VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

REFLECTION DETECTION AND REMOVAL FROM IMAGE SEQUENCES

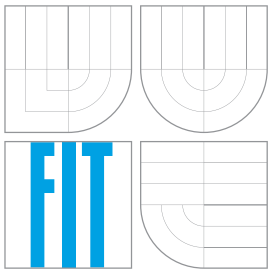
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. TOMÁŠ HODAŇ

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DETEKCE A ODSTRANĚNÍ ODLESKŮ ZE SEKVENCE SNÍMKŮ

REFLECTION DETECTION AND REMOVAL FROM IMAGE SEQUENCES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. TOMÁŠ HODAŇ

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. Ing. ADAM HEROUT, Ph.D.

BRNO 2013

Abstrakt

Cílem mé diplomové práce bylo studium existujících metod pro detekci a odstranění odlesků ze sekvence snímků, nalezení jejich omezení a návrh možných vylepšení. Konkrétně jsme se zaměřili na rovinné spekulární povrchy, jejichž vzhled může být modelován superpozicí odrazové a přenosové vrstvy. Prozkoumali jsme především metody využívající vzájemný pohyb vrstev jako hlavní klíč k jejich oddělení. Popsali jsme společný případ selhání těchto metod, který spočívá v neschopnosti správného oddělení oblastí s nevýraznou texturou ve směru pohybu kamery. Výsledkem našeho úsilí je metoda řešící tento problém. Jejím přínosem je nový způsob odhadu hran obou vrstev, kdy důraz je kladen na správné oddělení hran zmíněných problematických oblastí. Hrany vrstev jsou pak spolu s odhadem hloubkových map vrstev využity při odhadu barev obou vrstev. Odhad barev může být získán pomocí kvadratického programování nebo pomocí námi popsaného a méně výpočetně náročného alternativního přístupu. Výsledky navržené metody překonávají výsledky existujících metod, a to především v problematických oblastech.

Abstract

The aim of the Master's thesis was to study existing methods for detection and removal of specular reflection from image sequences, to find their limitations and to suggest possibilities of their improvements. Particularly, an attention was paid to planar specular (i.e. mirror-like) surfaces whose appearance can be modeled by linear superposition of reflection and transmission layer. We reviewed the existing motion-based methods and described their common degenerate case in terms of their disability to correctly recover regions with low frequency in the direction of camera motion. A new method designed to eliminate this degenerate case was suggested. Its contribution is a new approach to layer gradients estimation with a special treatment of the gradients forming edges of the problematic regions. The estimated layer gradients, together with estimated layer depth maps, are then used for recovery of layer colors which can be treated as a quadratic programming problem or can be done by a more efficient alternative approach which we introduced. The suggested method was shown to outperform the existing methods, especially in the problematic regions.

Klíčová slova

spekulární odlesk, detekce, odstranění, oddělení odrazové a přenosové vrstvy

Keywords

specular reflection, detection, removal, reflection and transmission separation

Citation

Tomáš Hodaň: Reflection Detection and Removal From Image Sequences, master's thesis, Brno University of Technology - FIT, 2013

Declaration

I hereby declare that I have carried out the Master's thesis independently under supervision of Doc. Ing. Adam Herout Ph.D. from Brno University of Technology and Dr. Robby T. Tan PhD from Utrecht University.

.....
Tomáš Hodaň
May 22, 2013

Acknowledgment

I would like to express my gratitude to Dr. Robby T. Tan PhD from Utrecht University who suggested me the topic and constantly supplied me with useful advice and scientific papers. I would also like to thank to Doc. Ing. Adam Herout Ph.D. who has shown an encouraging and consistent interest in my work and enabled my attendance of BMVA Computer Vision Summer School and my visit of Utrecht University.

Contents

Chapter 1

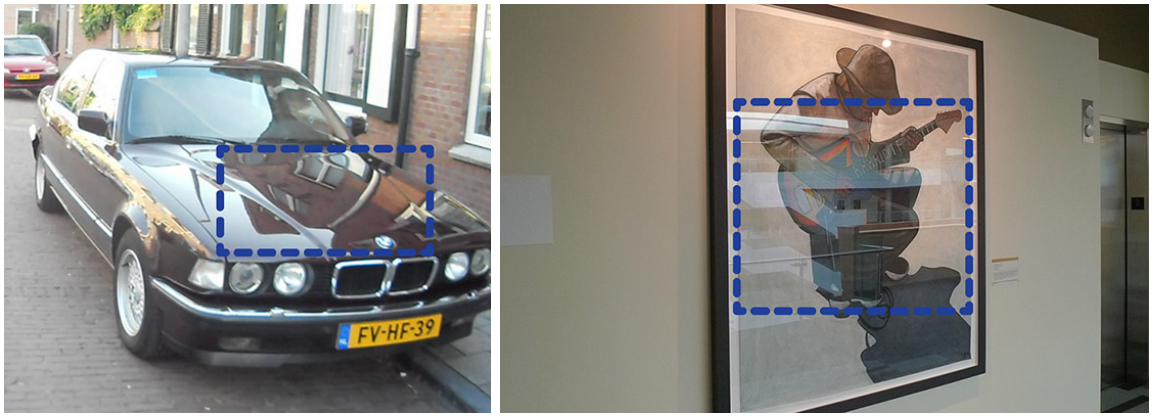
Introduction

1.1 Problem Formulation

The aim of the Master's thesis is to study existing methods for detection and removal of specular reflection, to find their limitations and to suggest possibilities of their improvements.

As the target was chosen a class of planar specular (i.e. mirror-like) surfaces whose appearance can be modeled or approximated by linear superposition of layers. Typical examples include a reflection superposed with texture of a reflective material, or a reflection superposed with a scene behind a transparent material. The problem of recovering specular reflection from this class of surfaces is usually referred to as the *reflection and transmission separation*.

1.2 Motivation



(a) (b)

Figure 1.1: **(a)** *Is there a car's hood or a building in the highlighted region?* **(b)** *Are we looking at the painting or the building interior?*

Specular reflection in general is a source of problems in many fields of computer vision. It makes the appearance of objects inconsistent which yields difficulties in object recognition and image segmentation. Its view-dependent nature then causes troubles in stereo vision

and optical flow estimation. The cause of these problems is that many methods rely on assumption about ideal diffusely reflecting surfaces and hence fail in the presence of non-Lambertian effects such as specular reflection.

An ambiguity in object recognition caused by specular reflection is illustrated in Figure 1.1(a). In the highlighted region, the system could be confused and think there is a real house on the hood. On the other hand, the appearance of the car is changed so significantly by the reflection that the car may not be recognized at all. A similar case can be seen in Figure 1.1(b) where the painting is interfered with reflection of the building interior.

The ability to separate reflection and transmission layers could eliminate the demonstrated ambiguity. Besides a potential increase of performance in the mentioned fields of computer vision, another important application could be found in image-based rendering and compression of scene appearance when view-dependent reflection layer and view-independent transmission layer could be treated separately.

As noted by Szeliski et al. [30], reflections and transparency are about as ubiquitous as images themselves and thus more attention should be paid to these often ignored phenomena. This Master's thesis aims to contribute to this effort.

1.3 Thesis Organization

- **Chapter 2** introduces the problem of reflection and transmission separation in more detail. The related theory is explained, possible approaches are reviewed, the degenerate case common for all motion-based methods is described and a pipeline of our suggested method, which is designed to eliminate the degenerate case, is introduced.
- **Chapter 3** describes the method for two-layer depth estimation which represents the first stage of our method.
- **Chapter 4** introduces our approach to layer gradients estimation which plays an important role in elimination of the degenerate case.
- **Chapter 5** reveals possible quadratic programming approach and also our more efficient alternative to recovery of layer colors using the estimated layer depths and gradients.
- **Chapter 6** presents the datasets which were used in our experiments, including the synthetic sequences which we created.
- **Chapter 7** gives more details about the implementation and the used libraries.
- **Chapter 8** concludes the thesis, highlights the contributions and suggests possible future work.

Within the term project, which I was working on during the first semester, the two-layer depth estimation described in Chapter 3 was implemented.

Chapter 2

Reflection and Transmission Separation

The theory related to the reflection and transmission separation is explained in Section 2.1. Possible approaches to this problem are reviewed in Section 2.2. Section 2.3 then describes the degenerate case of the motion-based approaches which we aim to eliminate in our suggested method whose pipeline is introduced in Section 2.4.

2.1 Related Theory

2.1.1 Geometry of Specular Reflection

In the case of a planar specular (or mirror-like) surface, the virtual image of a reflected scene point is independent of the camera location and thus it is located at a fixed point behind the specular surface (Figure 2.1). Moreover, the distance of the scene point from the specular surface is the same as of the virtual image.

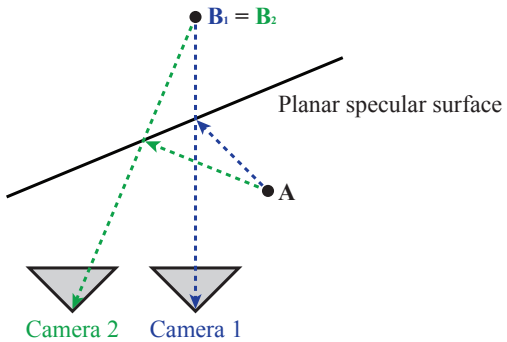


Figure 2.1: *Specular reflection on a planar surface* (the virtual image **B** of a scene point **A** is viewpoint independent).

More complicated situation occurs when the specular surface is curved. The position of the virtual image is viewpoint dependent and therefore not fixed any more [11]. Also the distances of the scene point and its virtual image from the specular surface usually vary. The locus of virtual image is given by a catacaustic curve (Figure 2.2) which is defined by

geometry of the specular surface and position of the reflected scene point. Any point of the locus is visible only along the tangent ray to the catacaustic curve.

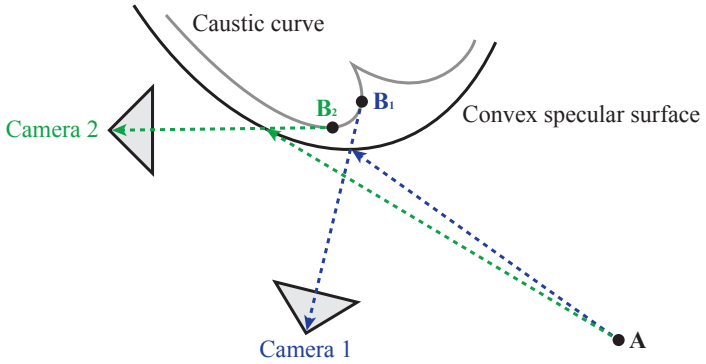


Figure 2.2: *Specular reflection on a convex surface* (the virtual image **B** of a scene point **A** is viewpoint dependent).

For a convex specular surface, the virtual image stays geometrically always behind this surface. The situation is more complex for a concave specular surface when the virtual image is in the front of the surface and moves rapidly towards negative infinity as the size of curvature radius increases to the size equal to the object distance. By further increasing the radius the virtual image jumps to positive virtual depths, being behind the specular surface.

To simplify the problem, it is assumed that the specular surface is flat or slightly curved. This implies that the virtual image is always behind the surface and its location can be considered to be approximately fixed.

2.1.2 Image Formation Process

In order to recover color of reflection and transmission light at a planar specular surface, it is necessary to define the image formation process at such surface. We use an additive layered model [27, 32] assuming that the irradiances of scene points along a viewing ray are linearly combined to produce a composite color for the corresponding pixel. In particular, a camera captures a linear combination of a light transmitted (refracted) through a material such as glass and of a reflected light which produces a stable virtual image behind the specular surface (Figure 2.3). For materials such as glossy textured surfaces (e.g. paintings matted with glass, textured countertops), the geometry corresponding to the transmitted light coincides with the reflective surface.

Since both transmitted and reflected components appear as stable virtual images, we cannot distinguish between them.[1] Hence, the layers are usually referred to according to their distance from the camera. The layer whose virtual image appears closer to the camera is called the *front* layer (I_0) and the other one is called the *rear* layer (I_1). However, in the datasets used in this thesis it is always the case that the front and the rear layer represent the transmission and the reflection layer respectively.

The observed composite image C can be described as:

[1]The virtual image of a reflected object can be closer to the camera than the transmitted object. This is an opposite case to the one illustrated in Figure 2.3.

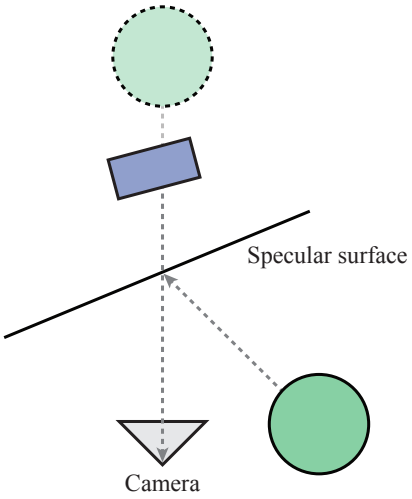


Figure 2.3: *Image formation process.* The light going from the blue box is transmitted through the transparent specular surface while the one going from the green sphere is reflected on the surface. The position of the reflected green sphere appears to be at its virtual image (the dashed sphere).

$$C = I_0 + \beta I_1, \tag{2.1}$$

where β is the reflective field determining how the incident light is attenuated at reflective material boundaries. It depends on material's BRDF (bidirectional reflectance distribution function) [19] or Fresnel reflection coefficient. The transmitted component passes through the most of materials, such as plate glass, clear coat or varnish, unattenuated. The values of β are non-zero at specular surfaces and zero at matte opaque surfaces.

When trying to recover the quantities I_0, I_1, and β, we cannot disambiguate between larger reflection coefficients β and larger amounts of incident light I_1. To resolve this ambiguity, we can suppose that reflective surfaces have a constant non-zero coefficient β. This constant factor can be folded into I_1 and its value can be assumed to be 1 at specular surfaces and 0 at matte opaque surfaces [27].

2.1.3 Mixing Model

As the input, we assume a sequence of images C_v taken from several views $v \in V$ at a static scene containing a specular surface. There are no limitations on the extent of the specular surface. It can be present in one or several local regions but can also cover the whole image.

Each of the input images can be expressed as a mixture of layers which are warped from the reference image:

$$C_v = T_{v,0} \circ I_0 + (T_{v,0} \circ \beta) \cdot (T_{v,1} \circ I_1), \tag{2.2}$$

where $T_{v,l}$ is a function warping an image from the reference view to the view v using the depth map of layer l and the camera parameters from the two views (the warping maps are piecewise continuous but can have gaps at depth discontinuities).

Because the reflective material associated with β is not always directly observable, we cannot reliably estimate its depth d_β and therefore consider $d_\beta = d_0$ (as in [27, 32]). This approximation provides a correct model when the transmission components coincide with the specular surface. When these components are located behind the specular surface, it is still a sensible choice unless there are significant depth discontinuities in d_0.

2.1.4 Relative Motion of Layers

From the geometry of reflection on a planar specular surface (Section 2.1.1) follows that the virtual images of reflection and transmission layer are usually located at different depths.[2] Hence, when the camera moves, the layers move relatively to each other at a rate depending on their depth (Figure 2.4). This is used as the main clue by the motion-based methods (Section 2.2.2).

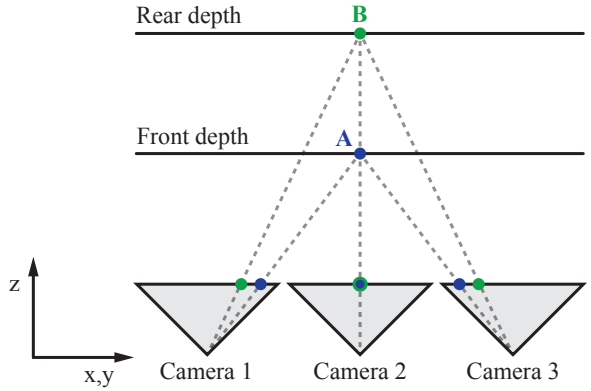


Figure 2.4: *Relative motion of points at different layers.* Their relative motion is most noticeable when the camera moves laterally. The apparent motion (disparity) of the 3D points is then inversely proportional to their depth.

2.2 Possible Approaches

This section reviews several directions from which the problem of reflection and transmission separation has been addressed.

2.2.1 Optical Approaches

One direction of approaches uses principles from optics to obtain different mixtures by alteration of mixing coefficients. The layers are then extracted by minimization of their statistical dependency.

Some methods introduce polarization filters in photography and perform Independent Component Analysis (ICA) [12, 24, 5]. Other obtain the mixtures with different camera focuses and use multichannel Blind Deconvolution (BD) [23, 22].

Limitation of these methods lies in the assumption of static mixing (i.e. there is no layer motions between different mixtures) which is not always easy to satisfy in practise.

[2] We ignore the case when the reflection and transmission components are at the same depth, as well as the other authors focused on the motion-based methods, which are described in Section 2.2.2.

2.2.2 Motion-Based Approaches

Another direction utilizes the relative motion of layers (described in Section 2.1.4) as the main clue for their separation.

In their work, Shizawa et al. [25, 26] provide an elegant solution to extract the relative motion of layers by estimation of multiple optical flow. Unfortunately, this approach has not been applied to real images or simulated image sequences with noise and thus the level of its robustness is unknown.

Szeliski et al. [30] at first estimate motion (approximated by homography) of the dominant (i.e. higher contrast) layer. This motion estimate is then used to transform the composite images such as the dominant layer is aligned. After this initial registration, they compute a min-composite to recover an upper bound on the dominant layer colors, and then use a max-composite of the residual difference images which is used to obtain a lower bound of colors of the second layer and to recover its associated motion. For each pixel, the min-/max-composite is obtained by taking the minimum/maximum of the stack of registered images. At the end, to refine both layer colors they solve a constrained least-squares problem to minimize the deviation from the mixing model (2.2).

Gai et al. [13] extended the Blind Source Separation (BSS) framework [7] to handle mixtures with not only unknown mixing coefficients (as the methods mentioned in 2.2.1) but also unknown parametric layer motions. At first, they search for the layer motions by maximizing the correlation of gradients. Clusters of correlated gradients are then assigned into respective layers. To recover layer colors, they solve a quadratic programming problem tending to agree with the mixing model and the estimated layer gradients. This is currently the only motion-based method which can cope with varying mixing coefficients. Moreover, their method can also deal with more than two layers.

Another methods use motion to estimate a two-layer depth map which is then employed to separate layer colors by a constrained least-squares problem, similarly as was done by Szeliski et al. [30]. Since these methods do not rely on any parametric motion model, they can deal with more complex scenes with depth discontinuities in both layers. Methods taking this approach were developed by Tsin et al. [32] and Sinha et al. [27]. The latter one suggested a system to model both reflection and transmission layer from several known views and then to interpolate between them to render a novel view. We have adopted its part for two-layer depth map estimation in our work.

2.2.3 Approaches Using a Single Image

Besides the methods using an image sequence as the input (all the methods mentioned in Sections 2.2.1 and 2.2.2), there are also methods using only a single image.

The work of Levin et al. [17] is based on a simple prior knowledge that the correct decomposition should have a small number of edges and corners. The algorithm first searches a database of natural images for possible decompositions and then minimizes a cost function measuring the number of edges and corners. The method often fails because the candidate decompositions do not include any suitable decomposition, or because the optimization process is too complex and cannot find the optimal separation.

Another method working with a single image was developed by Yeung et al. [33]. It focuses on an easier problem of images where only the background layer has substantial image gradients and thus its application is very limited.

2.3 Degenerate Case of Motion-Based Approaches

As has been reviewed in Section 2.2, the most practical clue for layers separation seems to be the relative motion of layers. Methods exploiting this clue do not suffer from any requirement of special scene settings (as the methods based on clues from optics) or from the ambiguous optimization problem (as the methods using only a single image). This is the reason why the attention was paid mainly on the motion-based methods in this thesis.

During our analysis of the existing motion-based methods, we observed that all of them share one common degenerate case: **they are not able to correctly separate regions with low frequency (i.e. low texture) in the direction of camera motion**. These problematic regions do not have any apparent motion and thus cannot be tracked and reliably assigned to the correct layer. To the best of our knowledge, this limitation has not been handled or described in any publication yet.

The problem appears especially when the camera moves approximately in an uniform direction. But since we typically assume to have several consecutive video frames as the input, this can be often the case.

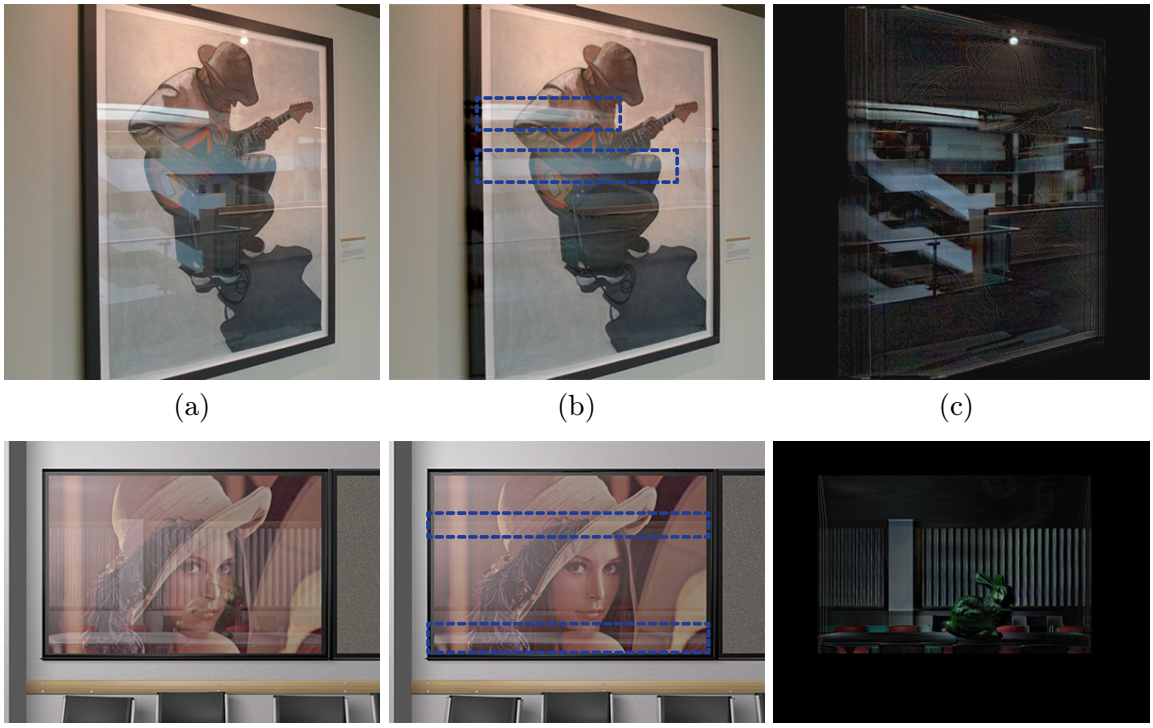
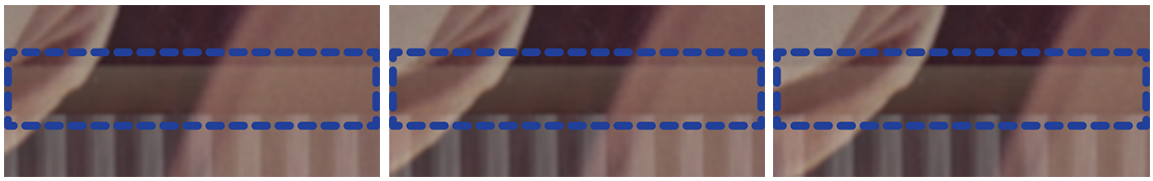


Figure 2.5: *Problematic regions.* **(a,d)** Sample of the input image sequence. **(b,e)** Estimated transmission layer (some of the problematic regions are highlighted). **(c,f)** Estimated reflection layer.

Results presented by Sinha et al. [27], where the horizontal reflection elements were assigned to the wrong layer,[3] can be seen in Figure 2.5(b,c). Similar problem can be seen

[3]It should be noted that for their purpose (i.e. image-base rendering) this separation is sufficient. However, we use their results to illustrate the problem with layer separation since it is common for all the existing motion-based methods.

in Figure 2.5(e,f) which shows results of our implementation of the color recovery method presented by Tsin et al. [32].



(a) (b) (c)

Figure 2.6: *Detailed view at a problematic region from Figure 2.5(e) in three consecutive frames taken with laterally moving camera (moving from right to left).*

A concrete example of the problematic region is the reflected wooden frame in Figure 2.6. It does not have any significant texture in the direction of camera motion. Although it has in fact the same motion as the curtain located below it, its motion is not apparent. On the other hand, in the transmission layer (containing e.g. a part of the hat) and the lower part of the reflection layer (containing the curtain), there is a significant texture in the direction of the camera motion and therefore the motion of these regions is apparent.

Even if we had the correct motion/depth estimates, the problematic regions would cause troubles also in the color recovery stage (as is demonstrated in Figure 2.5(e,f) where the ground truth of layer depths was used to generate these results). A common approach, used by the motion-based methods to recover layer colors, is to minimize an energy expressing deviation from the mixing model (2.2). As is described in detail in Section 5, some of the methods introduce extensions of the energy which encourages smoothness or agreement with the estimated gradients. None of these extensions is capable to reliably deal with the mentioned problematic regions. Since the min-composite (described in Section 2.2.2) is often used to initialize the color recovery process [30, 27, 32], the methods tend to assign the problematic regions to the transmission layer, as is evident from Figure 2.5.

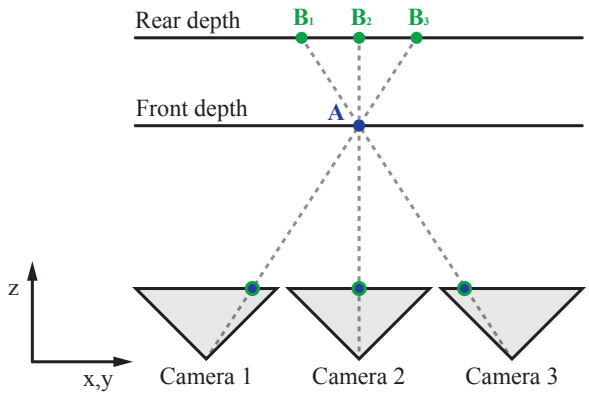


Figure 2.7: *Geometry of the problematic situation.* The color of the front layer point is still mixed with the same color of the rear layer.

The cause of the failure in color recovery can be understood from Figure 2.7. There are three laterally shifted cameras capturing a scene consisting of both transmission (front) and reflection (rear) layer whose depths are known. We cannot recover color of the front

layer point if there is a problematic region behind it in the rear layer because the color of the front layer is still mixed with the same color of the rear layer. This situation can appear also vice versa, i.e. the problematic region can be in the front layer.

2.4 Suggested Method

Our aim was to create a method which could handle the described degenerate case, i.e. we wanted the method to be able to assign each problematic region to the correct layer.

The key idea is to handle the degenerate case in the gradient space at first. This is done by estimation of image gradients of both layers with a special treatment of the problematic gradients.[4] The gradient space is sparser and it is thus easier to correctly assign the problematic gradients to the correct layer (using suitable topological observations). The estimated layer gradients are then used in the color recovery process where the gradients of layer colors are encouraged to agree with these gradients. We showed that if the layer gradients are estimated correctly, the layer colors can be recovered reliably even in the problematic regions.

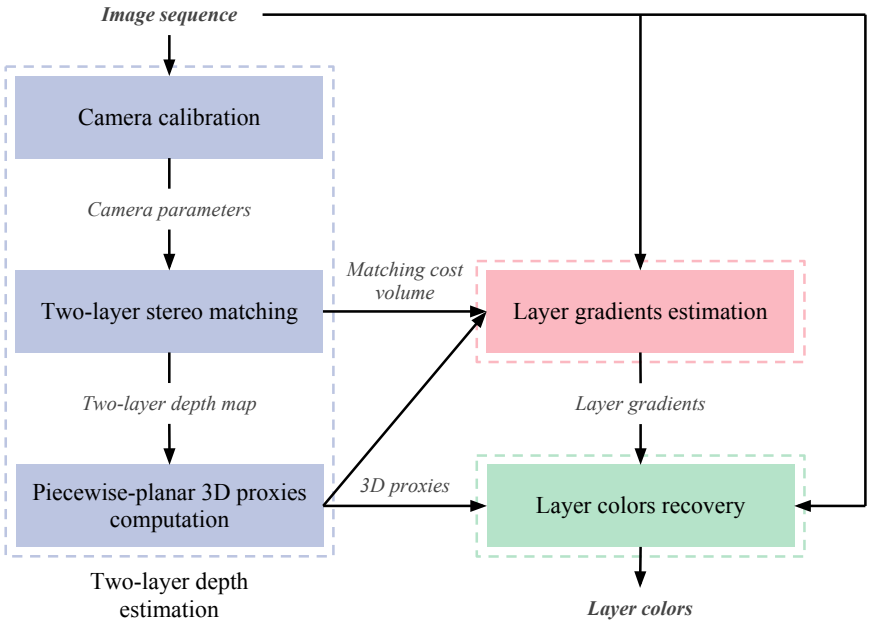


Figure 2.8: *Pipeline of the suggested method.*

The suggested method receives an image sequence as the input and produces color estimation of reflection and transmission layer (as seen from the reference view) as the output. The whole pipeline of the method is illustrated in Figure 2.8. It consists of three main stages:

1. **Two-layer depth estimation**
   Two-layer depth map in the form of piecewise-planar 3D proxies is estimated in this

[4] The problematic gradients are the gradients forming edges of the problematic regions.

stage. Layer motion determined by this depth approximation can deal with more complex structure including depth discontinuities and thus is more descriptive than the often used motion approximation by homography (used in e.g. [30, 13]). The most of the techniques employed within this stage were adapted from Sinha et al. [27].

2. **Layer gradients estimation**
   At first, we detect the problematic gradients as the gradients whose depth cannot be determined (i.e. the corresponding pixel has very low matching cost at the most of considered depths). Then we separate gradients with apparent motion whose depth can be determined reliably. Each such image gradient is assigned to the layer whose depth at that pixel (determined by the 3D proxies) is closer to the depth at which the gradient has the minimum matching cost. In the next step, the problematic gradients are clustered into groups representing individual edges and each group is assigned to the layer where it leads to better *topological fitness* with the already assigned gradients.

   From the studied motion-based methods, only the method by Gai et al. [13] performs explicit layer gradients estimation and then use these gradients for the color recovery. However, they do not take any special treatment of the problematic edges and therefore cannot assign them reliably to the correct layer.

3. **Layer colors recovery**
   Once the 3D proxies and the layer gradients have been estimated, layer colors are recovered by minimizing an energy expressing deviation from the mixing model and the estimated gradients. One way how to accomplish minimization of this energy is to treat it as a quadratic programming problem, as was done by Gai et al. [13]. Since this approach has very high computational demand, we have described an alternative approach which reduces computational time significantly.

More details about these stages can be found in Chapters 3, 4 and 5.

Chapter 3

Two-Layer Depth Estimation

This chapter gives a detailed description of the method for two-layer depth map estimation which was adopted from the image-based modeling and rendering system presented by Sinha et al. [27].

The method begins with camera calibration (Section 3.1). This is followed by multi-image stereo matching whose task is to estimate up to two depths for each pixel, one for transmission and one for reflection layer (Section 3.2). The estimated two-layer depth map which is often sparse and noisy, is then refined by extracting representative scene planes and assigning each pixel to one or two of these planes by a graph cut optimization (Section 3.3).

3.1 Camera Calibration by Structure from Motion

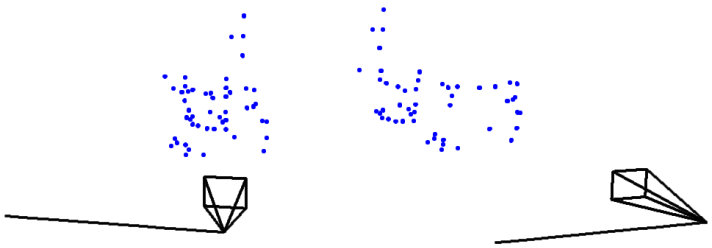
The camera calibration needs to be performed in order to estimate extrinsic and intrinsic camera parameters for the input image sequence. These parameters are later used for multi-image stereo matching.

We used Automatic Camera Tracking System (ACTS) by Zhang et al. [34] which performs the calibration by structure from motion approach [35]. It tracks feature points over consecutive frames and then use feature trajectories to reconstruct their 3D positions and to simultaneously obtain the camera parameters (Figure 3.1 and 3.2). The feature points can be described either with SIFT (Scale Invariant Feature Transform), KLT (Kanade-Lucas-Tomasi) or ENFT (Efficient Non-consecutive Feature Tracking). More details about the descriptors can be found in [29].

In the case of two layers, the calibration is more challenging because there might be many spurious features created by intersections of edges from different layers. Motion of these features do not correspond to motion of any real 3D points and thus can confuse the calibration process.

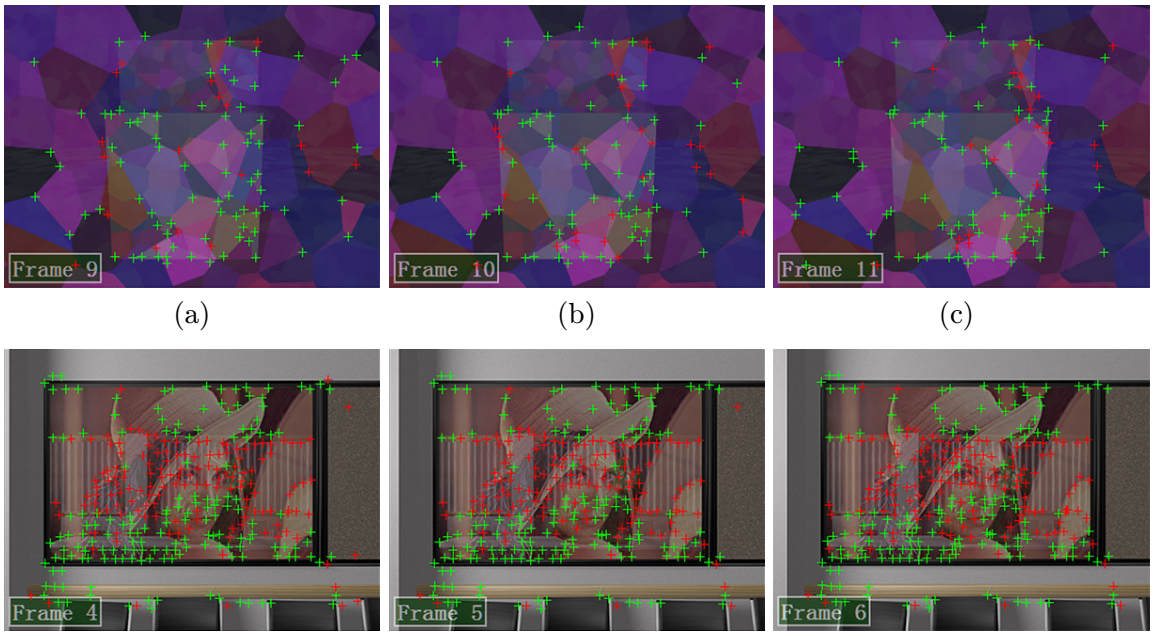
In our experiments, the motion estimates produced by ACTS (using KLT) were sufficient for scenes obtained by laterally moving camera (i.e. the BLOCKS, the CONFERENCE and the GUITARIST sequence - described in Chapter 6). For the synthetic sequences, the motion estimates led to almost the same results in the following stages of depth estimation as the ground truth of camera motion. Nevertheless, for scenes obtained by rather irregularly moving camera (i.e. the MUZEUM and the STATUE sequence), the motion estimates were poor and led to a failure in the following stages. For this reason we worked mainly with the laterally moving camera as the camera calibration was not the main target of the thesis.

We did not analyze or experiment with any other calibration system. However, a promis-



(a) (b)

Figure 3.1: *Two views at the estimated camera trajectory and the 3D sparse reconstruction (blue points) of the CONFERENCE sequence.* This sequence was obtained with a camera moving laterally, thus the estimated camera trajectory produced by ACTS is correct.



(a) (b) (c)

(d) (e) (f)

Figure 3.2: *Sample frames from the BLOCKS and the CONFERENCE sequence with visualized KLT feature points.* (The feature is green if there is a match in some neighbouring frames.)

ing alternative to ACTS could be the system by Snavely et al. [28] which was used with more success in the method by Sinha et al. [27].

3.2 Two-Layer Stereo Matching

Stereo matching is a process of taking two or more images and estimating depth map of a scene by finding matching pixels and converting their relative 2D positions into 3D depths [29]. A typical stereo matching method consists of the following steps [15]:

1. **Matching cost computation**
   The goal is to measure similarity of pixels in one image with pixels falling into search space in the other images. The search space is usually given by a range of plausible disparities (Section 3.2.1).

2. **Cost aggregation**
   Matching cost computation is generally ambiguous (wrong matches can easily have a lower cost than correct ones, due to noise, etc.). This ambiguity can be decreased by an aggregation which connects the matching costs within a certain neighbourhood and thus implies smoothness (Section 3.2.2).

3. **Depth computation**
   This is usually done by selecting the depth with the lowest matching cost, i.e. by winner takes all strategy. In our case, we want to determine one or two depths (Section 3.2.3).

4. **Depth refinement**
   Further refinement is often done to remove peaks, check the consistency or interpolate gaps. Within the implemented method, representation of the scene by the piecewise-planar 3D proxies (Section 3.3) can be seen as a step with this purpose.

For the sake of completeness, note that these steps are common for local methods. In the case of global methods, the cost aggregation and depth computation is done simultaneously by optimization of a global energy function [15].

3.2.1 Computation of the Matching Cost Volume

The matching cost can be calculated by the absolute or squared difference of intensities, colors or image gradients of image patches. Statistics-based measures, such as normalized cross correlation (NCC) [27] or mutual information (MI) [15], can be used as well.

We followed the implementation of Sinha et al. and used NCC which is defined as:

$$NCC(\mathbf{q}_0, \mathbf{q}_1) = \frac{\sum_{x,y} \left(\mathbf{q}_0(x,y) - \overline{\mathbf{q}}_0\right)\left(\mathbf{q}_1(x,y) - \overline{\mathbf{q}}_1\right)}{\sqrt{\sum_{x,y} \left(\mathbf{q}_0(x,y) - \overline{\mathbf{q}_0}\right)^2} \sqrt{\sum_{x,y} \left(\mathbf{q}_1(x,y) - \overline{\mathbf{q}_1}\right)^2}}, \tag{3.1}$$

where $\mathbf{q}_0$ and $\mathbf{q}_1$ are image patches and $\overline{\mathbf{q}}_0$ and $\overline{\mathbf{q}}_1$ their mean values. Subtraction of the mean values from respective patches yields accentuation of high frequencies. NCC hence performs similarly to matching gradient images.

Knowing the camera parameters, we can constrain the space of searching for matching pixels. In the case of stereo matching method working with two input images, we can use the epipolar line corresponding to a pixel in one image to constrain the search for matching pixels in the other image. A typical step is to *rectify* the images so that corresponding horizontal scanlines are epipolar lines [29]. Afterwards, the correspondence of a pixel can be searched only at plausible disparities[1] in respective horizontal scanline in the other image.

[1] In the geometry of rectified images, *disparity* d is inversely proportional to depth: $d = fB/Z$, where f is the focal length, B is the baseline (the distance between cameras), and $x' = x + d(x,y)$, $y' = y$ describes the relationship between corresponding pixel coordinates in the left and the right image [29].

For a scene with only Lambertian surfaces, two images are usually enough to estimate a decent depth map. But as we want to deal with reflections and thus to estimate up to two depths for each pixel, the ambiguity in searching for pixel correspondences gets larger. To decrease this ambiguity, more than just two images need to be considered.

It is not possible to rectify an arbitrary collection of images simultaneously unless their optical centers are collinear [29]. This is not a problem in our case because it is enough to rectify and match the input images pairwise (considering pairs between the reference image and each of the neighbours). The pairwise matching costs could be then combined to obtain the final costs of the reference image. However, to follow the method by Sinha et al., the *plane sweep* algorithm [10, 29] was used instead of this pairwise rectification.

Plane Sweep

The main idea of this multi-image stereo matching method is to sweep a fronto-parallel plane through the scene and to measure the photoconsistency of images as they are warped onto this plane (Figure 3.3).

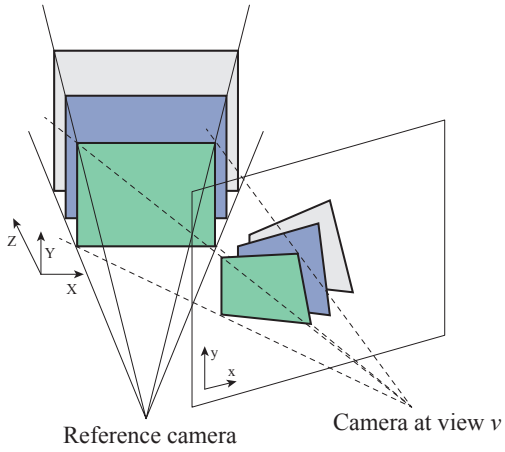


Figure 3.3: *Sweeping plane as seen from the reference camera.* (This figure was adopted from [29].)

To avoid interpolation of missing pixel values it is actually better to project the warped images to the reference image plane and compute the matching costs there. The matching cost $c_{\mathbf{p},Z}$ of pixel $\mathbf{p} = (x, y)$ at depth Z is computed as:

$$c_{\mathbf{p},Z} = \sum_{v \in V} NCC(\mathbf{q_p}, \mathbf{q}_{\mathbf{p}}^{v,Z}), \tag{3.2}$$

where $\mathbf{q}_p$ is the patch centered at pixel $\mathbf{p}$ in the reference image and $\mathbf{q}_p^{j,Z}$ is the patch of image from view v which is transformed using the sweeping plane located at depth Z. Size of both patches is $\mu \times \mu$ (in our implementation μ is set to 3). V is a set of available views. In our experiments we usually worked with 2 or 4 views when on each side of the reference view was a half of them.

In practise, we found that it is beneficial to compute the cost not only from pairs of the reference image and its neighbours but from all possible pairs. In regions with significant

gradients in the front layer, a match in the rear layer can be better revealed using this approach.

The depth range in which the plane is swept is determined by the sparse 3D reconstruction obtained during the camera calibration process. To reflect the nature of perspective projection, the spacing of the sweeping plane positions is proportional to depth (i.e. at smaller depths the spacing is smaller than at larger depths). The spacing between the first two positions is set to correspond to at most one pixel shift in each of the views. Every other position of the sweeping plane is set to yield one pixel larger shift. In the case of lateral camera motion of constant speed (which is approximately assumed), these shifts correspond to linear disparities.

All the matching costs $c_{\mathbf{p}}$ are stored in the *matching cost volume* $M(\mathbf{p}, d)$, where d is disparity which in fact encodes depth levels of the sweeping plane. This volume is also known as the *disparity space image* (DSI) [29].

Slices of the matching cost volume at several disparities d can be seen in Figure 3.4. We currently calculate the matching costs only in regions in which a corresponding pixel is available in all views v and thus the side borders are usually undefined.

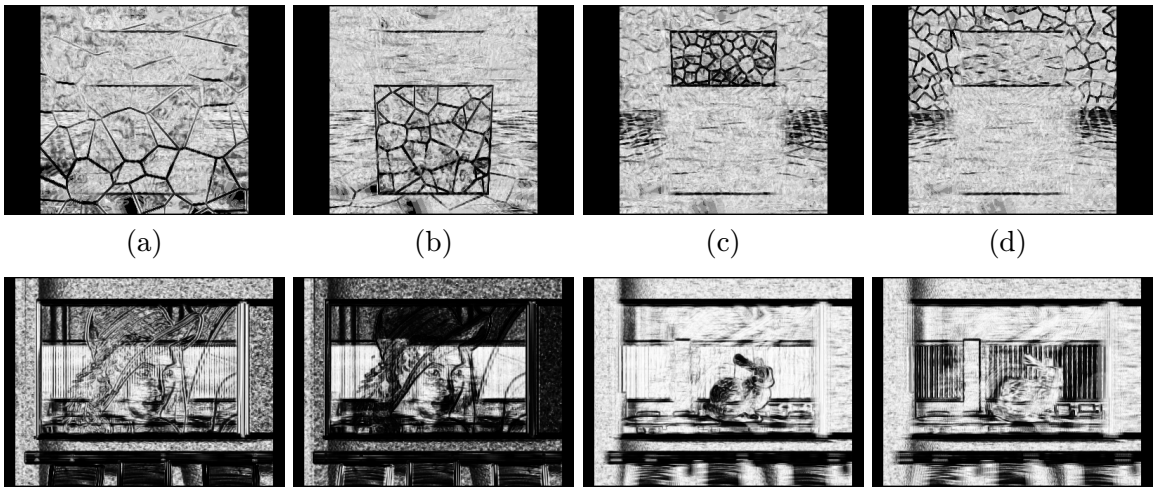


Figure 3.4: *Slices of the matching cost volume at disparities corresponding to depth of distinctive scene components of the BLOCKS* **(a-d)** *and the CONFERENCE sequence* **(e-h)**. (The darker the intensity, the lower the matching cost, i.e. the higher the photoconsistency.)

3.2.2 Semi-Global Aggregation

The problem of matching cost aggregation can be formulated as minimization of a global energy function $E(\mathbf{d})$ of disparity image $\mathbf{d}$. We want the energy to express a smoothness constraint while allowing depth discontinuities at significant intensity gradients. This can be formulated as:

$$E(\mathbf{d}) = \sum_{\mathbf{p}} \left(M(\mathbf{p}, d_{\mathbf{p}}) + \sum_{\mathbf{n} \in \mathcal{N}_{\mathbf{p}}} P_1 T[|d_{\mathbf{p}} - d_{\mathbf{n}}| = 1] + \sum_{\mathbf{n} \in N_{\mathbf{p}}} P_2 T[|d_{\mathbf{p}} - d_{\mathbf{n}}| > 1] \right), \quad (3.3)$$

where $M(\mathbf{p}, d_\mathbf{p})$ is the matching cost for pixel $\mathbf{p} = (x, y)$ and disparity $d_\mathbf{p}$. $T[]$ is a test operator which is 1 if its argument is true and 0 otherwise. The second term adds a constant penalty P_1 for all pixels $\mathbf{n}$ in the neighbourhood $\mathcal{N}_\mathbf{p}$ of pixel $\mathbf{p}$ for which the disparity changes a little bit (i.e. 1 pixel). Using a lower penalty for small changes allows an adaptation to slanted or curved surfaces. The last term then adds a larger penalty P_2 for all larger disparity changes. P_2 is inversely proportional to the magnitude of intensity gradient to favour depth discontinuities at strong intensity edges in the image. However, it has to be always ensured that $P_2 \geq P_1$.

Unfortunately, this 2D global minimization is NP-complete [15]. Moreover, it produces only one depth estimate per pixel although we want to estimate two depths for pixels in reflective regions.

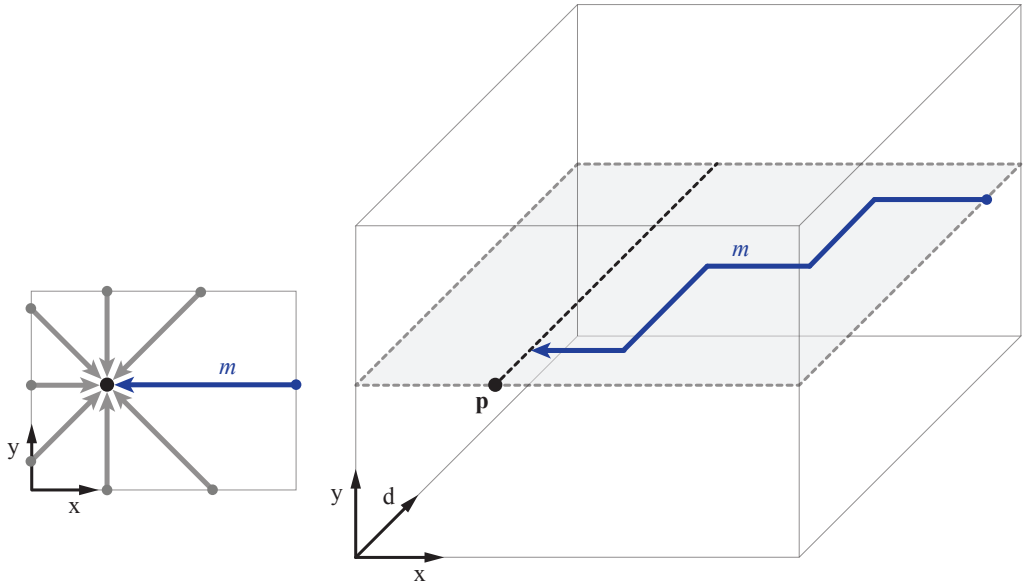


Figure 3.5: *A minimum cost path m goes through the minimum costs from the border of the matching cost volume to disparity d of pixel $\mathbf{p}$ while taking into account penalties for disparity changes.*

Possible solution is to use the semi-global approach presented by Hirschmüller [15] which is known to be computationally less demanding but providing similar accuracy as global approaches.

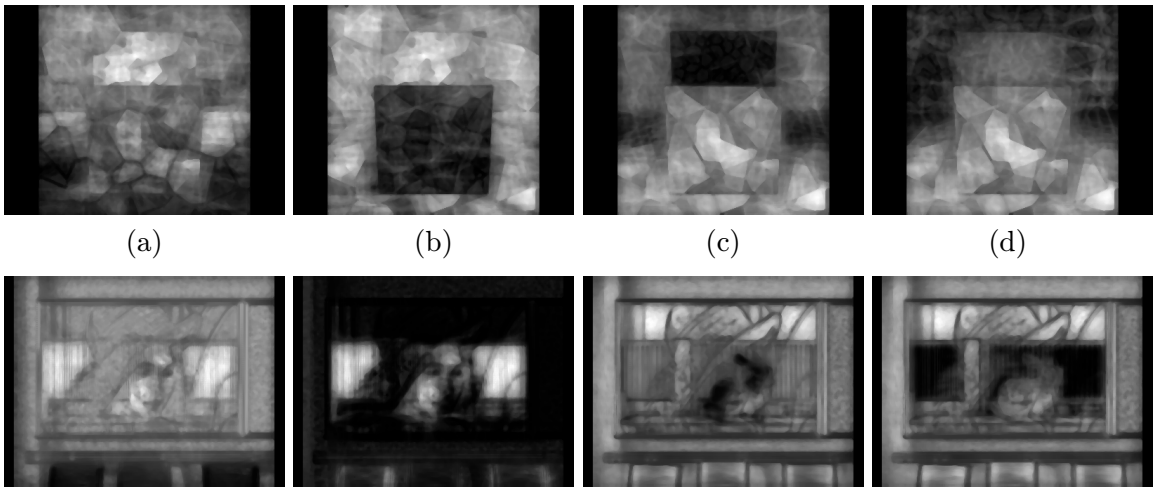
The aggregated cost $S(\mathbf{p}, d)$ of pixel $\mathbf{p}$ at disparity d is calculated by summing the costs of several *1D minimum cost paths* that end at coordinates $(\mathbf{p}, d)$ in the matching cost volume M (Figure 3.5):

$$S(\mathbf{p}, d) = \sum_\mathbf{r} L_\mathbf{r}(\mathbf{p}, d), \tag{3.4}$$

where $L_\mathbf{r}(\mathbf{p}, d)$ is the cost of a 1D minimum cost path traversed in direction $\mathbf{r}$. It is defined recursively as:

$$L_{\mathbf{r}}(\mathbf{p}, d) = M(\mathbf{p}, d) + \min(L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d), \\ L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d-1) + P_1, \\ L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d+1) + P_1, \\ \min_i L_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, i) + P_2), \tag{3.5}$$

where $M(\mathbf{p}, d)$ is the matching cost computed in Section 3.2.1 to which the minimum cost of the previous pixel $\mathbf{p} - \mathbf{r}$ of the path is added, including the appropriate penalty for discontinuities. This actually implements the behavior of (3.3) along an arbitrary 1D path. In practise, 8 or 16 path directions $\mathbf{r}$ are considered.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 3.6: *Slices of the aggregated cost volume S at disparities corresponding to depth of distinctive scene components of the BLOCKS* **(a-d)** *and the CONFERENCE sequence* **(e-h)**. (Compare these results with Figure 3.4 to see the effect of aggregation.)

3.2.3 Two-Layer Depth Determination

For each pixel $\mathbf{p}$, its 1D distribution (over all disparities d) of aggregated costs $S(\mathbf{p}, d)$ is analyzed and, whenever possible, two disparity estimates per pixel are determined.

In particular, a set of local minima d_i^* is at first enumerated such that $S(\mathbf{p}, d_i^*) < S(\mathbf{p}, d)$ for all adjacent disparities where $d_i^* - w_1 \leq d \leq d_i^* + w_1$. The parameter w_1 controls size of the considered neighbourhood (usually set to 2 or 3). Once the minima are detected, sub-pixel refinement is performed by taking the minimum of a quadratic function fitted to cost values in the considered neighbourhood.

The global minimum is selected as the primary disparity d_p of pixel $\mathbf{p}$. To select the secondary disparity d_s, all minima within an interval $d_p \pm w_2$ are at first discarded. Afterwards, if at most M local minima remain, the smaller one is taken as the secondary disparity. If the number of remaining minima is larger than M, the secondary disparity is not determined as it is likely to be unreliable. The parameters w_2 and M are typically se to 5 and 2 respectively.

3.3 Approximation by Piecewise-Planar 3D Proxies

It is often the case that one of the layers, usually the reflection one, is less apparent. The aggregation of matching costs is particularly important for depth estimation of this weaker layer because its gradients are attenuated and thus their evidence in the matching cost volume is not so obvious. However, even after the aggregation step the per-pixel depth estimates of the weaker layer are often noisy. Moreover, the estimates can be sparse because the weaker layer might be noticeable only in isolated regions.

One way to refine the two-layer depth map is to represent each depth layer by piecewise planar 3D proxies[2] which leads to a more stable and dense representation of the weaker layer. The price for this step is a loss of details which are likely to be noisy anyway.

At the beginning, the primary and secondary disparities estimated in the previous section are converted to a set of 3D points. This is done by back-projection of 2D image points to depths which are given by the estimated disparities.[3]

To compute the piecewise-planar 3D proxies, normal vectors of the 3D points are estimated at first (Section 3.3.1). A set of representative scene planes is then extracted by clustering of the 3D points on the basis of coplanarity (Section 3.3.2). In the last step, a pixel labeling is computed such that each pixel is assigned to maximum of two scene planes (Section 3.3.3) which forms the resulting 3D proxies.

3.3.1 Surfels Fitting

Normal vectors of the 3D points are estimated by local plane-fitting. For each 3D point $\mathbf{P}_i = (X_i, Y_i, Z_i)$, at most two planes are fitted to its adjacent 3D points (from both layers) whose projection falls into a $\vartheta \times \vartheta$ pixel neighbourhood of the projection of $\mathbf{P}_i$ (ϑ is usually set to 7).

Sinha et al. [27] suggested to do the fitting by sequential RANSAC [29]. In our implementation we took slightly different approach based on the observation that the 3D points from different layers have usually large mutual distance. We try to cluster the 3D points into two groups and if these groups are distinct enough, we fit the plane by regular RANSAC to each of the groups separately. If the groups are similar, we fit only one plane to all the 3D points.

Each fitted plane p_i is stored as a *surfel* (i.e. surface element [20]) $\mathbf{s}_i = \{\mathbf{P}_i^s, \mathbf{n}_{p_i}\}$, where $\mathbf{P}_i^s$ is the 3D point where the ray through camera center and $\mathbf{P}_i$ intersects the plane p_i. $\mathbf{n}_{p_i}$ is the normal vector of the fitted plane p_i. To reduce spurious estimates, surfels that lie at a grazing angle to the camera ray are pruned.

Surfels fitted to our synthetic scenes are visualized in Figure 3.7. For the BLOCKS sequence, the whole scene is represented very well. We can clearly see all the surfaces of the scene (i.e. the specular surface in the front, faces of the two blocks, the back wall and also the horizontal floor). In the case of the CONFERENCE sequence, we can also see the specular surface and the back wall and we can notice even the bunny in the middle. Nevertheless, a representation of the horizontal table surface is missing. This is because of the lack of any distinctive texture on the table which caused the failure of the stereo matching.

[2] *Proxy* is used to refer to an approximate geometric information.

[3] By back-projecting a 2D point $\mathbf{p} = (x, y)$ we get a 3D line l going through the camera center and the 3D point $\mathbf{P}' = (x, y, f)$ which lies at depth equal to the camera focal length f (considering the pinhole camera model). Knowing the depth estimate Z of the point $\mathbf{p}$ (given by the estimated disparity), we can obtain its corresponding 3D point $\mathbf{P} = (X, Y, Z)$ by taking the point on the line l at depth Z [4].

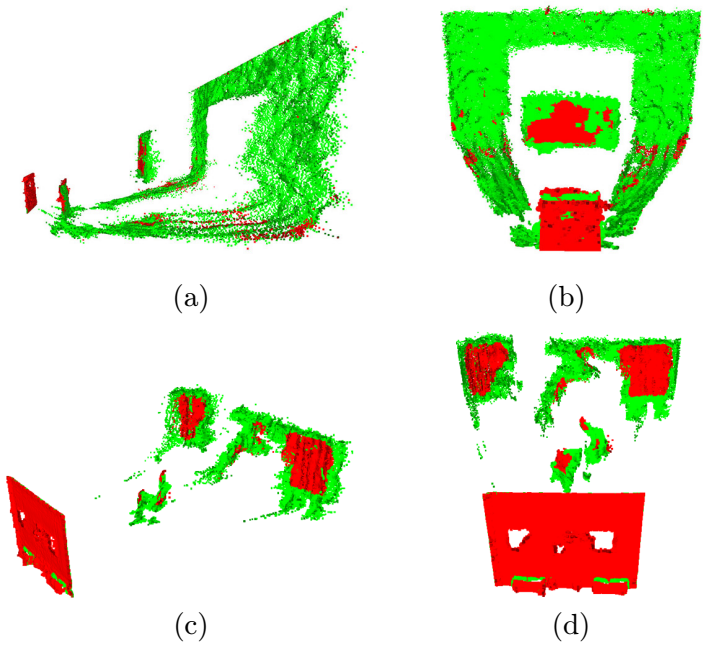


Figure 3.7: *Visualization of the fitted surfels for the BLOCKS* **(a,b)** *and the CONFERENCE sequence* **(c,d)**. The shading is based on the estimated normal vectors. For the most of pixels on the reference image plane, there is at least one estimated surfel (red color). If there are two surfels, then the front one is red and the rear one is green.

Note also that the front surfels and rear surfels (in Figure 3.7, they are red and green respectively) are mixed together and thus do not reflect the real layers. This problem is addressed in the next step where clustering of coplanar surfels is performed.

3.3.2 Plane Extraction from Clusters of Surfels

The representative scene planes $\Pi = \{\pi_i\}$ are extracted by seed-and-grow surfel clustering on the basis of coplanarity. This is motivated by [9] and is conceptually similar to k-means clustering. The clustering aims to minimize the total approximation error $\sum_{i,j} f(s_j, \pi_i)$, where $f(s, \pi) = |Z - Z_\pi|/Z$ measures the error caused by assigning surfel s to plane π. Z is depth of the surfel point $\mathbf{P}^s$ and Z_π denotes depth of the 3D point at which the ray going through camera center and point $\mathbf{P}^s$ intersects the plane π.

On convergence, average surfels of the resulting clusters with at least m surfels in each of them (m is set to 32 by default) are selected as the representative scene planes. Figure 3.8 visualizes the clustering results.

3.3.3 Graph Cut Optimization

Once we have extracted the representative scene planes, the goal is now to assign each pixel to one or two of these planes. In other words, we want to determine extent of the extracted planes in the final piecewise-planar depth map. This problem can be solved by a graph cut optimization of a multi-label Markov random field (MRF). The MRF is formed as a graph with nodes $\mathcal{P}$ which represent image pixels, and edges $\mathcal{N}$ which connect each node with its 4-neighbourhood.

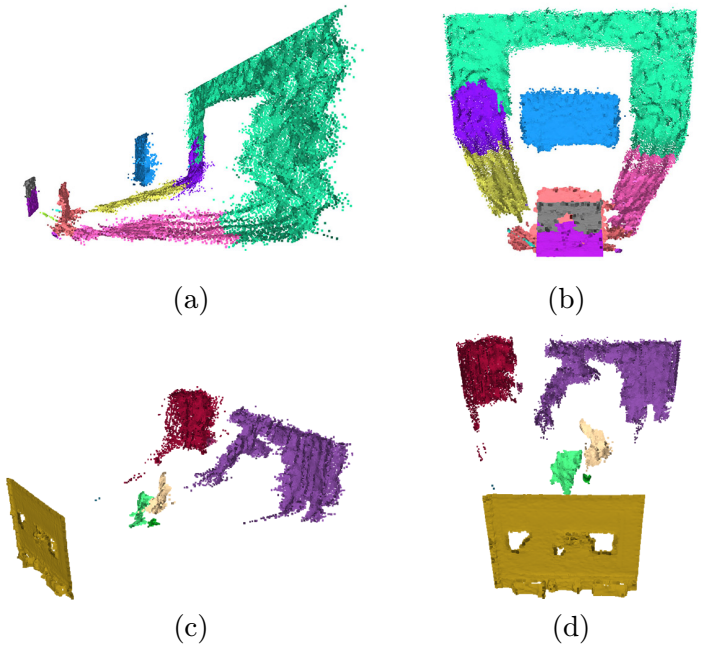


Figure 3.8: *Visualization of the surfel clusters for the BLOCKS* **(a,b)** *and the CONFERENCE sequence* **(c,d)**.

The set of labels $\{1, \ldots, (M_1 + M_2)\}$ consists of labels for M_1 extracted individual planes from the set Π and labels for M_2 pairs of overlapping planes denoted as $Q = \{(\pi_u, \pi_v)\}$, where $u, v \in (1 \ldots M_1)$. Label l represents an individual plane when $1 \leq l_p \leq M_1$ and a pair of overlapping planes when $M_1 < l_p \leq (M_1 + M_2)$.

The resulting labeling L is found by minimization of this energy:

$$E(L) = \sum_{p \in \mathcal{P}} E_p(l_p) + \sum_{(p,q) \in \mathcal{N}} E_{pq}(l_p, l_q). \tag{3.6}$$

where $E_p(l_p)$ is the data term which measures the penalty of assigning pixel p to label l_p based on how well the corresponding plane(s) approximate the estimated depth(s) at that pixel. The smoothness term $E_{pq}(l_p, l_q)$ measures the penalty of assigning neighbouring pixels p and q to labels l_p and l_q respectively.

The approximation to the Maximum a Posteriori (MAP) labeling can be obtained using the α-expansion algorithm [3]. More details about this graph cut optimization, including definition of the data and the smoothness term, can be found in [27].

The resulting two-layer piecewise-planar 3D proxies produced by our implementation are visualized in Figure 3.9.

3.3.4 Reflection Detection

The reflectivity field can be easily extracted from the labeling L obtained by the graph cut optimization. If a pixel has been assigned to a pair of planes, it is classified as a reflective pixel and its β value is set to 1. Otherwise the pixel is opaque and its β value is set to 0.

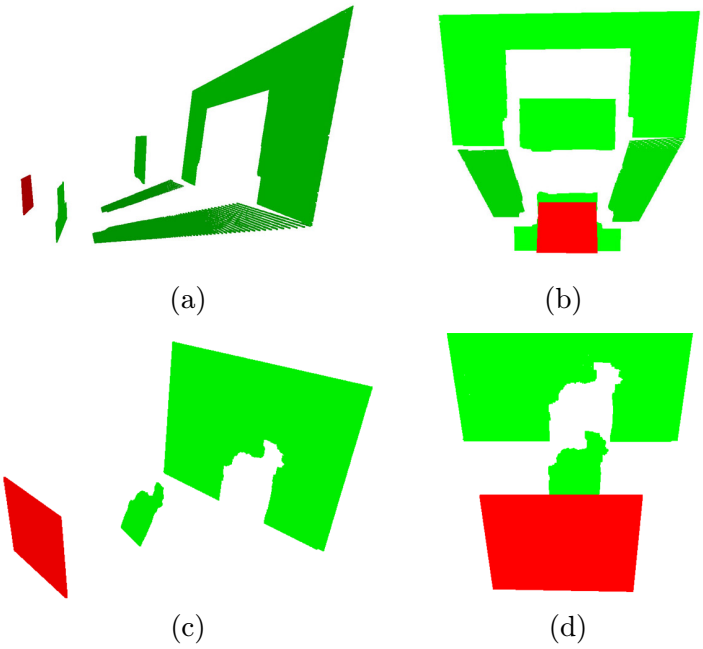


Figure 3.9: *Visualization of the final piecewise-planar 3D proxies for the BLOCKS* **(a,b)** *and the CONFERENCE sequence* **(c,d)**. The red and the green proxies represent final depth estimations of the front and the rear layer respectively.

3.4 Results of Two-Layer Depth Estimation

Depth maps of the front and the rear layer given by the estimated two-layer piecewise-planar 3D proxies can be seen in Figure 3.10. Detected reflectivity fields, the ground truth for the synthetic sequences and results by Sinha et al. [27] for the GUITARIST sequence are presented as well.

It is possible to distinguish the most of the dominant scene elements in the estimated depth maps but the finer details (such as the ears of the bunny) are not captured very well. This is caused by an effort of the graph cut optimization to produce compact proxies by applying the smoothness constraint. This effort is beneficial when closing holes in the depth estimates (notice for example that the front plane holes visible in Figure 3.8(d) disappeared in Figure 3.9(d)). On the other hand, the same effort reduces the finer details.

Despite the lack of details, this depth representation can describe the layer motion between different views obviously better than homography which is often used [30, 13]. The benefit can be seen later in Section 3.1.

In the case of the GUITARIST sequence, we have not achieved the quality of the depth estimation presented by Sinha et al. [27]. We suppose that this could be caused by the worse performance of the camera calibration system we used (as was discussed in Section 3.1). For the other real sequences, which were obtained by rather irregularly moving camera (i.e. the MUZEUM and the STATUE sequence), the camera motion estimates were highly incorrect and could not be used to produce any presentable results.

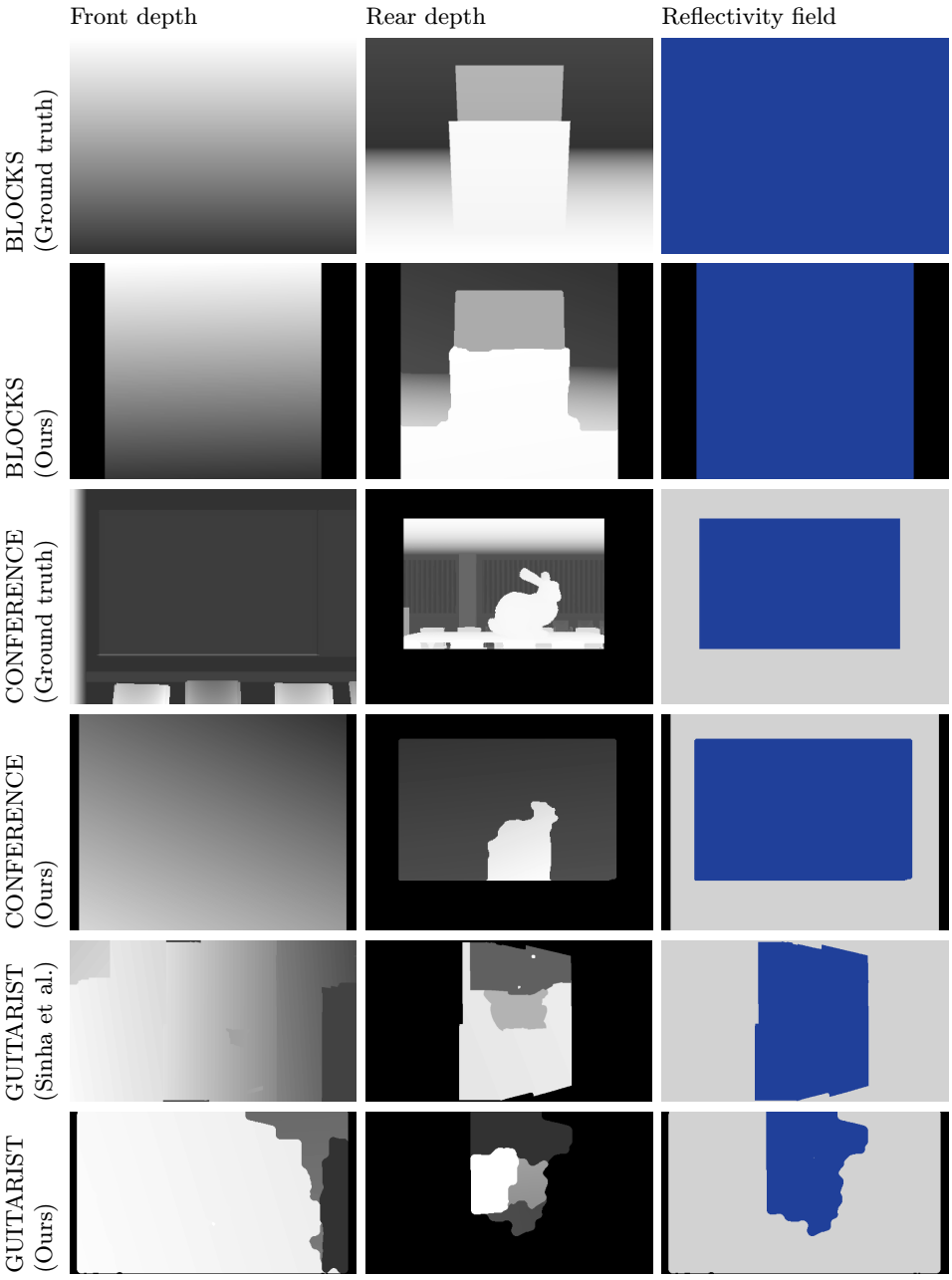
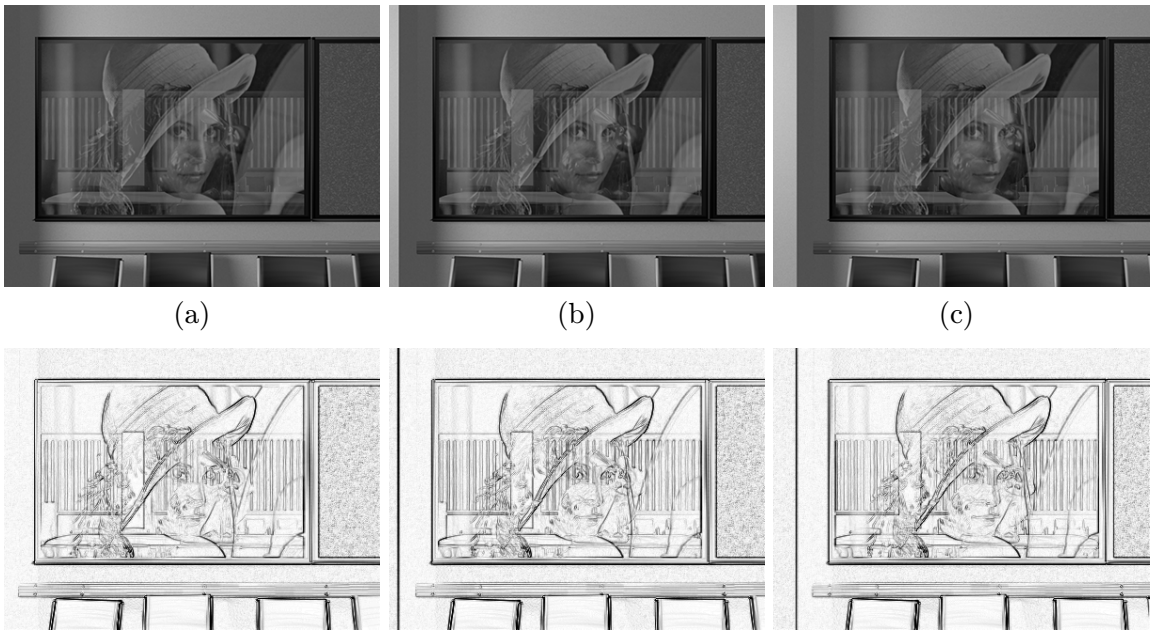


Figure 3.10: *Results of the two-layer depth estimation.* In the depth maps, brighter color means smaller depth while black color represents undefined values. In the right column, blue color represents the detected reflectivity fields (i.e. the regions where two depths were estimated).

Chapter 4

Layer Gradients Estimation

For human eyes, the significant image gradients (i.e. edges) are an important clue to distinguish between relatively moving layers. This is demonstrated by Figure 4.1 where we are able to observe the two layers from the grayscale images as well as from their corresponding gradient images. At the same time, our brain is able to distinguish motion and structure of both layers. The image gradients contain sufficient information even to assign the problematic regions to the correct layer (such as the wooden frame in the reflection layer discussed in Section 2.3). Apparently, the fundamental key to this ability is an understanding of the scene. But is it possible to accomplish the assignment of the problematic regions also using only local features?



(a) (b) (c)

(d) (e) (f)

Figure 4.1: *Grayscale images and their corresponding gradients.*

In our approach, we detect problematic gradients at first (i.e. gradients with no apparent motion) and cluster them into groups representing individual problematic edges (Section 4.3). Gradients with apparent motion are then separated into respective layers

using knowledge about their depth (Section 4.4). As the last step, each problematic edge is assigned to the layer where it has better topological fitness which is measured locally (Section 4.5).

4.1 Gradient Approximation

Image gradient $\nabla C(x,y) = G(x,y) = \big(G_x(x,y), G_y(x,y)\big)$ is a vector that points in the direction of largest possible intensity increase of an image $C(x,y)$. As its approximation, we use differences between horizontal and vertical neighbours[1], such as:

$$G_x = C(x+1,y) - C(x,y), \tag{4.1}$$
$$G_y = C(x,y+1) - C(x,y). \tag{4.2}$$

The gradient magnitude can be then calculated as follows:

$$G(x,y) = \sqrt{G_x^2 + G_y^2}. \tag{4.3}$$

This simple approximation of image gradient is used in order to simplify the color recovery process (Section 5.1) which utilizes equations (4.1) and (4.2).

4.2 Gradient Types

In the process of layer gradients estimation we distinguish between two types of image gradients:

1. **Gradients with apparent motion**
   Their motion in the gradient field is apparent and thus their depth can be determined.
2. **Gradients with no apparent motion (also called *problematic* gradients)**
   They have no apparent motion in the gradient field and thus their depth cannot be reliably determined. These gradients usually form edges of the problematic regions and require a special care.

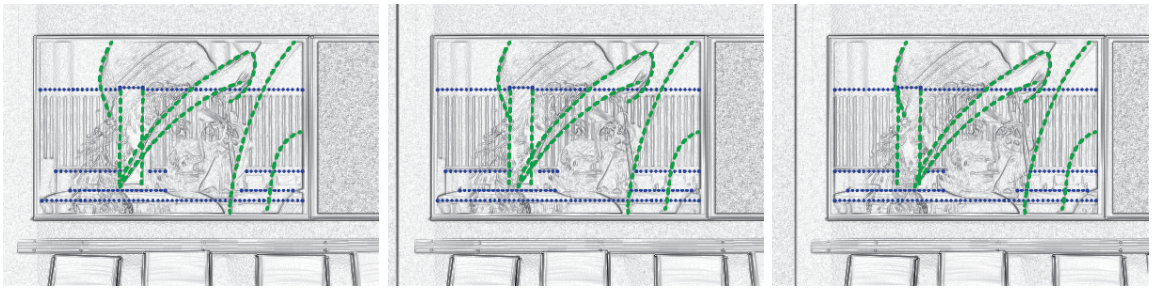
Examples of both gradient types can be seen in Figure 4.2.

4.3 Detection of Problematic Edges

4.3.1 Probability of Problematic Gradient

To detect the problematic gradients, we exploit the fact that it has low matching cost at the most of considered disparities (we can get these costs from the matching cost volume $M(x,y,d)$ described in Section 3.2.1). However, this condition alone is not sufficient because it is satisfied also in textureless regions. For this reason we considered also gradient magnitudes.

[1] In practise, the gradient components G_x and G_y are computed by a convolution with the horizontal and vertical difference filters ($[0,-1,1]$ and $[0,-1,1]^\top$).



(a) (b) (c)

Figure 4.2: *Examples of the gradient types in three consecutive frames (taken by laterally moving camera).* Some of the gradients with apparent motion are highlighted with green dashed line and some of the problematic gradients with blue dotted line.

The probability that a gradient at pixel (x, y) is problematic is defined as follows:

$$P(x, y) = w(x, y) \left(1 - \tilde{M}(x, y)\right), \tag{4.4}$$

where $\tilde{M}(x, y)$ is median of the matching costs $M(x, y, d) \in [0, 1]$ for all disparities d and pixel (x, y). The weight $w(x, y)$ is applied to attenuate the probability at less significant gradients. It is expressed as:

$$w(x, y) = (1 - \alpha) + \alpha \left|\mathbf{r} \cdot \tilde{G}(x, y)\right|, \tag{4.5}$$

where $\tilde{G}(x, y)$ is the median gradient (calculated separately for its x and y component) for pixel (x, y) in all input images. $\mathbf{r}$ is a unit vector perpendicular to the dominant direction of camera motion.[2] Coefficient $\alpha \in [0, 1]$ controls strength of the attenuation and the role of the term $(1 - \alpha)$ is to keep the weight in the range $[0, 1]$. We typically set α to 0.5.

One possible way how to get the dominant direction of camera motion is to use principal components analysis (PCA). Note that if the camera motion has no dominant direction, the depth of all edges is likely to be deterministic and thus the problematic probability P will be very low for all pixels.

We used median of gradients in (4.5) to avoid gradients with apparent motion. This is based on an observation that the temporal derivation of problematic gradients is close to zero (i.e. the gradient at pixel (x, y) is very similar in all input images) which does not hold for the gradients with apparent motion. For similar reason we also used median of the matching costs in (4.4).

It can be also observed that the direction of edges formed by problematic gradients is the same as the dominant direction of camera motion. We use this observation and project the gradients to the direction $\mathbf{r}$ which is perpendicular to the dominant camera motion. This helps to further reduce spurious gradients.

Note that the weight $w(x, y)$ alone would not be enough to determine the probability that a gradient is problematic because it is high also for edges of objects at larger depths.

[2] Absolute value of the dot product of the median gradient and the unit vector $\mathbf{r}$ gives us the magnitude of projection of the median gradient to the direction determined by $\mathbf{r}$.

Gradients at these edges have no apparent motion too, but if their direction is not the same as the dominant direction of camera motion, their depth is deterministic. For this reason we need to use the median $\tilde{M}(x, y)$ of the matching costs which can distinguish these gradients.

Examples of the calculated problematic probability $P(x, y)$ are in Figure 4.3(a-c).

4.3.2 Clustering into Groups Representing Edges

To be able to measure the topological fitness (Section 4.5), we need to cluster the problematic gradients into groups representing individual problematic edges.

The clustering is done by tracing edges formed by gradients with high problematic probability (4.4). To achieve this, we again use the observation that the direction of problematic gradients is perpendicular to the dominant direction of camera motion. It is applied by a special type of thresholding with hysteresis where we use a lower threshold t_1 to expand the cluster in the dominant direction and a higher threshold t_2 to expand the cluster in the perpendicular direction (we typically set $t_1 \approx 0.1$ and $t_2 \approx 0.4$). The expanding is started at a gradient with $P(x, y) \geq t_2$.

The problematic gradients are then approximated as:

$$G_P(x, y) = f(x, y) \left(\mathbf{r} \cdot \tilde{G}(x, y) \right) \mathbf{r}, \tag{4.6}$$

where $f(x, y)$ serves to filter out gradients which are not part of any problematic edge (i.e. $f(x, y) = 1$ if pixel (x, y) is part of any problematic edge and $f(x, y) = 0$ otherwise). The rest of the equation project the median gradient to the direction given by the unit vector $\mathbf{r}$. The intuition behind this equation is the same as for (4.5).

Detected problematic edges for several sample sequences are illustrated in Figure 4.3(d-f). Notice that some of the detected gradients in Figure 4.3(f) form edge segments which are not in the dominant direction of camera motion. But since these segments were expanded from some real problematic segments, this is not a problem because the topology is not violated.

4.4 Separation of Gradients with Apparent Motion

4.4.1 Initial Separation

The gradients with apparent motion are now defined as all gradients which are not part of any detected problematic edge. To separate them into respective layers we use the fact that their depth can be reliably estimated. For this estimation we use the matching cost volume $M(x, y, d)$ described in Section 3.2.1. It is calculated by normalized cross correlation (NCC) which performs similarly to matching gradient images and thus is very convenient for our purpose.

As was statistically confirmed by Gai et al.[13], image gradients of natural images are very sparse. The sparsity denotes that most gradients are approximately equal to zero and only a small part of them are significantly different from zero. In our case, this implies that each gradient is likely to reflect an intensity change in only one layer. This is violated only at intersections of edges from different layers.

To handle the situation at intersections we do not assign each gradient exclusively to only one layer. Instead we find the minimum matching cost for each layer and separate the

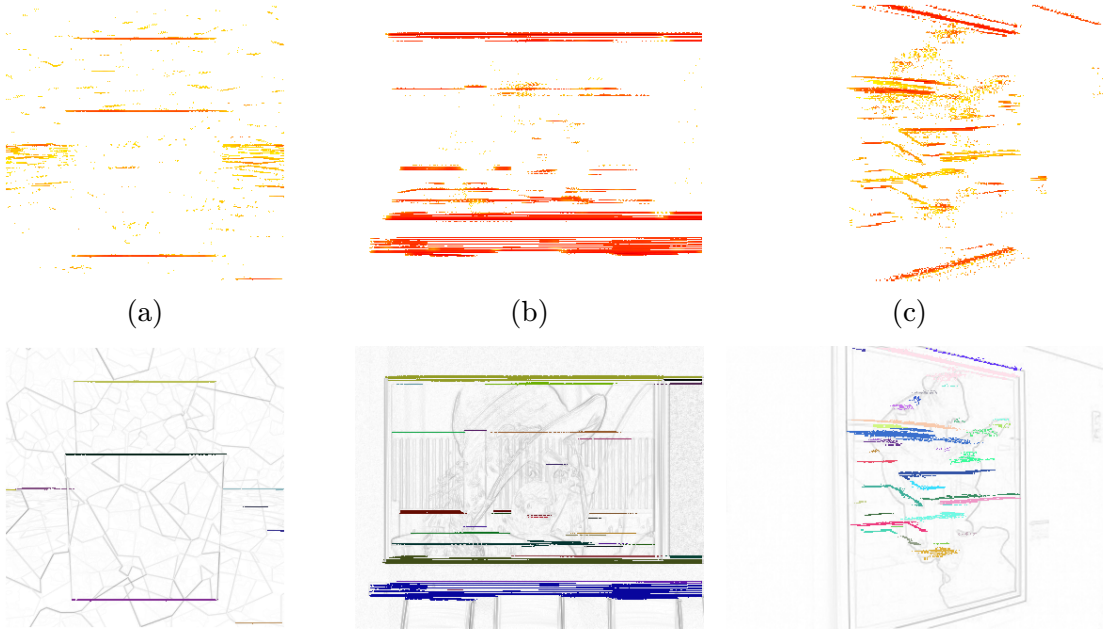


Figure 4.3: *Detection of problematic gradients (the input image sequences were obtained by laterally moving camera).* **(a-c)** Probabilities that gradients are problematic (the more reddish color, the higher is the probability). **(d-f)** Detected problematic edges (each edge has different color.

gradient proportionally to these costs. The gradient portions assigned to the front (G_0) and the rear (G_1) layer are given by:

$$G_0(x,y) = w_0(x,y)G(x,y), \tag{4.7}$$
$$G_1(x,y) = w_1(x,y)G(x,y), \tag{4.8}$$

where $w_0(x,y)$ and $w_1(x,y)$ are weights determining the gradient portions and are expressed as:

$$w_0(x,y) = 1 - \frac{M_{\text{min},0}}{(M_{\text{min},0} + M_{\text{min},1})}, \tag{4.9}$$
$$w_1(x,y) = 1 - \frac{M_{\text{min},1}}{(M_{\text{min},0} + M_{\text{min},1})}, \tag{4.10}$$

where $M_{\text{min},l}$ is the minimum of the matching cost volume $M(x,y,d)$ for the given pixel (x,y) and the set of disparities corresponding to the layer l.

The sets $D_0(x,y)$ and $D_1(x,y)$ of disparities related to the front and the rear layer are defined as:

$$D_0(x,y) = \left\{ d \;\middle|\; |d - d_0| \leq |d - d_1| \right\}, \tag{4.11}$$
$$D_1(x,y) = \left\{ d \;\middle|\; |d - d_0| > |d - d_1| \right\}, \tag{4.12}$$
$$\tag{4.13}$$

where d_0 and d_1 are disparities corresponding to depths (they are inversely proportional) given by the 3D proxies of the layers.

4.4.2 Growing Over Problematic Intersections

The intersections of edges with apparent motion have been handled in the previous section. But we also need to deal with the situation when edges with apparent motion intersect the problematic edges. Since the gradients belonging to problematic edges were not assigned to any layer yet, the edges with apparent motion are interrupted. The situation is illustrated in Figure 4.4.

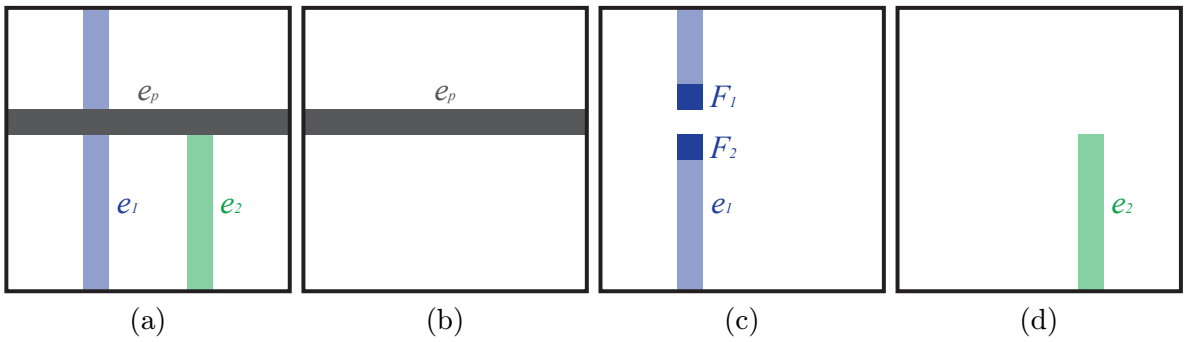


Figure 4.4: *Interruption at intersections with problematic edges.* After detection of the problematic edge e_p **(b)** and separation of the edges e_1 and e_2 into the two layers **(c,d)**, the edge e_1 is interrupted at the position of former intersection with e_p. To correct the topology we need to join the spurious endpoints F_1 and F_2.

To join the spurious endpoints we perform iterative growing on the basis of gradient magnitude similarity. The growing is performed on problematic free image gradients which are given by:

$$G'(x,y) = G(x,y) - G_P(x,y), \tag{4.14}$$

where the problematic gradients G_P are subtracted from the original gradients of the reference image.

The problematic edges disappeared in G' while the other edges were preserved even at intersections with the problematic edges. Justification of this consequence can be based on findings by Rothwell et al. [21] who showed that gradient magnitude near junctions can be considered to be reliable because it reflects well the real jumps in image intensity (unlike gradient orientation which is not always reliable at junctions). Since we assume the additive mixing model (2.1), we can also assume that gradient magnitude at intersection point is a sum of gradient magnitudes of both intersecting edges. Hence, after subtraction of the problematic gradients we should get a good estimate of gradients forming the intersecting edge.

In each iteration of the growing, we go through gradients $G'(x,y)$ at pixels belonging to a problematic edge and if we find a layer with gradient of similar magnitude in the 8-neighbourhood, we assign the gradient at the current pixel to that layer. The growing process continues until the gradients are stable. We denote the refined layer gradients as G''_0 and G''_1.

Currently we rely on the earlier mentioned sparsity of gradients and do not consider the case where edges from different layers intersect a problematic edge at the same point. However, this could be solved by proportional separation of the gradient between both layers (the portions could be given by magnitudes of the similar gradients found in the neighbourhood).

4.5 Separation of Problematic Gradients

4.5.1 Essential Assumptions

The remaining task is to separate the problematic edges into the layers. We assign each problematic edge to that layer where it leads to better *topological fitness* with the already assigned gradients. To measure the topological fitness we use two essential assumptions:

1. **Every problematic edge is a part of some more complex contour.**
2. **An intersection is more likely caused by edges from different layers.**

The first assumption has several degenerate cases. First of all, if only a part of an object is covered by an image, we do not see its whole contour and thus the problematic edge can be isolated. Another degenerate case, when the problematic edge is isolated, is when a contour of an object collapses into a single edge or when there is some synthetic texture consisted of separated lines.

We applied the second assumption in order to reduce these degenerate cases. The assumption is based on our observation of natural scenes and is demonstrated by Figure 4.5 where it can be seen that the most of the intersections are really caused by edges from different layers.

(a) (b) (c)

Figure 4.5: *Examples of intersections caused be edges from different layers (highlighted with green dots).*

4.5.2 Measurement of Topological Fitness

Using the introduced assumptions, the topological fitness induced by the problematic edge e when assigned to the layer l is measured as follows:

$$F(e,l) = \frac{1}{D(G_{0,e,l}) + D(G_{1,e,l}) + \gamma X(e,l)}, \tag{4.15}$$

where $D(G_{i,e,l})$ measures discontinuities of magnitudes of a gradient field. If $i = l$, the gradient field is given by $G_{i,e,l} = G_i'' + G_P$, and by $G_{i,e,l} = G_i''$ otherwise. $X(e, l)$ measures intersections in layer l when the edge e is assigned to this layer. The regularization coefficient γ serves to control the trade-off between the two measures.

A contour of an object is typically given by gradients without any big jumps in their magnitudes along the contour. Hence, a correctly assigned edge should yield lower discontinuity because it completes a contour and thus eliminates edge endpoints (Figure 4.6). In order to attain the lowest discontinuity in front and rear layer simultaneously, (4.15) reflects the situation in both of them.

Each edge e is assigned to the layer l where it leads to higher topological fitness $F(e, l)$. Figure 4.7 shows and discusses possible assignments in a sample situation.

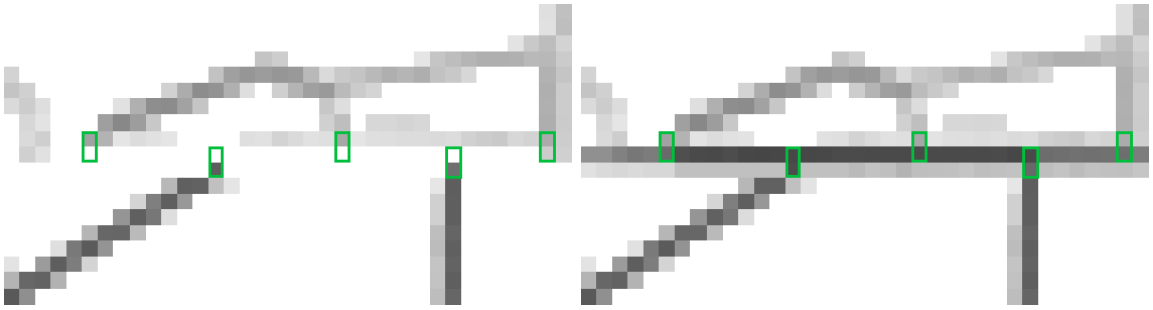


Figure 4.6: *Reduction of gradient magnitude discontinuities.* **(a)** Magnitudes of a gradient field without the problematic edge assigned. **(b)** Magnitudes of the same gradient field when the problematic edge is assigned. Notice the reduction of magnitude jumps between the highlighted neighbours.

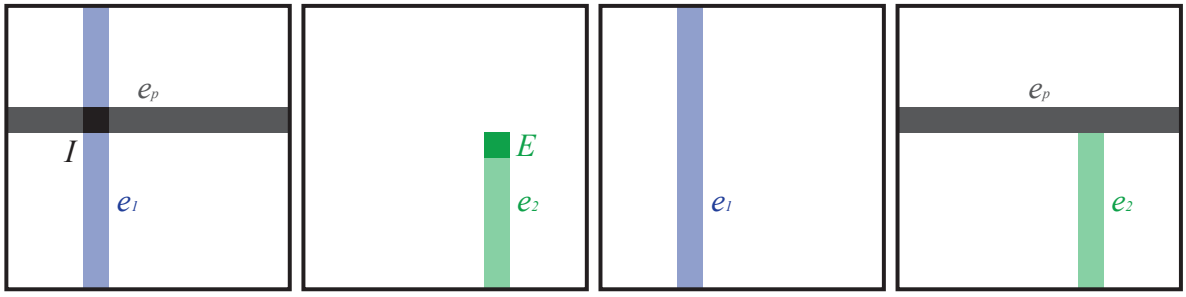


Figure 4.7: *Two possible assignments of a problematic edge* e_p. **(a,b)** Layer gradients with the problematic edge assigned to the first layer. One endpoint E and one intersection I was induced by this assignment. **(c,d)** Layer gradients with the problematic edge assigned to the second layer. This assignment is preferred since it leads to no endpoints (which are sources of additional discontinuity in gradient magnitudes) and no intersections.

Measurement of Gradient Magnitudes Discontinuity

The discontinuity of a gradient field $\hat{G}$ is measured by a sum of gradient magnitudes of magnitudes of $\hat{G}$:

$$D(\hat{G}) = \sum_{(x,y)} \left\| \nabla \left\| \hat{G}(x,y) \right\| \right\|. \tag{4.16}$$

In practise, it could be calculated only in a neighbourhood of the examined problematic edge.

Measurement of Intersections

To measure intersections induced by problematic edge e in layer l, we simply sum up magnitudes of gradients which were assigned to layer l at pixels belonging to the edge e during growing over problematic intersections (Section 4.4.2). It is expressed as:

$$X(e,l) = \sum_{(x,y)\in e} \left\| G''_l(x,y) \right\|. \tag{4.17}$$

4.6 Results of Layer Gradients Estimation

Results of the suggested approach to layer gradients estimation are presented in Figure 4.8 where they can be compared with the ground truth (in the case of synthetic sequences) and the results of the method by Gai et al. [13]. Gradients of the original reference image are shown as well.

Our method produced decent results which do not encounter any significant problems unless the assumption about sparsity of gradients is violated. A problem might arise when the gradients are cluttered, such as in the region where the curtain is covered by the feather on Lena's hat (in the CONFERENCE sequence). The main cause of a potential failure in these regions comes from ambiguity in matching costs which are used to estimate depth of significant gradients.

Regarding the problematic edges, the majority of them was assigned to the correct layer. However, sometimes the detection did not cover the whole edge and some remainders were left in the wrong layer. This is the case of e.g. table edges (again in the CONFERENCE sequence). The main part of these edges was assigned correctly to the rear layer, but their margins were not covered and were wrongly placed to the front layer.

Also notice that the gradient estimation do not contain less apparent details and edges formed by gradually increasing gradients. These elements are comprised of non-significant gradients whose depth cannot be recovered reliably and hence are ignored. Our results have also undefined values at side borders because it uses the depth estimations (Section 3.4) which cover only the region visible in all input images.

From the studied motion-based methods for reflection and transmission separation, the method by Gai et al. [13] is the only one which performs estimation of layer gradients. As was mentioned in Section 2.2.2, it achieves the separation by clustering of correlated gradients. However, they do not take any special treatment of the problematic gradients (these are the uncorrelated gradients in their case). Instead, the problematic gradients are ignored and not assigned to any layer. In our experiments with the method by Gai et

al., this led to a noticeable absence of gradients (especially in the estimations of rear layer gradients). Moreover, the estimated gradients do not usually form continuous edges and thus not preserve topology. For these reasons, our results are evidently of finer quality.

It should be noted that the method by Gai et al. is potentially able to extract more than two layers (it approximates motion of each layer by homography). Hence, it is theoretically able to describe the motion similarly as the piecewise-planar 3D proxies which are estimated in our approach. But the search for plausible homographies is very exhaustive and in our experiments it led to decent results only for the simple BLOCKS sequence where it was able to find motions also of the face of the rear block and the back wall. To be able to find motion also of tilted surfaces (such as the floor), the search space would need to be enlarged and the search process would be even more demanding.

We did not conduct any experiments with image sequences where the mixing coefficients of layers vary from frame to frame. But since the matching cost is calculated by normalized cross correlation (Section 3.2.1), we assume that the layer gradients could be still recovered quite well.

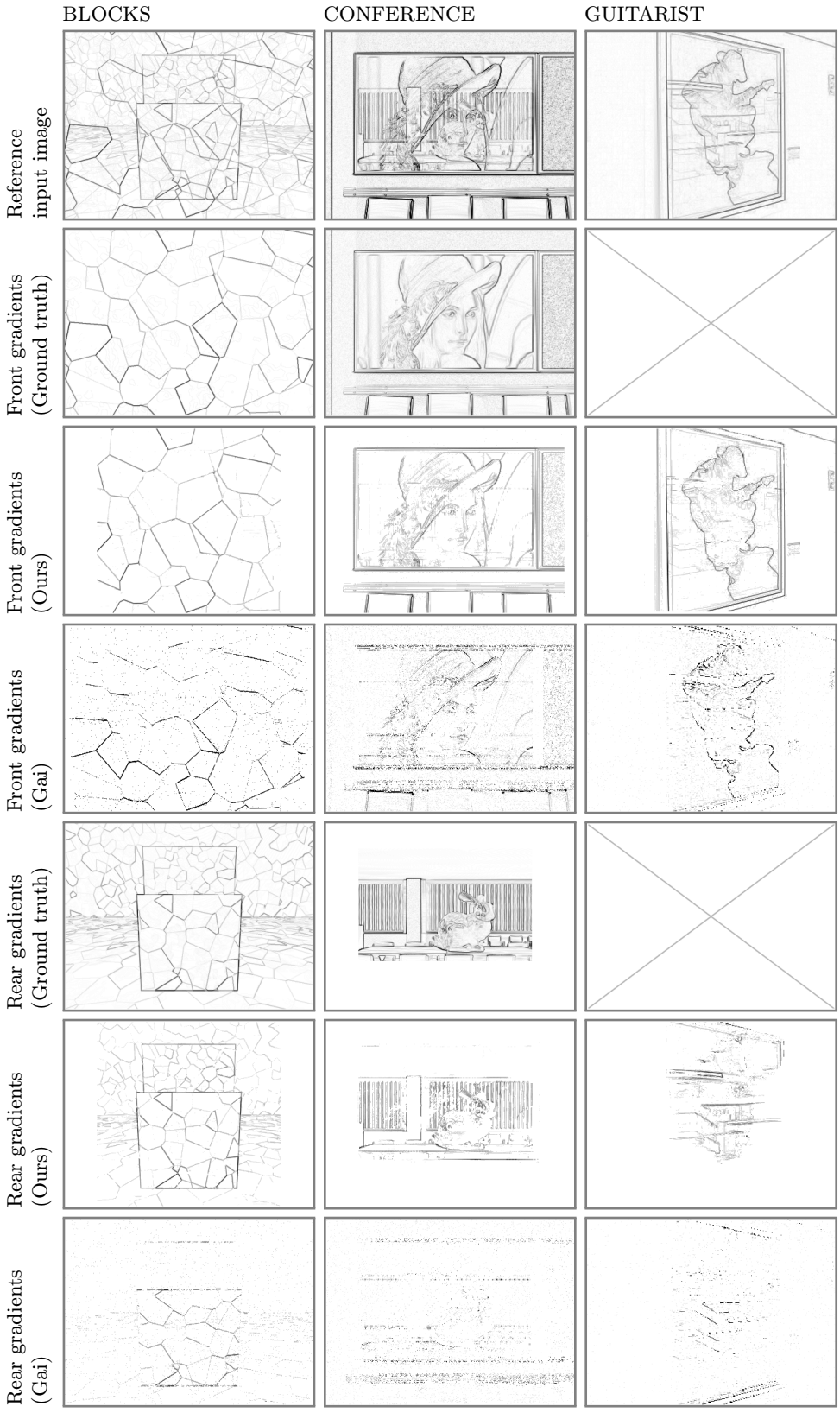


Figure 4.8: *Results of the layer gradients estimation.*

Chapter 5

Layer Colors Recovery

Having the estimated depths and gradients of both layers, we can now also recover their colors by minimization of a proper energy function whose different formulations are reviewed in Section 5.1. Section 5.2 then describes the quadratic programming approach to minimization of this energy and a more efficient alternative approach which we have proposed.

The described recovery process works in fact with a single channel image and thus the most of the results shown in Section 5.3 were calculated using only image intensities. In the case of color images, the process can be performed separately on each channel and the final layer colors can be obtained by merging the individual results. Several results calculated using color images are shown and discussed as well.

5.1 Energy Formulation

The most of the existing methods [30, 32, 27] estimate layer intensities I_0 and I_1 by minimization of an energy which expresses deviation from the mixing model:

$$E(I_0, I_1) = \sum_{p \in \mathcal{P}} \sum_{v \in \mathcal{V}(p)} \sigma\big(C_v(p)\big) \Big(C_v(p) - \widetilde{C}_v(p)\Big)^2, \tag{5.1}$$

where C_v is the input image from view v. $\widetilde{C}_v$ is the image from view v synthesised from the estimated layers I_0 and I_1 using the mixing model which was defined in Section 2.1.3. The role of the function $\sigma(x)$ is to exclude saturated pixels (for 8-bit images, it is equal to 1 when $x < 255$, otherwise it is equal to 0). $\mathcal{P}$ is an *upgradeable set* of pixels in the reference view, such that $\mathcal{P} = \{p | \beta(p) = 1 \wedge \mathcal{V}(p) \neq \emptyset\}$, where β is the reflectivity field and $\mathcal{V}(p)$ is a set of views that contain projections of 3D points from both layers corresponding to pixel p in the reference image.

This energy is usually extended by a regularization term encouraging smoothness which helps to alleviate streaking effects caused by pixels interacting mostly in the direction of motion [30]. The extended energy can have this form:

$$E(I_0, I_1) = \sum_{p \in \mathcal{P}} \left((1-\lambda) \sum_{v \in \mathcal{V}(p)} \sigma\big(C_v(p)\big) \Big(C_v(p) - \widetilde{C}_v(p)\Big)^2 + \lambda \sum_{q \in \mathcal{N}(p), i} \Big(I_i(p) - I_i(q)\Big)^2 \right), \tag{5.2}$$

where $\mathcal{N}(p)$ is a small neighbourhood of pixel p, i is the layer index and λ is a trade-off coefficient between the two terms. This energy is very similar to the one introduced by Tsin et al. [32]. It was only completed by the exclusion of the saturated pixels which was adapted from [13, 27].

However, as was argued by Gai et al. [13] and as also follows from the analysis of problematic regions described in Section 2.3, the mixing model is not sufficient for the color recovery. Motivated by Gai et al., we can formulate the energy to express not only deviation from the mixing model but also from the estimated image gradients:

$$E(I_0, I_1) = \sum_{p \in \mathcal{P}} \left((1-\lambda) \sum_{v \in \mathcal{V}(p)} \sigma\big(C_v(p)\big) \Big(C_v(p) - \widetilde{C}_v(p) \Big)^2 + \lambda \sum_{i,k} \Big| \nabla_k I_i(p) - G_{i,k}(p) \Big|^a \right), \tag{5.3}$$

where ∇_k denotes the k-th component of image gradient (i.e. x or y component) and G_i are the estimated image gradients of layer i. The parameter a, which controls the penalty function for deviation from the estimated image gradients, can be set to 1 or 2, as discussed later.

The required agreement with the estimated image gradients is fundamentally better approach than the general smoothness constraint which is used in (5.2). Agreement with gradients leads to sharper results because it reduces the smoothing force on significant gradients while still encourages the smoothness in regions with small gradient magnitudes. Moreover, if the layer gradients are estimated accurately, the required agreement with gradients helps to handle the problematic regions.

Figure 5.1 demonstrates the benefit of exploiting gradients in the color recovery process (the ground truth of gradients was used in this example). It can be seen that as the trade-off coefficient λ increases (i.e. the agreement with the gradients gets more important), the separation of layers gets better, especially in the problematic regions. When $\lambda = 1$, the separation is nearly identical to the ground truth. When $\lambda = 0$, no agreement with gradients is encouraged and thus the energy to be minimized is identical to (5.1).

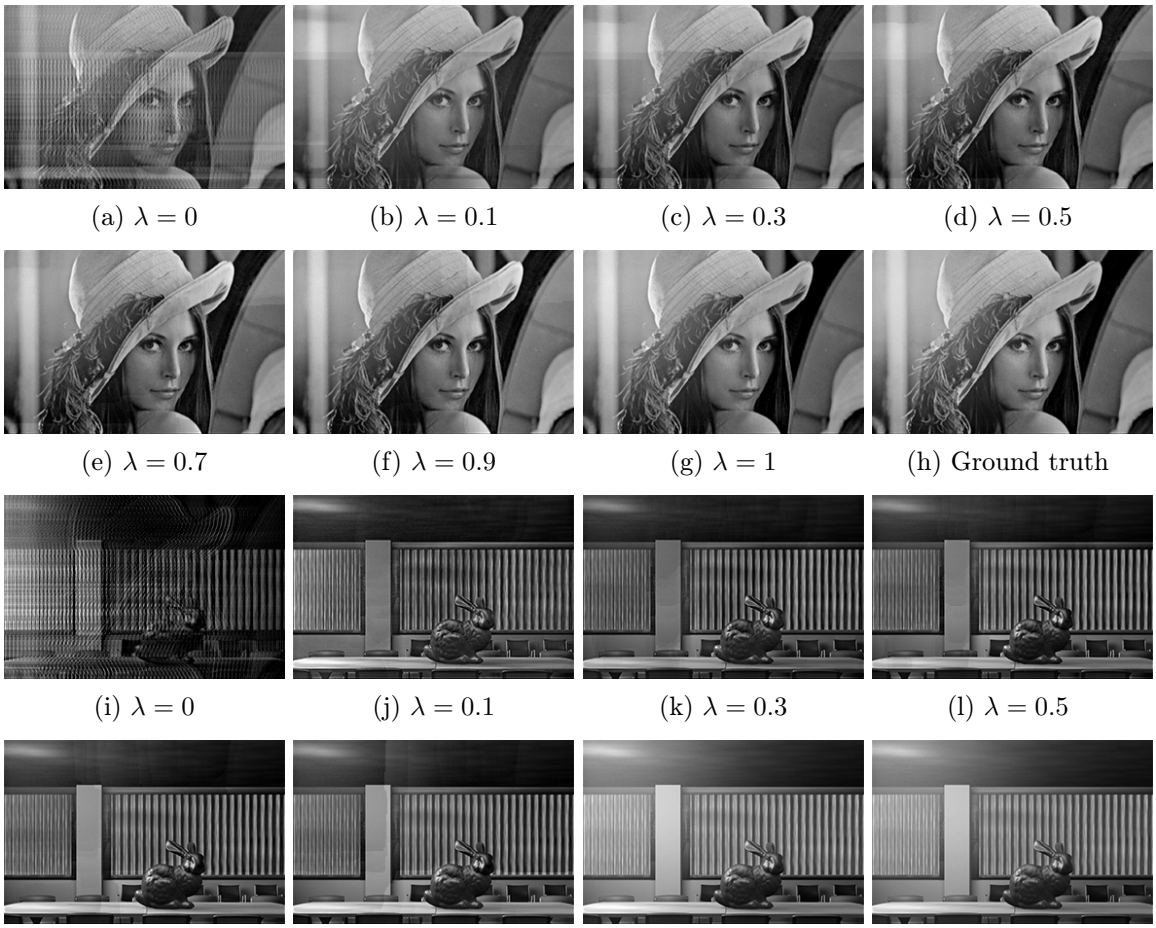
The potential of the approach using layer gradients can be seen also from the example with a real image sequence (Figure 5.2). In this case, layer gradients were obtained by manual labeling and the motion estimation was produced by the method of Gai et al. [13]. Although the motion estimation was not totally perfect, the recovered layer intensities are of high quality (λ was set to 0.9). This confirms that the incorporation of layer gradients is very beneficial in the color recovery process.

5.2 Energy Minimization

5.2.1 Quadratic Programming

Gai et al. [13] formulated the minimization of energy (5.3) as a quadratic programming problem. To measure the gradient difference, they used $\mathcal{L}_1$-norm (i.e. $a = 1$) and argued that it is suitable for all kinds of sparse signals, such as also image gradients.

In the energy function (5.3), the pixel transformation (applied to pixels in order to obtain the synthesized mixtures $\widetilde{C}_v$) is a location to location operation without any pixel value changed. The gradient operation is a combination of the horizontal and vertical difference filters. Outputs of these operations are linear w.r.t. intensities of both layers



(a) $\lambda = 0$ (b) $\lambda = 0.1$ (c) $\lambda = 0.3$ (d) $\lambda = 0.5$

(e) $\lambda = 0.7$ (f) $\lambda = 0.9$ (g) $\lambda = 1$ (h) Ground truth

(i) $\lambda = 0$ (j) $\lambda = 0.1$ (k) $\lambda = 0.3$ (l) $\lambda = 0.5$

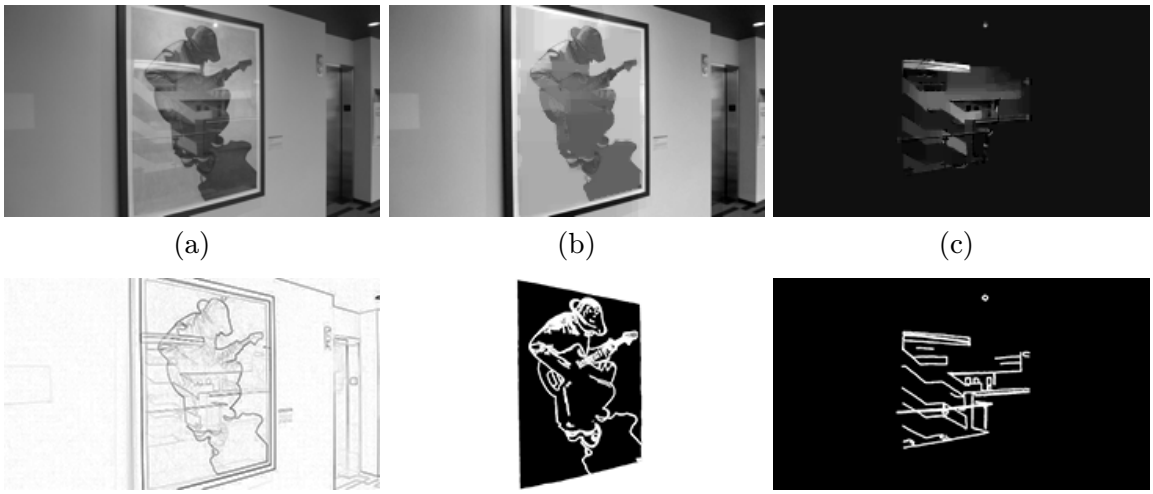
(m) $\lambda = 0.7$ (n) $\lambda = 0.9$ (o) $\lambda = 1$ (p) Ground truth

Figure 5.1: *The role of gradients for color recovery.* **(a,i)** Ground truth of the front and the rear intensities. **(b-h,j-p)** Estimations for different values of λ using the ground truth of layer gradients and layer motions estimated with the method by Gai et al. [13] (this motion estimation is close to ground truth for this sequence). (The results were computed with the quadratic programming approach (Section 5.2.1) and were saturated.)

I_0 and I_1. Hence, the minimization of $E(I_0, I_1)$, s.t. the nonnegative constraint of layer intensity, can be rewritten into the following matrix form:

$$\begin{aligned} &\min_{l} : (\mathbf{A}l - \delta)^\top (\mathbf{A}l - \delta) + |\mathbf{E}l - \tau| \\ &\text{s.t.} : l \geq 0, \end{aligned} \tag{5.4}$$

where l is a large vector which stores intensities of both layers I_0 and I_1. $\mathbf{A}$ is a sparse mixing matrix with $N\|l\|$ rows and $M\|l\|$ columns, where N and M is the number of input mixtures and the number of layers respectively ($M = 2$ in our case). The matrix $\mathbf{A}$ contains two elements equal to 1 in each row at indices of pixels to be mixed, the rest is filled by 0. $\mathbf{A}l$ is then a vector with $M\|l\|$ columns containing pixels of all synthesized mixtures $\widetilde{C}_v$. δ is a vector including all pixels of the observed mixtures C_v. $\mathbf{E}$ is a sparse matrix with



(a) (b) (c)

(d) (e) (f)

Figure 5.2: *Layer intensities recovered using manually labeled gradients and layer motions estimated with the method by Gai et al. [13].* **(a)** Reference image. **(b,c)** Estimations of the front and the rear intensities. **(d)** Gradients of the reference image. **(e,f)** Labels of significant gradients of the front and the rear layer (obtained manually).

$2M\|l\|$ rows and $M\|l\|$ columns which serves for calculation of gradients components of the estimated layers and $\mathbf{E}l$ is then a vector with $2M\|l\|$ columns, where one half contains x-components and the other half contains y-components. τ is a vector including the components of the estimated gradients.

By introducing auxiliary vector w for mixing differences and vectors ϵ^+ and ϵ^- for positive and negative differences of gradients components,[1] the minimization problem (5.4) becomes:

$$\begin{aligned} \min_{l,w,\epsilon^+,\epsilon^-} &: w^\top w + \mathbf{1}^\top(\epsilon^+ + \epsilon^-) \\ \text{s.t.} &: \mathbf{A}l - \delta = w,\ \mathbf{E}l - \epsilon^+ + \epsilon^- = \tau, \\ &\epsilon^+ \geq 0, \epsilon^- \geq 0, l \geq 0. \end{aligned} \tag{5.5}$$

The above problem, when solved by quadratic programming, yields a local solution which actually varies from all the other local solutions only by an additive constant. There is in fact no unique solution since one can add a constant value to one layer and subtract the same value from the other, as long as all pixel values are nonnegative.

Run Time of the Quadratic Programming

Gai et al. [13] utilized the highly optimized implementation of the interior point algorithm [6] from *MOSEK* library [2]. Their program takes around 10 minutes to solve the problem even for small images of size $240 \times 135px$.[2] For images of double size, the program takes more

[1] We need to store the positive and the negative differences separately to simulate $\mathcal{L}_1$-norm. The mixing differences are expressed only by w because we take their square an thus the sign is eliminated.

[2] Quad core 2,2 GHz CPU on a PC with 8GB RAM was used.

than one hour on average. Although this cannot be considered as any complexity analysis, it at least illustrates the very high computational demand of this quadratic programming problem when solved in the traditional way.

5.2.2 Alternative Approach to Minimization

Instead of treating the minimization as a quadratic programming problem, we can exploit some specific properties of our task to speed it up dramatically. In particular, we can iteratively minimize the energy by alternation between refinement of the front and the rear layer intensities. This strategy was used by Tsin et al. [32] to minimize the energy (5.2). Our goal is now to adapt this strategy to minimize the energy (5.3).

Initial Estimation

At the beginning, the min-composite [30] is taken as an initial estimate of the front layer intensities I_0. For each pixel, the min-composite is given by the minimum of a stack of registered images. The registration is done by warping of images from all views to the reference image plane which is performed by inverse version of the warping functions $T_{v,0}$ from the mixing model (Section 2.1.3). The min-composite is a good upper bound on possible values of the front layer intensities because contributions from the rear layer can only add to the intensity at a given pixel.

Rear Layer Refinement

It is now described how a new estimate of the rear layer intensities I_1 can be obtained given a fixed current estimate of the front layer intensities I_0. The refinement of the front layer intensities then proceed symmetrically. Both steps are repeated until convergence which is proved below.

The energy (5.3) can be rewritten into the following form:

$$E(I_0, I_1) = \sum_{p \in \mathcal{P}} \left((1-\lambda) \sum_{v \in \mathcal{V}(p)} A(p) + \lambda \big(B_0(p) + B_1(p) \big) \right), \tag{5.6}$$

$$A(p) = \sigma\big(C_v(p)\big) \Big(C_v(p) - \widetilde{C}_v(p) \Big)^2, \tag{5.7}$$

$$B_0(p) = \sum_k \Big| \nabla_k I_0(p) - G_{0,k}(p) \Big|^a, \tag{5.8}$$

$$B_1(p) = \sum_k \Big| \nabla_k I_1(p) - G_{1,k}(p) \Big|^a, \tag{5.9}$$

where $A(p)$ is a term measuring deviation from the mixing model and $B_0(p)$ and $B_1(p)$ are terms measuring deviation from the estimated layer gradients.

Because all occlusions have been taken care of by the definitions of $\mathcal{V}$ and $\mathcal{P}$, the warped difference image $W(x) = T_{v,1}^{-1} \circ \big(C_v(x) - \widetilde{C}_v(p) \big)$ has the same difference values as the original one. As a result:

$$A(p) = \sigma\big(C_v(p)\big)\Big(C_v(p) - \widetilde{C}_v(p)\Big)^2, \tag{5.10}$$

$$= \sigma\big(C_v(p)\big)\Big(T_{v,1}^{-1} \circ \big(C_v(p) - \widetilde{C}_v(p)\big)\Big)^2, \tag{5.11}$$

$$= \sigma\big(C_v(p)\big)\Big(T_{v,1}^{-1} \circ \big(C_v(p) - T_{v,0} \circ I_0(p)\big) - I_1(p)\Big)^2. \tag{5.12}$$

Since I_0 is assumed to be fixed, also the term $B_0(p)$ is fixed and thus we can ignore it when updating the rear intensities. Consequently, we can consider this energy:

$$F(I_1) = \sum_{p \in \mathcal{P}} \left((1-\lambda) \sum_{v \in \mathcal{V}(p)} A(p) + \lambda B_1(p) \right), \tag{5.13}$$

whose minimization certainly leads to minimization of energy $E(I_0, I_1)$.

If we now incorporate (5.12) into (5.13), set the parameter a in the term $B_1(p)$ (5.9) to 2 and consider the gradient components to be calculated by convolution with the horizontal and vertical difference filters ($[0, -1, 1]$ and $[0, -1, 1]^\top$), we get:

$$F(I_1) = \sum_{p \in \mathcal{P}} \left((1-\lambda) \sum_{v \in \mathcal{V}(p)} \sigma\big(C_v(p)\big)\Big(T_{v,1}^{-1} \circ \big(C_v(p) - T_{v,0} \circ I_0(p)\big) - I_1(p)\Big)^2 \right.$$
$$\left. + \lambda\Big(I_1(p_r) - I_1(p) - G_{1,x}(p)\Big)^2 + \lambda\Big(I_1(p_b) - I_1(p) - G_{1,y}(p)\Big)^2 \right), \tag{5.14}$$

where p_r is the right neighbour of pixel p and p_b is its bottom neighbour.

By taking derivative of this quadratic energy function w.r.t. $I_1(p)$ and setting it equal to 0, the solution can be expressed as:

$$I_1(p) = \left((1-\lambda) \sum_{v \in \mathcal{V}(p)} \sigma\big(C_v(p)\big)\Big(T_{v,1}^{-1} \circ \big(C_v(p) - T_{v,0} \circ I_0(p)\big)\Big) \right.$$
$$\left. + \lambda\Big(I_1(p_r) - G_{1,x}(p)\Big) + \lambda\Big(I_1(p_b) - G_{1,y}(p)\Big) \right) \frac{1}{(1-\lambda)\sum_{v \in \mathcal{V}(p)} \sigma\big(C_v(p)\big) + 2\lambda}. \tag{5.15}$$

However, the intensity estimates of neighbouring pixels are coupled and thus we cannot calculate them directly using this equation. Fortunately, they can be estimated by solving a sparse system of linear equations which can be efficiently done by iterative methods, such as the *Gauss-Seidel* or *Jacobi* method [14]. Both methods are guaranteed to converge because the matrix of our linear system is strictly diagonally dominant which is one of the sufficient conditions of their convergence.

Note that the new estimate of the rear intensity for a pixel p (5.15) is actually a weighted average of estimates from all the available views and of estimates given by the required agreement with the earlier estimated gradients.

The following clamping operation is performed after each refinement step to assure the intensity is in the range of $[0, 255]$:

$$I_1'(p) = \max\{\min\{I_1(p), 255\}, 0\}. \tag{5.16}$$

Proof of Convergence

Similarly as in [32], the convergence of this alternative approach is guaranteed because each step of the rear intensities refinement decreases the energy $E(I_0, I_1)$ (5.6) by minimizing $A + \lambda B_1$ and each step of the front intensities refinement decreases the same energy by minimizing $A + \lambda B_0$. The method converges to a local solution which is equal to the global optimal solution up to an additive constant (as discussed in Section 5.2.1).

The clamping operation (5.16) does not affect the convergence because the quadratic energy (5.14) has only one minimum. If the minimum is outside of the range $[0, 255]$, the closer border of this range is the minimum solution in the valid range.

Gradually Emphasized Agreement with Gradients

We found that it is beneficial to repeat the whole recovery process with gradually increasing value of the trade-off coefficient λ, when the next stage is always initialized with the results of the previous one. By starting at lower values of λ, the agreement with the mixing model is favoured at first. This provides a good base for the later stages when the details are refined by favouring the agreement with the estimated gradients.

In our experiments, we usually start at $\lambda \approx 0.1$ which is then gradually increased up to $\lambda \approx 0.9$ (with a step of ≈ 0.1).

Run Time of the Alternative Approach

This alternative approach (with the gradually emphasized agreement with gradients) needed only a few seconds ($\approx 3s$) to recover layer intensities from images of size $240 \times 135px$. For images of double size, it took $\approx 5s$. These values were measured using the same computer as was used to measure the run time of the quadratic programming approach.

5.3 Results of Layer Colors Recovery

As the color channels do not interact in the described color recovery process, we conducted the most of experiments on grayscale images which is sufficient to show the quality of results and to explain common artefacts. Estimated front layer intensities are shown in Figure 5.3, corresponding rear layer intensities in Figure 5.4. We also show several results on color images (Figure 5.5) when the estimation of gradients and the color recovery process was performed for each channel separately and the results were then merged to obtain the final color estimates.

The best results were obtained by the quadratic programming (Section 5.2.1) using the layer gradients estimated with the method introduced in Chapter 4. The most of the problematic regions was separated reliably, even though their subtle footprints can be usually found in the wrong layer. These artefacts are caused by slight imprecision in the layer gradients estimation. Because the layer gradients for the BLOCKS sequence were estimated very well, the agreement with these gradients was emphasized by setting the trade-off parameter λ to a higher value ($= 0.9$). However, the results were similarly good even for smaller values. For the other sequences, λ was set to 0.1.

The alternative approach (Section 5.2.2) could also separate the most of the image regions quite well but it is more sensitive to defects in estimated gradients. This has the effect of *intensity waves* propagated from the wrong gradients. These artefacts can be seen also in the recovered colors in Figure 5.5. Our experiments showed that as the required precision of solution increases (it is set by termination criterion when solving the system of liner equations), the quality of results gets closer to the results of quadratic programming. However, more experiments should be done for more detailed analysis of this approach. For the presented results, the trade-off parameter λ was gradually increased from 0 to 0.95 for the BLOCKS sequence and from 0 to 0.7 for the other sequences. The step of λ was set to 0.01 in all cases. The required precision of solution for each value of λ was set to $\epsilon = 0.001$.

Gai et al. [13] use also the quadratic programming approach but the defects of their gradients estimation lead to lower quality of the recovered intensities. As was discussed in Section 4.6, the method could produce better results for the BLOCKS sequence if it is encouraged to describe the motion of the rear layer by more homographies. In the presented results, the motion of the rear layer was described by only one homography which captured mainly motion of the front block.

In layer colors of the BLOCKS sequence recovered by the alternative approach (Figure 5.5), we can notice that the blue channel was not separated well. This is caused by the fact that the separation problem can be solved only up to an additive constant which was discussed in Section 5.2.

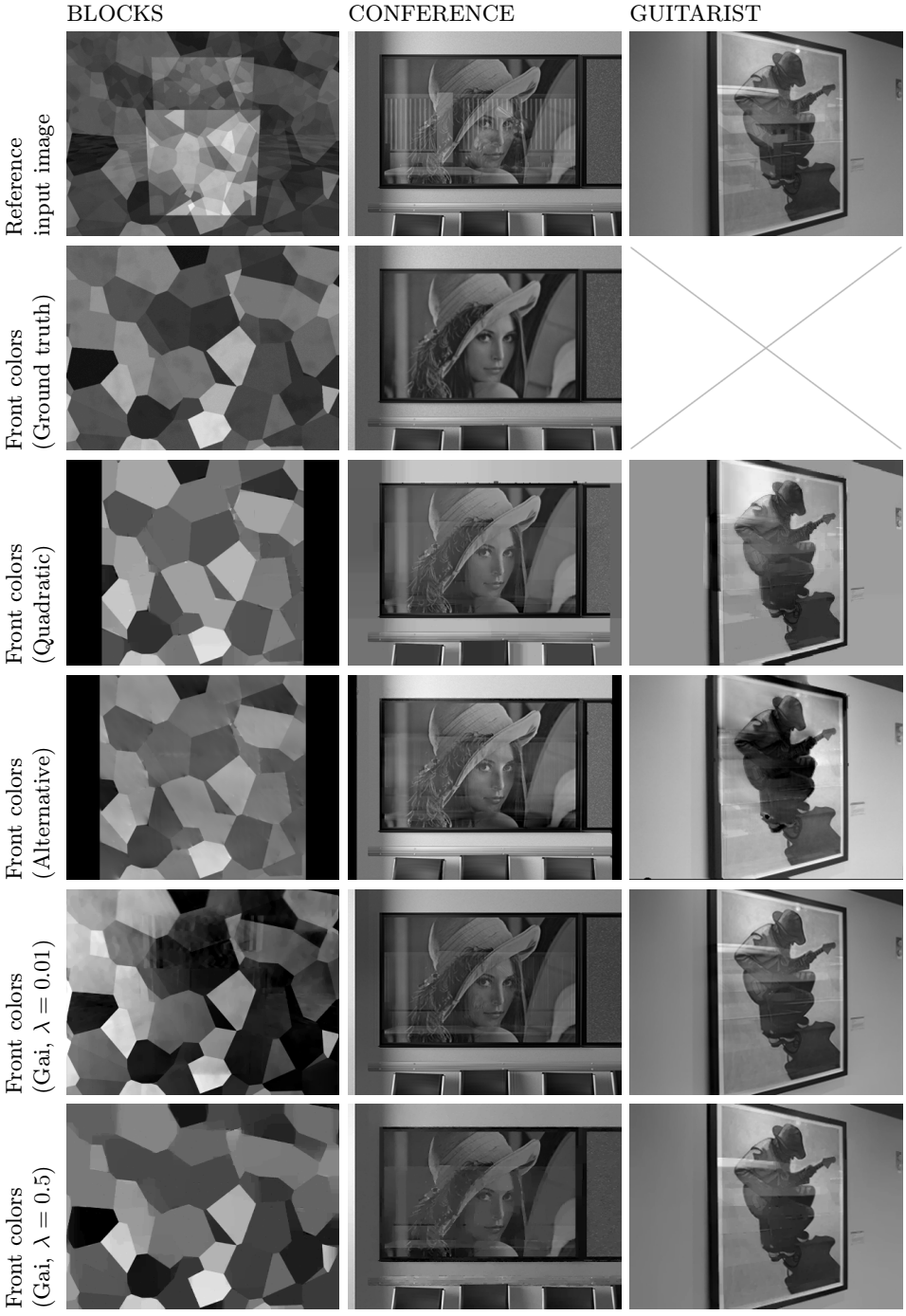


Figure 5.3: *Results of the front layer intensities estimation.*

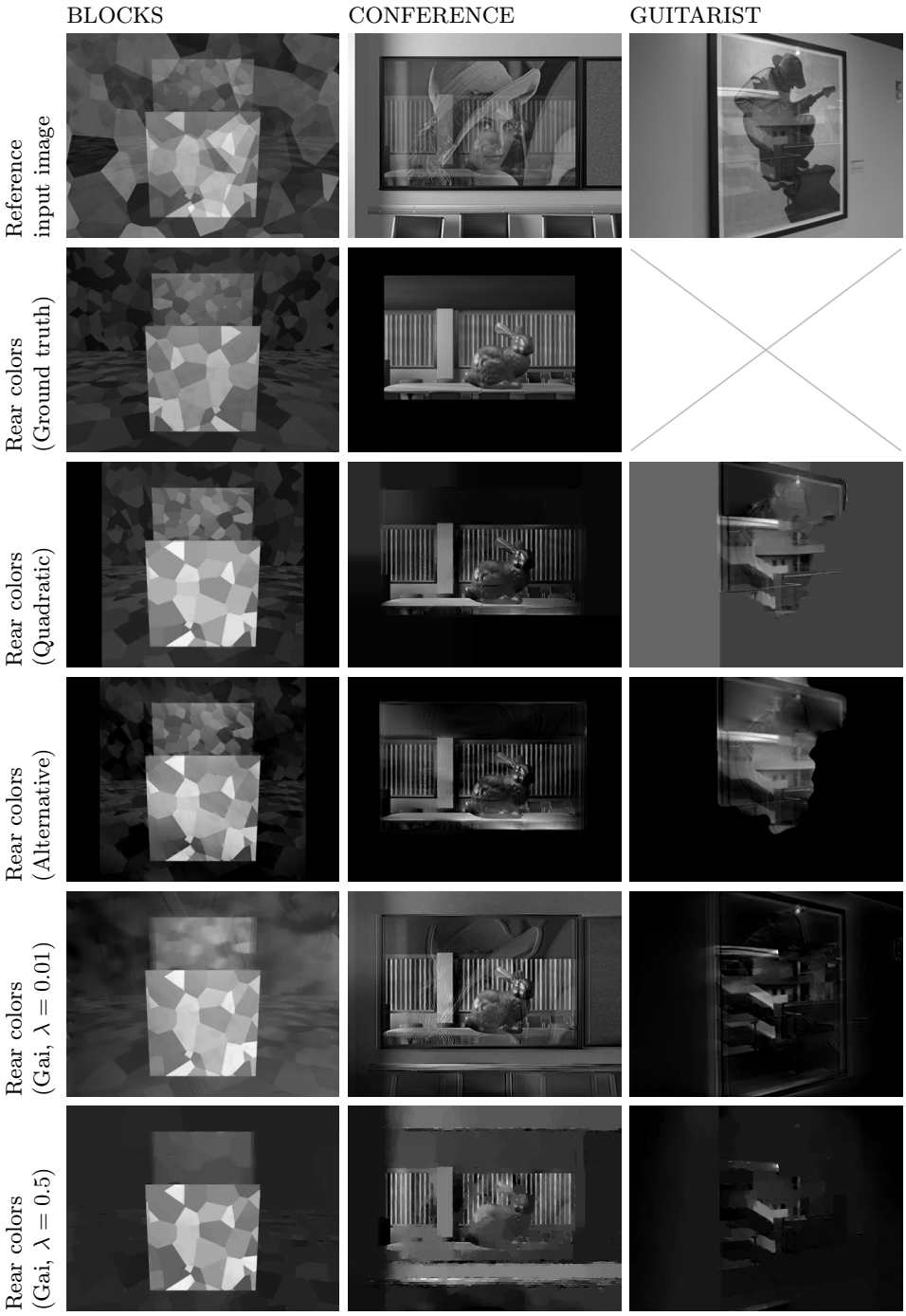


Figure 5.4: *Results of the rear layer intensities estimation.*

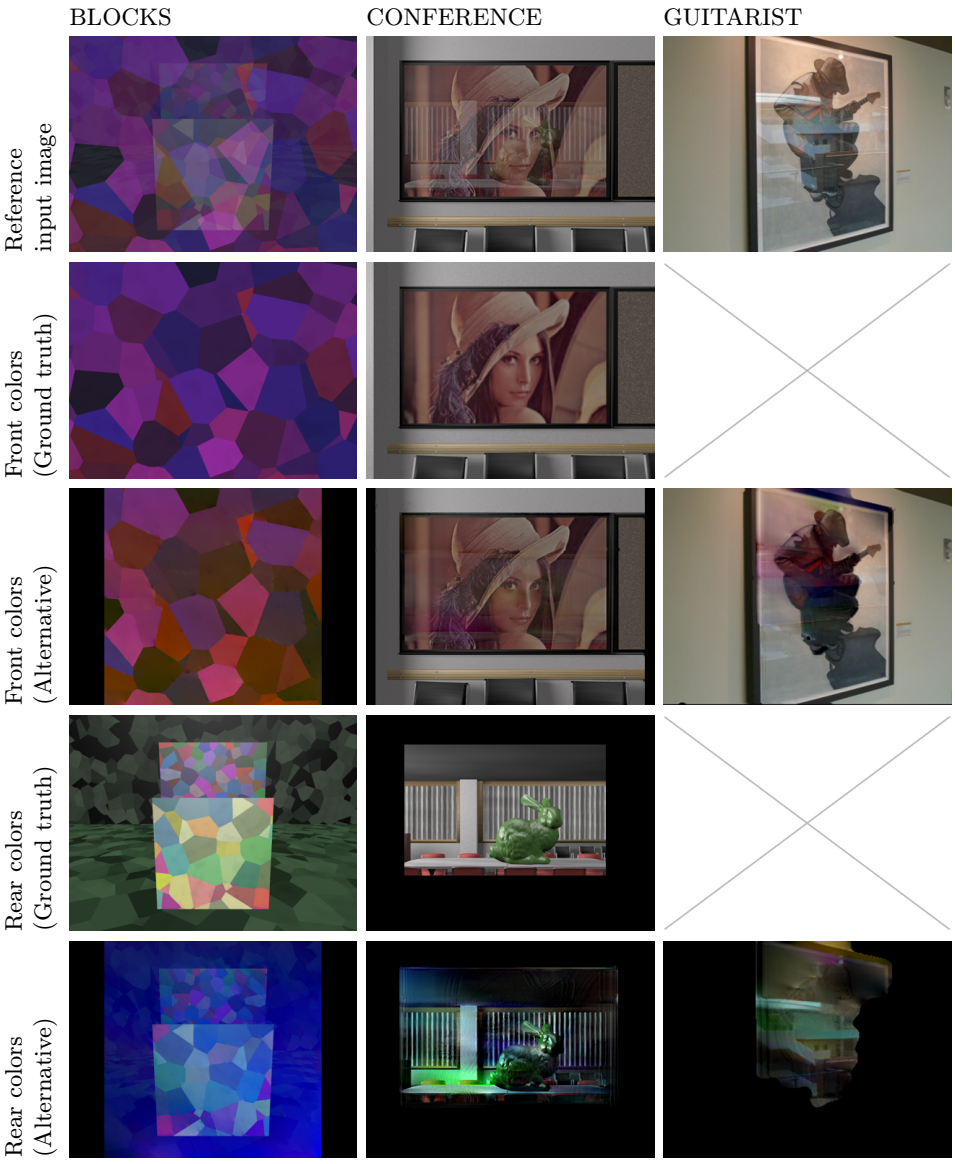


Figure 5.5: *Results of the front and the rear layer colors estimation.*

Chapter 6

Datasets

This chapter introduces in more detail the image sequences which were used for our experiments. All the sequences were obtained with a camera moving laterally and their reflection and transmission layers were mixed in all images with approximately constant mixing coefficients. In the type of scenes we mainly focused on, the specular surface is typically a textured plane and the reflection layer contains some noticeable depth discontinuities (i.e. it cannot be reliably described just by homography).

We have created two synthetic image sequences (Section 6.1) whose ground truth of depths, colors and gradients is available and can be used for evaluation of results. To validate our implementation on real data, we also used image sequences of Sinha et al. [27] (Section 6.2).

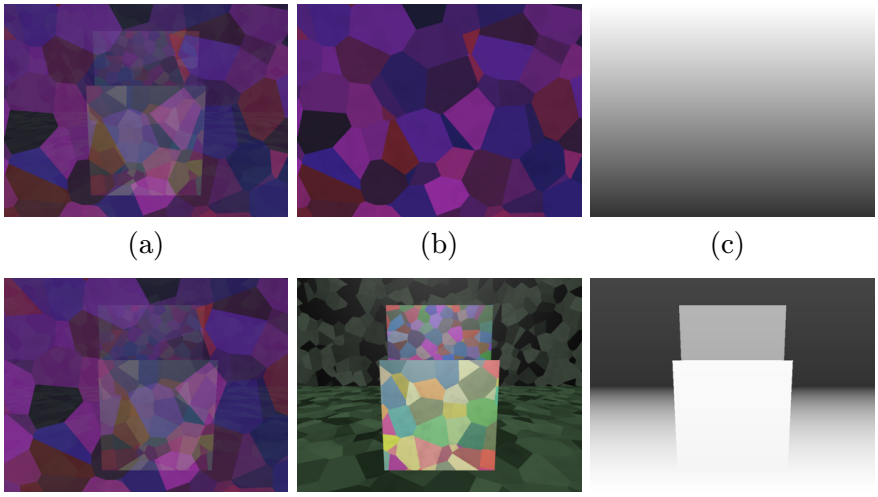
6.1 Synthetic Image Sequences

The first synthetic sequence is called the BLOCKS sequence (Figure 6.1). It is a very simple scene whose main role is to provide simple data suitable for initial debugging. The scene, from which the sequence was rendered, consists of two blocks which are reflected on a specular surface. The Voronoi diagram was used as the texture of all scene elements. There is no opaque region in the front layer so there are two depths at every pixel.

The other created sequence is called the CONFERENCE sequence (Figure 6.2). Its purpose is to provide more challenging images which are closer to real ones. As the base of the scene, we used a 3D model of a real conference room of the Lawrence Berkeley National Laboratory [18]. To create a textured specular surface, we hung up the well known image of Lena on the wall of this room and added the mirror effect on this image. Moreover, to achieve some more significant depth discontinuities in the reflection layer, we placed the Stanford Bunny [1] on the table.

6.2 Real Image Sequences

The employed real sequences were collected by Sinha et al. [27] using regular hand-held photography. To achieve constant exposure and color balance, they either locked the exposure on the camera, or chose subsets of images where the exposure was fairly constant. Some of the sequences are presented in Figure 6.3.



(a) (b) (c)

(d) (e) (f)

Figure 6.1: *The BLOCKS sequence.* **(a,d)** Sample images. **(b,e)** Ground truth of transmission and reflection colors. **(c,f)** Ground truth of transmission and reflection depths.

(a) (b) (c)

(d) (e) (f)

Figure 6.2: *The CONFERENCE sequence.* **(a,d)** Sample images. **(b,e)** Ground truth of transmission and reflection colors. **(c,f)** Ground truth of transmission and reflection depths.

(a) (b) (c)

Figure 6.3: *Sample images of the GUITARIST sequence* **(a)**, *the MUZEUM sequence* **(b)** and *the STATUE sequence* **(c)**.

Chapter 7

Implementation

The majority of our implementation of the suggested method (described in Section 2.4) was written in C++ programming language. Besides that, we also adapted a MATLAB code written by Gai et al. [13] which implements the quadratic programming approach to color recovery.

Here is a list of employed libraries:

- **OpenCV** [4] - a library for computer vision, image processing and machine learning.
- **MRF Minimization** [31] - a library used for graph cut optimization of a multi-label Markov random field (described in Section 3.3.3).
- **Mosek** [2] - a high performance library for large-scale optimization problems (used in the MATLAB code adapted from Gai et al.).
- **pugixml** [16] - a light-weight XML processing library used to parse application parameters provided in an XML file.

Of a great help was also this software:

- **MeshLab** [8] - an open source system for processing and editing of unstructured 3D triangular meshes and point clouds. It was very beneficial especially for debugging of the two-layer depth estimation. The visualizations in Section 3.3 were also obtained using this software.
- **Blender** - an open source 3D modeling software which was used to render the synthetic scenes presented in Section 6.1.
- **Sublime Text** - a sophisticated text editor reducing the *transaction cost* during development to minimum.

The source code of our implementation is available on the supplemental CD.

Chapter 8

Conclusion

Within the Master's thesis our goal was to study existing methods for detection and removal of specular reflection, to find their limitations and to suggest possibilities of their improvements. Particularly, an attention was paid to planar specular (i.e. mirror-like) surfaces whose appearance can be modeled or approximated by linear superposition of reflection and transmission layer.

We considered different ways to solve the problem and decided to concentrate on motion as the main clue. We reviewed the existing motion-based methods and described their common degenerate case in terms of their disability to correctly separate regions with low frequency in the direction of camera motion.

We suggested a new method designed to eliminate the degenerate case, i.e. to correctly separate the problematic regions. It assumes an image sequence as the input and consists of three main stages. At first, a depth map for both layers is estimated in the form of piecewise-planar 3D proxies. The method by Sinha et al. [27] was adapted for this purpose. Afterwards, image gradients are separated into the two layers using gradients matching costs at different depths while taking a special treatment of edges of the problematic regions. Each such edge is assigned to the layer where it has better topological fitness. In other words, the edge is assigned to the layer where it leads to better continuity of gradients and smaller number of intersections which are argued to be more likely caused by edges from different layers. Once the 3D proxies and the layer gradients have been estimated, layer colors are recovered by minimization of an energy expressing deviation from the mixing model and the estimated gradients.

The final minimization can be treated as a quadratic programming problem, similarly as was done by Gai et al. [13]. We suggested also a more efficient alternative approach whose results are not as superior as the results of the quadratic programming approach, but since it is noticeably faster it can be favoured when the run time is the priority.

The suggested method was experimentally validated and the results of all three stages were discussed. The calculated 3D proxies usually represent the dominant scene elements well but lack for finer details. However, the proxies can describe the layer motion between different views still better than homography which is often used in other methods. The layer gradients estimated by our approach were shown to be of high quality unless the assumption about sparsity of gradients is violated. It was also shown that if the layer gradients are estimated accurately, the layer colors can be recovered reliably even in the problematic regions and thus the degenerate case, which is common for the existing methods, can be eliminated.

8.1 Contributions

- **A new method for reflection and transmission separation was suggested.** It was inspired mainly by the work of Sinha et al. [27] and Gai et al. [13] and its main contribution is the ability to handle the degenerate case described in Section 2.3.
- **A new approach to layer gradients estimation has been devised.** The estimated gradients play an important role in the color recovery process where they help to assign the problematic regions to the correct layer. But since image gradients in general represent one of the fundamental building blocks in image processing, there could be definitely found also some other applications which could benefit from the gradients separated into layers.
- **An alternative approach to the color recovery optimization was described**. It is in fact an application of the approach by Tsin et al. [32] to the energy function defined by Gai et al. [13]. It does not produce better results than the quadratic programming approach presented by Gai et al. [13], but it is noticeably faster.

8.2 Future Work

The main limitation of our current implementation lies in the process of camera calibration which is a necessary step in the two-layer depth estimation. We used Automatic Camera Tracking System (ACTS) by Zhang et al. [34] which usually failed to estimate more complex camera motion in the presence of specular reflections. As was mentioned in Section 3.1, the calibration in this case is more difficult because there might be many spurious features created by intersections of edges from different layers. To reduce this bottleneck, the camera calibration process should be studied in more detail and experiments with different approaches should be conducted.

When the problem with camera calibration is solved, the next step should be a more rigorous evaluation of the suggested method, especially for more real image sequences.

Besides these tasks related to our work, there can still be found open challenges in the problem of reflection and transmission separation. These include a recovery of reflection from a nonplanar specular surface and a more accurate two-layer depth estimation, to name a few.

Bibliography

[1] The Stanford 3D scanning repository, 2013. Available online: `http://graphics.stanford.edu/data/3Dscanrep` [cited 19/05/2013].

[2] MOSEK Aps. Mosek: High performance software for large-scale LP, QP, SOCP and MIP, 2013. Available online: `http://www.mosek.com` [cited 02/05/2013].

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[4] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Incorporated, 2008.

[5] Alexander M Bronstein, Michael M Bronstein, Michael Zibulevsky, and Yehoshua Y Zeevi. Sparse ICA for blind separation of transmitted and reflected images. *International Journal of Imaging Systems and Technology*, 15(1):84–91, 2005.

[6] Richard H Byrd, Mary E Hribar, and Jorge Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999.

[7] Andrzej Cichocki, Shun-ichi Amari, et al. *Adaptive blind signal and image processing*. John Wiley Chichester, 2002.

[8] Visual Computing Lab ISTI CNR. MeshLab (3d viewer), 2012. Available online: `http://meshlab.sourceforge.net` [cited 15/05/2013].

[9] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 905–914. ACM, 2004.

[10] R.T. Collins. A space-sweep approach to true multi-image matching. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on*, pages 358–363. IEEE, 1996.

[11] A. Criminisi, S.B. Kang, R. Swaminathan, R. Szeliski, and P. Anandan. Extracting layers and analyzing their specular properties using epipolar-plane-image analysis. *Computer vision and image understanding*, 97(1):51–85, 2005.

[12] H. Farid and E.H. Adelson. Separating reflections and lighting using independent components analysis. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1. IEEE, 1999.

[13] K. Gai, Z. Shi, and C. Zhang. Blind separation of superimposed moving images using image statistics. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(1):19–32, 2012.

[14] Richard Hamming. *Numerical methods for scientists and engineers*. Courier Dover Publications, 2012.

[15] H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):328–341, 2008.

[16] A. Kapoulkine. pugixml: Light-weight, simple and fast XML parser for C++ with XPath support, 2012. Available online: `http://code.google.com/p/pugixml` [cited 30/12/2012].

[17] A. Levin, A. Zomet, and Y. Weiss. Separating reflections from a single image using local features. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–306. IEEE, 2004.

[18] Morgan McGuire. Computer graphics archive, 2013. Available online: `http://graphics.cs.williams.edu/data` [cited 19/05/2013].

[19] United States. National Bureau of Standards and Fred Edwin Nicodemus. *Geometrical considerations and nomenclature for reflectance*, volume 160. US Department of Commerce, National Bureau of Standards Washington, D. C, 1977.

[20] H. Pfister, M. Zwicker, J. Van Baar, and M. Gross. Surfels: Surface elements as rendering primitives. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 335–342. ACM Press/Addison-Wesley Publishing Co., 2000.

[21] Charles A Rothwell, JL Mundy, W Hoffman, and V-D Nguyen. Driving vision by topology. In *Computer Vision, 1995. Proceedings., International Symposium on*, pages 395–400. IEEE, 1995.

[22] Yoav Y Schechner, Nahum Kiryati, and Joseph Shamir. Blind recovery of transparent and semireflected scenes. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 38–43. IEEE, 2000.

[23] Y.Y. Schechner, N. Kiryati, and R. Basri. Separation of transparent layers using focus. In *Computer Vision, 1998. Sixth International Conference on*, pages 1061–1066. IEEE, 1998.

[24] Y.Y. Schechner, J. Shamir, and N. Kiryati. Polarization-based decorrelation of transparent layers: The inclination angle of an invisible surface. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 814–819. IEEE, 1999.

[25] M. Shizawa and K. Mase. Simultaneous multiple optical flow estimation. In *Pattern Recognition, 1990. Proceedings., 10th International Conference on*, volume 1, pages 274–278. IEEE, 1990.

[26] M. Shizawa and K. Mase. Principle of superposition: A common computational framework for analysis of multiple motion. In *Visual Motion, 1991., Proceedings of the IEEE Workshop on*, pages 164–172. IEEE, 1991.

[27] S.N. Sinha, J. Kopf, M. Goesele, D. Scharstein, and R. Szeliski. Image-based rendering for scenes with reflections. *ACM Transactions on Graphics (TOG)*, 31(4):100, 2012.

[28] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM transactions on graphics (TOG)*, volume 25, pages 835–846. ACM, 2006.

[29] R. Szeliski. *Computer vision: Algorithms and applications*. Springer, 2010.

[30] R. Szeliski, S. Avidan, and P. Anandan. Layer extraction from multiple images containing reflections and transparency. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 246–253. IEEE, 2000.

[31] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(6):1068–1080, 2008.

[32] Y. Tsin, S.B. Kang, and R. Szeliski. Stereo matching with linear superposition of layers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(2):290–301, 2006.

[33] Sai-Kit Yeung, Tai-Pang Wu, and Chi-Keung Tang. Extracting smooth and transparent layers from a single image. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7. IEEE, 2008.

[34] G. Zhang, Z. Dong, H. Jiang, Q. Li, and Y. Shao. ACTS: Automatic camera tracking system 2.0, 2012. Available online: `http://www.zjucvg.net/acts/acts.html` [cited 30/12/2012].

[35] G. Zhang, X. Qin, W. Hua, T.T. Wong, P.A. Heng, and H. Bao. Robust metric reconstruction from challenging video sequences. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

Appendix A

Content of Supplemental CD

- **datasets** - The datasets which were used in our experiments including the ground truth of our synthetic sequences.
- **doc** - The final report including its LaTeX sources.
- **src** - The source code of our implementation.
- **README** - A text file with instructions for how to run the implementation.