



University of Kentucky
UKnowledge

Theses and Dissertations--Computer Science

Computer Science


2018

Deep Probabilistic Models for Camera Geo-Calibration

Menghua Zhai

University of Kentucky, ted@cs.uky.edu

Author ORCID Identifier:

 <https://orcid.org/0000-0001-5874-238X>

Digital Object Identifier: <https://doi.org/10.13023/etd.2018.458>

[Right click to open a feedback form in a new tab to let us know how this document benefits you.](#)

Recommended Citation

Zhai, Menghua, "Deep Probabilistic Models for Camera Geo-Calibration" (2018). *Theses and Dissertations--Computer Science*. 74.

https://uknowledge.uky.edu/cs_etds/74

This Doctoral Dissertation is brought to you for free and open access by the Computer Science at UKnowledge. It has been accepted for inclusion in Theses and Dissertations--Computer Science by an authorized administrator of UKnowledge. For more information, please contact UKnowledge@lsv.uky.edu.

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Menghua Zhai, Student

Dr. Nathan Jacobs, Major Professor

Dr. Miroslaw Truszczyński, Director of Graduate Studies

Deep Probabilistic Models for Camera Geo-Calibration

DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for the
degree of Doctor of Philosophy in the
College of Engineering at the
University of Kentucky

By
Menghua Zhai
Lexington, Kentucky

Director: Dr. Nathan Jacobs, Professor of Computer Science
Lexington, Kentucky 2018

Copyright© Menghua Zhai 2018
ORCID: 0000-0001-5874-238X

ABSTRACT OF DISSERTATION

Deep Probabilistic Models for Camera Geo-Calibration

The ultimate goal of image understanding is to transfer visual images into numerical or symbolic descriptions of the scene that are helpful for decision making. Knowing when, where, and in which direction a picture was taken, the task of geo-calibration makes it possible to use imagery to understand the world and how it changes in time. Current models for geo-calibration are mostly deterministic, which in many cases fails to model the inherent uncertainties when the image content is ambiguous. Furthermore, without a proper modeling of the uncertainty, subsequent processing can yield overly confident predictions. To address these limitations, we propose a probabilistic model for camera geo-calibration using deep neural networks. While our primary contribution is geo-calibration, we also show that learning to geo-calibrate a camera allows us to implicitly learn to understand the content of the scene.

KEYWORDS: computer vision, geo-calibration, deep neural networks

Author's signature: Menghua Zhai

Date: December 6, 2018

Deep Probabilistic Models for Camera Geo-Calibration

By
Menghua Zhai

Director of Dissertation: Nathan Jacobs

Director of Graduate Studies: Mirosław Truszczyński

Date: December 6, 2018

ACKNOWLEDGMENTS

I would like to express the deepest appreciation to my advisor, Professor Nathan Jacobs, who not only financed my Ph.D. career, but also supported my research with his profound insight and knowledge. Over the years I have benefited from guidance, expertise, and patience. I have also learned a lot from his efficient work style and the way he attacks problems. I honestly felt lucky to have had the opportunity to work alongside him and I am proud of what we have accomplished together.

I would like to take this opportunity to give my thanks to several individuals who helped me through the path: Jinze Liu, who first got me enrolled to the graduate school; Qiang Ye, Judy Goldsmith, and Andrew Klapper, for intriguing my desire for higher-level knowledge. A huge appreciation to other members of my advisory committee, Ruigang Yang and Ramakanth Kavuluru, for their invaluable feedback during my defense.

I have had the privilege of working and collaborating with many individuals, including: Zachary Bessinger, Tawfiq Salem, Connor Greenwell, Ryan Baltenberger, Neil Moore, Samuel Schuler, and others. In the end, I would like to thank my friend and colleague, Scott Workman, with whom I worked closest and accomplished a great deal of research projects.

Table of Contents

Acknowledgments	iii
Table of Contents	iv
List of Figures	vi
List of Tables	xi
Chapter 1 Introduction	1
1.1 Background	2
1.2 Main Contributions	6
Chapter 2 Flexible Probabilistic Model for Geo-Calibration	10
2.1 Introduction	10
2.2 Approach	12
2.3 Evaluation	15
2.4 Conclusions	17
Chapter 3 A Constraint for Camera Pose	19
3.1 Introduction	19
3.2 Problem Formulation	22
3.3 Horizon Priors from Global Image Context	23
3.4 Horizon-First Vanishing Point Detection	25
3.5 Evaluation	30
3.6 Conclusion	34
Chapter 4 A Constraint for Time and Location	35
4.1 Introduction	35
4.2 Related Work	36
4.3 Estimating Geo-Temporal Image Features	38

4.4	Experiments	39
4.5	Conclusion	45
Chapter 5	A Constraint for Location and Orientation	47
5.1	Introduction	47
5.2	Related Work	48
5.3	Cross-view Supervised Training	51
5.4	Evaluation and Applications	54
5.5	Conclusion	62
Chapter 6	Summary	64
	Bibliography	66
	Vita	80

LIST OF FIGURES

1.1	A simplified flowchart of our approach. The algorithm inputs (images, time and manual annotations) are fed to different constraint models which work as constraint functions. The probabilistic distribution over complete geo-calibration parameters are then simulated jointly by fusing these constraint functions in a general model for geo-calibration.	6
2.1	We match objects in an image (top) to GIS reference data (left) to estimate a probability distribution over locations (right). The map (left) also shows the true camera location (green dot) and the top scoring sampled cameras (red arrows).	11
2.2	(left) A map showing two houses (black) and a hypothetical ground-truth camera frustum. (middle) The PDF (red is high probability) over location for a 2D camera model, in which constraints are based on distances between the camera and GIS objects. (right) The 4D model we propose enables richer constraints, which leads to a more accurate PDF.	12
2.3	Since the exact correspondence between an image pixel and a point on a GIS object is unknown, we compute the distance to the closest point of the object to compare with the estimated distance.	14
2.4	Performance comparison among methods. (left) Total distance. (right) Total area. For the grid method the finest sampling over spatial dimensions is 30m. .	16
2.5	Qualitative results for two query images (top). The resulting PDFs (middle) and map visualization of the top-15 samples (bottom) show that the proposed scoring function realistically captures the uncertainty.	18
3.1	An example result of our method. (left) Horizon line candidates, colored by their scores (red means high score), and the true horizon line (green dash). (right) The horizon line (magenta) estimated by our algorithm is very close to the true horizon line (green dash). Line segments are color coded based on the most consistent detected vanishing point.	19

3.2	Algorithm overview: 1) use global image context to estimate a prior over horizon lines (Section 3.3); 2) extract line segments; 3) identify the zenith VP (Section 3.4.1); 4) sample horizon line candidates consistent with the zenith VP (Section 3.4.2); 5) find VPs on horizon line candidates (Section 3.4.2); and 6) select the best horizon line based on the VPs it contains (Section 3.4.3). . . .	20
3.3	In homogeneous coordinates, lines (red lines) are defined by the normal (red arrow) of the plane (red triangle) they form with the origin (green dot). Two lines form a great circle (blue circle), whose normal (blue arrow) is their common point (blue dot) in homogeneous coordinates.	23
3.4	Example images from our training dataset (Section 3.3.2), each overlaid with the ground-truth horizon line.	24
3.5	Global image context imposes a strong prior on horizon line location. The output of our CNN is visualized as an overlaid heatmap, with red indicating more likely locations. For each image, the ground-truth horizon line (dash green) and the line that maximizes the prior (red) are shown.	26
3.6	Our method samples more horizon line candidates (red) near the ground truth (green dash) with (middle) global image context than without (left). In the case of sampling with global image context, the offset PDF, $p(o I)$ (blue curve), is fit from the CNN categorical probability distribution outputs (hollow bins). For clarity, we only show a reduced number of horizon line candidates and bins. . .	28
3.7	A ring-like graph (left) is converted into three nearly linear subgraphs (right) by partitioning around a node with minimal degree. For the subgraphs, the red node is mandatory, the dashed nodes are excluded, and a subset of the solid nodes are selected using dynamic programming.	29
3.8	For three benchmark datasets, the fraction of images (y-axis) with a horizon error less than a threshold (x-axis). The AUC for each curve is shown in the legend. For additional details see Section 3.5.	30
3.9	Two images where horizon line estimates are much better with global context (left) than without (right).	33
3.10	Example results produced by our method. (rows 1 and 3) Line segments color coded based on the most consistent VP, the ground-truth (green dash), and detected horizon lines (magenta). For clarity only the top two horizontal VPs are shown. (rows 2 and 4) The line segments (dots) and their VPs (rings) represented in homogeneous coordinates. (last column) Two failure cases of our method, caused by irregularly shaped objects (bottom) and short edges (top).	33

4.1	An overview of the proposed network architecture. Our approach learns mid-level feature representations for time (orange), location (blue), and image appearance (green) by optimizing for a set of conditional time and location estimation tasks.	38
4.2	Relating image appearance to the image context representation by visualizing the images that have the highest activation at three different neurons.	41
4.3	The time series of two neurons for a week of webcam imagery, with images showing the scene at various points. It appears that the top neuron is related to the diurnal cycle and the bottom is related to fogginess.	41
4.4	Cross-correlation scores between transient attributes and three representations. For each attribute, we show the peak correlation score. For a majority of attributes our proposed representation is more highly correlated than the ImageNet baseline.	42
4.5	Comparing mid-level features for transient attribute estimation. (left) Initializing from the weights of our proposed approach versus a network trained for image classification and a random baseline. (right) Initializing from our method, trained on different datasets.	42
4.6	Quantitative evaluation of localization performance shown as a cumulative distance error plot for the <i>YFCC</i> dataset.	43
4.7	Visualizing the time estimation loss for two cameras and varying number of images (darker is lower). The red arrows show the gradient and the green dot is the true location.	44
4.8	Quantitative evaluation of time estimation (month and hour) performance shown as a cumulative error plot for the <i>YFCC</i> dataset. Both methods performance better than the random chance.	45
4.9	Marginal probability of GMT predictions by varying latitude and longitude. . .	45
5.1	We learn to predict the ground-image segmentation directly from an aerial image of the same location, thereby transferring the semantics from the ground to the aerial image domain.	48

5.2	A visual overview of our network architecture. We extract features from an aerial image using the VGG16 architecture and form a hypercolumn using the PixelNet approach. These features are processed by three networks that consist of 1×1 convolutions: network A converts the hypercolumn into semantic features; network S extracts useful features from the aerial image for controlling the transformation; and network F defines the transformation between view-points. The transformation is applied, T , to the aerial semantic features to create a ground-level semantic labeling.	49
5.3	Examples of aligned aerial/ground image pairs from our dataset. (row 1) In the aerial images, north is the up direction. In the ground images, north is the central column. (row 2-4) Image dependent receptive fields estimated by our algorithm as follows: 1) fix ground locations (y, x) (locations in squares); 2) select all (i, j) (locations in contours) with high $\tilde{F}(i, j, y, x, S(I_a; \Theta_S))$ values. Corresponding fields between the aerial image and the ground image are shown in the same color.	54
5.4	Example outputs from our weakly supervised learning method on test images. For each aerial image (top), we show the pixel-level labeling inferred by our model, which uses only noisy ground image segmentation as labels. We visualize three classes: <i>road</i> (red), <i>vegetation</i> (green), and <i>man-made</i> (blue).	55
5.5	An example from the ISPRS dataset [94]. (left) Near infrared image; (middle) The same image after pre-processing; (right) Ground-truth annotation of the image.	56
5.6	Performance comparison of different initialization methods on the ISPRS segmentation task. The x -axis is the number of training images and the y -axis is average precision.	58
5.7	Qualitative results of orientation predictions on Cityscapes dataset (top) and CVUSA (bottom). The I_g , L_g and $L_{g'}$ are stacked vertically on the left side of the aerial image. We visualize three classes on the labels: <i>road</i> (red), <i>vegetation</i> (green), and <i>man-made</i> (blue). The discrete PDFs of the ground camera orientation are visualized with red arrows, whose lengths indicate the magnitudes. In the CVUSA results, the ground truth (green) and the optimal prediction (blue) are also shown with the orientation PDF. The last prediction result is a typical failure case of our method, where the scene is symmetric from the aerial point of view.	59
5.8	Histogram of the orientation errors on the CVUSA dataset.	59

5.9	Fine-grained geo-calibration results on CVUSA. (left) From top to bottom are the I_g , L_g , and $L_{g'}$ respectively. We visualize three classes on the labels: <i>road</i> (red), <i>vegetation</i> (green), and <i>man-made</i> (blue). (right) Orientation flow map (red), where the arrow direction indicates the optimal direction at that location and length indicates the magnitude. We also show the optimal prediction and the ground-truth frustums in blue and green respectively.	60
5.10	Synthesized ground-level views. Each row shows an aerial image (left), its corresponding ground-level panorama (top-right), and predicted ground-level panorama (bottom-right).	61

LIST OF TABLES

3.1	Algorithm parameters (given an $H \times W$ image).	30
3.2	Component error analysis (AUC).	32
5.1	Per-Class Precision on the ISPRS Segmentation Task	58

Chapter 1

Introduction

“Geo-locating suspects in real time? I’ve never seen anything like it.”

– Morgan Freeman in *Transcendence* (2014 film)

The ultimate goal of image understanding is to transfer visual signals (images/videos) into abstract symbolic descriptions of the world which are helpful for decision making. However, understanding images is not a trivial task for machines. In order to handle various complicated scenarios, researchers divide image understanding into different computer vision tasks: pedestrian detection for autonomous driving, face recognition for security systems, and image retrieval for search engines.

For many applications, knowing when, where, and in which direction a picture was taken is important scene understanding. However, most images do not carry such information. We refer to the task of estimating this information as *geo-calibration*. Our thesis focuses on developing geo-calibration algorithms.

Most recent geo-calibration algorithms are deterministic systems [8, 27, 70, 60]. Deterministic systems have some obvious drawbacks: 1) they cannot model the inherent uncertainties from images of ambiguous scenes, and 2) without a proper modeling of the uncertainty, the subsequent process can yield overly confident predictions. To address these problems, we propose to build probabilistic systems for camera geo-calibration.

We also show that learning to geo-calibrate a camera allows us to learn about the scene of the image. As an essential part of understanding the scene, how to extract image features is an important problem has been studied for decades. In recent years, learning based approaches for image feature extraction like convolutional neural networks have drawn a lot of attention due to the fact that they do not need expert knowledge about the target data. However, most of these feature learning methods require a large quantity of manually labeled datasets. In this thesis, we propose alternative ways to learn image features when

the manually labeled data is either insufficient or absent.

1.1 Background

To give a better understanding of our work, we would like to introduce some related computer vision concepts before we discuss the main topics. We start with the definition of geo-calibration, along with its literature. Next, we introduce the history of deep learning methods for geo-calibration, and data-driven scene matching approaches that use aerial imagery. In the end, we cover some related work of weakly supervised learning techniques.

1.1.1 Geo-Calibration of Outdoor Images

Automatic geo-calibration of outdoor images continues to grow in importance as a direct result of the increasing amount of imagery available via the Internet. Solving this problem is of great value for a wide variety of fields, with potential applications ranging from the forensic sciences [107] to environmental monitoring [138]. Conceptually, the geo-calibration task includes identifying the camera pose (*pose estimation*), the camera geographic location (*geo-localization*), and the time when the image was captured (*time estimation*).

- **Pose Estimation:** Pose estimation is to estimate the camera yaw, pitch, and roll angles, $(\theta_{yaw}, \theta_{pitch}, \theta_{roll})$ from an image. In the context of geo-calibration, the yaw angle θ_{yaw} is referred to as the geographic orientation angle of the camera. Li et al. [70] exploit geo-registered 3D points clouds to estimate camera pose. Vo et al. [114] propose a geo-localization network that can regress the geo-orientation angle of the camera. Agarwal et al. [2] keep track of the camera pose changes by matching SIFT [77] feature points detected between the input image and Google Street View panoramas. Horizon line detection is an essential step for some pose estimation algorithms, because horizon lines are closely related to camera pitch and roll angles. Collins and Weiss [18] formulate horizon line detection as a statistical estimation problem on the Gaussian Sphere. More recent work has explored the use of dual space [69, 140] representations. Among the clustering-based approaches, Xu et al. [133] improve this pipeline by introducing a new point-line consistency function that models errors in the line segment extraction step.
- **Geo-Localization:** The task of geo-localization estimates the geographic location, (lat, lon) , of the camera given image/video inputs. As an important step of

geo-localization, extracting location-dependent features from image data has drawn a great deal of attention from the vision community [55, 52, 54]. The common trend amongst these methods is that they take advantage of a large dataset of geo-referenced images. Hays and Efros [42] use a data-driven scene matching approach to localize a query image using a large dataset of geo-tagged images. Doersch *et al.* [27] extract location-dependent features that capture the relative appearance differences of large cities. Lin *et al.* [73] localize a ground-level image by learning the relationship between pairs of ground and aerial images¹ of the same location. Other techniques focus on urban environments and infer location using local image descriptors [101, 105]. Many other cues exist, such as the skyline [8, 93], sky appearance [67, 127], and shadows [59, 132].

- **Time Estimation:** The goal of time estimation is to estimate the time when the image was captured. Depending on the application, the accuracy of predictions range from hour to year. Matzen and Snavely [83] predict timestamps for photos by matching against a time-varying reconstruction of a scene. Hill *et al.* [43] estimate the time of the day by measuring the light intensity profiles captured by cameras. Methods are proposed to date yearbook photos by analyzing human appearance [98, 35]. Lee *et al.* [68] find visual patterns in buildings, relate them to certain time periods.

In conclusion, most of the geo-calibration algorithms we mention above rely on finding visual cues from input images. To extract useful visual features, a majority of these algorithms require strong human expertise about the data, thus, they can only be applied in limited scenarios. Therefore, we contribute to developing geo-calibration algorithms without the need of strong human expertise about the data.

1.1.2 Deep Learning in Geo-Calibration

In recent years, deep neural networks (DNNs) were proven to be extremely successful in many computer vision areas. It is widely accepted that DNNs have excellent performance in high-level visual feature learning. This valuable ability provides researchers powerful tools to solve the challenging geo-calibration tasks. Weyand and Kostrikov *et al.* [122] propose a convolutional neural network (CNN) to predict the geographic location of the input image. Walch *et al.* [117] aggregate learned CNN features with LSTM to generate

¹In the context of our thesis, we do not distinguish between satellite imagery and aerial imagery

global image representation for camera localization. Studies about camera pose estimation [137, 130, 45] have been done by identifying the horizon line with CNNs (one can derive the camera roll and pitch angles from the horizon line position giving the camera focal length).

Closely related to camera geo-calibration, problems of identifying camera intrinsics using deep neural networks have also been studied. Workman *et al.* [125] develop a CNN to estimate the camera focal length. Kendall *et al.* [60] propose a neural network that can identify 6-DOF camera parameters for specific scenes.

All these geo-calibration methods share the common property that they make the predictions directly out of DNNs. Besides these methods, there exists another big category of methods that geo-calibrate cameras using aerial imagery as geo-registered database. We will discuss them in the following section.

1.1.3 Ground-to-Aerial Geo-Calibration

Data-driven scene matching approaches are commonly used in geo-calibration. When localizing a camera, data-driven algorithms search for k-nearest neighbors of the query image in a database of the geo-tagged imagery. To reduce the work of querying, the searching process is usually carried out in a feature space of an applicable number of dimensions [42, 71, 136]. However, due to limited accuracies of GPS signals and the biased distribution of human population, geo-tagged ground-level images collected from smart devices are usually noisy and sparse in geographic space. To achieve better geo-calibration results, we need alternative image resources with more accurate geographic tags and more complete spatial coverage.

Benefiting from the fast-growing monitoring satellite and drone markets, geo-tagged aerial imagery has become more publicly accessible than ever. Aerial imagery downloading service like Microsoft Bing Map provides aerial images of various resolutions, which cover most areas of the world with accurate geographic registration, making aerial imagery a rich source for reference databases. We refer to geo-calibration methods which use the aerial imagery for geographic reference as ground-to-aerial geo-calibration. Data used for ground-to-aerial geo-calibration is referred to as cross-view pairs. A cross-view pair consists of a ground image and an aerial image at the same location (sometimes aligned in orientation).

Recent work on ground-to-aerial geo-calibration [73, 74, 126, 128] has shown that convolutional neural networks are efficient for extracting features from geo-tagged aerial imagery that can be matched to features extracted from the ground imagery. Vo *et al.* [114]

extend this line of work, demonstrating improved geo-localization performance by applying an auxiliary loss function to regress the ground-level camera orientation with respect to the aerial image. Hu and Feng *et al.* [104] achieve the state-of-the-art performance in geo-localization by extracting cross-domain global features using the NetVLAD [6] technique.

The key for the success of these methods is to learn descriptive features for ground-to-aerial imagery matching. In the following section, we will introduce literature about feature learning techniques.

1.1.4 Feature Learning with Weak Annotations

Manually engineering visual features is a common practice for computer vision tasks. One of successful examples of manual feature is SIFT [77], which is widely used by many computer vision algorithms. In recent years, automatic feature engineering has drawn more attention. Compared to the manual feature engineering, automatic feature engineering allows to learn useful features with much less (if not none) knowledge about the data. Among these learning approaches, deep neural networks are intensively studied. The most common approach for deep feature learning is full supervision, which usually requires a large number of manual annotations [135, 142, 121]. Unfortunately, such data is not always available for certain tasks or requires a huge amount of human labor. In contrast, unlabeled data is often plentiful (*i.e.* cellphone photos in Instagram or Youtube videos).

To lavage these unlabeled data, recent work has explored self-supervision methods (sometimes referred to as unsupervised learning or pretext tasks) for training deep neural networks that capture useful visual representations [26, 91]. These methods typically exploit some known quantity of the data, like pixel color values, to avoid expensive manual annotation. By learning these quantities associated with the data, one can obtain useful features of the image. For example, Zhang et al. [139] show how synthesizing colors for a grayscale image is a powerful pretext task for learning visual representations. Pathak et al. [90] exploit low-level motion-based grouping cues for unsupervised feature learning. As one of new learning techniques that address the lack of massive amounts of labeled data, domain adaptation forces the feature generating model to adapt another feature domain so that it improves the performance when dealing with new input data [30, 31, 97, 119, 145]. Duan *et al.* [28] propose to learn a linear projection to transfer from the source feature subspace to the target subspace. Sohn *et al.* [106] improve the performance of video face recognition using an adversarial-based approach to adapt the network to unlabeled video imagery.

We have explained four related computer vision concepts so far. In the following sec-

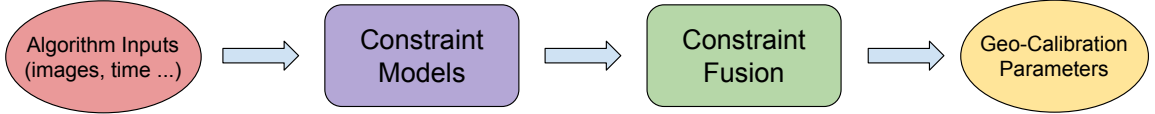


Figure 1.1: A simplified flowchart of our approach. The algorithm inputs (images, time and manual annotations) are fed to different constraint models which work as constraint functions. The probabilistic distribution over complete geo-calibration parameters are then simulated jointly by fusing these constraint functions in a general model for geo-calibration.

tion, we are going to discuss our main contributions.

1.2 Main Contributions

We make contributions in three main areas: 1) we proposed a reflexible probabilistic model to jointly estimate the geo-calibration parameters, 2) we decomposed the full geo-calibration into several partial geo-calibration tasks, each of which provides a strong constraint function that fits into the probabilistic model, and 3) we show that learning to geo-calibrate a camera is a useful pretext for image feature learning.

The first two contributions make our main approach. Our algorithm feeds inputs (images, time, and manual annotations) to different constraint functions, which we can fuse together to simulate the joint distribution over the complete geo-calibration parameters. We present the flowchart of our approach in Figure 1.1. Furthermore, based on the belief that the content of the image is usually dictated by the camera pose, geographic location and image capture time. We treat geo-calibration information as pretext task and show that they are useful for image feature learning.

1.2.1 Flexible Probabilistic Model for Geo-Calibration

We design a flexible probabilistic model to estimate the probability over all geo-calibration parameters. First, we construct a scoring function to model the process of camera projecting the scene to image frame at given pose, location, and time. The scoring function measures the “fitness” between these geo-calibration parameters and the appearance of the image. It consists of a series of constraint functions, The scoring function is proportional to the probability distribution over the geo-calibration parameters. Since the integral over all possible parameters is intractable, we propose a Markov Chain Monte Carlo (MCMC) sampling approach to approximate the underlying probability distribution. Our model is flexible because it allows different constraint functions contribute to the scoring function. The details of this model can be found in Chapter 2.

1.2.2 Developing Constraint Functions

The scoring function of the probabilistic model is defined as a weighted summation of a series of constraint functions. A constraint function outputs a score (or probability) of the input geo-calibration parameters given the input image. In the rest of the work, we focus on developing good constraint functions for different sets of geo-calibration parameters.

- **Constraint for Camera Pose:** We build the constraint function for camera roll and pitch angles. Assuming the intrinsic parameters are known, camera roll and pitch angles can be derived from the horizon line position on the image. Compared to directly detecting the camera pose, identifying the horizon line is easier because it can be explicitly estimated from the scene layout. In our work, we create a convolutional neural network to estimate the prior distribution over the horizon line. For each horizon line candidate sampled from the prior distribution, we assign a consistency score to it by measuring the consistencies of vanishing points detected on it. The constraint function output is defined as the multiplication of the prior probability of the associated horizon line and its consistency score. We present the algorithm details in Chapter 3.
- **Constraint for Time and Location:** In this work, we estimate the probability over the camera geographic location and the capture time of the input image. Our network has two branches that predict the image capture time and the geographic location simultaneously. For time estimation, our network is able to predict the joint distribution over the month of the year and the UTC hour of the day when the input image was captured. For camera geo-localization, our network outputs the discrete distribution over the geographic location of the camera. Better results can be procured with the image capture time as auxiliary input (if it is known). We present the algorithm details in Chapter 4.
- **Constraint for Location and Orientation:** The previous algorithm only estimates locations approximately due to the limited capacity of the deep neural network. In this work, we propose a ground-to-aerial geo-calibration method that can further refine the prediction results and estimate the camera geo-orientation. We designed a network that can jointly learn the semantic layout of the aerial image and its projection to the ground-level perspective. When computing the value of the constraint function, we feed the aerial image of a given location and orientation to the network. By comparing the aerial-to-ground layout with the real layout of the

ground image, we are able to measure the consistency between the two and use it as the value of our constraint function. We present the algorithm details in Chapter 5.

1.2.3 Geo-Calibration for Feature Learning and Image Understanding

Full supervision algorithms are powerful tools to learn useful image features. However, full supervision algorithms require large number of annotated data, which takes a lot of human labor and is usually unavailable. On the other hand, images with annotations like tags and GPS locations uploaded by individual users or captured automatically by smart devices are plentiful on the Internet. To use these image resources to understand the world, we explore two approaches that use time and locations as weak annotations: 1) *ground-to-aerial segmentation*, and 2) *geo-temporal feature learning* with time/location tagged images.

- **Ground-to-Aerial Segmentation:** Semantic segmentation plays an important role in image understanding. The training for semantic segmentation tasks require a large number of annotated data. However, most large scale annotated datasets like ImageNet [95] and COCO [75] are mostly made of ground-level images. By comparison, annotated datasets for aerial imagery are rare. To deal with this problem, we proposed a DNN to learn semantic segmentation for aerial images with only ground-level annotations. Our method exploits the spatial correspondences between images in the cross-view pair, where the aerial imagery and ground imagery are associated by location and geo-orientation. In this work, we transfer the semantic knowledge from the ground domain to the aerial domain by identifying their latent geometric correspondences. Similar to *Reprojection Losses* [33, 38, 144, 134], which are proven to be successful in monocular depth estimation, our network learns the geometric projection from aerial to ground, and to semantically segment the aerial images jointly. We present the learning details in Chapter 5
- **Location and Time as Pretext Task:** Smart devices like cellphones and webcams get more sophisticated nowadays. Most of them are equipped with clocks and GPS modules. Thanks to the development of social networks, researchers now can find large amount of images online that are tagged with time and geographic locations. In Chapter 4, we propose an algorithm that treats the location and time as pretext tasks to learn useful image features. By learning to predict the geographic location and the

capture time of the query image, our network learns to extract geo-temporal features from the image.

In summary, the main contributions of our work include developing a probabilistic framework to solve geo-calibration problems, and proposing new methods to learn image features with geo-calibration information. We are going to present our approaches in detail in the following chapters.

Chapter 2

Flexible Probabilistic Model for Geo-Calibration

2.1 Introduction

Automatic image localization continues to grow in importance as a direct result of the increasing amount of imagery available via the Internet. Conceptually the task is straightforward; given an image, identify the location it was captured in the world directly from image data. Solving this problem is of great value for a wide variety of fields, with potential applications ranging from the forensic sciences [107] to crowd-sourced environmental monitoring [138].

However, recognizing the geo-location and geo-orientation of an arbitrary outdoor image is an extremely challenging task. Many methods have been proposed; the most common approach is to build a large database of images with known location and localize a query image using either local [71, 101] or global [42, 27] image features. This approach is not applicable when no nearby ground-level imagery exists in the reference database, such as when the image was not captured near a popular tourist destination. Even when reference imagery is available, the appearance of the objects may not be visually distinctive, for example a train track or a body of water.

Instead of matching visually against a reference image set, we exploit the large quantities of publicly available geospatial data to build a geographic database containing geometric information for objects of interest in the world, such as roads, churches, bodies of water, water towers and golf courses. Given a query image, we identify visible objects of interest in the image and apply the Metropolis-Hastings MCMC algorithm to randomly sample possible cameras. We assign a score to each hypothetical camera and use these

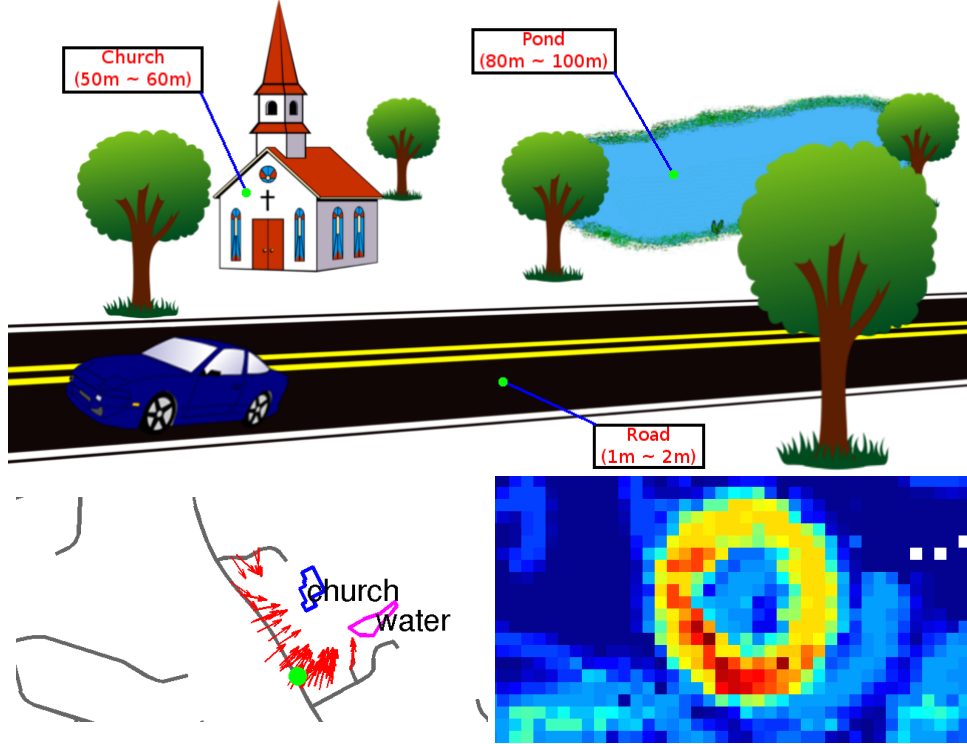


Figure 2.1: We match objects in an image (top) to GIS reference data (left) to estimate a probability distribution over locations (right). The map (left) also shows the true camera location (green dot) and the top scoring sampled cameras (red arrows).

samples to approximate the probability distribution over the camera parameters and extract candidate locations. Figure 2.1 gives an brief view of our approach.

Our key contributions are: 1) a flexible approach to the camera geo-calibration problem that supports priors over camera parameters and constraints that relate image annotations, camera geometry, and a geographic database and 2) an extensive comparison of this approach to uniform and grid-based sampling on real-world data.

2.1.1 Related Work

Self-localization has been heavily studied in the robotics community. The task is to estimate the probability density function (PDF) over the robot’s state space (location and orientation). Early methods attempted to estimate this density by discretizing the state space [15]. These grid-based approaches suffered from large computational overhead and memory requirements. To overcome these limitations, probabilistic particle-based methods like MCMC were investigated [23, 32, 41, 86]. The idea is to approximate the probability density using randomly drawn samples, while maintaining performance even for high dimensional state spaces.

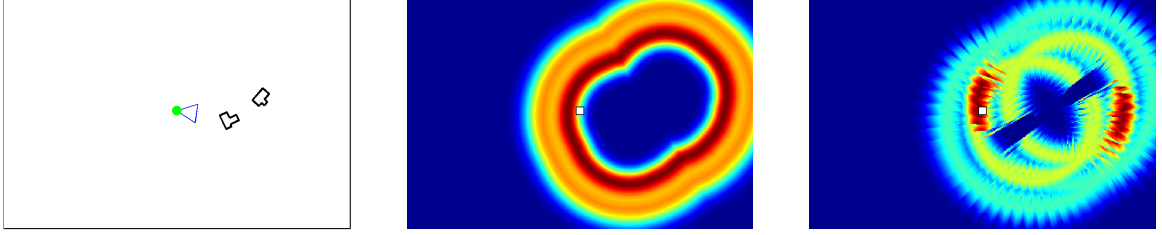


Figure 2.2: (left) A map showing two houses (black) and a hypothetical ground-truth camera frustum. (middle) The PDF (red is high probability) over location for a 2D camera model, in which constraints are based on distances between the camera and GIS objects. (right) The 4D model we propose enables richer constraints, which leads to a more accurate PDF.

Extracting location-dependent features from image data has drawn a great deal of attention from the vision community [55, 52, 54]. The common trend amongst these methods is that they take advantage of a large dataset of geo-referenced images. Hays and Efros [42] use a data-driven scene matching approach to localize a query image using a large dataset of geo-tagged images. Doersch et al. extract location-dependent features that capture the relative appearance differences of large cities. Lin et al. [73] localize a ground-level image by learning the relationship between pairs of ground and aerial images of the same location. Other techniques focus on urban environments and infer location using local image descriptors [101, 105]. Li et al. [70] exploit geo-registered 3D points clouds to estimate camera pose. Many other cues exist, such as the skyline [8, 93], sky appearance [67, 127], and shadows [59, 132].

Our work attempts to combine these two research directions. We use publicly available geospatial data to build a large geographic database containing geometric information for objects in the world, identify objects of interest in the query image, and use a probabilistic approach to estimate camera geo-calibration. To our knowledge, our approach is the first to apply a probabilistic particle-based MCMC algorithm for single image camera geo-calibration using a large geographic database.

2.2 Approach

Given a query image, captured at an unknown location in a known region of interest (ROI) and annotated with the location of geographic objects (e.g., buildings and roads), our goal is to estimate the intrinsic and extrinsic camera parameters. We base these estimates on a geographic information system (GIS) database that contains the location and extent of objects in the ROI. Assuming a simplified pinhole camera model with square pixels and

zero skew, the full camera geo-calibration problem is seven-dimensional: three position parameters, three orientation parameters, and the field of view. For this work, we assume that most photos are taken about five feet above the ground with little tilt or roll and reduce our model to four dimensions: $\Theta = (\textit{Latitude}, \textit{Longitude}, \textit{Azimuth}, \textit{FOV})^\top$. Adapting to higher or lower dimensional camera models is straightforward and may be useful depending on the available image annotations and GIS data. We find our proposed 4D state space to be good trade-off between the lack of descriptive power of lower-order camera models and higher-order camera models that require more computational time and memory resources. See Figure 2.2 for an example that compares our proposed model to a 2D model using only location.

Since one-to-one matching between image objects and GIS objects is likely not possible, we instead seek to estimate, $f(\Theta; \mathbf{C})$, the probability distribution function (PDF) over the camera parameters, Θ , given a set of constraints, \mathbf{C} . In the following section, we propose a function (an unnormalized density) that encodes constraints on the geo-calibration. This score function, $S(\Theta; \mathbf{C})$, encodes both prior knowledge about the camera and the geometric relationship between image annotations and the geographic database. In Section 2.2.2 we show how to estimate $f(\Theta; \mathbf{C})$ by sampling from this scoring function using an MCMC-based strategy.

2.2.1 Scoring Function

Proportional to the probability distribution of the camera parameters, $f(\Theta; \mathbf{C})$, the scoring function, $S(\Theta; \mathbf{C})$, is defined as a linear combination of multiple constraint functions:

$$S(\Theta; \mathbf{C}) = \sum w_i g(c_i, \Theta) \propto f(\Theta; \mathbf{C}), \quad (2.1)$$

where w_i is a weight and $g(c_i, \Theta)$ is a constraint function which is larger if the calibration, Θ , is more consistent with the image information, c_i . While the scoring function can take any type of constraint functions, we define a variety of functions in our implementation, including geometric configuration of multiple weak correspondences, constraints on geographic location, and priors over the field of view and camera orientation, to demonstrate the performance of our model. We focus on the first constraint specifically, the latter are simple Gaussian and uniform distributions.

Given the extent of an object in the image, and an estimated range of distances, (d_{min}, d_{max}) , from the camera to the object, we propose a constraint function that measures the geometric consistency between the object in the image and the GIS database. Given a hypothetical camera, Θ , we compute the distance, d , from the camera,

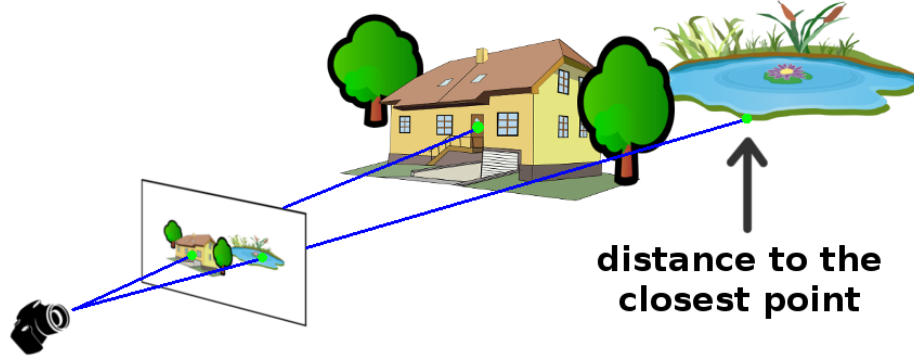


Figure 2.3: Since the exact correspondence between an image pixel and a point on a GIS object is unknown, we compute the distance to the closest point of the object to compare with the estimated distance.

through the object pixel, to the closest point of the object (see Figure 2.3), then the consistency function, $g(c_i, \Theta)$, takes form of a Gaussian distribution function, with mean $\mu = d - (d_{min} + d_{max})/2$, and standard deviation $\sigma = d_{max} - d_{min} + o$, where o is a constant offset ($o = 10m$ for all our experiments) to avoid dividing by zero. If no object of the correct type intersects the pixel ray, let $g(c_i, \Theta) = 0$.

2.2.2 Monte Carlo Markov Chain

The MCMC method is an efficient approach for sampling from high dimensional spaces. Its principal advantage over naive sampling strategies is that it visits high probability areas more frequently than the areas of low probability. We use the Metropolis-Hastings (MH) algorithm [17], a popular member of the MCMC family. The algorithm generates a set of possible cameras as follows: randomly sample a camera and compute its score; randomly sample (propose) a new nearby camera and compute its score; if the score of the new camera is higher, replace the old camera with the new camera, otherwise replace the old probabilistically; repeat this process many times, recording samples along the way. After a sufficiently large number of iterations, this process generates a set of possible cameras that are independent samples from the PDF, $f(\Theta; C)$, subject to some technical conditions. See Alg. 1 for details of the MH algorithm. We run multiple chains to mitigate issues with poor initial samples and local maxima.

Aside from the scoring function, $S(\Theta; C)$, the proposal distribution, $p(\Theta)$, which specifies how new cameras are sampled from a current camera (Alg. 1.4), is the most important choice when using an MCMC approach. We use the joint distribution of independent Gaus-

Require: \mathbf{C} (constraint set), and $maxIter$

- 1: Initialize camera parameters, Θ_0
- 2: $i \leftarrow 0$, $\mathbb{S} \leftarrow \emptyset$, $s_0 \leftarrow S(\Theta_0; \mathbf{C})$
- 3: **while** $i < maxIter$ **do**
- 4: sample new camera parameters: Θ_{i+1}
- 5: scoring: $s_{i+1} \leftarrow S(\Theta_{i+1}; \mathbf{C})$
- 6: $\mathbb{S} \leftarrow \mathbb{S} \cup \langle \Theta_{i+1}, s_{i+1} \rangle$
- 7: **if** $s_{i+1} < s_i$ **then**
- 8: $s_{i+1} \leftarrow s_i$, $\Theta_{i+1} \leftarrow \Theta_i$ with prob. $\frac{s_i - s_{i+1}}{s_i}$
- 9: **end if**
- 10: $i \leftarrow i + 1$
- 11: **end while**
- 12: **return** \mathbb{S}

Algorithm 1: Metropolis-Hastings Algorithm (MCMC)

sian random variables:

$$p(\Theta_{i+1}|\Theta_i) = \prod_{j=1}^{N_{dim}} \frac{1}{\sigma_j} \phi\left(\frac{\Theta_{i+1,j} - \Theta_{i,j}}{\sigma_j}\right),$$

where σ_j denotes the sampling step size on the j -th dimension, and $\phi(x)$ denotes the PDF of the standard normal distribution. As with all MH-based algorithms, the sampling performance is sensitive to the choice of step size. If the step size is too large, the algorithm converges slowly; if too small, the Markov chain may be trapped in a local maximum.

2.3 Evaluation

We compare our approach with two baseline methods, grid sampling and uniform random sampling, on real-world GIS data. The quantitative and qualitative results demonstrate the value of the proposed MH-based approach.

2.3.1 Methods

Dataset: We build a reference geographic database from OpenStreetMap¹ data, which contains the location and extent of many types of objects around the world. We only include roads, water, churches, residential buildings, and commercial buildings. For each query, we focus on a different $5km \times 5km$ ROI, in Kentucky, USA, each containing approximately 900 objects.

¹<http://www.openstreetmap.org>

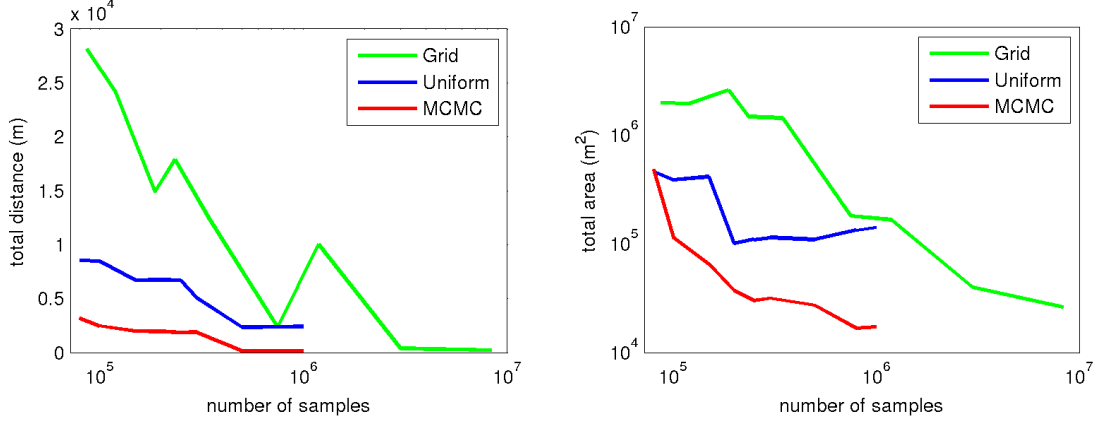


Figure 2.4: Performance comparison among methods. (left) Total distance. (right) Total area. For the grid method the finest sampling over spatial dimensions is 30m.

Metrics: We propose two evaluation metrics that simulate the process of manually verifying the camera location. For both, we generate a candidate list by sorting samples by their scores, then scan from the top until the ground-truth location is found. Given a set of samples from the state space and their corresponding scores, we greedily select the top- N candidates in terms of score, subject to the constraint that each of them is at least 200 meters away from the others spatially. Then, if the k -th candidate is the first candidate that is d ($d < 100$) meters away from the ground truth, we define its *total distance* to the ground truth as follows: $T_d = 200(k - 1) + d$. Intuitively, this metric enforces the diversity of candidate locations. The first term penalizes the ranking of the ground-truth candidate. The distance, d , distinguishes the accuracy of predictions in the case that two ground-truth candidate locations have the same ranking. By definition, a lower total distance to find the ground-truth candidate is better. The definition of the *total area*, T_a , is similar: we center a $l \times l$ patch around each candidate and compute the union of the areas from the top candidate patch to the patch that covers the ground-truth location. We set l to be 50m initially, if no patch covers the ground truth, we double the size of l and redo the computation until the ground-truth location is covered.

Implementation Details: We use the following fixed set of parameters, which we selected empirically, for all experiments. The ranges from which azimuth and FOV are sampled are $[0, 2\pi]$ and $[\pi/3, 2\pi/3]$ respectively. For the grid method, 50 angles are evenly sampled for azimuth, and 6 for the field of view. Thus at each geographic location, a total of 300 scoring samples are generated. For MCMC, 200 chains are independently processed in parallel. The step size is 100m for location, $\pi/20$ for azimuth, and $\pi/30$ for FOV.

2.3.2 Evaluation on Synthetic Data

We manually constructed twenty-five synthetic queries using our geographic database. We hand-picked a location for each camera, such that there exists nearby objects in the database and adjusted the camera azimuth and field of view such that objects were visible in the view frustum. We then projected the objects onto the image frame to obtain labeled pixels, and for each provided a min/max distance from the camera to them (consistent with the actual distance). To simulate the estimation error in real life, we set the distance range to be $1/20$ the length of the actual distance.

For each query, we confined the search area to a $5km \times 5km$ neighborhood that includes the ground-truth location. We computed the accuracy in terms of the total distance, T_d , and the total area, T_a , and the computation time in terms of the number of samples. The results of this experiment are shown in Figure 2.4. MCMC outperforms the other two baseline methods in both accuracy and speed. On average, our method only requires $1/10$ of computational time to converge to the same order of accuracy than the grid method.

2.3.3 Evaluation on Real Data

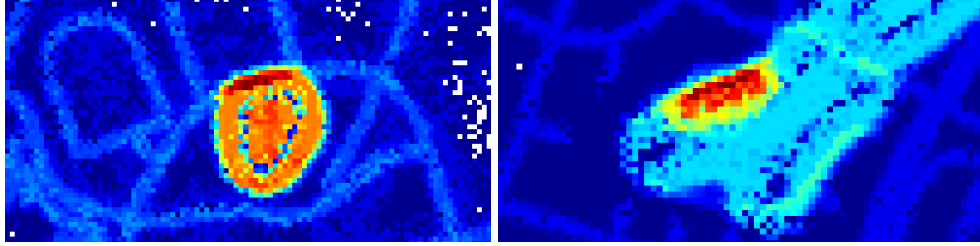
We evaluate our method using two real query images obtained from Google Street View. We hand-picked two locations, downloaded the corresponding equirectangular panoramas and extracted a perspective image from each. For each query, we labeled objects and estimated the min/max distance from the camera to the object in the world. Figure 2.5 shows the qualitative result of this experiment. Our method generates high scoring samples that are close to the ground truth. We also found that simple prior constraints can dramatically affect localization accuracy. By restricting the range of the azimuth and FOV to be within 5° of the ground truth, the distance from the top sample to the ground-truth location decreased from $12.5m$ to $0.6m$ and from $1.73m$ to $0.9m$, respectively, for two test cases.

2.4 Conclusions

We proposed an MCMC-based approach framework for single image camera geo-calibration which leverages a large geographic database. Our results demonstrate the superiority of our method versus several baseline methods, without requiring nearby ground-level imagery as is typical for most vision techniques. While we applied only a small set of primitive constraints in this work, the proposed framework is able to take any kind of constraints. In the following chapters, we will explore to develop more sophisticated constraint functions.



(a) Query images



(b) PDF of location



(c) Local geographic database and top-15 samples

Figure 2.5: Qualitative results for two query images (top). The resulting PDFs (middle) and map visualization of the top-15 samples (bottom) show that the proposed scoring function realistically captures the uncertainty.

Copyright© Menghua Zhai, 2018.

Chapter 3

A Constraint for Camera Pose

3.1 Introduction

In this chapter, we develop a constraint function for the camera pose. Since the camera roll and pitch angles can be derived from the horizon line position, our algorithm focuses on detecting the horizon line from the input image.

Automatic vanishing point (VP) and horizon line detection are two of the most fundamental problems in geometric computer vision [13, 81]. Knowledge of these quantities is the foundation for many higher level tasks, including image mensuration [21], facade detection [76], geo-localization [7, 127], and camera calibration [4, 40, 51, 64]. Recent work in this area [5, 123, 133] has explored novel problem formulations that significantly increase robustness to noise.

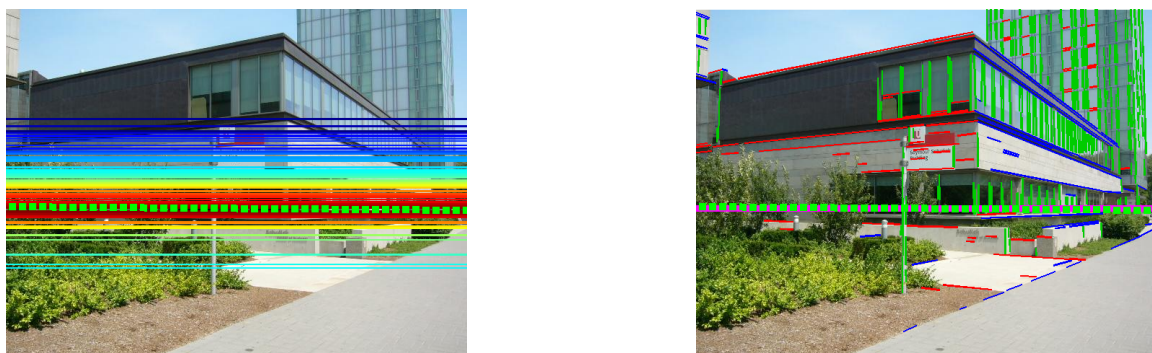


Figure 3.1: An example result of our method. (left) Horizon line candidates, colored by their scores (red means high score), and the true horizon line (green dash). (right) The horizon line (magenta) estimated by our algorithm is very close to the true horizon line (green dash). Line segments are color coded based on the most consistent detected vanishing point.

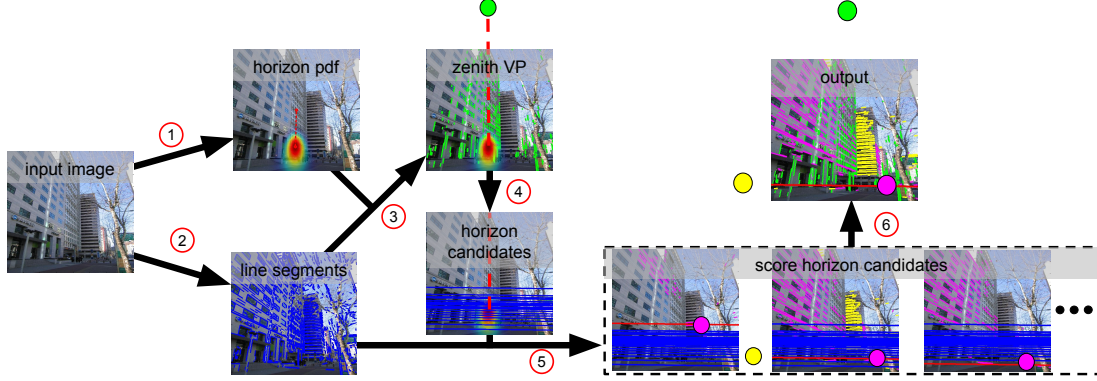


Figure 3.2: Algorithm overview: 1) use global image context to estimate a prior over horizon lines (Section 3.3); 2) extract line segments; 3) identify the zenith VP (Section 3.4.1); 4) sample horizon line candidates consistent with the zenith VP (Section 3.4.2); 5) find VPs on horizon line candidates (Section 3.4.2); and 6) select the best horizon line based on the VPs it contains (Section 3.4.3).

A vanishing point results from the intersection of projections of a set of parallel lines in the world. In man-made environments, such sets of lines are often caused by the edges of buildings, roads, and signs. VPs can typically be classified as either vertical, there is one such VP, and horizontal, there are often many such VPs. Given a set of horizontal VPs, there are numerous methods to estimate the horizon line. Therefore, previous approaches to this problem focus on first detecting the vanishing points, which is a challenging problem in many images due to line segment intersections that are not true VPs.

Our approach is to propose candidate horizon lines, score them, and keep the best (Figure 3.1). We use a deep convolutional neural network to extract global image context and guide the generation of a set of horizon line candidates. For each candidate, we identify vanishing points by solving a discrete-continuous optimization problem. The final score for each candidate line is based on the consistency of the lines in the image with the selected vanishing points.

This seemingly simple shift in approach leads to the need for novel algorithms and has excellent performance. We evaluated the proposed approach on two standard benchmark datasets, the Eurasian Cities Dataset [12] and the York Urban Dataset [24]. To our knowledge, our approach has the current best performance on both datasets. To evaluate our algorithm further, we also compare with the previous state-of-the-art method (Lezama et al. [69]) on a recently introduced dataset [131]; the results shows that our method is more accurate and much faster.

The main contributions of this work are: 1) a novel method for horizon line/vanishing point detection, which uses global image context to guide precise geometric analysis; 2)

a strategy for quickly extracting this context, in the form of constraints on possible horizon lines, using a deep convolutional neural network; 3) a discrete-continuous method for scoring horizon line candidates; and 4) an evaluation of the proposed approach on three benchmark datasets, which highlights that our method is both fast and accurate.

3.1.1 Related Work

Vanishing points and the horizon line provide a strong characterization of geometric scene structure and as such have been intensely studied for decades [13, 81]. For example, Hoiem et al. [44] show how the horizon line improves the accuracy of object detection. A wide variety of methods have been introduced to estimate these quantities. We provide a brief overview of the main approaches, refer to [110] for a comprehensive review.

Two distinct categories of methods exist, distinguished by the features they use. The first group of methods [12, 20, 24, 100] operate directly on lower-level features, such as edge pixels or image gradients. The second group of methods [3, 24, 69, 80, 113, 123, 133] build on top of the closely related problem of line segment detection. Our work is most closely related to the latter category, so we focus our discussion towards them.

The dominant approach to vanishing point detection from line segments is to cluster the line segments that pass through the same location. Various methods of clustering have been explored, including RANSAC [14], J-linkage [111], and the Hough transform [46]. Once the line segments have been clustered, vanishing points can be estimated using one of many refinement procedures [69, 100, 111, 123, 133].

These procedures typically minimize a nonlinear objective function. An important distinction between such methods is the choice of point and line representation and error metric. Collins and Weiss [18] formulate vanishing point detection as a statistical estimation problem on the Gaussian Sphere, which is similar to the geometry we use. More recent work has explored the use of dual space [69, 140] representations. Among the clustering-based approaches, Xu et al. [133] improve this pipeline by introducing a new point-line consistency function that models errors in the line segment extraction step.

Alternatives to clustering-based approaches have been explored. For example, vanishing point detection from line segments has been modeled as an Uncapacitated Facility Location (UFL) problem [5, 113]. To avoid error accumulation issues encountered by a step-by-step pipeline method, Barinova et al. [12] solve the problem in a unified framework, where edges, lines, and vanishing points fit into a single graphical model.

Our approach is motivated by the fact that properties of the scene, including objects, can provide additional cues for vanishing point and horizon line placement than line segments

alone. Unlike existing methods that use J-linkage [111, 133] or similar techniques to find an initial set of VPs by clustering detected lines followed by a refinement step, our approach first proposes candidate horizon lines using global image context.

3.1.2 Approach Overview

Our approach is motivated by two observations: 1) traditional purely geometric approaches to vanishing point detection often fail in seemingly nonsensical ways and 2) identifying the true vanishing points for many scenes is challenging and computationally expensive due to the large number of outlier line segments. Driven by these observations, we propose a two part strategy. First, we use global image context to estimate priors over the horizon line and the zenith vanishing point (Section 3.3). Using these priors, we introduce a novel VP detection method (Section 3.4) that samples horizon lines from the prior and performs a fast one-dimensional search for high-quality vanishing points in each. Both steps are essential for accurate results: the prior helps ensure a good initialization such that our horizon-first detection method may obtain very precise estimates that are necessary for many scene understanding tasks. See Figure 3.2 for an overview of our algorithm.

3.2 Problem Formulation

The goal of this work is to detect the horizon line, the zenith vanishing point, and any horizontal vanishing points from a single image. The remainder of this section defines the notation and basic geometric facts that we will use throughout. For clarity we use unbolded letters for points in world coordinates or the image plane and bolded letters for points or lines in homogeneous coordinates. We primarily follow the notation convention of Vedaldi and Zisserman [113].

Given a point (u, v) in the image plane, its homogeneous coordinate with respect to the calibrated image plane is denoted by:

$$\mathbf{p} = [\rho(u - c_u), \rho(v - c_v), 1]^T / \Sigma ,$$

where ρ is a scale constant, (c_u, c_v) is the camera principal point in the image frame, which we assume to be the center of the image, and Σ is the constant that makes \mathbf{p} a unit vector.

In homogeneous coordinates, both lines and points are represented as three-dimensional vectors (Figure 3.3). Computing the line, \mathbf{l} , that passes through two points, $(\mathbf{p}_1, \mathbf{p}_2)$, and the point, \mathbf{p} , at the intersection of two lines, $(\mathbf{l}_1, \mathbf{l}_2)$, are defined as follows:

$$\mathbf{l} = \frac{\mathbf{p}_1 \times \mathbf{p}_2}{\|\mathbf{p}_1 \times \mathbf{p}_2\|} \quad \mathbf{p} = \frac{\mathbf{l}_1 \times \mathbf{l}_2}{\|\mathbf{l}_1 \times \mathbf{l}_2\|} . \quad (3.1)$$

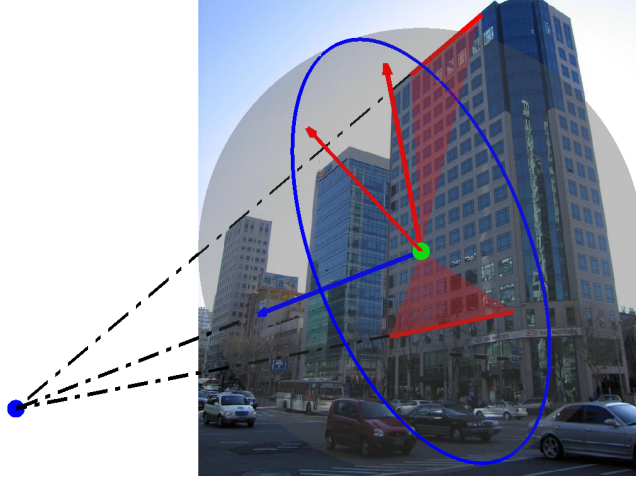


Figure 3.3: In homogeneous coordinates, lines (red lines) are defined by the normal (red arrow) of the plane (red triangle) they form with the origin (green dot). Two lines form a great circle (blue circle), whose normal (blue arrow) is their common point (blue dot) in homogeneous coordinates.

We denote the smallest angle between two vectors \mathbf{x} and \mathbf{y} with $\Theta_{\mathbf{x},\mathbf{y}} = |\cos^{-1}(\mathbf{x}^T \mathbf{y})|$. We use this to define the consistency between a line, \mathbf{l} , and a point, \mathbf{p} , as: $f_c(\mathbf{p}, \mathbf{l}) = \max(\theta_{con} - \Theta_{\mathbf{p},\mathbf{l}}, 0)$. The maximum value of consistency between a vanishing point and a line segment is θ_{con} . This will occur if it is possible to extend the line segment to contain the vanishing point.

3.3 Horizon Priors from Global Image Context

Recent studies show that deep convolutional neural networks (CNNs) are adaptable for a wide variety of tasks [135], and are quite fast in practice. We propose to use a CNN to extract global image context from a single image.

We parameterize the horizon line by its slope angle, $\alpha \in [-\pi, \pi)$, and offset, $o \in [0, \inf)$, which is the shortest distance between the horizon line and the principal point. In order to span the entire horizon line parameter space, we “squash” o from pixel coordinates to the interval $[0, \pi/2)$, through a one-to-one function, $w = \tan^{-1}(o/\kappa)$, in which κ is a scaling factor that affects how dense the sampling is near the center of the image.

3.3.1 Network Architecture

For our task, we adapt the popular AlexNet [65] architecture, which was designed for object recognition as part of the ImageNet ILSVRC-2012 challenge [96]. It consists of five



Figure 3.4: Example images from our training dataset (Section 3.3.2), each overlaid with the ground-truth horizon line.

convolutional layers, each followed by a non-linearity (rectified linear unit), and occasionally interspersed with pooling and local response normalization. This is followed by three fully connected layers (referred to as ‘fc6’, ‘fc7’, and ‘fc8’). A softmax is applied to the final output layer to produce a categorical distribution over 1000 object classes. We use this as a foundation to create a CNN that simultaneously generates a categorical distribution for each horizon-line parameter.

We modify the original AlexNet architecture in the following way: The first five convolutional layers are left unmodified. These layers are initialized with weights from a network trained for object detection and scene classification [141]. We remove the original fully connected layers (‘fc6’–‘fc8’) and add two disjoint sets of fully connected layers (‘fc6 α ’–‘fc8 α ’ and ‘fc6 w ’–‘fc8 w ’), one for each target label, α and w . We convert the slope, α , and the squashed offset, w , into independent categorical labels by uniformly dividing their respective domains into 500 bins. We randomly initialize the weights for these new layers.

We train our network using stochastic gradient descent, with a multinomial logistic loss function. The learning rates for the convolutional layers are progressively increased such that the latter layers change more. The new fully connected layers are given full learning rate.

3.3.2 Training Database

To support training our model of global image context, we construct a large dataset of images with known horizon lines. We make use of equirectangular panoramas downloaded from Google Street View in large metropolitan cities around the world. We identified a set of cities based on population and Street View coverage. From each city, we downloaded panoramas randomly sampled in a $5km \times 5km$ region around the city center. This resulted in 11 001 panoramas from 93 cities. Example cities include New York, Rio de Janeiro, London, and Melbourne.

We extracted 10 perspective images from each panorama with randomly sampled horizontal field-of-view (FOV), yaw, pitch, and roll. Here yaw is relative to the Google Street

View capture vehicle. We sampled horizontal FOV from a normal distribution with $\mu = 60^\circ$ and $\sigma = 10^\circ$. Similarly, pitch and roll are sampled from normal distributions with $\mu = 0^\circ$ and $\sigma = 10^\circ$ and $\sigma = 5^\circ$, respectively. Yaw is sampled uniformly. We truncate these distributions such that horizontal FOV $\in [40^\circ, 80^\circ]$, pitch $\in [-30^\circ, 30^\circ]$, and roll $\in [-20^\circ, 20^\circ]$. These settings were selected empirically to match the distribution of images captured by casual photographers in the wild.

Given the FOV, pitch, and roll of a generated perspective image, it is straightforward to compute the horizon line position in image space. In total, our training database contains 110 010 images with known horizon line. Figure 3.4 shows several example images from our dataset annotated with the ground-truth horizon line.

3.3.3 Making the Output Continuous

Given an image, I , the network outputs a categorical probability distribution for the slope, α , and squashed offset, w . We make these distributions continuous by approximating them with a Gaussian distribution. For each, we estimate the mean and variance from 5 000 samples generated from the categorical probability distribution. Since the relationship between w and o is one-to-one, this also results in a continuous distribution over o . The resulting distributions, $p(\alpha|I)$ and $p(o|I)$, are used in the next step of our approach to aid in detecting the zenith VP and as a prior for sampling candidate horizon lines. To visualize this distribution we observe that the horizon line can be uniquely defined by the point on the line closest to the principal point. Therefore, we can visualize a horizon line distribution as a distribution over points in the image. Figure 3.5 shows this distribution for two images.

3.4 Horizon-First Vanishing Point Detection

We propose an approach to obtain accurate estimates of the horizon line, the zenith vanishing point, and one or more horizontal vanishing points. Given an image, our approach makes use of the distributions estimated from global image context (Section 3.3) and line segments extracted with LSD [116]. The algorithm consists of the following major steps:

1. detect the zenith vanishing point (Section 3.4.1)
2. detect horizontal vanishing points on horizon line candidates (Section 3.4.2)
3. score horizon line candidates with horizontal vanishing points (Section 3.4.3)

The remainder of this section provides details for each of these steps.



Figure 3.5: Global image context imposes a strong prior on horizon line location. The output of our CNN is visualized as an overlaid heatmap, with red indicating more likely locations. For each image, the ground-truth horizon line (dash green) and the line that maximizes the prior (red) are shown.

3.4.1 Detecting the Zenith Vanishing Point

To detect the zenith vanishing point, we first select an initial set of line segments using the zenith direction, \mathbf{l}_z , from the global image context, then use the RANSAC [14] algorithm to refine it. The zenith direction is the line connecting the principal point and the zenith vanishing point, which is uniquely determined by the horizon line slope (see supplemental material for a proof).

We compute our initial estimate of \mathbf{l}_z using the global image context by choosing the value that maximizes the posterior: $\hat{\alpha} = \arg \max_{\alpha} p(\alpha|I)$. To handle the presence of outlier line segments, we first select a set of candidate vertical line segments as the RANSAC inputs by thresholding the angle between each line segment and the estimated zenith direction, $\Theta_{\mathbf{l}, \mathbf{l}_z} < \theta_{ver}$. For a randomly sampled pair of line segments with intersection, \mathbf{p} , we compute the set of inlier line segments, $\{\mathbf{l} \mid f_c(\mathbf{p}, \mathbf{l}) > 0\}$. If the largest set of inliers has a sufficient portion (more than 2% of candidate line segments), we obtain the final estimate of the zenith vanishing point, \mathbf{z} , by minimizing the algebraic distance, $\|\mathbf{l}^T \mathbf{p}\|$ using singular value decomposition (SVD), and update the zenith direction, \mathbf{l}_z . Otherwise, we keep the zenith direction estimated from the global image context.

3.4.2 Detecting Horizontal Vanishing Points

We start with sampling a set of horizon line candidates, $\{\mathbf{h}_i\}_1^S$, that are perpendicular to \mathbf{l}_z in the image space, under the distribution of horizon line offsets, $p(o|I)$. See Figure 3.6 for

examples of horizon line sampling with and without global context.

For each horizon line candidate, we identify a set of horizontal VPs by selecting points along the horizon line where many line segments intersect. We assume that for the true horizon line the identified horizontal VPs will be close to many intersection points and that these intersections will be more tightly clustered than for non-horizon lines. We use this intuition to define a scoring function for horizon line candidates.

As a preprocessing step, given the zenith direction, \mathbf{l}_z , and a horizon line candidate, \mathbf{h} , we filter out nearly vertical line segments ($\Theta_{\mathbf{l}, \mathbf{l}_z} < \theta_{ver}$), which are likely associated with the zenith vanishing point, and nearly horizontal line segments ($\Theta_{\mathbf{l}, \mathbf{h}} < \theta_{hor}$), which result in noisy horizon line intersection points. We remove such lines from consideration because they lead to spurious, or uninformative, vanishing points, which decreases accuracy.

Given a horizon line candidate, \mathbf{h} , and the filtered line segments in homogeneous coordinates, $\mathcal{L} = \{\mathbf{l}_i\}$, we select a set of horizontal VPs, $\mathcal{P} = \{\mathbf{p}_i\}$, by minimizing the following objective function:

$$g(\mathcal{P}|\mathbf{h}, \mathcal{L}) = - \sum_{\mathbf{p}_i \in \mathcal{P}} \sum_{\mathbf{l}_j \in \mathcal{L}} f_c(\mathbf{p}_i, \mathbf{l}_j) \quad (3.2)$$

subject to:

$$\Theta_{\mathbf{p}_i, \mathbf{p}_j} > \theta_{dist} \text{ and } \langle \mathbf{p}_i, \mathbf{h} \rangle = 0, \forall (i, j).$$

The constraint prevents two vanishing points from being too close together, which eliminates the possibility of selecting multiple vanishing points in the same location.

We propose the following combinatorial optimization process for obtaining an initial set of vanishing points, followed by a constrained nonlinear optimization to refine the vanishing points.

Initialization by Random Sampling and Discrete Optimization

To choose an initial set of candidate vanishing points, $\{\mathbf{p}_i\}_1^M$, we randomly select a subset of line segments, $\{\mathbf{l}_i\}_1^M$, and compute their intersection with the horizon line. We then construct a graph with a node for each vanishing point, \mathbf{p}_i , each with weight $\sum_{\mathbf{l}_j \in \mathcal{L}} f_c(\mathbf{p}_i, \mathbf{l}_j)$, which is larger if there are many line segments in the image that are consistent with \mathbf{p}_i . Pairs of nodes, (i, j) , are connected if the corresponding vanishing points, $\mathbf{p}_i, \mathbf{p}_j$, are sufficiently close in homogeneous space ($\Theta_{\mathbf{p}_i, \mathbf{p}_j} \leq \theta_{dist}$).

From this randomly sampled set, we select an optimal subset of VPs by maximizing the sum of weights, while ensuring no VPs in the final set are too close. Therefore, the problem of choosing the initial set of VPs reduces to a maximum weighted independent set problem, which is NP-hard in general. Due to the nature of the constraints, the resulting

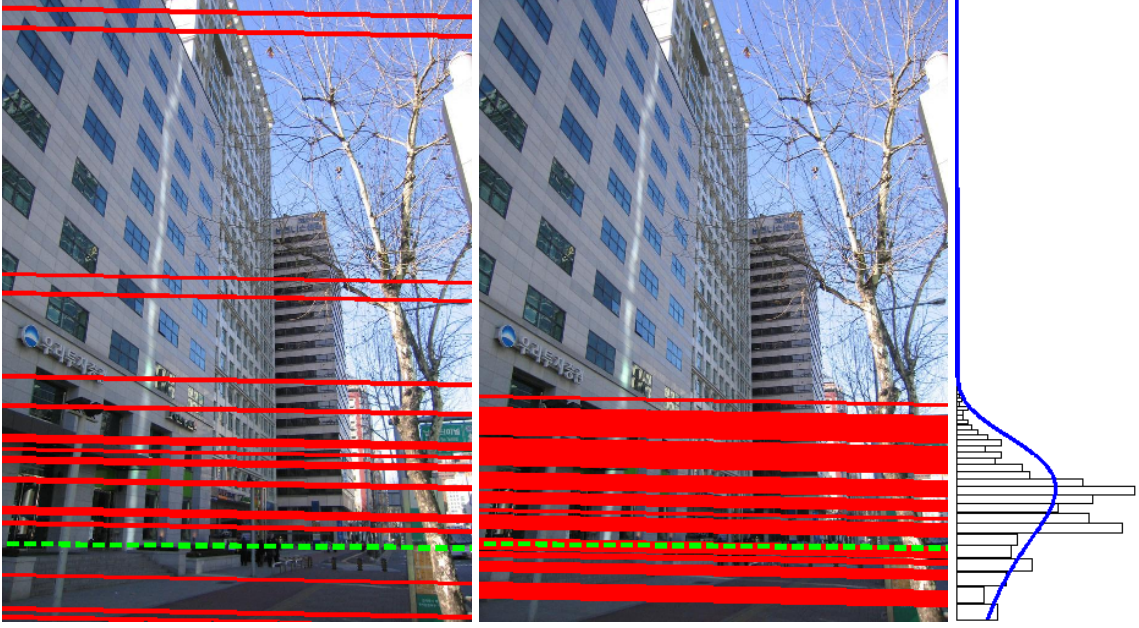


Figure 3.6: Our method samples more horizon line candidates (red) near the ground truth (green dash) with (middle) global image context than without (left). In the case of sampling with global image context, the offset PDF, $p(o|I)$ (blue curve), is fit from the CNN categorical probability distribution outputs (hollow bins). For clarity, we only show a reduced number of horizon line candidates and bins.

graph has a ring-like structure which means that, in practice, the problem can be quickly solved. Our solver exploits this sparse ring-like structure by finding a set of VPs that when removed convert the ring-like graph into a set of nearly linear sub-graphs (Figure 3.7). We solve each subproblem using dynamic programming. The set of VPs with maximum weight, $\{\mathbf{p}_i\}_{opt}$, is used as initialization for local refinement. Usually, 2–4 such vanishing points are found near the horizon line ground truth.

Vanishing Points Refinement

Since they were randomly sampled, the set of vanishing points selected during initialization, $\{\mathbf{p}_i\}_{opt}$, may not be at the optimal locations. We optimize their locations to further minimize the objective function (3.2). We perform an EM-like algorithm to refine the vanishing point locations, subject to the constraint that they lie on the horizon line:

- *E-step*: Given a vanishing point, \mathbf{p} , assign line segments that have positive consistency with \mathbf{p} : $\{\mathbf{l} | f_c(\mathbf{p}, \mathbf{l}) > 0\}$.
- *M-step*: Given the assigned line segments as a matrix, $\mathbf{l} = [\mathbf{l}_1, \mathbf{l}_2, \dots, \mathbf{l}_n]$, and the horizon line, \mathbf{h} , both represented in homogeneous coordinates, we solve for a refined vanishing point, \mathbf{p}^* , by minimizing the algebraic distance, $\|\mathbf{l}^\top \mathbf{p}\|$ such that $\mathbf{h}^\top \mathbf{p} =$

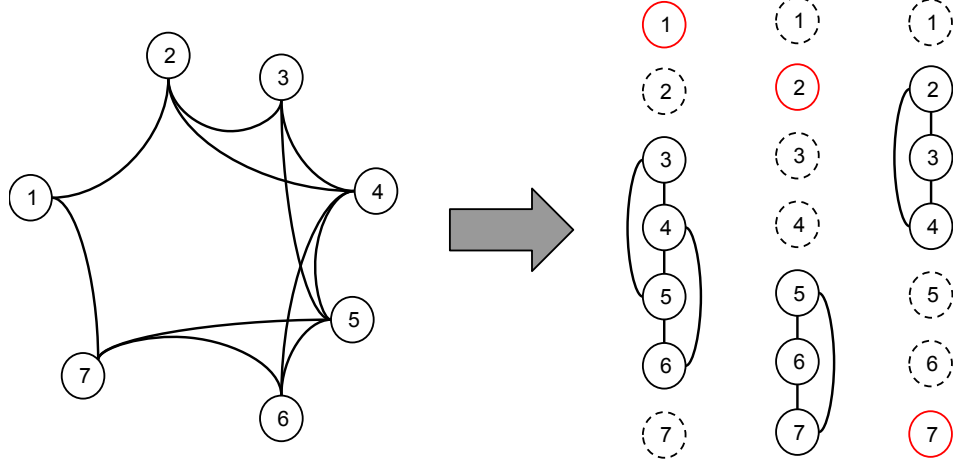


Figure 3.7: A ring-like graph (left) is converted into three nearly linear subgraphs (right) by partitioning around a node with minimal degree. For the subgraphs, the red node is mandatory, the dashed nodes are excluded, and a subset of the solid nodes are selected using dynamic programming.

0. We define a basis, \mathbf{B}_h , for the null space of \mathbf{h} , and reformulate the problem as $\lambda^* = \arg \min \|\mathbf{I}^\top \mathbf{B}_h \lambda\|$, which we solve using SVD. Given the optimal coefficients, λ^* , we reconstruct the optimal vanishing point as: $\mathbf{p}^* = \frac{\mathbf{B}_h \lambda^*}{\|\mathbf{B}_h \lambda^*\|}$.

We run this refinement iteration until convergence. In practice, this converges quickly; we run at most three iterations for all the experiments. The final set of optimized VPs is then used to assign a score to the current horizon line candidate.

3.4.3 Optimal Horizon Line Selection

For each horizon line candidate, we assign a score based on the total consistency of lines in the image with the VPs selected in the previous section. The score of a horizon line candidate, \mathbf{h} , is defined as:

$$score(\mathbf{h}) = \sum_{\{\tilde{\mathbf{p}}_i\}} \sum_{\mathbf{l}_j \in \mathcal{L}} f_c(\tilde{\mathbf{p}}_i, \mathbf{l}_j) . \quad (3.3)$$

To reduce the impact of false positive vanishing points, we select from $\{\mathbf{p}_i\}_{opt}$ the two highest weighted vanishing points (or one if $\{\mathbf{p}_i\}_{opt}$ contains only one element), $\{\tilde{\mathbf{p}}_i\}$, for horizon line scoring.

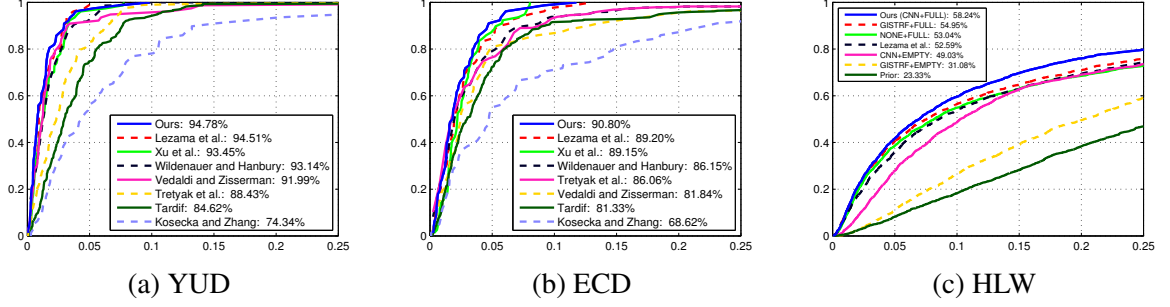


Figure 3.8: For three benchmark datasets, the fraction of images (y-axis) with a horizon error less than a threshold (x-axis). The AUC for each curve is shown in the legend. For additional details see Section 3.5.

3.5 Evaluation

We perform an extensive evaluation of our methods, both quantitatively and qualitatively, on three benchmark datasets. The results show that our method achieves state-of-the-art performance based on horizon-line detection error, the standard criteria in recent work on VP detection [12, 69, 113, 133]. Horizon detection error is defined as the maximum distance from the detected horizon line to the ground-truth horizon line, normalized by the image height. Following tradition, we show the cumulative histogram of these errors and report the area under the curve (AUC).

Our method is implemented using MATLAB, with the exception of detecting line segments, which uses an existing C++ library [116], and extracting global image context, which we implemented using Caffe [58]. We use the parameters defined in Table 3.1 for all experiments. This differs from other methods which usually use different parameters for different datasets.

Table 3.1: Algorithm parameters (given an $H \times W$ image).

Name	Usage(s)	Value
θ_{con}	Section 3.2	2°
ρ	Section 3.2	$2/\max(H, W)$
κ	Section 3.3	$1/5 \times H$
θ_{ver}	Section 3.4.1, Section 3.4.2	$\Theta_{l,z} < 10^\circ$
θ_{hor}	Section 3.4.2	$\Theta_{l,h} < 1.5^\circ$
S	Section 3.4.2	300 candidates
M	Section 3.4.2	20 line segments
θ_{dist}	Section 3.4.2, Section 3.4.2	$\Theta_{p_i, p_j} > 33^\circ$

3.5.1 Quantitative Evaluation

The York Urban Dataset (YUD) [24] is a commonly used dataset for evaluating horizon line estimation methods. It contains 102 images and ground-truth vanishing points. The scenes obey the Manhattan-world assumption, however we do not take advantage of this assumption. Figure 3.8a shows the performance of our methods relative to previous work on YUD. These results demonstrate that our method achieves state-of-the-art AUC, improving upon the previous best of Lezama et al. [69] by 0.28%, a relative improvement¹ of 5%. This is especially impressive given that our method only requires an average of 1 second per image, while Lezama et al. requires approximately 30 seconds per image.

The Eurasian Cities Dataset (ECD) [12] is another commonly used benchmark dataset, which is considered challenging due to the large number of outlier line segments and complex scene geometries. It contains 103 images captured in urban areas and, unlike the YUD dataset, not all images satisfy the Manhattan-world assumption. It provides reliable horizon line ground truth and is widely considered difficult for horizon line detection. To our knowledge, the previous state-of-the-art performance in terms of the AUC metric on this dataset was achieved by Lezama et al. [69]. Our algorithm improves upon their performance, increasing the state of the art to 90.8%. This is a significant relative improvement of 14.8%, especially considering their improvement relative to the state of the art was 0.5%. On ECD, our method takes an average of 3 seconds per image, while Lezama et al. requires approximately 60 seconds per image. We present the performance comparison with other methods in Figure 3.8b.

The Horizon Lines in the Wild (HLW) dataset [131] is a new, very challenging benchmark dataset. We use the provided test set, which contains approximately 2 000 images from diverse locations, with many images not adhering to the Manhattan-world assumption. Figure 3.8c compares our method with the method of Lezama et al. [69] (the only publicly available implementation from a recent method). Our method is significantly better, achieving 58.24% versus 52.59% AUC.

3.5.2 Component Error Analysis

Our method consists of two major components: global context extraction (Section 3.3) and horizon-first vanishing point detection (Section 3.4). This section provides an analysis of the impact each component has on accuracy.

To evaluate the impact of global context extraction, we considered three alternatives: our proposed approach (CNN), replacing the CNN with a random forest (using the Python

¹We define the relative improvement as $\frac{AUC_{new} - AUC_{old}}{1 - AUC_{old}}$.

Table 3.2: Component error analysis (AUC).

Method	YUD	ECD	HLW
Lezama et al. [69]	94.51%	89.20%	52.59%
NONE+FULL	93.87%	87.94%	53.04%
GISTRf+EMPTY	53.36%	32.69%	31.08%
GISTRf+FULL	94.66%	87.60%	54.95%
CNN+EMPTY	73.67%	67.64%	49.03%
CNN+FULL (Ours)	94.78%	90.80%	58.24%

“sklearn” library with 25 trees) applied to a GIST [87] descriptor (GISTRf), and omitting context entirely (NONE). When omitting the global context, we assume no camera roll (horizon lines are horizontal in the image) and sample horizon lines uniformly between $[-2H, 2H]$ (H is the image height). To evaluate the impact of vanishing point detection, we considered two alternatives: our proposed approach (FULL) and omitting the vanishing point detection step (EMPTY). When omitting vanishing point detection, we directly estimate the horizon line, (α, o) , by maximizing the posterior estimated by our global-context CNN, $p(\alpha, o|I)$.

Quantitative results presented in Table 3.2 show that both components play important roles in the algorithm and that CNN provides better global context information than GISTRf. Though our vanishing point detection performs well by itself (see column NONE+FULL), global image context helps improve the accuracy further. Figure 3.8c visualizes these results as a cumulative histogram of horizon error on HLW. To illustrate the impact of global image context, we present two examples in Figure 3.9 that compare horizon line estimates obtained using global context (CNN+FULL) and without (NONE+FULL). When using global context, the estimated horizon lines are very close to the ground truth. Without, the estimates obtained are implausible, even resulting in an estimate that is off the image.

3.5.3 Failure Cases

We highlight two representative failure cases in the last column of Figure 3.10. The top image fails due to the propagation of measurement errors from the short line segments. The bottom image is challenging because the curved structures lead to indistinct VPs. Despite this, global context helps our method produce plausible results, while other methods (*e.g.*, [12]) fail dramatically.

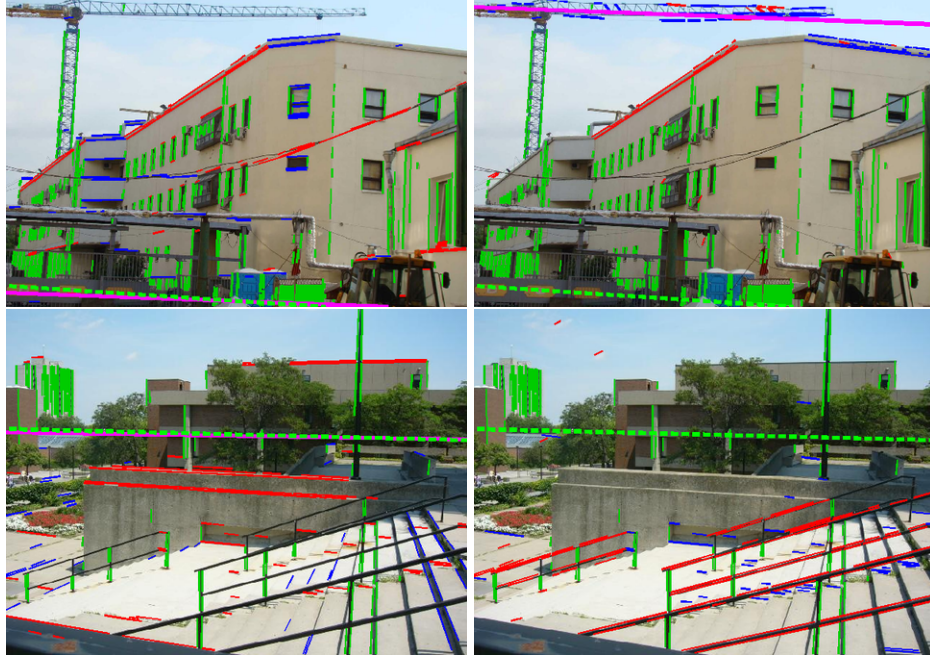


Figure 3.9: Two images where horizon line estimates are much better with global context (left) than without (right).

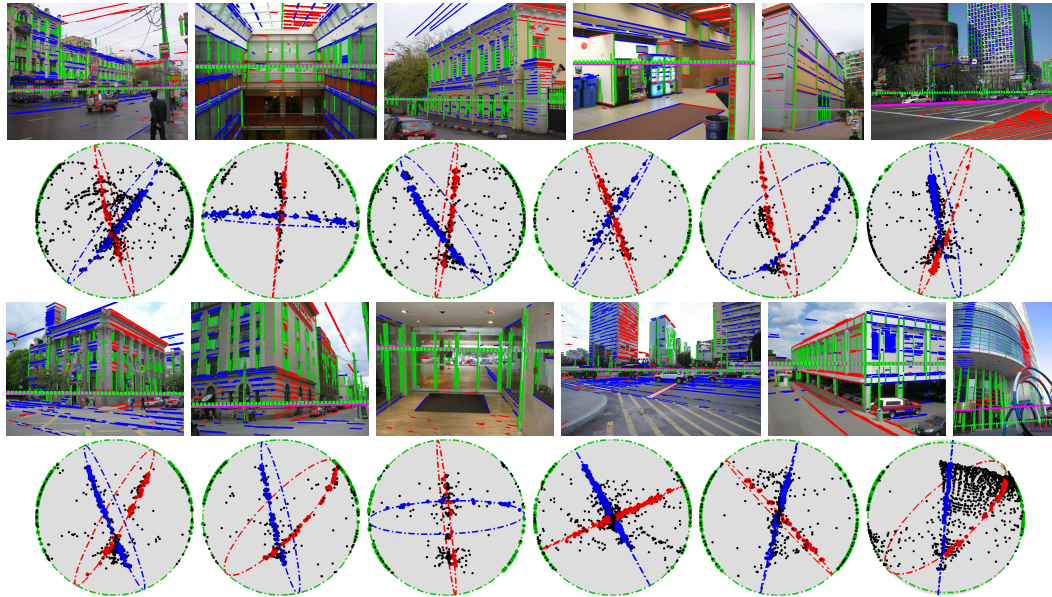


Figure 3.10: Example results produced by our method. (rows 1 and 3) Line segments color coded based on the most consistent VP, the ground-truth (green dash), and detected horizon lines (magenta). For clarity only the top two horizontal VPs are shown. (rows 2 and 4) The line segments (dots) and their VPs (rings) represented in homogeneous coordinates. (last column) Two failure cases of our method, caused by irregularly shaped objects (bottom) and short edges (top).

3.6 Conclusion

We presented a novel vanishing point detection algorithm that obtains state-of-the-art performance on three benchmark datasets. The main innovation in our method is the use of global image context to sample possible horizon lines, followed by a novel discrete-continuous procedure to score each horizon line by choosing the optimal vanishing points for the line. Our method is both more accurate and more efficient than the previous state-of-the-art algorithm, requiring no parameter tuning for a new testing dataset, which is common in other methods.

Chapter 4

A Constraint for Time and Location

4.1 Introduction

In this chapter, we focus on developing the constraint function for the capture time and the camera location for outdoor images. Outdoor images often contain sufficient visual information to understand geographic information about the scene, such as where the image was captured. Developing effective algorithms for this task has received significant attention for many years [42, 122]. The appearance of an outdoor scene can also change rapidly. These changes are often due to fleeting, or transient, attributes such as lighting and weather conditions, that dramatically affect the visual perception of an environment. For instance, consider a scene that changes from sunny and pleasant to rainy and brooding in mere minutes. Several methods have been proposed for automatically understanding and extracting these subtle characteristics from imagery [92, 78, 66, 10]. Estimating these types of transient attributes has importance in a number of fields, including: environmental monitoring [118, 29], as a pre-processing step for calibration [51, 127], and enabling semantic browsing of large photo collections [53, 66]. Our work fuses these two research areas by learning to estimate geo-temporal image features, which are related to when and where an image was captured.

Recently, a significant amount of work has explored how sources of supervision beyond manual annotation can be used to learn useful representations of images. In general, collecting manual annotations for millions, or perhaps billions, of images is prohibitively expensive. As Doersch summarizes [25], “The idea is that, given the right task, the computer can learn on its own to represent useful semantic properties of the visual world.” Such learning tasks are often referred to as *pretext tasks*; they serve as an intermediary target for learning the intended representation. For example, Doersch et al [26] show how spatial context can be used as a supervisory signal in order to learn a visual representation for

object discovery. Similarly, Pathak et al. [91] use context-based pixel prediction for pre-training a representation for classification, detection, and segmentation tasks. We extend this line of work by using time and location context to learn useful features from a large corpus of imagery.

Our work makes the following visual assumptions about the world. First, that photographs provide a direct source of context regarding the conditions under which they were captured. For example, the time of day that an image is captured is directly related to the brightness of the image (i.e., light to dark), season can indicate the expected weather conditions or how people are dressed, and location can provide evidence about anticipated styles such as architecture. Second, these context signals are hard to extract from an image, are potentially noisy (e.g., snow in April), and can be indicated by multiple sources (snow on the ground, people wearing heavy coats). These assumptions motivate our method which integrates image appearance, time, and location, the latter of which are typically recorded automatically by the imaging device.

In our approach, we explicitly model the relationship between the image, its geographic location, and the time of capture. We propose a novel convolutional neural network architecture that implicitly learns how to extract geo-temporal features from the imagery by optimizing for a set of location and time estimation tasks. Specifically, we structure our network to jointly learn feature representations for three related spaces: images, time, and location. To accomplish this, each representation, or combination of representations, is used to predict held out information. For example, the image representation and location representation (or the combination of both) are used to learn to predict when an image was captured. In total, three representations are learned using four classification tasks. We optimize all representations and tasks simultaneously, in an end-to-end fashion.

The main contributions of this work are: 1) a novel approach for learning geo-temporal image features from a large corpus of imagery without requiring image-level manual annotations; 2) an evaluation of the learned features on the task of transient attribute estimation, where our features outperform those from a network pre-trained using the strongly supervised ImageNet dataset [95]; 3) an evaluation of the accuracy of our learned estimators, highlighting the value of additional context; and 4) a novel location estimation method that uses the task of time estimation to localize a static camera.

4.2 Related Work

Image localization, or estimating where an image was captured, is an important problem in the vision community. Typically, the problem is formulated as image retrieval using a

reference database of ground-level images [42] or overhead images [74, 126, 129] with known location. Other methods have been proposed which take advantage of photometric and geometric properties such as sun position [67, 127], and many other cues. More recently, Weyand et al. [122] proposed to directly predict the geographic location of a single image using a deep convolutional neural network by classifying the query image into a set of spatial bins. This style approach is a special case of our proposed network.

Other work has explored how to estimate the time that an image was captured. Salem et al. [99] demonstrate that human appearance, including clothing and hairstyle, is a useful cue for dating images. Matzen and Snavely [83] predict timestamps for photos by matching against a time-varying reconstruction of a scene. Volokitin et al. [115] use representations extracted from CNNs to estimate ambient temperature and time of year for outdoor images. In our work we explore how location and time estimation can be used as tasks for learning geo-temporal features.

Attribute-based representations have become popular in outdoor scene understanding to help describe how the appearance of a scene changes over time. Laffont et al. [66] introduced a taxonomy of 40 transient attributes that describe intra-scene variations along with methods for identifying the presence of such attributes in an image. Using this dataset, Baltenberger et al. [10] introduce methods for estimating the presence of transient attributes using convolutional neural networks. Jacobs et al. [53] demonstrate that principle component analysis, when applied to webcam imagery, results in a decomposition that is closely related to natural changes in the scene, including the time of day, local weather conditions, and human activity. Similarly, a body of work has sought understand local weather conditions [50, 78]. Many studies have shown that these types of transient attributes can be useful for image and camera localization tasks [55, 10].

Recent work has explored the use of self-supervision (sometimes referred to as unsupervised learning or pretext tasks) for training deep neural networks that capture useful visual representations [26, 91]. For example, Zhang et al. [139] show how image colorization (synthesizing colors for a grayscale image) is a powerful pretext task for learning visual representations. Pathak et al. [90] exploit low-level motion-based grouping cues for unsupervised feature learning. These methods typically exploit some known quantity of the data (e.g., pixel color values) to avoid expensive manual annotation. As a byproduct, a useful visual representation is learned. In our work, we consider two novel pretext tasks, time and location estimation. Our work is most similar to Li and Snavely [72], who propose an approach for learning intrinsic images by observing image sequences with varying illumination over time. While their targets are reflectance and shading images of the scene, we focus on location and time.

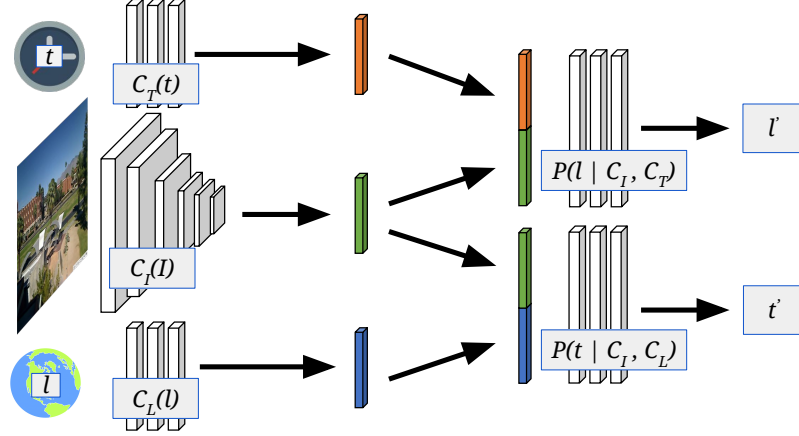


Figure 4.1: An overview of the proposed network architecture. Our approach learns mid-level feature representations for time (orange), location (blue), and image appearance (green) by optimizing for a set of conditional time and location estimation tasks.

4.3 Estimating Geo-Temporal Image Features

We propose a neural network architecture for learning geo-temporal features from images by optimizing for a set of location and time estimation tasks. An overview of the proposed architecture is shown in Figure 4.1. Our network takes three inputs: an image, I , the time the image was captured, t , and the location of capture, l . Each input is independently processed by a *context network* to extract mid-level features. Then, pairs of these features are used by *estimator networks* to predict distributions over time or location.

4.3.1 Context Networks

We use three *context networks*: a temporal context network, $C_T(t)$; a location context network, $C_L(l)$; and an image context network, $C_I(I)$. The output of each context network is a 128 dimensional feature with a sigmoid activation function. For the temporal context network, we parameterize the input timestamp using a one-hot encoding of month and hour of day, for a total of 12×24 dimensions. This encoding is flattened and passed to $C_T(t)$, which consists of three fully-connected layers (with 256, 512, and 128 channels respectively), the first two with ReLU activations. For the location context network, we parameterize the geographic location, l , using standard 3D ECEF coordinates, which we normalize by the Earth’s radius. Other than a different input, the location context network is identical to the temporal context network. For the image context network, we use the *InceptionV2* architecture [109], up to the last global average pooling layer, to extract features. We flatten the output feature map and append the same structure as the other context

networks.

4.3.2 Estimator Networks

The output of the context networks are used as input to four different estimator networks:

- *Location Estimator*, $P(l|C_I(I))$, which predicts location using only image features;
- *Time Estimator*, $P(t|C_I(I))$, which predicts time using only image features;
- *Time-conditioned Location Estimator*, $P(l|(C_I(I), C_T(t)))$, which predicts location using features from the image and the known capture time;
- *Location-conditioned Time Estimator*, $P(t|C_I(I), C_L(l))$, which predicts time features from the image and the known geographic location.

Aside from different output size and activations, the estimator networks have the same structure as the context networks. We discretize the output space for location and time, representing the probability as a categorical distribution (softmax activation). For location, we use 37×72 equal-angle lat./long. bins. For time, we use 12×24 “month \times hour” bins.

4.3.3 Implementation Details

We randomly initialize the *InceptionV2* network using the standard strategy [109]. We initialize all other network weights randomly using Xavier initializer [36] and simultaneously optimize them during training. For each estimator network, we have a cross entropy loss. We minimize the sum of these using the *Adam* optimizer [62] ($\beta_1 = 0.9$ and $\beta_2 = 0.999$). We use a learning rate policy that starts from 0.001 and decreases by half every $50k$ iterations. For regularization, we apply weight decay with rate of 0.0001. We train the proposed network for $2.5M$ iterations with batch size 32. We apply batch normalization [48] on every layer except the last (for both context and estimator networks). The input images are scaled to $[-1, 1]$ and augmented by a random crop to the size of 224×224 . We use Greenwich Mean Time (GMT) for all timestamps.

4.4 Experiments

We evaluate the context networks and estimator networks on various datasets, visualize specific features in the image context networks, and show that the image context features have strong correlations with transient image attributes.

4.4.1 Training and Evaluation Datasets

We use four main datasets to evaluate our approach. The *AMOS* dataset refers to a subset of the AMOS database [53], which is a collection of over a billion images captured from public outdoor webcams around the world. For our experiments, we use a subset of images: only from cameras with high-accuracy geo-location and images captured between 2002 to 2017. This resulted in images from 12,193 webcams from which we held-out 231 for testing. Each image has a timestamp recorded by the image collection process. The *YFCC* dataset refers to a subset of the Yahoo 100 million dataset [112], only including geotagged images from smart phones. We restricted the dataset to smart phone images since we found that non-phone images often had inaccurate timestamps. We filter out indoor images using the *Places* network [143]. This results in a training set of 892,662 images and a test set of 170,994 images. The *Hybrid* dataset refers to a combination of the *AMOS* and *YFCC* training sets (equally sampling for each mini-batch). The *TA* dataset refers to the Transient Attributes Dataset [66], which contains 8,571 images, each manually annotated with 40 transient attributes, such as sunny and cloudy.

4.4.2 Understanding the Image Context Representation

We conducted several experiments to relate image appearance to the representation learned by the image context network. To begin, we examined images that correspond with extremal activations. For this experiment, we used 10,000 images randomly sampled from the *YFCC* dataset and 7,732 images covering the year of 2015 from one camera (ID: 4308) in *AMOS* dataset. For each neuron of the image context representation, we selected the 10 images that result in the highest activation from the two different sets of images. Figure 4.2 shows a montage of images for three neurons. The neurons appear to capture semantically meaningful attributes, such as daylight, rainy, and winter. Similarly, we selected two neurons and visualized their signal over time using images from a static webcam. Figure 4.3 shows how scene appearance changes are related to the image context features. For the example shown, it appears that these neurons are related to daylight and fogginess. These experiments provide evidence that the mid-level representation captured by the image context network are related to static and transient scene attributes.

4.4.3 Analyzing Feature Correlation with Transient Attributes

To analyze quantitatively how much our model learns about transient attributes, we compute the cross correlations between a mid-level representation of the image context network

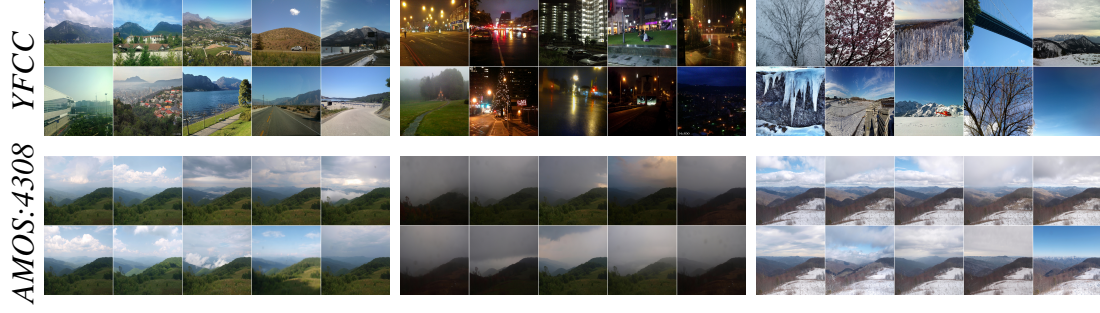


Figure 4.2: Relating image appearance to the image context representation by visualizing the images that have the highest activation at three different neurons.

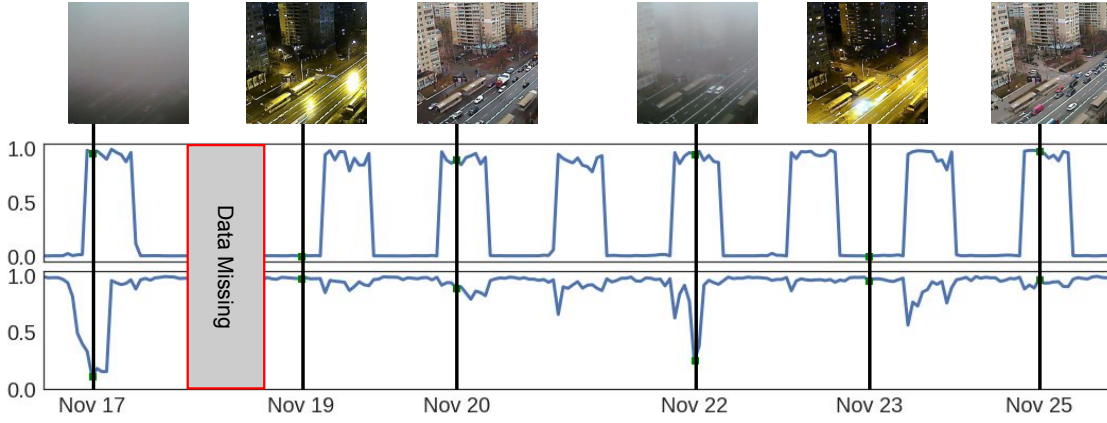


Figure 4.3: The time series of two neurons for a week of webcam imagery, with images showing the scene at various points. It appears that the top neuron is related to the diurnal cycle and the bottom is related to fogginess.

and the corresponding transient attribute labels of all test images in the *TA* dataset. As a baseline, we compare to features of the same architecture trained for ImageNet [48] classification and features sampled uniformly at random. We select the feature from the last pooling layer (*AvgPool_1a_7x7*), which is the deepest layer that this model and ours share in common. We compute the cross correlation scores between the feature and the transient attribute scores of each image, resulting in a 1024×40 cross correlation matrix, M , where the element m_{ij} is the cross correlation score between the i -th feature channel and the j -th transient attribute. Figure 4.4 shows, for each transient attribute, the maximum absolute correlation score over all feature channels. We observe that our proposed method learns features that are more correlated to the transient attributes ($\bar{\rho} = 0.414$) than the ImageNet network ($\bar{\rho} = 0.281$).

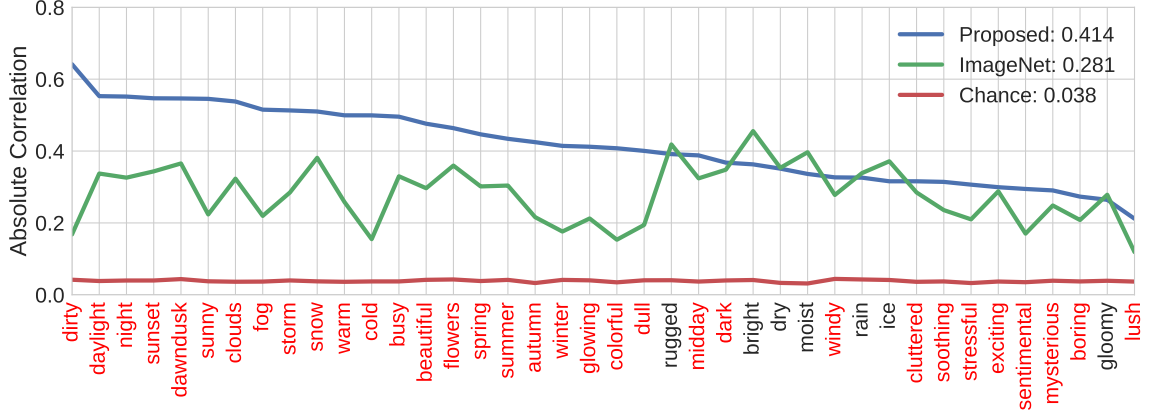


Figure 4.4: Cross-correlation scores between transient attributes and three representations. For each attribute, we show the peak correlation score. For a majority of attributes our proposed representation is more highly correlated than the ImageNet baseline.

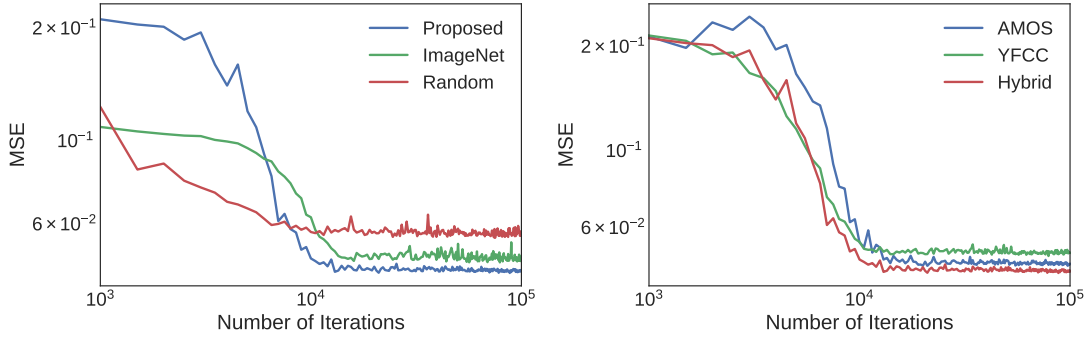


Figure 4.5: Comparing mid-level features for transient attribute estimation. (left) Initializing from the weights of our proposed approach versus a network trained for image classification and a random baseline. (right) Initializing from our method, trained on different datasets.

4.4.4 Comparing Mid-Level Features for Transient Attribute Estimation

The previous experiment showed that the image context network is capturing mid-level features correlated with transient attributes. In this section, we explore the ability of this representation for directly estimating transient attributes. Similar to the previous experiment, we truncate our model at the last pooling layer (in order to compare versus alternative initialization strategies), and add a final two-layer MLP with 40 outputs corresponding to the 40 transient attributes in the *TA* dataset. We train this network, initializing from the weights of models trained for different tasks, including variants of our method trained on the *AMOS*, *YFCC*, and *Hybrid* datasets. During training, the MLP portions are randomly initialized while the earlier layers are frozen. We evaluate the average mean squared error

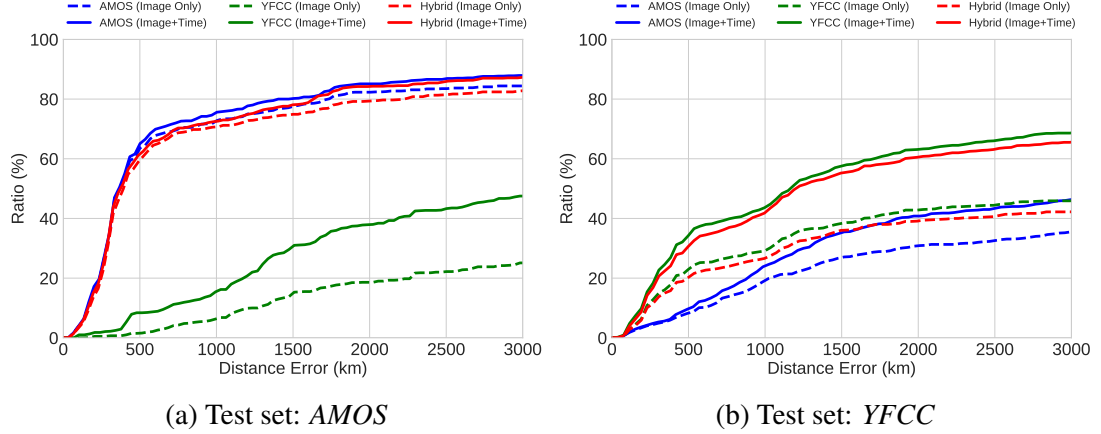


Figure 4.6: Quantitative evaluation of localization performance shown as a cumulative distance error plot for the *YFCC* dataset.

(MSE) for the test set every 500 iterations (batch size 32). Figure 4.5 shows the performance comparison among different mid-level features, including ImageNet and randomly initialized *InceptionV2*. Our features are superior to all baselines and perform best when learned using the *Hybrid* dataset.

4.4.5 Application: Image Localization

There are two image localization formulations that our network architecture enables. The straightforward approach is to use the location estimator (or the time-conditioned variant) to generate a probability distribution over a discrete set of location bins. An alternative approach is to optimize for a continuous location estimate by minimizing the loss of the location-conditioned time estimator network.

Discrete Localization Given an input image, I , we evaluate the location estimator $P(l|C_I(I))$ and the time-conditioned location estimator $P(l|C_I(I), C_T(t))$, which requires a timestamp, t . We trained our model on each dataset and perform quantitative evaluation using the test images from *AMOS* and *YFCC*, separately. We use the latitude/longitude center of the maximal bin as our location estimate. The results of this experiment are presented in Figure 4.6. We observe that the time-conditioned location estimator is superior in both cases. We also conclude that our model performs better if trained on the same imagery source with the test set, and training the network with the *Hybrid* dataset is competitive on both test sets.

Continuous Localization In this formulation, we use the location-conditioned time estimator, $P(t|C_I(I), C_L(l))$, to optimize for a continuous location estimate. Given the known image capture time t^* , the idea is that the true location should result in a low value for

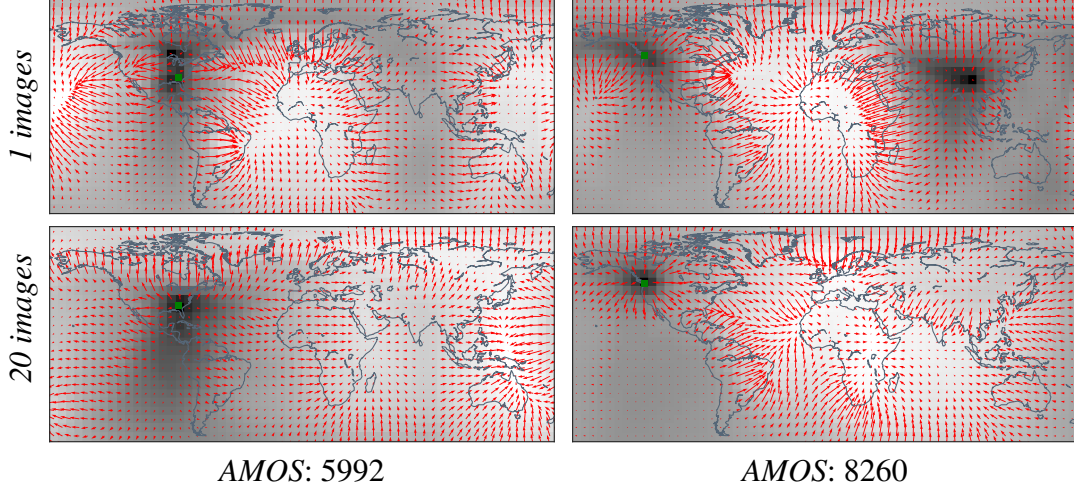


Figure 4.7: Visualizing the time estimation loss for two cameras and varying number of images (darker is lower). The red arrows show the gradient and the green dot is the true location.

the loss associated with the estimator, $\ell_t = \phi(P(t|C_I(I), C_L(l)), t^*)$, where $\phi(\circ, \circ)$ is the cross-entropy loss. Therefore, we can produce a location estimate by optimizing the location, l , with respect to ℓ_t . Unfortunately, an individual image does not typically yield a unique, or accurate, location estimate using this method. However, if we sum the loss across images captured at different times, we find that the minima of the function becomes more distinct. Figure 4.7 shows several qualitative examples of this localization strategy on static webcams, where darker colors correspond to more likely locations. We can see that as additional images are included in the loss, the uncertainty of the location prediction diminishes.

4.4.6 Application: Time Estimation

Using the time estimator and location-conditioned time estimator, our network is able to estimate the capture time of a query image. These estimators output a distribution in discrete 2D time space. To evaluate our estimates, we compare the ground-truth capture time and the marginal probabilities of our predictions on the *YFCC* test set, and present the cumulative error plot in Figure 4.8. We observe that including location is not useful for pinpointing the month. We suspect this is because most of our imagery is in the northern hemisphere, and changing the location within a hemisphere doesn't change the season. However, this is not the case when estimating the hour. To visualize this, we show in Figure 4.9 the impact of changing the location on the hour estimate. We compute the marginal hour distribution at different latitudes and longitudes. When performing a sweep over latitude, we fix

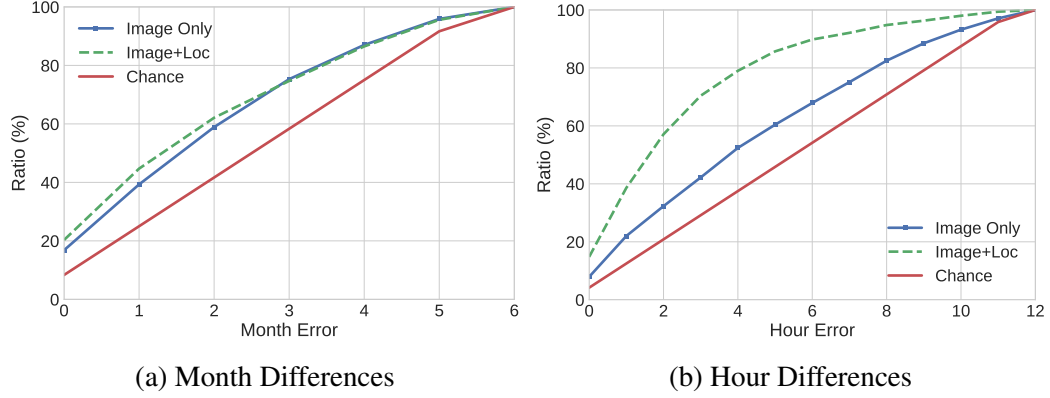


Figure 4.8: Quantitative evaluation of time estimation (month and hour) performance shown as a cumulative error plot for the *YFCC* dataset. Both methods performance better than the random chance.

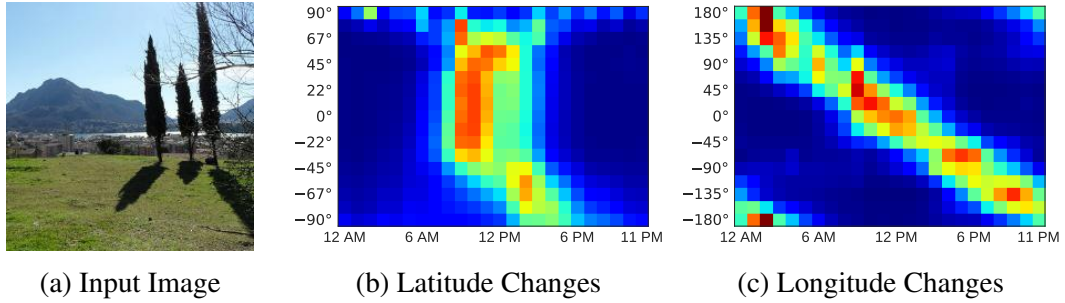


Figure 4.9: Marginal probability of GMT predictions by varying latitude and longitude.

the longitude value to be the ground truth (and vice versa). We found that the longitude of the image corresponds more with the hour prediction than the latitude, which matches expectations.

4.5 Conclusion

When learning about the world using images, the location and time an image was captured are useful pieces of metadata that are often available, but commonly overlooked. We presented a novel architecture for learning useful representations from images that takes advantage of this metadata. We found that for the task of transient attribute estimation, our method, despite being trained without manually obtained image-level annotations, learned image representations that outperform the representations learned using ImageNet, a rarely achieved feat on a frequently used benchmark for unsupervised representation learning tasks.

Copyright© Menghua Zhai, 2018.

Chapter 5

A Constraint for Location and Orientation

5.1 Introduction

In this chapter, we exploit overhead imagery to construct the constraint function for the camera orientation and location. Learning-based methods for pixel-level labeling of aerial imagery have long relied on manually annotated training data. Unfortunately, such data is expensive to create. Furthermore, its value is limited because a method trained on one dataset will typically not perform well when applied to another source of aerial imagery. The difficulty in obtaining datasets of sufficient scale for all modalities has hampered progress in applying deep learning techniques to aerial imagery. There have been a few notable exceptions [84, 88], but these have all used fairly coarse grained semantic classes, covered a small spatial area, and are limited to modalities in which human annotators are able to manually assign labels.

We propose a novel strategy for obtaining semantic labels for aerial image segmentation. See Figure 5.1 for a schematic overview of the approach. Our idea is to use existing methods for semantic image segmentation, which are tailored for ground images, and apply these to a large dataset of geo-tagged ground images. We use these semantically labeled images as a form of weak supervision and attempt to predict these semantic labels from an aerial image centered around the location of the ground image. We do not use a parametric transformation between the aerial and ground-level viewpoints. Instead, we use a dense representation, similar in spirit to the general representation, dubbed filter flow, described by Seitz and Baker [102].

There has been significant interest recently in predicting ground image features from

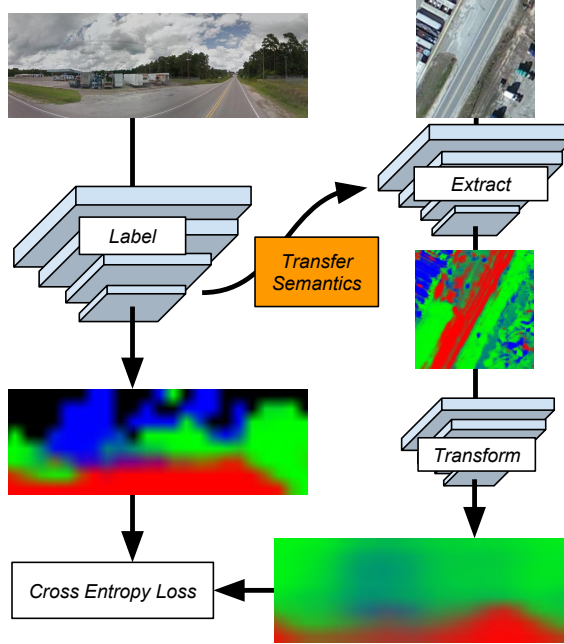


Figure 5.1: We learn to predict the ground-image segmentation directly from an aerial image of the same location, thereby transferring the semantics from the ground to the aerial image domain.

aerial imagery for the task of ground image geo-localization [128]. Our work is unique in that it is the first to attempt to predict a dense pixel-level segmentation of the ground image. We demonstrate the value of this approach in several ways.

Main Contributions: The main contributions of this work are: (1) a novel convolutional neural network (CNN) architecture that relates the appearance of a aerial image appearance to the semantic layout of a ground image of the same location, (2) demonstrating the value of our training strategy for pre-training a CNN to understand aerial imagery, (3) extensions of the proposed technique to the tasks of ground image localization, orientation estimation, and synthesis, and (4) an extensive evaluation of each of these techniques on large, real-world datasets. Together these represent an important step in enabling deep learning techniques to be extended to the domain of aerial image understanding.

5.2 Related Work

Learning Viewpoint Transformations Many methods have been proposed to represent the relationship between the appearance of two viewpoints. Seitz and Baker [102] model image transformations using a space-variant linear filter, similar to a convolution but vary-

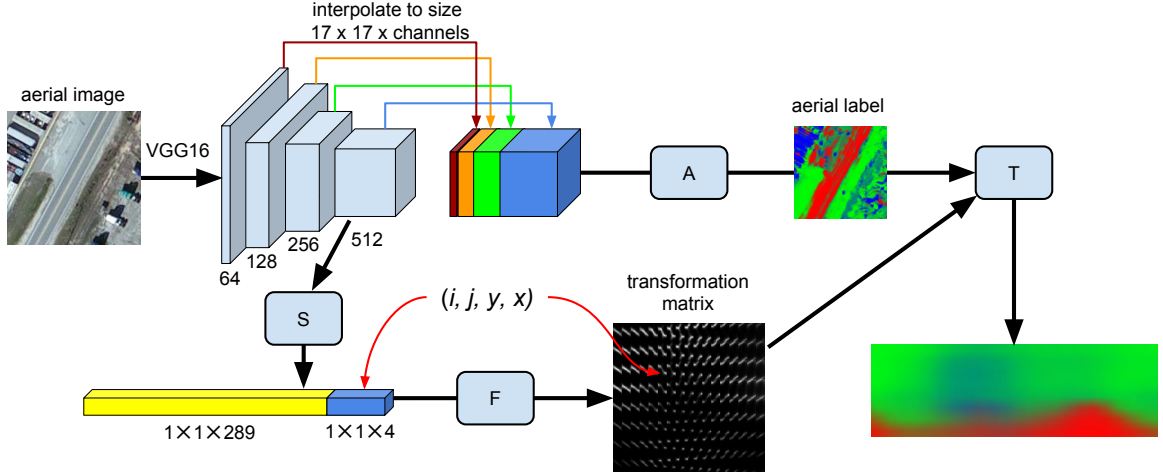


Figure 5.2: A visual overview of our network architecture. We extract features from an aerial image using the VGG16 architecture and form a hypercolumn using the PixelNet approach. These features are processed by three networks that consist of 1×1 convolutions: network A converts the hypercolumn into semantic features; network S extracts useful features from the aerial image for controlling the transformation; and network F defines the transformation between viewpoints. The transformation is applied, T , to the aerial semantic features to create a ground-level semantic labeling.

ing per-pixel. They highlight that a linear transformation of a vectorized representation of all the pixels in an image is very general; it can represent all standard parametric transformations, such as similarity, affine, perspective, and more. More recently, Jaderberg *et al.* [56] describe an end-to-end learnable module for neural networks, the spatial transformer, which allows explicit spatial transformations (*e.g.* scaling, cropping, rotation, non-rigid deformation) of feature maps within the network that are conditioned on individual data samples. Practically, including a spatial transformer allows a network to select regions of interest from an input and transform them to a canonical pose. Similarly, Tinghui *et al.* [145] address the problem of novel view synthesis. They observe that the visual appearance of different views is highly correlated and propose a CNN architecture for estimating appearance flows, a representation of which pixels in the input image can be used for reconstruction.

Relating Aerial and Ground-Level Viewpoints Several methods have been recently proposed to jointly reason about co-located aerial and ground image pairs. Luo *et al.* [79] demonstrate that aerial imagery can aid in recognizing the visual content of a geo-tagged ground image. Mátyus *et al.* [82] perform joint inference over monocular aerial imagery and stereo ground images for fine-grained road segmentation. Wegner *et al.* [120] build a map of street trees. Given the horizon line and the camera intrinsics, Ghouaiel and

Lefèvre [34] transform geo-tagged ground-level panoramas to a top-down view to enable comparisons with aerial imagery for the task of change detection. Recent work on cross-view image geo-localization [73, 74, 126, 128] has shown that convolutional neural networks are capable of extracting features from aerial imagery that can be matched to features extracted from ground imagery. Vo *et al.* [114] extend this line of work, demonstrating improved geo-localization performance by applying an auxiliary loss function to regress the ground-level camera orientation with respect to the aerial image. To our knowledge, our work is the first work to explore predicting the semantic layout of a ground image from an aerial image.

Semantic Segmentation of Aerial/Satellite Imagery There is a long tradition of using computer vision techniques for aerial and satellite image understanding [47, 124, 16]. Historically these two domains were distinct. Satellite imagery was typically lower-resolution, from a strictly top-down view, and with a diversity of spectral bands. Aerial imagery was typically higher-resolution, with a greater diversity of viewing angles, but with only RGB and NIR sensors. Recently these two domains have converged; we will use the term aerial imagery as we are primarily working with high-resolution RGB imagery. However, our approach could be applied to many types of aerial and satellite imagery. Kluckner *et al.* [63] address the task of semantic segmentation using a random forest to combine color and height information. More recent work has explored the use of CNNs for aerial image understanding. Mnih and Hinton propose a CNN for detecting roads in aerial imagery [84] using GIS data as ground truth. They extend their approach to handle omission noise and misregistration between the imagery and the labels [85]. These approaches require either extensive pixel-level manual annotation or existing GIS data. Our work is the first to demonstrate the ability to transfer a dense pixel-level labeling of ground imagery to aerial imagery.

Visual Domain Adaptation Domain adaptation addresses the misalignment of source and target domains [22]. A significant amount of work has explored domain adaptation for visual recognition [89]. Jhuo *et al.* [57] propose a low-rank reconstruction approach where the source features are transformed to an intermediate representation in which they can be linearly reconstructed by the target samples. Our work is most similar to that of Sun *et al.* [108], who propose a method for transferring scene categorizations and attributes from ground images to aerial imagery. Similar to our approach, they learn a transformation matrix which minimizes the distance between a source feature and the target feature. Our work differs in several ways: 1) we carry out the linear transformation not only in the

semantic dimensions but also in the spatial dimensions, 2) we constrain the transformation matrix such that the semantic meaning of the source feature and the target feature remains the same, 3) our transformation matrix is input dependent, and 4) we learn the transformation matrix as well as the source feature at the same time, in an end-by-end manner, which simplifies training.

5.3 Cross-view Supervised Training

We propose a novel training strategy for learning to extract useful features from aerial imagery. The idea is to predict the semantic scene layout, L_g , of a ground image, I_g , using only an aligned aerial image, I_a , from the same location. This strategy leverages existing methods for ground image understanding at training time, but does not require any ground imagery at testing time.

We represent semantic scene layout, L_g , as a pixel-level probability distribution over classes, such as *road*, *vegetation*, and *building*. We construct a training pair by collecting a georegistered ground panorama and an aerial image of the same location, orienting the panorama to the aerial image (panoramas are originally aligned with the road direction), and then extracting the semantic scene layout, L_g , of the panorama using an off-the-shelf method [9] with four semantic classes. We then use an end-to-end training strategy to learn to extract pixel-level features from the aerial image and transform them to the ground-level viewpoint.

5.3.1 Network Architecture

Our proposed network architecture is composed of four modules. A convolutional neural network (CNN), $L_a = A(I_a; \Theta_A)$, is used to extract semantic labels from the aerial imagery. Another CNN, $S(I_a; \Theta_S)$, uses features extracted from aerial imagery to help estimate the transformation matrix, $M = F(x_r, y_r, i_c, j_c, S(I_a; \Theta_S); \Theta_F)$, based on aerial image features and the pixel location in the respective images. Finally, we have a transformation module, $L_{g'} = T(L_a, M)$, that converts from the aerial viewpoint to the ground-level using the estimated transformation matrix, M . There are many choices for these components, and the remainder of this section describes the particular choices we made for this study. See Figure 5.2 for a visual overview of the architecture.

Aerial Image Feature Extraction For $A(I_a; \Theta_A)$, we use the VGG16 [103] base architecture and convert it to a pixel-level labeling method using the PixelNet approach [11].

The core idea is to interpolate intermediate feature maps of the base network to a uniform size, then concatenate them along the channels dimension to form a *hypercolumn*. In our experiments, we form the hypercolumn from $\text{conv}\{-1_2, 2_2, 3_3, 4_3\}$ of the VGG16 network. The hypercolumn, which is now $256 \times 256 \times 960$, is followed by three 1×1 convolutional layers, with 512, 512, and 4 output channels respectively. The first two 1×1 convolutions have ReLU activations, the final is linear. We designate the output of the final convolution as $L_a = A(I_a; \Theta_A)$. The output of this stage is transformed from a aerial viewpoint to a ground viewpoint by final stage of the network.

Cross-view Semantic Transformation We represent the transformation between the aerial and ground-level viewpoints as a linear operation applied channel-wise to L_a . To transform from the $h_a \times w_a \times 4$ aerial label, L_a , to the $h_g \times w_g \times 4$ ground label, $L_{g'}$, we need to estimate a $h_g w_g \times h_a w_a$ matrix, M . Given M , the transformation process is as follows: reshape the aerial label, L_a , into a $h_a w_a \times 4$ matrix, l_a ; multiply it by M to get $l_{g'}$; then reshape $l_{g'}$ to the size of the ground label, L_g , to form our estimate of the ground label, $L_{g'}$. We constrain M such that the sum of each row is unit. To account for the expected layout of the scene, and to handle the sky class (which is not visible from the aerial image), we carry out the transformation on the logits of l_a, f_a , and add a bias term, b to get the logits of $l_{g'}, f_{g'}$: $f_{g'} = M f_a + b$.

There are many ways of representing the transformation matrix, M . The naïve approach is to treat M as a matrix of learnable variables. However, this approach has two downsides: (1) the transformation does not depend on the content of the aerial image and (2) the number of parameters scales quadratically with the number of pixels in L_a and L_g .

We represent each element, M_{rc} , in the transformation matrix, M , as the output of a neural network, F , which is conditioned on the aerial image, I_a , and the location in the input and output feature maps. More precisely, each element $M_{rc} = F(x_r, y_r, i_c, j_c, S(I_a; \Theta_S))$, where $(i_c, j_c) \in [0, 1]$, is the aerial image pixel of the corresponding element, $(y_r, x_r) \in [0, 1]$ is the ground image pixel of the corresponding element.

We now define the architecture of the transformation estimation neural network, F . The value of the transformation matrix at location (r, c) is computed through a neural network, \tilde{F} , followed by a *softmax* function to normalize the impact of all pixels sampled from the

aerial image:

$$M_{rc} = F(r, c, S(I_a; \Theta_S)) = \frac{e^{\tilde{F}_{r,c}}}{\sum_{c'} e^{\tilde{F}_{r,c'}}},$$

where:

$$\begin{aligned}\tilde{F}_{r,c} &= \tilde{F}(i, j, y, x, S(I_a; \Theta_S)), \text{ and} \\ i &= \lfloor c/w_a \rfloor / h_a, \quad j = \text{mod}(c, w_a)/w_a, \\ y &= \lfloor r/w_g \rfloor / h_g, \quad x = \text{mod}(r, w_g)/w_g.\end{aligned}$$

The base network, \tilde{F} , is a multi-layer perceptron, with ReLU activation functions, that takes as input a 293-element vector. The network has three layers, with 128, 64 and 1 output channels respectively (refer to the lower part of Figure 5.2). The naïve approach can be considered a special case of this representation where we ignore the aerial image and use a one-hot encoding representation of rows and columns.

As described above, there are two main advantages of our approach of representing the transformation matrix: a reduction in the number of parameters when M is large and the ability to adapt to different aerial image layouts. An additional benefit is that if we change the resolution of our input and output feature maps it is easy to create a new transformation matrix, M , without needing to resort to interpolation.

5.3.2 Dataset

We collect our training and testing dataset from the CVUSA dataset [128]. CVUSA contains approximately 1.5 million geo-tagged pairs of ground and aerial images from across the United States. We use the Google Street View panoramas of CVUSA as our ground images. For each panorama, we also download an aerial image at zoom level 19 from Microsoft Bing Maps in the same location. We filter out panoramas with no available corresponding aerial imagery. Using the camera’s extrinsic information, we then warp the panoramas to align with the aerial images. We also crop the panoramas vertically to reduce the portion of the sky and ground pixels. In total, we collected 35,532 image pairs for training and 8,884 image pairs for testing. Some examples aerial/ground image pairs in our dataset are shown Figure 5.3.

5.3.3 Implementation Details

We implemented the proposed architecture using Google’s TensorFlow framework [1]. We train our networks for 10 epochs with the Adam optimizer [62]. We enable batch normal-



Figure 5.3: Examples of aligned aerial/ground image pairs from our dataset. (row 1) In the aerial images, north is the up direction. In the ground images, north is the central column. (row 2-4) Image dependent receptive fields estimated by our algorithm as follows: 1) fix ground locations (y, x) (locations in squares); 2) select all (i, j) (locations in contours) with high $\tilde{F}(i, j, y, x, S(I_a; \Theta_S))$ values. Corresponding fields between the aerial image and the ground image are shown in the same color.

ization [49] with decay 0.9 in all convolutional and fully-connected layers (except for the output layers) to accelerate the training process.

The training procedure is as follows: for a given cross-view image pair, (I_a, I_g) , we first compute for the ground semantic pixel label: $I_g \rightarrow L_g$, using SegNet [9]. We then minimize the cross entropy between L_g and $T(A(I_a; \Theta_A); \Theta_T)$ with respect to the model parameters, Θ_A and Θ_T . The resulting architecture requires a significant amount of memory to output the full final feature map, which would normally result in very small batch sizes for GPU training. Due to the PixelNet approach of using interpolation to scale the feature maps, we are able to perform sparse training. Instead of outputting the full-size feature map, we only extract a dense grid of points, the resulting feature map is $17 \times 17 \times 4$. Despite this, at testing time, we can provide an aerial image and generate a full-resolution, semantically meaningful feature map.

5.4 Evaluation and Applications

In this section, we will show that our network architecture can be used in four different tasks: 1) weakly supervised semantic learning, 2) aerial imagery labeling, 3) orientation regression and geo-calibration, and 4) cross-view image synthesis. Additional qualitative

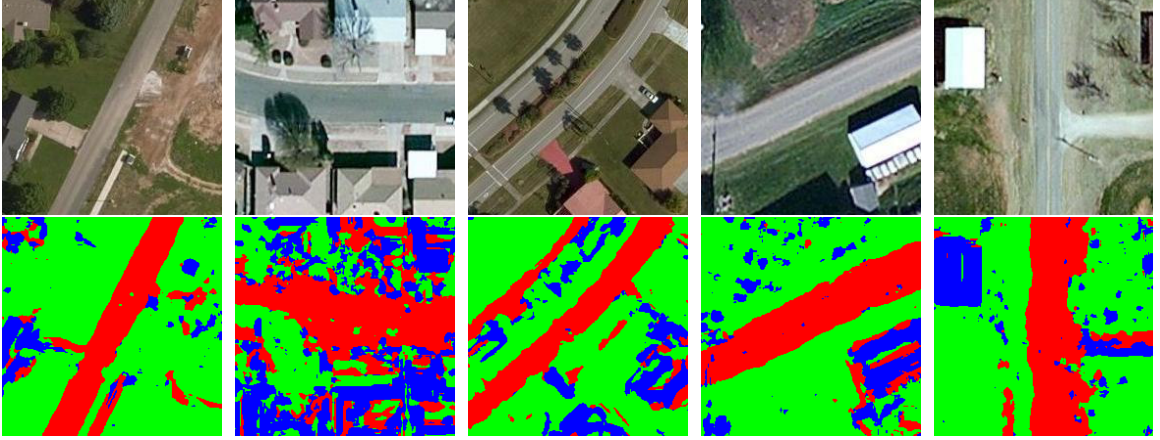


Figure 5.4: Example outputs from our weakly supervised learning method on test images. For each aerial image (top), we show the pixel-level labeling inferred by our model, which uses only noisy ground image segmentation as labels. We visualize three classes: *road* (red), *vegetation* (green), and *man-made* (blue).

results and the complete network structure used for cross-view image synthesis can be found in our supplemental materials.

5.4.1 Weakly Supervised Learning

We trained our full network architecture (with randomly initialized weights) to predict ground-level semantic labeling using the dataset described in Section 5.3.2. Figure 5.4 shows example output, L_a , from the aerial image understanding CNN. This demonstrates that the resulting network has learned to extract semantic features from an aerial image, all without any manual annotated aerial imagery.

While these results are compelling, they could be better with a higher quality ground-image segmentation method. The method we use, SegNet [9], was trained on mostly urban scenes, but many of our images are from rural and suburban areas. The end result is that certain classes are often mislabeled in the ground imagery, including *dirt* and *building*. In addition, because the panoramas and aerial images were not captured at the same time, we are unable to accurately model transient objects, such as vehicles and pedestrians. All these factors make the dataset very challenging for training. Given these limitations, it is, perhaps, surprising that the resulting aerial image segmentation method works so well. In the following section, we show using this network as a starting point for strongly supervised aerial image segmentation outperforms two standard initialization methods.



Figure 5.5: An example from the ISPRS dataset [94]. (left) Near infrared image; (middle) The same image after pre-processing; (right) Ground-truth annotation of the image.

5.4.2 Cross-view for Pre-training

We evaluate our proposed technique as a pre-training strategy for the task of semantic-pixel labeling of aerial imagery. Starting from the optimal weights from the previous section, we finetune and evaluate using the ISPRS dataset [94]. This dataset contains 33 true orthophotos captured over Vaihingen, Germany. The ground sampling distance is 9 cm/px and there are over 168 million pixels in total. Ground truth is provided for 16 photos; each pixel is assigned one of six categories: *Impervious surfaces*, *Building*, *Low vegetation*, *Tree*, *Car*, and *Clutter/background*.

Image Processing Compared to the Bing Maps imagery we used for pre-training, images in the ISPRS dataset are at a different spatial scale and the color channels represent different frequency bands (the R channel is actually a near infrared channel). To ensure that the pre-trained network weights are appropriate for the new dataset, we adjusted the scale and color channels as follows. We first resize the ISPRS images to the equivalent of Bing Maps zoom level 19. We then label a pixel as vegetation if $\frac{R}{R+G+B}$ is greater than 0.4. For each pixel labeled as vegetation, we halve the R channel intensity and swap the R and G channels. The resulting images, shown in Figure 5.5, are much closer in appearance to the Bing Maps imagery than the raw imagery.

Evaluation Protocol We split the 16 annotated images into training (images 5, 7, 11, 13, 15, 17, and 21), validation sets (images 1 and 3) and testing (images 23, 26, 28, 30, 32, 37, and 40). From each set we extracted a set of 224×224 subwindows (respectively 82, 12, and 34 from training, validation, and testing respectively). We then compared performance with different numbers of training images: 1, 2, 7, 20, 54, and 82. We evaluated the performance in terms of the average precision for all pixels. We ignore the *Cluster/background* pixels because of the low number of assigned pixels.

Training and Testing We used the same architecture with the aerial feature extractor, $A(I_a; \Theta_A)$, defined in Section 5.3.1, to do the semantic labeling on ISPRS. During training, we use the Adam optimizer to minimize the cross entropy between the network outputs and the labels. We use a batch size of 8, randomly sample 1,000 pixels per image for sparse training, and train the network till convergence. We run the validation set every 1,000 training iterations and save the optimal network weights for testing. During testing, we sample all pixels on the image to generate the dense labeling.

We experiment using three different initializations of the VGG16 convolutional layers and finetune the remaining layers of the network: 1) Ours: initialize with model pre-trained using our framework; 2) Random: initialize using Xavier initialization [37]; 3) VGG16: initialize with model pre-trained on ImageNet.

Since the VGG16 model we used in this experiment is trained without batch normalization, it may be less competitive. To achieve a fair comparison, we turned off batch normalization in this experiment and re-trained the network for 15 epochs to get the pre-trained model.

Our results (Figure 5.6) show that finetuning from the VGG16 model performs poorly on the aerial image labeling task. We think that the patterns it learned mostly from the ground image may hinder pattern learning for aerial imagery. Our method outperforms both of the other initialization strategies. We also present the prediction precision per class in Table 5.1. We highlight that our method does better especially on the *Building*, *Low Vegetation*, and *Tree* classes, which can also be found in pre-training annotations.

5.4.3 Cross-view for Geo-Calibration

We show how the ground-level feature maps we estimate from aerial imagery can be used to estimate the orientation and location of a ground image. We show quantitative results for the orientation estimation task and qualitative results for simultaneous orientation and location estimation. We use the following datasets for all experiments:

- **CVUSA:** We use the test set introduced in Section 5.3.2 to create this dataset, it has two parts for orientation estimation and geo-calibration, respectively. For the orientation regression task, we rotate the aerial image to a random angle. For the fine-grained geo-calibration experiment, we center-crop the aerial image around a random x, y offset, then rotate the image to a random angle. In both experiments, the heading direction of the ground images are the same. We center crop a 224×448 cutout from each ground image as the query image.

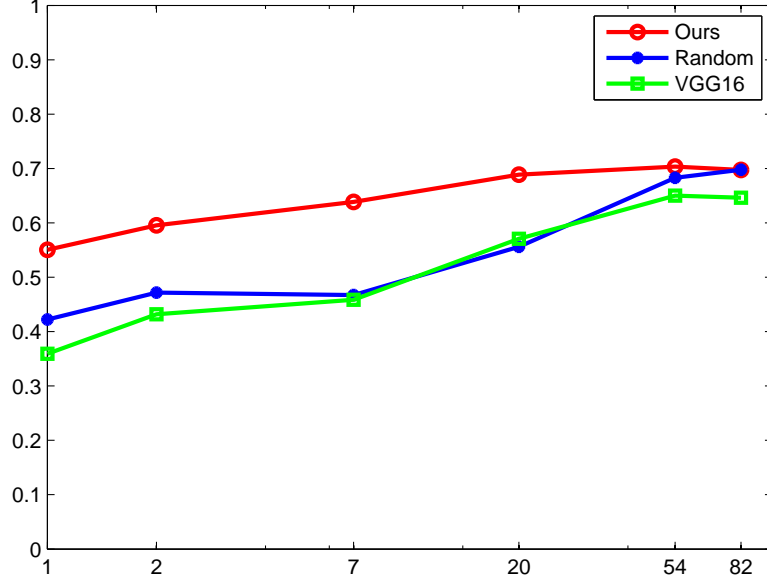


Figure 5.6: Performance comparison of different initialization methods on the ISPRS segmentation task. The x -axis is the number of training images and the y -axis is average precision.

Table 5.1: Per-Class Precision on the ISPRS Segmentation Task

Class	Init.	Number of training samples					
		1	2	7	20	54	82
Imp.	Ours	0.67	0.74	0.63	0.64	0.66	0.64
	Random	0.70	0.70	0.54	0.62	0.61	0.73
	VGG16	0.60	0.55	0.55	0.61	0.70	0.59
Bldg	Ours	0.72	0.76	0.76	0.80	0.75	0.78
	Random	0.56	0.62	0.63	0.64	0.82	0.71
	VGG16	0.78	0.72	0.71	0.69	0.70	0.75
Low.	Ours	0.37	0.43	0.51	0.65	0.67	0.67
	Random	0.29	0.29	0.29	0.37	0.67	0.64
	VGG16	0.25	0.25	0.29	0.44	0.53	0.57
Tree	Ours	0.68	0.54	0.71	0.71	0.74	0.74
	Random	0.42	0.46	0.49	0.56	0.71	0.69
	VGG16	0.36	0.44	0.50	0.55	0.65	0.74
Car	Ours	0.13	0.46	0.67	0.48	0.48	0.49
	Random	0.05	0.08	0.10	0.25	0.45	0.57
	VGG16	0.05	0.11	0.20	0.20	0.25	0.23

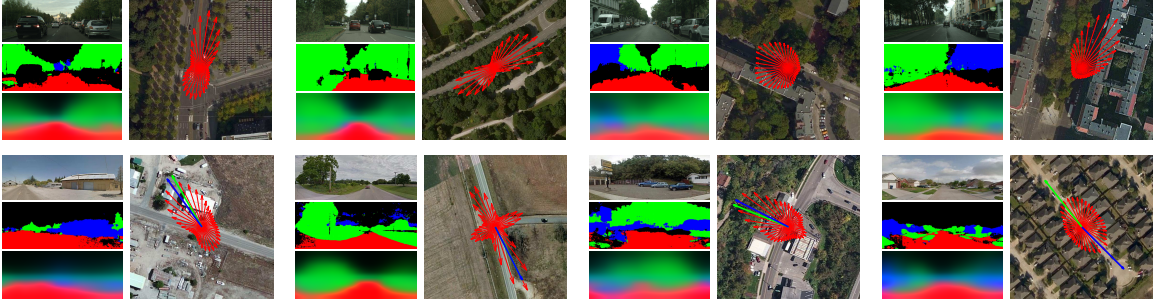


Figure 5.7: Qualitative results of orientation predictions on Cityscapes dataset (top) and CVUSA (bottom). The I_g , L_g and $L_{g'}$ are stacked vertically on the left side of the aerial image. We visualize three classes on the labels: *road* (red), *vegetation* (green), and *man-made* (blue). The discrete PDFs of the ground camera orientation are visualized with red arrows, whose lengths indicate the magnitudes. In the CVUSA results, the ground truth (green) and the optimal prediction (blue) are also shown with the orientation PDF. The last prediction result is a typical failure case of our method, where the scene is symmetric from the aerial point of view.

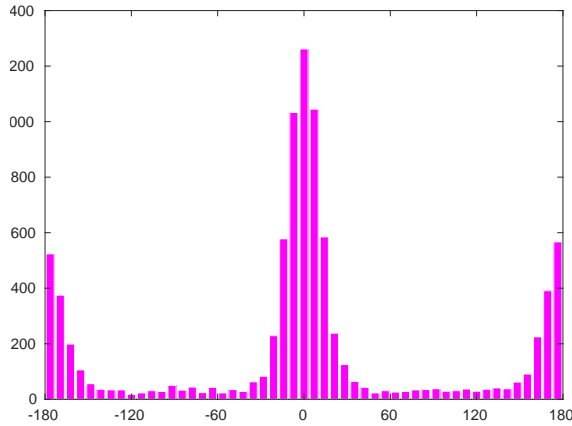


Figure 5.8: Histogram of the orientation errors on the CVUSA dataset.

- **Cityscapes:** The Cityscapes dataset [19] is a recently released benchmark dataset designed to support the task of urban scene understanding through semantic pixel labeling. It consists of stereo video from 50 different cities and fine pixel-level annotations for 5,000 frames and coarse pixel-level annotations for 20,000 frames.

Orientation Estimation For this task, we assume the location and focal length of the ground image, I_g , is known but the orientation is not. The intuition behind our method is that the semantic labeling of the ground image will be most similar to the feature map of the aerial image at the actual orientation. For a query ground image, I_g , the first step is to download the corresponding aerial image, I_a . We then infer the semantic labeling of the query image, $I_g \rightarrow L_g$, and predict the ground image label from the aerial image

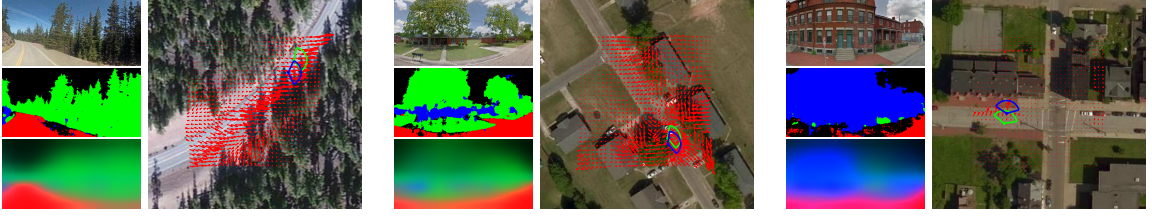


Figure 5.9: Fine-grained geo-calibration results on CVUSA. (left) From top to bottom are the I_g , L_g , and $L_{g'}$ respectively. We visualize three classes on the labels: *road* (red), *vegetation* (green), and *man-made* (blue). (right) Orientation flow map (red), where the arrow direction indicates the optimal direction at that location and length indicates the magnitude. We also show the optimal prediction and the ground-truth frustums in blue and green respectively.

using our learned network, $I_a \rightarrow L_{g'}$. We assign an energy to each possible orientation by computing the cross entropy between L_g and $L_{g'}$ in a sliding window fashion across all possible orientations. We select the orientation with the lowest energy. We present sample results in Figure 5.7 and a histogram of the orientation errors on the CVUSA dataset in Figure 5.8.

Fine-grained Geo-Calibration For this task, we assume that we know the focal length of the camera and have a rough estimate of the camera location (i.e., with 100 meters). We extract 256×256 aerial images from the area around our rough estimate and extract the corresponding ground-level feature maps. We apply our orientation estimation procedure to each feature map. The result is a distribution over orientations for each location. Figure 5.9 shows several example results, including the most likely direction for each location, as well as the most likely location and orientation pair.

5.4.4 Synthesizing Ground Images from Aerial Images

We propose a novel application to infer a ground image by using features extracted from our network. We begin by describing our network structure and then show qualitative results for different generated ground-level scenes.

Our network architecture is based on the deep, directed generative model proposed by Kim *et al.* [61]. Their model consists of two parts: a deep generator, G , which generates images that try to minimize a deep energy model, E . A low energy implies the image is real and high energy implies the image is fake. The architecture and training methods are inspired by generative adversarial networks [39], however it provides a energy-based formulation of the discriminator to address common instabilities of adversarial training.



Figure 5.10: Synthesized ground-level views. Each row shows an aerial image (left), its corresponding ground-level panorama (top-right), and predicted ground-level panorama (bottom-right).

A complete description of the architecture used to design the deep generator and deep energy model is provided in our supplemental materials. We begin by extracting an $8 \times 40 \times 512$ cross-view feature map, f , that has been learned to relate an aerial and ground image pair. The generator is given f along with random noise, z , as input. The generator outputs a 64×320 panorama, $I_{\hat{g}}$, that represents the predicted ground image. The cross-view feature, predicted panorama, and the ground truth panorama, I_g , are then passed into the energy model which returns an energy function,

$$E_{\Theta}(\mathbf{f}, \mathbf{I}_{\mathbf{g}^*}) = \frac{1}{\sigma^2} \mathbf{f}^T \mathbf{f} - \mathbf{b}^T \mathbf{f} - \sum_i \log(1 + e^{W_i^T \mathbf{I}_{\mathbf{g}^*} + b_i}),$$

similar to the free energy of a Gaussian Restricted Boltzmann Machine (RBM). Batch normalization [49] is applied in every layer of both models, except for the final layers. ReLU activations are used throughout the generator and Leaky ReLU, with leak parameter $\alpha = 0.2$, are used in the energy model. The models are updated in an alternating fashion, where the generator is updated twice for every update of the energy model. Both the generator and energy model are optimized using the Adam optimizer, with moment parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$. We train using batch sizes of 32 for 30 epochs.

Example outputs generated by our network are shown in Figure 5.10. Each row contains an aerial image (left), its respective ground panorama (top-right), and our prediction of the ground scene layout (bottom-right), which would ideally be the same as its above image. The network has learned the most common features, such as roads and their orientations, as well as trees and grass. However, it has difficulty hallucinating buildings and the sky, which is likely caused by highly variable appearance factors.

We note that the resolution of the synthesized ground-level panoramas is much lower than the original panorama, however adversarial generation of high-resolution images is an active area of research. We expect that in the near future we will be able to use our learned features in a similar manner to generate full-resolution panoramas. Additionally, algorithmic improvements to our ground image segmentation method would provide more photo-realistic predictions.

5.5 Conclusion

We introduced a novel strategy for using automatically labeled ground images as a form of weak supervision for learning to understand aerial images. The key is to simultaneously learn to extract features from the aerial image and learn to map from the aerial to the ground image. We demonstrated that by using this process we are able to automatically

extract semantically meaningful features from aerial imagery, refine these to obtain more accurate pixel-level labeling of aerial imagery, estimate the location and orientation of a ground image, and synthesize novel ground-level views. The proposed technique is equally applicable to other forms of imagery, including NIR, multispectral, and hyperspectral. For future work, we plan to explore richer ground image annotation methods to explore the limits of what is predictable about a ground-level view from an aerial view.

Chapter 6

Summary

The goal of geo-calibration is to learn the camera pose, location and the time when the image was captured. Our thesis focused on developing deep geo-calibration algorithms for image understanding. Compared to previous work, our approaches output probabilistic predictions that handle the uncertainty of the scene better. Furthermore, we show that learning to geo-calibrate a camera allows us to implicitly learn to understand the content of the scene.

In Chapter 2, we proposed a flexible model to compute the probability over geo-calibration parameters. We defined a score function in the space of geo-calibration parameters, to measure how well the geo-calibration parameters are able to describe the content of the input image. The score function is a summation of a series of weighted constraint functions. It is proportional to the probability density function (PDF) of the distribution over the geo-calibration parameter setting. Since the analytic solution to the integral of the score function does not exist, we approximated the PDF using the MCMC method. Our model is flexible such that one can apply any kind and any number of constraint functions. We also demonstrated in our experiments that the score function with more constraint functions yields more accurate predictions.

In Chapter 3, we developed a constraint function for the camera pose. Since the camera roll and pitch angles can be derived from the horizon line position, our algorithm focuses on identifying the horizon line from the input image. Compared to the previous work, which applies bottom-up approaches where the horizon line is estimated after finding vanishing points, we proposed a novel horizon-first approach where identifying possible horizon lines happens before detecting vanishing points. Firstly, we trained a CNN to estimate the probability distribution over the horizon line. Then we sampled horizon line candidates from it. For each horizon candidate, we detected likely vanishing points on it. Unlike the previous work which searches for vanishing points on the 2D image frame, our method is able to

accelerate this process by reducing the searching space to 1D (a line). The probability of the horizon line is measured by how convincing the vanishing points are found on it.

In Chapter 4, we developed the constraint function for the image capture time and the camera location. We trained a CNN on a large dataset to predict the discrete probabilities over the capture time and the camera location given the input image, as well as to learn a geo-temporal representation of the image. Unlike most full supervision methods which require tedious work for annotation collection, our training dataset consists of millions of webcam frames and smart phone photos with free time/location tags. In the experiments, we demonstrated that learning to geo-calibrate the camera helps learn useful image representations for image understanding.

In Chapter 5, we exploited overhead imagery to construct the constraint function for the camera orientation and location. We trained a CNN to estimate the semantic layout of the ground-view image from the overhead image. During training, the CNN learns the pixel-to-pixel correspondences from the aerial image to the ground image. Then it projects the semantic layout of the overhead image to the ground view according to the learned correspondences. When computing the output of the constraint function, we feed the aerial image of the given location and orientation to the network. By comparing the aerial-to-ground layout with the layout computed from the ground image, we measure the consistency between these two and use the consistency value as the output of the constraint function. In our experiments, we also demonstrated the efficiency of our method for pre-training CNNs for aerial image segmentation.

This thesis proposed a flexible probabilistic framework for image geo-calibration. The key to the success of this framework is to find constraint functions that can accurately model the relationship between geo-calibration parameters and the content of the image. We developed constraint functions for different sets of parameters to fit in the framework. Furthermore, we also showed that learning to geo-calibrate cameras helps learning useful features for image understanding. There are several possible future research directions for extending this work. For example, like aerial imagery, map data also provides rich information about the environment from the top-view perspective. In Chapter 5, we can incorporate map data as top-view imagery during training. Another future work direction is to use Bayesian neural networks (BNNs). Bayesian neural networks have drawn a lot of attentions in recent years. The probabilistic nature of BNNs makes them excellent tools to model uncertainties. We can replace CNNs with BNNs in our algorithms to achieve better performances. Overall, we hope our work can provide new ways for geo-calibration and image feature learning.

Bibliography

- [1] Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016. 53
- [2] Pratik Agarwal, Wolfram Burgard, and Luciano Spinello. Metric localization using google street view. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 3111–3118. IEEE, 2015. 2
- [3] Andrés Almansa, Agnès Desolneux, and Sébastien Vamech. Vanishing point detection without any a priori information. 25(4):502–507, 2003. 21
- [4] Matthew E Antone and Seth Teller. Automatic recovery of relative camera rotations for urban scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2000. 19
- [5] Michel Antunes and Joao P Barreto. A global approach for the detection of vanishing points and mutually orthogonal vanishing directions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 19, 21
- [6] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *CVPR*, pages 5297–5307, 2016. 5
- [7] Georges Baatz, Kevin Köser, David Chen, Radek Grzeszczuk, and Marc Pollefeys. Handling urban location recognition as a 2d homothetic problem. In *European Conference on Computer Vision*, 2010. 19
- [8] Georges Baatz, Olivier Saurer, Kevin Köser, and Marc Pollefeys. Large scale visual geo-localization of images in mountainous terrain. In *European Conference on Computer Vision*, 2012. 1, 3, 12

- [9] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015. 51, 54, 55
- [10] Ryan Baltenberger, Menghua Zhai, Connor Greenwell, Scott Workman, and Nathan Jacobs. A Fast Method for Estimating Transient Scene Attributes. In *IEEE Winter Conference on Applications of Computer Vision*, 2016. 35, 37
- [11] Aayush Bansal, Xinlei Chen, Bryan Russell, Abhinav Gupta, and Deva Ramanan. Pixelnet: Towards a General Pixel-level Architecture. *arXiv preprint arXiv:1609.06694*, 2016. 51
- [12] Olga Barinova, Victor Lempitsky, Elena Tretyak, and Pushmeet Kohli. Geometric image parsing in man-made environments. In *European Conference on Computer Vision*, 2010. 20, 21, 30, 31, 32
- [13] Stephen T Barnard. Interpreting perspective images. *Artificial intelligence*, 21(4):435–462, 1983. 19, 21
- [14] Robert C Bolles and Martin A Fischler. A ransac-based approach to model fitting and its application to finding cylinders in range data. In *International Joint Conference on Artificial Intelligence*, 1981. 21, 26
- [15] Wolfram Burgard, Dieter Fox, Daniel Hennig, and Timo Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *National Conference on Artificial Intelligence*, 1996. 11
- [16] Gong Cheng and Junwei Han. A survey on object detection in optical remote sensing images. *CoRR*, abs/1603.06201, 2016. 50
- [17] Siddhartha Chib and Edward Greenberg. Understanding the metropolis-hastings algorithm. *The American Statistician*, 49(4):327–335, 1995. 14
- [18] Robert T Collins and Richard S Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In *IEEE International Conference on Computer Vision*, 1990. 2, 21
- [19] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 59

- [20] James M Coughlan and Alan L Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *IEEE International Conference on Computer Vision*, 1999. 21
- [21] Antonio Criminisi, Ian Reid, and Andrew Zisserman. Single view metrology. *International Journal of Computer Vision*, 40(2):123–148, 2000. 19
- [22] Hal Daume III and Daniel Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126, 2006. 50
- [23] Frank Dellaert, Wolfram Burgard, Dieter Fox, and Sebastian Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1999. 11
- [24] Patrick Denis, James H Elder, and Francisco J Estrada. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *European Conference on Computer Vision*, 2008. 20, 21, 31
- [25] Carl Doersch. *Supervision Beyond Manual Annotations for Learning Visual Representations*. PhD thesis, Carnegie Mellon University, 2016. 35
- [26] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *IEEE International Conference on Computer Vision*, pages 1422–1430, 2015. 5, 35, 37
- [27] Carl Doersch, Saurabh Singh, Abhinav Gupta, Josef Sivic, and Alexei A. Efros. What makes paris look like paris? *ACM Transactions on Graphics (SIGGRAPH)*, 31(4), 2012. 1, 3, 10
- [28] Lixin Duan, Dong Xu, and Ivor Tsang. Learning with augmented features for heterogeneous domain adaptation. *arXiv preprint arXiv:1206.4660*, 2012. 5
- [29] Roman Fedorov, Piero Fraternali, and Marco Tagliasacchi. Snow phenomena modeling through online public media. In *IEEE International Conference on Image Processing*, 2014. 35
- [30] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, pages 2960–2967, 2013. 5

- [31] Basura Fernando, Tatiana Tommasi, and Tinne Tuytelaars. Joint cross-domain classification and subspace learning for unsupervised adaptation. *Pattern Recognition Letters*, 65:60–66, 2015. 5
- [32] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte carlo localization: Efficient position estimation for mobile robots. 1999. 11
- [33] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*. Springer, 2016. 8
- [34] Nehla Ghouaiel and Sébastien Lefèvre. Coupling ground-level panoramas and aerial imagery for change detection. *Geo-spatial Information Science*, 19(3):222–232, 2016. 50
- [35] Shiry Ginosar, Kate Rakelly, Sarah Sachs, Brian Yin, and Alexei A Efros. A century of portraits: A visual historical record of american high school yearbooks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1–7, 2015. 3
- [36] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feed-forward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010. 39
- [37] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feed-forward neural networks. In *Aistats*, volume 9, pages 249–256, 2010. 57
- [38] Clément Godard, Oisín Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, volume 2, page 7, 2017. 8
- [39] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 60
- [40] Lazaros Grammatikopoulos, George Karras, and Elli Petsa. An automatic approach for camera calibration from vanishing points. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62(1):64–76, 2007. 19
- [41] J-S Gutmann, Wolfram Burgard, Dieter Fox, and Kurt Konolige. An experimental comparison of localization methods. 1998. 11

- [42] James Hays and Alexei A. Efros. im2gps: estimating geographic information from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 3, 4, 10, 12, 35, 37
- [43] RD Hill, BJ Le Boeuf, and RM Laws. Elephant seals: population ecology, behavior, and physiology, 1994. 3
- [44] Derek Hoiem, Alexei A Efros, and Martial Hebert. Putting objects in perspective. *International Journal of Computer Vision*, 80(1):3–15, 2008. 21
- [45] Yannick Hold-Geoffroy, Kalyan Sunkavalli, Jonathan Eisenmann, Matt Fisher, Emiliano Gambaretto, Sunil Hadap, and Jean-François Lalonde. A perceptual measure for deep single image camera calibration. In *CVPR*, 2018. 4
- [46] Paul VC Hough. Machine analysis of bubble chamber pictures. In *International Conference on High Energy Accelerators and Instrumentation*, volume 73, 1959. 21
- [47] Andres Huertas and Ramakant Nevatia. Detecting buildings in aerial images. *Computer Vision, Graphics, and Image Processing*, 41:131–152, 1988. 50
- [48] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML’15, pages 448–456. JMLR.org, 2015. 39, 41
- [49] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 54, 62
- [50] Mohammad T. Islam, Nathan Jacobs, Hui Wu, and Richard Souvenir. Images+weather: Collection, validation, and refinement. In *IEEE CVPR Workshop on Ground Truth*, 2013. 37
- [51] Nathan Jacobs, Mohammad T. Islam, and Scott Workman. Cloud motion as a calibration cue. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 19, 35
- [52] Nathan Jacobs, Kyla Miskell, and Robert Pless. Webcam Geo-localization using Aggregate Light Levels. In *IEEE Workshop on Applications of Computer Vision*, 2011. 3, 12

- [53] Nathan Jacobs, Nathaniel Roman, and Robert Pless. Consistent temporal variations in many outdoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 35, 37, 40
- [54] Nathan Jacobs, Nathaniel Roman, and Robert Pless. Toward Fully Automatic Geo-Location and Geo-Orientation of Static Outdoor Cameras. In *IEEE Workshop on Applications of Computer Vision*, 2008. 3, 12
- [55] Nathan Jacobs, Scott Satkin, Nathaniel Roman, Richard Speyer, and Robert Pless. Geolocating static cameras. In *IEEE International Conference on Computer Vision*, 2007. 3, 12, 37
- [56] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 49
- [57] I-Hong Jhuo, Dong Liu, DT Lee, and Shih-Fu Chang. Robust visual domain adaptation with low-rank reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 50
- [58] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 30
- [59] Imran N Junejo and Hassan Foroosh. Estimating geo-temporal location of stationary cameras using shadow trajectories. In *European Conference on Computer Vision*, 2008. 3, 12
- [60] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Convolutional networks for real-time 6-dof camera relocalization. In *IEEE International Conference on Computer Vision*, 2015. 1, 4
- [61] Taesup Kim and Yoshua Bengio. Deep directed generative models with energy-based probability estimation. *arXiv preprint arXiv:1606.03439*, 2016. 60
- [62] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 39, 53
- [63] Stefan Kluckner, Thomas Mauthner, Peter M. Roth, and Horst Bischof. Semantic classification in aerial imagery by integrating appearance and height information. In *ACCV*, 2009. 50

- [64] Jana Košecká and Wei Zhang. Video compass. In *European Conference on Computer Vision*, 2002. 19
- [65] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012. 23
- [66] Pierre-Yves Laffont, Zhile Ren, Xiaofeng Tao, Chao Qian, and James Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Transactions on Graphics (SIGGRAPH)*, 33(4):149, 2014. 35, 37, 40
- [67] Jean-François Lalonde, Srinivasa G Narasimhan, and Alexei A Efros. What do the sun and the sky tell us about the camera? *International Journal of Computer Vision*, 88(1):24–51, 2010. 3, 12, 37
- [68] Stefan Lee, Nicolas Maisonneuve, David Crandall, Alexei Efros, and Josef Sivic. Linking past to present: Discovering style in two centuries of architecture. In *IEEE International Conference on Computational Photography (ICCP)*, 2015. 3
- [69] Jose Lezama, Rafael Grompone von Gioi, Gregory Randall, and Jean-Michel Morel. Finding vanishing points via point alignments in image primal and dual domains. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 2, 20, 21, 30, 31, 32
- [70] Yunpeng Li, Noah Snavely, Dan Huttenlocher, and Pascal Fua. Worldwide pose estimation using 3d point clouds. In *European Conference on Computer Vision*, 2012. 1, 2, 12
- [71] Yunpeng Li, Noah Snavely, and Daniel P Huttenlocher. Location recognition using prioritized feature matching. In *European Conference on Computer Vision*, 2010. 4, 10
- [72] Zhengqi Li and Noah Snavely. Learning intrinsic image decomposition from watching the world. *arXiv preprint arXiv:1804.00582*, 2018. 37
- [73] Tsung-Yi Lin, Serge Belongie, and James Hays. Cross-view image geolocalization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 3, 4, 12, 50
- [74] Tsung-Yi Lin, Yin Cui, Serge Belongie, and James Hays. Learning deep representations for ground-to-aerial geolocalization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 4, 37, 50

- [75] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 8
- [76] Jingchen Liu and Yanxi Liu. Local regularity-driven city-scale facade detection from aerial images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 19
- [77] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999. 2, 5
- [78] Cewu Lu, Di Lin, Jiaya Jia, and Chi-Keung Tang. Two-class weather classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 35, 37
- [79] Jiebo Luo, Jie Yu, Dhiraj Joshi, and Wei Hao. Event recognition: viewing the world with a third eye. In *ACM International Conference on Multimedia*, 2008. 49
- [80] Evelyne Lutton, Henri Maitre, and Jaime Lopez-Krahe. Contribution to the determination of vanishing points using hough transform. 16(4):430–438, 1994. 21
- [81] Michael J Magee and Jake K Aggarwal. Determining vanishing points from perspective images. *Computer Vision, Graphics, and Image Processing*, 26(2):256–267, 1984. 19, 21
- [82] Gellért Mátyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Hd maps: Fine-grained road segmentation by parsing ground and aerial images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3611–3619, 2016. 49
- [83] Kevin Matzen and Noah Snavely. Scene chronology. In *European Conference on Computer Vision*, 2014. 3, 37
- [84] Volodymyr Mnih and Geoffrey E Hinton. Learning to detect roads in high-resolution aerial images. In *European Conference on Computer Vision*, 2010. 47, 50
- [85] Volodymyr Mnih and Geoffrey E Hinton. Learning to label aerial images from noisy data. In *International Conference on Machine Learning*, 2012. 50
- [86] Sang Min Oh, Sarah Tariq, Bruce N Walker, and Frank Dellaert. Map-based priors for localization. 2004. 11

- [87] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001. 32
- [88] Sakrapeer Paisitkriangkrai, Jamie Sherrah, Pranam Janney, Van-Den Hengel, et al. Effective semantic pixel labelling with convolutional networks and conditional random fields. In *IEEE/ISPRS Workshop: Looking From Above: When Earth Observation Meets Vision*, 2015. 47
- [89] Vishal M Patel, Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Visual domain adaptation: A survey of recent advances. *IEEE Signal Processing Magazine*, 32(3):53–69, 2015. 50
- [90] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 5, 37
- [91] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 5, 36, 37
- [92] Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 35
- [93] Srikumar Ramalingam, Sofien Bouaziz, Peter Sturm, and Matthew Brand. Geolocalization using skylines from omni-images. In *IEEE Workshop on Search in 3D and Video (S3DV)*, 2009. 3, 12
- [94] Franz Rottensteiner, Gunho Sohn, Markus Gerke, and Jan Dirk Wegner. Isprs test project on urban classification and 3d building reconstruction. *Commission III-Photogrammetric Computer Vision and Image Analysis, Working Group III/4-3D Scene Analysis*, pages 1–17, 2013. ix, 56
- [95] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 8, 36
- [96] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C.

- Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015. 23
- [97] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226. Springer, 2010. 5
- [98] Tawfiq Salem, Scott Workman, Menghua Zhai, and Nathan Jacobs. Analyzing human appearance as a cue for dating images. In *WACV*, pages 1–8, 2016. 3
- [99] Tawfiq Salem, Scott Workman, Menghua Zhai, and Nathan Jacobs. Analyzing Human Appearance as a Cue for Dating Images. In *IEEE Winter Conference on Applications of Computer Vision*, 2016. 37
- [100] Grant Schindler and Frank Dellaert. Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 21
- [101] Grant Schindler, Panchapagesan Krishnamurthy, Roberto Lublinerman, Yanxi Liu, and Frank Dellaert. Detecting and matching repeated patterns for automatic geo-tagging in urban environments. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 3, 10, 12
- [102] Steven M Seitz and Simon Baker. Filter flow. In *IEEE International Conference on Computer Vision*, 2009. 47, 48
- [103] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 51
- [104] Hu Sixing, Feng Mengdan, Nguyen Rang M.H., and Lee Gim Hee. Cvm-net: Cross-view matching network for image-based ground-to-aerial geo-localization. In *CVPR*, pages 7258–7267, 2018. 5
- [105] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. *ACM Transactions on Graphics*, 25(3):835–846, 2006. 3, 12
- [106] Kihyuk Sohn, Sifei Liu, Guangyu Zhong, Xiang Yu, Ming-Hsuan Yang, and Manmohan Chandraker. Unsupervised domain adaptation for face recognition in unlabeled videos. In *ICCV*, pages 3210–3218, 2017. 5

- [107] Abby Stylianou, Austin Abrams, and Robert Pless. Finding jane doe: A forensic application of 2d image calibration. *Imaging for Crime Detection and Prevention*, 2013. 2, 10
- [108] Hao Sun, Shuai Liu, Shilin Zhou, and Huanxin Zou. Unsupervised cross-view semantic transfer for remote sensing image classification. *IEEE Geoscience and Remote Sensing Letters*, 13(1):13–17, 2016. 50
- [109] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 38, 39
- [110] Richard Szeliski. *Computer vision: algorithms and applications*. Springer, 2010. 21
- [111] J-P Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *IEEE International Conference on Computer Vision*, 2009. 21, 22
- [112] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data and new challenges in multimedia research. *CoRR*, abs/1503.01817, 2015. 40
- [113] Andrea Vedaldi and Andrew Zisserman. Self-similar sketch. In *European Conference on Computer Vision*, 2012. 21, 22, 30
- [114] Nam N Vo and James Hays. Localizing and orienting street views using overhead imagery. In *European Conference on Computer Vision*, pages 494–509. Springer, 2016. 2, 4, 50
- [115] Anna Volokitin, Radu Timofte, and Luc Van Gool. Deep features or not: Temperature and time prediction in outdoor scenes. In *CVPR Workshop on Robust Features*, 2016. 37
- [116] R Grompone Von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall. Lsd: A fast line segment detector with a false detection control. 32(4):722–732, 2010. 25, 30
- [117] Florian Walch, Caner Hazirbas, Laura Leal-Taixe, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization using lstms for structured feature correlation. In *ICCV*, pages 627–637, 2017. 3

- [118] Jingya Wang, Mohammed Korayem, and David J Crandall. Observing the natural world with flickr. In *IEEE International Conference on Computer Vision Workshops*, 2013. 35
- [119] Xiaolong Wang, Ali Farhadi, and Abhinav Gupta. Actions~ transformations. In *CVPR*, pages 2658–2667, 2016. 5
- [120] Jan D Wegner, Steven Branson, David Hall, Konrad Schindler, and Pietro Perona. Cataloging public objects using aerial and street-level images-urban trees. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6014–6023, 2016. 49
- [121] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, pages 499–515. Springer, 2016. 5
- [122] Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet-photo geolocation with convolutional neural networks. *arXiv preprint arXiv:1602.05314*, 2016. 3, 35, 37
- [123] Horst Wildenauer and Allan Hanbury. Robust camera self-calibration from monocular images of manhattan worlds. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 19, 21
- [124] Wolfram Willuhn and Frank Ade. A rule-based system for house reconstruction from aerial images. In *ICPR*, 1996. 50
- [125] Scott Workman, Connor Greenwell, Menghua Zhai, Ryan Baltenberger, and Nathan Jacobs. Deepfocal: A method for direct focal length estimation. In *ICIP*, 2015. 4
- [126] Scott Workman and Nathan Jacobs. On the location dependence of convolutional neural network features. In *IEEE/ISPRS Workshop: Looking From Above: When Earth Observation Meets Vision*, 2015. 4, 37, 50
- [127] Scott Workman, R. Paul Mihail, and Nathan Jacobs. A pot of gold: Rainbows as a calibration cue. In *European Conference on Computer Vision*, 2014. 3, 12, 19, 35, 37
- [128] Scott Workman, Richard Souvenir, and Nathan Jacobs. Wide-area image geolocalization with aerial reference imagery. In *IEEE International Conference on Computer Vision*, 2015. 4, 48, 50, 53
- [129] Scott Workman, Richard Souvenir, and Nathan Jacobs. Wide-area image geolocalization with aerial reference imagery. In *IEEE International Conference on Computer Vision*, pages 1–9, 2015. 37

- [130] Scott Workman, Menghua Zhai, and Nathan Jacobs. Horizon lines in the wild. In *British Machine Vision Conference (BMVC)*, 2016. 4
- [131] Scott Workman, Menghua Zhai, and Nathan Jacobs. Horizon lines in the wild. *arXiv preprint arXiv:1604.02129*, 2016. 20, 31
- [132] Lin Wu and Xiaochun Cao. Geo-location estimation from two shadow trajectories. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 3, 12
- [133] Yiliang Xu, Sangmin Oh, and Anthony Hoogs. A minimum error vanishing point detection approach for uncalibrated monocular images of man-made environments. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 2, 19, 21, 22, 30
- [134] Xinchun Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *NIPS*, pages 1696–1704, 2016. 8
- [135] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NIPS*, pages 3320–3328, 2014. 5, 23
- [136] Amir Roshan Zamir and Mubarak Shah. Accurate image localization based on google maps street view. In *ECCV*, pages 255–268. Springer, 2010. 4
- [137] Menghua Zhai, Scott Workman, and Nathan Jacobs. Detecting vanishing points using global image context in a non-manhattan world. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4
- [138] Haipeng Zhang, Mohammed Korayem, David J Crandall, and Gretchen LeBuhn. Mining photo-sharing websites to study ecological phenomena. In *International World Wide Web Conference*, 2012. 2, 10
- [139] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, 2016. 5, 37
- [140] Yong-Gang Zhao, Xing Wang, Liang-Bing Feng, George Chen, Tai-Pang Wu, and Chi-Keung Tang. Calculating vanishing points in dual space. *Intelligent Science and Intelligent Data Engineering*, pages 522–530, 2013. 2, 21
- [141] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems*, 2014. 24

- [142] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, pages 2921–2929, 2016. 5
- [143] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 40
- [144] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 8
- [145] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. *arXiv preprint arXiv:1605.03557*, 2016. 5, 49

Vita

Menghua Zhai

Education

2008–2012 B.E. in Telecom. Beijing University of Posts & Telecoms (BUPT)

Appointments

Research Assistant Center for Vis. & Virtual Envs., University of Kentucky
2014–2018 Lexington, KY

Research Assistant Intern NEC Labs
Summer 2017 *Cupertino, CA*

Algorithm Engineer Intern Image Technologies
Summer 2016 *Cambridge, MA*

Teaching Assistant Computer Science Department, University of Kentucky
 Fall 2013–Spring 2014 *Lexington, KY*

Publications

Refereed Conference Publications

- [1] Samuel Schuster, Menghua Zhai, Nathan Jacobs, and Manmohan Chandraker. Learning to look around objects for top-view representations of outdoor scenes. In *European Conference on Computer Vision (ECCV)*, 2018.

- [2] Menghua Zhai, Tawfiq Salem, Connor Greenwell, Scott Workman, Robert Pless, and Nathan Jacobs. Learning geo-temporal image features. In *British Machine Vision Conference (BMVC)*, 2018.
- [3] Tawfiq Salem, Menghua Zhai, Scott Workman, and Nathan Jacobs. A multimodal approach to mapping soundscapes. In *IEEE Geoscience and Remote Sensing Society (IGARSS)*, 2018.
- [4] Scott Workman, Menghua Zhai, David J. Crandall, and Nathan Jacobs. A unified model for near and remote sensing. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.
- [5] Menghua Zhai, Zachary Bessinger, Scott Workman, and Nathan Jacobs. Predicting ground-level scene layout from aerial imagery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [6] Scott Workman, Menghua Zhai, and Nathan Jacobs. Horizon lines in the wild. In *British Machine Vision Conference (BMVC)*, 2016.
- [7] N. Jacobs, S. Workman, and M. Zhai. Cross-view convolutional networks. In *2016 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–5, 2016.
- [8] Menghua Zhai, Scott Workman, and Nathan Jacobs. Camera geo-calibration using an mcmc approach. In *IEEE International Conference on Image Processing (ICIP)*, 2016.
- [9] Menghua Zhai, Scott Workman, and Nathan Jacobs. Detecting vanishing points using global image context in a non-manhattan world. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [10] Ryan Baltenberger, Menghua Zhai, Connor Greenwell, Scott Workman, and Nathan Jacobs. A fast method for estimating transient scene properties. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.
- [11] Tawfiq Salem, Scott Workman, Menghua Zhai, and Nathan Jacobs. Analyzing human appearance as a cue for dating images. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.
- [12] Scott Workman, Connor Greenwell, Menghua Zhai, Ryan Baltenberger, and Nathan Jacobs. Deepfocal: A method for direct focal length estimation. In *IEEE International Conference on Image Processing (ICIP)*, 2015.

- [13] Feiyu Shi, Menghua Zhai, Drew Duncan, and Nathan Jacobs. MPCA: EM-based PCA for mixed-size image datasets. In *IEEE International Conference on Image Processing (ICIP)*, pages 1807–1811, 2014.
- [14] Menghua Zhai, Feiyu Shi, Drew Duncan, and Nathan Jacobs. Covariance-based PCA for multi-size data. In *International Conference on Pattern Recognition (ICPR)*, 2014.

Honors and Awards

- Thaddeus B. Curtz Memorial Scholarship, University of Kentucky, 2017
- Verizon Fellowship, University of Kentucky, 2012–2013
- Academic Scholarship, Beijing University of Posts & Telecoms, 2008–2011

Service

- Reviewing for IEEE Computer Vision and Pattern Recognition (CVPR), 2018