize all test images in Rome using P4Pf, and we only drop 1 image when running P3P using the focal priors provided in the SfM model (the focals in Rome are not reliable). We localize all query images in the original Dubrovnik dataset. The worse localization errors are due to the underlying defects of the model, which are more visible when obtaining more inliers. We also localize all query images in the corrected version of Dubrovnik using the model's focal lengths, while dropping 1 image when this focal length prior is not used. We acquire the smallest median localization errors as well as in the first and third quartiles. We also obtain a much higher amount of images localized under the $18.3m$ threshold, and only one large localization error beyond the $400m$ mark.

Eng-Quad - 520 test images

Method	#images	#inliers	ratio	Q1	Median	Q3	time [s]
Sattler [61]	402	43	0.49	0.61	2.01	7.51	1.52
Sattler [62]	457	43	0.58	0.46	1.93	7.62	0.32
Ours (P3P)	509	112	0.66	0.33	0.67	1.47	0.69
Ours (P4Pf)	504	115	0.68	0.65	1.88	5.76	0.85

Dubrovnik (Corrected) - 777 test images

Method	#images	#inliers	ratio	Q1	Median	Q3	$< 18.3m$	$> 400m$	time [s]
Sattler [61]	771	70	0.72	0.57	1.44	4.61	707	1	2.58
Sattler [62]	775	69	0.74	0.59	1.58	4.91	705	4	0.75
Ours (P3P)	777	591	0.88	0.33	0.66	1.60	759	1	0.48
Ours (P4Pf)	776	589	0.88	0.47	1.11	3.32	720	3	0.48

Dubrovnik (Original) - 800 test images

Method	#images	#inliers	ratio	Q1	Median	Q3	$< 18.3m$	$> 400m$	time [s]
P2F [44]	753	-	-	7.5	9.3	13.4	655	-	0.73
Sattler [61]	783.9	≤ 100	-	0.4	1.4	5.9	685	16	0.31
Sattler [62]	795.5	≤ 100	-	0.4	1.4	5.3	704	9	0.25
Zeisl [88]	798	-	-	0.75	1.69	4.82	725	2	3.78
Ours (P3P)	800	358	0.65	1.09	7.92	27.76	550	10	0.62
Ours (P4Pf)	800	468	0.79	0.55	1.64	6.02	694	15	0.62

Rome (Original) - 1000 test images

Method	#images	#inliers	ratio	time [s]
P2F [44]	924	-	-	0.87
Sattler [61]	976.90	≤ 100	-	0.29
Sattler [62]	991	≤ 100	-	0.28
Ours (P3P)	999	281	0.54	0.75
Ours (P4Pf)	1000	458	0.83	0.74

Table 2.4: Quantitative results with respect to related work on camera localization. We report the median, first, and third quartile errors in meters with respect to the ground truth position.

The most inspiring results are the ones obtained in Eng-Quad, due to its difficult characteristics. Compared to [61, 62], we successfully register more than 100 and 50 images respectively, improving all localization errors except the first quartile when using P4Pf. Contrary to the argument exposed in [88], we obtain smaller localization errors when retrieving more inlier correspondences. Our approach is generally faster compared to [44, 88]. Timings differ with respect to [61, 62]. On one hand, our prioritized back matching algorithm adapts to the more difficult Eng-Quad, spending more time retrieving images with sufficient matches as shown in Figure 2.6. On the other hand, we quickly identify sufficient matches in Dubrovnik with the first or second top ranked images, yielding in faster localization times.

2.7 Discussion

Alternatives to global ratio testing for camera localization have usually focused on reducing the density of the global search space. These approaches are effective but tend to forget that other nearest neighbors beyond the first can be useful to find discriminative features. Our work proposes to shift this line of thought towards the belief that these additional nearest neighbors can be useful for large scale image localization if ratio-tested locally. Additionally, we demonstrated that a subset of the query features is enough to find landmarks in the test image, providing faster runtimes. From this perspective, we approached camera localization as a landmark recognition problem, providing a coarse to fine strategy that yields accurate correspondences with high inlier ratios by exploiting the co-visibility graphs of structure from motion models.

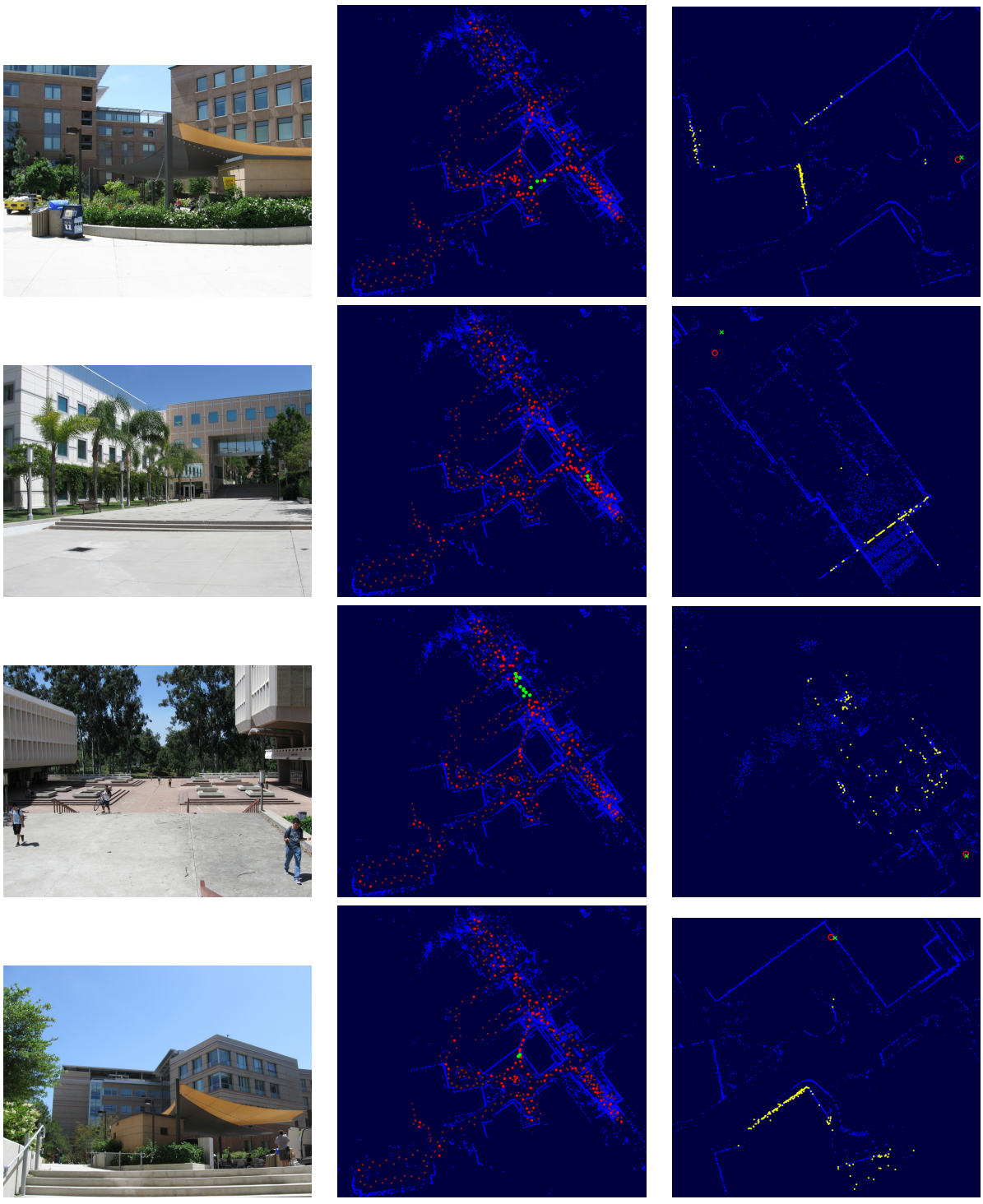


Figure 2.8: Qualitative results for the Eng-Quad dataset. Model images with at least one vote from our cluster-wise ratio test are highlighted in red. Our prioritized back matching algorithm quickly recognizes those images that depict the same landmark (green) with sufficient correspondences to find a camera pose. These correspondences (yellow) are used for fine camera pose estimation. A green cross indicates the localized camera with respect to the ground truth position, depicted as a red circle.

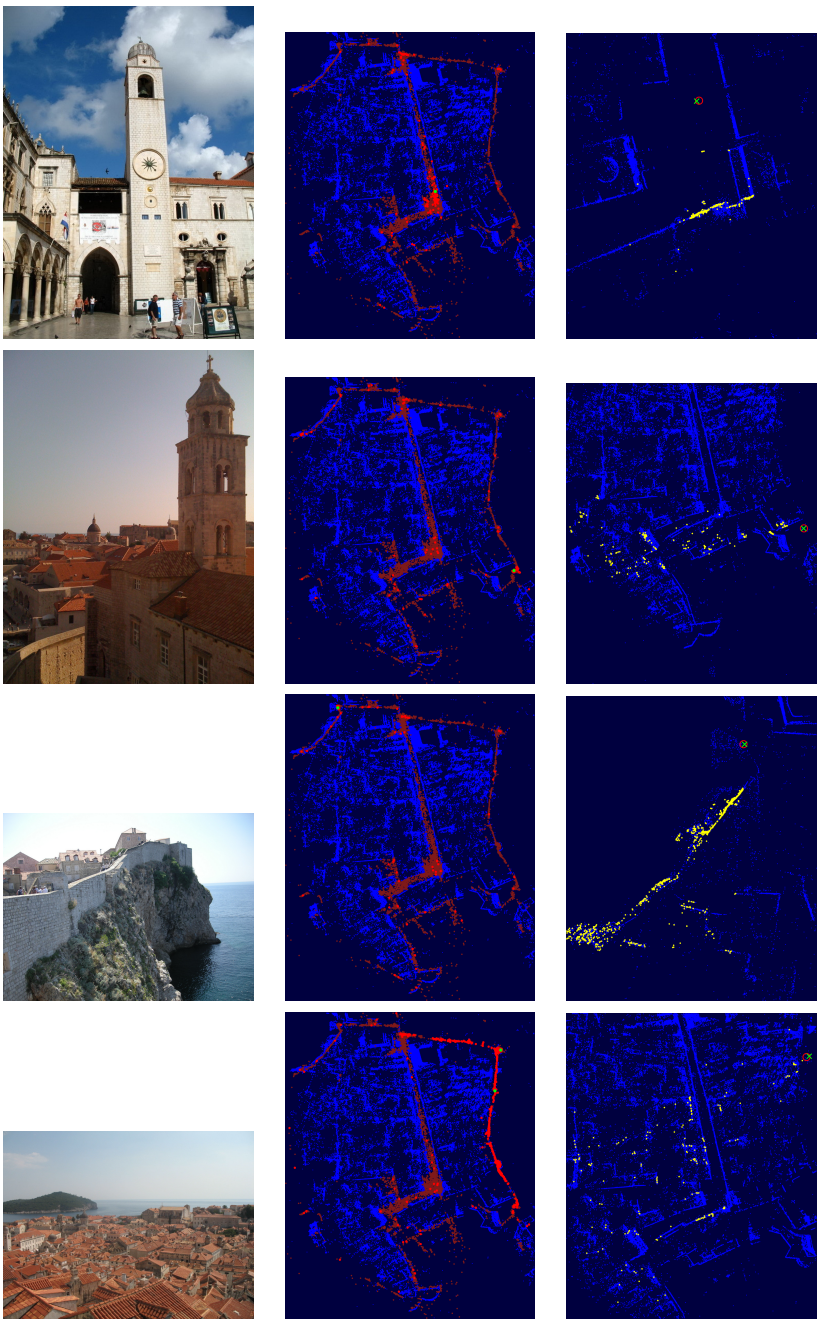


Figure 2.9: Qualitative results for the Dubrovnik dataset. Model images with at least one vote from our cluster-wise ratio test are highlighted in red. Our prioritized back matching algorithm quickly recognizes those images that depict the same landmark (green) with sufficient correspondences to find a camera pose. These correspondences (yellow) are used for fine camera pose estimation. A green cross indicates the localized camera with respect to the ground truth position, depicted as a red circle.

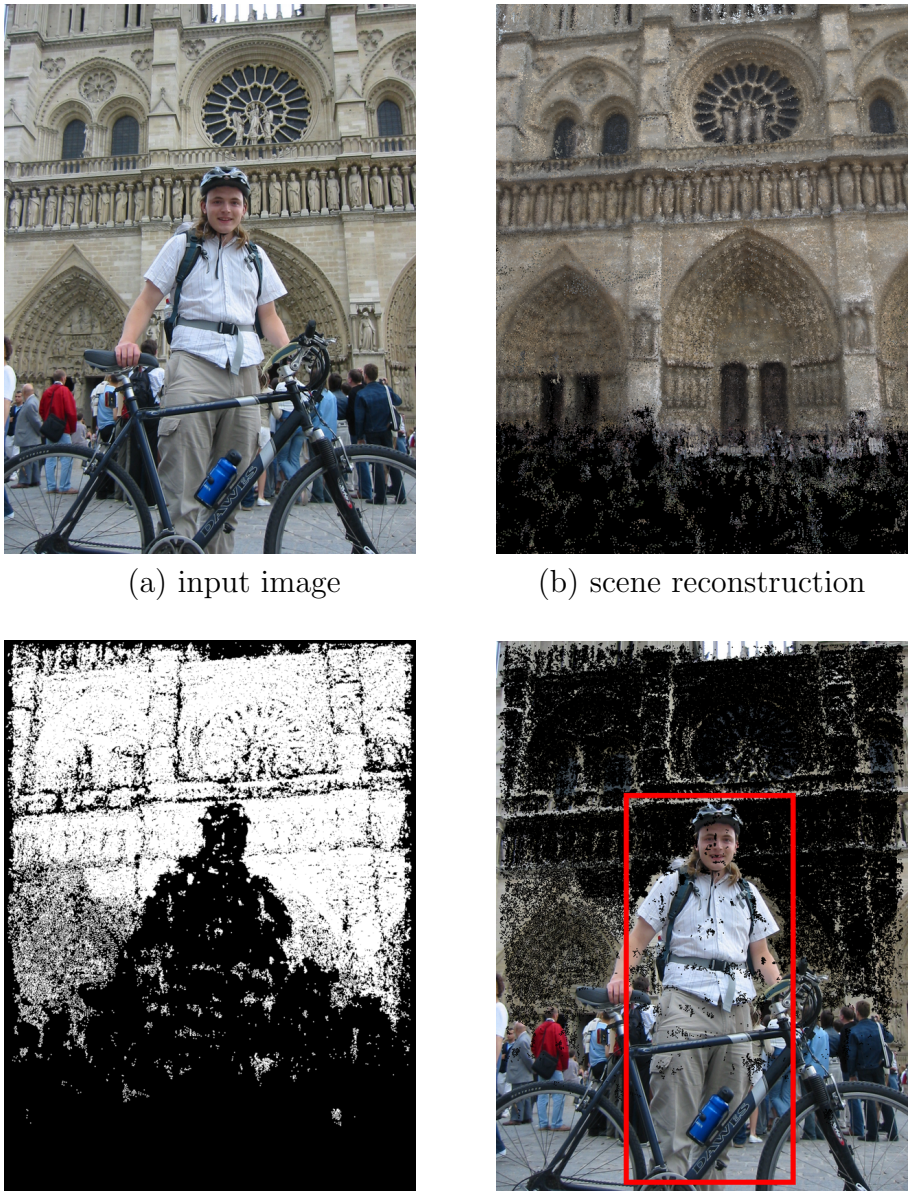
Chapter 3

Detecting Dynamic Objects with Multi-View Background Subtraction

3.1 Introduction

Consider an image of a popular tourist destination shown in Figure 3.1. How can we exploit the large set of photographs available online depicting this same general location in order to better understand the content of this particular image? It is useful to divide scene components into two categories: **dynamic objects** such as people, bikes, cars, pigeons or street vendors that move about and are likely to only appear in a single image taken at a particular time and **static backgrounds** such as buildings, streets, landscaping, or benches that are visible in many images taken in the same general location.

We propose two different approaches that utilize static scene analysis for detection. The first is to perform unsupervised analysis of a large set of scene images in order to automatically train **scene-specific object detectors**. At test time, if we have rough camera localization (*e.g.*, GPS coordinates), we can invoke the appropriate scene-specific detector rather than a



(a) input image (b) scene reconstruction

(c) background mask (d) detected foreground

Figure 3.1: Wide-baseline matching to a collection of photos provides estimates of which pixels belong to static background regions. (a) shows an input image and (b) shows the re-projection of a 3D model built from other images of the same scene. While this model is not realistic enough to allow for direct comparison of pixel values, because we have 3D structure we can easily compare the appearance of local image patches in the input image to corresponding patches in the image set used to build the model. (c) shows those patches for which this match score was above a given threshold. (d) based on this parsing of the scene into static background and dynamic foreground objects, we can eliminating spurious false-positives and improve object detector performance.

generic detector. It seems obvious that an object detector trained with data from a specific scene has the potential to perform better than a generic detector since it can focus on modeling specific aspects of a scene which may be discriminative. If resources are available to perform ground-truth labeling for images collected from every possible scene location, we could simply use existing methods to train a large collection of specialized detectors (one for each object category appearing in each possible scene). However, this is not a scalable solution as it requires labeling positive examples in each possible scene as well as training a huge bank of object detectors. Our key observation is that while acquiring scene-specific positive training instances is expensive, it is possible to automatically produce large quantities of scene-specific negative training instances in an unsupervised manner by identifying portions of a scene that are likely to be static background.

The second approach which we term **multi-view background subtraction** is inspired by a classic trick used to analyze video surveillance data or webcam image streams. When a scene is repeatedly imaged by a fixed camera, one can build up a model of the scene background (*e.g.*, by computing the median color of a pixel over a sequence of images) and compare it to a new image (subtraction) in order to segment out regions that are likely to correspond to dynamic objects of interest. Unfortunately, such a model is tied to the pixel coordinate system and hence offers little help for understanding a new image taken from a novel viewpoint or with a different camera. If instead we model the static background in world coordinates (*e.g.*, as a high-quality 3D mesh) and accurately estimate the camera pose for a test image, we can render the appropriate background image and perform subtraction as before to identify static and dynamic image regions. While this might have seemed like an unrealistic idea even ten years ago, the availability of robust algorithms for SfM and MVS along with high-coverage mapping and imaging of the world suggests that high-quality 3D models and precise camera localization of novel photos will be soon be commonplace for a variety of scenes, particularly urban outdoor environments (see, *e.g.*, [61]).

At their core, both of these approaches tackle the same problem of modeling static background for a scene. Scene-specific object detectors implicitly contain a model of the scene background derived from negative training examples. Since the detectors are used in a sliding window fashion, this model of the background is translation invariant and must function well at any image location. Multi-view background subtraction goes one step further by synthesizing a spatially varying model of the background. The detector then competes with the background model in order to explain the image contents at each image location. A key distinction is that the former works during training to generate a large collection of object detectors while the later demands more substantial test-time inference. In our experiments, we find both approaches useful and often provide independent benefits in detection performance.

In the remainder of the chapter we discuss the SfM and MVS tools we use to analyze image collections, give specifics of the scene-specific background model and multi-view background subtraction approaches, and finally describe a set of experiments evaluating their efficacy. We conclude with a brief discussion of related work.

3.2 Isolating Backgrounds with Multi-View Stereo

We propose to use large photo collections in a unsupervised manner to build up a model of the static rigid background appearance in a given scene. Such photos will necessarily contain non-static objects but these can generally be rejected as they are not consistent from one photo to the next. Our basic technical tools are robust structure-from-motion and multi-view stereo.

3.2.1 Recovering Camera Pose and Scene Geometry

There is a large body of work which has emerged over the last few years on the problem of camera localization and large scale SfM [73, 33, 1, 22]. We use an off-the-shelf software pipeline to reconstruct the static scene in which our objects are placed. After computing SIFT descriptors for a collection of image keypoints [46], we use Bundler [71] which performs sparse keypoint matching and bundle-adjustment in order to estimate scene structure and camera pose from a large collection of un-calibrated images. Once camera poses have been estimated, we use PMVS [25] to perform dense reconstruction using multi-view stereo. Since stereo matches do not need to be computed across all pairs of views, we use CMVS [24] in order to perform view clustering prior to running PMVS which significantly increases the speed of reconstruction. This generally yields high quality reconstructions like that one shown in Figure3.1(b).

When presented with a novel test image, we would like to similarly estimate the camera calibration and pose. This can again be accomplished using SIFT keypoint matching to find correspondences and standard methods for camera calibration from epipolar geometry. Since our dataset is small we use simple batch processing with Bundler. In a real system, such matching can be carried out incrementally with high accuracy and accelerated with fast indexing in order to scale perform matching to large world-wide datasets [61, 43]. For example [43] demonstrate rapid indexing millions of images and tens of millions of keypoints.

3.2.2 Identifying Background Pixels

Given a high-quality 3D model of a scene and a known camera pose and calibration, it is straightforward to synthesize an image from that viewpoint as shown in Figure3.1(b). Comparison of this re-projected scene with the actual image should indicate which pixels that differ from the static scene and hence are likely to be dynamic objects of interest.

Unfortunately, simply computing the difference between the re-projected image and the test image does not work well in practice. While the models generated by the above pipeline are quite compelling, they are not pixel-perfect. Renderings of point cloud models typically lack fine-scale features that provide cues for object detection. Furthermore, because the library of images used in constructing the model are taken across a huge range of lighting conditions with different cameras, even when the recovered geometry is perfect, the estimated average color may be quite different than the particular color that appears in a novel test image.

While one could develop an image differencing scheme that is robust to these variations, we observe that the problem of *determining when an image from a novel viewpoint matches a model is exactly the problem that multi-view stereo algorithms are designed to solve!* Rather than comparing an image patch to the model, we compare it directly to the appearance of corresponding patches in the images in our test library. We describe the basic matching function we use and refer the reader to [25] for more details.

Consider a point p on our scene reconstruction which is predicted to be visible in our test image I. Let $V(p)$ be the set of all images in our image collection that depict this same point (including only those views where the point p is visible based on the reconstruction). We compute a measure of photometric discrepancy between the image collection and our test image given by

$$match(p) = \frac{1}{|V(p)|} \sum_{J \in V(p)} h(p, I, J) \tag{3.1}$$

where $h(p, I, J)$ compares the color at a set of points sampled from a local plane tangent to the static background reconstruction at p and projected into the test image I and each other image J using the recovered camera poses. These sample points lie on a 5x5 grid on the tangent plane. Their color is estimated from each image using bilinear interpolation and the colors compared using normalized correlation.

Using this match score we generate a **background score map** that indicates the quality of match for each pixel to the images in the dataset. Where appropriate, we can threshold this score map $match(p) > \alpha$ to yield a binary **background mask** as shown in Figure 3.1(c). In our experiments we used a threshold correlation score of $\alpha = 0.5$ but the system performance is very robust to this choice (Figure 3.2). To compute patch matches in our test and training we used a modified version of the publicly available PMVS software [25, 23] which implements the necessary patch matching functionality in order to compute background masks. Note that while we use the same discrepancy function as PMVS, our goal is slightly different. Stereo reconstruction only requires finding a few high-scoring matches across the whole image set in order to estimate geometry and color of the point p. Once it has found enough views, it is free to ignore many images in which p may actually be visible. In our case, we would like to estimate a dense collection of match scores over the entire surface visible from the test image even if this particular test image does not offer the best match for the point. Additionally, our choice of maximal discrepancy threshold used in producing the mask is higher than would typically be used in matching in order to provide denser support in each individual image.

In the following two sections, we describe two different ways in which to use this background mask. First, at training time to generate negative training examples in an unsupervised manner. Second, at test time to prune detection responses which fire on regions that are estimated to be background.

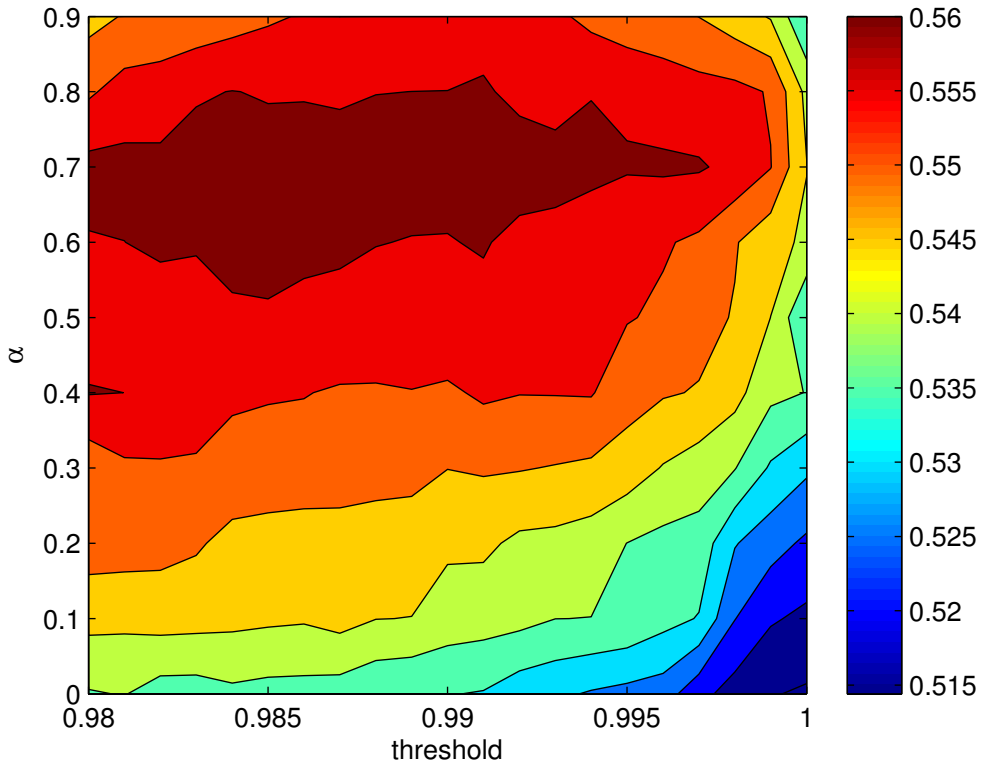


Figure 3.2: Contour plot of empirical tests. α indicates the value used for PMVS reconstruction of the background model. The threshold shows the percentage of true positive training windows kept given a certain background threshold.

3.3 Training Detectors with Scene Specific Background Models

A standard approach to detection is to train a sliding window classifier that distinguishes the object of interest from background. We propose to use the information about the scene derived automatically from a collection images of that scene in order to tune the detector to perform better in that particular context. This can be accomplished by selecting negative training instances from those regions of the image that are expected to be background based on the match score (Equation 3.1).

Rather than including all possible negative windows of an image, we utilize a standard approach of hard-negative mining [19] in order to generate a concise collection of negative instances with which to train the detector. Given an initial estimate of the template (*e.g.*, derived from a generic training set), we run the detector on images (or parts of images) known not to contain the object. Any location where the detector responds at a level greater than the SVM margin specified by the current weight vector is added to the pool of negatives as it may constitute a support vector. This process of hard-negative mining and retraining of the classifier are interleaved until no further negatives are found at which point the final weight vector is computed.

When ground-truth annotations of positives for a scene specific dataset are available, hard negative mining can easily be used by just dropping any candidate negative windows that overlap significantly with a ground-truth positive. However, labeling images is a labor intensive process. Instead we use the background mask as a proxy that can be produced in an unsupervised manner. Let B_i be the set of pixels inside a bounding box associated with a

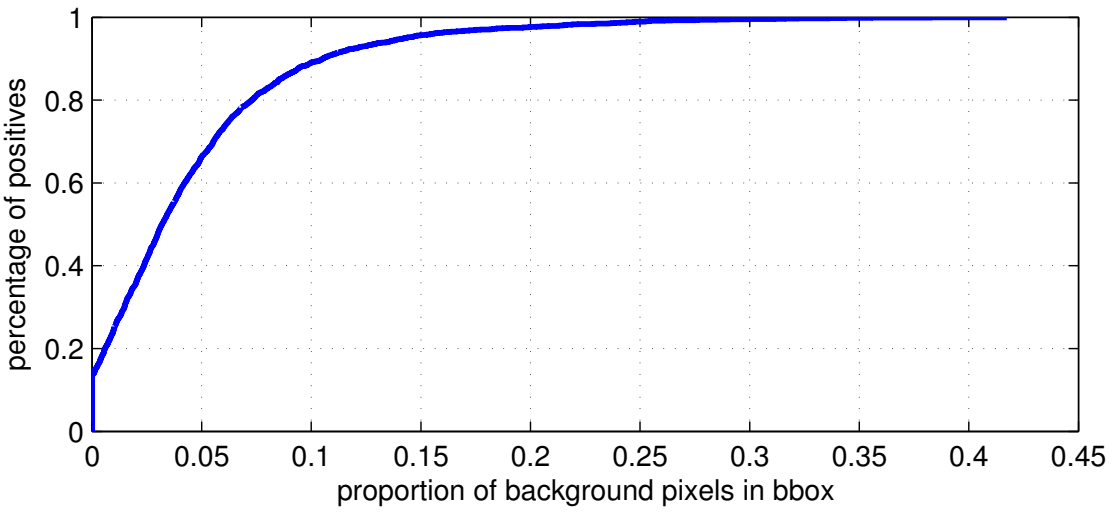


Figure 3.3: Cumulative distribution of the proportion of background pixels $q(i)$ inside true-positive object instances in the scene specific training set.

candidate detection i. We compute the proportion of background pixels in this region as:

$$q_i = \frac{1}{|B_i|} \sum_{p \in B_i} (match(p) > \alpha)$$

Figure 3.3 shows the distribution of the background mask proportions, q_i, over the set of true-positive detections in our training dataset. We use a conservative criteria, declaring a candidate window background if $q_i > 0.2$. As can be seen in the figure, by and large positive instances are not confused with static background. Using this criteria, less than 3% of the ground-truth positives are incorrectly judged as part of the background by this criteria.

3.4 Multi-View Background Subtraction

Background-subtraction has long been used in the surveillance community as it provides a useful approach to scene segmentation (see, *e.g.*, [74]). A closely related problem is that of video stabilization which yields background subtraction in the case of video with relatively

high frame rates [67]. Our scenario differs in that the images we consider may be taken from very different cameras, sparsely sampled in time, different lighting, etc., which make tracking-based approaches used in video inappropriate. Instead we use SfM to estimate the camera parameters of a novel test image and then utilize the same technique described in Section 2 for identifying background pixels, namely those that are photo-consistent with our model and image collection.

For a novel image, we can view the background mask as a hypothesized segmentation and ask if the detection is consistent with this segmentation. Motivated by previous work on combining segmentation and detection, we investigated several mechanisms for measuring consistency with the background mask. This included examining consistency with average shape masks derived from example segmentations of each object and explicitly learning a mask template from example training data (see Experiments). We also tested using GrabCut [58] or super-pixels in order to refine the background mask estimate based on local image evidence such as discontinuities in color and texture. In the end we found that simply using the proportion of background mask pixels inside the bounding box with the same simple threshold used in generating scene specific negatives ($q(i) > 0.2$) was as effective at removing false-positives as any of these more elaborate schemes.

3.5 Experimental Results

In order to evaluate performance of our ideas we constructed a labeled dataset of pedestrians from a collection of images of Notre Dame provided online by the authors of Bundler [71]. Images were annotated with bounding boxes around each pedestrian using a protocol similar to the PASCAL VOC detection set. Bounding boxes were tight around the visible portion of the person. Difficult examples such as seated or heavily occluded people were included in the ground truth but tagged as "difficult" and not used in computing the test performance

	DT	DT+SS$^-$	DPM	DPM+SS$^-$
Detection	0.296	0.395	0.455	0.551
+MVBS	0.412	0.430	0.558	0.552
POP [37]	0.323	0.322	0.348	0.323
POP+SfM	0.405	0.406	0.404	0.337

	DT+FS$^-$	DT+FS	DPM+FS$^-$	DPM+FS
Detection	0.41	0.43	0.55	0.63

Table 3.1: Average Precision for pedestrian detection with scene-specific detectors. **DT** is the baseline Dalal-Triggs template detector [13] and **DPM** is the deformable parts model of [19]. **+SS$^-$** indicates the detector was trained with automatically acquired scene-specific negative instances. **+MVBS** prunes detections whose bounding box contain more than 20% estimated background pixels based on wide-baseline matching. **POP** shows the results obtained by using the approach in [37], while **+SfM** enforces the horizon estimation using structure from motion results. The unsupervised scene-specific model performs significantly better than the baseline, with multi-view background-subtraction providing additional gains in detector precision. For comparison we also show performance for fully supervised scene-specific training. **+FS** indicates results using scene specific positive and negative instances, **+FS$^-$** uses only negative instances. Our unsupervised approach achieves similar levels of performance and is scalable to large numbers of scenes.

numbers presented here (*i.e.*, they neither count as true or false positives). Low resolution people (< 40 pixels) were marked as difficult or not annotated. Benchmarking used the standard PASCAL detection benchmark criteria in which 50% overlap between a detection and ground-truth bounding box is sufficient (where overlap is the ratio of intersection area to area of union).

The 401 images available were randomly split into 200 test and 201 training. The training images were used when automatically generating negative examples to train the scene-specific detector as well as during algorithm development to validate the choice of bounding box mask threshold parameter and SVM regularization.

Baseline Detectors: We focus our experiments on two popular object detectors. First, we consider an implementation of the Dalal-Triggs (DT) rigid template model [13]. We train our implementation of the detector on positive and negative examples provided in the INRIA

Person dataset. The resulting baseline detector achieves an AP=0.79 on the INRIA test set, comparable to the results reported elsewhere [13]. Performance of this baseline detector on the Notre Dame test dataset (AP=0.296) is lower than on INRIA due to the greater variety of appearances of pedestrians labeled in the ND dataset (*e.g.*, more poses, occlusion and truncation, wider range of scale, etc.) We set the SVM regularization parameter C to maximize performance on the Notre Dame training images.

In addition to the DT model, we also test with the *deformable parts model* (DPM) of Felzenswalb *et al.* [19], using the implementation from [27] trained on the PASCAL VOC 2007 (train+val) dataset [18]. This model is substantially more complex but achieves better baseline performance (AP=0.455) by modeling deformation as well as mixture components which better detect truncated people marked in the dataset.

Training Scene-Specific Background Models: The 201 training images were used in building the scene specific background model (denoted **DT+SS$^-$** and **DPM+SS$^-$** in the figures). For this purpose we did *not* use the ground-truth annotations but did utilize the background mask with the $q_i > 0.2$ background threshold. We found that both models performed significantly better when trained with scene specific negatives. DT improved from 0.296 to 0.395 and DPM from 0.455 to 0.551 average precision (compare columns of Table 3.1). We did find it necessary to decrease the degree of regularization when changing the size of the training data set (C went from 0.1 to 0.01).

We also compared our scene-specific background model with unsupervised hard-negative mining to a supervised version (**FS-**) in which the scene-specific negatives were chosen to not overlap with any positive bounding boxes by more than 10%. This achieved an AP of 0.41 for DT and 0.55 for DPM suggesting that our unsupervised negative mining based on masks is capturing most of the useful negative examples. Finally, we evaluated fully supervised versions (**FS**) of the DT and DPM models which include both scene-specific positives and

negatives in addition to the INRIA and PASCAL training sets, respectively. This yielded an AP of 0.43 for DT and 0.63 for DPM respectively.

In training the scene specific model, it is useful to start with a pre-trained model and then perform additional passes of hard negative mining on the scene specific images. We found that this hot-starting was significantly more efficient than retraining the model from scratch. For example, training the DT detector from scratch took 83 minutes compared to only 37 minutes to hot start. Similarly, the DPM model takes days to train on the whole PASCAL dataset but only hours to hot start.

Multi-View Background Subtraction: We tested the multi-view background subtraction scheme (MVBS) using simple thresholding by rejecting detections with $q(i) > 0.2$. Results are shown in Figures 3.4, 3.5 and Table 3.1. Rejecting such false positives increased the average precision of both detectors. In the case of the DT detector, the combination of MVBS and SS^- training achieves even better performance while the DPM model saturates at 0.55 average precision.

In addition to the simple mask thresholding scheme, we also experimented with learning various features derived from the mask including the mean count of background pixels in the bounding box, the mean match score, and a spatial mask template with various spatial binnings. We found that none of these gave huge performance gains over the simplest thresholding. Learning a spatial mask template on the ND training set with spatial binning at the same resolution of the HOG descriptor gave $AP = 0.476$ while using pixel-sized bins yielded performance of $AP = 0.480$. However, the resulting templates had relatively little structure and are likely over-fit to the statistics of the background masks recovered for this particular scene rather than being universally applicable for all pedestrians.

Figure 3.6 shows qualitative example outputs of the baseline detector, scene-specific detector and the effect of multi-view background subtraction. There are many textured regions on

the cathedral facade where the baseline detector produces false positives. In particular, the carved human figures on the facade naturally match the template well. The model trained with additional scene-specific negatives is able to reject some of the false-positives as it finds very similar examples in the training set which are used as negative support vectors.

Geometric Context: A skeptical reviewer might be concerned that all we are doing is removing those detections up "in the sky", something that could be accomplished using SfM alone without constructing a dense background mask. To check this, we estimated the position of the horizon line based on the recovered camera pose for each test image. Since the plaza in front of the Cathedral is largely planar, we do not expect any pedestrians to appear floating above the horizon. This simple check of geometric consistency also achieves substantial performance improvements for the Dalal-Triggs detector, raising the average precision to $AP = 0.42$. However, multi-view background subtraction is able to prune additional detections which satisfy geometric constraints but include patches of facade visible in the training set providing small but distinct gains over purely geometric pruning (see +GC curves in Figure 3.4 and qualitative examples in Figure 3.6). This distinction would be even more obvious in a more complicated scene with elevated structures (balconies, stairs, playground equipment, trolley platforms, etc.)

Putting objects in perspective: We evaluated the Putting Objects in Perspective (POP) system of Hoiem *et al.* [37] which performs more sophisticated joint probabilistic inference over the camera pose, scene geometry, and detection hypotheses. We considered two different scenarios. In the first we simply substituted our baseline detector but used the camera pose and geometry priors graciously provided by the authors. In the second scenario, we replaced the default prior Gaussian distribution over horizon line position with a tightly peaked double-exponential (b=0.005) centered at the horizon estimate based on SfM camera pose estimation. We also tried using a prior derived from the camera heights produced during bundle adjustment but were unable to find a scaling that yielded better results than the

prior from the original paper.

For the default camera pose priors, the POP inference routine is able to boost the DT detector performance from 0.296 to 0.323. Substituting in the much stronger horizon estimate produced by SfM provides a much more significant boost, up to an average precision of 0.4. Surprisingly, these gains are not present when using the DPM detector. We believe this might be because the conversion of the detector score into a probability based on logistic fitting produces an overestimate of the detector confidence which skews the inference result.

3.6 Discussion

Much of the work on general-purpose object recognition has focused on detecting objects against arbitrary backgrounds of non-objects or other object categories. Such systems are typically trained with negative examples taken from random images off the web. The idea proposed here is in some ways counter to much contemporary research in category-level object recognition which has focused on generic detectors that will work in a wide range of environments. Indeed, it is a topic of hot debate whether current models are over-fitting to even the most general detection benchmarks [79]. Of course, such over-fitting is useful if you know which dataset (in this case scene) that you will be tested on.

Perhaps the most closely related work to ours is the "Putting objects in perspective" by Hoiem *et al.* [37] which makes joint inferences about scene geometry, camera pose and detection likelihoods. POP only attempts to encode generic prior knowledge about the scene geometry and camera pose in the form of a surface orientation classifier [39]. In contrast, we argue that for many scenes, it is not unreasonable to expect that other photos of the same scene are available from which to do more aggressive geometric reasoning. It thus seems worthwhile to revisit the idea of geometric context in the setting of large-scale SfM

which can provide much more reliable estimates of scene geometry for many parts of a novel test image as well as camera pose. From a research perspective, this would help isolate the benefits of geometric context for detection from the difficulties of single-image geometry estimation. Our experiment with pruning detections based on the horizon line from camera pose estimates touches on this but one could clearly go much further. For example, one could utilize the surface estimates returned from multi-view stereo or even re-project a 3D map which was annotated with "affordances" indicating what spatial volumes are likely to contain which objects and in which poses.

Finally, it seems valuable to think how recent work on robust reconstruction relates to problems of recognition. Efforts such as Google Street View that are assembling ever larger collections of images and other data into rich maps and models of city-scapes must constantly deal with dynamic objects (tourists, cars, pigeons, trash, etc.) which constitute outliers to be ignored during matching or better yet eliminated. However, from a broader perspective of scene understanding, *one model's outlier is another model's signal* and these annoyances should be transmuted into useful cue for recognizing dynamic objects.

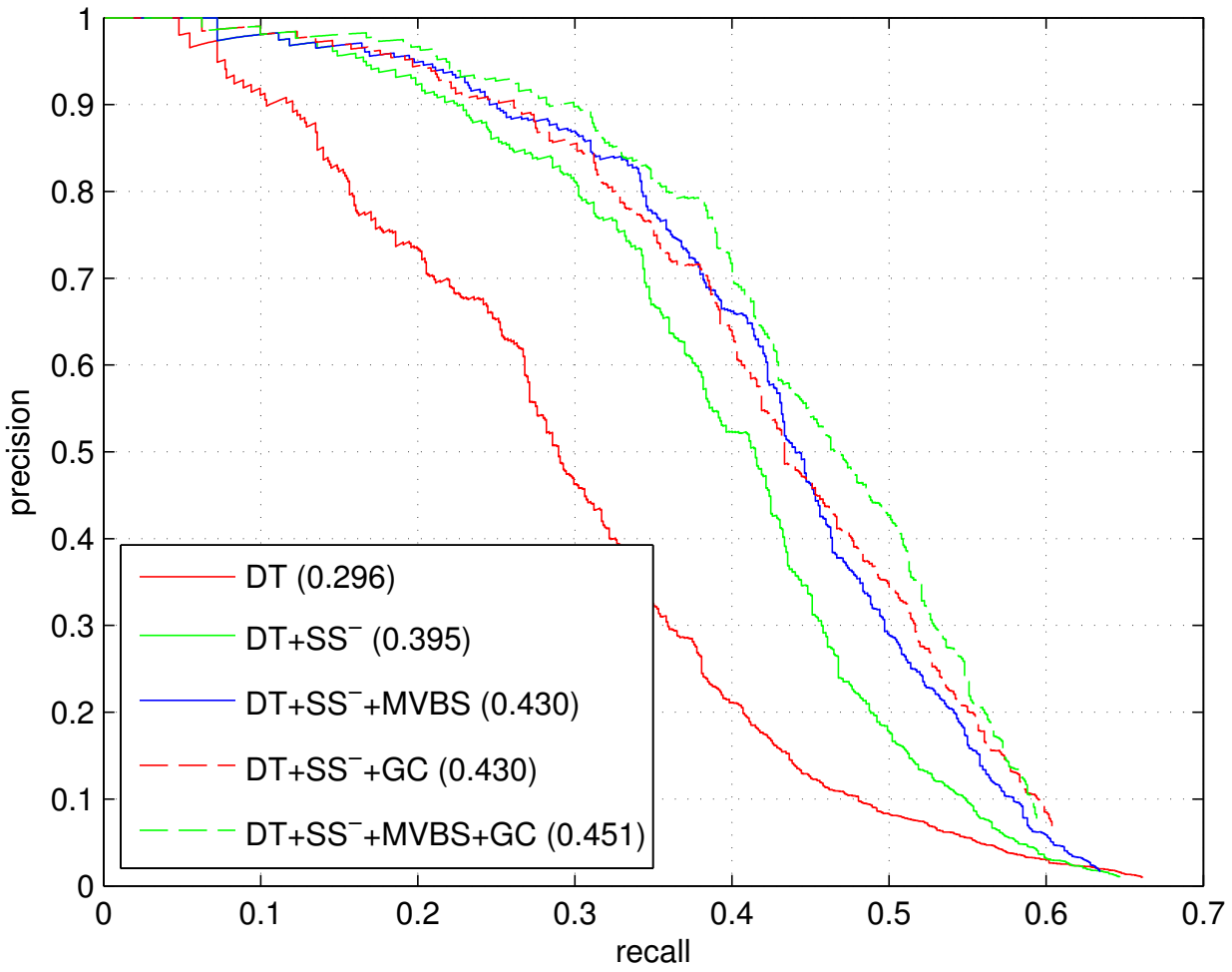


Figure 3.4: Precision-Recall for pedestrian detection with scene specific detectors. **DT** is the baseline Dalal-Triggs template detector trained on the INRIA dataset. **+SS⁻** is trained using scene-specific negative instances mined in an unsupervised manner from images of Notre Dame. **+GC** prunes detections where the bottom of the detection appears above the horizon based on the camera pose estimated using SfM. **+MVBS** prunes detections whose bounding box contain more than 20% estimated background pixels based on multi-view matching. The scene-specific model performs significantly better than the baseline with multi-view background-subtraction and geometric consistency both providing additional gains in detector precision.

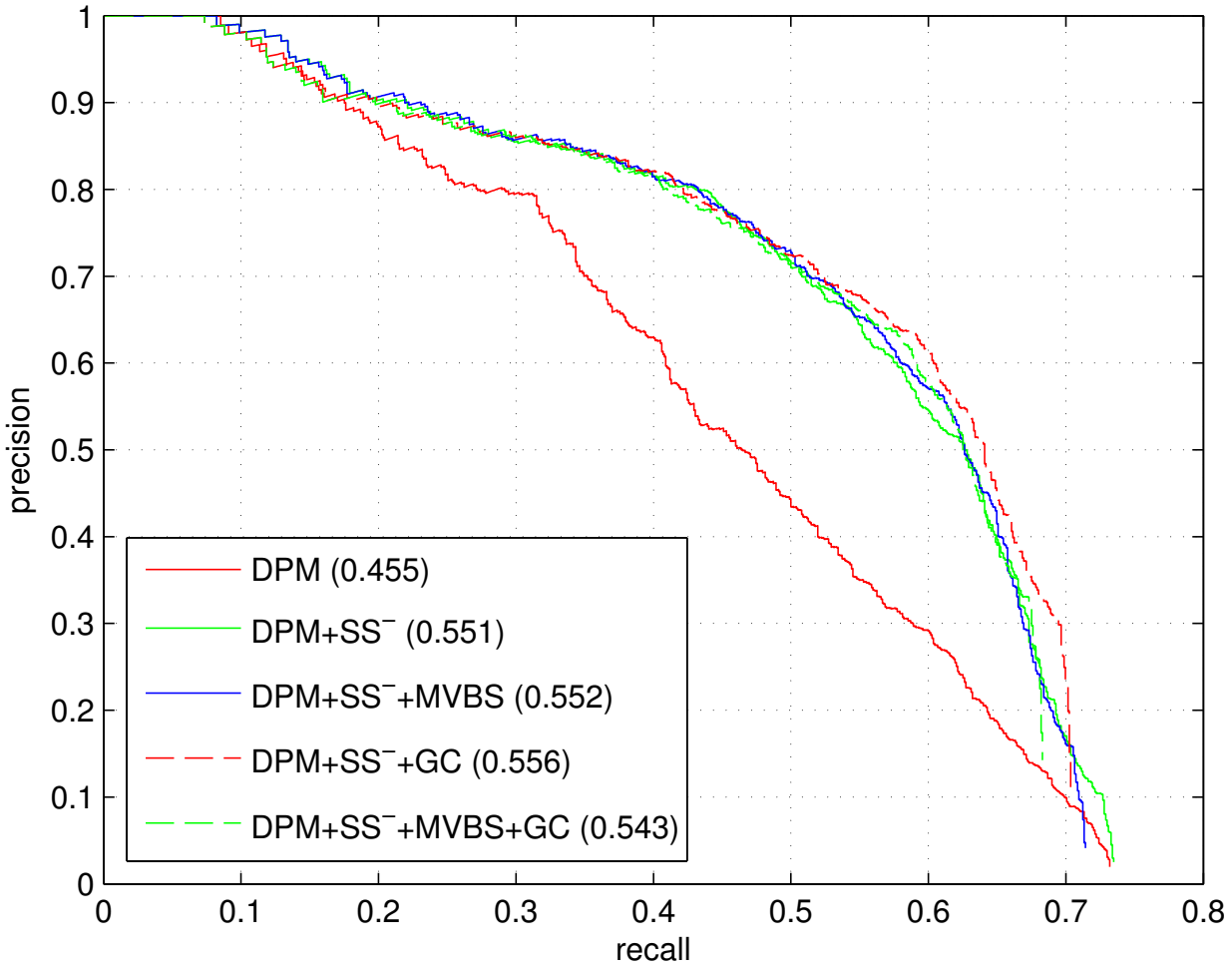


Figure 3.5: Precision-Recall for pedestrian detection with scene specific detectors. **DPM** is the baseline Deformable Parts Model template detector trained on the PASCAL dataset. **+SS$^-$** is trained using scene-specific negative instances mined in an unsupervised manner from images of Notre Dame. **+GC** prunes detections where the bottom of the detection appears above the horizon based on the camera pose estimated using SfM. **+MVBS** prunes detections whose bounding box contain more than 20% estimated background pixels based on multi-view matching. The scene-specific model performs significantly better than the baseline. Multi-view background subtraction and geometric consistency both provide additional gains in detector precision although the gains are less than in the case of the DT template detector.

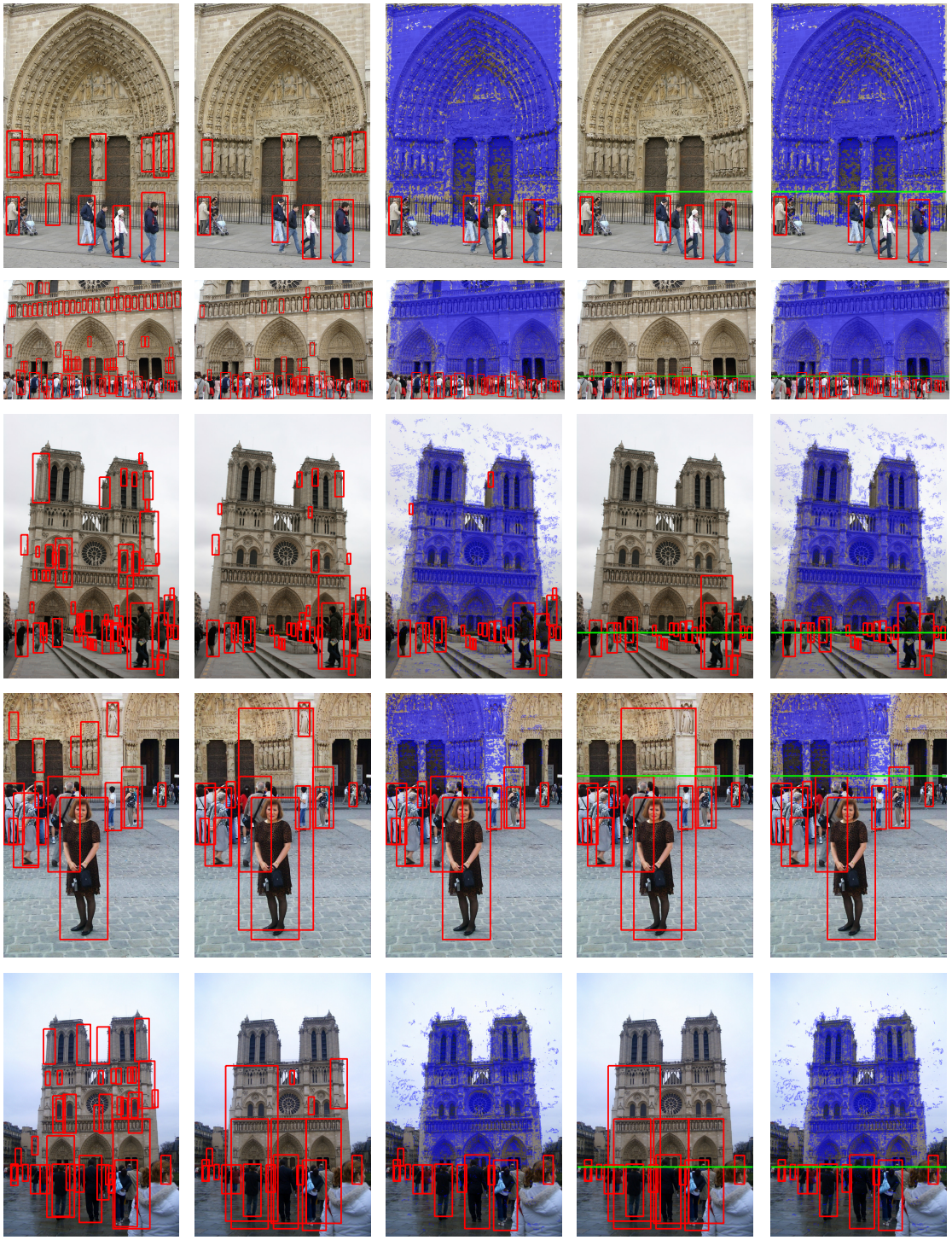


Figure 3.6: Example detector outputs at 50% recall. Unsupervised scene specific training makes the detector better able to reject common distractors (*e.g.*, the statues in row 2). MVBS can prune additional false positives at test-time by performing stereo matching to a database of existing images. Note that MVBS is able to remove some false positives which are not caught by geometric consistency (GC) with the horizon line because the hypothesized detections overlap heavily with regions identified as background.

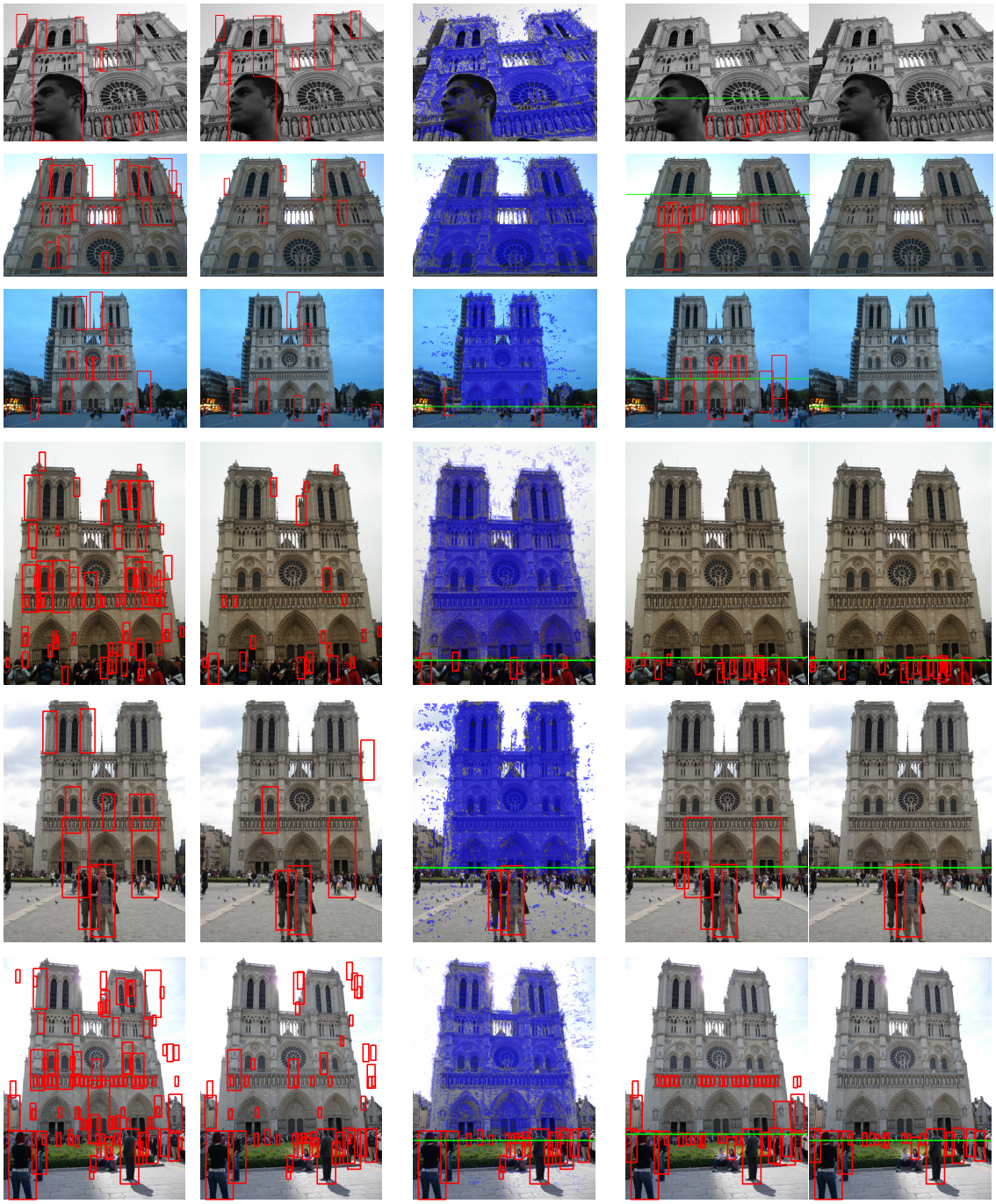


Figure 3.7: Example detector outputs at 50% recall. Scene specific training makes the detector better able to reject common distractors while MVBS can prune additional false positives. Putting Objects in Perspective (PoP) performance improves with accurate horizon estimation provided by structure from motion (SfM).

Chapter 4

Lifting GIS Maps into Strong Geometric Context for Scene Understanding

4.1 Introduction

Geographic Information Systems (GIS) are a set of resources and tools that study and analyze geographic information. This discipline of geoinformatics has been broadly used for multiple purposes like business, planning, transport/logistics, or telecommunications. However, Computer Vision has only lightly approached it. We believe that this type of resources is essential for the world of scene understanding. We argue that a huge number of photographs taken in outdoor urban areas are pictures of known scenes for which rich geometric scene data exists in the form of GIS maps and other geospatial data resources. Robust image matching techniques make it feasible to resection a novel image against large image datasets to produce estimates of camera pose on a world-wide scale [43]. Once a test photo has been

precisely localized, much of this contextual information can be easily backprojected into the image coordinates to provide much stronger priors for interpreting image contents. For the monocular scene-understanding purist, this may sound like "cheating", but from an applications perspective, such strong context is already widely available or actively being assembled and should prove hugely valuable for improving the accuracy of image understanding.

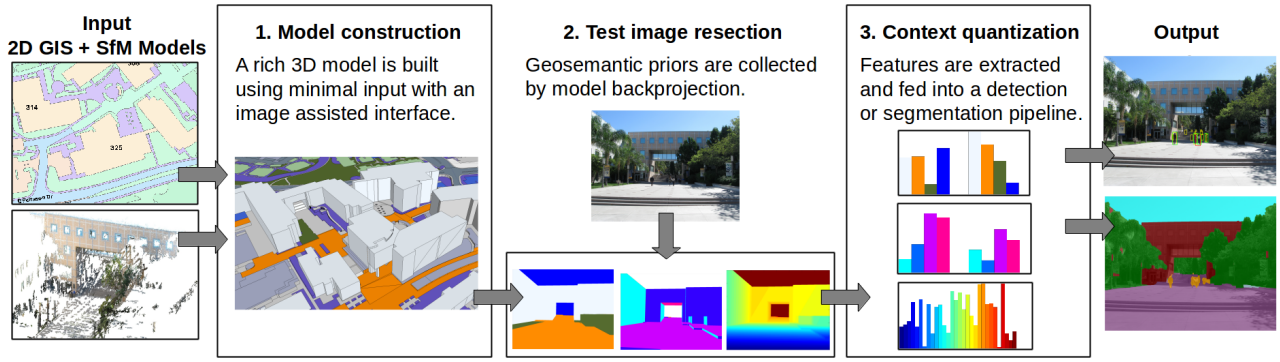


Figure 4.1: System overview: 2D GIS and SfM data are aligned to build a 3D model with minimal effort using an image-assisted Sketchup plug-in. This fused geocontext provides a basis for efficiently transferring rich geometric and semantic information to a novel test image where it is used to improve performance of general scene understanding (depth, detection, and segmentation).

The role of GIS map data in automatically interpreting images of outdoor scenes appears to have received relatively little attention in computer vision. [49] used a re-scoring framework in which the score of a candidate detection was jointly constrained by the geospatial position of the object in the scene. Perhaps most closely related work to our approach is [83] which uses a CRF to simultaneously estimate depth and scene labels using a strong 3D model. However, [83] assumes that both camera localization and a CAD model of a scene with relevant categories (*e.g.*, trees) are provided as inputs. In contrast, we address the construction of a 3D scene model by combining image and map data as well as test-time camera localization using model alignment and resectioning techniques. Interestingly, we also show that lifted GIS data can improve segmentation accuracy even for semantic labels that are not present in the GIS map model (*e.g.*, trees, retaining walls).

Rescoring object detector outputs based on contextual and geometric scene constraints has

been suggested in a wide variety of settings. For example, Hoiem *et al.* [37] used monocular estimation of a ground-plane to rescore car and pedestrian detections, improving the performance of a contemporary baseline detector [13]. Similarly, [3] showed improvement on hard to detect objects such as fire hydrants. However, the benefit of scene geometry constraints appears to be substantially less when a more robust baseline detector such as DPM [19] is used. For example, [15] reported that monocular ground-plane constraints failed to improve the performance of a DPM pedestrian detector. Similarly, [49] reported that geometric rescoring based on road maps did not improve performance of a DPM car detector (although rescoring did improve detection for a weaker detector that had been trained to predict viewpoints). In contrast to this previous work, we use a richer rescoring model that allows for non-flat supporting surfaces and integrates additional geosemantic cues, resulting in a significant boost (5% AP) in pedestrian detection even when rescoring a strong baseline model.

To study the role of strong geometric context for image understanding, we have collected a new dataset consisting of over six thousand images covering a portion of the UC Irvine campus for which relative camera pose and 3D coordinates of matching points has been recovered using structure from motion. We describe a method for aligning this geometric and photometric data with 2D GIS maps order to quickly build 3D polygonal models of buildings, sidewalks, streets and other static structures with minimal user input (Section 4.2). This combined geosemantic context dataset serves as a reference against which novel test images can be resectioned, providing a rich variety of geometric and semantic cues for further image analysis. We develop a set of features for use in detection and segmentation (Sections 4.3 and 4.4) and experimentally demonstrate that these contextual cues offer significantly improved performance with respect to strong baselines for scene depth estimation, pedestrian detection and semantic segmentation (Section 4.5).

The contribution of our work is twofold. First, we demonstrate a pipeline that performs

precise resectioning of test images against GIS map data to generate geosemantic context applicable to multiple scene understanding tasks. Secondly, we show that simple cues derived from the GIS model can provide significantly improved performance over baselines for depth estimation, pedestrian detection and semantic segmentation. We briefly discuss the differences between our results and closely related work. Figure 4.1 shows an overview of the system pipeline.

4.2 Lifting GIS Maps

In this section, we describe the construction of a dataset for exploring the use of strong GIS-derived geometric context. We focus on novel aspects of this pipeline which include: aligning GIS and SfM data, a user-friendly toolbox to lift 2D geosemantic information provided by a map into 3D geometric context models, and model-assisted camera resectioning to allow quick and accurate localization of a test image with respect to the context model.

Image database acquisition We collected a database of 6402 images covering a large area (the engineering quad) of UC Irvine's campus. Images were collected in a systematic manner using a pair of point and shoot cameras attached to a monopod. We chose locations so as to provide approximately uniform coverage of the area of interest. Images were generally collected during break periods when there were relatively few people present (although some images still contain pedestrians and other non-rigid objects). The dataset is available at http://vision.ics.uci.edu/datasets/.

Running off-the-shelf incremental structure from motion (*e.g.*, [1, 72]) on the entire dataset produces a 3D structure that is qualitatively satisfying but often contains metric inaccuracies. In particular, there can be a significant drift over the whole extent of the recovered model that makes it impossible to globally align the model with GIS map data. These reconstruction

results were highly dependent on the quality of the initial matches between individual camera pairs. However, we found that the excellent incremental SfM pipeline implementation in OpenMVG [54], which uses the adaptive thresholding method of [52], yielded superior results in terms of the accuracy and number of recovered cameras and points. Our final model included 4929 succesfully bundled images.

Global GIS-structure alignment We obtained a 2D GIS map of the campus maintained by the university's building management office. The map was originally constructed from aerial imagery and indicates polygonal regions corresponding to essential campus infrastructure tagged with semantic labels including building footprints, roadways, firelanes, lawns, etc.

We would like to align our SfM reconstruction with this model. One approach is to leverage existing sets of geo-referenced images. For example, [49] used aerial LiDAR and Google Street View images with known geo-coordinates in order to provide an absolute coordinate reference. While this is practical when the 3D GIS data has already been generated, we start from a flat 2D model depicting an environment for which no precisely georegistered images were readily available.

To quickly produce an initial rough alignment, we project the point cloud recovered using SfM onto a ground-plane and have a user select 3 or more points (typically building corners as they are more visible). We run Procrustes alignment to match the user selected points to the real coordinates in our 2D GIS data. This global 2D alignment is sufficient to register the SfM model in geographic coordinates for the initial construction of a 3D model. Once the 3D GIS model is defined (see below), we used an iterative closest point approach to automatically refine the alignment of the GIS model and SfM point cloud.

Image-assisted 3D model construction We developed a custom plug-in for the 3D modeling tool Sketchup to allow efficient user-assisted "lifting" of 2D GIS map data into a full 3D model. We imported the GIS 2D polygons with their corresponding semantic labels into the workspace. The user is then presented with a choice of images to load from the globally aligned SfM model. When an image is selected, the 3D model view is adjusted to match the recovered camera extrinsic and intrinsic parameters and the image is transparently overlaid, providing an intuitive visualization as shown in Figure 4.2. The user can then extrude flat 2D polygons (*e.g.*, building footprints) to the appropriate height so that the backprojected view into the selected camera matches well with the overlaid image.

A full 3D mesh model can be easily constructed starting from the aligned 2D map by extruding buildings up, carving stairs down, tilting non-fully horizontal surfaces, etc. using standard Sketchup modeling tools and guided by the image overlay. Additional geometry can be easily created, as well as adding additional semantic labels or corrections to the original data. With the assistance of these aligned camera views, constructing a fairly detailed model in Sketchup covering 10 buildings took approximately 1-2 hours. This can be considered an offline task since the modeling effort is performed only once, as buildings are largely static over time.

GIS-assisted test image resectioning To estimate camera pose at test time, we resection each test image against the SfM image dataset based on 2D-3D matching correspondences using a RANSAC-based 3-point absolute pose procedure [41]. We developed several techniques for improving the accuracy of this resectioning. As the bundled image dataset grows to cover larger areas, resectioning accuracy generally falls since the best match for a given feature descriptor in the test image is increasingly likely to be a false positive. To leverage knowledge of spatial locality and scale to large datasets, we partitioned the bundled cameras into $k = 10$ clusters using k-means over database camera positions. Points in the

Figure 4.2: We developed a custom Sketchup plug-in that imports camera parameters computed during bundle adjustment. The user easily models the 3D geometry by extruding, tilting, and carving the 2D map data until it naturally aligns with the image.

bundled model were included in any camera cluster in which they were visible yielding 10 non-disjoint clusters of points.

Each test image was resectioned independently against each spatial cluster and the best camera pose estimate among the clusters was selected using geometric heuristics based on the GIS model. We measured the distance to ground (height) and camera orientation with respect to the gravity direction in the GIS model. All camera poses below the ground plane, higher than 4 meters, or tilted more than 30 degrees were discarded. If a camera pose was geometrically reasonable in more than one cluster, we selected the estimate with the highest number of matched inliers. *Incorporating these heuristic constraints increased the proportion of correctly resection test images substantially (from 49% to 59%).* This improvement would not possible without the GIS-derived 3D model since the ground elevation varies significantly and cannot be captured by a simple global threshold on the camera coordinates.

Resectioning of a test image produces immediate predictions of scene depth, surface orientation, and semantic labels at every image pixel. It is interesting to compare these to previous works that attempt to estimate such geometric or semantic information from a single image. By simply resectioning the picture against our 3D model we are immediately able to

make surprisingly accurate predictions without running a classifier on a single image patch! In the remainder of the chapter, we discuss how to upgrade these "blind" predictions by incorporating backprojected model information into standard detection and segmentation frameworks.

4.3 Strong Geometric Context for Detection

Estimating camera pose of a test image with respect to a 3D GIS model aids in reasoning about the geometric validity of hypothesized object detections (*e.g.*, pruning false positives in unlikely locations or boosting low-scoring candidates in likely regions). We describe a collection of features that capture these constraints and use them to improve detection performance. We incorporate these features into a pedestrian detector by training an SVM with geometric context features (GC-SVM) that learns to better discriminate object hypotheses based on the geosemantic context of a candidate detection.

3D geometric context Let a candidate 2D bounding box in a test image I have an associated height in pixels h^{im}. We use a deformable part model that consists of a mixture over three different template filters: full-body, half upper body, and head. We set the image detection height h^{im} based on which mixture fires as 1, 2 or 3 times the bounding box height respectively.

If we assume that the object is resting on a horizontal surface and the base of the object is visible, then we can estimate the 3D location the object is occupying. We find the depth at intersection z_i of the object with respect to the camera by shooting a ray from the camera center through the bounding box's "feet" and intersecting it with the 3D model. Importantly, unlike many previous works, the ground is not necessarily a plane (*e.g.*, our model includes ground at different elevations as well as stairs and ramps). Given camera focal length f, we

can estimate the height in world coordinates by the following expression:

$$h_i = \frac{z_i}{f} h^{im} \tag{4.1}$$

Unfortunately, the object's "feet" might not be visible at all times (*e.g.*, a low wall is blocking them). Let h_μ be the "physical height" of an average bounding box (*i.e.*, the height of an average human). We collect all possible intersections with the model (*e.g.*, both the intersections with a blocking wall and the ground plane behind it) and choose the h_i that minimizes $(h_i - h_\mu)^2$.

An alternative method to hypothesize an object's height is by its inverse relation with depth. We can estimate an object's depth z_o based on the expected average human height h_μ. A bounding box of size h^{im} in the image has an expected distance

$$z_o = h_\mu \frac{f}{h^{im}} \tag{4.2}$$

from the camera. Given z_o, we produce a second height estimate h_o by tracing a ray through the center top of the detection to a depth z_o and then measuring the distance from this estimated head position to the ground plane.

For each height estimator h_i, h_o we also extract a corresponding semantic label associated with the GIS-model polygon where the feet intersect and record binary variables w_i, w_o indicating whether the polygon has a "walkable" semantic label and n_i, n_o indicating whether it has a horizontal surface normal. Our feature vectors for each estimate are given by:

$$F_i = [v_i(h_i - h_\mu)^2, w_i, n_i, (1 - v_i)] \tag{4.3}$$

$$F_o = [v_o(h_o - h_\mu)^2, w_o, n_o, (1 - v_o)] \tag{4.4}$$

where $h_\mu = 1.7m$ is the average human height and v_i, v_o are binary variables indicating

whether the corresponding height could be measured. For example, if the ray to the foot to compute z_i does not intersect the model or if the depth estimate z_o is behind the model surface, we mark the estimate as invalid and zero out all but the last entry of the feature vector.

2D geosemantic context In addition to the height and foot location, we extract geosemantic context by backprojecting model semantic labels and surface normals into the image plane and look at the distribution inside of the object bounding box. For each bounding box b, we hypothesize a full-body bounding box given the detection's mixture as previously mentioned. We then split such box vertically into 3 parts (top, center, bottom) and collect normalized histograms H_b of the distribution of semantic labels and surface normals in each subregion. We account for 5 GIS labels (building, plants, pavement, sky, unknown) and 4 discretized surface normal directions (ground, ceiling, wall, none).

$$H_b = [H_{top}, H_{center}, H_{bottom}] \tag{4.5}$$

To allow the learned SVM weights to depend on the original detector mixture component m, we construct an expanded feature vector from these histograms:

$$F_b = [\delta(m = 1)H_b, \delta(m = 2)H_b, \delta(m = 3)H_b] \tag{4.6}$$

where $m \in 1, 2, 3$ indicates a full-body, upper-half, or head mixture respectively. For example, this allows us to model that upper-body detections are correlated with the presence of some vertical, non-walkable surface such as a wall occluding the lower body.

We train an SVM to rescore each detection using a final feature vector that includes the

detector score s along with the concatenated context features

$$F = [s, F_i, F_o, F_b]$$

4.4 Strong Geometric Context for Segmentation

The geometric and semantic information contained in the GIS data and lifted into the 3D GIS model can aid in reasoning about the geometric validity of class hypotheses (*e.g.*, horizontal surfaces visible from the ground are not typically buildings). We describe methods for using such constraints to improve semantic segmentation performance. We follow a standard CRF-based labeling approach from Gould *et al.* [29] which uses an augmented set of image features from [69]. We explore simple ways to enhance this set of features using GIS data and study its influence on semantic labeling accuracy.

GIS label prior distributions The GIS-derived model provides an immediate estimation of pixel labels based on the 4 semantic labels in the original GIS map (building, plants, pavement, and sky). If a camera pose is known, we can backproject the model into the image plane and transfer the polygon label in the GIS model to the projected pixel. However, camera pose estimation is not perfect and might contain minimal deviations from its ground truth pose. In order to account for slight camera pose inaccuracies, we define a 16-dimensional feature descriptor to softly handle these cases. Given an image I, a pixel $x \in I$, and a backprojected GIS semantic label $g(x)$, we define the feature $h_k^r(x)$

$$h_{r,k}^s(x) = \frac{1}{N} \sum_{y:\|y-x\|<r} \delta(g(y) = k) \tag{4.7}$$

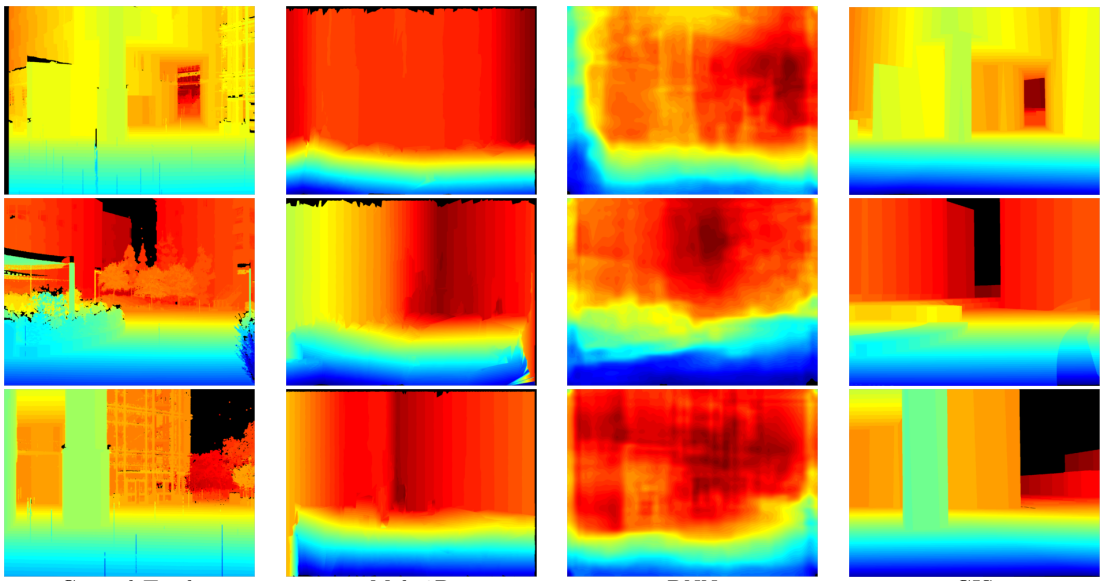


Figure 4.3: Qualitative depth comparison. Our GIS backprojected depth map is shown in the last column. While it lacks many details such as foliage and pedestrians which are not included in our coarse GIS-based 3D model, simply backprojecting depth provides substantially more accurate estimation than existing monocular approaches.

as the normalized count of class k pixels in a circular disc of radius r around x, where N is the number of pixels in the disc. In our experiments, we define r so that the angular error of the camera pose is 0, 1, 3, and 5 degrees.

GIS surface normal distributions In a similar manner, a surface normal can be quickly estimated for any pixel by backprojecting the 3D model into the camera plane. Surface normals can be discriminative of certain classes like pavement, roads, buildings, etc. Following the same structure as in equation 4.7, we define the 12-dimensional feature $h_n^r(x)$ as

$$h_{r,k}^n(x) = \frac{1}{N} \sum_{y:\|y-x\|<r} \delta(n(y) \in N_k) \tag{4.8}$$

where N_k is one of 3 possible surface orientation bins: horizontal (ground), horizontal (ceiling), vertical (wall).

GIS Depth features Depth can also be efficiently estimated from a 3D model when a camera pose is known. Following other methods like [86], we extract HOG features [13] to encode depth variations. We did find a substantial gain in certain categories when adding these features into the model (*e.g.*, wall).

DPM as a context feature Inspired by other works that try to create segmentation-aware detectors [20, 42, 48, 57], we also incorporate the outputs of category-specific object detectors in our segmentation model. To do so, we collect the scores of a DPM detector for an object category c and generate a DPM feature map h_c by assigning to every pixel the maximum score of any of the candidate detection boxes intersecting the given pixel. Let Ω_c be the set of candidate detections and b_i, s_i the bounding box and score for the ith detection, then

$$h_c^o(x) = \max_{i \in \Omega_c}(s_i \cdot \delta(x \in b_i)) \tag{4.9}$$

4.5 Experimental results

In our experiments, we started with a test set comprising 570 pictures taken in the area covered by the 3D model. These images were collected over different days and several months after the initial model dataset was acquired. Of these images, 334 images (59%) were successfully resectioned using the cluster-based approach from Section 4.2. This success rate compares favorably with typical success rates for incremental bundle adjustment (*e.g.*, [49] register 37% of their input images) even though our criteria for correctness is more stringent (we manually scored test images rather than relying on number of matching feature points). To evaluate performance of detection, we annotated resectioned images with ground-truth bounding boxes for 1484 pedestrians using the standard PASCAL VOC labeling practice

Threshold	Make3D	DNN	GIS
$\delta < 1.25$	11.03%	28.21%	67.04%
$\delta < 1.25^2$	30.71%	45.09%	82.37%
$\delta < 1.25^3$	49.47%	56.04%	88.25%

Table 4.1: Quantitative comparison of depth estimation methods: Make3D [64], Deep Neural Network [17], and GIS backprojection. We list the proportion of depths within a specified maximum allowed relative error $\delta = \max(\frac{d_{gt}}{d_{est}}, \frac{d_{est}}{d_{gt}})$, where d_{gt} is the ground truth depth and d_{est} is the estimated depth at some point. The DNN model predictions were re-scaled to match our ground truth data since the model provided is adapted only for indoor scenes and it was not practical to collect enough laser scans to retrain the model for our dataset.

including tags for truncated and partially occluded cases. We used 167 images for training our geometric context rescoring framework and left the remaining 167 for testing. We also manually segmented 305 of these images using the segmentation tool provided by [47] and labeled each segment with one of 9 different semantic categories. We split the segmentation data into 150 images for training and 155 for testing.

4.5.1 Monocular Depth Estimation

To verify that the coarse-scale 3D GIS model provides useful geometric information, despite the lack of many detailed scene elements such as trees, we evaluated resectioning and back-projection as an approach for monocular depth estimation. While our approach is not truly monocular since it relies on a database of images to resection the camera, the test time data provided is a single image and constitutes a realistic scenario for monocular depth estimation in well photographed environments.

To establish a gold-standard estimate of scene depth, we scanned 14 locations of the area covered by our dataset using a Trimble GX3D terrestrial laser scanner. We took the scans at in a range of resolution between 5 and 12 cm in order to keep the total scanning time manageable, resulting in roughly a half a million 3D points per scan. We mounted a camera on top of the laser scanner and used the camera focal length to project the laser-based 3D point cloud onto the camera image plane, interpolating depth values to obtain a per-pixel

depth estimate. We then resectioned the test image and synthesized the depth map predicted by our 3D GIS model.

Table 4.1 shows quantitative results of our GIS backprojection depth estimation against other single-image depth approaches for the 14 scan dataset. We used the provided pre-trained models included in [64, 17] as baselines for comparison. Since the pre-trained DNN model [17] was only available for indoor scenes (trained from Kinect sensor data), we estimated a scaling and offset factors for the output that minimized the relative error over the 14 images. While the scaled pre-trained DNN model is probably suboptimal, it is not possible to retrain the model without collecting substantially larger amounts of training data using specialized hardware (a laser scanner). In contrast, our proposed approach of simply backprojecting GIS data greatly outperforms image-predictions using only camera pose and a map, with no training data required!

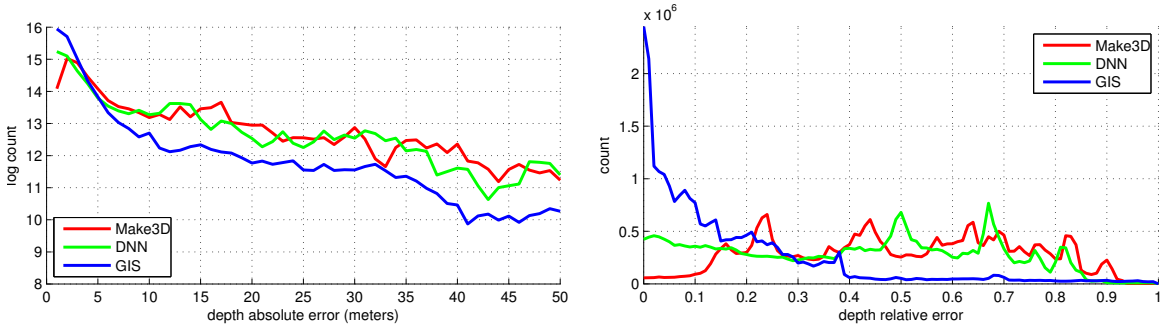


Figure 4.4: Accuracy of predicted depth estimates compared to gold-standard provided by a laser-scanner for 14 images. Top histogram shows the distribution (log counts) of absolute depth errors between Make3D, DNN and depth computed from resectioning and backprojecting GIS derived 3D model. The bottom plot shows the distribution of relative error $|d_{est} - d_{gt}|/d_{gt}$.

4.5.2 Object Detection

We evaluated our geometric and semantic object rescoring scheme applied to the widely used deformable part model detector (DPM) [19]. We used a standard non-maxima suppression

ratio of 0.5 in the intersection over union overlap.

Of the 167 training image split, we ran DPM and collected the set of geometric context features F described in Section 4.3 along with the appropriate label indicating whether the bounding box was a true or false positive. We trained a linear SVM classifier where we set the regularization $C = 4$ using 5-fold cross-validation. To accommodate class imbalance and maximize average precision, we used cross-validation to set the relative penalty for misclassified positives to be $4x$ larger than for negatives.

On test data the standard DPM detector score provided a baseline average precision (AP) of **0.457**, while our GC-SVM model based on geosemantic features F achieved an AP of **0.507**. It is important to note that previous attempts to incorporate GIS-based geometric rescoring into a DPM classifier provided little to no improvement. In [49], 3D context between cars and streets allowed for improved geometric reasoning about car orientation but only small gains in detection performance. In fact their final VP-LSVM and VP-WL-SSVM models had lower average precision than a baseline DPM model.

Model	Overall	building	plants	pavement	sky	ped.	ped. sit	bicycle	bench	wall
SUN CRF	0.309	0.767	0.581	0.863	0.872	0.007	0.000	0.000	0.000	0.000
+GIS-Only	0.287	0.709	0.404	0.852	**0.899**	0.007	0.000	0.001	0.000	0.000
+DPM	**0.323**	**0.774**	**0.586**	**0.864**	0.873	**0.129**	0.000	**0.001**	0.000	0.000
GIS Backproj.	0.242	0.688	0.099	0.810	0.581	0.000	0.000	0.000	0.000	0.000
GIS CRF	**0.290**	**0.730**	**0.316**	**0.847**	**0.705**	0.000	0.000	0.000	0.000	**0.014**
ENGQ CRF	0.561	0.917	0.886	0.925	0.949	0.400	0.010	0.370	0.241	0.348
+GIS	0.584	0.937	0.894	0.936	0.963	0.394	0.060	0.385	0.208	**0.481**
Depth	0.569	0.935	**0.895**	0.937	0.957	0.390	0.011	0.358	0.179	0.455
Labels	0.575	**0.938**	0.892	0.933	**0.966**	0.374	0.064	0.366	0.221	0.419
Normals	0.568	0.935	0.893	0.933	0.961	0.389	0.007	0.385	0.192	0.418
+DPM	0.590	0.920	0.892	0.929	0.947	**0.583**	0.013	0.482	0.184	0.360
+DPM+GIS	**0.627**	0.936	0.894	**0.938**	0.961	0.568	**0.108**	**0.520**	**0.245**	0.472

Table 4.2: Quantitative segmentation results for models trained with generic (SUN) and scene specific (ENGQ) data. Accuracy is measured using PASCAL intersection-over-union (IOU) protocol. Adding geosemantic (+GIS) and detection (+DPM) features outperformed the baseline models. Combining both methods gave the best overall results in the scene specific model.

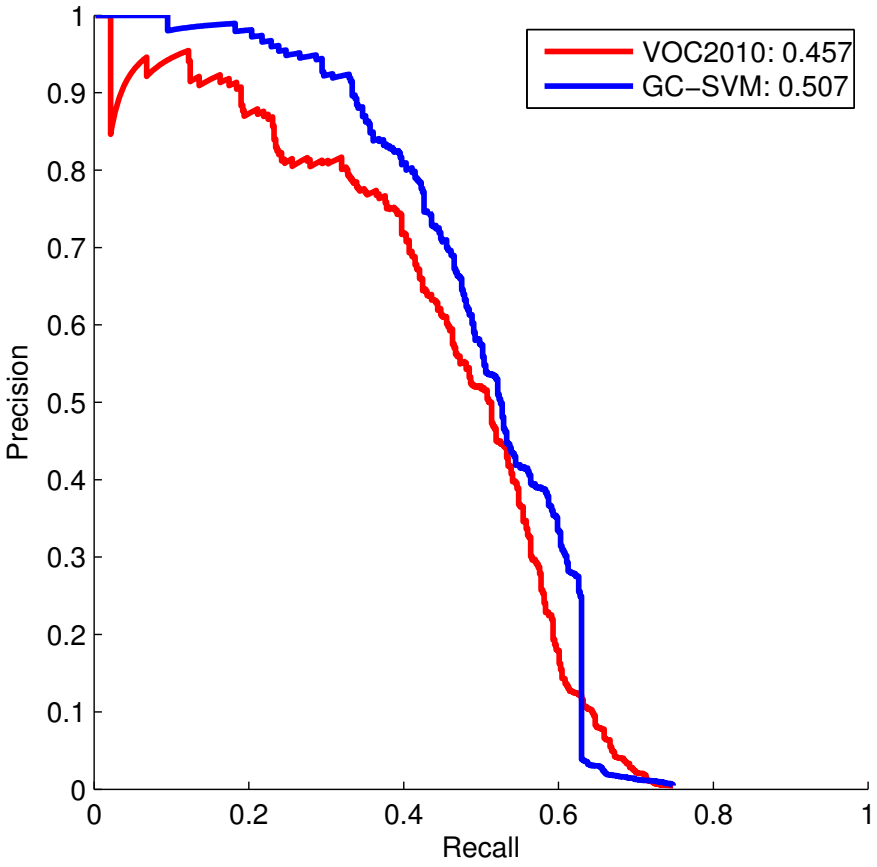


Figure 4.5: Geometric context aids in recognizing discriminative 3D and 2D features that improve the average precision in pedestrian detection. Our GC-SVM obtained a 5% boost in AP with respect to the standard DPM model.

4.5.3 Semantic Segmentation

We built two baseline segmentation models using the CRF-based multi-class segmentation code provided by [28]. We trained one model using our labeled training set of engineering quad images (CRF ENGQ) as a scene-specific model. We also trained a generic model using images collected from the SUN dataset [85] by querying for multiple categories in the SUN label set that are semantically equivalent to the 9 categories labeled in our engineering quad dataset (CRF SUN). Finally, we added our GIS and DPM features to the ENGQ model and evaluated their effects on segmentation performance.

Detectors Improve Segmentation We collected DPM score features as described previously for two objects of interest: pedestrian and bicycle. Figure 4.2 shows the influence of adding these features into the model (+DPM rows). We raised pedestrian segmentation accuracy from 0.400 to 0.583 and bicycle from 0.370 to 0.482 in the ENGQ model.

It is interesting to note how these detection priors mix with geosemantic information. In the presence of geometric context, pedestrian segmentation was slightly hurt. However, sitting pedestrian segmentation is boosted from almost 0 accuracy up to 0.108. Bicycle also benefits from geometric context and boosts from 0.458 to 0.530.

GIS-aware Segmentation To evaluate the influence of GIS features alone, we first trained a CRF model without the image features from [29] and only used the contextual features described in Section 4.4 (GIS CRF). This yielded relatively good accuracy in the 4 labels present in the GIS model, but poor results for many others. This is quite natural since our GIS model does not include detailed elements such as benches, and provides no information about what pixels might be a bike or pedestrian on any given day. However, this model still improves a simple "blind" backprojection of the GIS labels.

On the other hand, combining these GIS features with standard image features gave a significant benefit, outperforming the image CRF baseline in almost all categories (+GIS rows in Figure 4.2). It is interesting to note that labeling of some categories that did not appear in the GIS map data (*e.g.*, bench and wall) is still improved significantly by the geometric context provided in the model (wall is boosted from 0.348 to 0.481, presumably since the local appearance is similar to building but the geometric context is not). This is in contrast to, *e.g.*, [83], where all labels were included in either the GIS or detector driven priors.

Scene Specific vs Generic Models Even without precisely resectioning the test image, there is a significant gain in accuracy from knowing the rough camera location. When the camera location is completely unknown, the best we can do is invoke the CRF trained on generic SUN data (0.309 accuracy). However, if we know the camera is located somewhere on the engineering quad, we can invoke the scene specific CRF trained on ENGQ to boost performance to 0.561. With resectioning, we can further utilize the 3D model (+GIS) to gain an additional 3% in segmentation performance by utilizing geosemantic context features in the unary potential classifier. This gain is quite significant given that some class baselines are already over 90% in accuracy leaving little room for improvement.

4.6 Discussion

The rapid growth of digital mapping data in the form of GIS databases offers a rich source of contextual information that should be exploited in practical computer vision systems. We have described a basic pipeline that allows for integration of such data to guide both traditional geometric reconstruction as well as semantic segmentation and recognition. With a small amount of user supervision, we can quickly lift 2D GIS maps into 3D models that immediately provide strong scene geometry estimates (typically less than 5-10% relative depth error), greatly outperforming existing approaches monocular depth estimation and providing a cheap alternative to laser range scanners. This also provides strong geometric and semantic context features that can be exploited to improve detection and segmentation.

DPM GC-SVM

Figure 4.6: Detection results at 0.6 recall. Geosemantic context successfully removes high score false positives at unlikely places without adding too many low score ones at coherent regions. This successful trade-off benefits the performance at almost all levels of recall.

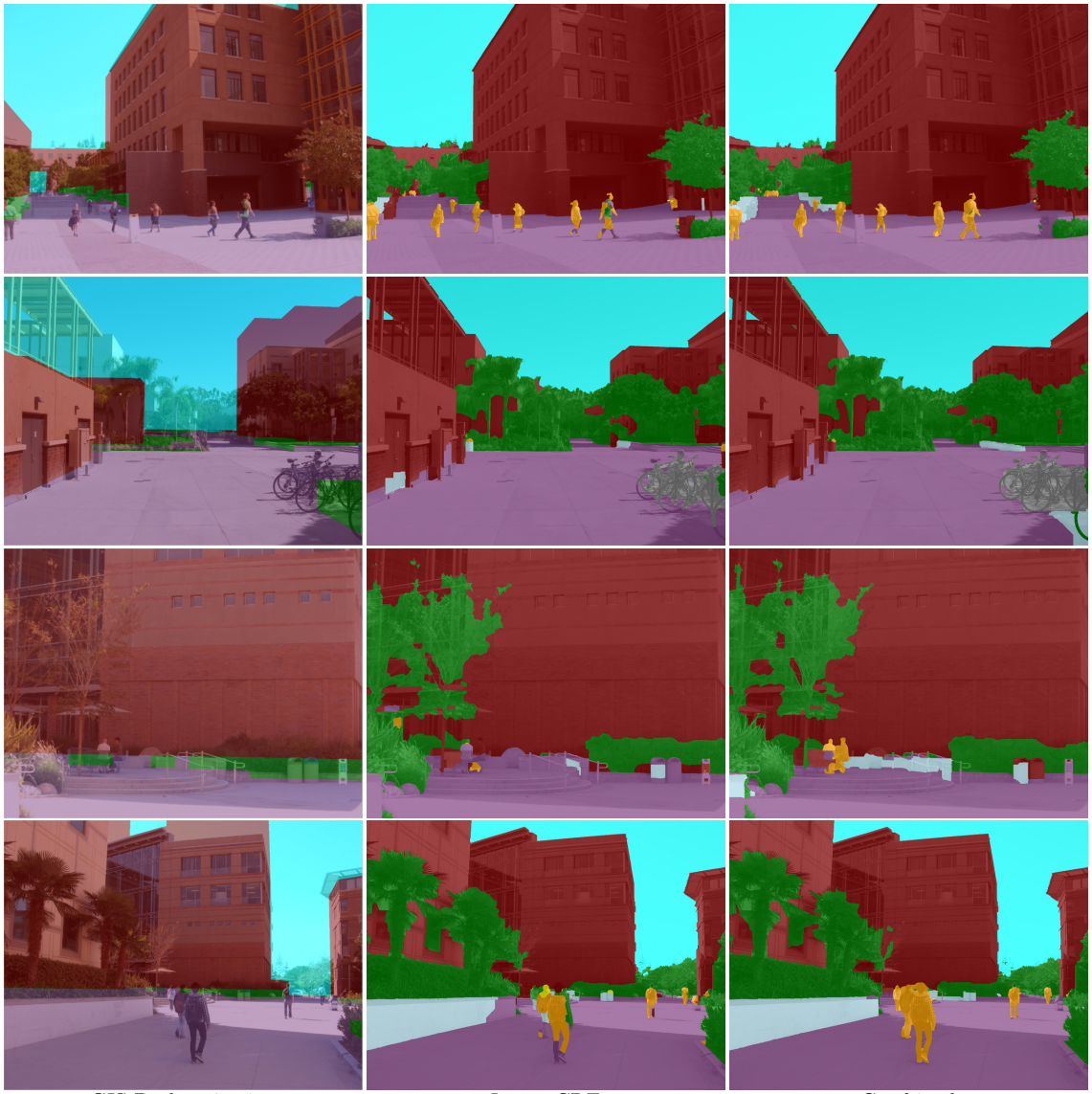


Figure 4.7: Qualitative segmentation results with overlaid images. Our combined model improves over a image-based CRF by incorporating features derived from GIS (depth, labels, normals) and a DPM detector.

Bibliography

[1] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. Building Rome in a day. *ICCV*, pages 72–79, Sept. 2009.

[2] S. Ardeshir, K. M. Collins-Sibley, and M. Shah. Geo-semantic segmentation. In *CVPR*, 2015.

[3] S. Ardeshir, A. R. Zamir, A. Torroella, and M. Shah. Gis-assisted object detection and geospatial localization. In *ECCV*. 2014.

[4] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *CVPR*, 2012.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[6] P. Beardsley, P. Torr, and A. Zisserman. 3d model acquisition from extended image sequences. In *European conference on computer vision*, pages 683–695. Springer, 1996.

[7] I. Biederman. *On the semantics of a glance at a scene*. 1981.

[8] M. Bujnak, Z. Kukelova, and T. Pajdla. A general solution to the p4p problem for camera with unknown focal length. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[9] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In *European conference on computer vision*, pages 778–792. Springer, 2010.

[10] S. Cao and N. Snavely. Graph-based discriminative learning for location recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 700–707, 2013.

[11] N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool. 3d urban scene modeling integrating recognition and reconstruction. *IJCV*, 2008.

[12] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3001–3008. IEEE, 2011.

[13] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages I: 886–893, 2005.

[14] T. Dekel, S. Oron, M. Rubinstein, S. Avidan, and W. T. Freeman. Best-buddies similarity for robust template matching. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2021–2029. IEEE, 2015.

[15] R. Díaz, S. Hallman, and C. C. Fowlkes. Detecting dynamic objects with multi-view background subtraction. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 273–280. IEEE, 2013.

[16] R. Díaz, M. Lee, J. Schubert, and C. C. Fowlkes. Lifting gis maps into strong geometric context for scene understanding. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9, March 2016.

[17] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014.

[18] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. Pascal VOC2007 results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[19] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 32(9):1627–45, Sept. 2010.

[20] S. Fidler, R. Mottaghi, A. Yuille, and R. Urtasun. Bottom-up segmentation for top-down detection. In *CVPR*, 2013.

[21] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[22] J. Frahm, P. Fite-Georgel, and D. Gallup. Building Rome on a cloudless day. In *ECCV*, 2010.

[23] Y. Furukawa. CMVS. http://grail.cs.washington.edu/software/cmvs/.

[24] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski. Towards Internet-scale multi-view stereo. In *CVPR*, 2010.

[25] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *TPAMI*, 1(1):1–14, 2010.

[26] R. Gherardi, M. Farenzena, and A. Fusiello. Improving the efficiency of hierarchical structure-and-motion. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010.

[27] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. http://people.cs.uchicago.edu/ rbg/latent-release5/.

[28] S. Gould. Darwin: A framework for machine learning and computer vision research and development, v1.9. *JMLR*, 2012.

[29] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

[30] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *ECCV*. 2014.

[31] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[32] M. Havlena and K. Schindler. Vocmatch: Efficient multiview correspondence for structure from motion. In *European Conference on Computer Vision*, pages 46–60. Springer, 2014.

[33] J. Hays and A. A. Efros. im2gps: estimating geographic information from a single image. In *CVPR*, 2008.

[34] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV*, 2009.

[35] V. Hedau, D. Hoiem, and D. Forsyth. Thinking inside the box: Using appearance models and context based on room geometry. In *ECCV*. 2010.

[36] J. Heinly, J. L. Schonberger, E. Dunn, and J.-M. Frahm. Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3287–3295, 2015.

[37] D. Hoiem, A. Efros, and M. Hebert. Putting Objects in Perspective. *CVPR*, 2:2137–2144, 2006.

[38] D. Hoiem, A. A. Efros, and M. Hebert. Automatic photo pop-up. In *ACM Transactions on Graphics (TOG)*, 2005.

[39] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. *ICCV*, 1:654–661, 2005.

[40] D. Hoiem, A. A. Efros, and M. Hebert. Closing the loop in scene interpretation. In *CVPR*, 2008.

[41] L. Kneip, D. Scaramuzza, and R. Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *CVPR*, 2011.

[42] L. Ladickỳ, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr. What, where and how many? combining object detectors and CRFs. In *ECCV*. 2010.

[43] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide Pose Estimation using 3D Point Clouds. In *ECCV*, 2012.

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

[45] D. Lin, S. Fidler, and R. Urtasun. Holistic scene understanding for 3d object detection with rgbd cameras. In *ICCV*, 2013.

[46] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2):91–110, Nov. 2004.

[47] M. Maire, S. X. Yu, and P. Perona. Hierarchical scene annotation. In *BMVC*, 2013.

[48] A. Martinović, M. Mathias, J. Weissenberg, and L. Van Gool. A three-layered approach to facade parsing. In *ECCV*. 2012.

[49] K. Matzen and N. Snavely. Nyc3dcars: A dataset of 3d vehicles in geographic context. In *ICCV*, 2013.

[50] R. Mohr, L. Quan, and F. Veillon. Relative 3d reconstruction using multiple uncalibrated images. *The International Journal of Robotics Research*, 14(6):619–632, 1995.

[51] L. Moisan, P. Moulon, and P. Monasse. Automatic homographic registration of a pair of images, with a contrario elimination of outliers. *Image Processing On Line*, 2:56–73, 2012.

[52] P. Moulon, P. Monasse, and R. Marlet. Adaptive structure from motion with a contrario model estimation. In *ACCV*. 2012.

[53] P. Moulon, P. Monasse, and R. Marlet. Global fusion of relative motions for robust, accurate and scalable structure from motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3248–3255, 2013.

[54] P. Moulon, P. Monasse, R. Marlet, and Others. Openmvg. an open multiple view geometry library. `https://github.com/openMVG/openMVG`.

[55] D. M. Mount and S. Arya. Ann: library for approximate nearest neighbour searching. 1998.

[56] M. Muja and D. G. Lowe. Flann, fast library for approximate nearest neighbors. In *International Conference on Computer Vision Theory and Applications (VISAPP09)*. INSTICC Press, 2009.

[57] H. Riemenschneider, S. Sternig, M. Donoser, P. M. Roth, and H. Bischof. Hough regions for joining instance localization and segmentation. In *ECCV*. 2012.

[58] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 23(3):309–314, 2004.

[59] M. Rumpler, A. Irschara, A. Wendel, and H. Bischof. Rapid 3d city model approximation from publicly available geographic data sources and georeferenced aerial images. In *CVWW*, 2012.

[60] S. Satkin and M. Hebert. 3dnn: Viewpoint invariant 3d geometry matching for scene understanding. In *ICCV*, 2013.

[61] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2D-to-3D matching. *ICCV*, 2011.

[62] T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *European Conference on Computer Vision*, pages 752–765. Springer, 2012.

[63] T. Sattler, C. Sweeney, and M. Pollefeys. On sampling focal length values to solve the absolute pose problem. In *European Conference on Computer Vision*, pages 828–843. Springer, 2014.

[64] A. Saxena, M. Sun, and A. Y. Ng. Learning 3-d scene structure from a single still image. In *ICCV*, 2007.

[65] J. L. Schonberger, A. C. Berg, and J.-M. Frahm. Paige: pairwise image geometry encoding for improved efficiency in structure-from-motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1009–1018, 2015.

[66] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[67] Y. Sheikh, O. Javed, and T. Kanade. Background Subtraction for Freely Moving Cameras. *ICCV*, pages 1219–1225, Sept. 2009.

[68] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[69] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*. 2006.

[70] N. Snavely, S. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. *ACM Transactions on Graphics (TOG)*, 2006.

[71] N. Snavely, S. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. *ACM Transactions on Graphics (TOG)*, 2006.

[72] N. Snavely, S. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *IJCV*, 2007.

[73] N. Snavely, I. Simon, M. Goesele, R. Szeliski, and S. M. Seitz. Scene Reconstruction and Visualization From Community Photo Collections. *Proceedings of the IEEE*, 98(8):1370–1390, Aug. 2010.

[74] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. *CVPR*, 1999.

[75] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. Ieee, 2008.

[76] L. Svarm, O. Enqvist, M. Oskarsson, and F. Kahl. Accurate localization and pose estimation for large 3d models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 532–539, 2014.

[77] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE transactions on pattern analysis and machine intelligence*, 32(5):815–830, 2010.

[78] P. H. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156, 2000.

[79] A. Torralba and A. Efros. Unbiased look at dataset bias. *CVPR*, pages 1521–1528, 2011.

[80] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustmenta modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999.

[81] H. Uchiyama, H. Saito, M. Servieres, and G. Moreau. AR GIS on a physical map based on map image retrieval using LLAH tracking. In *MVA*, 2009.

[82] L. Wang and U. Neumann. A robust approach for automatic registration of aerial images with untextured aerial lidar data. In *CVPR*, 2009.

[83] S. Wang, S. Fidler, and R. Urtasun. Holistic 3d scene understanding from a single geo-tagged image. 2015.

[84] C. Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 127–134. IEEE, 2013.

[85] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

[86] R. Yang and R. Yang. Action segmentation and recognition based on depth hog and probability distribution difference. In *ICIC*. 2014.

[87] A. R. Zamir and M. Shah. Image geo-localization based on multiplenearest neighbor feature matching usinggeneralized graphs. *IEEE transactions on pattern analysis and machine intelligence*, 36(8):1546–1558, 2014.

[88] B. Zeisl, T. Sattler, and M. Pollefeys. Camera pose voting for large-scale image-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2704–2712, 2015.