Coherent Scene Understanding with 3D Geometric Reasoning

Jiyan Pan

CMU-RI-TR-14-06

April 2014

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Thesis Committee:
Takeo Kanade, *Co-Chair*
Martial Hebert, *Co-Chair*
Yaser Sheikh
Derek Hoiem, *University of Illinois at Urbana-Champaign*

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.

Carnegie Mellon

Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

Thesis

Coherent Scene Understanding with 3D Geometric Reasoning

Jiyan Pan

Submitted in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
in the field of Robotics

ACCEPTED:

Takeo Kanade	May 9, 2014
Thesis Committee Co-chair	Date
Martial Hebert	May 9, 2014
Thesis Committee Co-chair	Date
Reid G. Simmons	May 15, 2014
Program Chair	Date

APPROVED:

Randal E. Bryant	May 15, 2014
Dean	Date

Abstract

When looking at a single 2D image of a scene, humans could effortlessly understand the 3D world behind the scene even though stereo and motion cues are not available. Due to this remarkable human capability, one of the ultimate goals of computer vision is to enable machines to automatically infer the 3D structure of a scene given a single 2D image. This dissertation proposes methods that produce a geometrically and semantically coherent 3D interpretation of urban scenes from a single image, and shows the benefits of reasoning in 3D when analyzing 2D images.

In this dissertation, we model an urban scene using three types of elements. The first type is global geometries such as ground plane and gravity direction. The second type is objects such as cars and pedestrians that have definitive shapes and extents. The third type is vertical surfaces such as building facades that do not have definitive shapes and extents. Such a modeling allows for a richer characterization of an urban scene than existing works.

To tackle the inherent ambiguity involved in recovering the 3D structure from a single 2D image, we systematically identify geometric constraints among the three types of elements in our model, and encode such constraints in a Conditional Random Field (CRF). For objects, we consider both their global geometric compatibility with ground plane and gravity direction, and their local geometric compatibility between adjacent objects. For building facades, we decompose them into a set of continuously-oriented planes mutually related by 3D geometric relationships, and constrained by nearby objects in 3D. We also propose a generalized RANSAC algorithm to make the inference of the model tractable. We show that performing 3D geometric reasoning using our model benefits individual tasks such as object detection, viewpoint estimation, and facade layout recovery. In addition, it yields a more informative interpretation of the 3D scene behind the image.

Acknowledgments

I am eternally grateful to my advisor Takeo Kanade for years of invaluable guidance on my research. He has granted me great freedom to explore and pursue various research topics and ideas I am passionate about, while keeping me on the right track all the time by providing me with both a clear picture of high-level directions and many key insights into specific algorithms. Moreover, Takeo has been imparting on me the powerful philosophies of conducting great research by striking subtle balances, and giving effective presentations that could capture audience's attention. Those philosophies will benefit me for life.

I also want to express my most profound gratitude to my co-advisor Martial Hebert who has given me crucial guidance and tremendous help that has considerably facilitated my research progress. I have constantly received inspirations from Martial on how to tackle bottlenecks from brand new angles, and how to properly perceive and evaluate my work against the vast sea of relevant literature. My research process would have been much bumpier had it not been for Martial.

I am very much obliged to Yaser Sheikh, Derek Hoiem, and Eric Xing who have shared with me their deep insights into my thesis research and offered me powerful ideas and suggestions. I am also truly thankful to Mei Chen for being my mentor and giving me invaluable advice and huge encouragements both during and beyond my internship with Intel Labs Pittsburgh. My special thanks also go to Henry Schneiderman, Sharath Pankanti, Quanfu Fan, Pavel Berkhin, and Ye Chen, who have substantially broadened my vision about conducting cutting-edge research in industry.

Over the past several years, I have been truly fortunate to work with so many amazing researchers at CMU. I am really grateful to Daniel Huber, Kris Kitani, Seung-il Huh, Zhaozheng Yin, Kang Li, Lee Weiss, Phil Campbell, Abhinav Gupta, Yan Li, Yuandong Tian, Henry Kang, Scott Satkin, Myung Hwangbo, Daniel Munoz, and David Lee, for all the beneficial discussions and help they have provided me. I also owe much to Suzanne Muth, Suzette Gambone, and Yukiko Kano for their great support for so long.

Finally, I sincerely thank my family for giving me unconditional love, encouragement and support that means so much to me.

Contents

List of Figures

Chapter 1

Introduction

One of the key features of human visual intelligence is being able to recover the underlying 3D structure of a scene from a single image. For example, given a monocular visual input of an urban scene, we are able to acquire a comprehensive situational awareness, in which not only do we know if there are cars, pedestrians or building facades present in the scene, but we also understand how they are supported by the ground plane, how their locations and poses are influenced by the gravity direction, how they are located relative to one another in 3D space, and how we as observers are positioned with respect to the entire scene. The ability to achieve such a coherent 3D understanding of a scene is crucial to many robotics applications such as vision-based navigation, planning, and manipulation [61, 62], as well as higher-level visual intelligence tasks such as activity recognition and behavior understanding [17, 26, 69].

Not only does reasoning about the 3D scene behind the image provide important information that traditional 2D algorithms such as object detection cannot provide, it also greatly improves the performance of those traditional tasks as a result of resolving visual ambiguities. For example, when we look at the two image patches shown in the lower-left corner of Figure 1.1, it is hard to tell what is contained in the green box, yet it is likely that the red box encloses a pedestrian dressed in black. However, when we look at the entire image and obtain a sense of the 3D scene layout behind the image, we can tell for sure that the green box actually contains a car, since

Figure 1.1: 3D Geometric reasoning explains away visual ambiguities.

it rests on the road with the right pose, size, and location. Also, the red box cannot contain a pedestrian, since our sense of 3D geometry tells us that the content in the red box would have been too short for a pedestrian. In fact, our subconscious thought process of inducing global geometric context from local entities and utilizing it to regulate what we perceive to be present in the scene is crucial to the understanding of a complex scene where visual ambiguities abound.

1.1 Problem

The problem we address in this dissertation is that, given a single 2D image of an urban scene, design an automated system to recover the 3D structure behind the scene and the camera viewpoint with respect to the scene. More specifically, as is illustrated by the output of our work in Figure 1.2, the system estimates the gravity direction (indicated by the magenta horizon line), infers the ground plane orientation and distance from the camera (indicated by the yellow grid), detects objects such as cars and pedestrians (indicated by the green/red bounding boxes), estimates the 3D layout of those objects (indicated by the rectangles and dots in the top-down views),

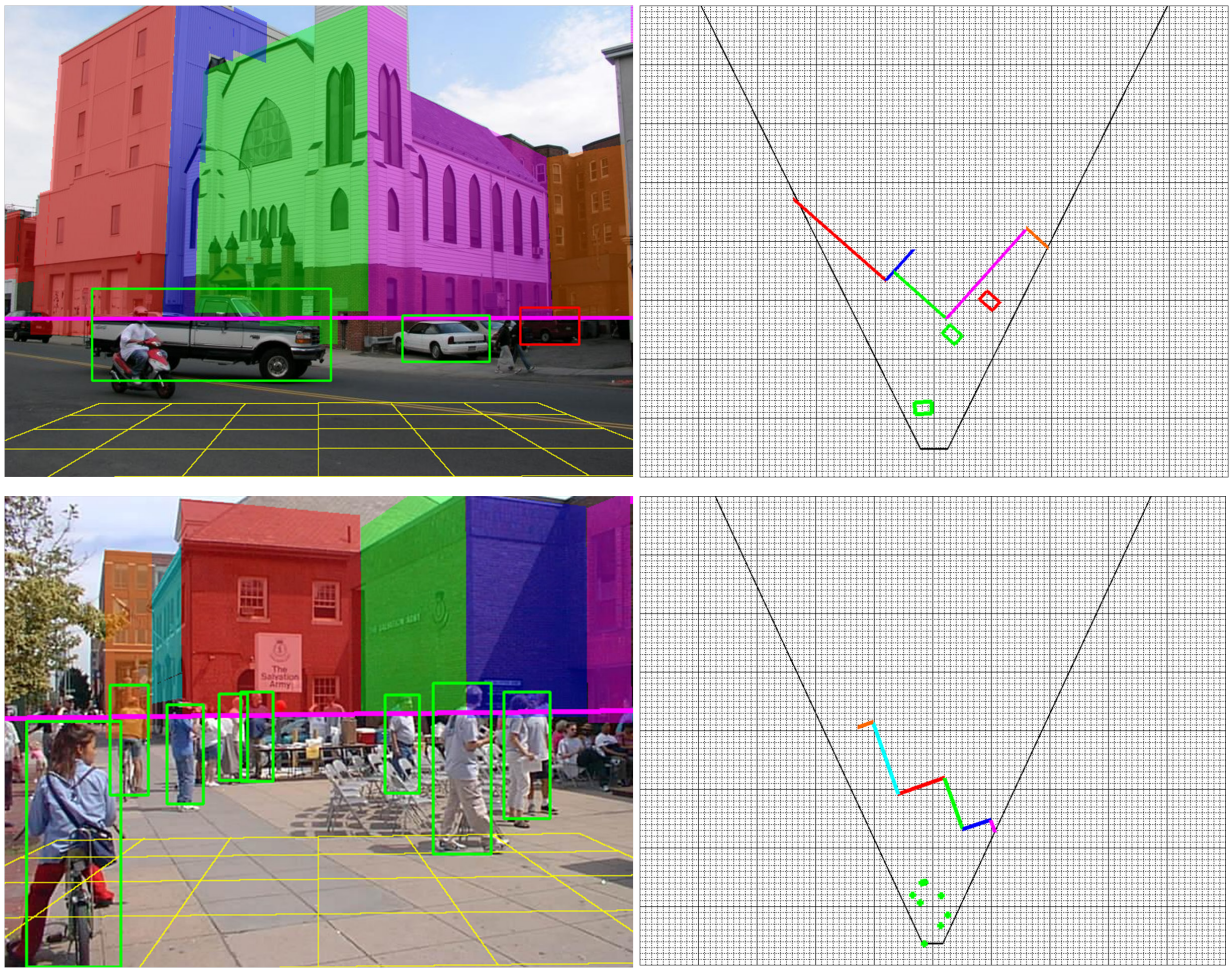

Figure 1.2: Two examples of the output of our scene understanding system. Given a single 2D image of an urban scene, our system recovers an object/planar-level 3D structure of the scene as well as the camera viewpoint. Here, the left column displays the viewpoint estimation result, object detection result, and facade detection/decomposition result, where the magenta horizon line indicates gravity direction, the yellow grid (whose spacing is 1 meter) indicates the ground plane, the green/red bounding boxes indicate detected objects, and the color-shaded regions indicate distinctive building facades. The right column displays the system-generated top-down view of the scene. The thick black lines indicate the viewing boundaries, the green/red rectangles indicate detected cars, the green/red dots indicate detected pedestrians, and the colored lines indicate distinctive building facades. The spacing is 1 meter for the fine (thin) black grid, and 10 meters for the coarse (thick) black grid.

segments out building regions and decomposes them into distinctive facade planes (indicated by the color-shaded regions), and infers the 3D layout of those building facades (indicated by the colored lines in the top-down views).

Note that we are not trying to generate a detailed 3D point cloud of the scene [58, 59, 60], but an object/planar-level approximation of the 3D scene. In fact, psychological experiments on human subjects show that humans are poor at estimating the relative depth of disconnected points or surfaces, but good at indicating surface orientations and making slant judgments [41, 42, 78]. In other words, humans reason about and make sense of visual input not in terms of 3D point cloud, but in terms of higher-level entities such as surfaces and objects. Also, for most applications such as vision-based autonomous navigation, such a level of approximation provides the most critical information (*e.g.* will I bump into a car or run into a wall if I continue to walk this way?) while discarding less relevant details such as the exact 3D shape of an object or the fine structures of a building facade.

Recovering an object/planar-level approximation of the 3D scene has been investigated in the works of Hoiem *et al.* [31, 34], Gupta *et al.* [25], Bao *et al.* [3, 66], and Gould *et al.* [23], *etc.*, where many individual 3D reasoning components (such as estimating 3D object/surface orientation, inferring spatial layout, and estimating camera viewpoint) have been studied, and some 3D reasoning systems under strong assumptions have been proposed. However, it remains a challenging problem to systematically integrate various sources of 3D geometric information by quantitatively modeling different types of geometric constraints in a more general setting, and to perform tractable reasoning and inference over all those inter-related geometric variables. This dissertation aims at addressing this problem.

1.2 Challenges

1.2.1 Resolving Ambiguity

The task of recovering 3D structure from a single image itself is an especially challenging one regardless of the level of approximation. The root cause stems from the loss of depth information during the image forming process. An infinite number of different 3D structures could be projected to form exactly the same 2D image. Without resorting to the prior knowledge of the world (*e.g.* 3D geometric constraints among objects and facades), it is mathematically impossible to recover the 3D structure of a scene from a single 2D image.

1.2.2 Estimating Global 3D Geometries

Global 3D geometries such as gravity direction and ground plane location play a critical role in imposing 3D geometric constraints, because they set up a stage for the reasoning over local entities such as objects and facades. However, with just a single image, how do we obtain the global 3D geometries? If we use local entities to estimate the global 3D geometries, we must make sure invalid entities (*e.g.* false detections returned by an object detector) do not corrupt the estimation. On the other hand, validating local entities requires the geometric setup provided by the global 3D geometries. How do we solve this chicken-and-egg problem?

1.2.3 Dealing with Building Facades

Unlike objects, building facades usually do not have a definitive shape, and could have an arbitrary extent. While segmenting out facade regions in a 2D image has enjoyed a relatively high level of success [51], locating those building regions in 3D remains under-studied. A possible reason is that the lack of a definitive shape and size, together with their huge variety of 3D structures, makes building facades particularly hard to model geometrically, and therefore even harder to estimate their 3D locations. For example, from the size of an object in a 2D image and the

prior knowledge of its real-world size, we could get a clue of how far it is from the camera. Building facades, however, could have a large variety of heights, making its depth estimation highly ambiguous. In addition, unlike objects that are usually in the foreground with visible ground contact, building facades are typically in the background and hence their ground contact lines are often occluded, undermining a strong cue of depth estimation. How do we perform geometric reasoning over building facades when facing these difficulties?

1.2.4 Handling Uncertainty in 3D Geometric Reasoning

In order to perform 3D geometric reasoning, we need to formally represent the 3D geometric relationships among gravity/ground, objects, and facades, as well as the relationships between 2D shapes in the image and their corresponding 3D geometries. Theoretically speaking, such relationships are deterministic. However, as we can only observe 2D patterns in the image and cannot directly measure 3D geometric properties, most of the 3D geometric properties have to be inferred with some degree of uncertainty. In addition, many entities in the image that produce 3D geometric relationships are invalid (*e.g.* due to false detections), leading to distractive invalid relationships that add to even more uncertainties. How do we reason over a complex set of 3D geometric constraints with a high level of uncertainty?

1.2.5 Maintaining Tractability

In our modeling of 3D urban scenes, we allow for a higher degree of flexibility. For example, gravity direction and ground plane orientation could independently take any direction as long as they are within 45 degrees from the camera's vertical axis; orientations of facade planes could take a continuous range of directions instead of being classified into several coarse categories. When performing probabilistic reasoning, those continuous variables need to be quantized. However, due to the combination of multiple continuous variables, even a reasonably fine quantization would result in an exponentially large search space. Even with state-of-the-art inference algo-

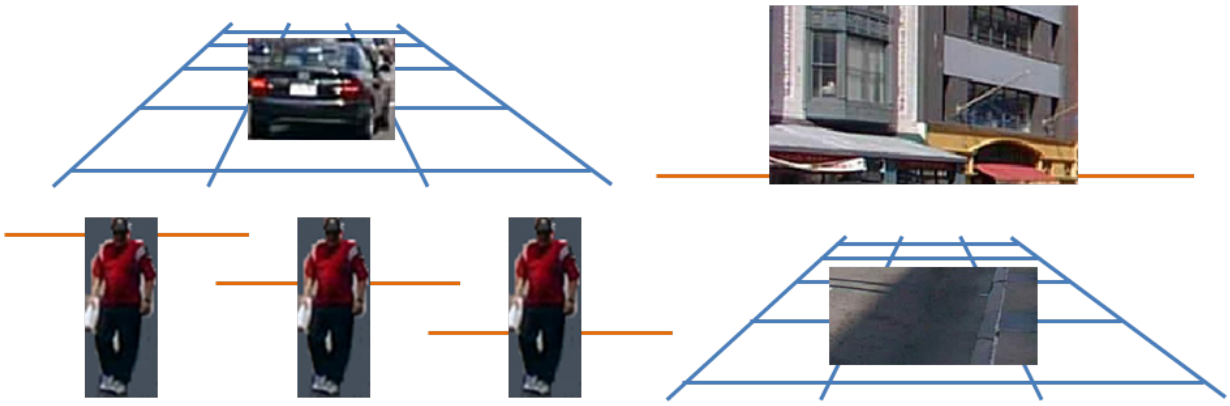

Figure 1.3: Objects and surfaces give cues about the 3D environment it resides in. A car would suggest ground orientation and height, a building facade would suggest gravity direction, a pedestrian would suggest a range of possible gravity directions, and a road surface would suggest ground orientation.

rithms, the space and computational complexity would still become intractable. How do we maintain tractability while still retaining accuracy?

1.3 Our Approach

The core idea of our 3D geometric reasoning approach draws inspiration from how humans perform scene understanding. When we look at an image patch that contains an object, such as the one shown in Figure 1.3, not only do we recognize what it is, but we also form a global 3D geometry hypothesis in our mind from the local appearance of the object. Each object we observe in an image either strengthens or weakens the global geometry hypothesis with its own cues about the 3D environment. The global geometry hypothesis induced by the objects, on the other hand, in turn serves as a 3D geometric context for all the objects we detect. The belief of the existence of an object consistent with the 3D geometric context is reinforced, while the confidence about those inconsistent detections is suppressed. The same applies to surface regions as well.

While the core idea is conceptually simple, we could see the inherent chicken-and-egg problem mentioned in the previous section – invalid and inaccurate local detections should be re-

moved to obtain a good estimate of the global 3D geometries, while removing those bad detections requires a good estimate of the global 3D geometries. Generally, a RANSAC algorithm [16] could serve to solve this problem by 1) randomly selecting a minimal set of local entities that could determine a global 3D geometry hypothesis, and 2) evaluating each hypothesis using the support from local entities to select the best hypothesis. Note that in our case, the minimal set of local entities is just a single object/surface. As the estimate from a single object/surface is usually inaccurate or even invalid, and the number of objects/surfaces is limited, it is likely none of the hypotheses generated by a conventional RANSAC process would be good. To overcome this difficulty, we propose a generalized RANSAC algorithm, in which a global 3D geometry hypothesis is obtained by pooling together the estimates from all individual local entities *with a set of random weights* assigned to them. When there is a sufficient number of such random combinations, at least one of the hypotheses would be generated mostly by *multiple valid* local entities. Such a hypothesis would be of a better quality, because it simultaneously suppresses corruption from invalid local estimates while reducing noise from valid yet inaccurate local estimates.

In addition to generating good hypotheses, another key to the effectiveness of our approach is how to design a hypothesis evaluation system that identifies, encodes and imposes 3D geometric constraints both between global 3D geometries and individual local entities, and among local entities themselves. We systematically examine the geometric relationships among global and local variables in a general setting. For an individual object, its geometric relationship with global 3D geometries is constrained by its 2D locations, 2D landmarks, 3D pose, and real-world height. For a pair of adjacent objects, they are mutually constrained by occlusion ordering and space occupancy. To deal with building facades, we decompose them into a set of distinctive facade planes with different orientations. As we will show in this dissertation, such an approximation of building facades by a set of planes offers a good balance between flexibility and regularity. For an individual facade plane, it is constrained by global 3D geometries via its apparent ground contact line, semantic segmentation, real-world height, and 3D orientation. For a pair of adjacent

facade planes, they are mutually constrained by convex corner, occlusion ordering, and parallel alignment. For an object and a facade plane that are potentially interacting, they are mutually constrained by occlusion ordering. We construct a Conditional Random Field (CRF) [44] to represent all the interacting relationships in the graph structure, and encode the corresponding geometric constraint strengths in the unary and pairwise potentials.

Note that since we adopt a generalized RANSAC approach, the CRF inference is performed under a given hypothesis of global 3D geometries. As a result, all the interactions involving global 3D geometric variables are absorbed into the unary potentials of individual local entities. Since the global 3D geometric variables are all continuous vectors that would have generated pairwise potentials containing a huge number of elements, removing them from the nodes of the CRF makes the inference tractable.

1.4 Datasets

In this dissertation, we use three datasets that contain urban scenes to evaluate our approach and compare with competing algorithms.

1.4.1 LabelMe

We use the dataset compiled by Hoiem *et al.* [34] when the evaluation of object detection and viewpoint estimation are involved. This dataset contains 422 outdoor images from the LabelMe dataset [57]. Containing 923 cars and 720 pedestrians in total, the dataset covers a multitude of outdoor urban scenes and include a wide variety of object pose and size, making the dataset very challenging. The annotation of the dataset includes bounding boxes for cars and pedestrians. Heavily occluded objects are not annotated. In addition, 100 of the 422 test images are annotated with the horizon position. Hoiem *et al.* also collects a training/validation dataset containing 51 images.

To allow for a finer geometric representation of an object than just a bounding box, we provide additional annotations of object landmarks. For cars, we manually mark in the image the 2D locations of the three visible ground-contacting points of wheels and the top point of the wheel closest to the camera. Wheel height is the distance between the top and bottom landmarks of the wheel closest to the camera. For pedestrians, we mark the 2D locations of head and foot in the image (*i.e.* top and bottom landmarks). Pedestrian height is the distance between the two landmarks. Please see Figure 1.4 for examples. We train a landmark regressor on the training/validation set to predict the top and bottom landmarks of an object given its 2D image features.

From those landmarks, we automatically compute the pose of cars and pedestrians with respect to the camera. Given the bottom landmark and the horizon, we compute the pitch and roll angles of objects. For cars, we also compute their yaw angles from the three ground-contacting landmarks. Please see Figure 3.1 for the definition of pitch, roll, and yaw angles. We use these angles to train a pose regressor on the training/validation set that estimates object pose given its 2D image features.

In addition to evaluating object detection and viewpoint estimation, we also use this dataset for a qualitative evaluation of facade decomposition and localization, as well as a quantitative evaluation of joint reasoning with both objects and facades. Among the 422 test images, 225 of them contain building facades, and this subset is used for the evaluation involving building facades.

1.4.2 Geometric Context

We use the Geometric Context dataset [33] to evaluate facade decomposition. The dataset contains 300 outdoor images, 250 for testing and 50 for training/validation. The images in the dataset are segmented and annotated with seven surface layout labels – “support”, “sky”, “planar left”, “planar center”, “planar right”, “non-planar porous”, “non-planar solid”. Here, “support”class is

Figure 1.4: Examples of the LableMe dataset with annotations of object bounding boxes, object landmarks, and horizon. Here, green rectangles are object bounding boxes; dots with different colors are object landmarks, with yellow lines highlighting the frame of a car defined by its landmarks; and the magenta line is the horizon.

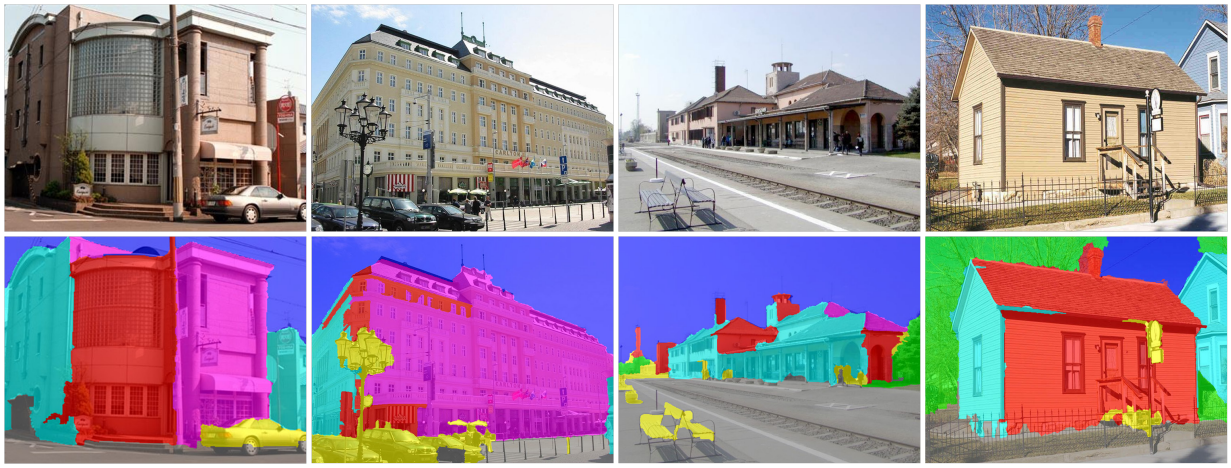

Figure 1.5: Examples of the Geometric Context dataset with annotations of surface layout labels. The color code is as follows. Magenta – planar right; cyan – planar left; red – planar center; green – non-planar porous; yellow – non-planar solid; blue – sky; grey – support.

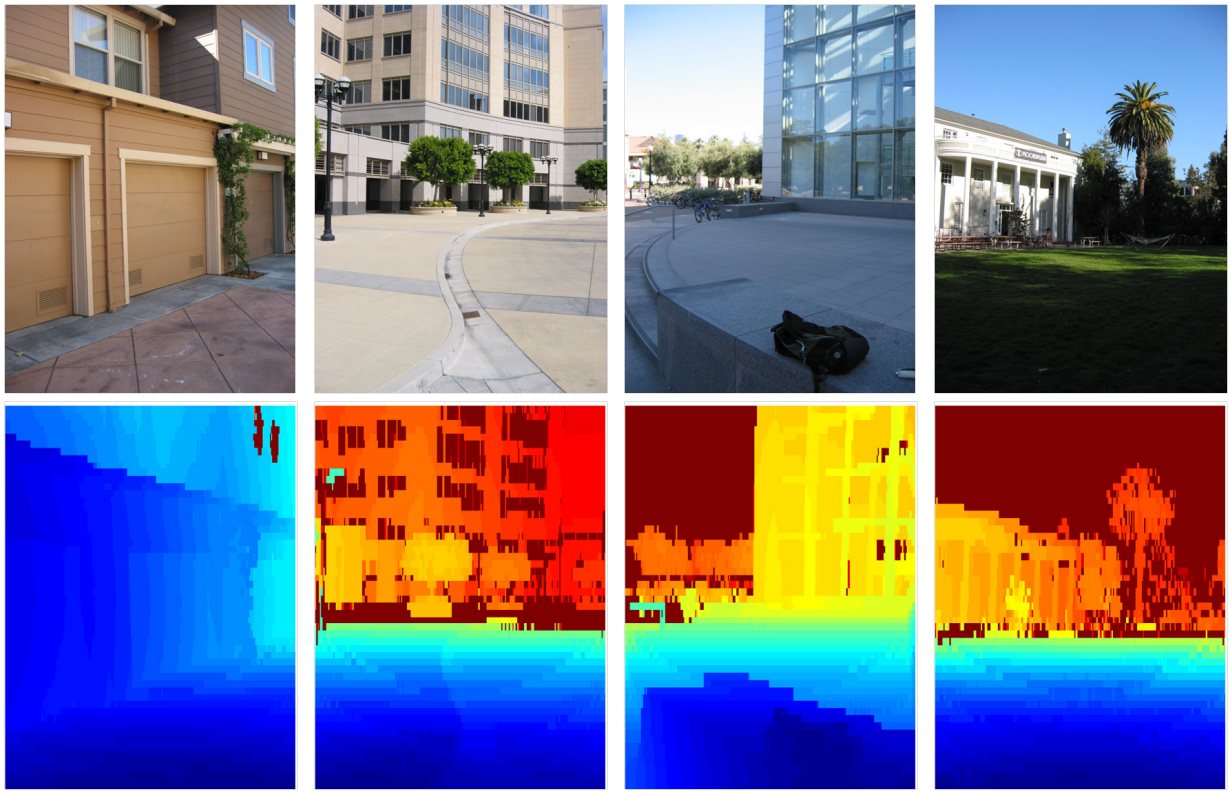

Figure 1.6: Examples of the Make3D dataset with annotations of depth maps acquired using a laser scanner. Here, depth value increases from blue to red. Maximum depth is capped at 80m. Note that, as some windows cannot reflect laser back to the scanner, their depth measurement is incorrect.

mostly located on the ground region; “planar left”, “planar center”, and “planar right”classes are mostly located on building facade regions; “non-planar porous”class is mostly located on tree or plantation regions; “non-planar solid”class is mostly located on foreground objects such as cars, pedestrians, or trunks.

Since the dataset comes with the ground-truth annotation of a coarse categorization of planar surface orientation, we could quantize our continuous estimation of facade orientation into “left”, “center”, and “right”classes and compare against the ground-truth and other competing algorithms that estimate surface layout. As our approach focuses on building facades, we collect among the 250 test images all the 55 images that contain building facades, and use these images for evaluation. Please see Figure 1.5 for examples.

1.4.3 Make3D

To perform a quantitative evaluation of facade localization, we perform experiments on the Make3D dataset of Saxena *et al.* [48, 60]. The dataset contains 534 images of urban and natural scenes, among which 400 images are for training and the remaining 134 images are for testing. In addition to the ground truth of semantic segmentation, the Make3D dataset also provides the ground truth of depth-maps acquired using a laser scanner. Those depth-maps make it possible for us to evaluate the performance of our facade reasoning algorithm in estimating the absolute depths of facades, complementing the evaluation of orientation-based decomposition performance using the Geometric Context dataset [33]. Among the 134 testing images, 51 images contain building facades. As none of them contains any objects, those images are only used to evaluate facade reasoning. Please see Figure 1.6 for examples.

1.5 Contributions

The main contribution of this dissertation is a 3D geometric reasoning system that is able to coherently recover the camera viewpoint and an optimal 3D layout of objects and building facades given a single 2D image of an urban scene. We achieve this by integrating multiple global and local 3D geometries from gravity, ground plane, objects and building facades with a set of geometric constraints, and perform probabilistic reasoning over a multitude of geometric and validity variables using those constraints. More specifically, our contributions include:

1) We systematically identify 3D geometric relationships among global and local entities with richer representations and less assumptions. For example, we use both pitch and roll angles as well as landmark locations to estimate the orientation and distance of an object, as opposed to using bounding box heights and locations in [31, 34], or pitch angles and bounding box areas in [3, 66]. Such a characterization allows for an unrestricted orientation of objects and a finer estimate of object dimension. We also use a continuous

3D vector instead of a coarse category (as in [31, 34]) to represent facade orientation. Besides, building facades are composed of a set of mutually constrained planes instead of just blocks as in [25]. Camera could have both pitch and roll movements with respect to gravity and ground, and gravity direction does not have to be perpendicular to the ground.

2) We develop a generalized-RANSAC-CRF approach to make inference tractable for both the optimal values of global 3D geometries (*i.e.* gravity and ground) and the optimal layout of local objects and building facades. In doing so, we circumvent exhaustive search [3] and avoid inference over a probabilistic graphical model (PGM) [31, 34] involving huge potential tables. In addition, unlike the Hough voting method in [66], out approach is able to simultaneously suppress outliers from invalid local entities and reduce noise from inaccurate local entities.

3) We develop a facade decomposition and localization algorithm based on 3D geometric reasoning over mutually constrained planes. Unlike the superpixel-based methods in [35] or [48], our approach reasons over combinations of planes, and thus yields a greater coherence and a higher-level understanding of the scene. On the other hand, unlike the block-based method in [25], our plane-based approach is more flexible in representing building facades of higher complexity, and is able to return a numeric parameterization of facade planes instead of just a qualitative modeling of the facades. We also propose a quadrilateral-based sampling algorithm to effectively detect candidate facade planes and make plane-based facade reasoning tractable. To our knowledge, we are among the first to propose an algorithm that is able to generate a plane-based quantitative 3D layout of building facades with continuous orientations.

4) Different from the works in [34] or [3] that assume local entities are conditional independent given global 3D geometries, we consider the interactions among local entities in addition to the constraints between global 3D geometries and individual local entities. Capturing the interactions among objects and facades by our approach improves object

detection, 3D layout recovery, and viewpoint estimation. In addition, to our knowledge, there are few works in the literature that perform 3D geometric reasoning involving all three elements – gravity/ground, objects, and building facades.

Chapter 2

Related Work

In this chapter, we review some existing works in the literature that are related to our approach presented in this dissertation. While there are few existing works that address exactly the same problem as defined in Section 1.1, our work is inspired by researchers working on related problems, and takes advantage of existing powerful algorithms that produce input features to our system.

2.1 Coherence from 2D Context

Researchers have long been aware of the importance of contextual cues in improving the performance of various computer vision tasks by imposing common-sense coherence in the output. The type of contextual cues relatively easier to handle are 2D ones. For example, a cow is usually surrounded by the grass region in an image, a pedestrian is expected to be found above the ground region in an image, and sky is expected to be located in the upper part of an image, etc. As such 2D contextual cues in the image domain could resolve visual ambiguities that are hard to deal with by stand-alone object detectors or image segmentors, many researchers have explored different types of 2D context, including scene-object relationship [5, 47, 52], region-object relationship [28], object-object relationship [5, 53], neighborhood-object relationship [2],

region-region relationship [65], object spatial prior in the image [11], *etc.* The relative usefulness of some of the 2D contextual relationships are investigated by Zheng *et al.* in [77]. While 2D context is helpful in many cases, it is still highly limited, because the essential relationships among objects and regions exist in the 3D world instead of the 2D projection of the world. For instance, in Figure 2.1, humans could immediate tell two vehicles in the image are incoherent with the scene due to their implausibility in the 3D world. Nevertheless, those two vehicles do not violate any 2D contextual relationships and are thus unable to be spotted by algorithms utilizing 2D context alone. Therefore, directly leveraging 3D contextual information is indispensable for a better coherent scene understanding.

2.2 Depth Map Estimation

There have been a significant amount of research focusing on directly measuring the 3D structure of the scene using multi-view stereo [40, 50, 67, 70] and/or active depth sensing techniques like LIDAR [12, 21, 37, 39, 63]. While these approaches are quite robust and widely used in robotics applications, they have important limitations. For example, they are effective only when the range is relatively small (typically within dozens of meters), and they are not applicable to understanding a single image posted on a wall or displayed on a screen where 3D information has to be *inferred* instead of being *measured*.

Saxena *et al.* did pioneering research on inferring a dense depth map from a single still image [59, 60]. They use low-level image features within and between superpixels to reconstruct the depth and 3D orientation of each superpixel. As their features and MRF models are completely based upon superpixels, they do not involve object detection and therefore do not utilize object-level information which provides strong cues for the layout of a scene. The authors made a simple extension of their model to incorporate object information in [58] and reported improved results on depth estimation. However, in their approach, 3D scene layout does not affect object detection. Consequently, harmful effects from false detections cannot be avoided. Also, only

Figure 2.1: An illustration of how 2D context alone is insufficient in detecting incoherence within a scene. The two vehicles that are evidently incompatible with the scene do not violate any 2D contextual relationships.

object location is utilized in their method, although object pose is equally important.

A more sophisticated system that produces a dense depth map is developed by Gould *et al.* [22, 48] in which semantic information is used to train a separate depth prediction model for each semantic class, and object location is utilized as geometric constraints for depth estimation. The authors show that incorporating object semantics reduces depth estimation error. Nevertheless, depth estimation and scene layout recovery is done as an independent stage after object detection is finalized [22]. As a result, object detection receives little benefit from scene layout estimation. Although horizon location in the image is involved when forming regions and objects [22, 23, 24], it only produces a feature for the classifier, and the authors do not seek to explicitly model and utilize global 3D geometries.

2.3 3D Scene Layout Recovery

To understand the 3D structure behind an urban scene in an image, estimating an absolute depth for each pixel is not critical. What really matters is the ability to reason over higher-level elements such as gravity/ground, local objects and facade planes, or, in other words, to recover the 3D layout of a scene. To achieve this purpose, we need to explicitly model global 3D geometries (*i.e.* gravity and ground), and make them interact with the detection and localization of local entities, so that they could mutually benefit each other. To solve the chick-and-egg problem inherent in this process (as is mentioned in Section 1.2), several approaches have been proposed. The first approach is to employ a Probabilistic Graphical Model (PGM) that takes both global and local geometries as its nodes whose values are to be inferred from the graph, as is done in the ground-breaking work by Hoiem *et al.* [31, 34]. In [31, 34], the authors build a tree-structured Bayes net in which local detections are attached to the common global geometric parameters. Inference over the Bayes net produces the optimal configuration for both the (quantized) global geometric parameters and the validity of each local detection. The authors show that 3D reasoning greatly improves object detection, while the object evidence also improves horizon estimation [31, 34].

As a first attempt to construct such a reasoning system, Hoiem's work nevertheless has several limitations. For example, global 3D geometries are simplified as the row index of the horizon line and a camera height. Such a simplification prohibits the camera to have any roll angle with respect to the ground plane. Also, gravity direction is assumed to be always identical to the ground plane orientation. When these assumptions are relaxed, the search space would be significantly enlarged, and the number of entries in the resulting pairwise potential tables would be intractably huge even with a moderately fine quantization of gravity direction and ground plane parameters. In addition to the tractability issue in a more general setting, the simplified geometric relationships mentioned in [31, 34] are effective only when camera pitch is small. Another limitation is that the height of a bounding box is assumed to be the height of an object in the image, which is often not true especially for objects like cars. Also, object pose is not

considered in the algorithm.

Instead of optimizing the ground plane parameters in a PGM, Bao *et al.* propose to enumerate all possible (quantized) parameterizations of the ground plane within a predefined range [3]. This approach removes some of the limitations of [31, 34]. For example, the camera could have a non-zero roll angle with respect to the ground plane, and relative pitch angle is introduced to characterize the local geometry of object candidates. However, as the algorithm conducts an exhaustive search for the optimal ground plane in a combinatorial space, the search has to be confined in a narrow range, and/or the quantization has to be relatively coarse.

An improved method is presented in [66], where all object candidates cast votes for the ground plane parameters in a discretized Hough voting space, and the peak in the voting space is regarded as the optimal ground plane. Although this method circumvents both the PGM inference and exhaustive search of the ground plane, it is not able to prevent false detections from corrupting the votes. While the authors employ an EM-like iterative loop to alleviate this problem, yet it is susceptible to local minimum – if the estimate of the ground plane in the first loop is qualitatively wrong, it has little chance to recover in subsequent loops.

While performing 3D geometric reasoning, all of the methods mentioned in this section assume that local entities are conditionally independent of each other given the global 3D geometries, and thus only check the compatibility of local entities with respect to the global 3D geometries. Such a compatibility check could identify object candidates 1 and 2 in Figure 2.2 as false detections. However, even if two adjacent local entities are both compatible with the global 3D geometries, they may still not be able to co-exist due to the violation of local 3D geometric constraints such as depth ordering (candidates 3 and 4 in Figure 2.2) and/or space occupancy (candidates 5 and 6 in Figure 2.2). Considering these local 3D geometric constraints is important in removing many of the false detections produced by an off-the-shelf object detector. Also note that none of the methods mentioned in this section attempts to recover the 3D layout of building facades.

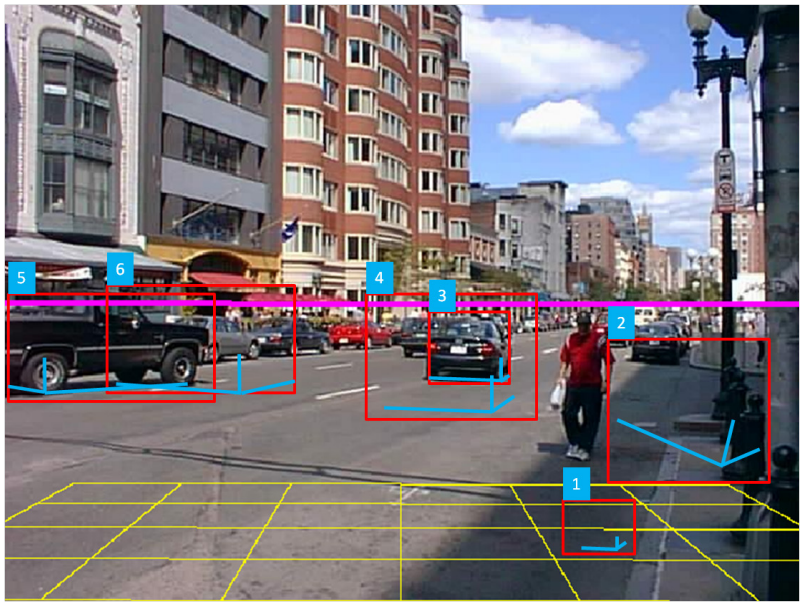

Figure 2.2: Given the ground plane represented by the yellow grid, car candidates 1 and 2 are incompatible with the global 3D geometries due to their implausible size and pose (illustrated by the blue frames). Candidates 3 and 4 cannot co-exist due to their conflict in depth ordering. Candidates 5 and 6 cannot co-exist due to their conflict in space occupancy.

Figure 2.3: An urban scene in which building facades cannot be modeled by blocks (since the Manhattan world assumption does not hold). In addition, while the red and blue regions are adjacent in the image, their corresponding facade planes do not form a connected structure in 3D.

2.4 3D Facade Reasoning

While there are few existing works in the literature that perform 3D geometric reasoning over both objects and facades, many researchers have been working on recovering facade geometries.

Pioneering works on modeling facade geometries, such as Geometric Context proposed by Hoiem *et al.*, focus on classifying super-pixels into different orientation labels (*e.g.* planar left, planar center, or planar right) [23, 33, 35, 36]. While this approach does not produce higher-level concepts such as planes and blocks, it generates useful cues of coarse surface orientations. Using such cues, Gupta *et al.* assemble blocks from two segments with different vertical orientations, and locate those blocks by fitting their ground and sky contact lines [25]. This method generates a rich high-level interpretation of the scene, yet the interpretation is qualitative both in facade orientation and depth. Besides, approximating building facades by blocks cannot model more complex cases such as the one shown in Figure 2.3. In addition, this method makes no attempt to recover camera viewpoint with respect to facades, since the reasoning is mostly qualitative.

Quantitative modeling of surface orientation can be found in many works of indoor scene understanding, where surface orientation is determined by the span of two orthogonal vanishing lines [6, 27, 46, 76]. While we also use vanishing lines to compute plane orientations, the methods developed for indoor scene understanding cannot be applied to outdoor facade analysis. This is because those methods typically simplify the room as a box, and all the other vertical surfaces are confined within the box and parallel to the walls. By contrast, building facades in outdoor scenes are located in an open space and usually have more complex structures. While the algorithm in [45] does not simplify the room as a box, it heavily relies on a common ceiling to define vertical walls. This is not applicable to outdoor scenes either.

Instead of modeling surfaces, Ramalingam *et al.* constructs a wire frame model of building facades by lifting vanishing lines into 3D space using orthogonality constraints derived from a strong Manhattan-world assumption [54]. A limitation of this approach is that, even when the Manhattan-world assumption holds true, all the building facades have to be connected in a single

wire frame model. In many cases, however, building facades occlude each other and do not form a connected structure. This is exemplified by the facades in red and blue in Figure 2.3.

Chapter 3

Geometric Variables and Visual Cues

In this chapter, we first discuss the geometric variables considered in our reasoning system. Those geometric variables include both 2D and 3D, local and global ones, and most of them are not directly observable. A major goal of our reasoning system is to infer the values of those geometric variables. Secondly, we describe different types of visual cues that are extracted from an image and fed into our 3D geometric reasoning system. Many of such visual cues themselves are obtained from sophisticated state-of-the-art algorithms focusing on specific visual tasks, and are therefore reasonably informative. Yet we will show in later chapters that our 3D geometric reasoning system further improves the performance of those visual tasks while providing a richer and more coherent interpretation of the 3D scene behind the image.

3.1 Geometric Variables

The coordinate systems we use in our work are shown in Figure 3.1. Variables in red letters are defined with respect to the camera coordinate system $\{o, x, y, z\}$ whose origin is at the camera center. Variables in blue letters are defined with respect to the ground coordinate system $\{O, X, Z\}$, whose origin is on the ground plane. In addition, variables in green letters are defined in the image plane.

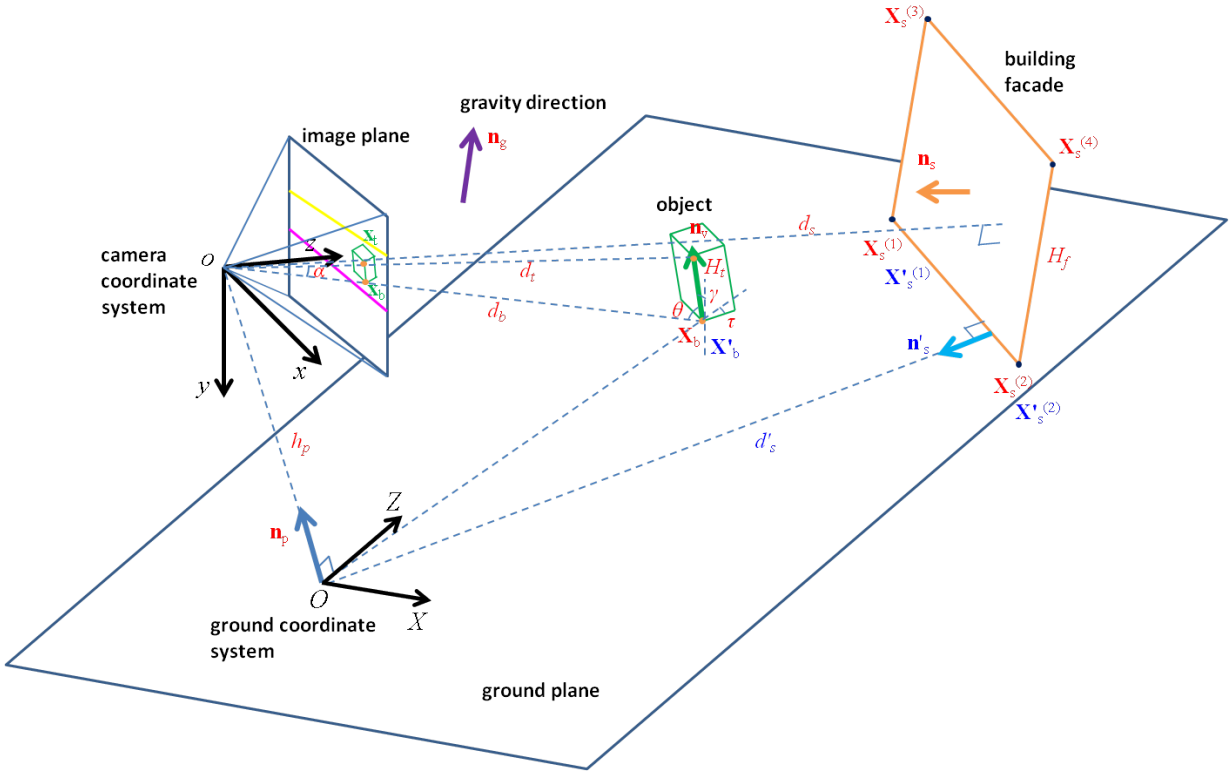

Figure 3.1: Geometric variables in two coordinate systems. Please see text for more details. For the sake of clarity, we do not draw the projection of the facade onto the image plane.

Among all the geometric variables, three of them are global – (inverse) gravity direction $\mathbf{n_g}$, ground plane orientation $\mathbf{n_p}$, and ground plane height h_p. These variables have interactions with every individual object and facade plane. Note that in our general setting, gravity direction and ground plane orientation do not have to align with the y-axis or fall within the $y-z$ plane of the camera coordinate system, nor do they have to agree with each other. In other words, the camera is allowed to have both pitch and roll motions relative to gravity and ground, and ground plane is allowed to be tilted with respect to gravity. Both gravity direction and ground plane orientation could induce a horizon line in the image, which is the intersecting line between the image plane and the origin-passing plane perpendicular to the gravity direction/ground plane orientation. The two horizon lines are illustrated by the magenta and yellow lines in the image plane in Figure 3.1.

Each individual object involves the following local geometric variables: object vertical orientation $\mathbf{n_v}$, object pitch angle θ, object roll angle γ, object yaw angle τ, object depths d_t and d_b for top and bottom landmarks, locations $\mathbf{x_t}$ and $\mathbf{x_b}$ of the top and bottom landmarks in the image, real world height H_t of the top landmark, object vertical viewing angle α, camera and ground coordinate $\mathbf{X_b}$ and $\mathbf{X'_b}$ of the bottom landmark, respectively.

For each facade plane, the following local geometric variables are involved: surface orientation $\mathbf{n_s}$, distance from camera d_s, ground contact line orientation $\mathbf{n'_s}$, ground contact line distance d'_s, camera coordinates of the facade plane corners $\mathbf{X_s^{(1)}} \sim \mathbf{X_s^{(4)}}$ and their projections on the image plane $\mathbf{x_s^{(1)}} \sim \mathbf{x_s^{(4)}}$ (not shown in Figure 3.1), ground coordinates $\mathbf{X_s^{'(1)}}, \mathbf{X_s^{'(2)}}$ of the two endpoints of the ground contact line, and real world height H_f.

The camera coordinate system $\{o, x, y, z\}$ and the ground coordinate system $\{O, X, Z\}$ are related as follows:

$$\mathbf{O} = -h_p \cdot \mathbf{n_p}, \tag{3.1}$$

$$\mathbf{e_Z} = \mathbf{m}\{\mathbf{n_p} \times (\mathbf{e_z} \times \mathbf{n_p})\}, \tag{3.2}$$

$$\mathbf{e_X} = \mathbf{e_Z} \times \mathbf{n_p}. \tag{3.3}$$

Here, $\mathbf{O}$ is the location of the ground origin O in the camera coordinate system. $\mathbf{e_X}$ and $\mathbf{e_Z}$ are the unit vectors of the X and Z axes with respect to the camera coordinate system. $\mathbf{e_z} = (0, 0, 1)^T$ is the unit vector of the z axis with respect to the camera coordinate system. Function $\mathbf{m}\{\mathbf{v}\}$ normalizes a vector $\mathbf{v}$ to unit length. Intuitively, the Z axis of the ground coordinate system lies at the intersection between the ground plane and the plane spanned by $\mathbf{n_p}$ and the z axis of the camera coordinate system.

We will discuss the relationships among those geometric variables in later chapters.

3.2 Detected Objects

Objects such as cars and pedestrians in an urban scene are one of the key focuses of our scene understanding system. Not only do we want to determine if they exist in the scene, but we also want to locate existing objects in 3D space. As we will show in the next chapter, reasoning over the 3D locations of objects would significantly help the task of determining their existence. In addition, each individual object itself is able to provide information regarding global 3D geometries such as gravity direction and ground plane parameters.

In order to utilize objects in our scene understanding task, we must first acquire a set of candidate detections of objects, and then use them to help infer global 3D geometries, which would in turn, together with other 3D geometric constraints, prune out invalid candidates. We obtain object candidates using the state-of-the-art Deformable Part Model (DPM) detector [13, 14, 15, 18, 20] with publicly available code at [19]. The DPM detector returns candidates whose overall appearance and part appearances are similar to the learned template *and* the 2D locations of the parts are consistent with the learned prior. The feature used by the DPM detector is histogram of oriented gradients (HOG) [9] which has been popular within the computer vision community. The classifier used by the DPM detector is latent support vector machine (LSVM) [15] that handles

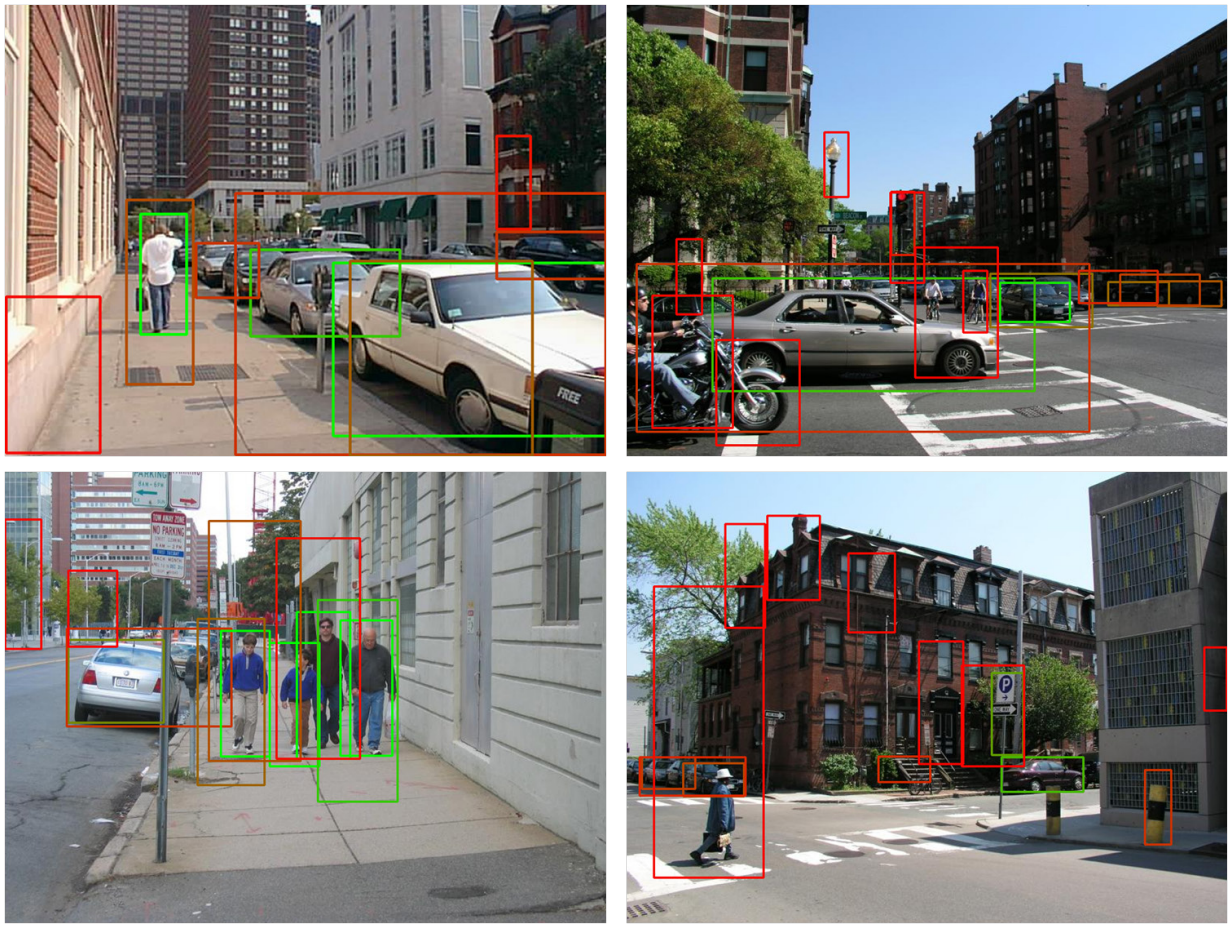

Figure 3.2: Examples of the output of the DPM detector [18, 19]. Green bounding boxes indicate the most confident detections with score 1, red bounding boxes indicate the least confident detections with score 0, and all the other colors from green to red indicate the detections with scores between 1 and 0.

latent variables of 2D part locations within the bounding box. The DPM detector also considers contextual cues such as the presence of other objects and 2D locations of bounding boxes to re-score each detection [15]. Further improvements include introducing grammar models [20] and adopting a cascade approach [14].

Examples of the detection results by the DPM detector on several images from the LabelMe dataset [34] are shown in Figure 3.2. Here, the bounding box color reflects the detection score – red meaning the least confident detection with score 0, green meaning the most confident detection with score 1, and every other color from red to green indicates a detection score in

between, with yellow meaning a borderline detection with score 0.5. We can see that while the results are quite reasonable, some valid detections have a score below 0.5, mostly due to occlusions. Also, some invalid detections have a score above 0.5, mostly due to confusing local appearance.

While the DPM detector achieves the state-of-the-art performance, all the reasoning and computations are done in the 2D image domain. As is mentioned in Chapter 1, geometric reasoning in 3D would further resolve visual ambiguities and boost object detection performance. To enable 3D geometric reasoning, for each object candidate, we estimate its 3D pose (*i.e.* pitch, roll, and yaw angles with respect to the camera) and 2D locations of top and bottom landmarks defined in Section 1.4. This is essentially a regression task, mapping from image features extracted from the patch enclosing the object candidate to a continuous variable. To achieve this purpose, we use the HOG feature and train a hierarchical discriminant regressor (HDR) [38] on the LabelMe training/validation set [34] with the ground truth of landmark locations and 3D pose we provide. The accuracies of the learned regressors are shown in Figure 3.3. Here, the errors of the pitch, roll, and yaw angles are measured in degree, and the landmark error is measured in the ratio of landmark deviation to bounding box size. We do not predict the yaw angle of pedestrians because of their near-circular footprints. We can see that prediction of roll is more accurate than pitch, which is similar to human experience. In addition, prediction of pedestrian pitch is harder than car pitch, which is also expected due to the pole-like shape of pedestrians. We can also see that prediction of the yaw angles of cars is less accurate than pitch and roll angles. This is due to the much larger variations in yaw angles in the training set. Yet 18 degrees of error still provides informative cues of car heading. For landmark prediction, the error is around 10% of bounding box size. Acknowledging the nature of uncertainty in the prediction of geometric properties, we adopt a probabilistic approach in 3D geometric reasoning, as we will discuss in later chapters.

	Pitch	Roll	Yaw	Landmark
Car	2.65	1.07	18.4	0.10
Pedestrian	5.07	0.97	N/A	0.12

Figure 3.3: Prediction errors of object geometric properties on the LabelMe dataset [34]. For pitch, roll, and yaw angles with respect to the camera, the error is measured in degree. For landmark location, the error is measured in the ratio of the landmark deviation to the bounding box size. We do not predict the yaw angle of pedestrians, because their footprints are small and circular, unlike the footprints of cars which are larger and rectangular.

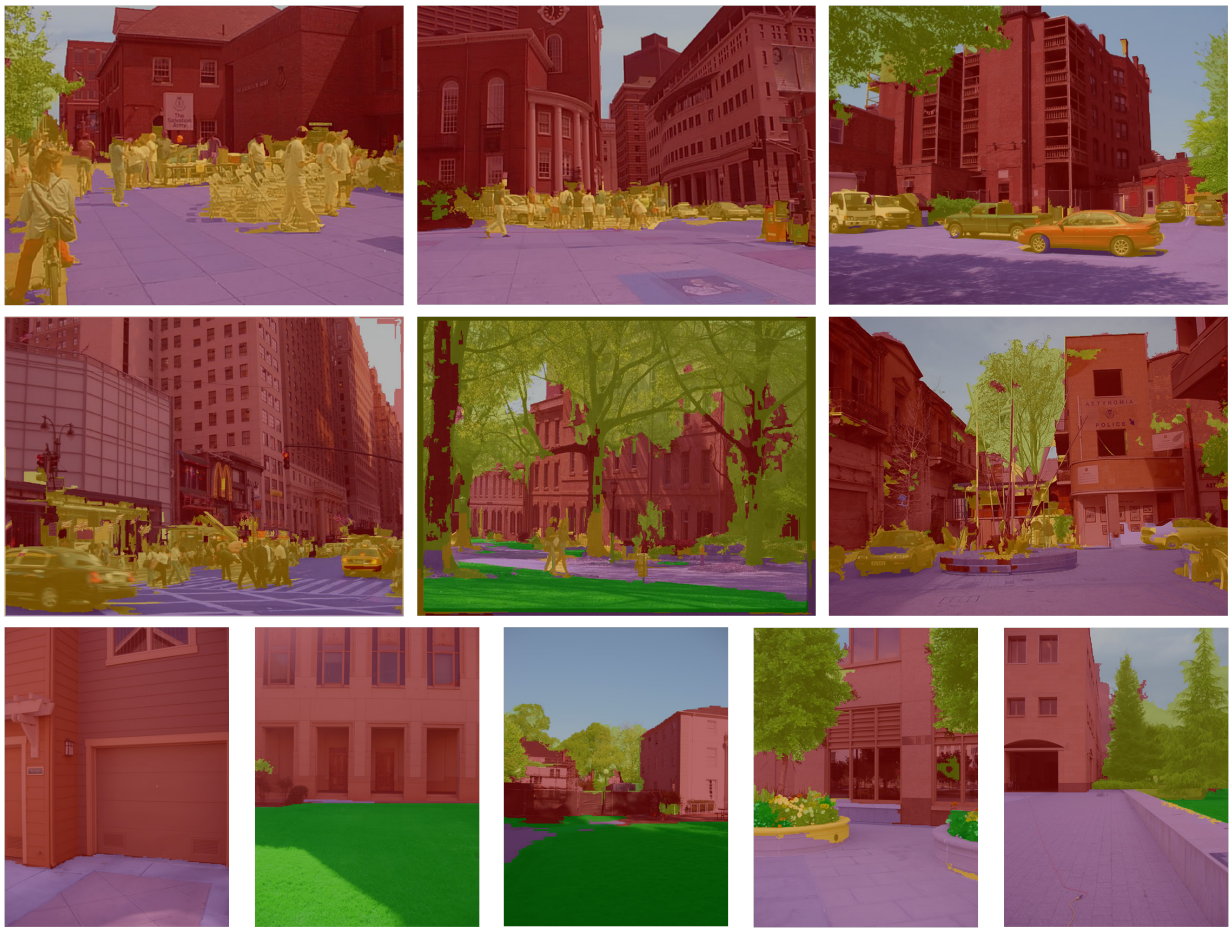

Figure 3.4: Examples of semantic segmentation results using the SHL algorithm [51]. The three rows from top to bottom show the results on the LabelMe [34], Geometric Context [33], and Make3D [48] datasets, respectively. The color code of semantic labels is as follows: red – building; purple – road; green – grass; grey – sky; light green – tree; yellow – foreground.

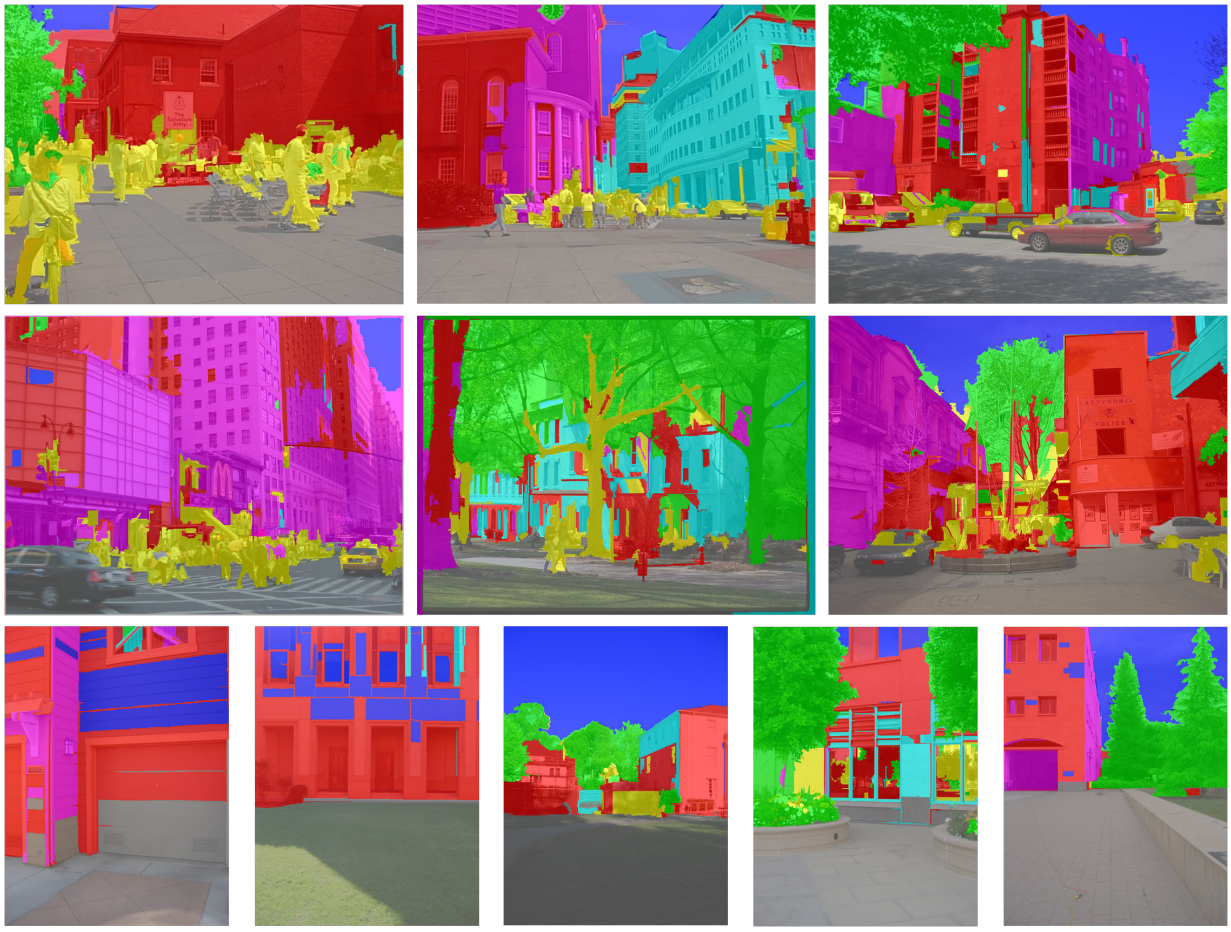

Figure 3.5: Examples of surface layout segmentation results using Hoiem's surface layout algorithm [33]. The examples in this figure are exactly the same as those in Figure 3.4. The color code of surface layout labels is as follows: grey – support; blue – sky; cyan – planar left; red – planar center; magenta – planar right; green – non-planar porous; yellow – non-planar solid.

3.3 Semantic Segmentation

Semantic cues play an important part in our 3D scene understanding system. For example, reasoning over the 3D orientations and locations of building facades requires the system to get a sense of where the building region is in the 2D image. Besides, the road/grass region in the image provides useful cues of where the ground contact lines of facade planes could and could not be in the image, and therefore imposing constrains on the depth of the facade planes. In addition, when using vanishing lines to estimate horizon, we discard line segments from image regions other than building and road to suppress noise. Also, the sky and road/grass regions limit what the ground orientation could be, and therefore assist in estimating global 3D geometries.

We obtain semantic segmentations using the Stacked Hierarchical Labeling (SHL) algorithm proposed by Munoz *et al.* [51]. This algorithm adopts a hierarchical labeling process that predicts semantic label distributions of image regions in a coarse to fine manner – a region with a smaller scale predicts its label distribution by utilizing both its own features and the label distributions of larger regions covering it, thereby effectively exploiting its higher-order spatial context. The authors show that this approach achieves impressive semantic segmentation results while consuming much less computation than other structured models.

As the LabelMe dataset [34] and Geometric Context dataset [33] do not provide ground truth for semantic segmentation, we train the SHL model on the Stanford Background dataset [23] and apply the learned model on the LabelMe and Geometric Context datasets. Since the Make3D dataset [48, 60] comes with ground-truth semantic segmentations, we train the SHL model on the 400 training images of the dataset. Some typical examples of the segmentation results on the three datasets are shown in Figure 3.4. Note that while the output of the SHL algorithm is a semantic label distribution for each pixel, we are showing the most likely label for the sake of clarity. We can see that the SHL algorithm achieves good performance, even for cross-dataset cases. While there still exist some mistakes, our aggregate use of the semantic segmentation feature as well as 3D geometric constraints makes our system insensitive to those isolated mistakes.

3.4 Surface Layout Segmentation

Another informative feature we utilize in our 3D geometric reasoning system is the surface layout segmentation produced by Hoiem's surface layout algorithm in [33]. While semantic segmentation decomposes an image into regions with different semantics, surface layout segmentation decomposes an image into regions with different surface geometries – "support", "sky", "planar left", "planar center", "planar right", "non-planar porous", and "non-planar solid". While the two types of segmentations provide some redundant information (*e.g.*, "support"and "non-planar porous"usually cover similar regions as "road"and "tree", respectively), surface layout segmentation also provides important information that semantic segmentation lacks – categorical orientations of vertical surfaces. Along with vanishing lines and orientation maps (which will be discussed in the next two sections), surface layout features are used in decomposing the building region into distinctive facade planes.

We use the pre-trained model provided by Hoiem *et al.* [32] to obtain surface layout segmentations on all the three datasets. For the sake of comparison, the results on the same set of images in Figure 3.4 are shown in Figure 3.5. Note that here we are showing the most likely surface layout label instead of the label distribution produced by the algorithm. We can see that although the surface layout labeling is somewhat noisy, it is still informative in general.

3.5 Horizon from Vanishing Lines

Horizon lines in an image reflects the values of global 3D geometries. More specifically, both gravity direction and ground plane orientation with respect to the camera would yield a horizon line in the image. When the ground is tilted with respect to gravity, the two horizon lines would differ, as is shown in Figure 3.6.

In images of man-made environments such as urban scenes, many edges are parallel in 3D. The lines defined by those edges intersect at the same point or at infinity in the image domain.

Figure 3.6: Vanishing lines corresponding to vertical and horizontal vanishing points, and horizon lines corresponding to gravity and ground. The thiner lines are vanishing lines, where the red ones correspond to the vertical vanishing point, and the green, blue, and magenta ones correspond to three different horizontal vanishing points. The thicker lines are horizon lines, where the yellow one corresponds to ground plane orientation, and the magenta one corresponds to gravity direction. Note that as the visible ground is tilted with respect to gravity direction, the ground horizon deviates from the gravity horizon.

Those intersecting lines are called vanishing lines, and the intersection point of those lines is called vanishing point. In Figure 3.6, for example, the thiner lines with the same color are the vanishing lines parallel in 3D and therefore intersecting at the same vanishing point in 2D.[1]

Among all the vanishing points, there are two special types. The first type is vertical vanishing point, which corresponds to a group of parallel lines whose direction is the same as the gravity direction. In typical urban scenes, vertical vanishing lines mostly fall on the vertical edges of building facades, and form the largest group of vanishing lines. The second type is horizontal vanishing point, which corresponds to a group of parallel lines whose direction is perpendicular to the vertical vanishing direction. In typical urban scenes, horizontal vanishing lines mostly fall

[1]Since the vanishing lines are automatically extracted by our algorithm, there exist a few mistakes.

on the horizontal edges of building facades. The orthogonality between the vertical vanishing direction and each horizontal vanishing direction could be used to estimate focal length, with which the 3D vector of gravity direction can be computed. Therefore, vertical and horizontal vanishing lines provide strong cues for the gravity direction. In Figure 3.6, the red vanishing lines belong to the vertical vanishing point, and the green, blue and magenta vanishing lines belong to three different horizontal vanishing points. Note that horizontal vanishing directions do not have to be orthogonal to each other.

To identify vertical and horizontal vanishing lines and estimate vertical and horizontal vanishing point positions, we follow an approach inspired by the one in [43], with two major differences – RANSAC-based initialization and removal of the Manhattan world assumption.

We first extract line segments from the building and road regions using the line detection algorithm presented in [72, 73].[2] With the Expectation-Maximization (EM) algorithm [10], we cluster those line segments into groups where each group contains lines passing or close to a common vanishing point. To handle vanishing points at infinity, we use Gaussian sphere to do the clustering [1, 4, 7]. Unlike the method in [43] that initializes vanishing points by grouping line segments with similar orientations in the image, we adopt a RANSAC approach [16] to initialize vanishing points. More specifically, we randomly select two line segments and obtain their intersection point. Then we count how many line segments are nearly aligned with the intersection point. For each intersection point with sufficient aligned line segments, we evaluate its score by computing the average degree of alignment. After drawing a large number of random intersection points, we select the one with the highest score as the first vanishing point. All the line segments associated with this vanishing point are removed, and the RANSAC process repeats to extract other vanishing points, until no line segment remains.

While there could be many vanishing points, what we need are vertical and horizontal vanishing points. Unlike the method in [43] that assumes there are only three orthogonal vanishing points (one vertical, two horizontal) corresponding to the three axes of the world reference frame

[2]The code is publicly available at [71].

(*i.e.* the Manhattan world assumption), we do not assume there are only two horizontal vanishing points, nor do we assume horizontal vanishing points are orthogonal to each other. The only constraint we impose is that the vertical vanishing point must be orthogonal to each horizontal vanishing point. Assuming a reasonable initial focal length, we start by identifying vertical vanishing lines as the most dominant group of vanishing lines whose direction is within $\pi/4$ from the camera's vertical axis. After obtaining the vertical vanishing point, we identify horizontal vanishing points as the ones that are nearly orthogonal to the vertical vanishing point. Note that there could be more than two horizontal vanishing points. Then we search for the optimal focal length that maximizes the average orthogonality between the vertical vanishing direction and every horizontal vanishing directions. The optimal vertical vanishing direction serves as a good hypothesis of the gravity direction.

We compare the horizon estimation accuracy between the method in [43] and our algorithm in which the Manhattan world assumption is relaxed (*i.e.*, there could be more than two horizontal vanishing directions, and they are not required to be orthogonal to each other). The experiment is conducted on the 100 images of the dataset compiled by Hoiem *et al.* [34] with ground-truth horizons. Estimation error is defined as the angle between the estimated vertical vanishing direction and the ground-truth vertical vanishing direction derived from the ground-truth horizon. The estimation error of the method in [43] is 2.77 degrees, while the estimation error of our algorithm is 2.53 degrees.

Our method achieves a higher accuracy mostly because of the RANSAC-based initialization, since a majority of the urban images in the dataset satisfy the Manhattan world assumption. Two examples are shown in Figure 3.7. In each row, the thick magenta line in the left image is our result, and the level black line in the right image is the result of the method in [43]. In the left image, red lines are vertical vanishing lines, and horizontal vanishing lines belonging to different horizontal vanishing points are indicated in different colors. In the right image, red, green, and blue lines indicate the lines belonging to the vertical vanishing point and two horizontal

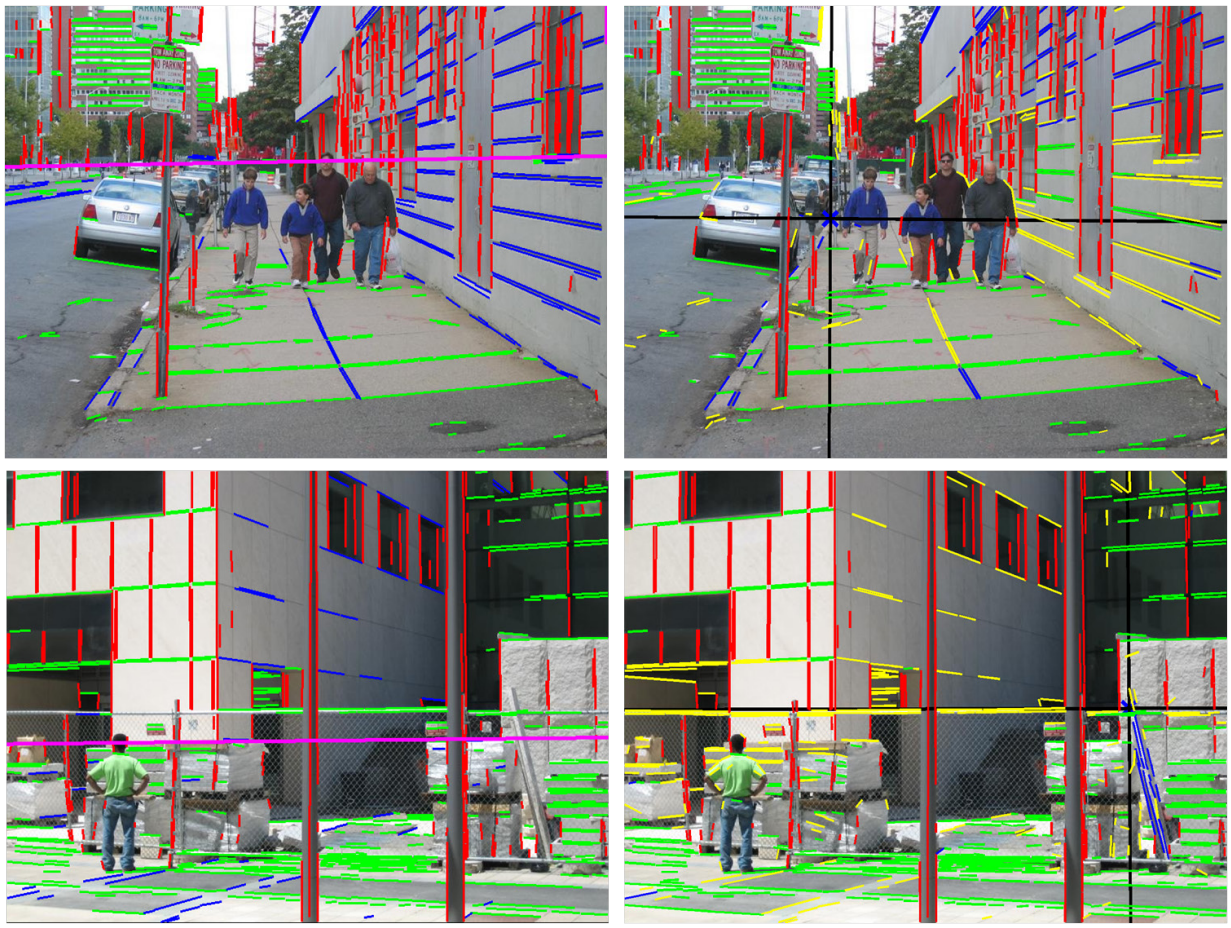

Figure 3.7: Qualitative comparison of horizon estimation between our algorithm and the method in [43] when the Manhattan world assumption is satisfied. The left column is the result of our algorithm, where the estimated horizon is indicated by the thick magenta line. The red lines are vertical vanishing lines, and the other lines with different colors indicate horizontal vanishing lines belonging to different horizontal vanishing points. The right column is the result of the method in [43], where the estimated horizon is indicated by the level black line. The red, green, and blue lines belong to the vertical and two horizontal vanishing points, respectively, while the yellow lines are outlier lines.

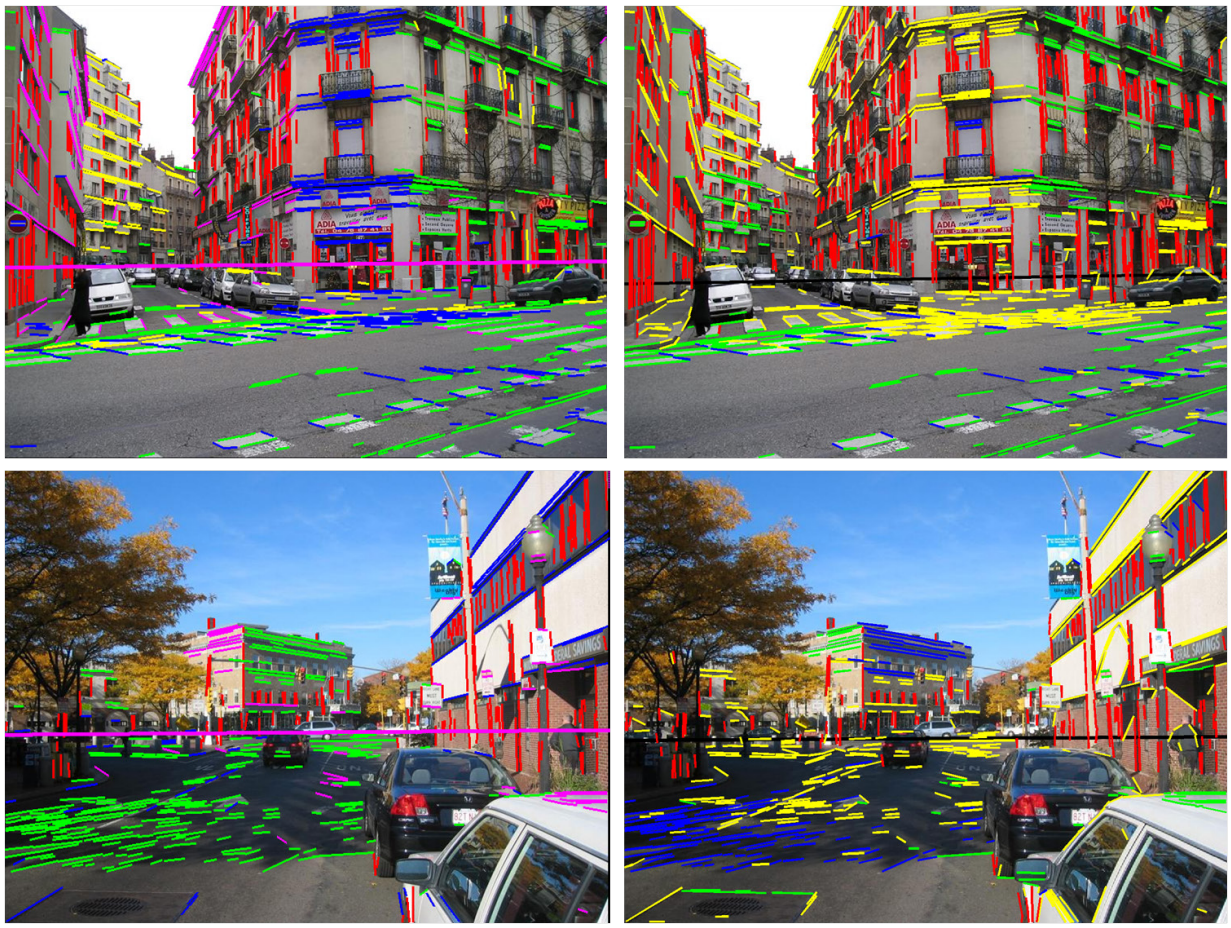

Figure 3.8: Qualitative comparison of horizon estimation between our algorithm and the method in [43] when the Manhattan world assumption is violated. The left column is the result of our algorithm, where the estimated horizon is indicated by the thick magenta line. The red lines are vertical vanishing lines, and the other lines with different colors indicate horizontal vanishing lines belonging to different horizontal vanishing points. The right column is the result of the method in [43], where the estimated horizon is indicated by the level black line. The red, green, and blue lines belong to the vertical and two horizontal vanishing points, respectively, while the yellow lines are outlier lines.

vanishing points, respectively, while yellow lines are outlier lines. We can see that even when the Manhattan world assumption holds true in these examples, the method in [43] mistakenly groups many line segments due to bad initialization from which EM is unable to recover.

When images violating the Manhattan world assumption are encountered, the method in [43] is not effective at all. Two examples are shown in Figure 3.8. Note that our algorithm manages to identify all the non-orthogonal horizontal vanishing points. This is very important for 3D facade reasoning.

After obtaining the gravity direction from vanishing lines, we estimate the ground plane orientation as follows. If the horizon line derived from the gravity direction trespasses neither the road/grass nor the sky semantic regions (which are obtained from semantic segmentation in [51]), the ground orientation is identical to the gravity direction; otherwise, we search around the gravity direction and find the closest direction whose horizon line does not trespass the road/grass or the sky region. Please see Figure 3.6 for example.

3.6 Orientation Map from Line Sweeping

In our 3D geometric reasoning system, we need to decompose each building region into a set of distinctive facade planes, each of which has a different orientation. The most important cue of the orientation of a facade region is the directions of vanishing lines located within the region – if the region is truly a facade plane, its orientation can be determined by the vertical vanishing lines and the (single set of) horizontal vanishing lines located within the region. However, as vanishing lines could often be sparse, a facade region may not contain at least one vanishing line for both the vertical and horizontal vanishing directions. Therefore, it would be desirable to generate a dense orientation map from those vanishing lines.

We use an approach similar to the line sweeping algorithm proposed by Lee *et al.* [45]. The original algorithm in [45] is as follows. (Please also see Figure 3.9 for illustration.) Suppose vanishing line l_v belongs to the vertical vanishing point v_v, and vanishing line l_{h1} belongs to a

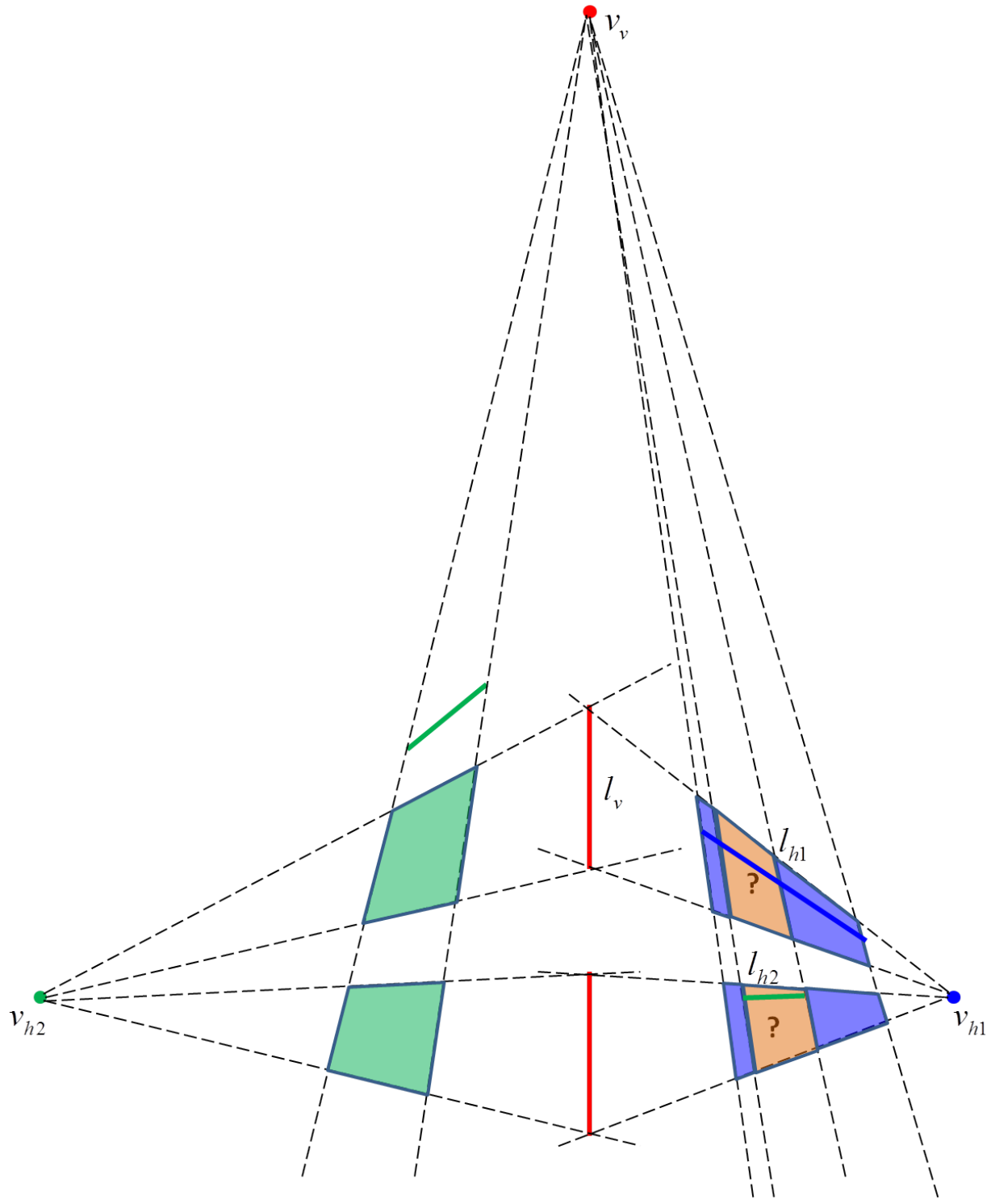

Figure 3.9: Illustration of the line sweeping algorithm proposed in [45]. Please see text for more details.

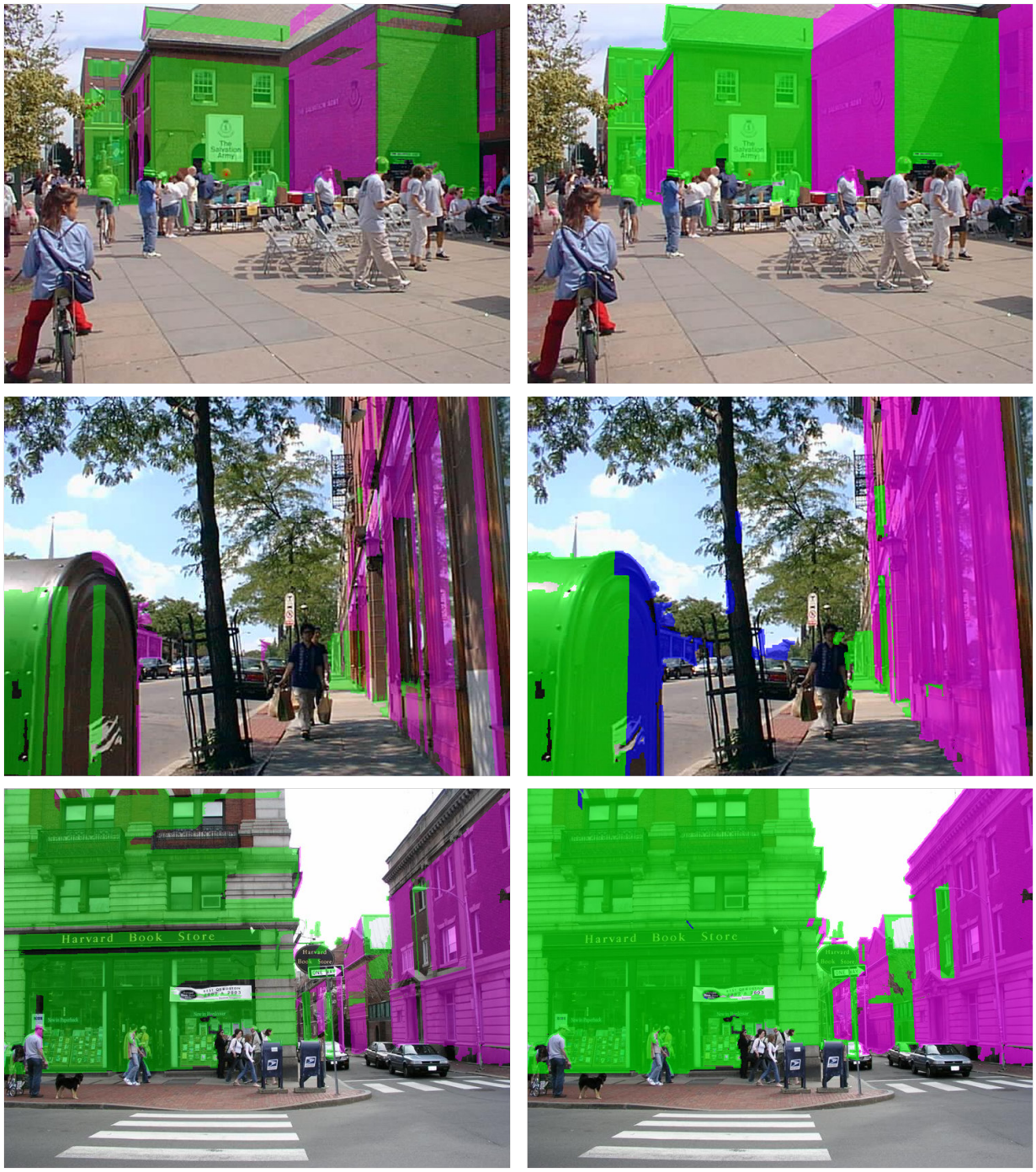

Figure 3.10: Comparison between the original line sweeping algorithm [45] (left column) and our improved version (right column). Facade regions with different orientations are shaded with different colors. Undetermined regions are not shaded.

certain horizontal vanishing point v_{h1}. We first sweep l_v around v_v within the triangular region defined by v_{h1} and the two endpoints of l_v. For each direction, the sweeping is stopped when the line hits another line that neither belongs to v_v nor v_{h1}. We record the swept region as $\mathbf{R}_{v,h1}$. Then we sweep l_{h1} around v_{h1} confined by v_v in the same manner, resulting in another swept region $\mathbf{R}_{h1,v}$. The intersection region $\mathbf{R}_{v,h1} \cap \mathbf{R}_{h1,v}$ has an orientation $\mathbf{v}_v \times \mathbf{v}_{h1}$, where $\mathbf{v}$ is the Gaussian sphere vector corresponding to vanishing point v. However, if, say, $\mathbf{R}_{v,h2} \cap \mathbf{R}_{h2,v}$ happens to claim the same region, then the orientation of this region is regarded as undetermined. This phenomenon is more likely to occur when line segments are near the horizon line and therefore mistakenly grouped to the wrong vanishing point, as is illustrated by the horizontal green line segment and the orange regions in Figure 3.9. The orientations of the regions that do not belong to any intersection regions $\{\mathbf{R}_{v,hi} \cap \mathbf{R}_{hi,v}\}|_i^k$ are also undetermined, where k is the total number of horizontal vanishing points.

As blocked sweepings and conflicting claims would result in undetermined regions in the original algorithm, the resulting orientation maps could only cover part of the building region in most cases, as is shown in the left column of Figure 3.10. To increase the coverage, we make several improvements upon the original algorithm. Firstly, we break a line segment into shorter sub-segments and do line sweeping independently for each sub-segment. This way, many regions that are otherwise unaccessible to the original line segment could be swept by some of its sub-segments.

Secondly, we assign weights to each line segment by computing the probability of the line segment belonging to each horizontal vanishing point as follows. We first obtain the angle between the line segment and the direction pointing from the midpoint of the line segment to each horizontal vanishing point. Such an angle is then converted to a likelihood score by a Gaussian function.[3] The probability of the line segment belonging to different horizontal vanishing points is computed by normalizing the likelihood scores over all horizontal vanishing points. For example, in Figure 3.9, l_{h1} has a much higher probability of belonging to v_{h1} than l_{h2} belonging to v_{h2}

[3]The larger the angle, the lower the likelihood.

due to the normalization process. As a result, the originally undetermined orange regions would be regarded as belonging to the blue regions.

Thirdly, we fill in the vertical and horizontal gaps between adjacent determined quadrilateral regions. Our improved line sweeping algorithm produces a much denser orientation map, as is shown in the right column of Figure 3.10. We also quantitatively compare the coverage ratio on the LabelMe dataset [34]. The coverage ratio is defined as the ratio between the area of the determined building region to the area of the entire building region. Our improvements on the line sweeping algorithm increase the coverage ratio from 67.6% to 95.9%.

Chapter 4

Coherent Object Detection with 3D Geometric Context

In this chapter, we model and utilize the 3D geometric interactions between global 3D geometries (*i.e.* gravity and ground plane) and individual objects as well as the interactions among adjacent objects to ensure 3D geometric coherence of detected objects.

In achieving this purpose, we make the following contributions. Firstly, we formally establish constraints among geometric variables of gravity, ground, and objects shown in Figure 3.1 in a general setting, providing a more accurate and flexible representation of geometric relationships and properties than the works in [3, 34, 66].

Secondly, we propose a novel generalized RANSAC method to address the chicken-and-egg problem of jointly estimating global 3D geometries and detecting objects mentioned in Section 1.2. Unlike previous approaches in [3, 34, 66], our proposed method does not require an exhaustive search of global 3D geometries in a limited range, nor does it involve those global geometric variables into an intractable inference over a probabilistic graphical model (PGM). In addition, our generalized RANSAC method could simultaneously suppress the corruption of global geometry estimation caused by false detections, *and* improve the accuracy of global geometry estimation by reducing the noise of inaccurate estimates produced by true detections.

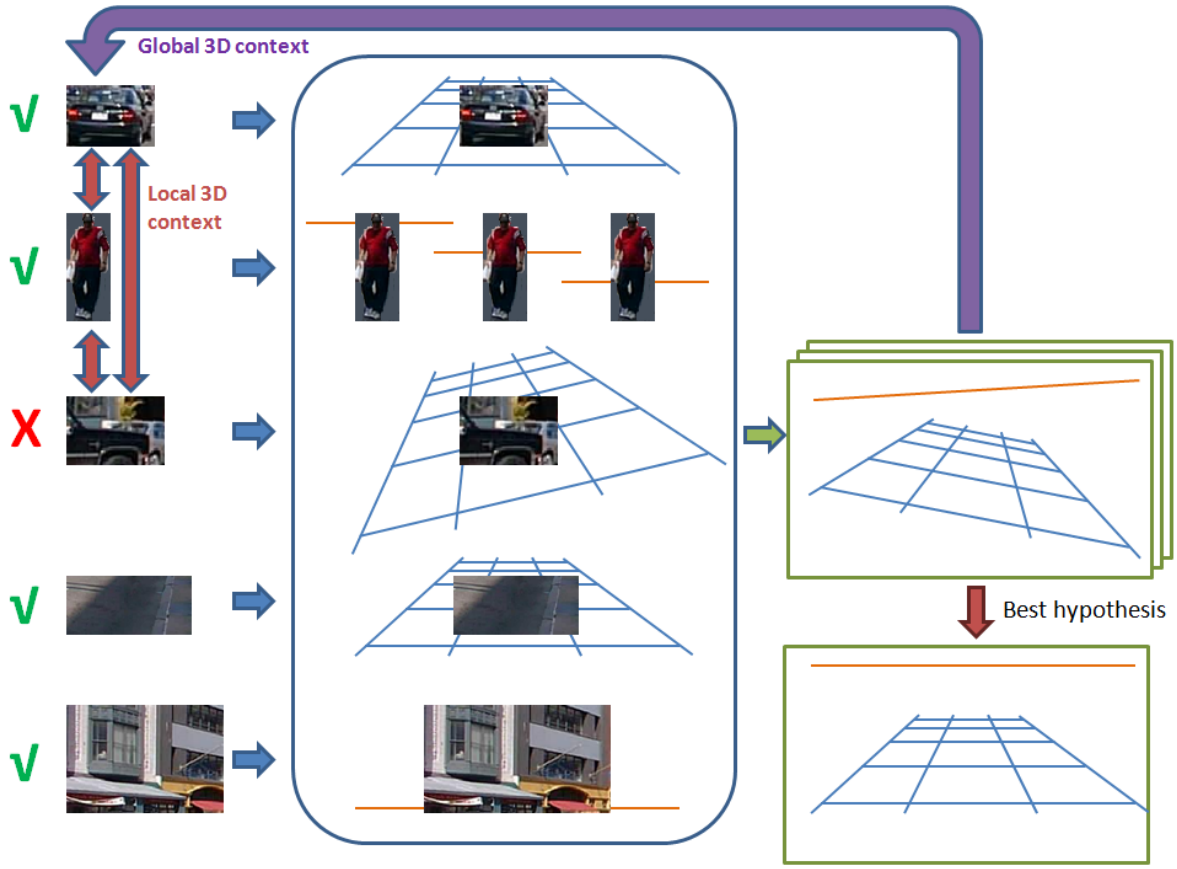

Figure 4.1: The overall scheme of our algorithm. Here, gravity direction w.r.t. camera is represented by the orange horizon line, and the ground plane is represented by the blue mesh.

Thirdly, different from the works in [3, 34, 66] that only consider the constraints of global 3D geometries imposed on each individual object candidate, we also introduce local 3D geometric constraints between object candidates, and integrate the global and local 3D geometric constraints in a Conditional Random Field (CRF) [44].

Attributed to these contributions, our algorithm outperforms the state of the art in terms of object detection and global 3D geometry estimation on the challenging outdoor images from the LabelMe dataset [34].

4.1 Overview

The overall scheme of our algorithm is illustrated in Figure 4.1. We start by generating object/surface candidates (including cars, pedestrians, vertical and horizontal surface regions) using state-of-the-art object detectors (*e.g.* Deformable Part Model (DPM) mentioned in Section 3.2) and surface segmentation algorithms (*e.g.* Surface Layout Model (SLM) mentioned in Section 3.4). Each object/surface candidate gives an estimate of the global 3D geometries (*i.e.* gravity direction and ground plane parameters) based on their 2D appearance. Those noisy estimates are then pooled together using a generalized RANSAC algorithm to generate a set of global 3D geometry hypotheses. Given each hypothesis, we compute the compatibility of each object/surface candidate and infer their validity according to global and local 3D geometric context. The quality of each hypothesis is obtained as a result of the inference procedure. Finally the hypothesis with the highest quality is selected as the optimal estimate of the global 3D geometries, and the inference result of object candidate validity associated with the best hypothesis gives the final object detection result.

4.2 Geometric Relationships Among Variables

The coordinate systems we use in our geometric reasoning are illustrated in Figure 3.1. For a detailed description of the coordinate systems as well as the geometric variables in Figure 3.1, please refer to Section 3.1. Now we list the relationships that exist among global geometric variables and object geometric variables:

$$d_t = H_t \sin\theta / \sin\alpha; \tag{4.1}$$

$$d_b = H_t(\sin\theta / \tan\alpha + \cos\theta); \tag{4.2}$$

$$\mathbf{n_v} = \mathbf{m}\{d_t\mathbf{r}\{\mathbf{x_t}, f\} - d_b\mathbf{r}\{\mathbf{x_b}, f\}\}; \tag{4.3}$$

$$\mathbf{n_v} = \mathbf{g}\{-\mathbf{r}\{\mathbf{x_b}, f\}, \theta, \gamma\}; \tag{4.4}$$

$$\alpha = \arccos\{< \mathbf{r}\{\mathbf{x_t}, f\}, \mathbf{r}\{\mathbf{x_b}, f\} >\}; \tag{4.5}$$

$$h_p = -d_b < \mathbf{n_p}, \mathbf{r}\{\mathbf{x_b}, f\} >; \tag{4.6}$$

$$\mathbf{n_v} = \begin{cases} \mathbf{n_p} & \text{for cars} \\ \mathbf{n_g} & \text{for pedestrians} \end{cases} \tag{4.7}$$

Here, the function $\mathbf{r}\{\mathbf{x}, f\}$ computes the unit vector pointing from the camera center towards pixel location $\mathbf{x}$ in the image plane under focal length f. The function $\mathbf{m}\{\mathbf{v}\}$ normalizes a vector $\mathbf{v}$ to unit length. The function $\mathbf{g}\{\mathbf{v}, \theta, \gamma\}$ rotates a unit vector $\mathbf{v}$ by pitch angle θ and roll angle γ. The function $< \mathbf{v_1}, \mathbf{v_2} >$ computes the inner product of two vectors.

In addition, we could also estimate the distributions of object pitch, roll and yaw angles as well as top and bottom landmark locations given object appearance I and category c: [1]

$$\theta \sim p_{a1}(I, c); \tag{4.8}$$

$$\gamma \sim p_{a2}(I, c); \tag{4.9}$$

$$\tau \sim p_{a3}(I, c); \tag{4.10}$$

$$\mathbf{x_t} \sim p_{mt}(I, c). \tag{4.11}$$

[1] We do not estimate the yaw angle of pedestrians.

$$\mathbf{x_b} \sim p_{mb}(I, c). \tag{4.12}$$

Another cue we could use is the prior knowledge that predicts the distribution of object height given its category c:

$$H_t \sim p_h(c). \tag{4.13}$$

Among all the geometric variables, only $\mathbf{x_t}$ and $\mathbf{x_b}$ are directly observable. When multiple object candidates are present, we are faced with a large set of equations containing non-linear and even non-deterministic constraints. Moreover, many false detections would produce invalid equations. Therefore, instead of attempting to directly solve them, we use those equations to propose hypotheses of the global 3D geometries $(\mathbf{n_g}, \mathbf{n_p}, h_p)$, and to evaluate those hypotheses.

4.3 Generating Global 3D Geometry Hypotheses

Once we have identified the relationships between global and local geometric variables, we could use local entities to generate hypotheses of global 3D geometries. More specifically, each object/surface candidate gives its own (potentially noisy or even invalid) estimate of gravity direction or ground plane parameters, and a generalized RANSAC algorithm is used to generate hypotheses of gravity and ground from a pool of local estimates.

4.3.1 Estimates of Global 3D Geometries from Local Entities

For each object candidate: Using the constraints from the equations listed above, we estimate a non-parametric distribution of the vertical orientation $\mathbf{n_v}$ of each object candidate from the appearance of the image patch enclosing it. Instead of resorting to a multi-view object detector that returns only a handful of discrete object poses, we directly estimate the mean/variance of

the pitch and roll angles of a detected object by applying a regressor trained using the Hierarchical Discriminant Regression model (HDR) [38] with HoG features [9] on the LabelMe training dataset [34, 57]. We also estimate the mean/variance of the 2D positions of the landmarks using the HDR regressor. Here, the top and bottom landmarks for a car are the top and bottom locations of the wheel closest to the camera (which are stabler and better-defined), and for a pedestrian, they are the head and foot locations.[2] The pitch and roll angles, together with landmark locations, produce a non-parametric distribution of object vertical orientation $\mathbf{n_v}$, according to the algorithm summarized in Figure 4.2. Following constraint 4.7, the vertical orientation distributions obtained from cars are regarded as the estimations of the ground plane orientation $\mathbf{n_p}$, and those from pedestrians are for the gravity direction $\mathbf{n_g}$.

In addition to estimating $\mathbf{n_g}$ and $\mathbf{n_p}$, given the vertical orientation, each object candidate also provides cues for the ground plane height h_p according to its size and location. The algorithm for using an object candidate to generate a non-parametric distribution of h_p is summarized in Figure 4.3.

For each surface candidate: We use Hoiem's surface layout algorithm [33] to identify image regions corresponding to vertical and horizontal (*i.e.* support) surfaces.[3] Within the vertical surface region, we extract line segments and compute vertical and horizontal vanishing points (VP) using the algorithm described in Section 3.5. To account for uncertainty, each vanishing point is represented by a set of line-intersection points, as is illustrated in Figure 4.4a. The vertical direction $\mathbf{n_v}$ of the surface can be estimated directly from the vertical VP direction[4], or from the cross-product of a pair of horizontal VP directions. Therefore, for the vertical VP, it directly yields a set of $\mathbf{n_v}$ samples from the directions of its constituent line-intersection points. For each pair of horizontal VPs, we compute the cross product of the directions of their respec-

[2]For a quantitative evaluation of pose and landmark estimation, please see Section 3.2.

[3]Alternatively, we could also use the line sweeping algorithm [45] to identify vertical and horizontal surfaces. However, when considering both vertical and horizontal surfaces, the orientation map obtained from the line sweeping algorithm becomes much noisier. As a result, we do not use it to obtain vertical and horizontal surfaces.

[4]Here, the direction of a point means a 3-dimensional unit vector pointing from the camera center to the point on the image plane.

Input: I -- The image patch enclosing the object candidate

Output: $D[\mathbf{n_v}]$ -- Non-parametric distribution of the vertical orientation of the object candidate

Predefined: $\mathbf{G_n} = \{\mathbf{n_i}\}$ -- A dense grid on the upper half of the unit sphere

Procedure:

1. Apply pose regressor to image patch I to obtain the mean and variance of the pitch angle θ and the roll angle γ. (Constraints 4.8 and 4.9)
2. Apply landmark regressor to image patch I to obtain the mean and variance of the top landmark $\mathbf{x_t}$ and bottom landmark $\mathbf{x_b}$. (Constraints 4.11 and 4.12)
3. Apply prior knowledge to obtain the mean and variance of the real world height H_t of the top landmark. (Constraint 4.13)
4. Sample a large number of $\{\theta, \gamma, \mathbf{x_t}, \mathbf{x_b}, H_t\}$ according to the mean and variance of those variables
5. For each sample of $\{\theta, \gamma, \mathbf{x_t}, \mathbf{x_b}, H_t\}$:
   5.1. Apply constraint 4.4 to obtain a sample of vertical orientation $\mathbf{n_{v1}}$.
   5.2. Apply constraints 4.5, 4.1, 4.2, and 4.3 to obtain another sample of vertical orientation $\mathbf{n_{v2}}$.
6. Pool together all samples of $\mathbf{n_{v1}}$ and $\mathbf{n_{v2}}$, and apply kernel density estimation (KDE) to obtain the density at each node $\mathbf{n_i}$ on the grid $\mathbf{G_n}$.
7. Normalize the densities by the total number of samples to obtain the final distribution $D[\mathbf{n_v}]$.

Figure 4.2: Algorithm for generating a non-parametric distribution of the vertical orientation $\mathbf{n_v}$ of an object candidate. The space under consideration covers the entire upper dome of unit sphere, and the resolution is 1.5 degrees.

Input: I -- The image patch enclosing the object candidate
$\mathbf{n_v}$ -- Vertical orientation of the object candidate
$\mathbf{n_p}$ -- Ground plane orientation
Output: $D[h_p]$ -- Non-parametric distribution of the ground plane height
Predefined: $\mathbf{G_h} = \{h_i\}$ -- A dense grid on the scalar range from 0 to 50
Procedure:
1. Apply landmark regressor to image patch I to obtain the mean and variance of the top landmark $\mathbf{x_t}$ and bottom landmark $\mathbf{x_b}$. (Constraints 4.11 and 4.12)
2. Apply prior knowledge to obtain the mean and variance of the real world height H_t of the top landmark. (Constraint 4.13)
3. Sample a large number of $\{\mathbf{x_t}, \mathbf{x_b}, H_t\}$ according to the mean and variance of those variables
4. For each sample of $\{\mathbf{x_t}, \mathbf{x_b}, H_t\}$:
   4.1. Apply constraints 4.4, 4.5, 4.2, 4.6 to obtain a sample of ground plane height h_p.
5. Pool together all samples of h_p, and apply kernel density estimation (KDE) to obtain the density at each node h_i on the grid $\mathbf{G_h}$.
6. Normalize the densities by the total number of samples to obtain the final distribution $D[h_p]$.

Figure 4.3: Algorithm for using an object candidate to generate a non-parametric distribution of the ground plane height h_p. The range under consideration is 0-50m, and the resolution is 0.05m.

tive constituent line-intersection points and generate a set of $\mathbf{n_v}$ samples. The $\mathbf{n_v}$ samples from the vertical VP and all pairs of horizontal VPs are pooled together to generate a non-parametric distribution of $\mathbf{n_v}$ over the dense grid $\mathbf{G_n}$ using Kernel Density Estimation (KDE) [74].

Estimating the vertical direction of a horizontal surface region (*e.g.* a road) is similar, except that its vertical VP is highly unreliable and therefore not used. Please see Figure 4.4b for illustration.

In urban scenes, vertical surfaces usually correspond to building facades which typically agree with the gravity direction, while horizontal surfaces usually fall on roads which relate to the ground plane orientation. Therefore, we have $\mathbf{n_v} = \mathbf{n_g}$ for vertical surfaces and $\mathbf{n_v} = \mathbf{n_p}$ for horizontal surfaces.

We also estimate horizon from all building and road regions (please see Section 3.5), compute the distribution of gravity direction/ground plane orientation using a Gaussian model, and add the distribution to the pool of distributions from local entities.

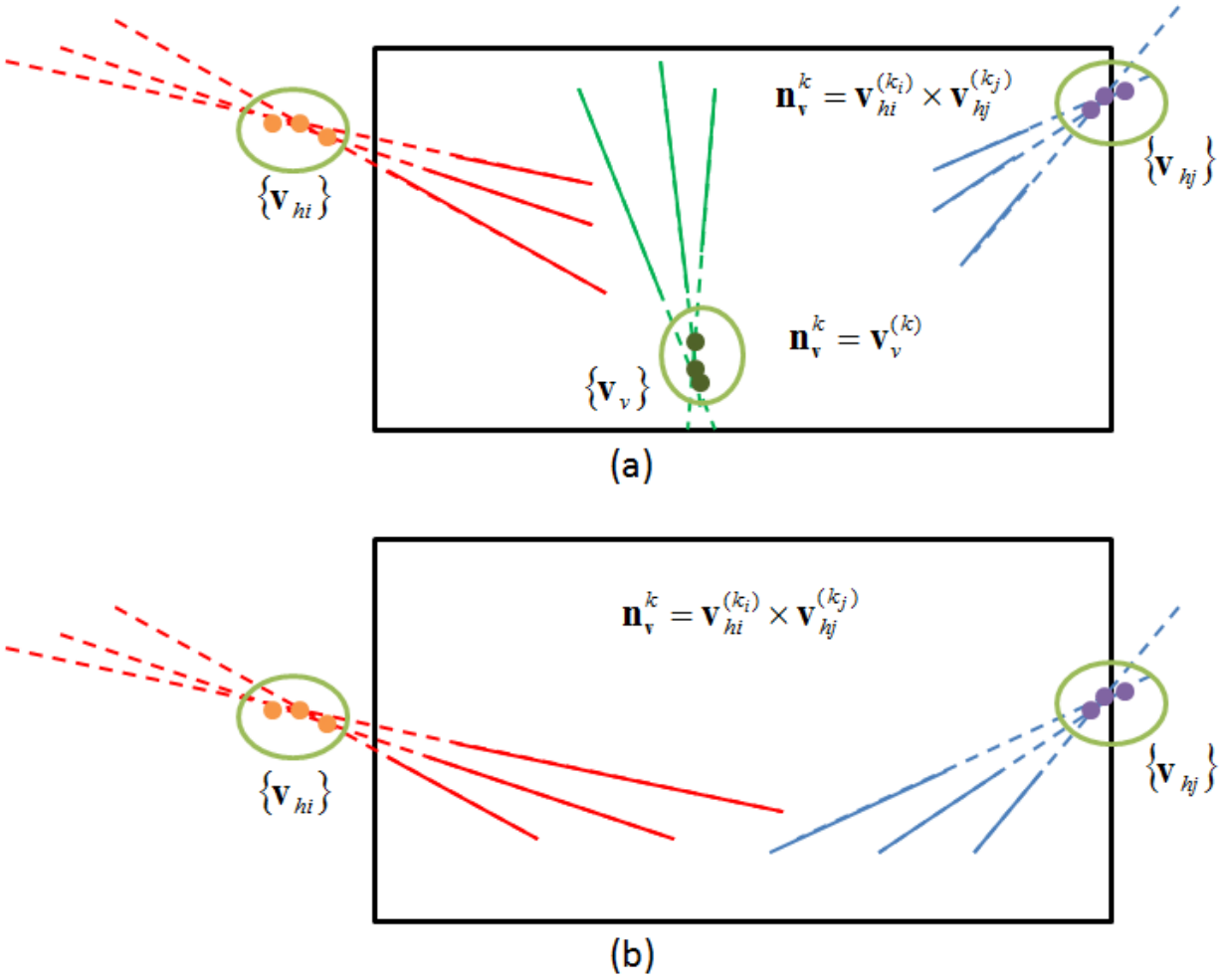

Figure 4.4: Estimating the vertical direction of a surface region. Here, $\mathbf{n}_{\mathbf{v}}^{k}$ is the k-th sample of the vertical direction $\mathbf{n}_{\mathbf{v}}$ of a surface. $\{\mathbf{v}_v\}$ and $\{\mathbf{v}_{hi}\}$ are the set of line-intersection points belonging to the vertical and i-th horizontal vanishing points, respectively. $\mathbf{v}_{hi}^{(k_i)}$ is the k_i-th line-intersection point within $\{\mathbf{v}_{hi}\}$. The meaning is similar for $\mathbf{v}_{v}^{(k)}$. (a) Vertical surface. Its vertical direction is derived from the vertical vanishing point and pairs of horizontal vanishing points. (b) Horizontal surface. Its vertical direction is derived from pairs of horizontal vanishing points.

An example of each object/surface candidate producing a non-parametric distribution of its vertical direction is shown in Figure 4.5. We can see that the distribution of vertical direction of a pedestrian is elongated along the pitch direction, which is expected. Also, the estimate from the horizontal surface is more likely to be inaccurate due to the lack of the vertical vanishing direction and the lack of more than one dominant horizontal vanishing directions.

Overall, as is shown in Figure 4.6b, local estimates look messy, partly due to the existence of several false detections, and partly due to the estimation noise in true detections. In the next section, we discuss how to generate at least one good hypothesis from those messy estimates.

4.3.2 Pooling Local Estimates to Generate Hypotheses using Generalized RANSAC

One of the keys for RANSAC to succeed is that at least one hypothesis should be close to the ground truth. In our case, a single object/surface candidate (*i.e.* observation) alone can generate a hypothesis of the global 3D geometries. Ideally, we could simply use a single observation (*i.e.* the minimal set) to generate a hypothesis. However, as single observations (even if they are true detections) tend to be noisy, it is likely that none of the hypotheses generated by the minimal set is close to the ground truth. On the other hand, if we use all the observations with equal weights, false detections would corrupt the hypothesis. Therefore, we propose a generalized RANSAC algorithm to both inhibit outliers and reduce noise.

After each object/surface candidate has estimated a distribution of the global 3D geometries, we generate a set of mixed distributions by mixing individual distributions together with randomly generated weights. For each mixed distribution, we find its modes using the mean-shift algorithm [8] and take those modes as hypotheses. When the set of mixed distributions is large enough, at least one of them would mostly come from multiple valid object/surface candidates. Furthermore, by finding modes of their mixed distribution (which is equivalent to averaging) we also reduce the noise level.

Figure 4.5: Examples of object/surface candidates estimating distributions of ground plane orientation and gravity direction. Counter-clockwise, the four images show the cases of a car, a pedestrian, the horizontal surface, and the vertical surface, respectively. The distributions are represented by the dense blue points on the unit sphere shown in the upper right corner of each image. The peak of each distribution is illustrated by a red dot, and its corresponding horizon is displayed by a yellow (for ground) or magenta (for gravity) line in the image. Top and bottom landmarks of the car and pedestrian are indicated by the magenta and green dots, respectively. Best viewed on screen.

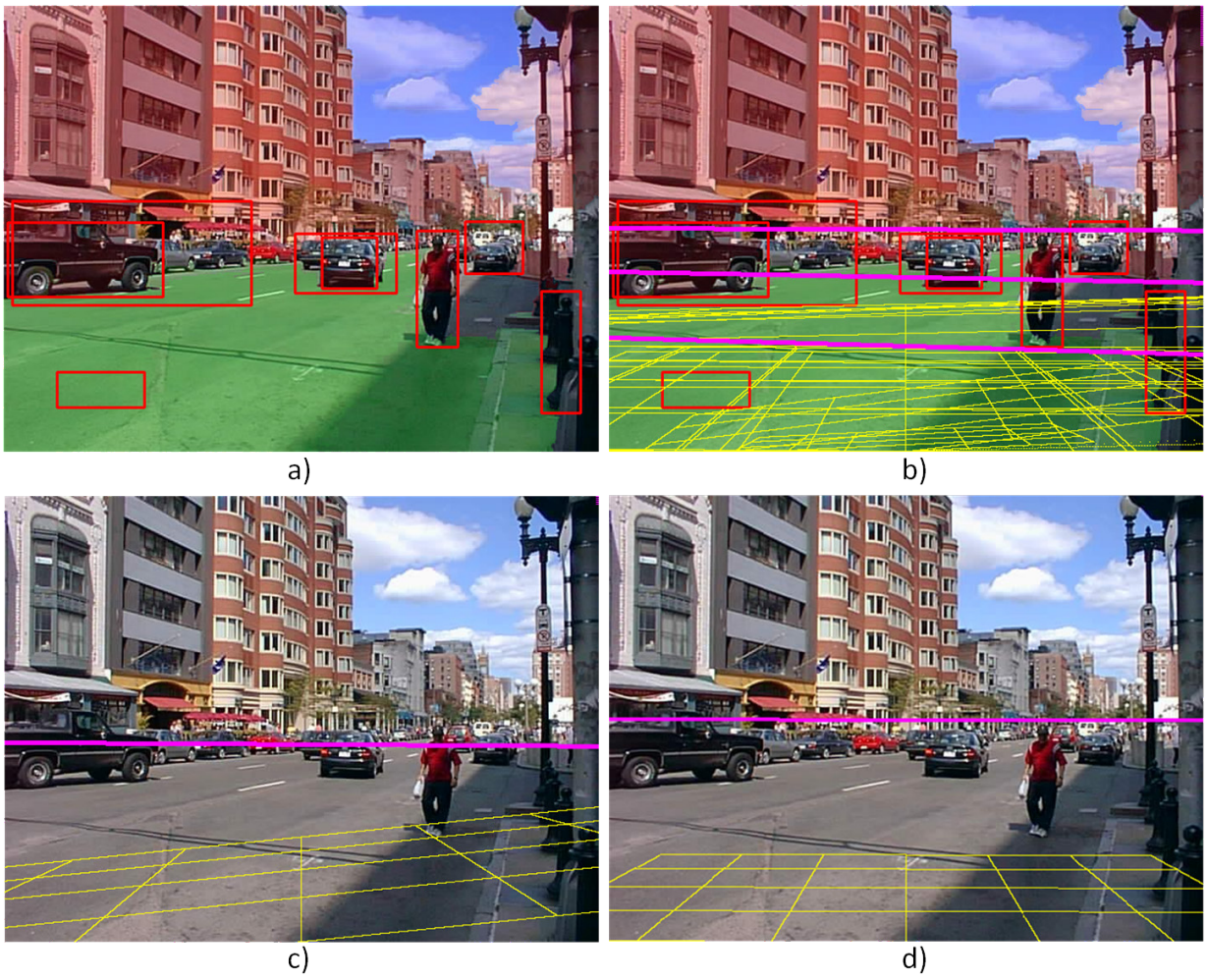

Figure 4.6: Generate hypotheses of global 3D geometries from object/surface candidates. a) Object/surface candidates. Here, red and green shades indicate vertical and horizontal surface candidates, respectively. b) Estimates of global 3D geometries given by individual object/surface candidates. Here, magenta lines represent gravity horizons, and yellow grids indicate ground planes, where the grid size is 1m. For display purposes, only the mode of each non-parametric distribution is shown. c) A bad hypothesis. d) A good hypothesis.

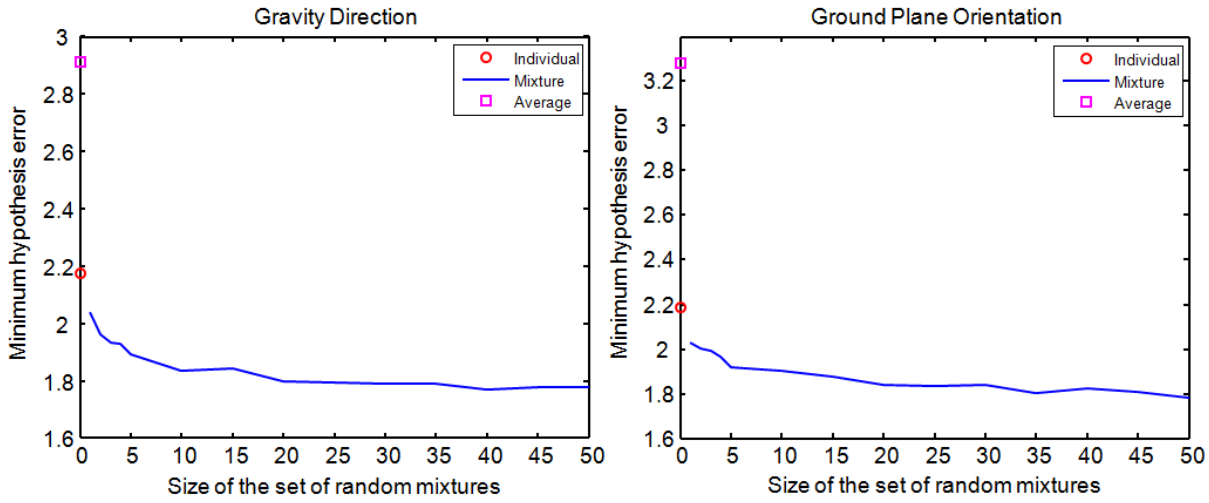

Figure 4.7: Comparing different methods of generating the hypotheses of gravity direction and ground plane orientation. The y-axis is the smallest error among all the hypotheses generated by a given method, and its unit is degree. The red circle shows the result of directly using the modes from each individual distribution. The magenta square indicates the result of computing the modes of the average of all the individual distributions. The blue curve plots the result of our generalized RANSAC approach which extracts modes from a set of random mixtures of individual distributions.

To verify this claim, we compare the error of hypothesis generation in three cases: 1) obtaining hypotheses by directly using the modes from each individual distribution estimated by each object/surface candidate; 2) obtaining hypotheses by computing the modes of the average distribution over *all* the distributions estimated by object/surface candidates; 3) obtaining hypotheses using our generalized RANSAC approach. The error of hypothesis generation is defined as the difference between the ground truth and the *best* hypothesis among all the hypotheses generated.

We perform experiments on the 100 images that are provided with the ground-truth horizon in the LabelMe dataset [34]. The row index of the ground-truth horizon is converted to the ground-truth orientation vector which is compared against the hypotheses of gravity direction/ground orientation by measuring the angle between them. For each image, we vary the size of the set of random mixtures from 1 to 50.

The results of our experiment are plotted in Figure 4.7. We can see that the generalized RANSAC approach (blue curve) has the smallest error. Using the average of all the observations

(the magenta square) produces the worst performance due to the corruption by many invalid candidates. Directly using each individual observation (the red circle) performs better, yet it still yields less accurate hypotheses than our generalized RANSAC method due to the noise in individual estimation. Also, in our method, as the size of the set of random mixtures grows, the error decreases. This is expected, because the ground truth is more likely to be covered when we try more combinations. In our experiment, generating 50 random mixtures is sufficient.

We observe that when we generate 50 random mixtures, the number of hypotheses for gravity direction or ground plane orientation is usually less than 50 after near-duplicate hypotheses are combined. This number is very small compared to the search space we consider – the entire upper dome of the Gaussian sphere. Even with such a small number of hypotheses, our method is able to keep the error less than 1.8 degrees.

In implementation, we first generate hypotheses of $\mathbf{n_g}$ and $\mathbf{n_p}$. Then for each 2-tuple hypothesis $(\tilde{\mathbf{n}}_\mathbf{g}, \tilde{\mathbf{n}}_\mathbf{p})$, we compute the distribution of h_p from each object candidate according to the algorithm in Figure 4.3, and generate hypotheses of h_p by randomly mixing those distributions and finding modes. The resulting hypotheses of h_p supplement the original 2-tuple hypothesis to form a set of 3-tuple hypotheses $(\tilde{\mathbf{n}}_\mathbf{g}, \tilde{\mathbf{n}}_\mathbf{p}, \tilde{h}_p)$. Note that the proposed hypotheses are continuous despite the discrete grids $\mathbf{G_n}$ and $\mathbf{G_h}$.

Two qualitative examples of bad and good hypotheses from the hypothesis set are shown in Figure 4.6c and d.

4.4 Evaluating Global 3D Geometry Hypotheses

Given a global 3D geometry hypothesis $(\tilde{\mathbf{n}}_\mathbf{g}, \tilde{\mathbf{n}}_\mathbf{p}, \tilde{h}_p)$, we evaluate its quality by measuring how well it is supported by object/surface candidates after excluding the influence of outliers. For this purpose, we evaluate geometric compatibilities of each object/surface candidate, and employ a CRF to infer the validity of each candidate. The optimal score of the objective function used in the CRF inference is regarded as the quality of the current global 3D geometry hypothesis.

4.4.1 Types of 3D Geometric Context

Before evaluating the geometric compatibilities of local entities, we first describe different types of 3D geometric context under which invalid local entities are identified.

We consider four types of 3D geometric context (please see Figure 4.8 for illustrations):

1) **Common ground.** Objects supported by the common ground cannot have arbitrary pose and/or location. For an object like a pedestrian that has a point contact with the ground, its bottom location is constrained to be on the ground plane; for an object like a car that has a planar contact with the ground, not only is its bottom location constrained to be on the ground plane, but its pose is constrained to be the same as the ground plane orientation as well. Also, for horizontal surfaces corresponding to the ground plane, their orientation should be consistent with the ground plane orientation.

2) **Gravity direction.** The orientation of some upright objects such as pedestrians usually follows the gravity direction. Also, for vertical surfaces corresponding to building facades, their orientation should be consistent with the gravity direction.

3) **Depth ordering.** Assume all objects are opaque. Objects with a greater depth must be at least partially visible for them to be detected. If an object farther away from the camera is mostly or even totally occluded by a closer object, then the two objects are incompatible with each other.

4) **Space occupancy.** The same space volume can be occupied by only one object. If the footprints of two objects on the ground plane overlap with each other, then they are incompatible with each other.

The first two types of geometric context (*i.e.* common ground and gravity direction) affect local entities everywhere in the scene, and are therefore global geometric context. The last two types of geometric context (*i.e.* depth ordering and space occupancy) only affect neighboring (either in 2D or 3D) local entities, and are therefore local geometric context.

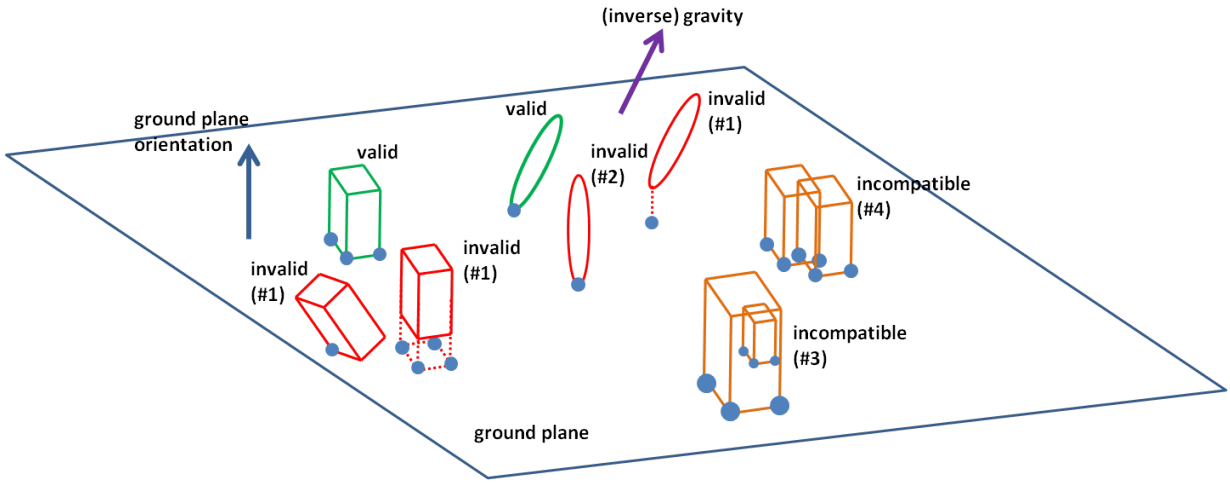

Figure 4.8: Different types of geometric context. Objects consistent with all types of geometric context are displayed in green. Invalid and incompatible objects that violate at least one type of geometric constraints are shown in red and orange. Here, cubes represent cars and ellipsoids represent pedestrians. Blue dots indicate ground contact points. The specific type of geometric constraint violated is indicated by the number after "#".

4.4.2 Computing Geometric Compatibilities

Having identified different types of geometric context, we use the geometric relationship equations in Section 4.2 to quantitatively compute the geometric compatibilities of local entities.

Global Geometric Compatibility

For an individual object candidate, we use two sources of geometric constraints to compute its global geometric compatibility. The first source involves non-deterministic regressor outputs. We compute the pitch $\tilde{\theta}$ and roll $\tilde{\gamma}$ of the object candidate according to constraint 4.4, where $\mathbf{n_v} = \tilde{\mathbf{n}}_\mathbf{g}$ for pedestrians and $\mathbf{n_v} = \tilde{\mathbf{n}}_\mathbf{p}$ for cars. $\tilde{\theta}$ and $\tilde{\gamma}$ are then checked against the regressor outputs from constraints 4.8 and 4.9. The resulting compatibility score is

$$s_{g1} = \exp\{-\frac{(\tilde{\theta} - \theta_0)^2}{2\sigma_\theta^2}\} \cdot \exp\{-\frac{(\tilde{\gamma} - \gamma_0)^2}{2\sigma_\gamma^2}\} - 0.5, \tag{4.14}$$

where θ_0, γ_0 and σ_θ^2, σ_γ^2 are the mean and variance of the pitch and roll regressor outputs, respectively.

The second source of geometric constraints involves deterministic geometric relationships. Given the ground plane hypothesis $\tilde{\mathbf{n}}_\mathbf{p}$ and $\tilde{h}_p$, we compute the bottom landmark depth d_b using constraint 4.6. This gives us the 3D coordinate $\mathbf{X_b}$ of the bottom landmark. According to constraint 4.3, we search for the optimal 3D coordinate $\mathbf{X_t}$ of the top landmark along its line of sight using gradient descent, such that the direction of $\mathbf{X_t} - \mathbf{X_b}$ has the best match with $\tilde{\mathbf{n}}_\mathbf{v}$ (which equals $\tilde{\mathbf{n}}_\mathbf{g}$ for pedestrians or $\tilde{\mathbf{n}}_\mathbf{p}$ for cars). The length of $\mathbf{X_t} - \mathbf{X_b}$ is checked against the prior knowledge of the real world height H_t of the top landmark. Denote the angle between $\tilde{\mathbf{n}}_\mathbf{v}$ and the direction of $\mathbf{X_t} - \mathbf{X_b}$ as δ, then the resulting compatibility score is

$$s_{g2} = \exp\{-\frac{\delta^2}{2\sigma^2}\} \cdot \exp\{-\frac{(\|\mathbf{X_t} - \mathbf{X_b}\| - H_t)^2}{2\sigma_H^2}\} - 0.5, \tag{4.15}$$

where σ_H^2 is the variance of H_t according to prior knowledge, and σ is a parameter set as 20 degrees.

Estimate of the bottom landmark image coordinate $\mathbf{x_b}$ could be noisy (please see Figure 3.3), and a small error in $\mathbf{x_b}$ could translate into a huge displacement in 3D coordinate when $\mathbf{x_b}$ is near the ground horizon, significantly decreasing the compatibility score s_{g2} of an otherwise valid object. To solve this problem, we compute s_{g2} at the nodes of a 5-by-5 grid centered at $\mathbf{x_b}$, and the highest s_{g2} score is used. Here the grid spacing is 5% of bounding box size.

The final compatibility score s_g for an individual object candidate is the average of s_{g1} and s_{g2}.

For an individual surface candidate, its global geometric compatibility also comes from two sources. Firstly, we check how well the vertical orientation distribution produced by the surface candidate agrees with $\tilde{\mathbf{n}}_\mathbf{g}$ (for vertical surface) or $\tilde{\mathbf{n}}_\mathbf{p}$ (for horizontal surface). This produces a compatibility score s_{g1} with range between -0.5 and 0.5. Secondly, we check the plausibility of the location of the surface candidate with respect to the ground horizon in the image. For the horizontal surface candidate, denote the proportion of the surface region *above* the ground horizon as r_h, then the compatibility score s_{g2} is $-r_h$ with range between -1 and 0. For the verti-

cal surface candidate, this type of compatibility does not apply, as it usually straddles across the horizon. The final compatibility score s_g is the average of s_{g1} and s_{g2} for the horizontal surface candidate, and is s_{g1} for the vertical surface candidate.

Local Geometric Compatibility

To compute the level of depth ordering conflict, for each pair of object candidates i and j whose bounding boxes overlap, we check if the bounding box of the farther candidate j is located mostly or completely within the bounding box of the closer candidate i. If yes, then they are unlikely to co-exist due to the occlusion conflict resulting from the depth ordering, as is the case of candidates 3 and 4 in Figure 2.2. Therefore, we define the pairwise compatibility score $s_{ij}^{(ocl)}$ related to depth ordering as

$$s_{ij}^{(dep)} = -(|R_{ij}|/|R_j|)^{\lambda}, \tag{4.16}$$

where $|R_{ij}|$ is the overlapping area of candidates i and j, $|R_j|$ is the area of candidate j, and λ is a parameter set as 5.

To compute the level of space occupancy conflict, for each pair of object candidates, we compute the overlap of their footprints on the ground plane. They are unlikely to co-exist due to space occupancy conflict if their footprint overlap is significant, as is the case of candidates 5 and 6 in Figure 2.2. The pairwise compatibility score $s_{ij}^{(ocp)}$ related to space occupancy is therefore defined as

$$s_{ij}^{(ocp)} = -|R'_i \cap R'_j|/|R'_i \cup R'_j|, \tag{4.17}$$

where $|R'_i \cap R'_j|$ and $|R'_i \cup R'_j|$ are the intersection and union areas of the footprints of candidates i and j, respectively. The footprint of a car candidate is obtained by estimating its yaw angle τ using Equation 4.10 and rotating the rectangle around the ground coordinate of the car candidate

by τ. The size of the rectangle is the typical real-world length and width of a car.[5] The footprint of a pedestrian candidate is simply a circle around its ground coordinate. The diameter of the circle is the typical real-world width of a pedestrian.

4.4.3 Inferring Candidate Validity with a CRF

We construct a CRF over the object/surface candidates to infer their validity. Each candidate forms a node, and two object candidates have an edge between them if their bounding boxes and/or footprints overlap. The objective function the CRF attempts to maximize is

$$V(\mathbf{o}) = \sum_{k=1}^{2} \omega_k^{(s)}(o_k^{(s)}) + \sum_i \omega_i(o_i) + \sum_{(i,j)\in\mathbf{E}} \varphi_{ij}(o_i, o_j). \tag{4.18}$$

Here, o is the binary validity indicator. $\omega_1^{(s)}(o_1^{(s)})$, $\omega_2^{(s)}(o_2^{(s)})$, and $\omega_i(o_i)$ are the unary potentials for the vertical surface candidate, horizontal surface candidate, and object candidate i, respectively. Their values are defined as $\omega(o=1) = s_g + (s_d - 0.5)$ and $\omega(o=0) = 0$, where s_g is the compatibility score defined in the previous section, and s_d is the segmentation confidence or detection confidence returned from the surface segmentation algorithm or object detection algorithm. $\varphi_{ij}(o_i, o_j)$ is the pairwise potential between object candidates i and j. Its value is s_{ij} when both o_i and o_j are 1; otherwise its value is 0. Here, s_{ij} could either be $s_{ij}^{(dep)}$ or $s_{ij}^{(ocp)}$ depending on the type of the edge. If both of them exist, then s_{ij} is the smaller of the two.

After the inference is completed, the quality of the current global 3D geometry hypothesis is the maximum value V^* of the objective function. After all the hypotheses are evaluated, the one with the highest quality is selected. This optimal hypothesis is further refined by the valid object/surface candidates associated with it.

[5] We have also tried to obtain the footprint of a car candidate by mapping wheel locations in the image to the ground plane. It offers no improvement and is numerically less stable.

4.5 Experiments

We evaluate our approach on the test dataset compiled by Hoiem *et al.* [34] which contains 422 random outdoor images from the LableMe dataset [57]. Those images cover a multitude of outdoor urban scenes and include a wide variety of object pose and size, making the dataset very challenging. The dataset contains 923 cars and 720 pedestrians in total.

Hoiem *et al.* also collects a training/validation dataset containing 51 images. We use this set to train our pose and landmark regressors. The prior distribution of pedestrian height in our experiment follows $N(H_p; 1.7, 0.09)$, and the prior distribution of wheel height is $N(H_w; 0.6, 0.25)$. The typical length and width of a car are 3m and 2m, respectively, and the typical width of a pedestrian is 0.5m. When proposing hypotheses from distributions, 50 random mixtures are usually enough, and the total number of hypotheses to evaluate is in the hundreds. For each global geometry hypothesis, we use max-product belief-propagation [30] to do inference over the CRF. We run our algorithm under multiple focal lengths centered at the value estimated by the algorithm described in Section 3.5, and the one that yields the highest value of V^* is adopted. It takes less than 10 minutes to process a 640-by-480 image with Matlab code.

Comparison with The State of The Art: Using the top-performing Deformable Part Model (DPM) [15] as the baseline detector, we compare the object detection performance of our approach with Hoiem's algorithm in [34]. The result of Hoiem's algorithm is generated by running their published code with the DPM detector outputs. The ROC curves are plotted in the left panel of Figure 4.9. The average precision (AP) of our approach is 50.5%, achieving a boost of more than 10% over the AP of the baseline detector at 40.1%. Surprisingly, Hoiem's algorithm performs worse than the baseline in the realm of lower false positive rates, yielding an AP of 30.8%. We observe that Hoiem's model is not effective for car candidates returned by the DPM detector. This is probably because Hoiem's algorithm takes the bounding box height as the object height in the image. This approximation is poor when a non-planar object, such as a car, is viewed from a non-zero pitch angle. As our algorithm explicitly estimates the landmark locations in the

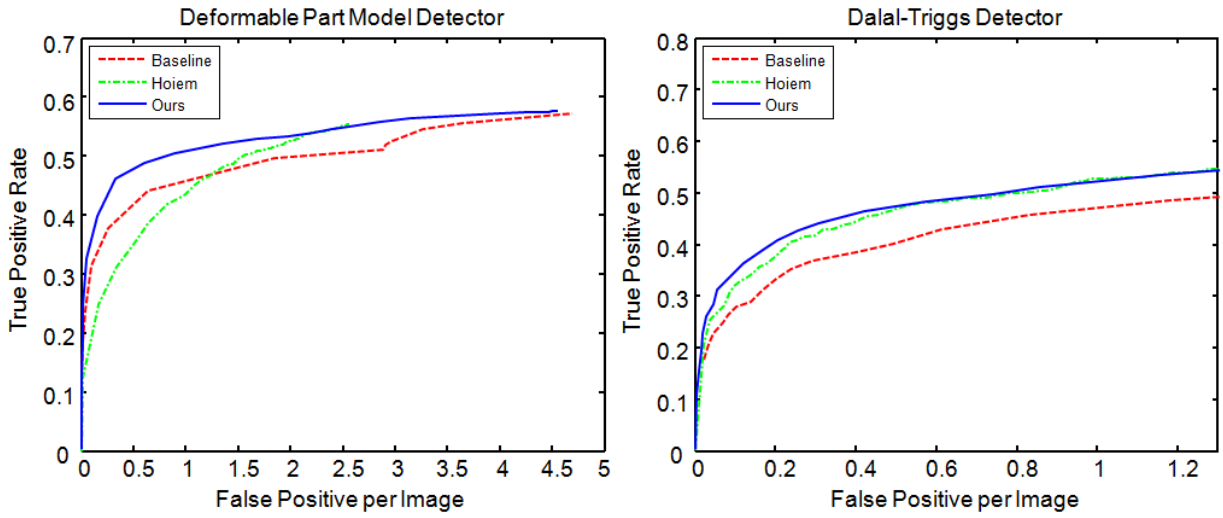

Figure 4.9: Comparison of object detection performance. The baseline detector in the left panel is DPM [15], and in the right panel is Dalas-Triggs [9]. Our algorithm significantly boosts the performance over the baseline detector. It also outperforms Hoiem's algorithm [34] while making less assumptions.

image, it does not have this problem. Interestingly, Hoiem's algorithm performs very well when the Dalal-Triggs detector [9] is employed, as is shown in the right panel of Figure 4.9. This is probably because the height of the bounding boxes returned by the Dalal-Triggs detector are more regulated due to their fixed aspect ratio.

In addition to comparing object detection, we also evaluate global geometry estimation. The dataset provides the ground truth of horizon in the form of the row index where the horizon is located. It does not distinguish between gravity and ground horizons, since the two are almost the same for most of the images in the dataset. After converting the row index of a horizon to the corresponding orientation vector, we compute the error of an estimated gravity direction (or ground plane orientation) by measuring the angle between it and the ground truth orientation vector. The results are shown in Figure 4.10. Despite not using any prior, our method has a smaller error in horizon estimation than Hoiem's algorithm. The estimated ground plane height by our method is centered around 1.5 meters, close to the typical eye level.

It is worth noting that, unlike Hoiem's algorithm, we do not assume the ground plane is

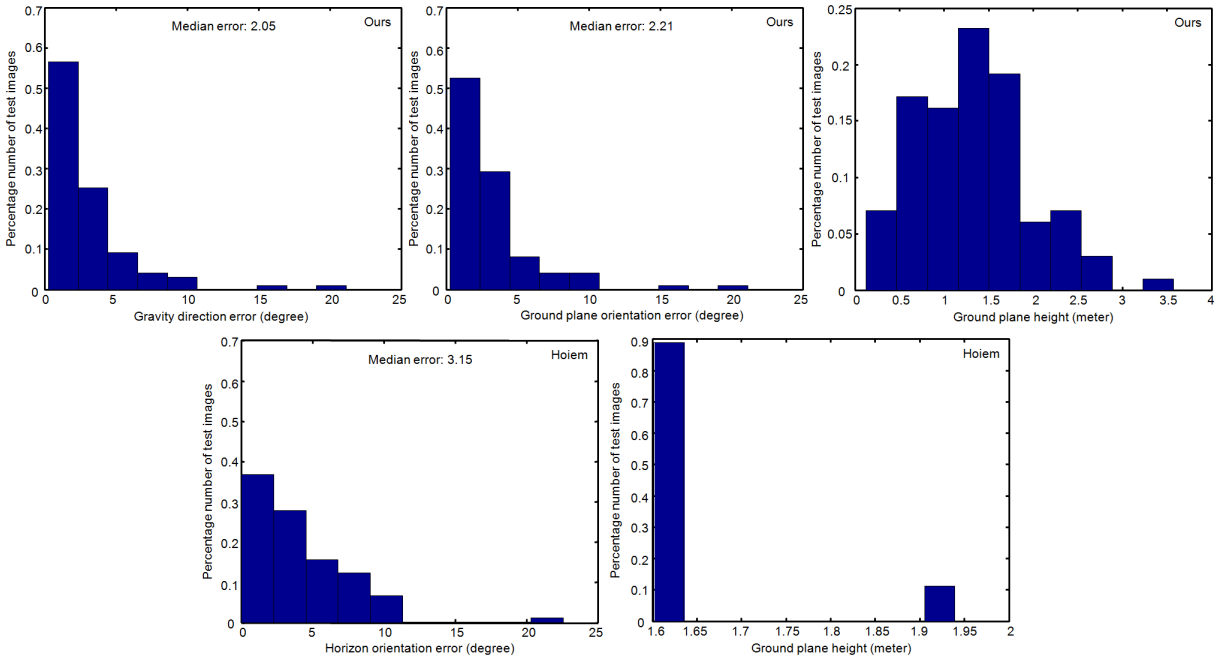

Figure 4.10: Comparison of global geometry estimation performance. The first row shows the distributions of gravity direction error, ground orientation error, and ground height from our algorithm. The second row shows the results of Hoiem's algorithm [34]. Our algorithm has a smaller error in horizon estimation. We are not able to compute the error of the ground plane height estimation due to the lack of ground truth. However, both the algorithms peak at around 1.5 - 1.6 meters, roughly corresponding to the eye level. Best viewed on screen to zoom in.

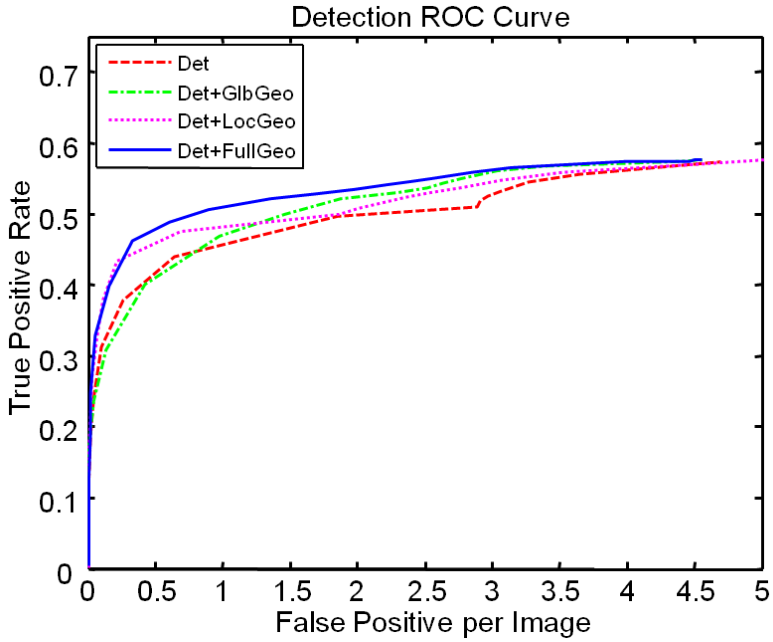

Figure 4.11: Contribution of individual components. Here, "Det"shows the result of the DPM baseline detector; "Det+GlbGeo"shows the result of including global geometric context alone; "Det+LocGeo"shows the result of including local geometric context alone; "Det+FullGeo"shows the result of our full system using both types of context.

perpendicular to the gravity direction, or has a zero roll and a small pitch. Even with a greater flexibility, our approach still outperforms Hoiem's algorithm both in object detection and global geometry estimation, on the test dataset that largely satisfies those assumptions.

As we do not have access to the code of the algorithms proposed in [3] or [66], we are not able to directly compare with their performance. Yet according to what the authors report in [3], their method does not perform as well as Hoiem's algorithm on a subset of the 422-image test dataset. In [66], the authors use their own baseline detector and a different subset of images. As a result, we could only compare the performance gain over the baseline. The algorithm in [66] achieves a gain of 5.1% in average precision, while our approach achieves a gain of 10.4%.

Global and Local 3D Geometric Context: Different from existing works, we use both global and local 3D geometric context when inferring the validity of object candidates. Global geometric context checks the geometric consistency of an individual object candidate with the

global geometry hypothesis, while local geometric context checks the geometric consistency of nearby object candidates. To evaluate the contributions of the two types of geometric context, we run our algorithm with only one of them enabled. The results are plotted in Figure 4.11. We can see that both the global and local 3D geometric contexts enhance detection performance, and the highest gain is achieved when they are applied simultaneously.

Benefit is mutual: Not only does 3D geometric context enhance object detection performance, but coherent object detection in turn improves the estimation of gravity and ground horizons. To verify this argument, we estimate the gravity direction and ground plane orientation from vertical and horizontal surfaces alone. The median estimation error is 2.62 degrees for gravity direction and 4.85 degrees for ground plane orientation. Also, when we estimate the gravity direction and ground plane orientation using the vanishing lines from the entire building and road regions (please see Section 3.5), the errors are 2.53 degrees for gravity direction and 2.94 degrees for ground plane orientation. By contrast, the errors of our full system are 2.05 and 2.21 degrees, respectively.

Qualitative evaluation: Several examples of the object detection results of our algorithm are shown in Figure 4.12. Please refer to the figure caption for the meanings of different types of boxes. We can see that some false detections from DPM are rejected due to inconsistency with global 3D geometries (*e.g.* the huge "pedestrian"in (a)); some false detections from DPM are rejected due to inconsistency with local 3D geometries (*e.g.* the rejected "cars"in (c)). Also, some true objects missed by DPM are recovered due to their geometric consistency (*e.g.* the pedestrians in (b)). The case when gravity direction and ground orientation do not agree is shown in (f), where the ground horizon is illustrated by the thick yellow line. It deviates from the gravity horizon. A failure case is shown in (k), where a "car"is mistakenly recovered due to its high geometric compatibility. In another failure case shown in (l), two truncated cars are wrongly rejected because our regressors are not trained on truncated cars.

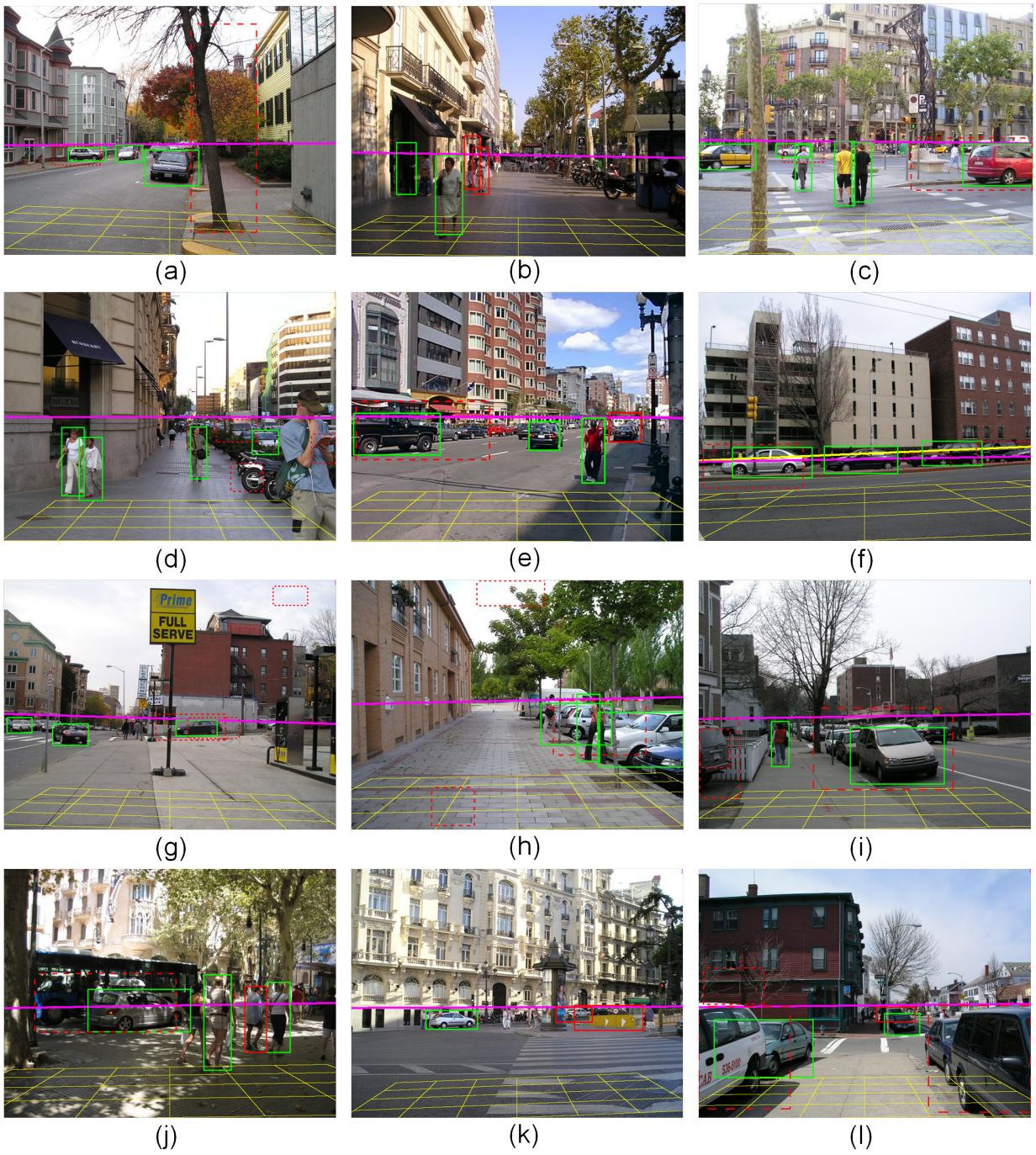

Figure 4.12: Examples of our coherent object detection results. **Solid green box**: the object is detected by both the DPM detector and our algorithm. **Solid red box**: the object is missed by the DPM detector but recovered by our algorithm. **Dotted red box**: the object is detected by the DPM detector but rejected by our algorithm. **Magenta line**: gravity horizon. **Yellow grid**: ground plane where the grid spacing is 1m. The detection threshold of both the DPM detector and our algorithm is set as 0.5.

4.6 Conclusion

We have presented an object detection algorithm that ensures geometric coherence in the 3D world. Compared with existing approaches, the major contributions of our work include 1) a more flexible modeling of the scene that treats gravity direction and ground orientation separately, 2) a more systematic representation of the geometric relationships between scene and objects, 3) a generalized RANSAC algorithm that enables both outlier suppression and noise reduction in hypothesis generation, 4) incorporating both global and local 3D geometric context with a CRF, 5) including surface regions in estimating and evaluating global 3D geometry. Due to these factors, our algorithm achieves a superior performance on a challenging dataset.

Chapter 5

Inferring 3D Layout of Building Facades with 3D Geometric Context

In the previous chapter, we treat the entire building region as a single vertical surface region. To achieve a better understanding of the scene, it is desirable to recover the 3D layout of the building facades as well.

In this chapter, we present a novel algorithm that infers the 3D layout of building facades from a single image of urban scenes. Different from existing methods that only yield coarse orientation labels[23, 33, 35, 36] or qualitative block approximations [25], our algorithm quantitatively reconstructs building facades in 3D space using a set of planes mutually related by 3D geometric constraints. Each plane is characterized by a continuous orientation vector and a distance distribution. An optimal solution is reached through inter-planar interactions. Due to the quantitative and plane-based nature of our geometric reasoning, our model is more expressive and informative than existing approaches. Experiments show that our method compares competitively with the state of the art on both 2D and 3D measures, while yielding a more informative interpretation of the 3D scene behind the image.

Note that in this chapter, we do not consider any information from objects present in the scene. Joint reasoning of both facades and objects will be discussed in the next chapter.

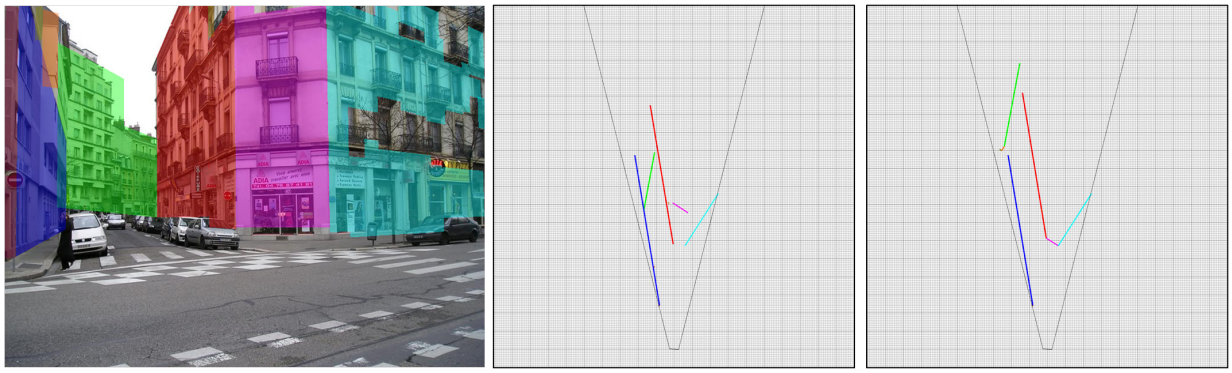

Figure 5.1: Our algorithm detects building facades in 2D image, decomposes them into distinctive planes of different 3D orientations, and infers their optimal depth in 3D space based on cues from individual planes and 3D geometric constraints among them. Left: Detected facade regions are covered by shades of different colors, each color representing a distinctive facade plane. Middle/Right: Ground contact lines of building facades on the ground plane before/after considering inter-planar geometric constraints. The coarser grid spacing is 10m.

5.1 Overview

Given a single image of an urban scene, automatically inferring the underlying 3D layout of building facades in the scene would significantly benefit many tasks in fields such as autonomous navigation and augmented reality. It goes beyond depth map estimation [48, 58, 60], because it provides a richer understanding of the scene, such as camera pose, locations of planes and blocks, and how they are related with each other in 3D [25].

Nevertheless, recovering building facades in 3D space is a particularly challenging task. The difficulty comes from the fact that building facades could have highly flexible combinations in 3D space, and therefore do not have a definitive shape either in 2D image or 3D space. In fact, they are "stuffs"rather than "objects", and are thus unable to be located by a single 3D coordinate. An example is shown in Figure 5.1, where the facades do not even follow the Manhattan world assumption.

Although we cannot locate building facades the same way we locate objects, we observe that unlike other regions such as trees or sky, building facades are more structured, and can be decomposed into a set of planes that can be represented quantitatively. The orientations and

locations of those planes are mutually constrained by their 3D geometric relationships derived from physical feasibility. In fact, modeling building facades as a set of planes with continuous orientations and quantitatively reasoning over their 3D locations using inter-planar geometric constraints would produce a richer interpretation of the facade scene than existing pixel/segment based approaches [33, 35, 48] and block-based approaches [25]. Our approach provides critical scene understanding information (*e.g.* quantitative orientation, depth, and relationships of facade planes) that existing algorithms do not provide.

The main contributions of this work are as follows.

1) We propose a plane-based fully quantitative model to infer the 3D layout of building facades, where each plane is represented by a continuous orientation vector and a distribution of depth values.
2) In such a model, multiple cues, such as semantic segmentation, surface layout, and vanishing lines, are utilized to detect and decompose the building region into distinctive planes.
3) The quality of an individual candidate plane is determined by its compatibility with both 2D evidence from image features and 3D evidence such as camera and building height;
4) We model different types of 3D geometric relationships among candidate planes, and apply a CRF to both determine their validity and infer their optimal depths.
5) We do not assume ground is horizontal or buildings are vertical with respect to the camera, nor do we assume buildings are perpendicular to the ground. We also do not make the Manhattan world assumption.

5.2 Plane-based 3D Modeling of Building Facades

5.2.1 Problem Formulation

The problem our model addresses is to detect a set of distinctive facade planes and estimate their 3D orientations and locations given a single image of urban scene. Here, a distinctive facade plane is defined as a facade plane whose orientation is different from the orientations of its adjacent facade planes; otherwise, two adjacent facade planes with the same orientation will be merged.

More formally, our model infers the optimal 3D layout of building facades by maximizing the following objective function:

$$V(\mathbf{o}, \mathbf{n_s}, \mathbf{d_s}, \mathbf{x_s}|I, h_p, \mathbf{n_p}, f, H_f) = \sum_{i \in \mathbf{P}} \omega(o_i, \mathbf{n_{si}}, d_{si}, \mathbf{x_{si}}|I, h_p, \mathbf{n_p}, f, H_f)$$
$$+ \sum_{(i,j) \in \mathbf{P_v}} \varphi_v(o_i, o_j, d_{si}, d_{sj}, \mathbf{n_{si}}, \mathbf{n_{sj}}, \mathbf{x_{si}}, \mathbf{x_{sj}}|h_p, \mathbf{n_p}, f) + \sum_{(i,j) \in \mathbf{P_o}} \varphi_o(o_i, o_j, d_{si}, d_{sj}, \mathbf{n_{si}}, \mathbf{n_{sj}}, \mathbf{x_{si}}, \mathbf{x_{sj}}|h_p, \mathbf{n_p}, f)$$
$$+ \sum_{(i,j) \in \mathbf{P_a}} \varphi_a(o_i, o_j, d_{si}, d_{sj}, \mathbf{n_{si}}, \mathbf{n_{sj}}, \mathbf{x_{si}}, \mathbf{x_{sj}}|h_p, \mathbf{n_p}, f), \quad (5.1)$$

where $\mathbf{o}, \mathbf{n_s}, \mathbf{d_s}, \mathbf{x_s}$ are variables that characterize facade planes. More specifically, for each plane i, $o_i, \mathbf{n_{si}}, d_{si}$ and $\mathbf{x_{si}}$ represent its validity (binary indicator), orientation (continuous vector), distance from camera center (continuous scalar), and spatial extent (continuous coordinates specifying the corners of the plane in the image). Please see Figure 3.1 for an illustration of these variables.[1] Note that as our model does not know in advance the total number of distinctive facade planes in the scene, it has to detect candidate planes with high recall and infer their validity after obtaining contextual information from other planes. For valid planes, it also needs to infer their optimal 3D geometries. Our algorithm that detects candidate facade planes will be detailed

[1]For clarity we do not show $\mathbf{x_s}$ in the figure. They are the projection of $\mathbf{X_s}$ onto the image plane.

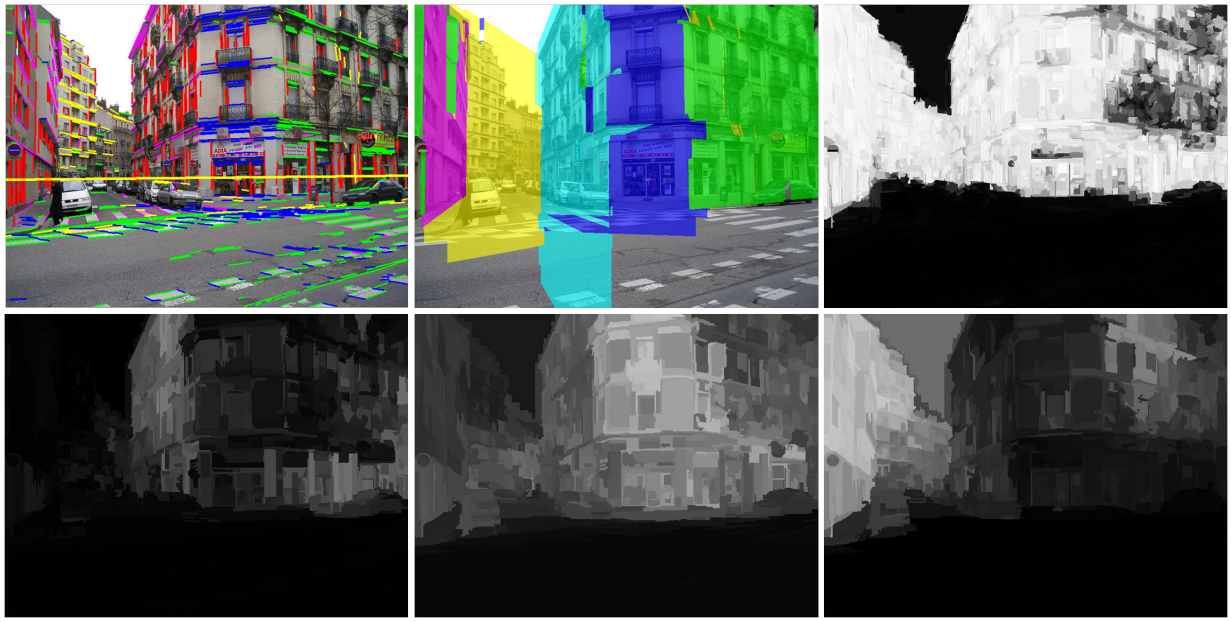

Figure 5.2: Image features utilized by our method. Top row from left to right: 1) vanishing lines and ground horizon, where the thick yellow line is the ground horizon, and the thiner lines in different colors represent vanishing lines belonging to different vanishing points, 2) orientation map from the line sweeping algorithm where each color represents an orientation corresponding to a specific horizontal vanishing point, 3) semantic segmentation score for the "building"region. Bottom row from left to right: surface orientation scores for "planar left", "planar center", and "planar right".

in the next section. The optimization problem is conditioned upon image features I, ground distance h_p from the camera center, ground orientation $\mathbf{n_p}$ with respect to the camera, focal length f, and facade height H_f. In this work, $\mathbf{n_p}$ and f are automatically computed in a preprocessing stage that will be discussed in the next section, and we assume a typical ground distance $h_p = 1.6$m. H_f is obtained from the prior knowledge of a typical range of facade heights.

The first term in the objective function is a unary potential for each individual plane, and it is summed over all candidate planes $\mathbf{P}$. The remaining three terms are pairwise potentials for planes with mutual constraints, and they are summed over a subset $(\mathbf{P_v}, \mathbf{P_o}, \mathbf{P_a})$ of candidate planes involved in those constraints. In the following sections, we describe these potentials in more detail.

5.2.2 Individual Compatibility

The unary potential ω is based on the product of two scores. The first score is the **image feature compatibility score**, measuring how well the 2D location of a facade plane in the image agrees with image features:

$$S_r(\mathbf{n_{si}}, \mathbf{x_{si}}|I) = \frac{1}{N(\mathbf{Q})} \sum_{n \in \mathbf{Q}} (s_n^{(bldg)} + g_n(\mathbf{n_{si}}) + < \mathbf{n_{si}}, \mathbf{v_n} >), \tag{5.2}$$

where $\mathbf{Q}$ is the image region of the facade plane defined by its corners $\mathbf{x_{si}}$, and N is the number of pixels within $\mathbf{Q}$. We consider three sources of information. 1) $s_n^{(bldg)}$ is the semantic label score of pixel n belonging to "building"region. We use Stacked Hierarchical Labeling proposed by Munoz *et al.* [51] to perform semantic segmentation (please see Section 3.3). 2) $g_n(\mathbf{n_{si}})$ is the orientation label score of pixel n having an orientation label consistent with the orientation $\mathbf{n_{si}}$ of the plane. Orientation labels are obtained from the surface layout algorithm proposed by Hoiem *et al.* [33] (please see Section 3.4). The orientation labels of interest here include "planar left", "planar center", and "planar right". Plane orientation $\mathbf{n_{si}}$ is quantized into one of these labels before computing the score g_n using the orientation label distribution of pixel n produced by the surface layout algorithm. 3) $< \mathbf{n_{si}}, \mathbf{v_n} >$ computes the inner product between the plane orientation $\mathbf{n_{si}}$ and the orientation vector $\mathbf{v_n}$ at pixel n derived from vanishing lines using our line-sweeping algorithm described in Section 3.6. An example of the image features we use are shown in Figure 5.2 [2]. Note that the three terms in the right-hand side of Equation 5.2 is comparable because they all range from 0 (totally implausible) to 1 (totally plausible).

The intuition behind Equation 5.2 is that if an image region indeed belongs to a building facade, then it should 1) be supported by the semantic cue that it belongs to the "building"region, 2) be supported by the surface layout cue that its orientation agrees with the dominant surface layout label within it, and 3) be supported by the vanishing line cue that its orientation is consistent with the dominant horizontal vanishing line direction within it.

[2]Here we do not limit the line sweeping region to be contained within the "building"semantic region.

The second score is the **geometric compatibility score**. Although the facade detection algorithm to be described in the next section returns the corners of the facade plane in the image, yet the ground contact line is often occluded (*e.g.* by cars parked along the road). As a result, the bottom boundary of the facade plane in the image is usually invisible and is therefore flexible, as is shown in the right image of Figure 5.3. When we place the facade plane (with a fixed orientation $\mathbf{n_{si}}$ returned by the facade detection algorithm) at different depths, it would result in a series of ground contact lines sweeping the blue and green regions illustrated in the left image of Figure 5.3. Not all depths are equally plausible. For example, if the facade plane is placed very close to the camera, the ground contact line would be the blue line in the right image of Figure 5.3. Such a depth is geometrically implausible because there is ground region above the ground contact line. On the other hand, if the ground plane is placed very far away from the camera, the ground contact line would be the green line in the right image of Figure 5.3. Such a depth is also geometrically implausible because the resulting facade height would be too large. Based on these intuitions, we compute the geometric compatibility score of a facade plane with orientation $\mathbf{n_{si}}$ placed at a certain depth d_{si} by checking if the ground contact line of the facade plane in the image is both above the ground region and below the building region. In addition, it checks if the distance between the ground contact line and the ground horizon line in the image yields a reasonable building height in 3D space:

$$S_d(\mathbf{n_{si}}, d_{si}, \mathbf{x_{si}} | I, h_p, \mathbf{n_p}, f, H_f) = \\ \min\{1 - \frac{1}{N(\mathbf{A})} \sum_{n \in \mathbf{A}} s_n^{(gnd)}, 1 - \frac{1}{N(\mathbf{B})} \sum_{n \in \mathbf{B}} s_n^{(bldg)}, H_f(\frac{\|\mathbf{x_{st}} - \mathbf{x_{sb}}\|}{\|\mathbf{x_{sm}} - \mathbf{x_{sb}}\|} \cdot h_p))\}, \quad (5.3)$$

where $\mathbf{A}$ is the region between the ground contact line and the ground horizon as is illustrated by the green region in Figure 5.3. This regions is not supposed to contain ground pixels. $\mathbf{B}$ is the region between the ground contact line and the bottom of the image as is illustrated by the blue region in Figure 5.3. This region is not supposed to contain building pixels. h_p, $\mathbf{n_p}$, and f

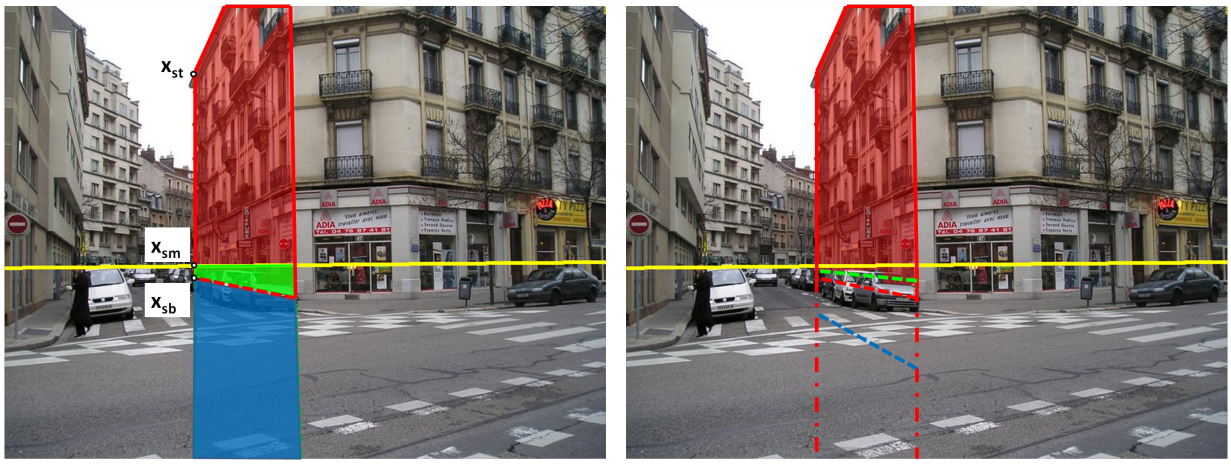

Figure 5.3: Illustration of the situation when computing the geometric compatibility score S_d. The yellow line is the ground horizon. In the left figure, the facade plane in the image is enclosed by the red lines; the visible part of the facade plane returned by our detection algorithm is indicated by the red shades, over which the image feature compatibility score S_r is computed. If there are ground pixels in the green region or building pixels in the blue region, S_d would be lower. $\mathbf{x_{st}}, \mathbf{x_{sb}}$ and $\mathbf{x_{sm}}$ are used to estimate the 3D height of the plane. The right figure shows the ground contact lines when the plane is placed at different depths. The blue ground contact line has a lower S_d score because part of the ground region is above it. The green ground contact line also has a lower S_d score, because the resulting facade height is not within a reasonable range.

are used to project the 3D ground contact line of the facade plane with orientation $\mathbf{n_{si}}$ at depth d_{si} to the 2D ground contact line in the image. $\mathbf{x_{si}}$ defines the top, left, and right boundaries of the facade plane in the image. As Figure 5.3 illustrates, $\mathbf{x_{st}}, \mathbf{x_{sb}}$ and $\mathbf{x_{sm}}$ are where the left boundary intersects with the top boundary, the (projected) ground contact line, and the ground horizon, respectively. $\|\mathbf{x_{st}} - \mathbf{x_{sb}}\|/\|\mathbf{x_{sm}} - \mathbf{x_{sb}}\| \cdot h_p$ is an estimate of the facade height in 3D. It should be within a reasonable range H_f of typical facade heights. Note that the three terms in the right-hand side of Equation 5.3 is comparable because they all range from 0 (totally implausible) to 1 (totally plausible).

After the two scores S_r and S_d are obtained, the unary potential ω is defined as

$$\omega(o_i, \mathbf{n_{si}}, d_{si}, \mathbf{x_{si}}|I, h_p, \mathbf{n_p}, f, H_f) =$$

$$\begin{cases} S_r(\mathbf{n_{si}}, \mathbf{x_{si}}|I) \cdot S_d(\mathbf{n_{si}}, d_{si}, \mathbf{x_{si}}|I, h_p, \mathbf{n_p}, f, H_f) - 0.25 & \text{if } o_i = 1 \\ 0 & \text{if } o_i = 0 \end{cases} \tag{5.4}$$

Here, the product of S_r and S_d is subtracted by 0.25 because the neutral values of both S_r and S_d are 0.5. When $S_r \cdot S_d < 0.25$, the unary potential penalizes the objective function if $o_i = 1$, indicating the candidate plane is unlikely to be valid based on the cues from itself.

5.2.3 Mutual Compatibility

As building facades are structured, geometric constraints exist among facade planes. The first type of constraints we consider is **convex-corner constraint**: if two facade planes are adjacent in the image *and* their orientations could form a convex corner in 3D, then their depths should be such that they connect in 3D along the convex fold [25], as is illustrated in Figure 5.4. As a result, all the facade planes connected by convex corners (such as the red, magenta, and cyan planes in Figure 5.1) could only move in space as a whole. To enforce such a constraint, the pairwise potential φ_v is defined as

$$\varphi_v(o_i, o_j, d_{si}, d_{sj}, \mathbf{n_{si}}, \mathbf{n_{sj}}, \mathbf{x_{si}}, \mathbf{x_{sj}}|h_p, \mathbf{n_p}, f) =$$

$$\begin{cases} -\infty & \text{if } o_i \neq 0 \text{ and } o_j \neq 0 \text{ and } \mathbf{x_{si}}|\mathbf{x_{sj}} \text{ and } \mathbf{n_{si}} \vee \mathbf{n_{sj}} \text{ and } d_{si} \veebar d_{sj} \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

where $\mathbf{x_{si}}|\mathbf{x_{sj}}$ means facade planes i and j are adjacent in the image, $\mathbf{n_{si}} \vee \mathbf{n_{sj}}$ means the orientations of facade planes i and j form a convex corner, and $d_{si} \veebar d_{sj}$ denotes the situation in

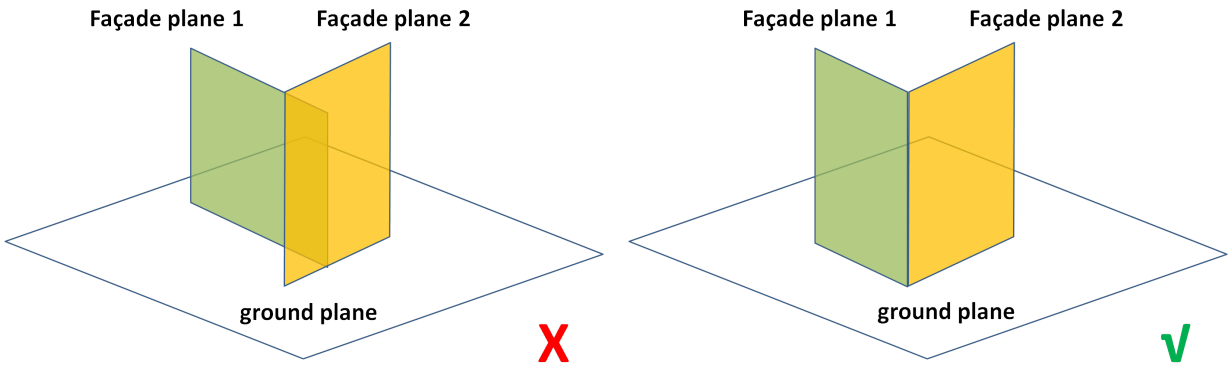

Figure 5.4: Facade planes that are adjacent in the image and form a convex corner in 3D must connect with each other along the convex fold.

which the depths of the two planes are such that they fail to connect along the convex fold in 3D (*i.e.*, their 3D ground contact lines do not meet and connect with each other). Negative infinity penalty is applied if this situation happens. In other words, the convex-corner constraint is a hard constraint, so that physical plausibility is always maintained.

The second type of constraints is **occlusion constraint**. If two facade planes are adjacent in the image *and* their orientations form a concave corner, we check if one of them is occluded by the other. Figure 5.5 illustrates three cases when two facade planes form a concave corner (other cases are symmetrical to one of these three cases). In the left case, it can be immediately determined that the blue plane is occluded by the red one. In the center case, we cannot say for sure which plane is occluded. In the right case where the blue plane does not extend beyond the red one, we check the vanishing lines and orientation map within a small region right next to the blue plane. If the region has the same orientation as the blue plane, then it becomes the left case; otherwise it becomes the center case. Although we cannot unambiguously determine the occlusion ordering for the center case, we could still reasonably assume that the plane facing more towards the camera is usually occluded, such as the red and green facade planes in Figure 5.1. While such an assumption could be violated, it holds true for most of the outdoor urban scenarios we have encountered. Also, as we will see in the equation below, the occlusion constraint is a soft constraint, meaning it could be overridden by other stronger evidence. Without loss of

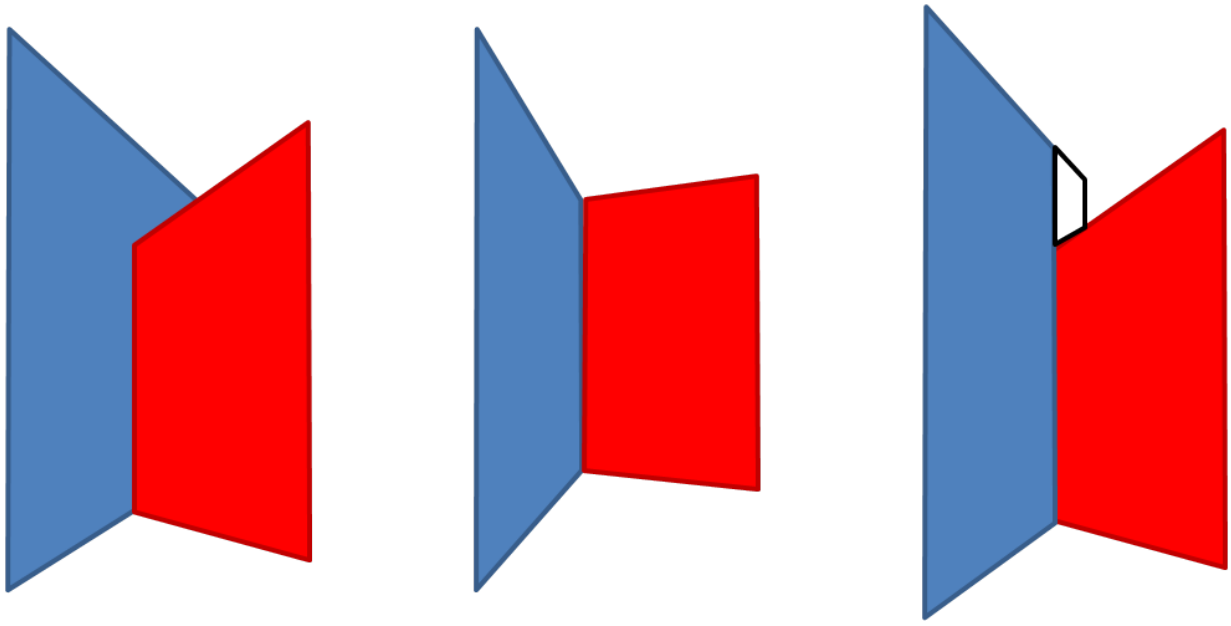

Figure 5.5: Three cases for determining the occlusion ordering between two facade planes that are adjacent in the image and form a concave corner in 3D. Please see text for details.

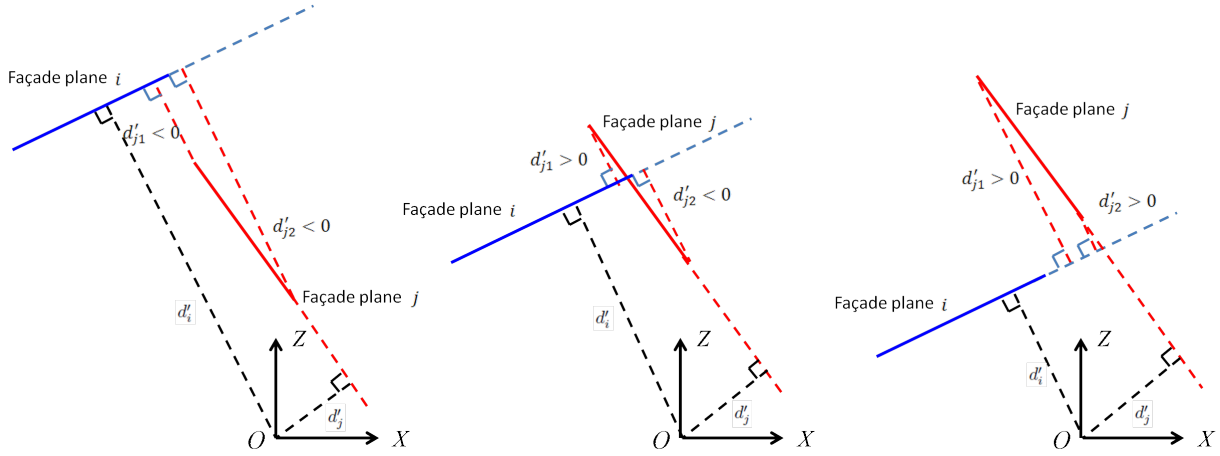

Figure 5.6: Given the occlusion ordering between two facade planes, this figure illustrates three possible cases when computing the occlusion constraint. The solid red and blue lines are the ground contact lines of the two facade planes in the ground coordinate system. d'_{j1} and d'_{j2} are signed distances from the two endpoints of the red line to the blue line. d'_i and d'_j are the distances from the ground origin to the ground contact lines of facade planes i and j, respectively. They can be easily obtained from d_{si} and d_{sj}.

generality, suppose facade plane i is occluded by facade plane j. Then the pairwise potential φ_o related to occlusion ordering is defined as

$$\varphi_o(o_i, o_j, d_{si}, d_{sj}, \mathbf{n_{si}}, \mathbf{n_{sj}}, \mathbf{x_{si}}, \mathbf{x_{sj}} | h_p, \mathbf{n_p}, f) = \begin{cases} -\max\{0, d'_{j1}, d'_{j2}\}^2/(2\sigma_1^2) & \text{if } o_i \neq 0 \text{ and } o_j \neq 0 \text{ and } \mathbf{x_{si}} | \mathbf{x_{sj}} \text{ and } \mathbf{n_{si}} \wedge \mathbf{n_{sj}} \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

Here, $\mathbf{n_{si}} \wedge \mathbf{n_{sj}}$ means the orientations of facade planes i and j form a concave corner. d'_{j1} and d'_{j2} are defined in Figure 5.6, where the solid blue and red lines are the ground contact lines of facade planes i and j, respectively, in the ground coordinate system. According to equation 5.6, the pairwise potential φ_o is 0 (*i.e.*, no penalty) when facade plane i is totally behind facade plane j; otherwise, a soft penalty is applied depending on the degree of violation. The soft constraint here serves to handle noise in the estimation of occlusion ordering.

The third type of constraints is **alignment constraint**: facade planes are encouraged to be aligned by having the same distance from the origin if they meet the following two criteria: 1) they have the same orientation, and (2) neither of them is occluding the other through facade planes connected to them by convex corners. Please see Figure 5.7 for illustration. Such a constraint is based on the observation that street-facing facades of different buildings are usually aligned. If facade planes i and j satisfy both the same-orientation and non-occlusion criteria, the pairwise potential φ_a encoding the alignment constraint between them is defined as

$$\varphi_a(o_i, o_j, d_{si}, d_{sj}, \mathbf{n_{si}}, \mathbf{n_{sj}}, \mathbf{x_{si}}, \mathbf{x_{sj}} | h_p, \mathbf{n_p}, f) = \begin{cases} -(d'_i - d'_j)^2/(2\sigma_2^2) & \text{if } o_i \neq 0 \text{ and } o_j \neq 0 \text{ and } \mathbf{n_{si}} = \mathbf{n_{sj}} \text{ and } Q_s(i,j) = 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

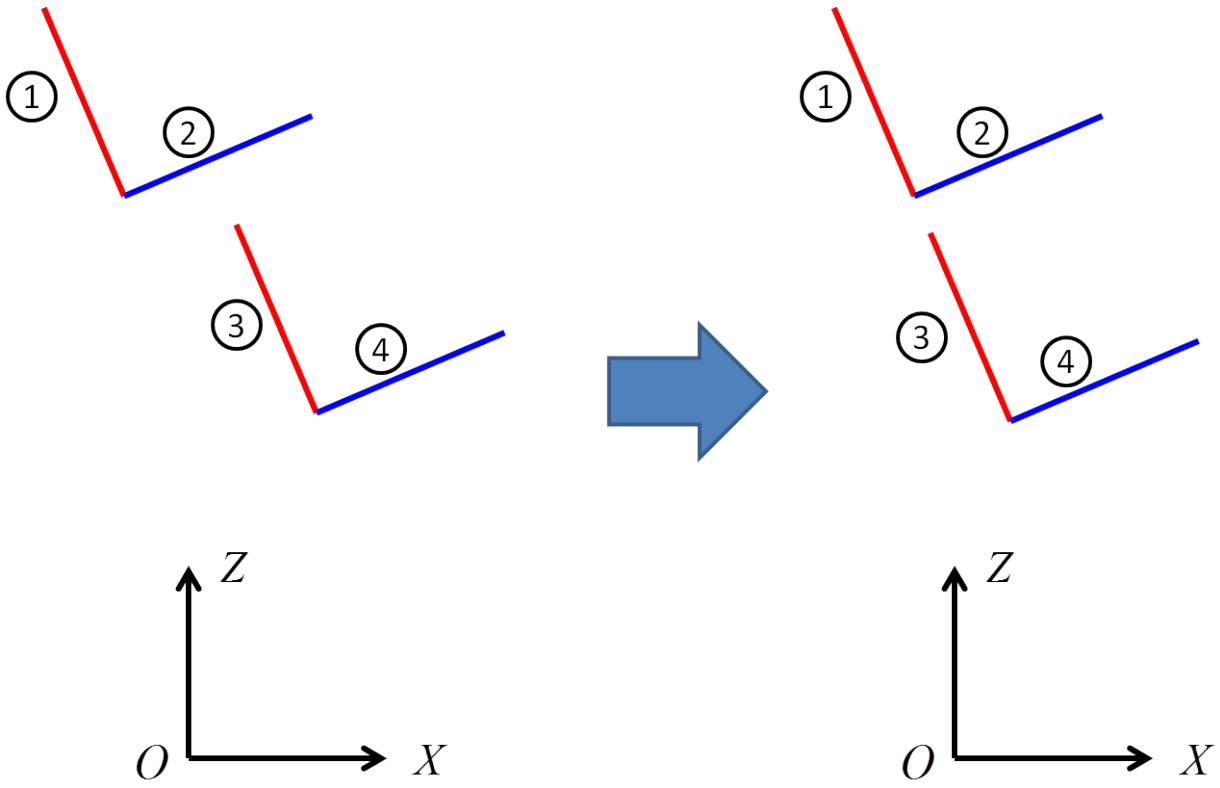

Figure 5.7: Illustration of the alignment constraint. Facade planes that satisfy both the same-orientation and non-occlusion criteria are encouraged to be aligned. Therefore, the configuration on the right is preferred over the one on the left. Although facade planes 2 and 4 have the same orientation, they do not meet the non-occlusion criterion. This is because facade plane 3 explicitly occludes facade plane 2, while facade plane 4 is connected to facade plane 3 through a convex corner. As a result, facade plane 4 indirectly occludes facade plane 2. The transitivity via convex corners does not apply to the occluded group. For example, although facade plane 3 occludes facade plane 2 and facade plane 2 is connected to facade plane 1 through a convex corner, facade plane 3 is not considered to be occluding facade plane 1.

where $Q_s(i,j) = 0$ means the two facade planes do not occlude each other through other planes connected to them via convex corners. d'_i and d'_j are the distances from the ground origin to the ground contact lines of facade planes i and j, respectively. The alignment constraint is also a soft constraint since we are just encouraging eligible planes to be aligned; other strong evidence could override such an encouragement.

The benefit of incorporating geometric constraints among facade planes is evident in Figure 5.1. After inter-planar geometric constraints are imposed, planes that form convex corners are correctly connected together, and planes in the background are correctly pushed backward.

5.3 Implementation Details

Optimizing the objective function in Equation 5.1 is challenging, because it is generally non-convex and involves binary, discrete and continuous variables, resulting in a huge search space. In addition, how do we acquire the candidate facade planes in the first place? In this section, we describe a quadrilateral-based sampling algorithm that detects a set of candidate facade planes, each of which comes with a 3D orientation $\mathbf{n_{si}}$ and corner coordinates $\mathbf{x_{si}}$. Using the output of the quadrilateral-based sampling algorithm, we are able to optimize Equation 5.1 in a tractable way. The detailed procedures of the preprocessing stage, the quadrilateral-based sampling algorithm, and the construction/inference of the CRF model are shown in Figure 5.8.

5.3.1 Quadrilateral-based Sampling Algorithm

We detect candidate facade planes via a sequential sampling of "quadrilaterals". A sample of a quadrilateral is formed by a random pair of vertical vanishing lines and a random pair of horizontal vanishing lines in the image (please see Figure 5.9 for examples). The 3D orientation of a quad (short for quadrilateral) is equal to the cross-product of the vertical and horizontal vanishing directions. A quad itself could be a facade plane, or multiple adjacent quads with the

1. Preprocessing
   1.1. Estimate vanishing points, focal length, and ground horizon (see Section 3.5)
   1.2. Assuming the ground plane distance $h_p = 1.6$m, establish the ground coordinate system using Equs. (3.1)-(3.3) (see Section 3.1)
   1.3. Compute soft semantic and surface layout labels (see Sections 3.3 and 3.4)
   1.4. Compute vertical orientation map using the line-sweeping algorithm (see Section 3.6)
2. Detect candidate façade planes using quadrilateral-based sampling algorithm
   2.1. Sample a large number of quadrilaterals in the image using a random pair of vertical vanishing lines and a random pair of horizontal vanishing lines as four sides
   For each quad (short for quadrilateral)
      2.1.1. Compute its 3D orientation as the cross-product of its vertical and horizontal vanishing directions
      2.1.2. Evaluate the image feature compatibility score S_r using Equ. (5.2)
   2.2. Select the top *k* quads based on their S_r scores
   For each top quad
      2.2.1. Expand the quad by annexing neighboring quadrilateral regions having high S_r scores (computed with the 3D orientation of the quad)
      2.2.2. Identify any existing adjacent quad that has the same orientation or forms a convex corner with this quad. They form a rigid group that moves together
      2.2.3. Place the group at a large set of quantized hypothetical depths in 3D
      For each depth
         2.2.3.1. Project the ground contact lines of the group back to the image
         2.2.3.2. Evaluate the geometric compatibility score S_d using Equ. (5.3). Note that the S_d of a group of planes is the minimum S_d of individual planes in the group
      2.2.4. Compute the hybrid score $S_r \cdot \hat{S}_d$ for this quad, where $\hat{S}_d$ is the maximum S_d score over all depths
   2.3. Select the quad with the highest hybrid score and add it to the existing set of quads
   2.4. Repeat 2.1 - 2.3 until the hybrid score of the best quad is below 0.5
   2.5. Merge adjacent quads with the same orientation into a single distinctive candidate façade plane
3. Infer validity and optimal depth of candidate façade planes
   3.1. Identify pairwise constraints among candidate planes according to Section 5.2.3
   3.2. Construct a CRF where each candidate plane is a node, and each pair of mutually related planes forms an edge
   3.3. For each candidate plane, compute its unary potentials over validity indicator o_i and depth d_{si} by evaluating Equ. (5.4). The S_r and S_d of the plane is the average score of all the quads belonging to the plane
   3.4. For each pair of mutually related planes, compute their pairwise potentials over validity indicators o_i, o_j and depths d_{si}, d_{sj} by evaluating one of the Equs. (5.5)-(5.7)
   3.5. Perform MAP inference over the CRF using max-product belief propagation

Figure 5.8: Procedure of our 3D facade detection and inference algorithm that implements the model described in Section 5.2.1.

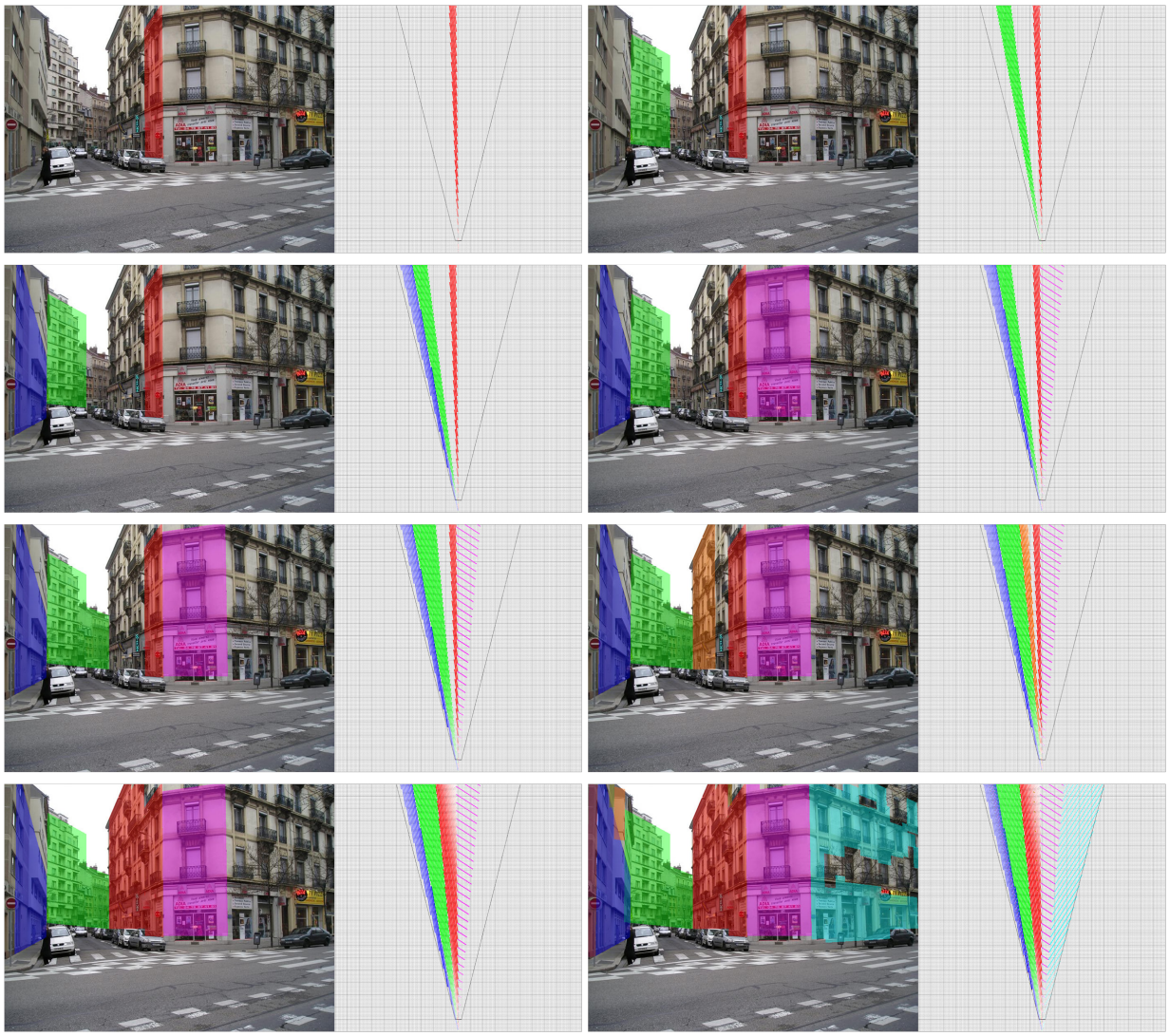

Figure 5.9: When detecting candidate facade planes, our algorithm selects and adds the best quad into the candidate pool in each iteration. Row by row, this figure shows the first seven quads selected by our algorithm, as well as all the selected quads when the search process is completed. Quads belonging to the same candidate distinctive facade plane share the same color. The plot right next to each image shows the unnormalized distribution of geometric compatibility score over a set of hypothetical depths for each candidate facade plane (please see Procedure 2.2.3). Colored lines are the ground contact lines of corresponding planes at different depths plotted in the ground coordinate system. The spacing of the coarser grid is 10m. We can see that as new quads (*e.g.* the seventh quad) are added and connected to existing quads through convex corners or mergers, the depth distributions of existing candidate facade planes might change. Contextual information is used even during the candidate search stage before constructing the CRF.

same 3D orientation could merge and form a larger facade plane. When quads are selected and added to the image one by one, such mergers often occur. For example, in the left image of the third row of Figure 5.9, the newly added quad is merged into the existing green quad to form a single plane. Also, in the left image of the fourth row, the newly added quad joins together the originally separate red and orange quads in the right image of the third row, and the three quads form a single plane.

In the quadrilateral-based sampling algorithm, we first sample a large number of quads and evaluate each sample using the image feature compatibility score S_r. The top k quads, which come with known orientation $\mathbf{n_{si}}$ and spatial extent $\mathbf{x_{si}}$, are selected and further evaluated by computing the optimal geometric compatibility score over a set of quantized hypothetical depth d_{si}. Note that the optimal geometric compatibility score is evaluated not just based on the current quad under evaluation, but a set of quads connected to the current quad via same-orientation or convex-corner relationships. For example, when evaluating the newly added red quad in the left image of the last row in Figure 5.9, all the quads within the red and magenta regions are involved. In this way, the convex-corner constraint is also utilized in the candidate plane detection process.

5.3.2 Tractability

In our approach, the key to make the inference tractable is to decouple $\mathbf{n_s}, \mathbf{x_s}$ from $\mathbf{o}, \mathbf{d_s}$ in Equation 5.1. When the quadrilateral-based sampling process is completed (*i.e.*, Procedure 2.5 is completed), we already have a set of candidate facade planes with known orientation $\mathbf{n_{si}}$ and spatial extent $\mathbf{x_{si}}$. The only variables that remain to be inferred are the validity indicator o_i and depth d_{si}. The objective function in Equation 5.1 is therefore reduced to

$$V(\mathbf{o}, \mathbf{d_s}) = \sum_{i \in \mathbf{P}} \omega(o_i, d_{si}) + \sum_{(i,j) \in \mathbf{P_v}} \varphi_v(o_i, o_j, d_{si}, d_{sj}) + \sum_{(i,j) \in \mathbf{P_o}} \varphi_o(o_i, o_j, d_{si}, d_{sj}) + \sum_{(i,j) \in \mathbf{P_a}} \varphi_a(o_i, o_j, d_{si}, d_{sj}), \quad (5.8)$$

where we have omitted the conditioning variables for the sake of clarity. Suppose we have quantized depth d_{si} into M bins, then the number of possible states over o_i and d_{si} is only $M+1$. This is significantly more tractable than the original optimization problem. We construct a CRF to encode all the inter-planar mutual constraints, and perform max-product belief propagation [30] to obtain the validity of each candidate facade plane and the optimal depth of each *valid* plane, as well as a MAP distribution over its possible depths.

5.3.3 Avoiding Local Extrema

Since we use a greedy approach when detecting candidate facade planes, it is important to reduce the risk of being trapped in local extrema. To achieve this purpose, we sample a large number of quads, and those quads go through two rounds of evaluation. The first round selects the top k quads with the highest S_r score. The second round further considers how well a quad (which has a fixed orientation and spatial extent) can be placed at different depths in 3D space, requiring the evaluation of the S_d score. Moreover, existing quads serve as a geometric context when evaluating a new quad in the second round. We found that when the search width k is greater than or equal to 4, performing multiple starts does not bring additional improvement over a single start.

5.4 Experiments

We evaluate the performance of our facade reasoning algorithm on three challenging datasets. All the parameters of our algorithm are fixed throughout our experiments, where the camera height h_p is assumed to be 1.6m, $\sigma_1 = 0.5$ in Equation 5.6, $\sigma_2 = 50$ in Equation 5.7, the number of random samples in Procedure 2.1 is 2000, the number k of top quads in Procedure 2.2 is set as 4, and the reasonable range of building height H_f is assumed to be between 5m and 30m, with a standard deviation of 1m on the lower end, and 10m on the higher end. On average, it takes about 20 minutes to process a 800-by-600 image with Matlab code.

5.4.1 3D Facade Layout

The key strength of our approach is that it is able to produce a richer interpretation of a facade scene – a 3D layout of facade planes with continuous orientations. Note that Gupta's block-based approach [25] also generates a 3D layout of building facades, yet their approach is mainly qualitative while ours is quantitative and probabilistic in nature. As a result, our approach is able to yield a quantitative top-down view of the 3D facade layout given a single image.

We apply our facade reasoning algorithm on the LabelMe dataset [34]. As the dataset does not provide the ground truth for a plane-wise decomposition of building facades or their 3D layout, we evaluate our approach qualitatively. Typical examples of both success and failure cases are shown in Figures 5.10 through 5.13. Here, in the first column, the image regions detected by our method as building facades are covered in color-shaded tiles (*i.e.* quads). Quads belonging to the same distinctive plane share the same color. The second column plots the top-down view of the unnormalized depth distribution of each candidate distinctive facade plane *before* considering inter-planar interactions. Such a distribution is part of the unary potentials. The third column plots the top-down view of the best locations of candidate distinctive facade planes given their (unary) depth distribution. The fourth column plots the top-down view of the final optimal locations of all valid distinctive facade planes *after* considering inter-planar

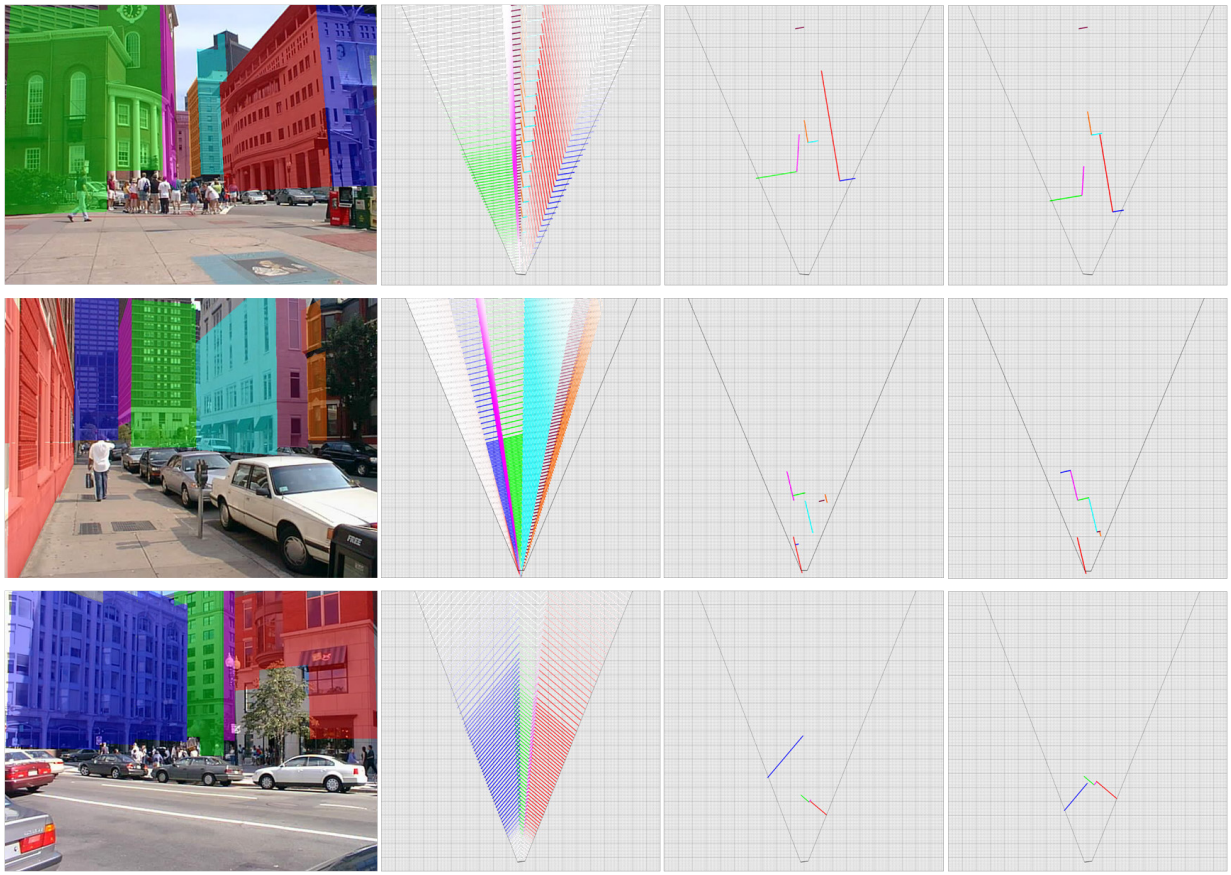

Figure 5.10: 3D facade layout estimation by our method on images from the LabelMe dataset [34]. Left to right: 1) Original image overlaid with color shades representing quads from distinctive facade planes. Quads from the same distinctive facade plane share the same color. 2) Depth distribution of candidate distinctive facade planes before running the CRF inference. 3) Best locations of candidate distinctive facade planes before running the CRF inference. 4) Optimal locations of valid distinctive facade planes after running the CRF inference. The viewing boundary is marked with black lines, and the coarser grid spacing is 10m.

interactions. As we can see, imposing inter-planar geometric constraints significantly regulates the depths of distinctive facade planes and results in a meaningful interpretation. In all top-down views, the coarser grid spacing is 10m.

In the LabelMe dataset [34], many complex facade structures can be found – many building facades are occluded by other objects or mutually occluded, and in some cases they are not Manhattan (*e.g.* row 1 in Figure 5.10 and row 1 in Figure 5.12). As our method makes no assumption on those conditions, they do not pose a problem. We also do not assume all building

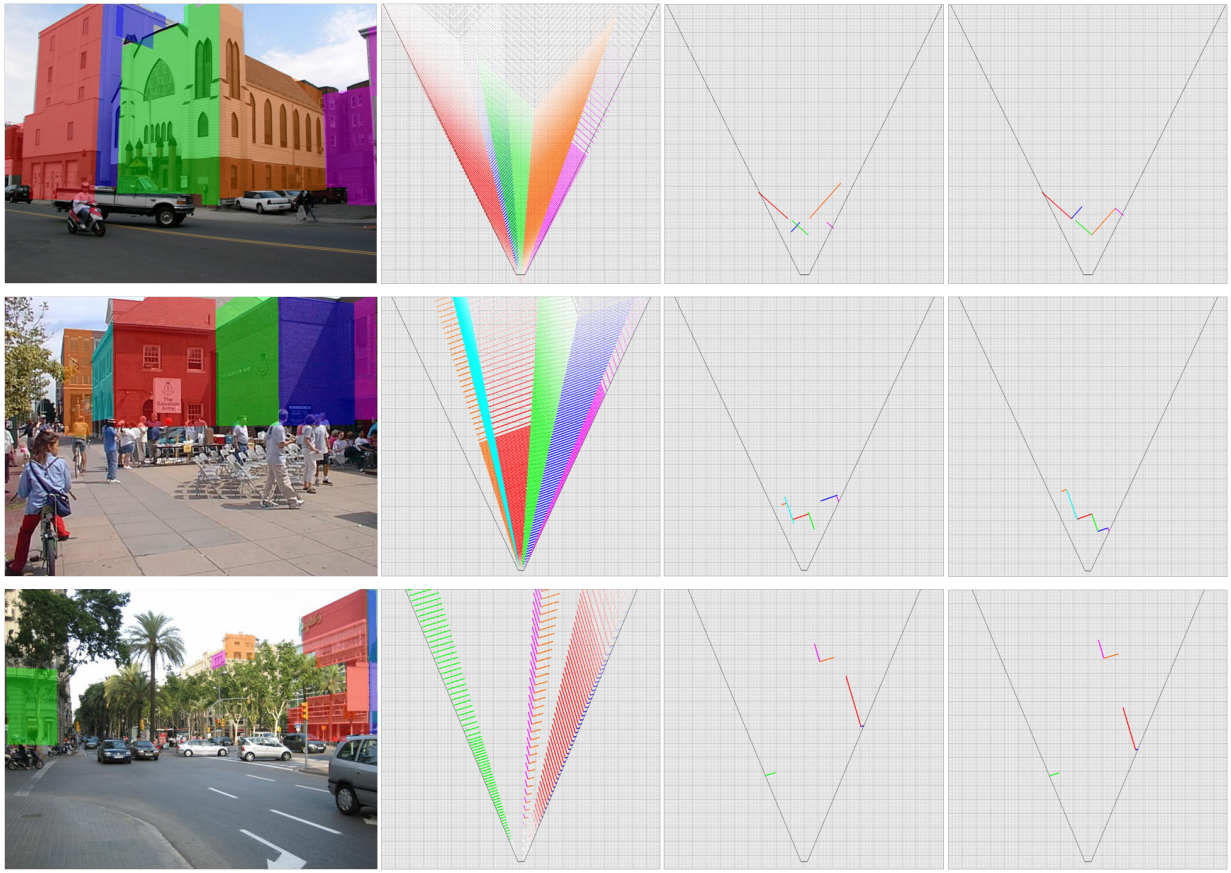

Figure 5.11: More examples of 3D facade layout estimation by our method on images from the LabelMe dataset [34]. Left to right: 1) Original image overlaid with color shades representing quads from distinctive facade planes. Quads from the same distinctive facade plane share the same color. 2) Depth distribution of candidate distinctive facade planes before running the CRF inference. 3) Best locations of candidate distinctive facade planes before running the CRF inference. 4) Optimal locations of valid distinctive facade planes after running the CRF inference. The viewing boundary is marked with black lines, and the coarser grid spacing is 10m.

Figure 5.12: Small mistakes made by our method during 3D facade layout estimation on the LabelMe dataset [34]. Left to right: 1) Original image overlaid with color shades representing quads from distinctive facade planes. Quads from the same distinctive facade plane share the same color. 2) Depth distribution of candidate distinctive facade planes before running the CRF inference. 3) Best locations of candidate distinctive facade planes before running the CRF inference. 4) Optimal locations of valid distinctive facade planes after running the CRF inference. The viewing boundary is marked with black lines, and the coarser grid spacing is 10m.

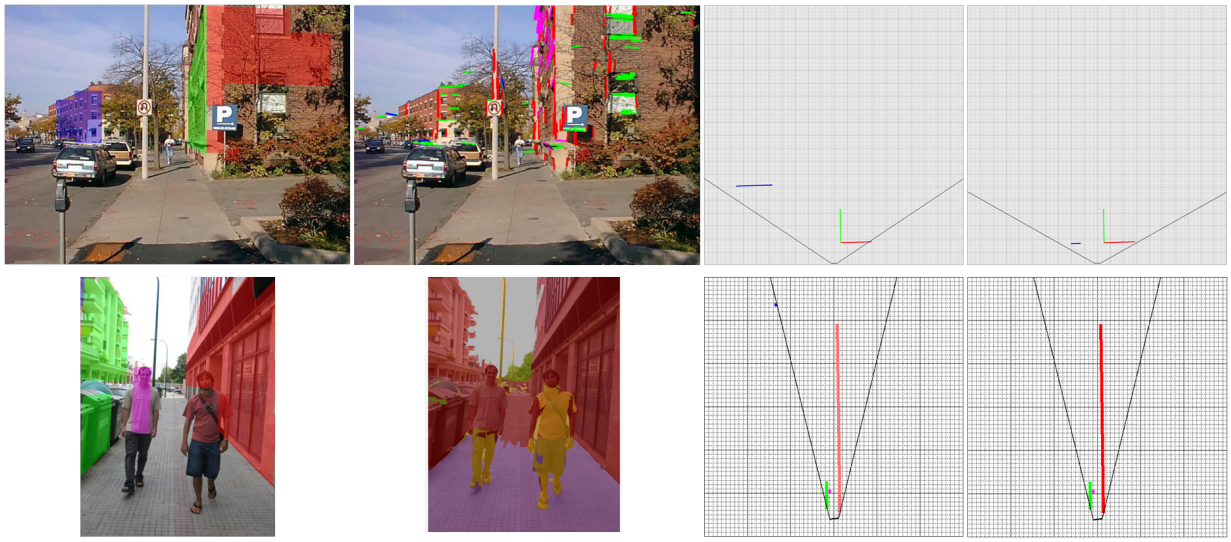

Figure 5.13: Major failure cases by our method during 3D facade layout estimation on the LabelMe dataset [34]. Left to right: 1) Original image overlaid with color shades representing quads from distinctive facade planes. Quads from the same distinctive facade plane share the same color. 2) Row 1: detected vertical and horizontal vanishing lines; Row 2: semantic segmentation by the SHL model. 3) Best locations of candidate distinctive facade planes before running the CRF inference. 4) Optimal locations of valid distinctive facade planes after running the CRF inference. The viewing boundary is marked with black lines, and the coarser grid spacing is 10m.

facades are inter-connected as [54] does. Therefore, many street scenes where adjacent buildings are separated by streets can be modeled by our approach (*e.g.* rows 1 and 3 in Figure 5.10, row 1 in Figure 5.11, row 3 in Figure 5.12).

In row 3 of Figure 5.11, we could also see that distinct facade planes having the same orientation (the red and magenta planes) are aligned after CRF inference. However, when other stronger evidence is present as is the case in row 2 of Figure 5.11, they are not blindly aligned.

Figure 5.12 includes examples in which our method makes not-so-annoying mistakes. In row 1, the optimal depth of the farther building is too small; in row 2, a distinctive facade plane is merged with its adjacent facade plane; in row 3, several minor "noise"facade planes caused by extruding balconies are present.

Figure 5.13 shows examples when major failures occur. In the first row, the farther left-facing facade is not detected due to lack of vanishing lines. In the second row, severe mistakes in semantic segmentation result in false facade planes being detected. These failures can be mitigated when vanishing line detection and semantic segmentation are improved.

5.4.2 Surface Layout Estimation

To our knowledge, our algorithm is among the first to be able to generate a 3D layout of building facades consisting of mutually-constrained planes with continuous orientations. While our algorithm allows for a greater flexibility and produces a richer interpretation than existing methods, does it negatively affect existing quantitative measures in the literature?

To answer this question, we first evaluate our algorithm in estimating surface layout on the Geometric Context dataset [33]. Among the 250 test images, 55 of them contain building facades. Therefore, we perform evaluation only on those 55 images. As the dataset does not provide ground truth for semantic segmentation, we train the Stacked Hierarchical Labeling model [51] on the Stanford Background Dataset [23] and use the resulting model to compute soft semantic labels. We also use the publicly available pre-trained Geometric Context classifier [32] to obtain

	Hoiem	Gupta	Ours	Ours w/o CRF	Orien. Map
Surface Layout Accuracy	72.87%	73.59%	74.82%	73.89%	71.20%

Figure 5.14: Comparison of surface layout accuracy. The methods from left to right are the algorithm from Hoiem *et al.* [35], Gupta *et al.* [25], our algorithm, our algorithm without CRF inference, and the line-sweeping algorithm mentioned in Section 3.6.

the soft geometric context labels.

The dataset comes with the ground truth of seven surface layout labels ("support", "sky", "planar left", "planar center", "planar right", "non-planar porous", "non-planar solid"). We compare our method with the block-based approach proposed by Gupta *et al.* [25] and the segment-based approach proposed by Hoiem *et al.* [35] on surface layout accuracy. To generate surface layout labels from the output of our algorithm, we quantize our continuous 3D orientation of facade planes into soft labels of "planar left", "planar center", or "planar right", and fuse them with the labels returned by the original Geometric Context classifier.

The comparison results are listed in Figure 5.14. We can see that our method achieves better accuracy than the state-of-the-art block-based and segment-based approaches. We also observe that if we remove the inter-planar geometric constraints from our algorithm, the accuracy drops as is shown in the 'Ours w/o CRF' column. Accuracy from the raw orientation map generated by the line-sweeping algorithm (Section 3.6) is the lowest, as it is highly susceptible to noise in vanishing lines.

We also show qualitative comparisons in Figure 5.15. The ground-truth surface layout is shown in column 1. We can see that in the first row, the segment-based approach (column 2) decomposes the facade into many unstructured orientation segments, while the block-based approach (column 3) represents the entire facade as a block. Our plane-based approach (column 4) approximates the true facade using a set of mutually-constrained planes and therefore generates a much better approximation. [3] A similar case can be observed in row 5. In rows 2-4,

[3]The ground-truth annotation is not totally correct. The red "planar center"regions should actually be magenta "planar right"ones.

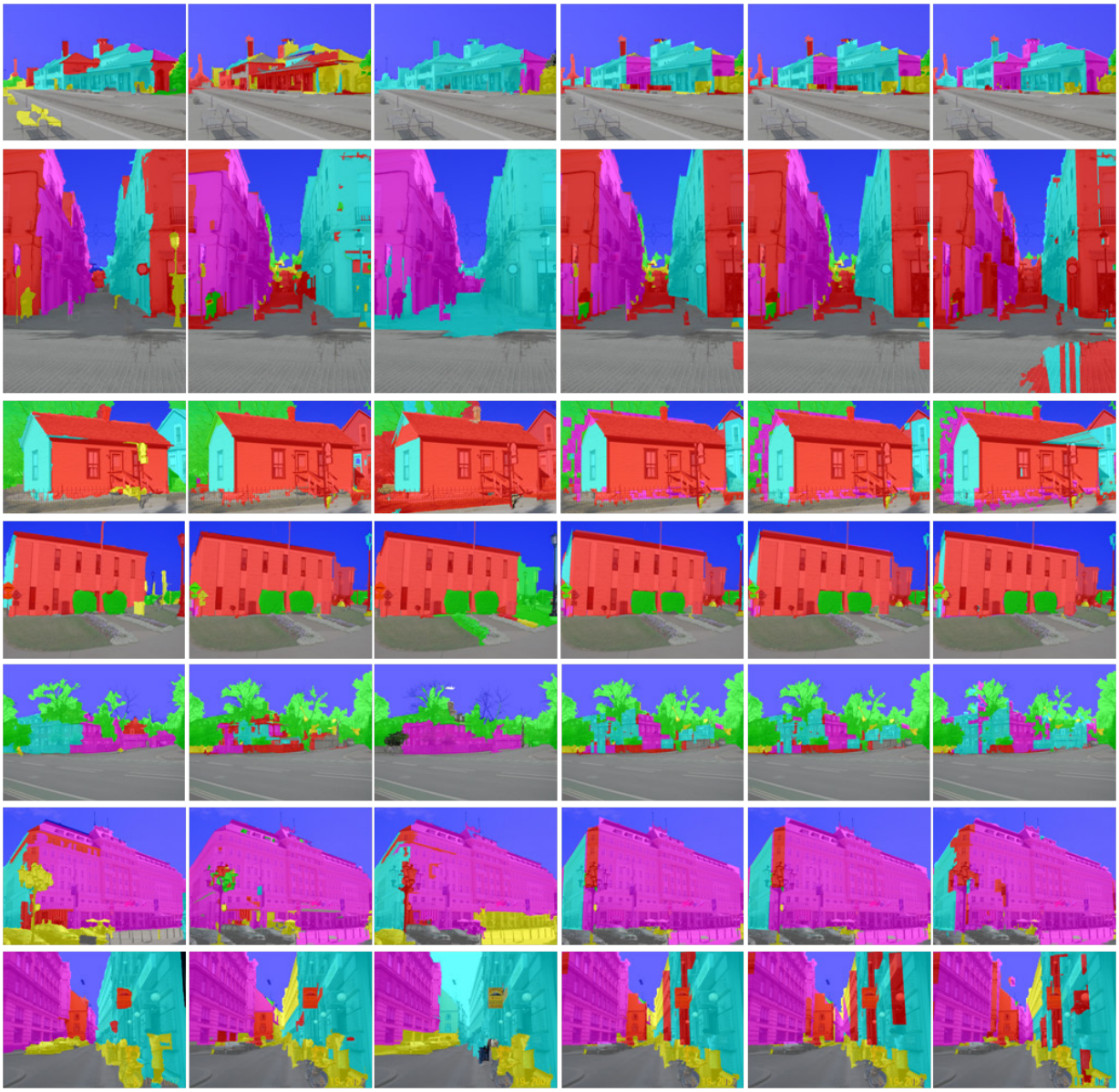

Figure 5.15: Qualitative comparisons of surface layout estimation. From left to right: Ground truth; Hoiem *et al.* [35]; Gupta *et al.* [25]; Ours; Ours w/o CRF; Raw orientation map (Section 3.6). Surface layout color code: Magenta – planar right; Cyan – planar left; red – planar center; green – non-planar porous; yellow – non-planar solid; blue – sky; grey – support.

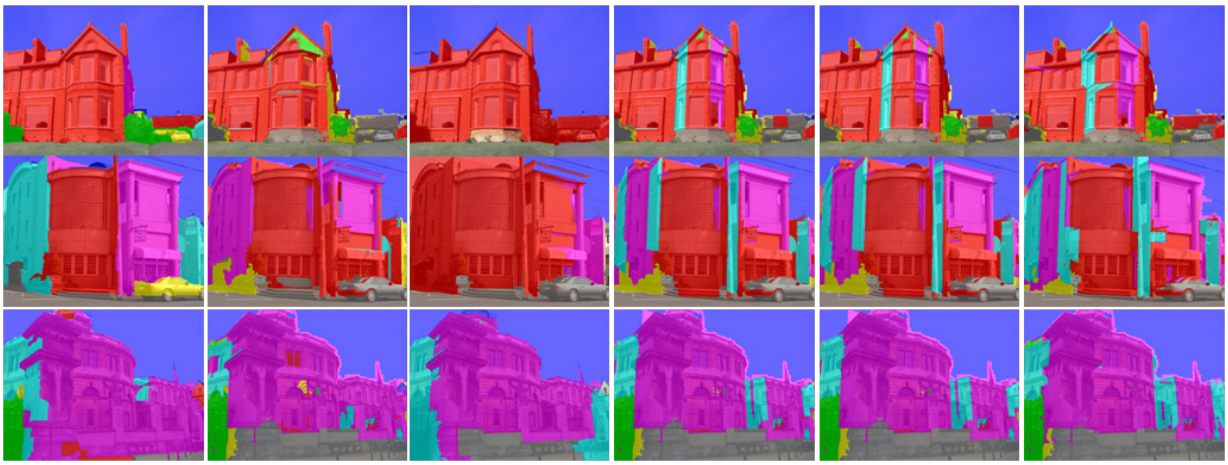

Figure 5.16: Failure cases of surface layout estimation. From left to right: Ground truth; Hoiem *et al.* [35]; Gupta *et al.* [25]; Ours; Ours w/o CRF; Raw orientation map (Section 3.6). Surface layout color code: Magenta – planar right; Cyan – planar left; red – planar center; green – non-planar porous; yellow – non-planar solid; blue – sky; grey – support.

our approach identifies distinctive facades missed out by the other approaches. In row 6 where a non-Manhattan facade exists, our approach identifies it unambiguously.

While inter-planar geometric constraints usually regularize plane depths (as we have seen in Figures 5.10 through 5.12), it could also improve surface layout estimation by removing invalid planes. In an example shown in row 7 of Figure 5.15, an invalid plane labeled with "planar center"(red) in column 5 was removed after the CRF inference, and the true orientation ("planar left", cyan) of that region is revealed in column 4.

Column 6 of Figure 5.15 shows that the surface layout estimation directly obtained from orientation maps is usually quite noisy. Our plane detection process helps inhibit such noise and results in a better result in Column 5, and inter-planar interactions further lead to an even better result in Column 4.

Several of our major failure cases are shown in Figure 5.16. The case in the first row fails because our approach discovers small non-Manhattan facade structures not included in the ground truth. It also failed to identify the garage in the distance. In the second row, our method again detects a fine structure not included in the ground truth, while mis-labeling a left-facing region due

to the deceiving vanishing lines from the rooftop. In the third row, our method also mis-labeled the left-facing part of a foreground structure due to strong noise in vanishing lines.

To get a better insight into the comparison between our approach and the state-of-the-art algorithms, we compute, for each image, the accuracy gain of our approach over the segment-based approach [35] and the block-based approach [25], respectively, and plot the histogram of accuracy gain over the test images.

The histograms for comparisons with the two approaches are shown in Figures 5.17 and 5.18, respectively. In each figure, typical examples are shown for positive, neutral, and negative accuracy gains, respectively. In both figures, we can see that our algorithm achieves good surface classification accuracy where vanishing lines are correctly detected and grouped. However, in image regions where missing or distracting vanishing lines dominate, the accuracy of our algorithm deteriorates, because vanishing lines play an important role in proposing and evaluating quadrilateral samples. A poor semantic segmentation result would also significantly undermine the surface classification accuracy of our algorithm, as is shown in the leftmost example in Figure 5.18. Please refer to the Appendix for a complete list of results on all test images.

Although our approach does not perform significantly better than the state-of-the-art algorithms in terms of surface layout classification, our approach is able to generate a quantitative, continuous 3D layout of the facade scene, which is beyond the capability of competing methods. Several examples are shown in Figure 5.19.

5.4.3 Depth Map Estimation

To see how our method performs in recovering absolute depth values, we also perform evaluation on the Make3D dataset of Saxena *et al.* [48, 60] which provides ground truth in pixel-wise depth value. As the dataset also provides ground-truth semantic segmentation, we train the Stacked Hierarchical Labeling model [51] on the 400 training images of the dataset. As for the Geometric Context classifier, we still use the publicly available pre-trained one [32]. Among the 134 test

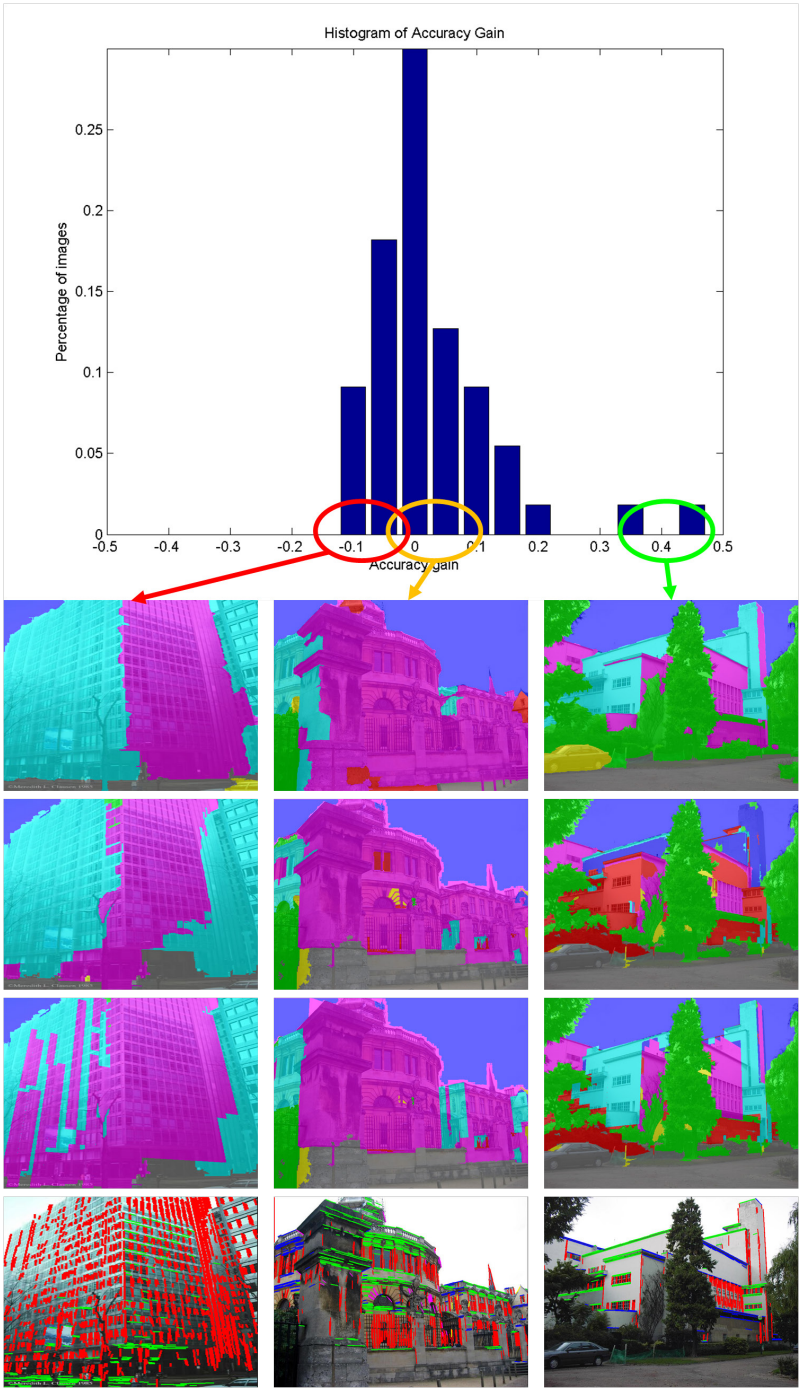

Figure 5.17: Histogram of accuracy gain of our algorithm over the segment-based approach in [35]. Typical examples corresponding to three different ranges of accuracy gain are also displayed, where the first row is the ground truth, the second row is the result of [35], the third row is our result, and the fourth row shows the vanishing lines. The color code is the same as in Figure 5.15.

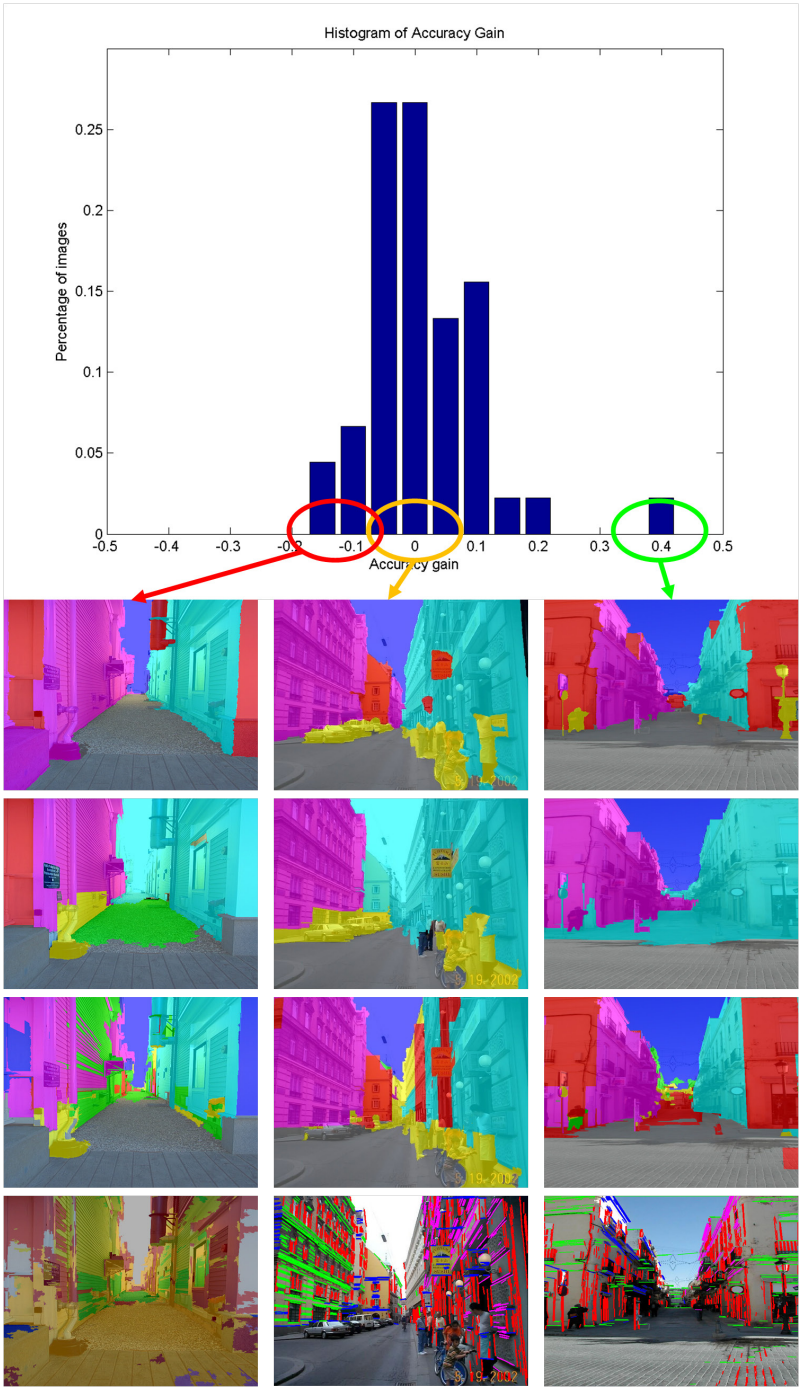

Figure 5.18: Histogram of accuracy gain of our algorithm over the block-based approach in [25]. Typical examples corresponding to three different ranges of accuracy gain are also displayed, where the first row is the ground truth, the second row is the result of [25], the third row is our result, and the fourth row shows the vanishing lines. Note that the leftmost image in the fourth row shows semantic segmentation, where the "building"region is indicated by the red shade. The color code of surface layout is the same as in Figure 5.15.

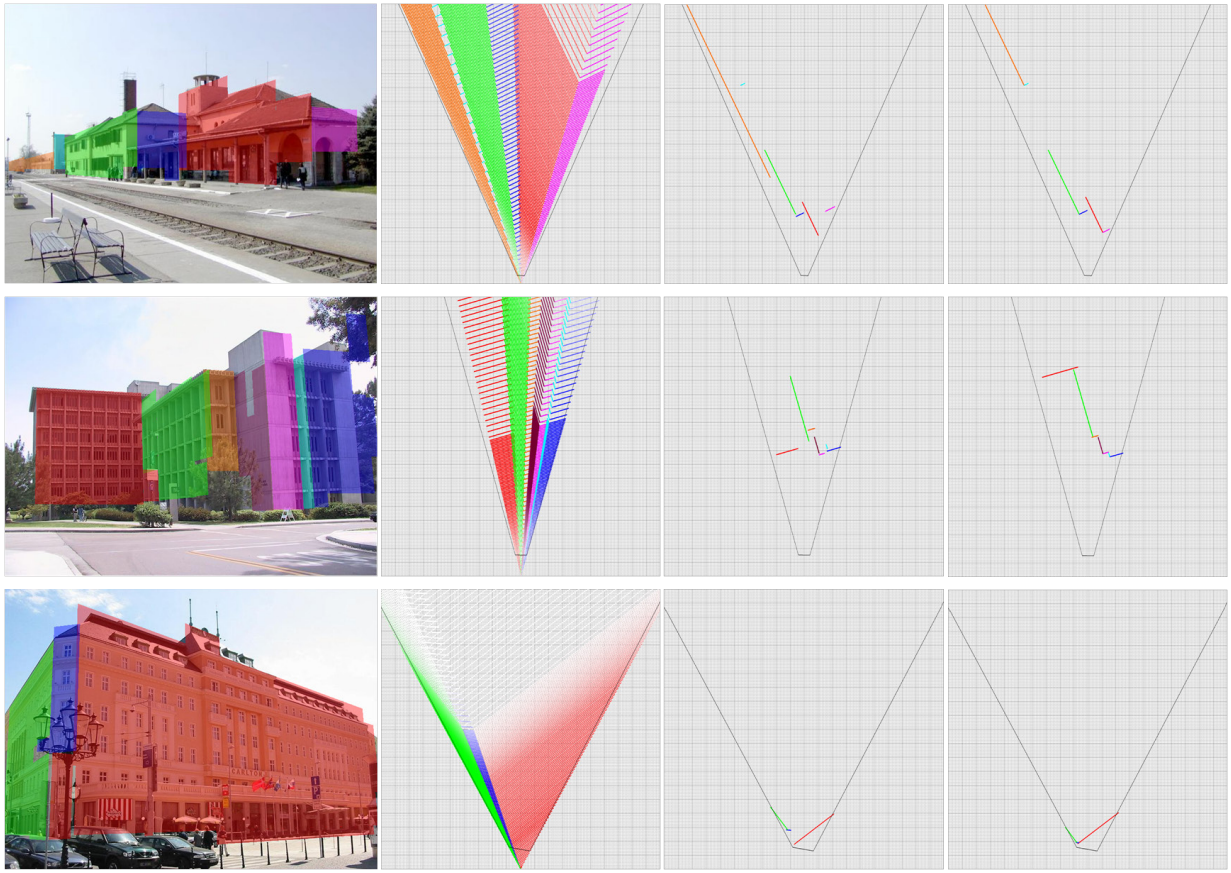

Figure 5.19: 3D facade layout estimation by our method on images from the Geometric Context dataset [33]. Left to right: 1) Original image overlaid with color shades representing quads from distinctive facade planes. Quads from the same distinctive facade plane share the same color. 2) Depth distribution of candidate distinctive facade planes before running the CRF inference. 3) Best locations of candidate distinctive facade planes before running the CRF inference. 4) Optimal locations of valid distinctive facade planes after running the CRF inference. The viewing boundary is marked with black lines, and the coarser grid spacing is 10m.

	Liu	Ours	Ours w/o CRF
Entire image	**0.144**	0.152	0.154
Exclude tree/foregnd.	0.130	**0.125**	0.127

Figure 5.20: Comparison of log depth error. The methods from left to right are the algorithm from Liu *et al.* [48], our algorithm, and our algorithm without CRF inference.

images, 51 images contain building facades and are used in evaluation.

We compare our method with the super-pixel based approach proposed by Liu *et al.* [48] on average log error of pixel-wise depth values. As our method returns the optimal orientation and depth of facade planes, it is straightforward to generate the depth values for pixels located in the facade region. As our method also returns camera pose, the depth values of pixels located on the ground can be easily computed as well. For pixels located on trees and foreground objects, our method is not intended to infer their depths. However, for the sake of comparison, we roughly estimate the depths of those pixels from the ground contact point of the semantic region they reside in. Note that this is a very rough estimation compared with the sophisticated model in [48] specifically trained on 400 training images to predict the depths of such pixels.

To make a fair comparison, we report results on the average log depth error of pixels over the entire image as well as over the image regions *excluding* trees and foreground objects. The results are shown in Figure 5.20. We can see that when evaluating on the entire image, Liu *et al.* achieves the best performance. However, if we exclude the tree and foreground regions, our method outperforms Liu's super-pixel based approach. Note that our method does not require any training in terms of depth prediction. We can also see that, again, removing the inter-planar geometric constraints from our approach degrades performance.

Some qualitative results of depth map estimation are shown in Figure 5.21. We could see that our method makes better depth estimation in facade regions than Liu's super-pixel based approach [48]. Also, as our method does not require training using ground-truth depth maps, it does not try to fit blindly to the defects of the ground truth such as the "infinity holes"at windows in row 2 (although it might negatively affect the final quantitative number). Depth estimation of

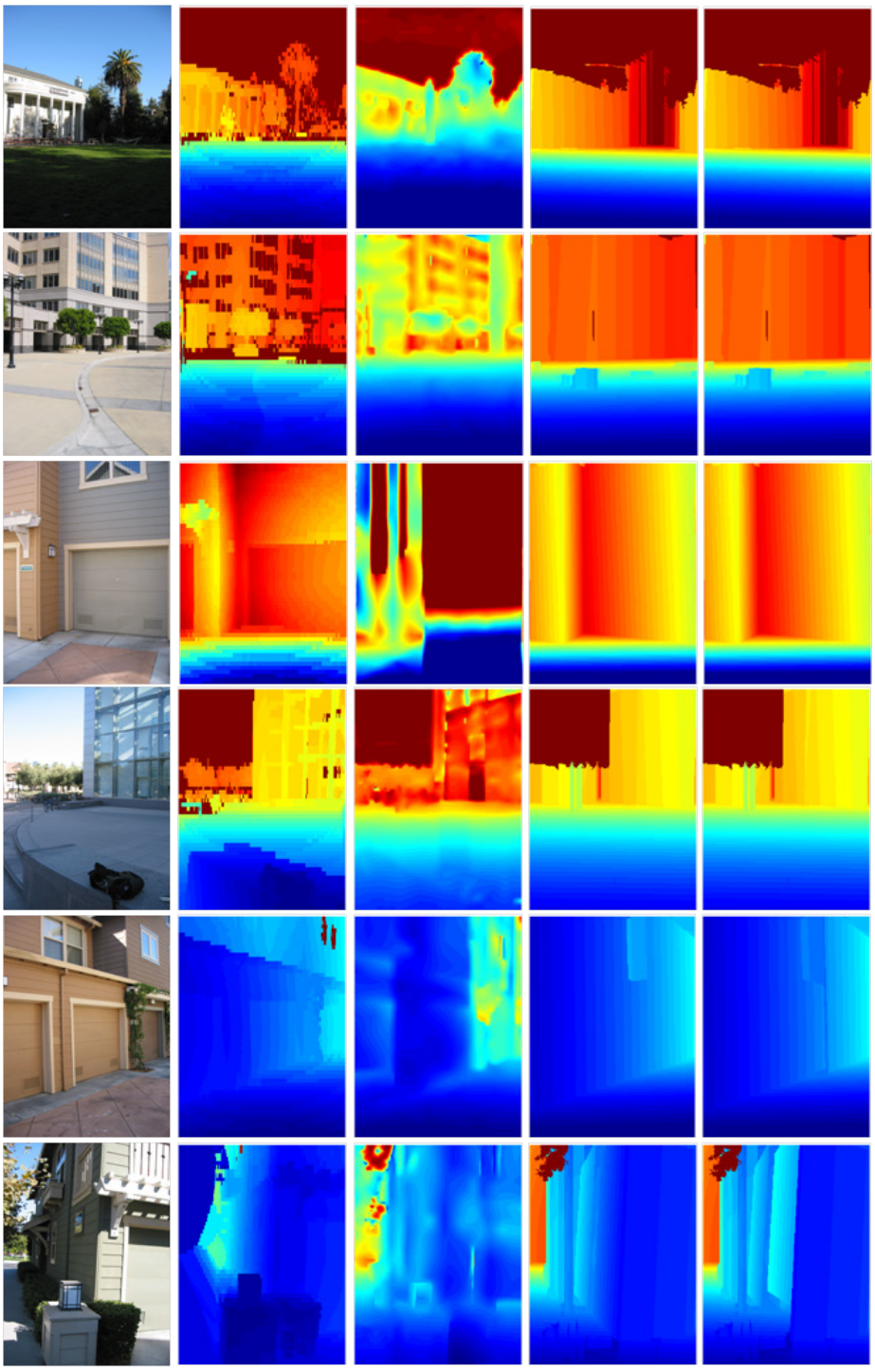

Figure 5.21: Qualitative comparisons of depth map estimation. Left to right: Original image; Ground truth; Liu [48]; Ours; Ours w/o CRF. Depth value increases from blue to red. The color code is scaled according to the ground-truth depth map.

ground region by our method is also better due to the 3D stage we have established using the facades. The benefit of imposing inter-planar geometric constraints could be seen in rows 5 and 6. In row 5, the depth of the second-floor camera-facing facade is correctly estimated after CRF inference, while in row 6, two misaligned left-facing facades in column 5 are correctly aligned after CRF inference in column 4.

In row 6, however, our method fails to identify the structure closest to the camera as it did not regard it as a facade region. Also, our method is not designed to estimate the depth of trees or foreground objects, and the make-shift algorithm described before only provides a very rough estimation. Liu's algorithm [48], nevertheless, is effective at estimating the depth of those regions. Please see row 4 for an example.

While our approach achieves a comparable performance with Liu's algorithm [48] in terms of depth map estimation, the output of our approach is beyond a depth map – it is a higher-level interpretation of building facades with mutually constrained planes, as is shown in Figure 5.22. The reason why inter-planar interactions do not play a significant role here is that facade planes in this dataset are relatively simple and have clearly visible ground contact lines in the image. As a result, cues from an individual facade plane are already sufficiently informative in locating it in 3D (please see the peaky distributions in the second column of Figure 5.22). However, when ambiguity is high (*e.g.*, the green facade plane in the first row of Figure 5.22), inter-planar interactions are critical in forcing facade planes into more geometrically plausible locations.

5.5 Conclusion

In this chapter, we propose a plane-based 3D facade reasoning system that recovers the 3D layout of building facades from a single 2D image of an urban scene. Multiple image cues and geometric constraints are incorporated into a probabilistic reasoning framework to achieve a coherent reconstruction of building facades in 3D space. Our method yields a more informative interpretation of the building facades in the scene while maintaining a competitive performance on several

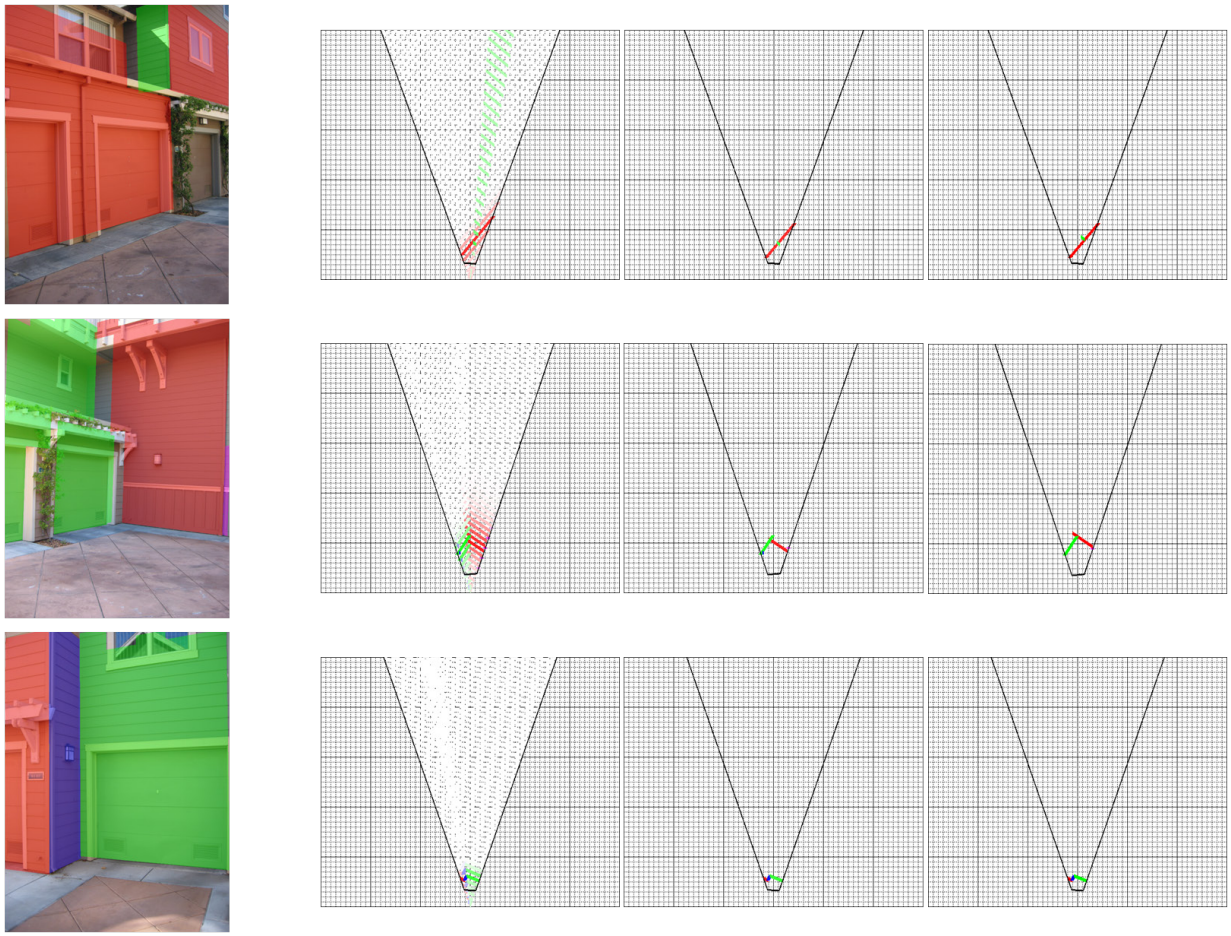

Figure 5.22: 3D facade layout estimation by our method on images from the Make3D dataset [48, 60]. Left to right: 1) Original image overlaid with color shades representing quads from distinctive facade planes. Quads from the same distinctive facade plane share the same color. 2) Depth distribution of candidate distinctive facade planes before running the CRF inference. 3) Best locations of candidate distinctive facade planes before running the CRF inference. 4) Optimal locations of valid distinctive facade planes after running the CRF inference. The viewing boundary is marked with black lines, and the coarser grid spacing is 10m.

quantitative measures. Compared with the block-based approach [25] that yields a coarse and qualitative modeling of the building facades, our method returns a numeric parameterization of a set of planes that compose the building facades. Compared with super-pixel based approaches that return a depth map [48], our method enables reasoning over planes and blocks and provides a higher-level understanding of the scene.

Chapter 6

Joint 3D Geometric Reasoning Over Objects and Building Facades

In the previous two chapters, we have discussed *separately* about recovering the 3D layout of objects and the 3D layout of building facades from a single image of an urban scene. We have shown that going beyond 2D reasoning and enforcing 3D geometric consistency not only produces a coherent 3D interpretation of the scene, but also improves the performance of traditional 2D visual tasks. In this chapter, we will discuss how to incorporate the algorithms of the previous two chapters into a single 3D geometric reasoning system, and what impacts the joint reasoning over both objects and facades would have on recovering the 3D structure behind an urban image.

6.1 Overview

Given a single image of an urban scene, jointly performing 3D geometric reasoning over both objects and building facades requires the modeling of five types of geometric relationships – between global 3D geometries (*i.e.*, gravity/ground) and individual objects, between global 3D geometries and individual facade planes, between individual objects, between individual facade planes, and between an object and a facade plane.

To our knowledge, very few works in the literature have attempted to build such a model. Hoiem *et al.* develops a system [35] that reasons about the relationships between global 3D geometries and objects, as well as relationships between objects and surfaces. The authors show that considering such relationships benefits scene understanding. However, in their model, objects do not take any volume, and surfaces are not in the sense of a set of mutually constrained planes with continuous 3D orientations. Rather, the interactions between objects and surfaces remain in the 2D domain – refining the surface labels of pixels belonging to objects, and using the "non-planar solid"surface label to improve object detection. An explicit 3D geometric relationship between a volumed object and a parameterized planar surface is not present in Hoiem's model. Gould *et al.* incorporate object semantics in the depth estimation task [22, 48], where the 2D locations of the pixels along a vertical line in the same object semantic region provide bounds of the depth of those pixels. While this idea proves effective, the authors treat objects as vertical cardboards, and therefore also do not model the interactions between a volumed object and a parameterized planar surface, or the intersections between two volumed objects. In addition, they do not explicitly seek to model and utilize global 3D geometries. In the "Blocks World"work by Gupta *et al.* [25], interactions between volumed facade blocks are modeled. However, their model is mostly qualitative. Moreover, objects are not involved in their model, nor are the global 3D geometries.

Built upon our works of the previous two chapters, we formally model all the five types of 3D geometric relationships and integrate them into our generalized-RANSAC-CRF framework in our joint reasoning system. The system produces a geometrically coherent 3D interpretation of the image in terms of all entities including gravity/ground, volumed objects, and building facades. Besides, joint reasoning over both objects and facades benefits facade layout recovery through the push-away effect from volumed objects and a better estimation of ground distance. Meanwhile, joint reasoning also benefits object layout recovery through the regulating of object headings by facade plane orientations. Also, joint reasoning yields the lowest error in the estimation of

global 3D geometries. Despite the involvement of all the entities, our reasoning system still remains tractable.

6.2 Joint Modeling of Objects and Building Facades

The task here is that, given a single image of an urban scene, we want to recover gravity direction, ground plane orientation, ground plane distance, object presence, object pose, object depth, object 2D location, facade plane presence, facade plane orientation, facade plane distance, and facade plane spatial extent. This task can be formalized as maximizing the following objective function:

$$
\begin{aligned}
&V(\mathbf{n_g}, \mathbf{n_p}, h_p, \mathbf{o_b}, \mathbf{\Theta_b}, \mathbf{d_b}, \mathbf{u_b}, \mathbf{o_s}, \mathbf{n_s}, \mathbf{d_s}, \mathbf{x_s} | I, f, H_t, H_f) = \sum_{i \in \mathbf{B}} \omega_b(o_{bi}, \mathbf{\Theta_{bi}}, d_{bi}, \mathbf{u_{bi}}, \mathbf{n_g}, \mathbf{n_p}, h_p | I, f, H_t) \\
&+ \sum_{i \in \mathbf{P}} \omega_s(o_{si}, \mathbf{n_{si}}, d_{si}, \mathbf{x_{si}}, \mathbf{n_p}, h_p | I, f, H_f) + \sum_{(i,j) \in \mathbf{B_n}} \varphi_b(o_{bi}, o_{bj}, \mathbf{\Theta_{bi}}, \mathbf{\Theta_{bj}}, d_{bi}, d_{bj}, \mathbf{u_{bi}}, \mathbf{u_{bj}}, \mathbf{n_p}, h_p | f) \\
&+ \sum_{(i,j) \in \mathbf{P_n}} \varphi_s(o_{si}, o_{sj}, d_{si}, d_{sj}, \mathbf{n_{si}}, \mathbf{n_{sj}}, \mathbf{x_{si}}, \mathbf{x_{sj}}, \mathbf{n_p}, h_p | f) + \sum_{i \in \mathbf{B_m}, j \in \mathbf{P_m}} \varphi_{sb}(o_{bi}, \mathbf{\Theta_{bi}}, d_{bi}, o_{sj}, d_{sj}, \mathbf{x_{sj}}, \mathbf{n_p}, h_p | f).
\end{aligned}
\tag{6.1}
$$

Here, $\mathbf{n_g}$ is gravity direction; $\mathbf{n_p}$ and h_p are ground plane orientation and distance; o_{bi}, $\mathbf{\Theta_{bi}}$, d_{bi}, and $\mathbf{u_{bi}}$ are the validity indicator, pose (pitch, roll, and yaw angles), depth, and 2D location (bounding box and landmark image coordinates) of object i, respectively; o_{si}, $\mathbf{n_{si}}$, d_{si}, and $\mathbf{x_{si}}$ are the validity indicator, orientation, distance, and spatial extent (image coordinates of facade corners) of facade plane i; I, f, H_t, and H_f are image features, focal length, real-world object height, and real-world facade height, respectively. Please refer to Figure 3.1 for an illustration of the geometric variables in the equation above.

ω_b and ω_s are the unary potentials of candidate objects and facade planes, respectively. Encoding global geometric constraints, the unary potentials measure the compatibility of each in-

dividual object and facade plane against the global 3D geometries as well as image features. $\mathbf{B}$ and $\mathbf{P}$ are the set of all candidate objects and facade planes, respectively. φ_b and φ_s are pairwise potentials between a pair of candidate objects and a pair of candidate facade planes, respectively, and φ_{sb} is the pairwise potential between a candidate object and a candidate facade plane. Encoding local geometric constraints, the pairwise potentials measure the compatibility between interacting local entities. $\mathbf{B_n}$, $\mathbf{P_n}$, $\mathbf{B_m}$, and $\mathbf{P_m}$ are the subsets of candidate objects and facade planes involved in pairwise geometric relationships.

While most potentials have been discussed in the previous two chapters, we summarize them here from a unified perspective and for the sake of completeness. The new potential introduced in this chapter is the pairwise potential between an object and a facade plane. We will describe it in more detail.

6.2.1 Object Unary Potential

As is mentioned in Section 4.4.3, the unary potential of an object is composed of the geometric compatibility score s_g, and the detector score s_d (which is essentially the image-feature compatibility score). Formally,

$$\omega_b(o_{bi}, \mathbf{\Theta_{bi}}, d_{bi}, \mathbf{u_{bi}}, \mathbf{n_g}, \mathbf{n_p}, h_p | I, f, H_t) = \begin{cases} s_g(\mathbf{\Theta_{bi}}, d_{bi}, \mathbf{u_{bi}}, \mathbf{n_g}, \mathbf{n_p}, h_p | f, H_t) + (s_d(\mathbf{u_{bi}}|I) - 0.5) & \text{if } o_{bi} = 1 \\ 0 & \text{if } o_{bi} = 0 \end{cases} \tag{6.2}$$

where the geometric compatibility score s_g is computed according to two sources of geometric constraints, as is described in Section 4.4.2. The first source measures the agreement between the global 3D geometries and the local estimate from object appearance by the regressor:

$$s_{g1}(\mathbf{\Theta_{bi}}, \mathbf{n_g}, \mathbf{n_p}) = \exp\{-\frac{(\tilde{\theta} - \theta_0)^2}{2\sigma_\theta^2}\} \cdot \exp\{-\frac{(\tilde{\gamma} - \gamma_0)^2}{2\sigma_\gamma^2}\} - 0.5, \tag{6.3}$$

where $\tilde{\theta}$ and $\tilde{\gamma}$ are the pitch and roll angles of the object computed from gravity direction $\mathbf{n_g}$ (for pedestrians) or ground plane orientation $\mathbf{n_p}$ (for cars), while θ_0 and γ_0 are the estimates of pitch and roll angles from the regressor, with variance σ_θ^2 and σ_γ^2, respectively.

The second source measures if the object height derived from the current values of geometric variables falls within a reasonable range according to prior knowledge H_t:

$$s_{g2}(d_{bi}, \mathbf{u_{bi}}, \mathbf{n_g}, \mathbf{n_p}, h_p | f, H_t) = \exp\{-\frac{\delta^2}{2\sigma^2}\} \cdot \exp\{-\frac{(\|\mathbf{X_t} - \mathbf{X_b}\| - H_t)^2}{2\sigma_H^2}\} - 0.5, \tag{6.4}$$

where $\mathbf{X_t}$ and $\mathbf{X_b}$ are the camera coordinates of the top and bottom landmarks computed from d_{bi} and $\mathbf{u_{bi}}$ with the algorithm detailed in Section 4.4.2 which also produces the variable δ. H_t and σ_H^2 define the reasonable range of object height. Also note that to handle noise in landmark estimation, we perturb the 2D location of the bottom landmark around the regressor output, and the highest s_{g2} score is taken.

After s_{g1} and s_{g2} are computed, $s_g = (s_{g1} + s_{g2})/2$.

6.2.2 Facade Plane Unary Potential

As is discussed in Section 5.2.2, two contributors to the unary potential of a facade plane consist of a image-feature compatibility score and a geometric compatibility score. The image-feature compatibility score

$$S_r(\mathbf{n_{si}}, \mathbf{x_{si}} | I) = \frac{1}{N(\mathbf{Q})} \sum_{n \in \mathbf{Q}} (s_n^{(bldg)} + g_n(\mathbf{n_{si}}) + < \mathbf{n_{si}}, \mathbf{v_n} >) \tag{6.5}$$

measures how likely the candidate facade plane belongs to the building semantic region ($s_n^{(bldg)}$),

and how consistent the orientation $\mathbf{n_{si}}$ of the candidate facade plane is with respect to the surface geometry g_n and orientation map $\mathbf{v_n}$.

The geometric compatibility score

$$S_d(\mathbf{n_{si}}, d_{si}, \mathbf{x_{si}}, \mathbf{n_p}, h_p | I, f, H_f) = \min\{1 - \frac{1}{N(\mathbf{A})} \sum_{n \in \mathbf{A}} s_n^{(gnd)}, 1 - \frac{1}{N(\mathbf{B})} \sum_{n \in \mathbf{B}} s_n^{(bldg)}, H_f(\frac{\|\mathbf{x_{st}} - \mathbf{x_{sb}}\|}{\|\mathbf{x_{sm}} - \mathbf{x_{sb}}\|} \cdot h_p))\} \quad (6.6)$$

measures the plausibility of the ground contact line (the first and second terms on the right-hand side) and 3D height (the third term on the right-hand side) of the candidate facade plane. Please refer to Section 5.2.2 for more details about the image-feature and geometric compatibility scores.

Note that in Chapter 5, we estimate global 3D geometries such as $\mathbf{n_g}$ and $\mathbf{n_p}$ in a preprocessing stage, and they are fixed throughout subsequent reasoning. In this chapter, we will integrate 3D facade reasoning into the generalized-RANSAC-CRF framework in which there could be multiple hypotheses of global 3D geometries. Therefore, we introduce an additional term to the unary potential of a facade plane. This term checks if the 3D orientation of the facade plane is perpendicular to the gravity direction:

$$S_g(\mathbf{n_{si}}, \mathbf{n_g}) = \exp(-\frac{\arcsin(< \mathbf{n_{si}}, \mathbf{n_g} >)^2}{2\sigma^2}), \quad (6.7)$$

where σ is a parameter that controls sensitivity.

Given S_r, S_d, and S_g, the unary potential of a facade plane is defined as

$$\omega_s(o_{si}, \mathbf{n_{si}}, d_{si}, \mathbf{x_{si}}, \mathbf{n_p}, h_p | I, f, H_f) =$$

$$\begin{cases} [(S_r(\mathbf{n_{si}}, \mathbf{x_{si}}|f) \cdot S_d(\mathbf{n_{si}}, d_{si}, \mathbf{x_{si}}, \mathbf{n_p}, h_p|I, f, H_f) - 0.25) + (S_g(\mathbf{n_{si}}, \mathbf{n_g}) - 0.5)]/2 & \text{if } o_{si} = 1 \\ 0 & \text{if } o_{si} = 0 \end{cases} \tag{6.8}$$

where the unary potential is the average of the neutrality-offset $S_r \cdot S_d$ and S_g.

6.2.3 Object Pairwise Potential

The pairwise potential between two candidate objects is determined by two local geometric constraints, as is described in Section 4.4.2. The first geometric constraint is the depth ordering constraint, with penalty

$$s_{ij}^{(dep)}(d_{bi}, d_{bj}, \mathbf{u_{bi}}, \mathbf{u_{bj}}) = -(|R_{ij}|/|R_j|)^{\lambda}, \tag{6.9}$$

where $|R_{ij}|$ is the overlapping area of the bounding boxes of objects i and j, and $|R_j|$ is the bounding box area of the farther object. This constraint penalizes the case in which the farther object is mostly or totally occluded by the closer one.

The second geometric constraint is the space occupancy constraint, with penalty

$$s_{ij}^{(ocp)}(\mathbf{\Theta_{bi}}, \mathbf{\Theta_{bj}}, d_{bi}, d_{bj}, \mathbf{u_{bi}}, \mathbf{u_{bj}}, \mathbf{n_p}, h_p|f) = -|R_i' \cap R_j'|/|R_i' \cup R_j'|, \tag{6.10}$$

where R_i' is the footprint of object i on the ground plane. The ratio of the intersection and union footprint areas of two adjacent objects reflects the degree of violation of the space occupancy constraint. Please see Section 4.4.2 for more details.

The pairwise potential between two candidate objects can now be defined as

$$\varphi_b(o_{bi}, o_{bj}, \mathbf{\Theta_{bi}}, \mathbf{\Theta_{bj}}, d_{bi}, d_{bj}, \mathbf{u_{bi}}, \mathbf{u_{bj}}, \mathbf{n_p}, h_p|f) =$$

$$\begin{cases} \min\{s_{ij}^{(dep)}(d_{bi}, d_{bj}, \mathbf{u_{bi}}, \mathbf{u_{bj}}), s_{ij}^{(ocp)}(\mathbf{\Theta_{bi}}, \mathbf{\Theta_{bj}}, d_{bi}, d_{bj}, \mathbf{u_{bi}}, \mathbf{u_{bj}}, \mathbf{n_p}, h_p|f)\} & \text{if } o_{bi} \neq 0 \text{ and } o_{bj} \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.11}$$

6.2.4 Facade Plane Pairwise Potential

Interactions between a pair of facade planes consist of three types of local geometric constraints – convex-corner constraint, occlusion constraint, and alignment constraint – depending on the relative orientation of the planes (please refer to Section 5.2.3). Incorporating the three types of constraints, the pairwise potential of facade planes can be written as

$$\varphi_s(o_{si}, o_{sj}, d_{si}, d_{sj}, \mathbf{n_{si}}, \mathbf{n_{sj}}, \mathbf{x_{si}}, \mathbf{x_{sj}}, \mathbf{n_p}, h_p|f) =$$

$$\begin{cases} -\infty & \text{if } o_{si} \neq 0 \text{ and } o_{sj} \neq 0 \text{ and } \mathbf{x_{si}}|\mathbf{x_{sj}} \text{ and } \mathbf{n_{si}} \vee \mathbf{n_{sj}} \text{ and } d_{si} \veebar d_{sj} \\ -\max\{0, d'_{j1}, d'_{j2}\}^2/(2\sigma_1^2) & \text{if } o_{si} \neq 0 \text{ and } o_{sj} \neq 0 \text{ and } \mathbf{x_{si}}|\mathbf{x_{sj}} \text{ and } \mathbf{n_{si}} \wedge \mathbf{n_{sj}} \\ -(d'_i - d'_j)^2/(2\sigma_2^2) & \text{if } o_{si} \neq 0 \text{ and } o_{sj} \neq 0 \text{ and } \mathbf{n_{si}} = \mathbf{n_{sj}} \text{ and } Q_s(i,j) = 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.12}$$

Here, the first row on the right-hand side encodes the convex-corner constraint, in which $\mathbf{x_{si}}|\mathbf{x_{sj}}$ means facade planes i and j are adjacent in the image, $\mathbf{n_{si}} \vee \mathbf{n_{sj}}$ means the orientations of facade planes i and j form a convex corner, and $d_{si} \veebar d_{sj}$ means their distances make them fail to connect along a common fold. The second row encodes the occlusion constraint, in which we assume

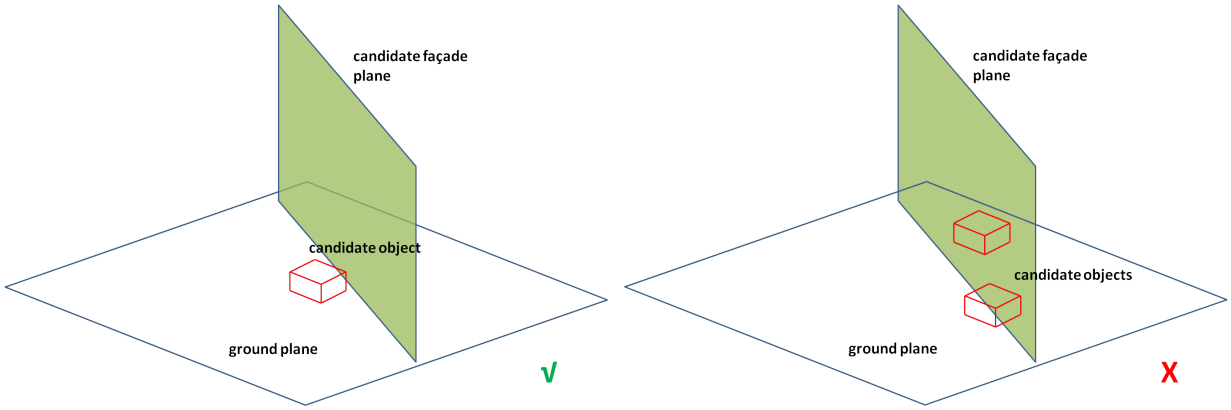

Figure 6.1: Illustration of the object-facade plane interaction. The candidate object and facade plane in the left image could coexist, while neither the candidate objects in the right image could coexist with the candidate facade plane.

without loss of generality that facade plane i is occluded by facade plane j, and $\mathbf{n_{si}} \wedge \mathbf{n_{sj}}$ means their orientations form a concave corner. Please refer to Figure 5.6 for the definition of d'_{j1} and d'_{j2}. The third row encodes the alignment constraint, in which $Q_s(i,j) = 0$ means the two facade planes do not occlude each other through other planes connected to them via convex corners (see Figure 5.7), and d'_i and d'_j are the facade plane distance in the ground coordinate system (see Figure 3.1).

6.2.5 Object-Facade Plane Pairwise Potential

If an object and a facade plane coexist in a scene, the facade plane must not occlude the object either partially or entirely, as is illustrated in Figure 6.1. A geometric constraint exists between an object and a facade plane if the viewing angle range of the footprint of the object overlaps with that of the ground contact line of the facade plane. Here, viewing angle is defined by the line connecting the origin of the ground coordinate system with a point on the footprint or the ground contact line, as is illustrated by the dashed lines in Figure 6.2. The pairwise potential between a candidate object and a candidate facade plane is computed based on the farthest point of the object footprint located behind the facade plane:

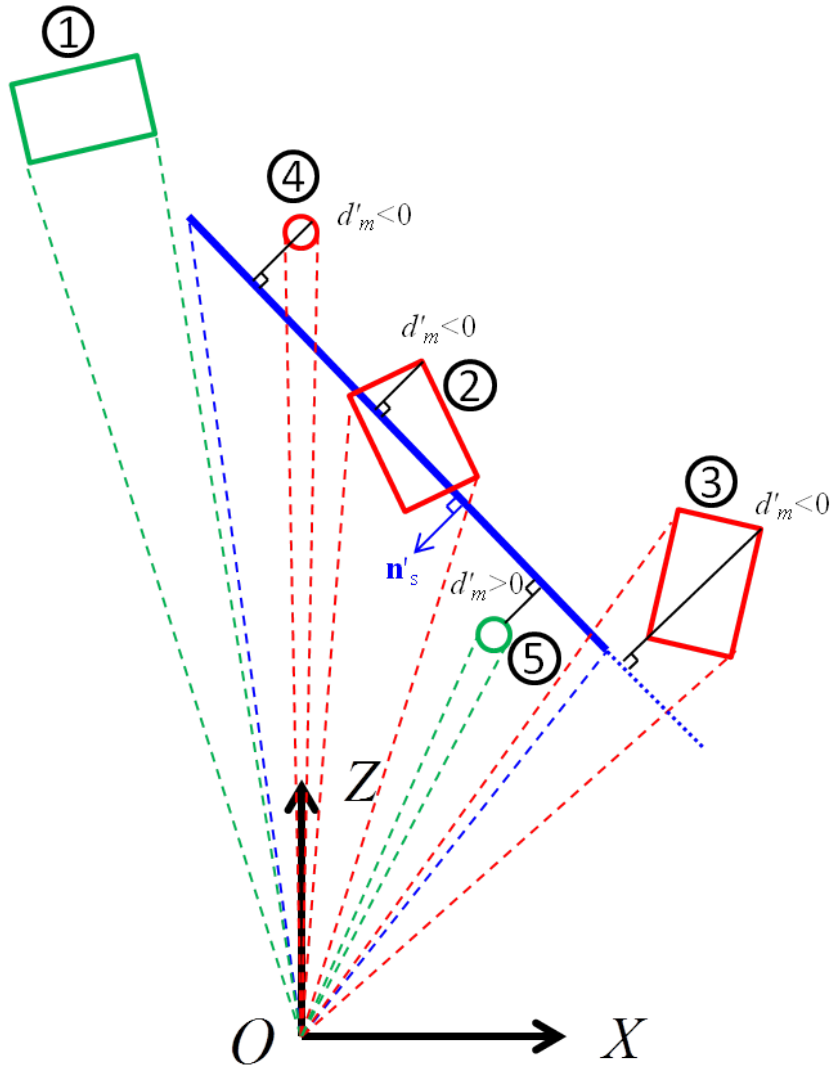

Figure 6.2: A top-down view of the interactions between an object and a facade plane. The thick blue line is the ground contact line of the candidate facade plane, and the rectangles and circles represent the footprints of candidate cars and pedestrians, respectively. Dashed lines delimit the viewing angle range of an object or the facade plane. d'_m is the smallest (*i.e.*, most negative) signed distance from the object footprint to the ground contact line of the facade plane. As the viewing angle range of object 1 does not overlap with that of the facade plane, object 1 has no interaction with the facade plane and therefore their coexistence incurs no penalty. The rest of the objects interact with the facade plane, among which only object 5 does not incur a penalty due to its positive value of d'_m, meaning it is completely in front of the facade plane.

$$\varphi_{sb}(o_{bi}, \boldsymbol{\Theta}_{\mathbf{bi}}, d_{bi}, o_{sj}, d_{sj}, \mathbf{x}_{\mathbf{sj}}, \mathbf{n}_{\mathbf{p}}, h_p | f) \;=\; \begin{cases} \min\{0, d'_m\}^2/(2\sigma_3^2) & \text{if } \mathbf{A}_{\mathbf{bi}} \cap \mathbf{A}_{\mathbf{sj}} \neq \varnothing \\ 0 & \text{otherwise} \end{cases} \tag{6.13}$$

where $\mathbf{A}_{\mathbf{bi}}$ and $\mathbf{A}_{\mathbf{sj}}$ are the viewing angle ranges of object i and facade plane j, respectively. d'_m is the smallest (*i.e.*, most negative) signed distance between a point on the object footprint and the ground contact line of the facade plane. When $d'_m \geq 0$, the object and facade plane do not violate the geometric constraint and therefore no penalty is imposed on the pairwise potential. Please see Figure 6.2 for illustration.

In most urban scenes, building facades usually align with the street, and cars usually travel or park along the street. Based on this observation, we use the facade plane interacting with a car to refine the yaw angle of the car produced by the appearance-based regressor. More specifically, if the heading of the car is within 30 degrees from the horizontal vanishing direction (or the orthogonal one) of the facade plane, its heading is set as the latter, and the refined heading is used to compute the footprint of the car.

6.3 Implementation Details

The objective function in Equation 6.1 involves multiple three-dimensional continuous vectors that, under a reasonably fine quantization, have an intractably large state space even after being factorized into smaller cliques. To make the inference tractable, we adopt and integrate the generalized-RANSAC-CRF scheme proposed in Chapter 4 and the quadrilateral-based sampling algorithm proposed in Chapter 5.

The overall scheme of our joint reasoning algorithm is illustrated in Figure 6.3, and the high-level procedures are listed in Figure 6.4.

There are three major factors that contribute to inference tractability. Firstly, the general-

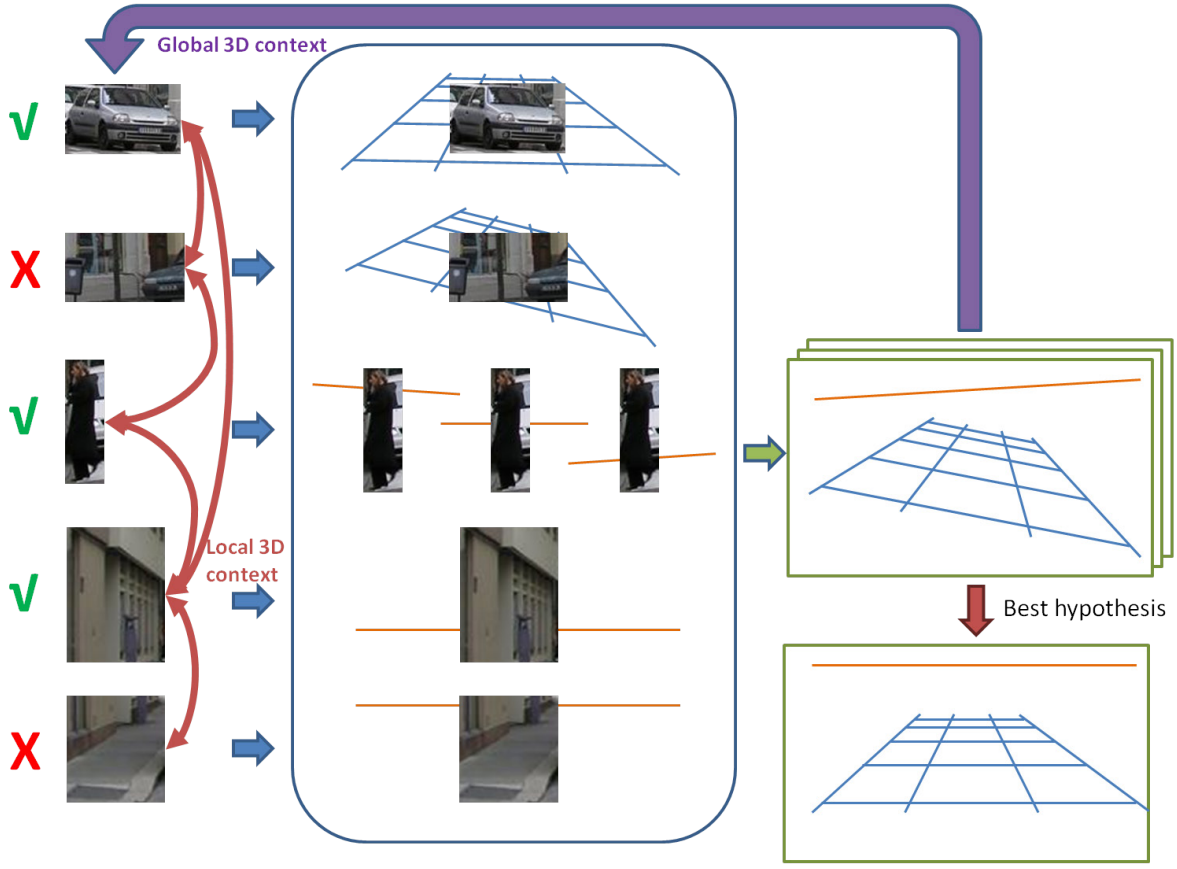

Figure 6.3: The overall scheme of our joint 3D geometric reasoning over objects and building facades. Here, gravity direction is represented by the orange horizon line, and the ground plane is represented by the blue mesh.

1. Preprocessing
   - 1.1. Detect objects and estimate their poses and landmark locations (Section 3.2)
   - 1.2. Compute soft semantic labels (Sections 3.3)
   - 1.3. Compute soft surface layout labels (Section 3.4)
   - 1.4. Estimate vanishing points, focal length, and horizons (Section 3.5)
   - 1.5. Compute vertical orientation map (Section 3.6)
2. Generate global 3D geometry hypotheses (Section 4.3)
   - 2.1. For each candidate object, compute its estimated distribution of $\mathbf{n_g}$ (for pedestrian) or $\mathbf{n_p}$ (for car)
   - 2.2. For the vertical/horizontal surface region, compute its estimated distribution of $\mathbf{n_g}/\mathbf{n_p}$
   - 2.3. Compute an estimated distribution of $\mathbf{n_g}$ and $\mathbf{n_p}$ from the horizons obtained in Proc. 1.4
   - 2.4. Generate hypotheses of $\mathbf{n_g}$ and $\mathbf{n_p}$ from a pool of distributions using the generalized RANSAC algorithm
   - 2.5. For each 2-tuple hypothesis $(\tilde{\mathbf{n}}_\mathbf{g}, \tilde{\mathbf{n}}_\mathbf{p})$, compute a distribution of h_p from each object candidate
   - 2.6. Generate hypotheses of h_p from a pool of distributions using the generalized RANSAC algorithm, and supplement the original 2-tuple hypothesis to form a set of 3-tuple hypotheses $(\tilde{\mathbf{n}}_\mathbf{g}, \tilde{\mathbf{n}}_\mathbf{p}, \tilde{h}_p)$
3. Obtain a set of candidate façade planes
   - 3.1. Under the $(\tilde{\mathbf{n}}_\mathbf{p}, \tilde{h}_p)$ value corresponding to the strongest mode, run the quadrilateral-based sampling algorithm (Proc. 2 in Figure 5.8)
4. Evaluate global 3D geometry hypotheses
   - 4.1. For each hypothesis $(\tilde{\mathbf{n}}_\mathbf{g}, \tilde{\mathbf{n}}_\mathbf{p}, \tilde{h}_p)$,
     - 4.1.1. Construct a CRF in which each node corresponds to a candidate object or façade plane, and each edge corresponds to a pairwise geometric constraint
     - 4.1.2. For each candidate object, compute its unary potential over state space $\mathbf{S}_{bi} = \{o_{bi}\}$ according to Equ. (6.2).
     - 4.1.3. For each candidate façade plane, compute its unary potential over state space $\mathbf{S}_{si} = \{d_{si}\} \cup \{o_{si} = 0\}$ according to Equ. (6.8)
     - 4.1.4. For each pair of interacting candidate objects, compute its pairwise potential over state space $\mathbf{S}_{bi} \times \mathbf{S}_{bj}$ according to Equ. (6.11)
     - 4.1.5. For each pair of interacting candidate façade planes, compute its pairwise potential over state space $\mathbf{S}_{si} \times \mathbf{S}_{sj}$ according to Equ. (6.12)
     - 4.1.6. For each pair of interacting candidate object and façade plane, compute its pairwise potential over state space $\mathbf{S}_{bi} \times \mathbf{S}_{sj}$ according to Equ. (6.13)
     - 4.1.7. Perform MAP inference over the CRF using max-product belief propagation, and compute the optimal value of the objective function in Equ. (6.1)
   - 4.2. Select the hypothesis with the highest optimal value of the objective function and refine the hypothesis with valid local entities

Figure 6.4: Procedures of our joint 3D geometric reasoning over objects and building facades.

ized RANSAC algorithm serves to isolate the global 3D geometries from the CRF inference as given hypotheses. This way, all the global geometric variables ($\mathbf{n_g}$, $\mathbf{n_p}$, h_p) can be placed in the potential terms as given variables. The generalized RANSAC approach consists of generating and evaluating global 3D geometry hypotheses. The hypothesis generation process (Proc. 2 in Figure 6.4) is exactly the same as the one described in Section 4.3 because all candidate facade planes share the same vertical vanishing direction which is already used in Section 4.3. The hypothesis evaluation process (Proc. 4 in Figure 6.4) is different due to the introduction of candidate facade planes. However, the resulting CRF is still tractable due to the sparse graph and small state space. The isolation of the global geometric variables from the CRF inference does not negatively impact inference quality due to the nature of RANSAC as a robust optimization algorithm.

Secondly, the quadrilateral-based sampling algorithm described in Chapter 5 isolates facade orientation $\mathbf{n_{si}}$ and spatial extent $\mathbf{x_{si}}$ from the CRF inference as known variables. This way, $\mathbf{n_{si}}$ and $\mathbf{x_{si}}$ can also be placed in the potential terms as given variables. As individual and mutual geometric compatibilities of facade planes are extensively utilized during the quadrilateral-based sampling process, removing $\mathbf{n_{si}}$ and $\mathbf{x_{si}}$ from the CRF inference has little impact on inference quality.

Thirdly, object pose $\mathbf{\Theta_{bi}}$ and landmarks $\mathbf{u_{bi}}$ are also isolated from the CRF inference. Object pose is estimated by an appearance-based regressor and refined by adjacent facade plane (see Section 6.2.5) before its value is used to compute potentials. Object landmark locations are also estimated by an appearance-based regressor, and perturbed to obtain the optimal geometric compatibility (see Section 6.2.1) before its value is used to compute potentials. Therefore, $\mathbf{\Theta_{bi}}$ and $\mathbf{u_{bi}}$ can also be placed in the potential terms as given variables. Since d_{bi} has a deterministic relationship with $\mathbf{\Theta_{bi}}$ and $\mathbf{u_{bi}}$ given the global 3D geometries, it also becomes a given variable. Due to the local optimization preformed before fixing their values, the impact of isolating $\mathbf{\Theta_{bi}}$ and $\mathbf{u_{bi}}$ from the CRF inference is also minimal.

After we have labeled all the aforementioned variables as given variables in the potential terms, the five potentials now become $\omega_b(o_{bi})$, $\omega_s(o_{si}, d_{si})$, $\varphi_b(o_{bi}, o_{bj})$, $\varphi_s(o_{si}, o_{sj}, d_{si}, d_{sj})$, and $\varphi_{sb}(o_{bi}, o_{sj}, d_{sj})$.[1] As a result, the state space $\mathbf{S}_{bi}$ for a candidate object is simply binary, and the state space $\mathbf{S}_{si}$ for a candidate facade plane has a dimensionality of only $M + 1$, where M is the number of quantization bins of facade plane depth. Such small state spaces make the inference very fast.

To further speed up the inference, we do not perform facade decomposition for each hypothesis. Rather, as is described by Proc. 3 in Figure 6.4, we run facade decomposition only once under the ground plane parameters corresponding to the strongest mode found in the hypothesis generation process. When evaluating each hypothesis, the geometric properties and potentials of each candidate facade plane are re-calculated under the current ground plane hypothesis. As facade decomposition (not layout) is not very sensitive to ground plane parameters, such a strategy has little impact on the final result.

6.4 Experiments

We evaluate our joint 3D reasoning system on the LabelMe dataset [34] and the Geometric Context dataset [33]. The LabelMe dataset provides ground-truth bounding boxes of cars and pedestrians, as well as ground-truth horizon. Therefore, we use this dataset to evaluate object detection and viewpoint estimation. Among the 422 test images, 225 of them contain urban scenes with building facades, and the total number of objects in those images is 1154. We conduct our experiments on those images. The Geometric Context dataset provides ground-truth surface orientation labels. Therefore, we use this dataset to evaluate facade decomposition. Among the 250 test images, 55 of them contain building facades, and those images are used in our evaluation.

All the parameters in our system are set the same as in Chapters 4 and 5. The parameter σ_3 in

[1] We have omitted the given variables for clarity.

the object-facade plane pairwise term φ_{sb} is set as 0.5. For a typical 800-by-600 image, it takes around 3 minutes to generate hypotheses and 10 minutes to evaluate those hypotheses and obtain the optimal results.

Object Detection: In this experiment, we want to compare the object detection performance of our system with other competitive object detection algorithms, and to evaluate to what extent 3D geometric reasoning involving objects and building facades might help improve object detection performance.

The detection ROC curve is shown in Figure 6.5, and the average precision and detection rate at 0.5 false positive per image are listed in Figure 6.6. Here, we compare our approach with the latest release of the Deformable Part Model (DPM) detector [19] and Hoiem's geometric reasoning algorithm [34]. Both our algorithm and Hoiem's algorithm use the output of the DPM detector running in high-recall mode as input candidate objects. We can see that our algorithm ("Jnt (full)") outperforms the DPM detector under all false positive rates, and especially in the domain of lower false positive rates, where the ability to resolve visual ambiguity plays a critical part. The average precision (AP) of our algorithm is almost 7% higher than the AP of the DPM detector. At 0.5 false positive per image, our algorithm correctly detects 116 more true positives than the DPM detector. The performance of Hoiem's algorithm is better than the DPM detector in the domain of higher false positive rates (above 3 false positive per image), but deteriorates fast when false positive rate gets lower. This is because of its use of bounding box height as object height, which is often inaccurate for cars especially when cars are closer to the camera.

We also compare our full system performance with the performance when we do not consider interactions among local entities. The curve of the latter case is indicated by "Jnt (w/o CRF)"in Figure 6.5. We can see that omitting pairwise geometric compatibilities between adjacent local entities results in a big drop in detection performance. Nevertheless, since the "Jnt (w/o CRF)"curve still considers the geometric constraints between global 3D geometries and individual local entities, its overall performance is still better than the DPM detector.

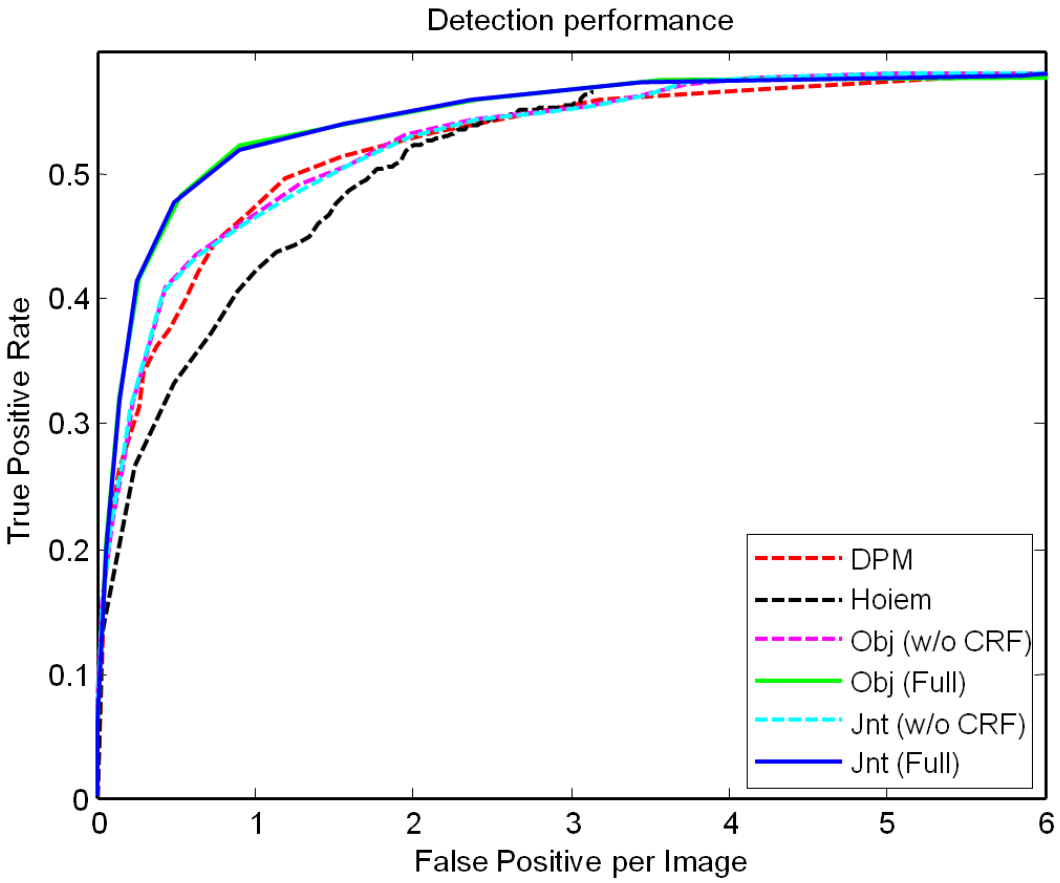

Figure 6.5: Comparison of object detection performance in terms of ROC curve. Here, "DPM"and "Hoiem"show the results of the Deformable Part Model [19] and the algorithm in [34], respectively. The result of our full system is shown by "Jnt (Full)". Our joint reasoning system without considering interactions among local entities is represented by "Jnt (w/o CRF)". The results of our reasoning system when leaving out building facades are indicated by "Obj (Full)"and "Obj (w/o CRF)".

	DPM	Hoiem	Obj (w/o CRF)	Obj (Full)	Jnt (w/o CRF)	Jnt (Full)
Average Precision	47.5%	43.9%	48.8%	**54.1%**	48.8%	**54.1%**
Detection Rate	38.7%	34.2%	42.5%	**48.0%**	42.5%	**48.0%**

Figure 6.6: Comparison of object detection performance in terms of average precision and detection rate at 0.5 false positive per image. Here, columns "DPM"and "Hoiem"list the results of the Deformable Part Model [19] and the algorithm in [34], respectively. The result of our full system is listed in column "Jnt (Full)". The result of our joint reasoning system without considering interactions among local entities is listed in column "Jnt (w/o CRF)". The results of our reasoning system when leaving out building facades are listed in columns "Obj (Full)"and "Obj (w/o CRF)".

To see if the introduction of building facades help in object detection, we obtain the detection results when we leave out building facades in the scene. The curves plotting those results are indicated by "Obj (Full)"and "Obj (w/o) CRF". It turns out building facades help little, if any, in improving object detection rate. We observe in our experiment that whenever an object and facade plane collide in 3D, it is always the object pushing away the facade plane, rather than the facade plane invalidating the object. This is due to the much higher flexibility in the depth of facade planes – pushing a facade plane one meter further away from the camera has a much lower cost than invalidating an otherwise plausible object.

Some qualitative examples are shown in Figures 6.7– 6.11. Each figure contains two examples, one in the first two rows and the other in the bottom two rows. For each example, object detection results using our full system is shown in image (d), while the results without utilizing 3D geometric constraints among local entities are shown in image (c). Here, a green bounding box means 3D geometric reasoning agrees with the DPM detector that the candidate object is valid; a red bounding box means 3D geometric reasoning recovers a candidate object originally rejected by the DPM detector; a blue bounding box means 3D geometric reasoning rejects a candidate object originally accepted by the DPM detector. Also, each color-shaded region indicates a distinctive facade plane. Comparing images (d) with (c), we can see that while the geometric constraints between global 3D geometries and individual local entities often help improve the detection performance over the DPM detector (*e.g.*, Figure 6.7(c_2), Figure 6.8(c_1), Figure 6.9(c_2), Figure 6.11(c_1)(c_2)), they also make mistakes when a false detection is also highly compatible with global 3D geometries (*e.g.*, Figure 6.7(c_1), Figure 6.8(c_2), Figure 6.9(c_1), Figure 6.10(c_2)). Incorporating geometric relationships among local entities further improve detection performance by resolving such ambiguities. For example, in Figure 6.7(d_1), pairwise geometric constraints rejected a false detection (blue box) not discovered in Figure 6.7(c_1), and corrected a mistakenly recovered false detection (red box) in Figure 6.7(c_1).

Car Heading: Although building facades do not help in improving object detection rate,

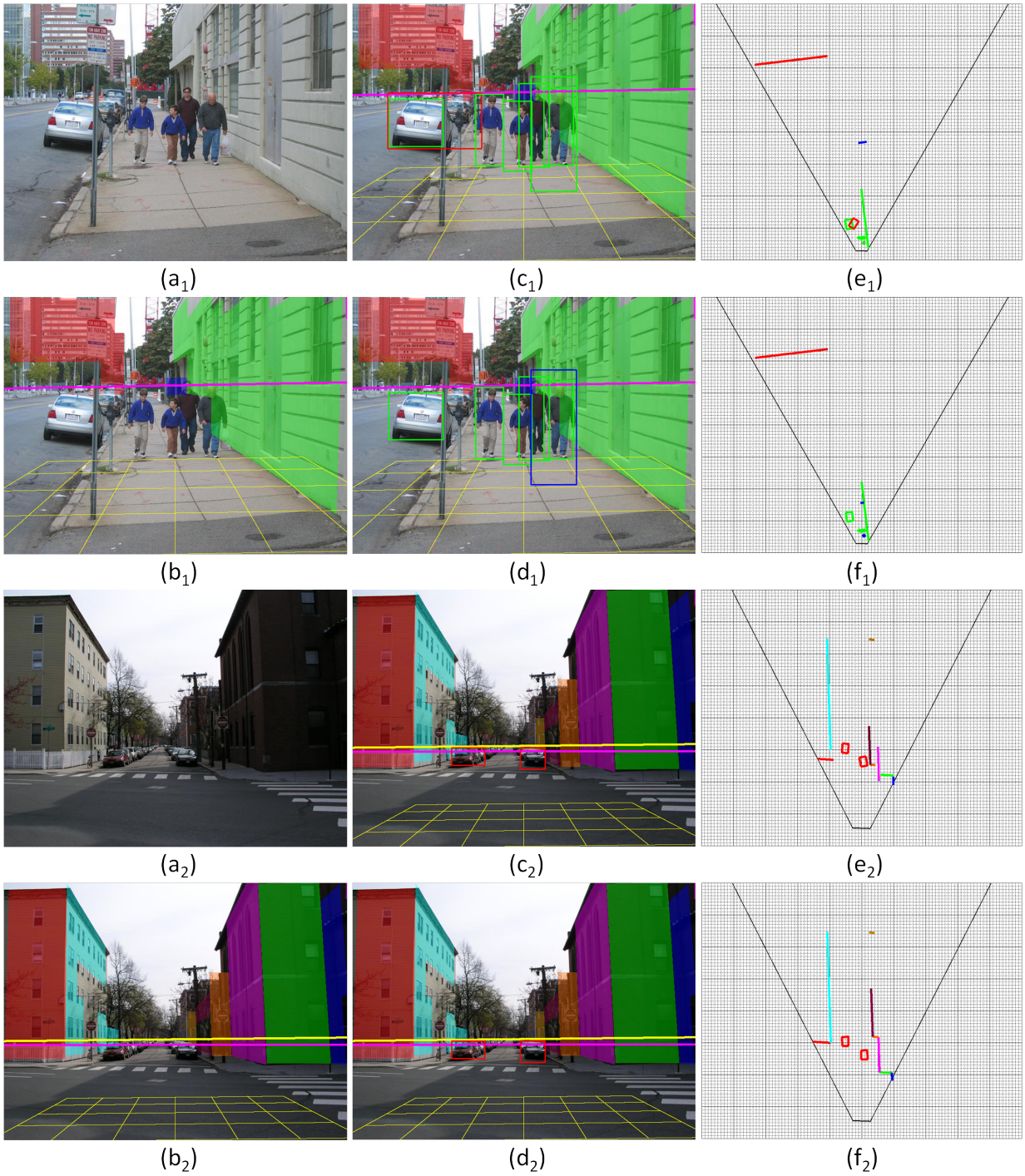

Figure 6.7: Joint 3D geometric reasoning of objects and building facades using our proposed approach. This figure shows two examples, located in the top two rows and bottom two rows, respectively. (a): Original image. (b): Reasoning only using facades. (c): Reasoning using both objects and facades, yet omitting interactions among local entities. (d): Reasoning using both objects and facades with our full system. (e): top-down view for the result in (c). (f): top-down view for the result in (d). Please see text for more details.

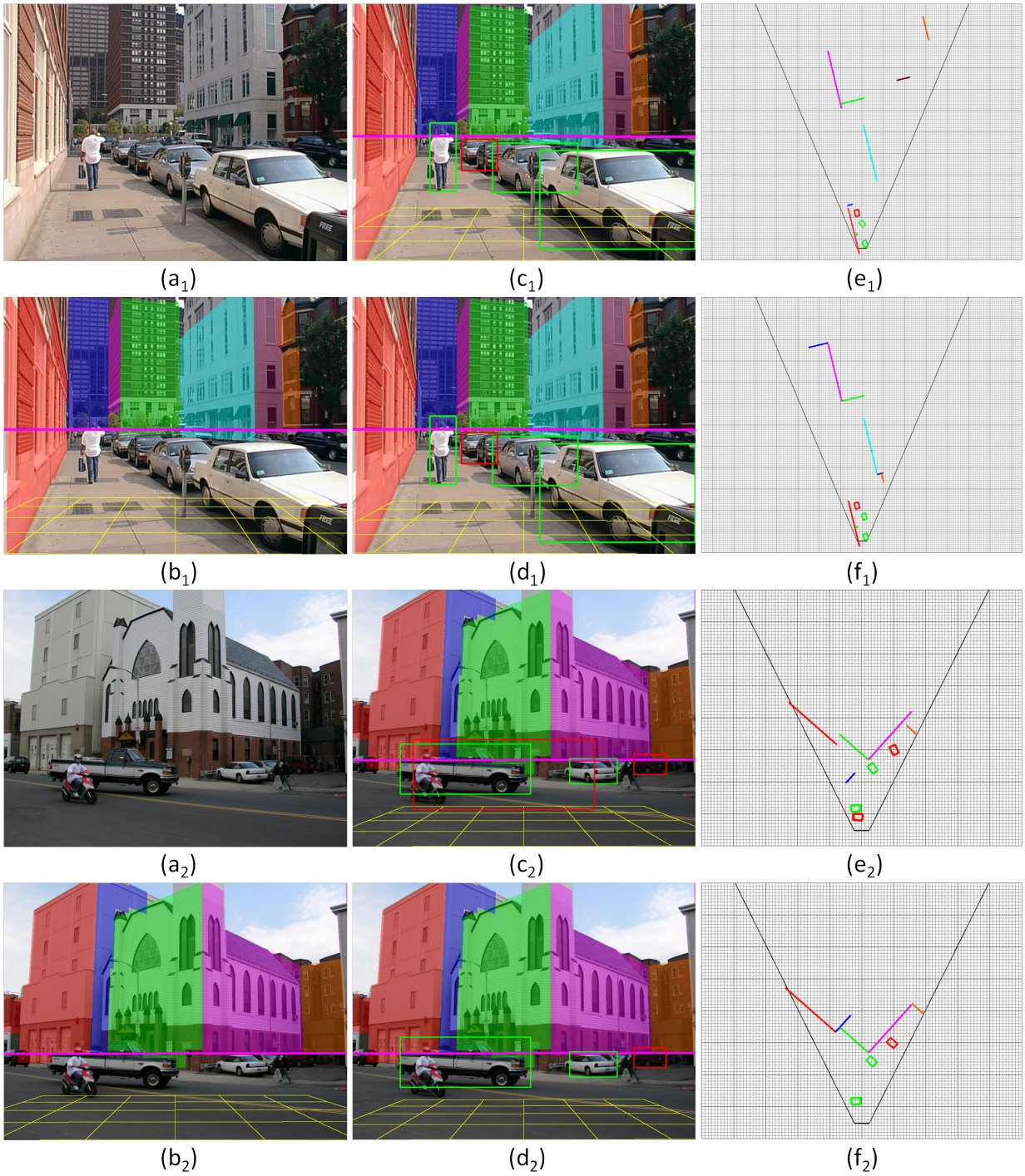

Figure 6.8: Joint 3D geometric reasoning of objects and building facades using our proposed approach. This figure shows two examples, located in the top two rows and bottom two rows, respectively. (a): Original image. (b): Reasoning only using facades. (c): Reasoning using both objects and facades, yet omitting interactions among local entities. (d): Reasoning using both objects and facades with our full system. (e): top-down view for the result in (c). (f): top-down view for the result in (d). Please see text for more details.

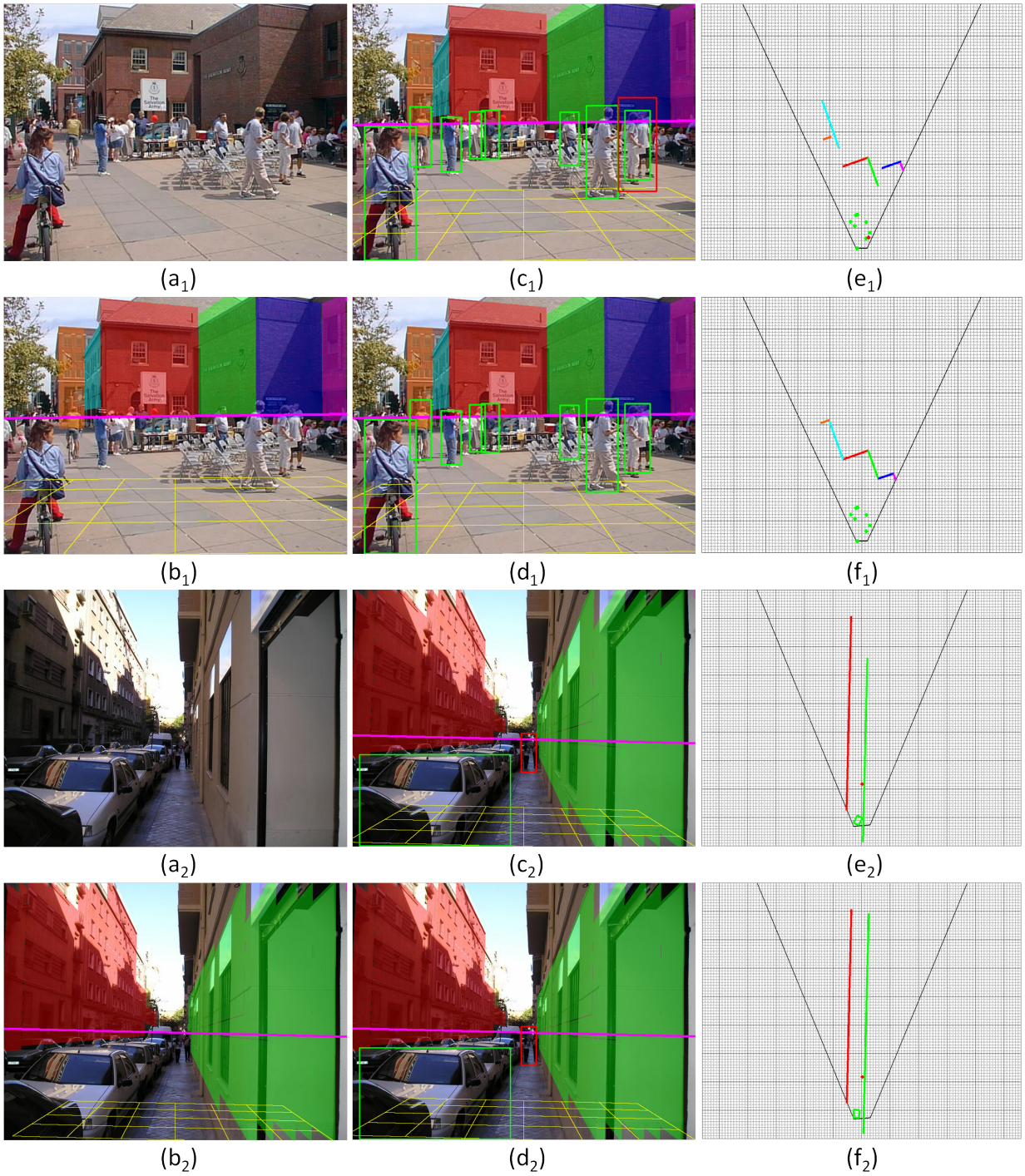

Figure 6.9: Joint 3D geometric reasoning of objects and building facades using our proposed approach. This figure shows two examples, located in the top two rows and bottom two rows, respectively. (a): Original image. (b): Reasoning only using facades. (c): Reasoning using both objects and facades, yet omitting interactions among local entities. (d): Reasoning using both objects and facades with our full system. (e): top-down view for the result in (c). (f): top-down view for the result in (d). Please see text for more details.

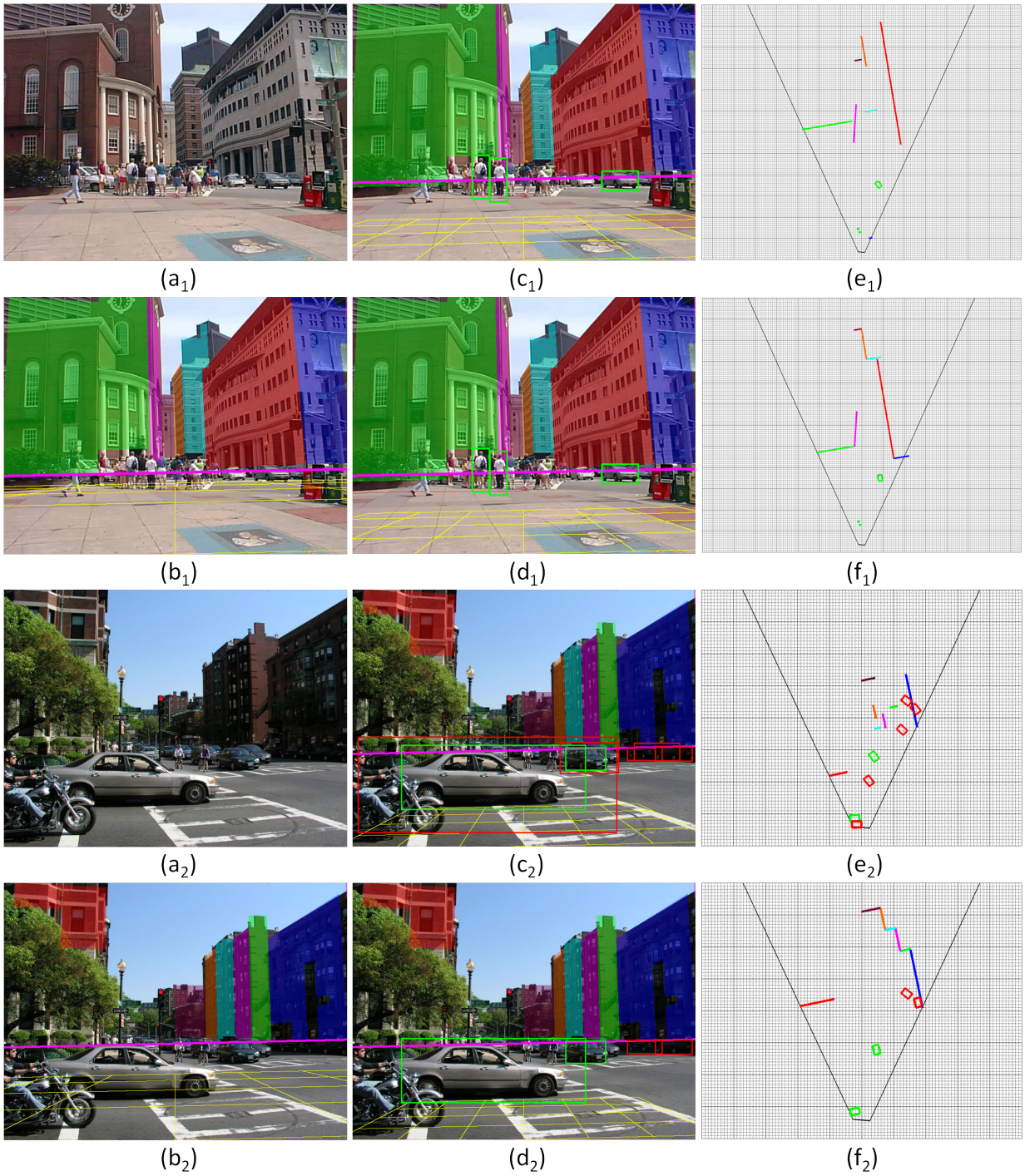

Figure 6.10: Joint 3D geometric reasoning of objects and building facades using our proposed approach. This figure shows two examples, located in the top two rows and bottom two rows, respectively. (a): Original image. (b): Reasoning only using facades. (c): Reasoning using both objects and facades, yet omitting interactions among local entities. (d): Reasoning using both objects and facades with our full system. (e): top-down view for the result in (c). (f): top-down view for the result in (d). Please see text for more details.

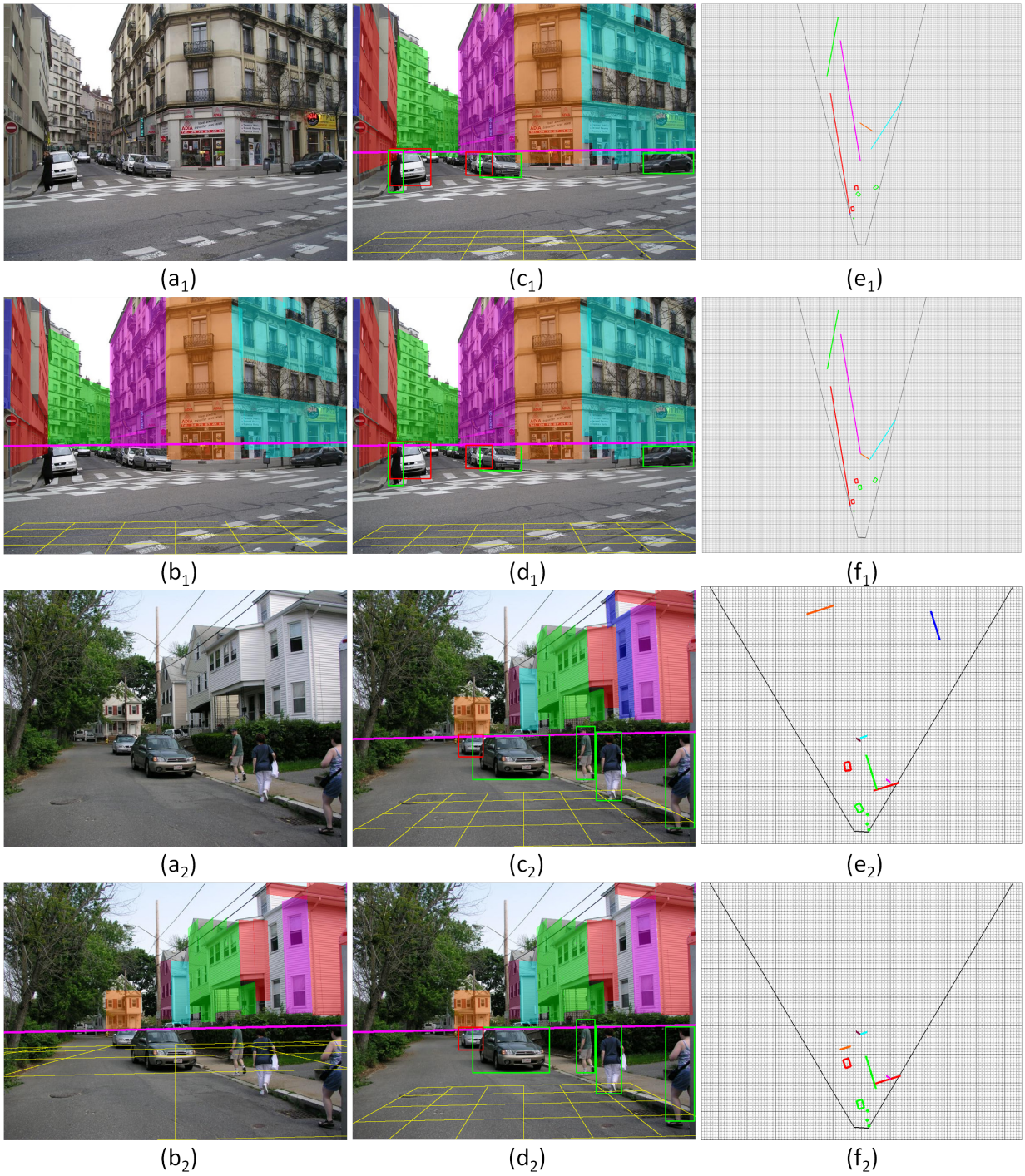

Figure 6.11: Joint 3D geometric reasoning of objects and building facades using our proposed approach. This figure shows two examples, located in the top two rows and bottom two rows, respectively. (a): Original image. (b): Reasoning only using facades. (c): Reasoning using both objects and facades, yet omitting interactions among local entities. (d): Reasoning using both objects and facades with our full system. (e): top-down view for the result in (c). (f): top-down view for the result in (d). Please see text for more details.

they do help improve the estimation of the yaw angles of cars. As is mentioned in Section 6.2.5, we use facade plane orientation to refine the heading of geometrically related cars under the assumption that the two should be consistent in urban street scenes.

For example, in 6.9(e_2) and (f_2), we show the top-down views of the scene layout corresponding to the results in 6.9(c_2) and (d_2), respectively. In the top-down view, colored thick lines, rectangles, and dots represent facade planes, cars and pedestrians, respectively. The color code in the top-down view is the same as in the image view. Black lines mark the viewing boundary, and the coarser grid spacing is 10m.

We can see that the inaccurate heading (estimated by the pose regressor) of the car in 6.9(e_2) is corrected by the orientation of the building facade color-shaded in red. When the regressor-estimated heading of a car deviates too much (over 30 degrees) from the direction of its geometrically related facade plane, it is left as is. An example is shown in 6.8(f_2), where the heading of the closest car is left unchanged after the pairwise geometric constraints are imposed. An error case is shown in 6.10(f_2), where the heading of the farthest car remains erroneous.

Quantitatively, the interactions between facade planes and cars have reduced the yaw estimation error averaged over all correctly detected cars from 17.2 degrees to 6.9 degrees.

Facade Decomposition: To check whether introducing object-facade interaction would improve facade decomposition, we repeat the experiment in Section 5.4.2, except that this time the CRF inference involves both objects and facade planes. For the sake of fair comparison, we compute the global geometric variables the same way as we did in Chapter 5 (*i.e.*, obtaining $\mathbf{n_g}$ and $\mathbf{n_p}$ from vanishing lines and setting $h_p = 1.6m$), and use the same set of candidate facade planes obtained in the experiment in Section 5.4.2. We had hoped that the presence of objects would invalidate more false candidate facade planes and therefore improve surface label prediction. However, the result in this experiment turned out to be exactly the same as the result in Section 5.4.2 – an accuracy of 74.82%.

The reason behind this is the same as why the object-facade interaction fails to improve object

	Gravity direction	Ground plane orientation	Ground plane distance
Vanish Lines	2.60	2.82	N/A
Facade only	2.59	2.37	0.78
Object only	**2.51**	2.37	**0.48**
Joint	**2.51**	**2.35**	**0.48**
Hoiem	3.54	3.54	0.56

Figure 6.12: Comparison of the estimation of global 3D geometries. The numbers in the table are estimation errors, where gravity direction and ground plane orientation errors are measured in degrees, and ground plane distance error is measured in meters. "Vanish Lines": vanishing-line based algorithm described in Section 3.5. "Facade only": our 3D reasoning system except that we only use facades. "Object only": our 3D reasoning system except that we only use objects. "Joint": our 3D reasoning system using both objects and facades. "Hoiem": the algorithm in [34].

detection: due to the high flexibility of facade depth, interacting objects and facade planes typically do not invalidate each other in the images we encounter; rather, facade planes are pushed away by confident objects. Since this pushing away does not change facade decomposition in 2D image, it does not lead to a better surface layout classification.

Camera Viewpoint Estimation: We also evaluate and compare the accuracy of camera viewpoint estimation using different approaches and variants on the LabelMe dataset [34]. Here, camera viewpoint is equivalent to global 3D geometries – gravity direction, ground plane orientation, and ground plane distance. Among the 100 test images that have ground-truth horizons, 65 of them contain building facades and are thus used in this experiment.

We list the results in Figure 6.12, where the numbers are estimation errors. Gravity direction and ground plane orientation errors are measured in degree, while ground plane distance error is measured in meter. We can see that our full system (fourth row) achieves the smallest error in estimating all the three global geometric variables. Yet reasoning over objects only (third row) is virtually as good. Reasoning over facades only (second row) achieves a good estimation of horizons (*i.e.* gravity direction and ground plane orientation), yet results in a significantly larger error in ground plane distance. This phenomenon implies objects are more informative in terms

of estimating length-related quantities such as depth and ground plane distance, while facades alone are much more ambiguous in length-related quantities, yet are sufficient to give a good estimate of orientation-related quantities such as gravity and ground horizons.

Examples in Figures 6.7– 6.11 also confirm this argument. In each example, the magenta line is the gravity horizon, and the yellow grid indicates the ground plane, where the grid spacing is 1m. When ground plane orientation differs from gravity direction, a yellow ground horizon is displayed as well. The image (b) in each example shows the result of reasoning using facades only. Note that this is different than the experiments in Chapter 5 where global 3D geometries are pre-computed or assumed. In this experiment, global 3D geometries are optimized using the generalized RANSAC algorithm. Comparing (b) and (d), we can see that while the estimated horizons are usually the same, estimated ground plane distance could be very different, such as the first example in Figure 6.10.

From Figure 6.12, we can also see that whether or not we leave out objects or facades, our 3D reasoning system always achieves a lower error than estimating global 3D geometries from vanishing lines without resorting to individual objects or facade planes. Our approach also outperforms Hoiem's algorithm in [34].

Facade Depth Estimation: While object-facade interaction does not improve facade decomposition, introducing objects into facade reasoning could help improve facade depth estimation by pushing facade planes to a non-conflicting depth and/or obtaining a better estimate of the ground plane distance. As the LabelMe dataset [34] does not provide ground truth in depth while the Make3D dataset [48] contains few, if any pedestrians or cars, we are not able to obtain a quantitative evaluation of the improvement in facade depth estimation due to the introduction of objects. However, plenty of qualitative examples substantiate this argument. Figure 6.13 shows three typical examples. The first column displays the distinctive facade planes in the image. The second and third columns show the 3D facade layout after reasoning over facades only and reasoning over both objects and facades, respectively. In the first example, the presence of objects

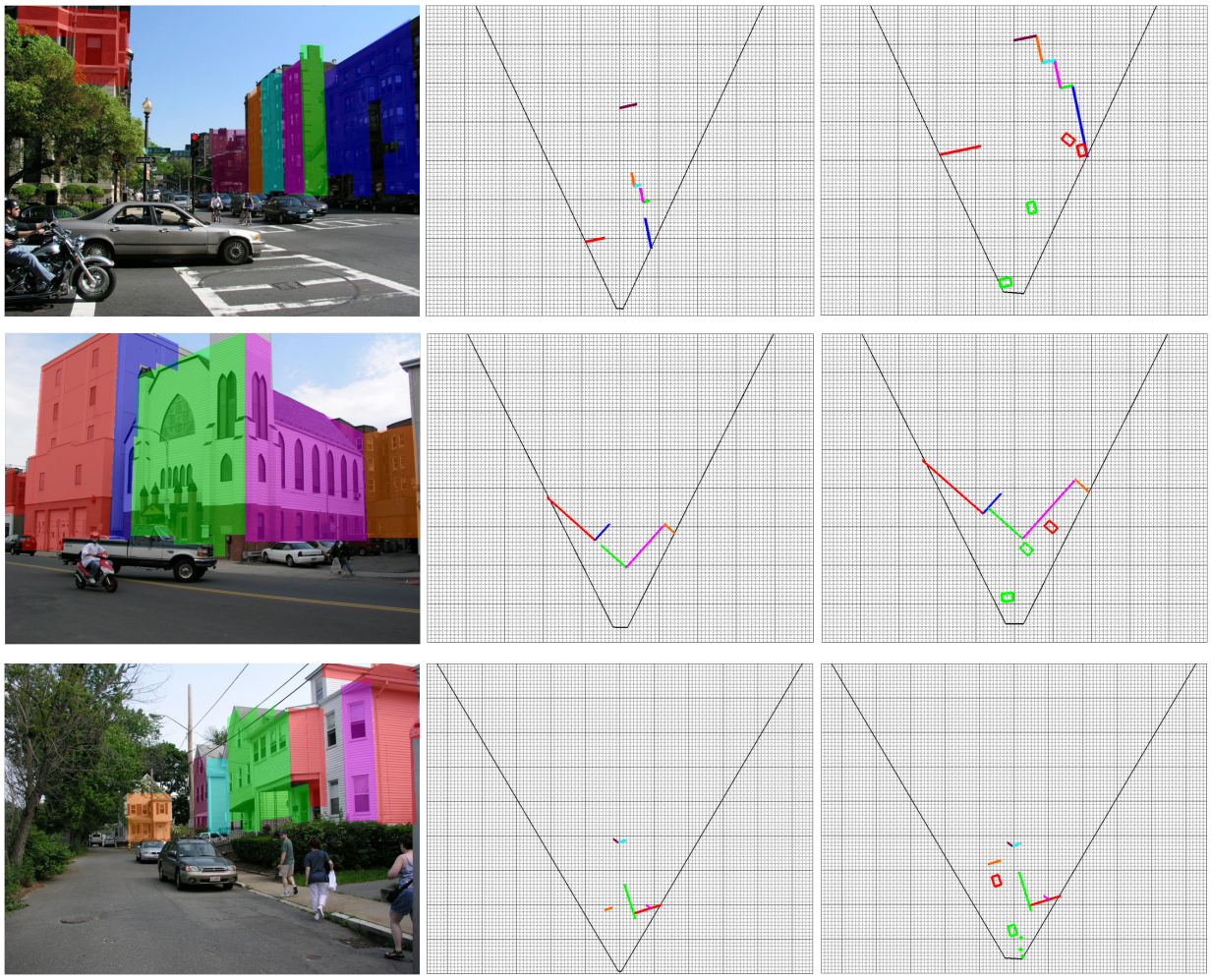

Figure 6.13: Objects help improve facade depth estimation. In each row, the left image shows distinctive facade planes in the image, the center image shows the 3D facade layout after reasoning over facades only, and the right image shows the 3D facade layout after reasoning over both objects and facades.

improves the estimate of ground plane distance (note the difference in the viewing boundary due to the change in ground plane distance), through which facade planes are placed to a more reasonable depth. In the second example, facade planes are pushed by objects to a more reasonable depth. In the third example, both factors contribute to the orange facade being placed behind the farther car.

6.5 Conclusion

In this chapter, we describe our full 3D geometric reasoning system involving global 3D geometries, objects, and building facades. We model two types of global 3D geometric constraints and three types of local 3D geometric constraints. Those constraints are encoded into unary and pairwise potentials in a unified objective function. Optimization of the objective function is made tractable through the generalized RANSAC algorithm, the quadrilateral-based sampling algorithm, and simplified local optimization. Utilizing the geometric interactions among all entities benefits individual tasks such as object detection, viewpoint recovery, object pose estimation, and facade depth estimation. Moreover, it enables our method to generate a richer 3D interpretation of the scene that is geometrically coherent.

As with most computer vision algorithms, our method still makes many mistakes (*e.g.*, erroneous facade decomposition in the second examples of Figure 6.10 and 6.11). However, whatever mistake our method makes, the output is always geometrically coherent in 3D. We believe making geometrically coherent mistakes is the first step towards human-level scene understanding.

Chapter 7

Future Directions

We have proposed in this dissertation a 3D geometric reasoning system that generates a coherent interpretation of the 3D structure behind a single 2D image of urban scenes. While this system has produced promising results, our work is still one of the early attempts in the grand endeavor of striving towards the extremely challenging goal of achieving human-level holistic scene understanding capabilities. Some potential ideas for future work are discussed below.

7.1 Semantic Parsing of Building Facades

As we have mentioned in Section 6.4, building facades in our reasoning system have high flexibility in depth. Such a high flexibility results from two factors. Firstly, the reasonable facade height has a large range – from 5m to 30m with a standard deviation of 10m on the higher end. Therefore, the facade plane could move as close to the camera as when the facade height starts to become too small, and move as far away from the camera as when the facade height starts to become too large. Secondly, the ground contact line of a facade plane is often occluded. Therefore, the facade plane could 1) move as close to the camera as when the ground contact line would start to cut across the ground semantic region or the facade plane would start to occlude a foreground object, and 2) move as far away from the camera as when the ground contact line would start to

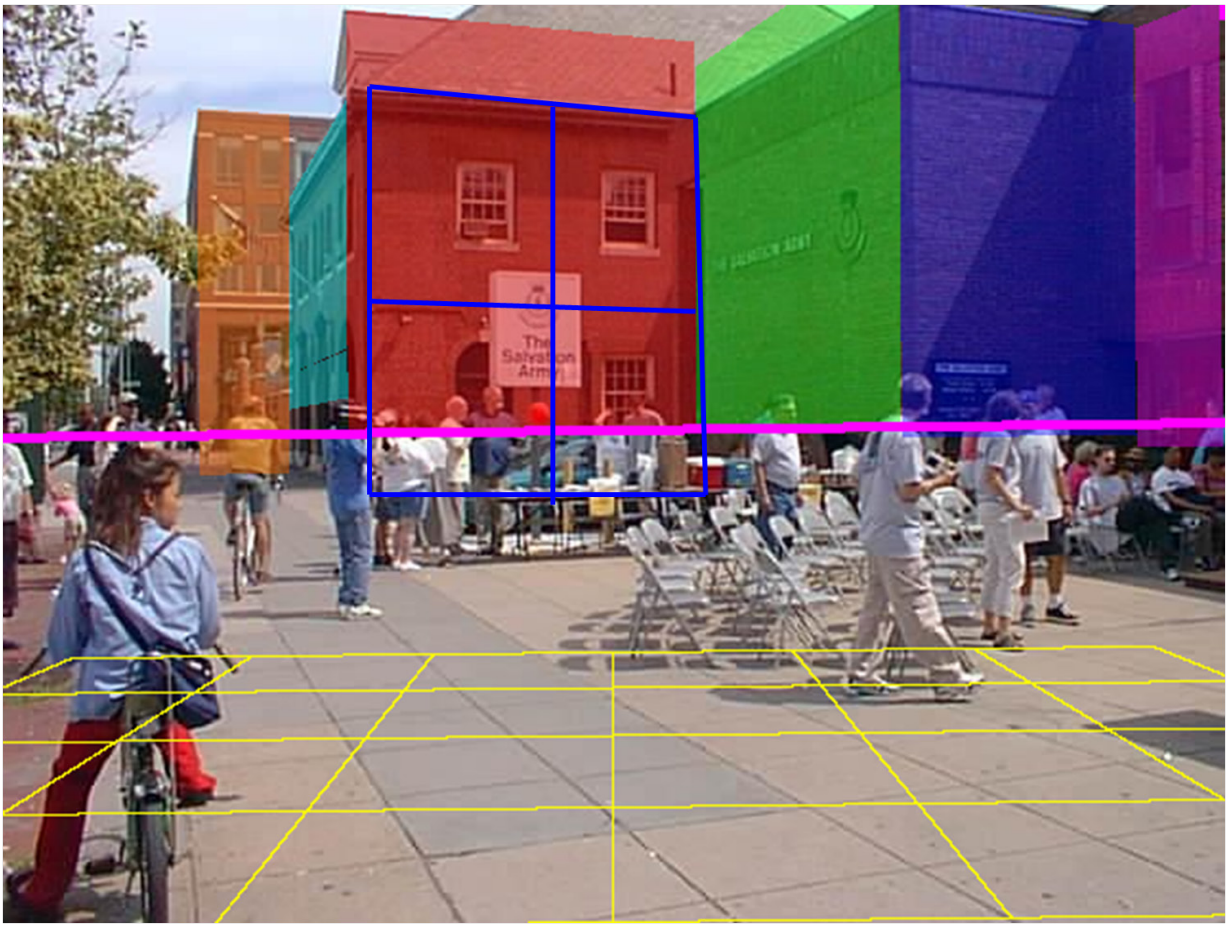

Figure 7.1: Parsing a facade plane into multiple floors reduces the uncertainty about its height and the location of its ground contact line.

cut across the building semantic region. This high flexibility in depth results in a significantly reduced capability of facade planes to identify invalid objects and effectively estimate ground plane distance.

One way to address this difficulty is to perform semantic parsing on building facades. More specifically, by analyzing the patterns within a facade plane, we could parse it into multiple floors with the help of windows and/or doors, and obtain the total number of floors. This way, we could obtain a much narrower estimate of the facade height by counting the total number of floors, and thus remove the first factor contributing to the high flexibility in depth. In addition, parsing a facade plane into floors would significantly reduce the uncertainty in the location of the ground

contact line of the facade plane. For example, once we are able to parse the red facade plane in Figure 7.1 into two floors, we could immediate tell that the ground contact line of the red facade plane is highly likely to be located at the bottom blue line. Together with the prior knowledge about the typical floor height and the location of the ground horizon, we could directly obtain an estimate of the ground plane distance using the ground contact line. For example, given the ground horizon (magenta line) and ground contact line (bottom blue line) in Figure 7.1, we could immediate tell that the ground plane distance is about one third of the floor height. Therefore, semantic parsing on building facades would both enhance the discriminative capability of facade planes on nearby objects, and enable facade planes to contribute more to the estimation of ground plane distance.

While several algorithms exist for parsing building facades [29, 55, 56, 64, 68, 75], doing so in a robust way is still challenging. Irregularities, occlusions, and perspective distortions could all significantly undermine the parsing accuracy. Yet once the robustness of facade parsing is improved, it could greatly benefit 3D geometric reasoning.

7.2 Handling Houses

While large buildings have highly regular facade planes from which vanishing directions are easier to obtain, smaller houses, especially those with slanted roofs, are much harder to reliably extract valid vanishing directions. An example is shown in Figure 7.2(c), where the house in the center-right of the image is mistakenly interpreted by our system as a single facade plane (in green). The root cause of this mistake is that the ranted roof of the house makes the left-facing facade lack line segments that clearly belong to the group of green vanishing lines in Figure 7.2(b). Consequently, the left-facing facade is dominated by line segments mistakenly classified into the blue vanishing group.

To address this problem, a potential method is to treat houses as a sort of objects with greater intra-category variations. A set of house detectors could be learned, each corresponding to a

Figure 7.2: A failure case of our geometric reasoning system. (a) Original image. (b) Vanishing lines and horizon. (c) Output of our system. (d) Top-down view of the output of our system. The house in the center-right is mistakenly regarded as a single facade plane.

Figure 7.3: A natural scene that lacks vanishing lines and well-defined objects.

certain view of house. The training examples of each detector also come with ground-truth 3D structures of houses. When a house is detected in a test image, the average 3D structure corresponding to the detector could help disambiguate the vanishing directions of line segments located in the house region. In the example of Figure 7.2, if the average 3D structure indicates the left half of the house region should be facing left, then many horizontal blue line segments in that region – which are equally likely to belong to the green vanishing group – could be relabeled as green.

7.3 From Urban Scenes to Natural Scenes

Compared to images of urban scenes, recovering the 3D scene structure from images of natural scenes is even more challenging. For example, our reasoning system will not work at all on the image shown in Figure 7.3. This is mostly due to the lack of regularities of man-made structures such as buildings and roads. As a result, vanishing lines are no longer informative in predicting global 3D geometries and surface orientations. Also, most natural scenes seldom

contain well-defined objects like cars or pedestrians, making object-based viewpoint recovery extremely difficult. In addition, the visual cues in natural scenes are even subtler and harder to model. For example, how do we as humans know that the lake in Figure 7.3 has a lower elevation than the meadow? How do we get an idea of how steep the mountain slopes are?

To handle natural scenes, a reasoning system may need to integrate information provided by multiple cues such as shape from shade, texture gradient, and common geometric properties of different semantic regions. With such cues, "piecewise"3D structures of local image regions could be roughly estimated, and the "pieces"of 3D structures could be stitched together following local and global geometric constraints. To obtain a good initial estimate of the overall 3D structure, we could use an approach similar to the label transfer algorithm proposed in [49], except that 3D structures instead of semantic labels are transferred.

Appendix A

Complete Results on Surface Layout Classification

In this appendix, we display the surface layout classification results of our approach described in Chapter 5 and the state-of-the-art algorithms on all 55 test images. Figures A.1 through A.11 show the comparisons between our approach and the algorithm in [35], and Figures A.12 through A.20 show the comparisons between our approach and the algorithm in [25].[1] In those figures, each row is a test example, and all the test examples are sorted in descending order according to the accuracy gain of our approach over the competing algorithm. In each row, the three images from the left show the ground-truth surface orientation labels, the result of the competing algorithm, and our result.

[1] We were not able to acquire the results of [25] on 10 test images. Therefore, only 45 images are shown.

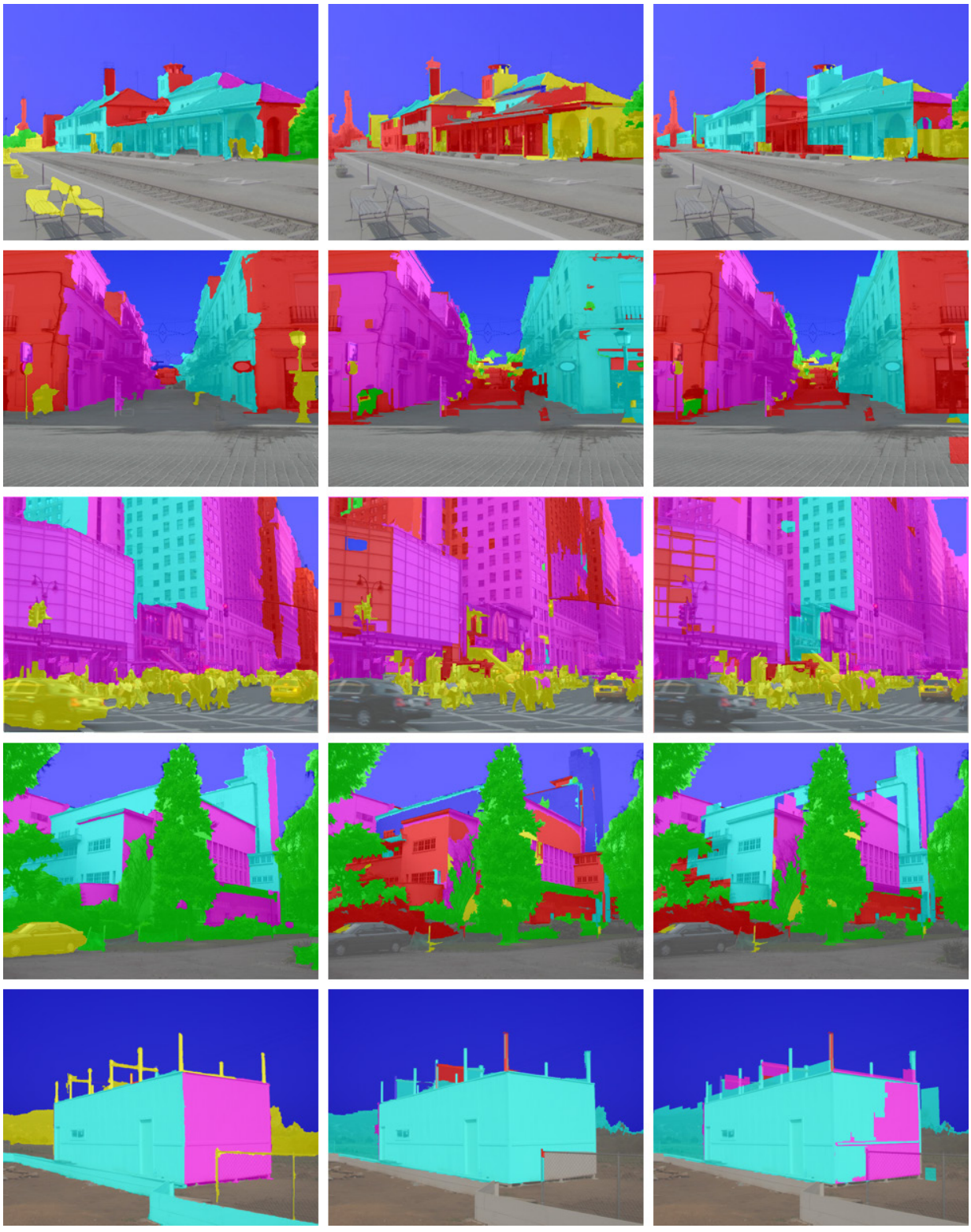

Figure A.1: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 1-5.

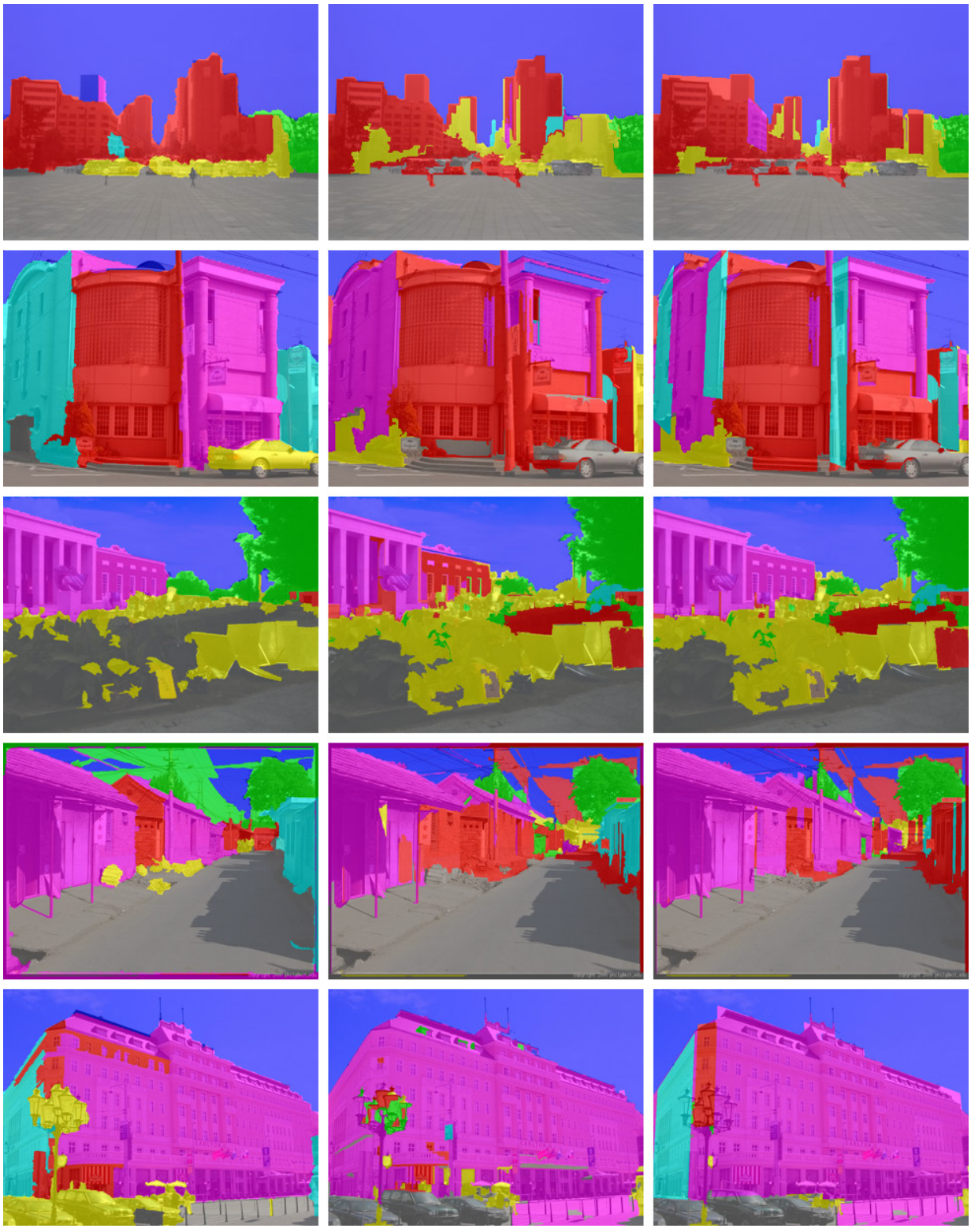

Figure A.2: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 6-10.

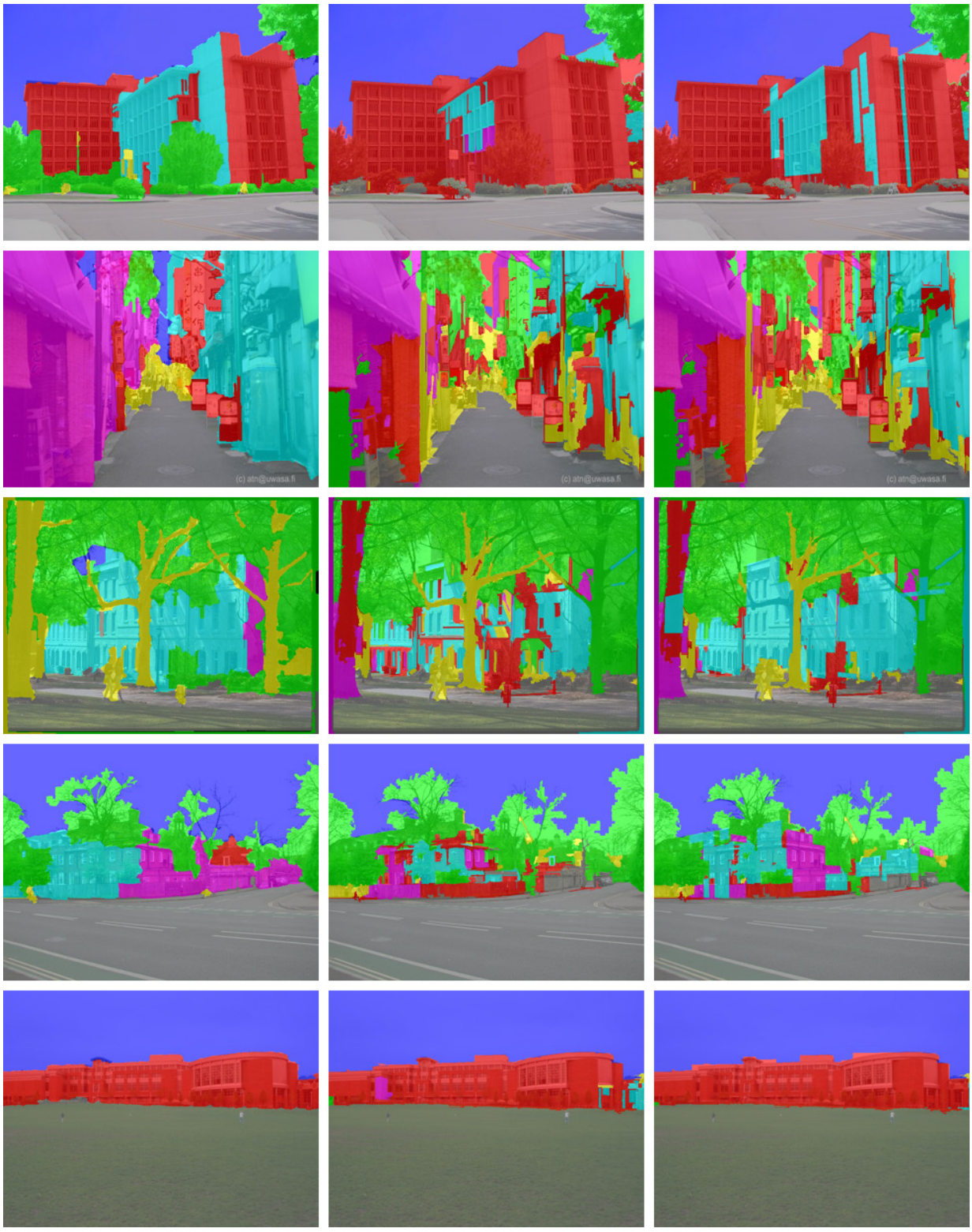

Figure A.3: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 11-15.

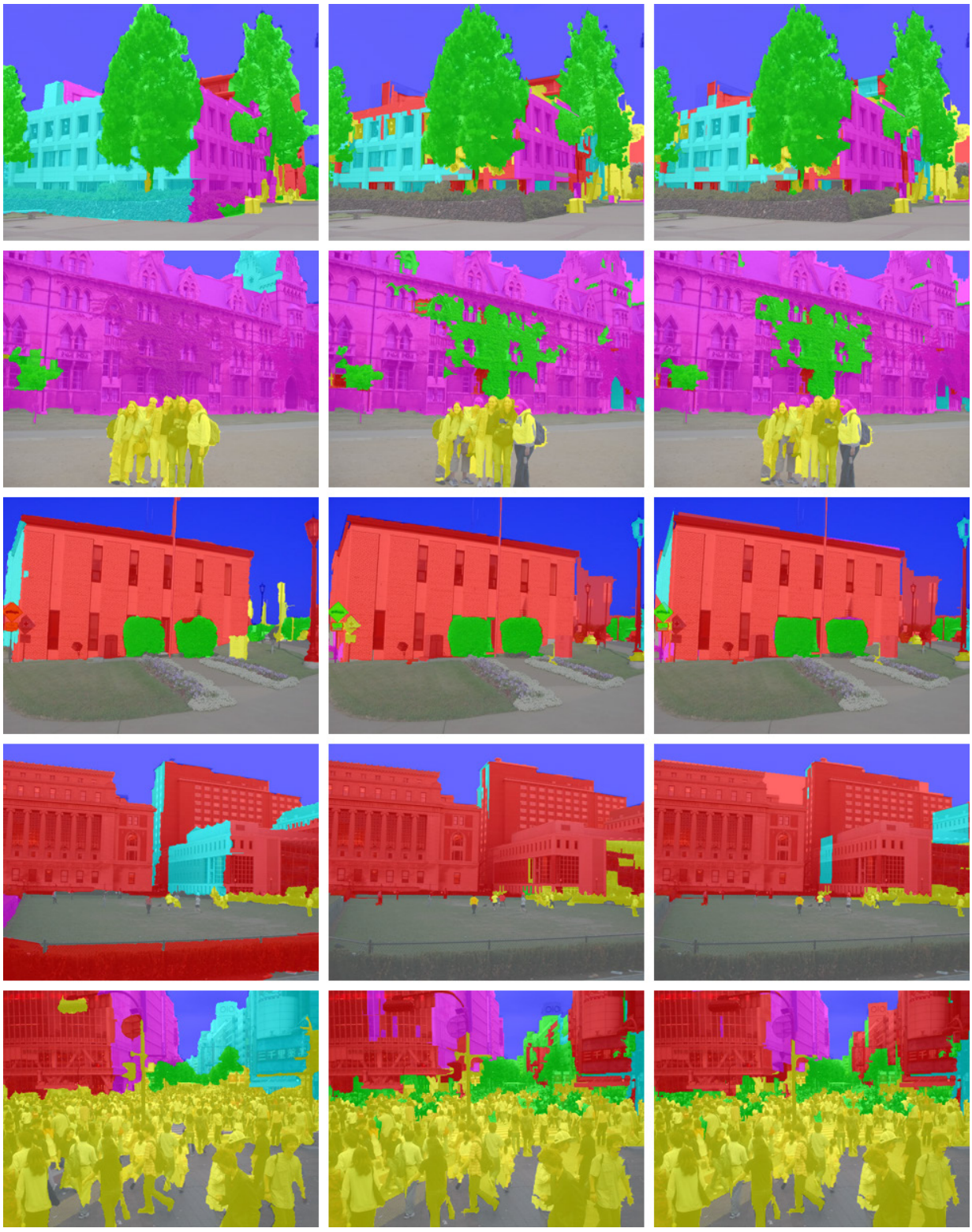

Figure A.4: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 16-20.

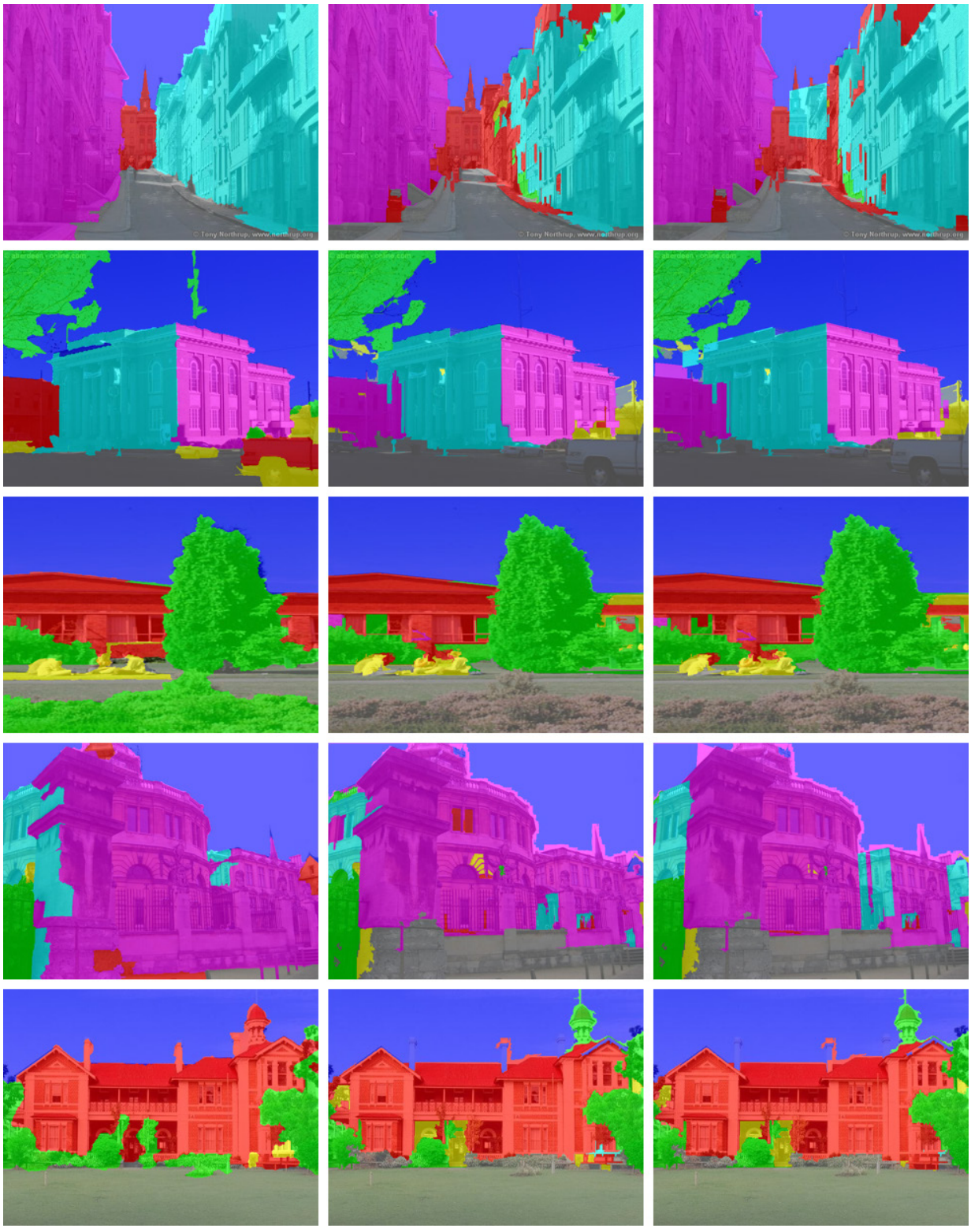

Figure A.5: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 21-25.

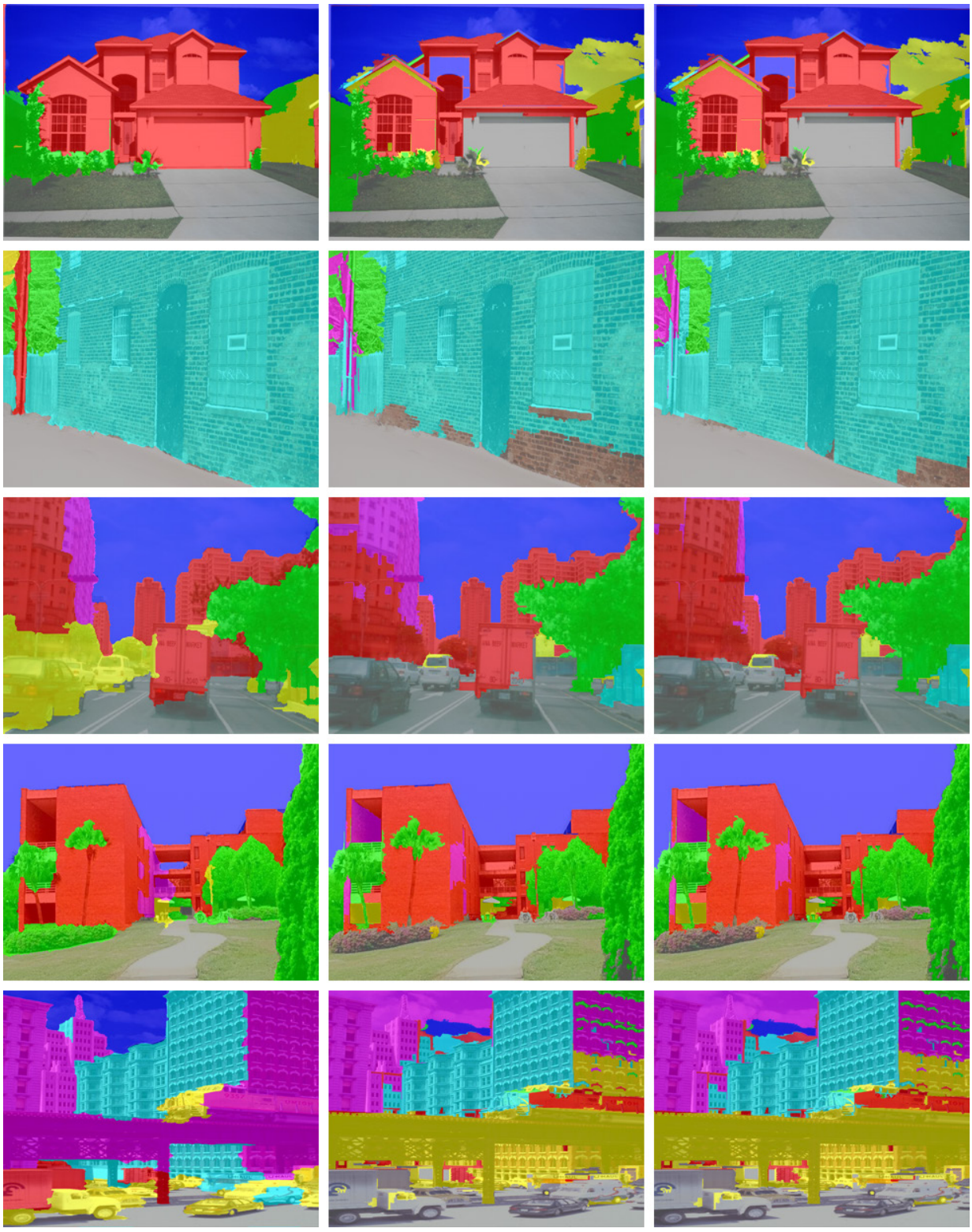

Figure A.6: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 26-30.

Figure A.7: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 31-35.

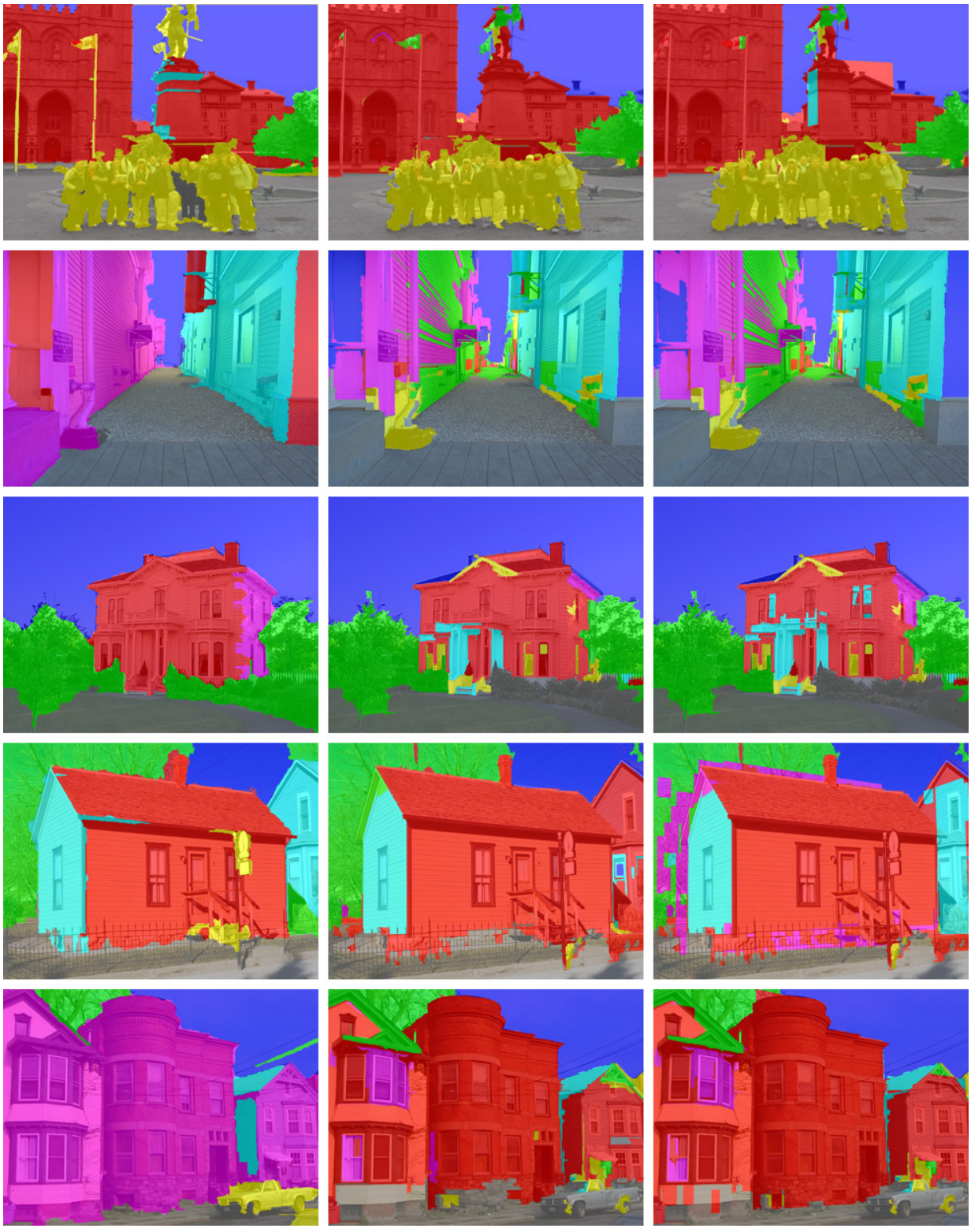

Figure A.8: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 36-40.

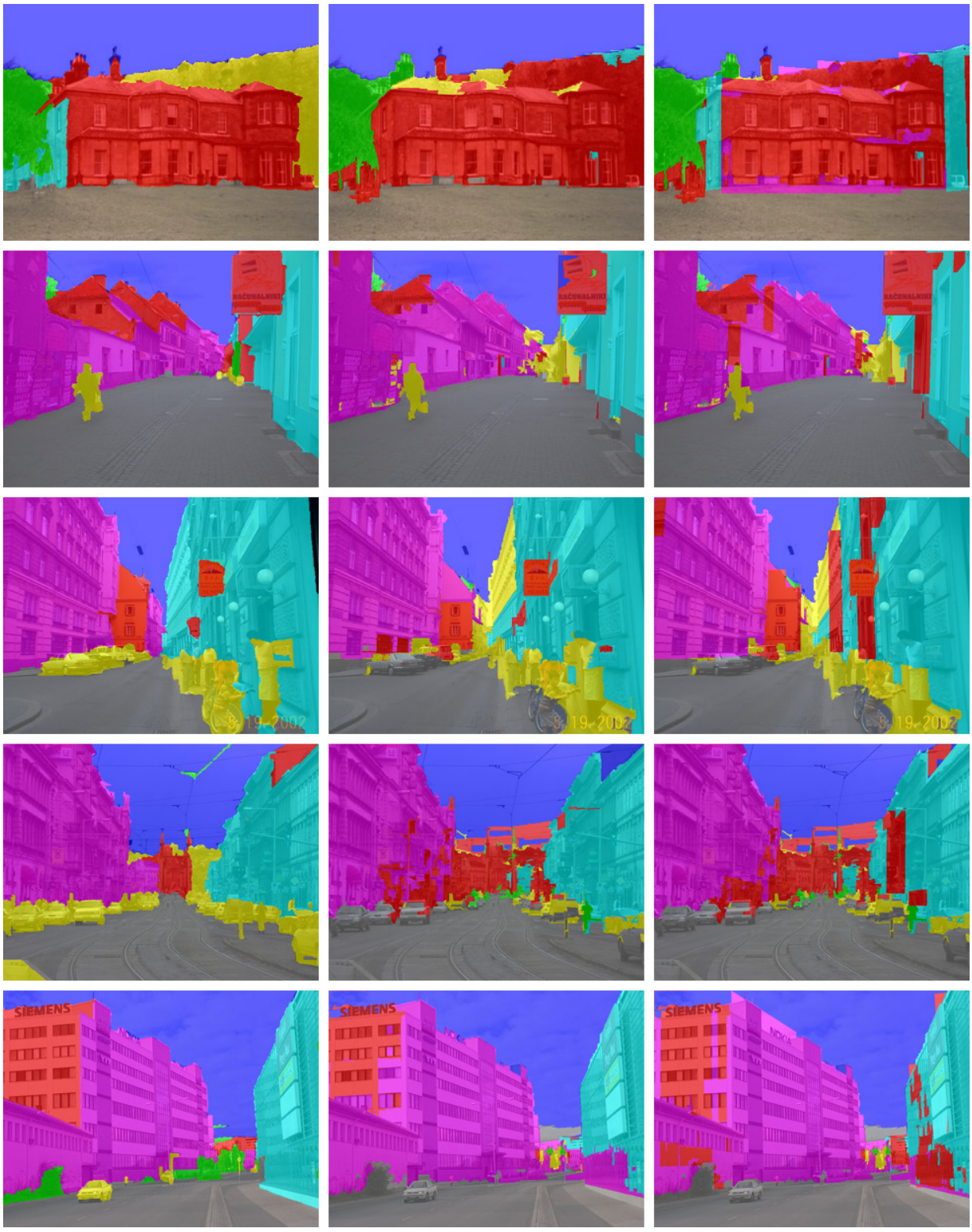

Figure A.9: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 41-45.

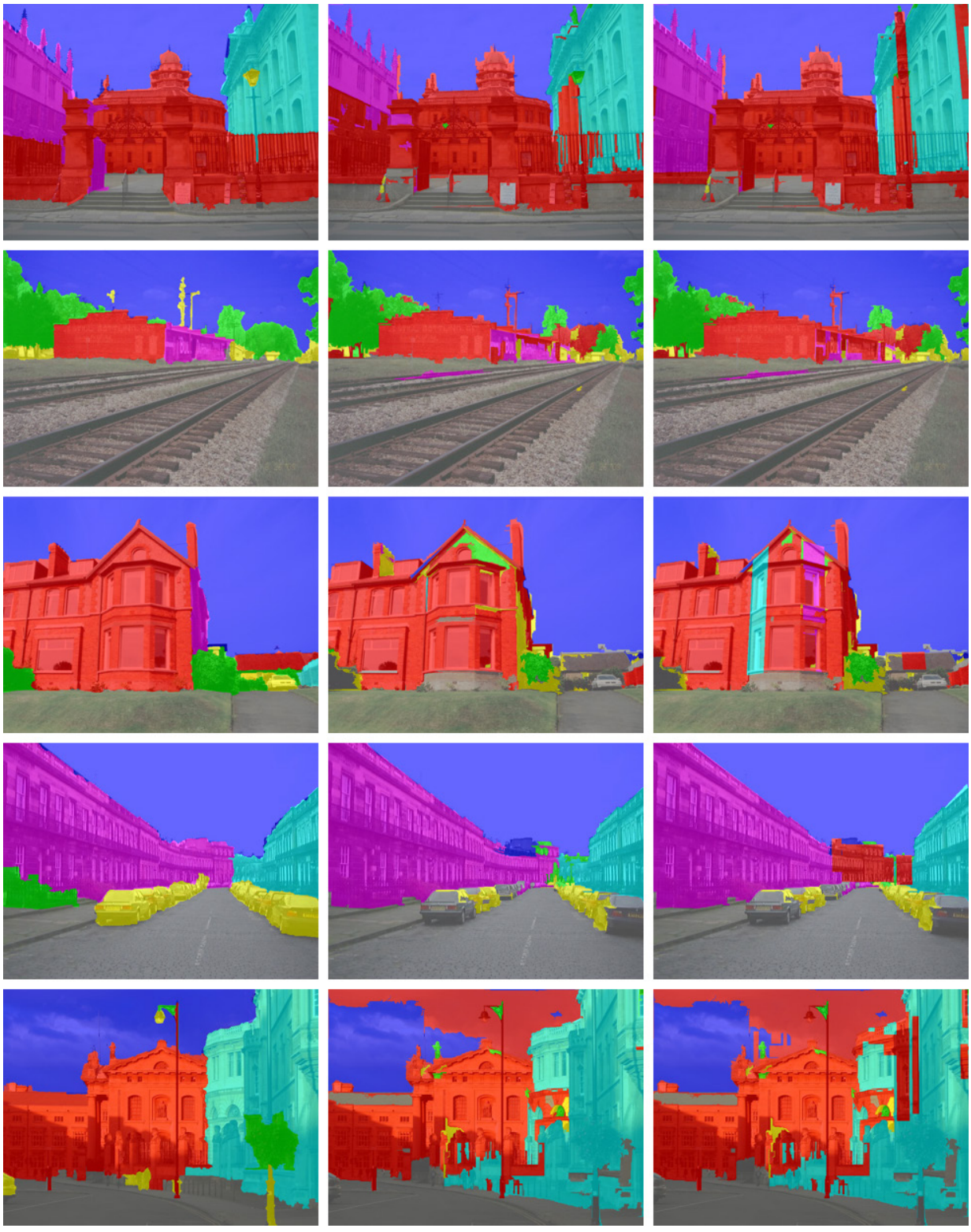

Figure A.10: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 46-50.

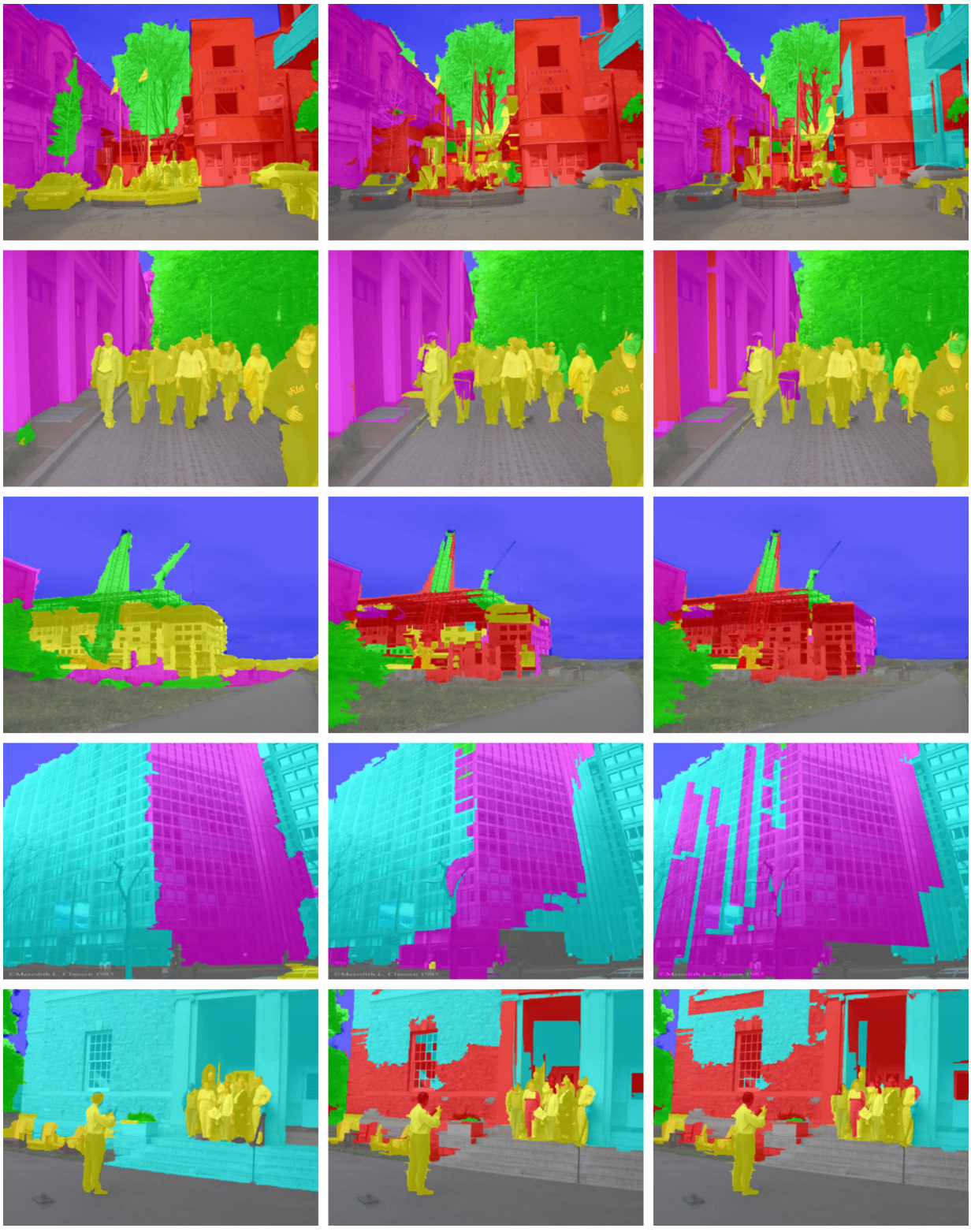

Figure A.11: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [35]. Images 51-55.

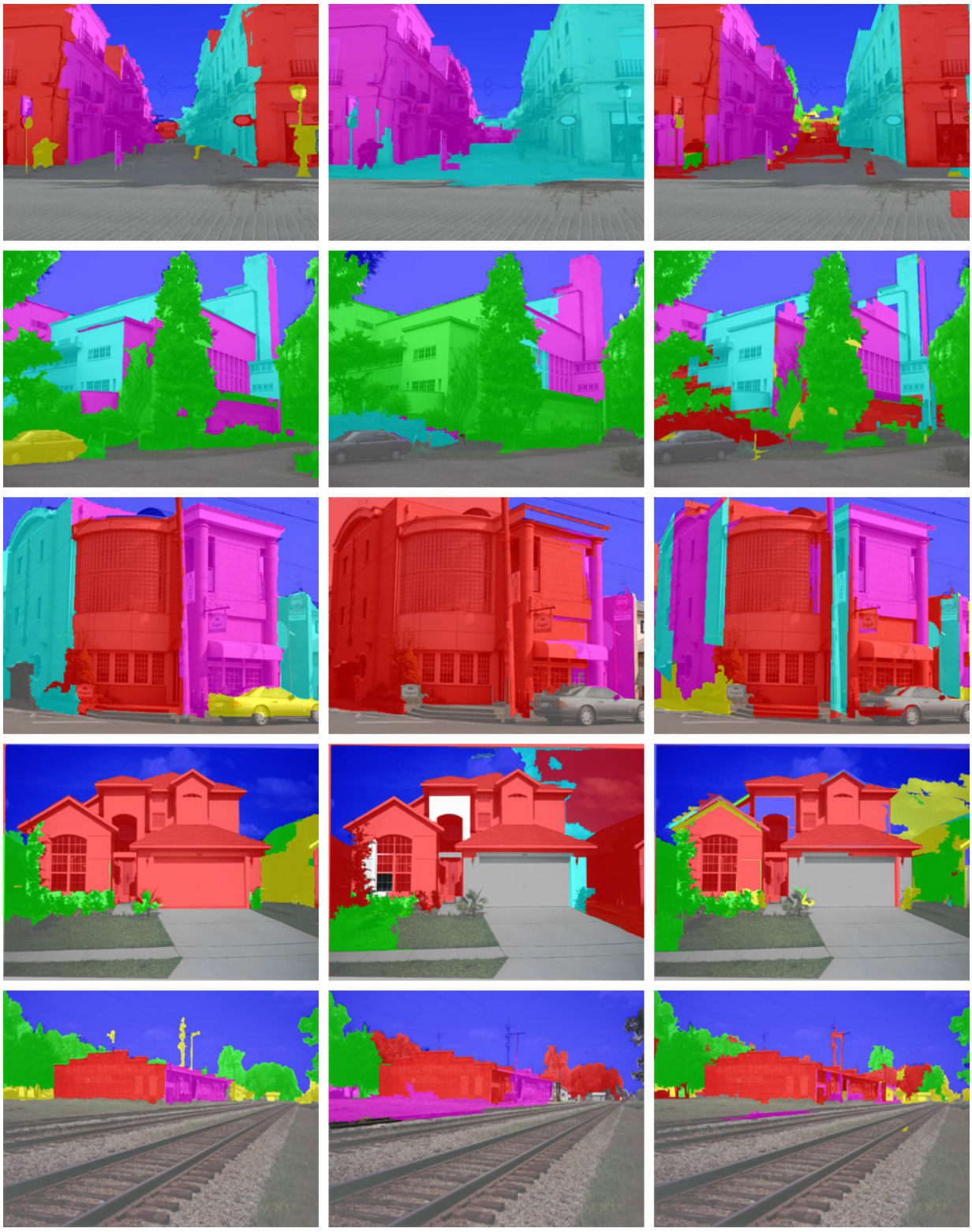

Figure A.12: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 1-5.

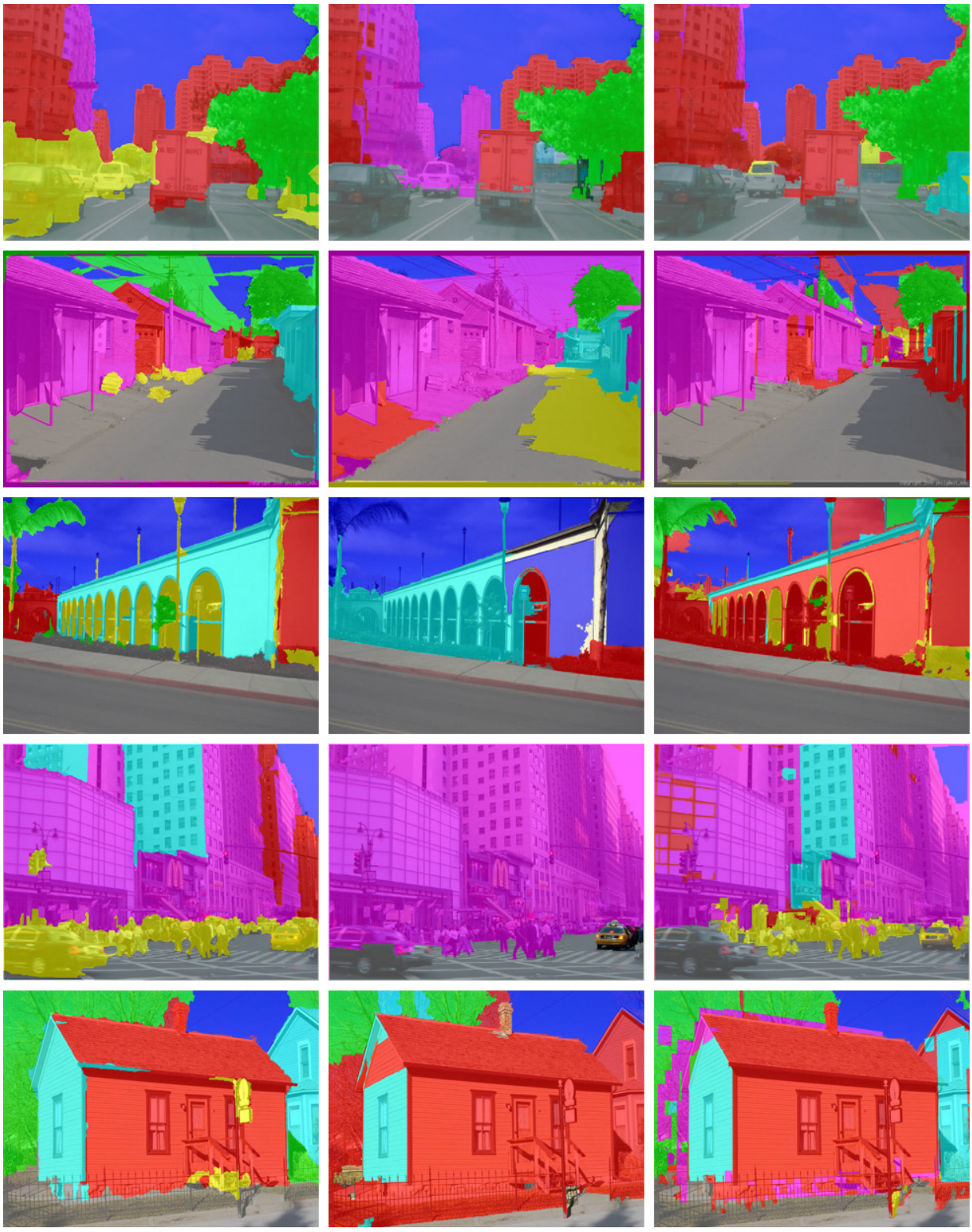

Figure A.13: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 6-10.

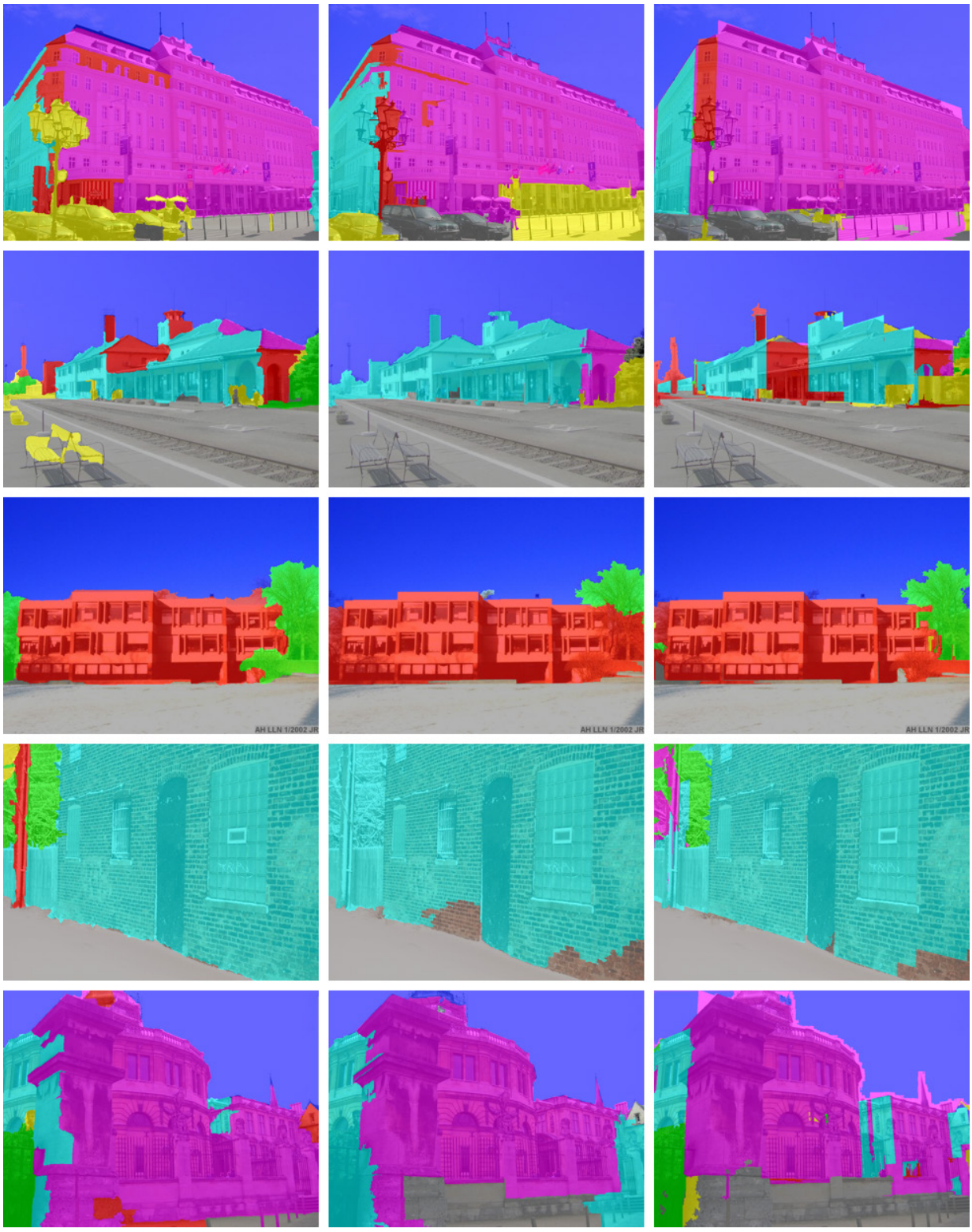

Figure A.14: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 11-15.

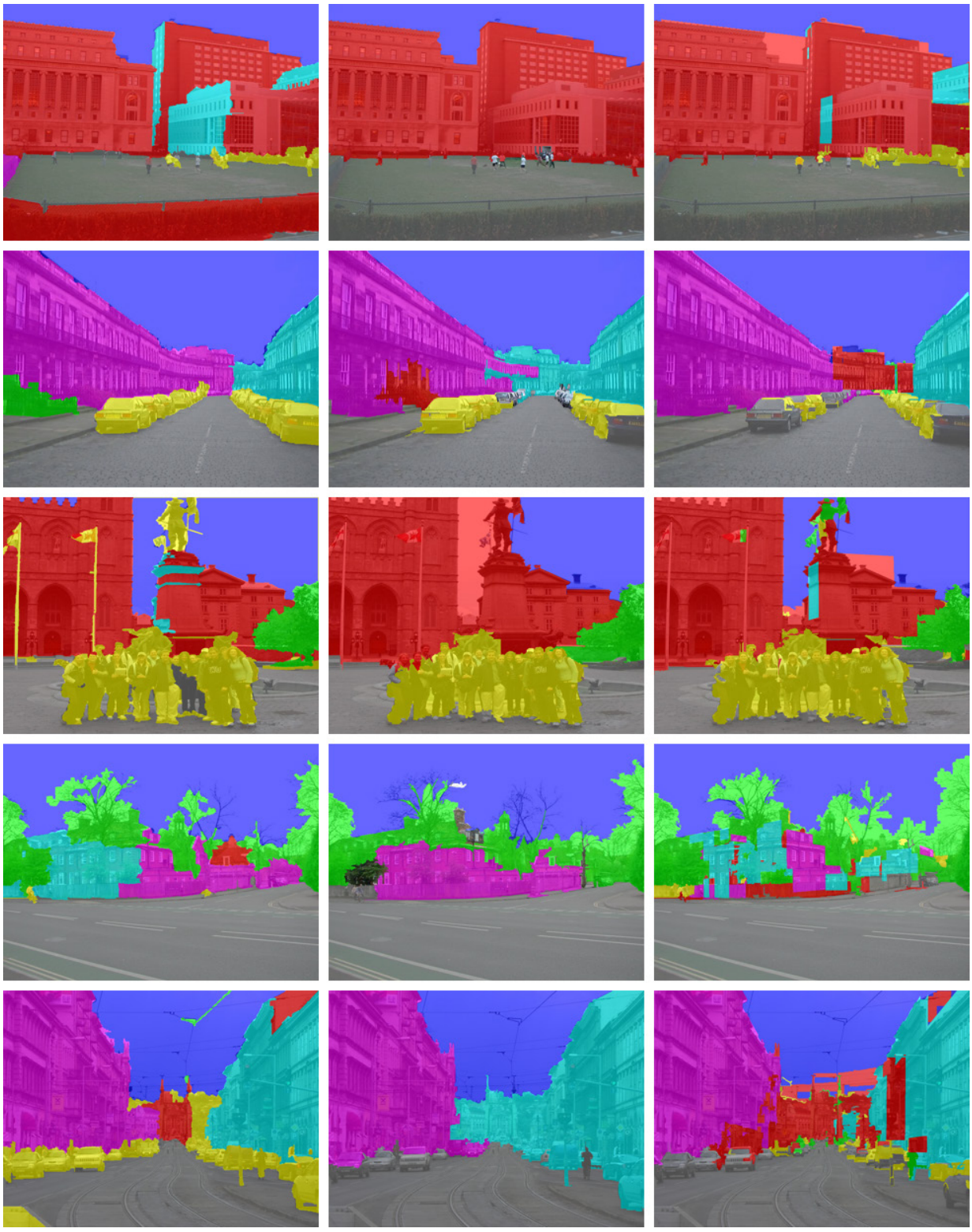

Figure A.15: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 16-20.

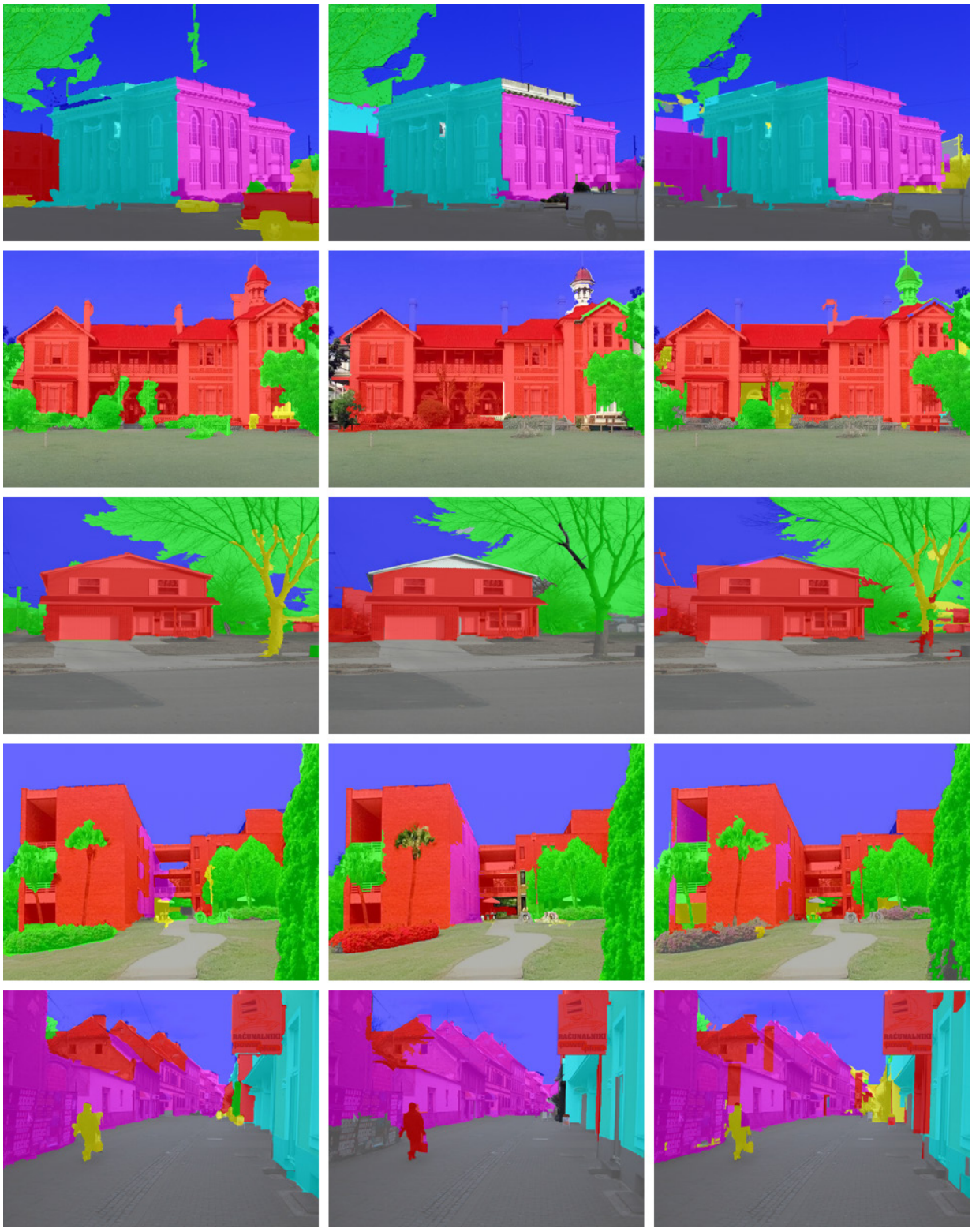

Figure A.16: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 21-25.

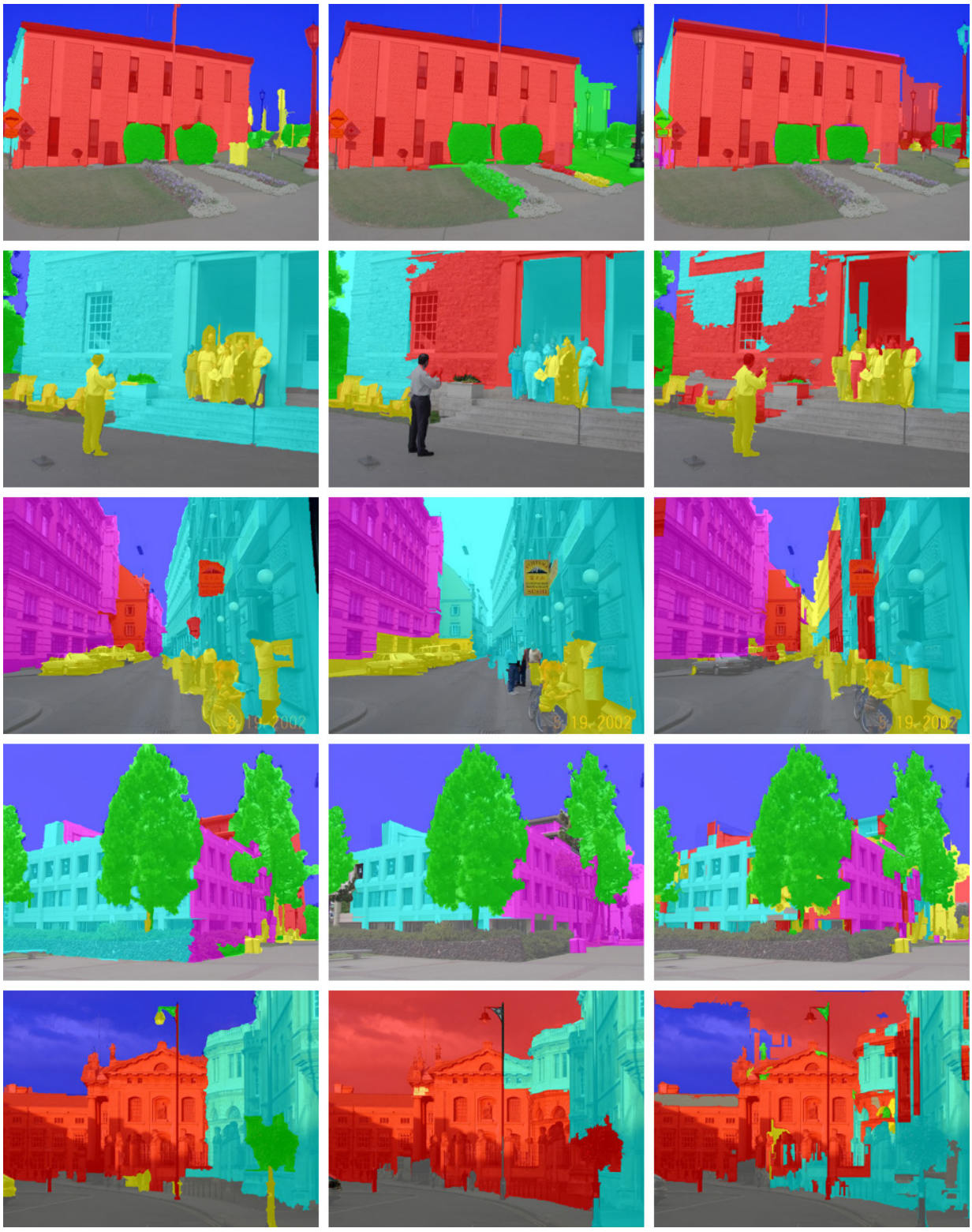

Figure A.17: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 26-30.

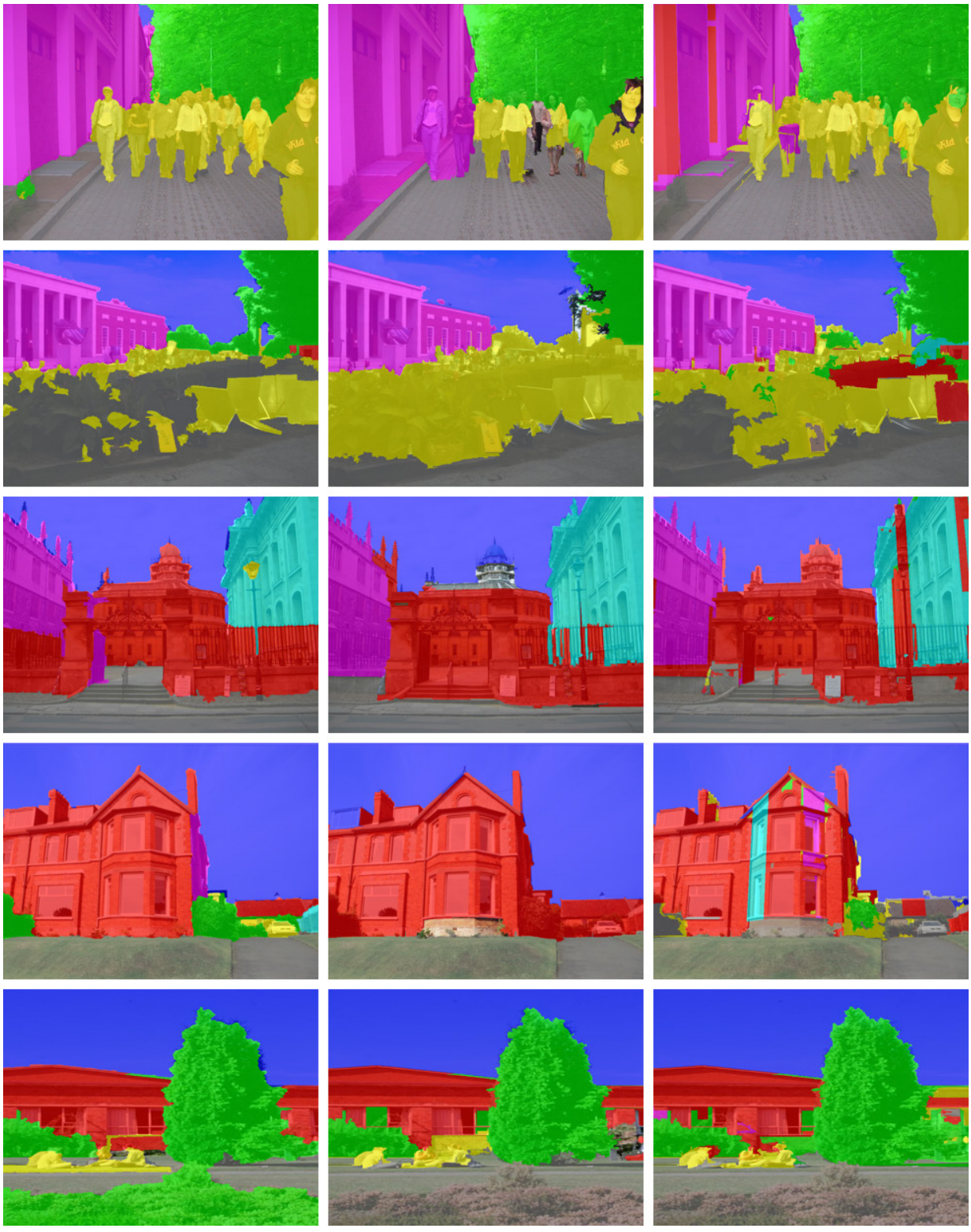

Figure A.18: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 31-35.

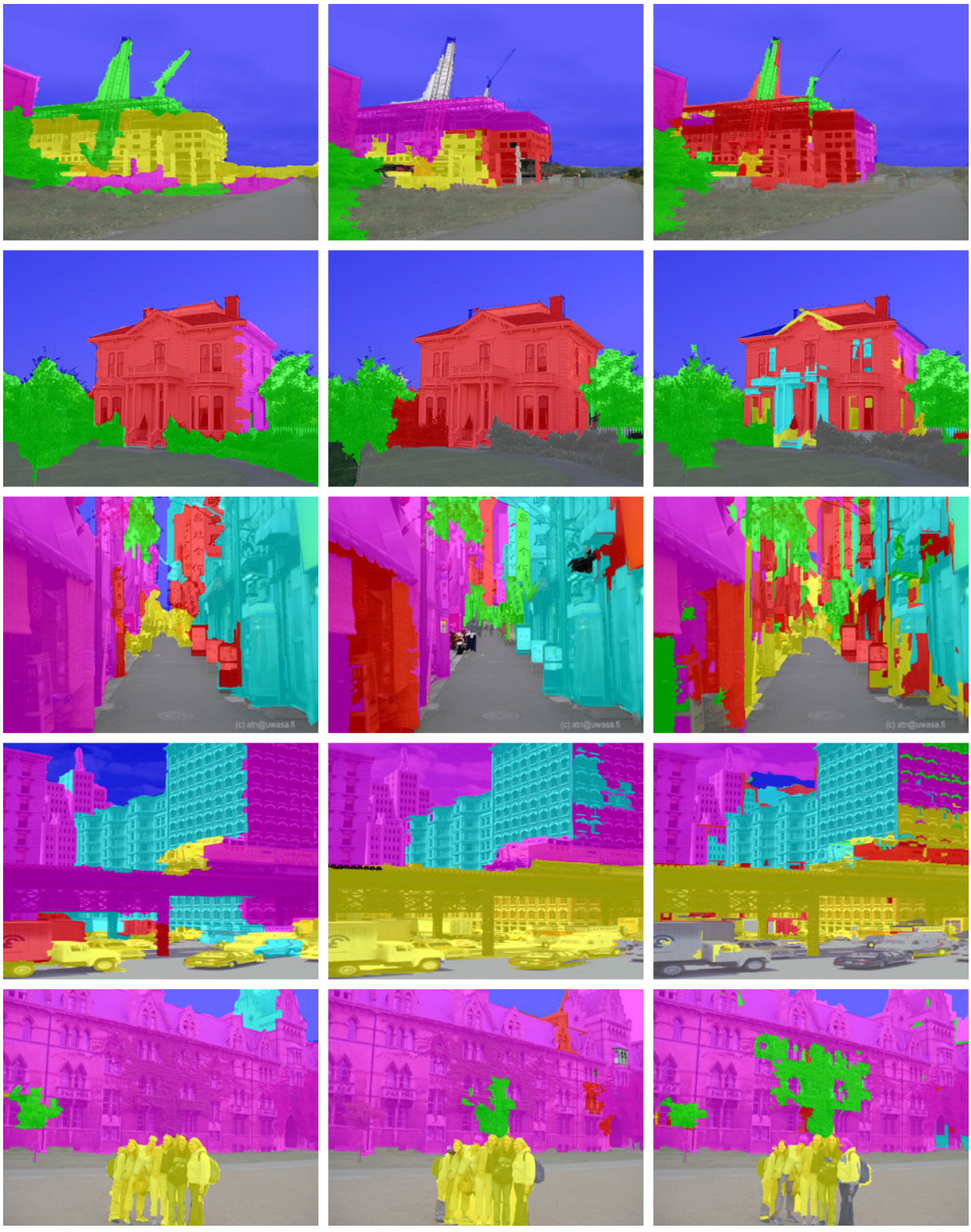

Figure A.19: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 36-40.

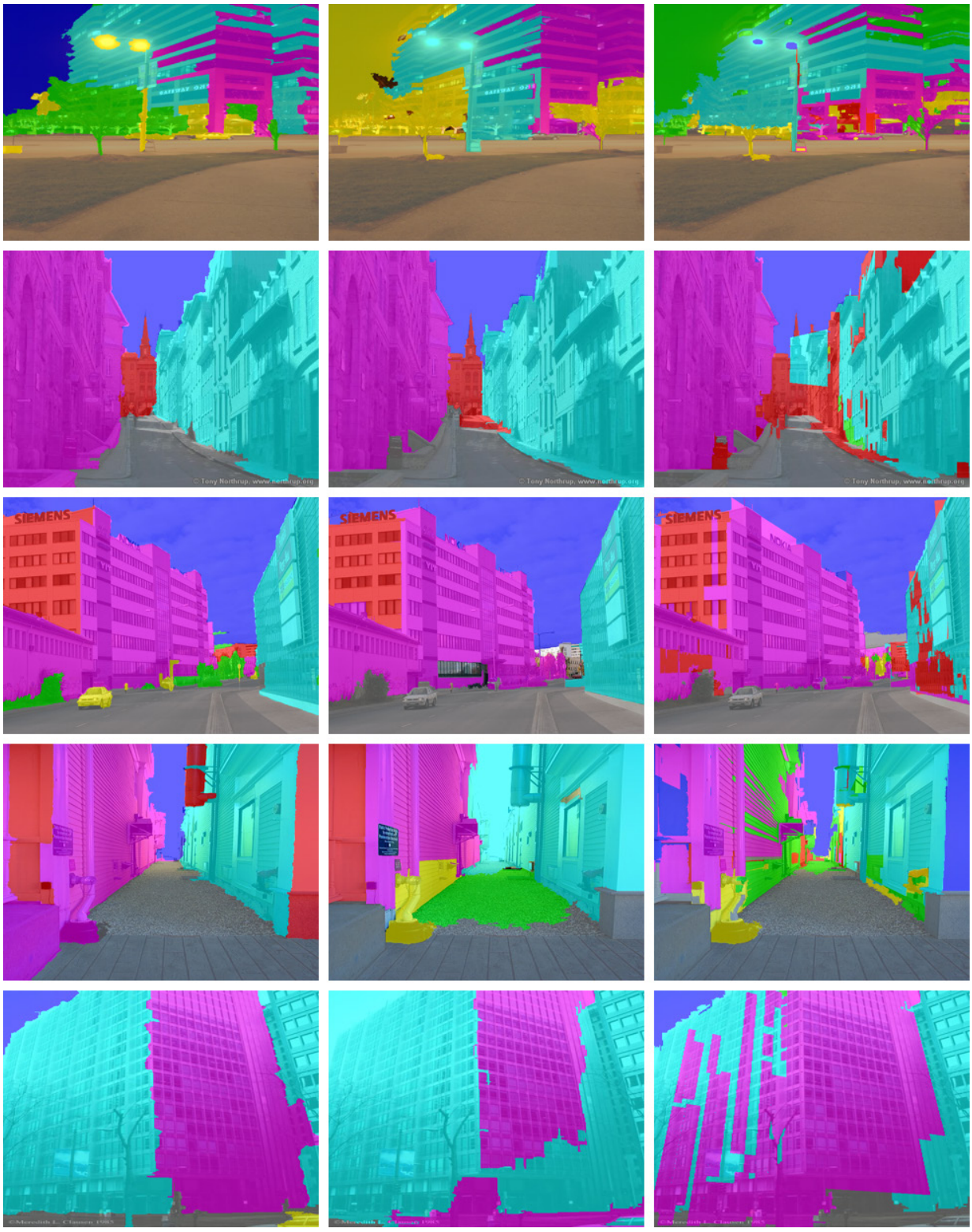

Figure A.20: Comparison of surface layout classification between our approach in Chapter 5 and the algorithm in [25]. Images 41-45.

Bibliography

[1] M. Antone and S. Teller. Automatic recovery of relative camera rotations for urban scenes. *CVPR*, 2000. 3.5

[2] C. Atanasoaei, C. McCool, and S. Marcel. A principled approach to remove false alarms by modeling the context of a face detector. *BMVC*, 2010. 2.1

[3] S. Y. Bao, M. Sun, and S. Savarese. Toward coherent object detection and scene layout understanding. *Image and Vision Computing*, 2011. 1.1, 1, 2, 4, 2.3, 4, 4.5

[4] B. Brillaut-O'Mahony. New method for vanishing point detection. *CVGIP: Image Understanding*, 54(2):289–300, September 1991. 3.5

[5] M. J. Choi, A. Torralba, and A. S. Willsky. A tree-based context model for object recognition. *PAMI*, 2012. 2.1

[6] W. Choi, Y.-W. Chao, C. Pantofaru, and S. Savarese. Understanding indoor scenes using 3d geometric phrases. *ICCV*, 2013. 2.4

[7] R. Collins. Vanishing point calculation as statistical inference on the unit sphere. *ICCV*, pages 400–403, 1990. 3.5

[8] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *PAMI*, 2002. 4.3.2

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR*, 2005. (document), 3.2, 4.3.1, 4.9, 4.5

[10] A. P. Dempster, N. M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977. 3.5

[11] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert. An empirical study of context in object detection. *CVPR*, 2009. 2.1

[12] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: efficient and robust 3d object recognition. *CVPR*, 2010. 2.2

[13] P. F. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. *CVPR*, 2008. 3.2

[14] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Cascade object detection with deformable part models. *CVPR*, 2010. 3.2

[15] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9):1627–1645, 2010. (document), 3.2, 4.5, 4.9

[16] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 1981. 1.3, 3.5

[17] D. Fouhey, V. Delaitre, A. Gupta, A. A. Efros, I. Laptev, and J. Sivic. People watching: human actions as a cue for single view geometry. *ECCV*, 2012. 1

[18] R. B. Girshick. From rigid templates to grammars: object detection with structured models. *Ph.D. dissertation, The University of Chicago*, 2012. (document), 3.2, 3.2

[19] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. http://people.cs.uchicago.edu/ rbg/latent-release5/. (document), 3.2, 3.2, 6.4, 6.5, 6.6

[20] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester. Object detection with grammar

models. *NIPS*, 2011. 3.2

[21] A. Golovinskiy, V. G. Kim, and T. Funkhouser. Shape-based recognition of 3d point clouds in urban environments. *ICCV*, 2009. 2.2

[22] S. Gould. Probabilistic models for region-based scene understanding. *Ph.D. Thesis, Stanford University*, 2010. 2.2, 6.1

[23] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. *ICCV*, 2009. 1.1, 2.2, 2.4, 3.3, 5, 5.4.2

[24] S. Gould, T. Gao, and D. Koller. Region-based segmentation and object detection. *NIPS*, 2009. 2.2

[25] A. Gupta, A. A. Efros, and M. Hebert. Blocks world revisited: image understanding using qualitative geometry and mechanics. *ECCV*, 2010. (document), 1.1, 1, 3, 2.4, 5, 5.1, 5.2.3, 5.4.1, 5.14, 5.4.2, 5.15, 5.16, 5.4.2, 5.18, 5.5, 6.1, A, 1, A.12, A.13, A.14, A.15, A.16, A.17, A.18, A.19, A.20

[26] S. Hadfield and R. Bowden. Hollywood 3d: recognizing actions in 3d natural scenes. *CVPR*, 2013. 1

[27] V. Hedau, D. Hoiem, and D. Forsyth. Recovering free space of indoor scenes from a single image. *CVPR*, 2012. 2.4

[28] G. Heitz and D. Koller. Learning spatial context: using stuff to find things. *ECCV*, 2008. 2.1

[29] J. Hernandez and B. Marcotegui. Morphological segmentation of building facade images. *ICIP*, 2009. 7.1

[30] T. Heskes. Stable fixed points of loopy belief propagation are minima of the bethe free energy. *Advances in NIPS*, 2003. 4.5, 5.3.2

[31] D. Hoiem. Seeing the world behind the image: spatial layout for 3d scene understanding. *Ph.D. Thesis, Carnegie Mellon University*, 2007. 1.1, 1, 2, 2.3

[32] D. Hoiem, A. A. Efros, and M. Hebert. Code for recovering surface layout from an image. http://www.cs.illinois.edu/homes/dhoiem/. 3.4, 5.4.2, 5.4.3

[33] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 2007. (document), 1.4.2, 1.4.3, 2.4, 3.4, 3.5, 3.3, 3.4, 4.3.1, 5, 5.1, 5.2.2, 5.4.2, 5.19, 6.4

[34] D. Hoiem, A. A. Efros, and M. Hebert. Putting objects in perspective. *IJCV*, 2008. (document), 1.1, 1.4.1, 1, 2, 4, 2.3, 3.2, 3.3, 3.4, 3.3, 3.5, 3.6, 4, 4.3.1, 4.3.2, 4.5, 4.9, 4.10, 5.4.1, 5.10, 5.4.1, 5.11, 5.12, 5.13, 6.4, 6.4, 6.5, 6.6, 6.12, 6.4, 6.4

[35] D. Hoiem, A. A. Efros, and M. Hebert. Closing the loop on scene interpretation. *CVPR*, 2008. (document), 3, 2.4, 5, 5.1, 5.14, 5.4.2, 5.15, 5.16, 5.4.2, 5.17, 6.1, A, A.1, A.2, A.3, A.4, A.5, A.6, A.7, A.8, A.9, A.10, A.11

[36] D. Hoiem, A. A. Efros, and M. Hebert. Recovering occlusion boundaries from an image. *IJCV*, 2011. 2.4, 5

[37] J. Huang and S. You. Point cloud matching based on 3d self-similarity. *CVPR*, 2012. 2.2

[38] W. S. Hwang and J. Weng. Hierarchical discriminant regression. *PAMI*, 2000. 3.2, 4.3.1

[39] G. Kamberov and G. Kamberova. Geometric integrability and consistency of 3d point clouds. *ICCV*, 2007. 2.2

[40] S. B. Kang and R. Szeliski. 3-d scene data recovery using omnidirectional multibaseline stereo. *CVPR*, 1996. 2.2

[41] J. J. Koenderink. Pictorial relief. *Phil. Trans. of the Roy. Soc.*, 1998. 1.1

[42] J. J. Koenderink, A. J. V. Doorn, and A. M. L. Kappers. Pictorial surface attitude and local depth comparisons. *Perception and Psychophysics*, 1996. 1.1

[43] J. Kosecka and W. Zhang. Video compass. *ECCV*, 2002. (document), 3.5, 3.5, 3.7, 3.8

[44] J. Lafferty, A. McCallum, and F. Pereirac. Conditional random fields: probabilistic models for segmenting and labeling sequence data. *ICML*, 2001. 1.3, 4

[45] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. *CVPR*, 2009. (document), 2.4, 3.6, 3.9, 3.10, 3

[46] D. C. Lee, A. Gupta, M. Hebert, and T. Kanade. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. *NIPS*, 2010. 2.4

[47] L-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding:classification, annotation and segmentation in an automatic framework. *CVPR*, 2009. 2.1

[48] B. Liu, S. Gould, and D. Koller. Single image depth estimation from predicted semantic labels. *CVPR*, 2010. (document), 1.4.3, 3, 2.2, 3.4, 3.3, 5.1, 5.4.3, 5.20, 5.4.3, 5.21, 5.22, 5.5, 6.1, 6.4

[49] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing via label transfer. *PAMI*, 33 (12), December 2011. 7.3

[50] T. Meltzer, C. Yanover, and Y. Weiss. Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. *ICCV*, 2005. 2.2

[51] D. Munoz, J. A. Bagnell, and M. Hebert. Stacked hierarchical labeling. *ECCV*, 2010. (document), 1.2.3, 3.4, 3.3, 3.5, 5.2.2, 5.4.2, 5.4.3

[52] K. Murphy, A. Torralba, and W. Freeman. Using the forest to see the tree: a graphical model relating features, objects and the scenes. *NIPS*, 2003. 2.1

[53] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. *ICCV*, 2007. 2.1

[54] S. Ramalingam and M. Brand. Lifting 3d manhattan lines from a single image. *ICCV*, 2013. 2.4, 5.4.1

[55] M. Recky and F. Leberl. Window detection in complex facades. *European Workshop on Visual Information Processing*, 2010. 7.1

[56] M. Recky and F. Leberl. Windows detection using k-means in cie-lab color space. *ICPR*, 2010. 7.1

[57] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *Technical report, MIT*, 2005. 1.4.1, 4.3.1, 4.5

[58] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. *NIPS*, 2005. 1.1, 2.2, 5.1

[59] A. Saxena, S. H. Chung, and A. Y. Ng. 3-d depth reconstruction from a single still image. *IJCV*, 2007. 1.1, 2.2

[60] A. Saxena, M. Sun, and A. Y. Ng. Make3d: learning 3-d scene structure from a single still image. *PAMI*, 2009. (document), 1.1, 1.4.3, 2.2, 3.3, 5.1, 5.4.3, 5.22

[61] K. Schmid and H. Hirschmuller. Stereo vision and imu based real-time ego-motion and depth image computation on a handheld device. *ICRA*, 2013. 1

[62] K. Schmid, T. Tomic, F. Ruess, H. Hirschmuller, and M. Suppa. Stereo vision based indoor/outdoor navigation for flying robots. *IROS*, 2013. 1

[63] A. Sehgal, D. Cernea, and M. Makaveeva. Real-time scale invariant 3d range point cloud registration. *ICIAR*, 2010. 2.2

[64] A. Serna, J. Hernandez, and B. Marcotegui. Adaptive parameter tuning for morphological segmentation of building facade images. *European Signal Processing Conference*, 2012. 7.1

[65] A. Singhal, J. Luo, and W. Zhu. Probabilistic spatial context models for scene content understanding. *CVPR*, 2003. 2.1

[66] M. Sun, S. Y. Bao, and S. Savarese. Object detection with geometrical context feedback loop. *BMVC*, 2010. 1.1, 1, 2, 2.3, 4, 4.5

[67] R. Szeliski and D. Scharstein. Symmetric sub-pixel stereo matching. *ECCV*, 2002. 2.2

[68] O. Teboul, L. Simon, P. Koutsourakis, and N. Paragios. Segmentation of building facades using procedural shape priors. *CVPR*, 2010. 7.1

[69] S. Vishwakarma and A. Agrawal. A survey on activity recognition and behavior understanding in video surveillance. *The Visual Computer*, 2013. 1

[70] G. Vogiatzis, C. Hernandez, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. *CVPR*, 2005. 2.2

[71] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Code for line segment detection. http://www.ipol.im/pub/art/2012/gjmr-lsd/. 2

[72] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Lsd: A fast line segment detector with a false detection control. *PAMI*, 2010. 3.5

[73] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Lsd: a line segment detector. *Image Processing On Line*, 2012. 3.5

[74] M. P. Wand and M. C. Jones. *Kernel smoothing*. London: Chapman and Hall/CRC, 1995. 4.3.1

[75] C. Wu, J. M. Frahm, and M. Pollefeys. Detecting large repetitive structures with salient boundaries. *ECCV*, 2010. 7.1

[76] Y. Zhao and S. C. Zhu. Scene parsing by integrating function, geometry and appearance models. *ICCV*, 2013. 2.4

[77] W. S. Zheng, S. Gong, and T. Xiang. Quantifying and transferring contextual information in object detection. *PAMI*, 2012. 2.1

[78] G. Zimmerman, G. Legge, and P. Cavanagh. Pictorial depth cues: a new slant. *Journ. of the Optical Soc. of America A*, 1995. 1.1