Hole Filling for 3-D View Synthesis Using Sparse Representations

Gabriel Schönung, Julian Habigt, Klaus Diepold

Technical Report

Hole Filling for 3-D View Synthesis Using Sparse Representations

Gabriel Schönung, Julian Habigt, Klaus Diepold

October 9, 2017

Institute for Data Processing
Technische Universität München

Gabriel Schönung, Julian Habigt, Klaus Diepold. *Hole Filling for 3-D View Synthesis Using Sparse Representations.* Technical Report, Technische Universität München, Munich, Germany, 2017.

Institute for Data Processing, Technische Universität München, 80290 München, Germany, http://www.ldv.ei.tum.de.

Abstract

The challenge of inpainting holes of color information in images has been widely addressed throughout the image processing research community. One of the most popular methods from Criminisi et al. [1] is exemplar-based and assigns filling priorities during the inpainting process. Another method based on Sparse Representations, which become more and more popular in the domain of signal processing, was proposed by Mairal et al. [2].
This work combines the idea of filling priorities and Sparse Representations. Furthermore it improves the results by inpainting over multiple scales and taking advantage of depth information. In addition, an exemplar-based method (MS-SSD-D) has been developed that uses multiple scales and depth as well. Whereas the results of the method using Sparse Representations (MS-KSVD-D) lie within the ones of existing algorithms, the MS-SSD-D method outperforms the available state-of-the-art algorithms in both objective and subjective quality measures of selected test images.
The main application area of the developed inpainting algorithms are the type of holes, that occur during 3-D view synthesis (typically large vertical holes).

Das Füllen fehlender Bildinformationen ist bereits umfassend von verschiedenen Forschungsgruppen in der Bildbearbeitung behandelt worden. Eine der bekanntesten Veröffentlichungen zu dem Thema von Criminisi et al. [1] vergibt Prioritäten, in welcher Reihenfolge die Bereiche des Loches gefüllt werden sollen und füllt dann mithilfe von Bildteilen des Restbildes. Ein weitere Methode von Mairal et al. [2] verwendet die in der Signalverarbeitung immer populärer werdende Methode der "Sparse Representations".
Diese Arbeit verbindet die Ideen der Füll-Prioritäten und der Sparse Representations. Des weiteren wird der Ansatz durch die Verwendung von mehreren Skalen von Füll-Bildstücken und von Tiefeninformation verbessert. Zudem wurde eine weitere Methode (MS-SSD-D) entwickelt, die mit dem Kopieren von Bildermaterial arbeitet und ebenfalls über obige Verbesserungen verfügt. Während die Methode, die sich der Sparse Representations bedient (MS-KSVD-D), ähnliche Ergebnisse wie die aktuellen weit verbreiteten Methoden liefert, übertrifft die MS-SSD-D Methode diese sowohl in objektiven als auch in subjektiven Qualitätskriterien bei einer Auswahl von Testbildern.
Das Hauptanwendungsgebiet der entwickelten Füll-Algorithmen sind die Löcher, die entstehen, wenn neue Bilder durch 3-D Synthesealgorithmen erzeugt werden (typischerweise große vertikale Löcher).

Contents

1. Introduction

1.1. Motivation

Systems that support 3-D video had their breakthrough on the consumer market a few years ago[1], however the efforts to establish an industry-wide standard trace back more than 10 years. In 2002, the Advanced Three-dimensional Television System Technologies (ATTEST) project defined their goal of developing a commercially feasible European 3D-TV broadcast system [3]. Part of this project was to develop an algorithm that could synthesize new 3-D views from existing views plus a corresponding depth map (as depth information can be transmitted very efficiently). One big challenge that remains today is the lack of visual information that can occur during this process, so called "holes". Many approaches have been proposed including the use of Sparse Representations [2], which became very popular in the domain of signal processing in recent years. An older approach consists of assigning priorities for the filling order to the boundary of a hole according to the reliability of the surrounding information and the type of structure before filling it with other image patches [1]. Unfortunately both methods do not function very well when it comes to the task of filling the size of holes that can occur during the 3-D view synthesis process. This technical report aims to develop a new algorithm that produces better results for use in Depth-Image Based Rendering (DIBR) for 3-D video.

1.2. Inpainting: Prior Art

Inpainting techniques, also known as hole filling or image completion have been widely discussed in several publications during the last ten years. In order to classify the most popular methods, the inpainting process can be divided in two separate parts: Given a hole with missing information, at first a prioritization or order has to be established concerning how the hole should be filled. There are many possibilities regarding what this prioritization can be based on, for example the structure or the texture at the border of the hole, the depth information or the filling direction. In the actual filling step, the widely used methods are either exemplar-based, dictionary-based or use differential equations.
One of the most fundamental methods in inpainting has been developed by Criminisi et al. [1]. They compute a filling priority at the hole borders using a "confidence term" and a "data term". The confidence term measures the reliability of the information surrounding the hole. The data term helps to propagate structure, the so called isophotes, into the inpainted region by calculating an image gradient. Furthermore the filling step of this inpainting algo-

[1] http://www.electronicsweekly.com/Articles/29/01/2010/47903/3d-tv-market-is-set-for-big-growth.htm

rithm is exemplar-based. This means that the image patch is taken, which has the smallest Sum of Squared Differences (SSD) compared to the existing information in the image patch that is about to be inpainted. As a result the propagation of image texture is better. The method is presented in detail in Section 2.4.
Many researchers adopted the concept of Criminisi et al. for their own purpose. Luo et al. [4] simply summed up the confidence and the data term whereas Criminisi et al. multiplied it. The reason was that in the original framework the isophote would stop propagating when the confidence approaches zero (e.g. in the middle of large holes). In addition to that, Luo et al. included the depth information in the cost function of the exemplar-based filling step. Daribo et al. [5] and Gautier et al. [6] also modify Criminisi's approach by including depth information. Daribo et al. multiply the confidence and data term by an additional "level regularity term" which is defined as the inverse variance of the depth level. This favors image patches with the same depth. The filling is done through differential equations. Gautier et al. modified the data term such that the gradient is replaced by a more robust structure tensor, the Di Zenzo matrix [7]. The computation of the data term is similar to the Harris corner detection algorithm. To include depth information the tensor is computed over the color channels and the depth value. Additionally the depth information is used for the SSD in the filling step to favor image patches from the same depth level. Oh et al. [8] proposed a concept where patches are only coming from the background. They only consider neighboring regions for the inpainting step. When a hole is located both on foreground and background, they "replace the boundary region bordering on foreground to the background region located on the opposite side". In [9], Cheng et al. limit the search region for the patch matching also with the background information. Furthermore, they compute strength magnitude on candidate patches based on the inverse geometric distance and the partial image gradient.
A different approach on inpainting priorities has been developed by Xu et al. [10]. They define a structure sparsity of an image patch by measuring the sparseness of the similarities with its neighboring patches. This results in a better propagation of image structure. A hole is eventually filled by a sparse linear combination of candidate patches.
Another very popular inpainting method of Bertalmio et al. [11] is based on Partial Differential Equation (PDE) known from fluid dynamics. The image intensity is seen as a "stream function" for a two-dimensional incompressible flow. A PDE is established for the flow with respect to the isophotes. A steady state solution gives a vector field that is used for the filling process.
A method that integrates temporal information in the inpainting process was presented by Wexler et al. [12]. They compute spatial and temporal derivatives for each space-time point and include this information in a similarity measure.
Exploiting scene geometry can be of interest for inpainting. Hervieux et al. [13] establish depth planes over different color segments. Then they extend the depth layers into the hole and inpaint along the layers.
A very promising approach has been presented by Mairal et al. [2]. The filling step is dictionary-based and the dictionary is learned with the K-SVD algorithm developed by Aharon et al. [14]. Mairal et al. extended the work of Aharon et al., which was origi-

nally designed for denoising of black-and-white images, to color images and the specific task of inpainting. The K-SVD algorithm is an "iterative method that alternates between sparse coding of the examples based on the current dictionary and a process of updating the dictionary atoms to better fit the data" [14]. The learned dictionary is later used to find sparse signal representations of the patch that has to be inpainted. As this method is the basis of this work, the idea and the algorithm are detailed in Section 2.3.
In order to make it more performant and robust, Mairal et al. [15] developed a multiscale version.
The publication that comes closest to this work is from Shen et al. [16]. They create a dictionary by direct sampling from the intact source regions. Priorities are assigned as presented by Criminisi et al. Then they treat the filling process as a signal recovery process where every patch can be described as a sparse representation of the dictionary. To solve the optimization problem of finding the dictionary coefficients over the minimization of the error subject to a certain sparsity, they use the Least Absolute Shrinkage and Selection Operator (LASSO) method.
The limitations of existing techniques, that are relevant for this work, are presented in Section 2.7.

1.3. Scope of the Report

This technical report describes a software for filling holes in images that occur during 3-D view synthesis. It will be based on existing work of Sparse Representations and boundary priorities. Further improvements such as multiscale and depth-layer-based inpainting will be investigated and added.
In order to be properly evaluated, the new concept will be implemented in Matlab in form of a toolbox. Knowing existing holes, the software will fill them with color information. The user can give input parameters and files through command line and an input file. The choice of parameters and the impact of the proposed improvements will be quantified. The performance of the new technique will be compared to existing methods through objective and subjective tests.

2. Theoretical Background

In the first part of this chapter, the motivation for why inpainting is an important challenge for 3-D video is presented (see Section 2.1). Due to the fact that a lot of signal and image processing happens in the human visual perception system, an introduction to human vision is given in Section 2.2. Next how the brain solves the problem of perceptual filling is presented. As the concept of Sparse Representations is a main focus of this work, its application in the visual cortex is explained. In Section 2.3, the mathematical theory behind Sparse Coding is derived and the concept described in detail. The other main concept of assigning filling priorities is explained in Section 2.4. More methods for multiscale and depth-based inpainting methods are detailled in Section 2.5 and 2.6. To conclude, the limitations of existing inpainting methods are exposed and further directions for this work are gathered (see Section 2.7).

2.1. View Synthesis for 3-D Video

One of the biggest success stories in 3-D video was the movie *James Cameron's Avatar* in 2010[1]. It took Cameron more than 7 years[2] to develop the stereoscopic camera system which made this movie so successful. The same year, major events were broadcast in 3-D (such as 2010 FIFA World Cup kickoff on ESPN) and the first 3D-TV channels went on air. By the end of 2012 the market share of 3-D capable devices in the global LCD TV panel shipments is estimated to be 25.7%[3]. The evolution from two-dimensional to 3-D video offers a new sensation of depth similar to that of the real world. This is based on the perception from two different projections of the scene by the the two eyes of the viewer. Whereas two-dimensional video is limited to monocular cues for depth perception such as linear perspective or occlusion, 3-D video allows binocular cues, mainly stereopsis.
3-D video capable LCD TV panels can be divided in two main categories: stereoscopic displays that require the viewer to wear eyewear and autostereoscopic displays which do not. Stereoscopic displays use various multiplexing technologies to transmit two different projections on the viewer's eye: color anaglyph systems, polarization or time multiplexing [17]. The last represents the major part of 3-D video capable LCD TV panels sales at the moment. Those technologies do not require complex processing as the used 3-D coding and formats only include stereo video data. Clearly the biggest inconvenience of stereoscopic systems is that the viewer has to wear eyewear. This problem has been partly solved with the emerging autostereoscopic displays. These displays allow the viewer to perceive

[1] http://www.boxofficemojo.com/alltime/world/
[2] http://video.google.com/videoplay?docid=-241532803911842846
[3] according to NPD DisplaySearch

two different projections of a scene with the help of parallax barriers or lenticular displays. The simplest application of these technologies are two-view displays where the pair of projections can only be perceived in one specific zone.
Multiview displays are a more advanced form where different stereo pairs can be perceived in a number of fixed zones. Another advantage of these displays is the realization of a "look around" effect. It means that the viewer can reveal background objects which are occluded by the foreground when changing the view perspective [18]. This requires an almost seamless transition from one view to another. As a result the disparity change between neighboring views should be kept as small as possible. Another challenge is that a good depth impression is desired and therefore this requires a large overall disparity range. The more views displayed, the higher the screen resolution required. Until now, no more than 50 views can be offered with each view in high resolution [18]. Compared to stereoscopic displays (with two views), the number of views needed for an autostereoscopic multiview display are significantly higher. One possibility to obtain them would be to create and transmit all views using the multiview coding profile of H.264/AVC [19]. It takes advantage of statistical dependencies between neighboring views. However the bitrate is linearly proportional to the number of views as shown in Fig. 2.1.

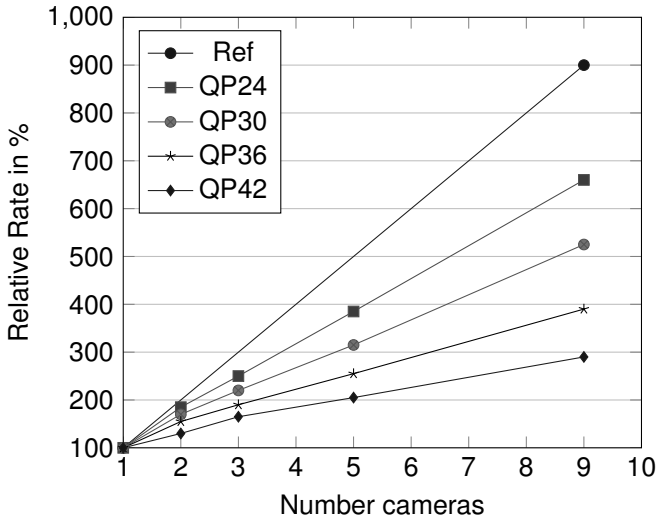


Figure 2.1.: Relative bitrate compared to one view with H.264/AVC MVC coding (adapted from [18]).

The bitrate can be radically reduced when transmitting the depth map in addition to the color information of one or two views. There are several methods which can synthesize additional views as described later. As this step takes place at the receiver, it could create different number of views for different displays. The corresponding format would be independent from the receiving device. The whole system is presented in Fig. 2.2.

Before transmitting additional depth maps, they have to be created first.
All the video material before the area of the new 3-D technology is filmed by monocular cameras. A depth map could be either estimated through automatic methods or manually

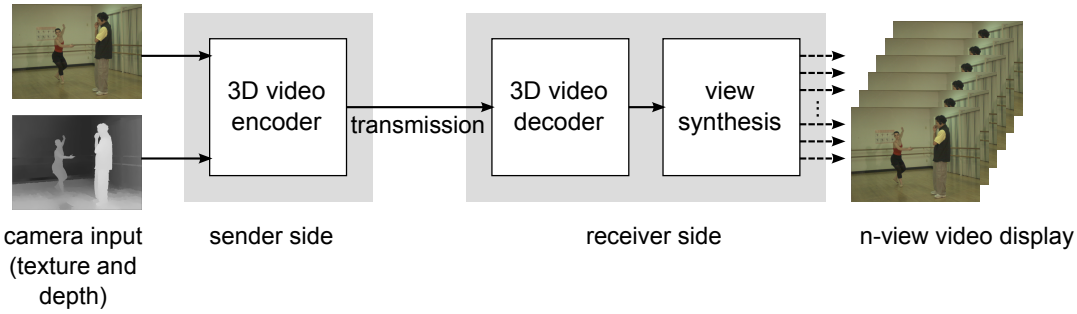


Figure 2.2.: 3-D Video system using depth maps.

by a user. The quality of the output of automatic methods is very limited whereas pure manual input is time-consuming. Semi-automatic conversion techniques where user input is propagated over several frames [20] provide the best trade-off. Current 3-D movies are filmed by modern stereoscopic cameras. The process of creating depth information in this setup is called Structure-from-Stereo (SfS). It consists usually of three steps: camera calibration, stereo matching and depth reconstruction [21]. The development of a novel camera system is part of the ATTEST project [3]. This project was started in 2002 sponsored by the European Commission with the goal of developping a flexible, 2D-compatible and commercially feasible 3-D TV system for broadcast environments. The new camera system uses an additional sensor for measuring the distance from the scene to the camera at each pixel. Such structured-light 3-D scanners are nowadays widely used in consumer products such as Microsoft Kinect.
After having established a way to create depth information that can be transmitted, additional views have to be synthesized at the receiver. A popular method is warping or morphing, where feature correspondences between a source and a target image are established. Based on this information, the rest of the image is "warped" using a deformation field [22]. Another technique which is also used in the ATTEST project is DIBR [23]. It consists of two steps: First, with the help of the depth data the color information is projected to a 3-D model. Then this 3-D model is projected to the image plane of a new virtual camera. This method will be the basis of this work.
Many challenges arise during this process but the focus of this work are so called occlusions. Regions that are occluded by foreground objects in one view can become visible in another (see Fig. 2.3). This is a major problem, especially in the case of only one transmitted view. The process of completing this missing information is called inpainting or hole filling.

2.2. Visual Perception System

As many existing methods in image processing rely on the human visual perception system, this should be the starting point to find an appropriate algorithm for the above mentioned task (Section 2.2.1). Interesting concepts on perceptual filling and completion in human

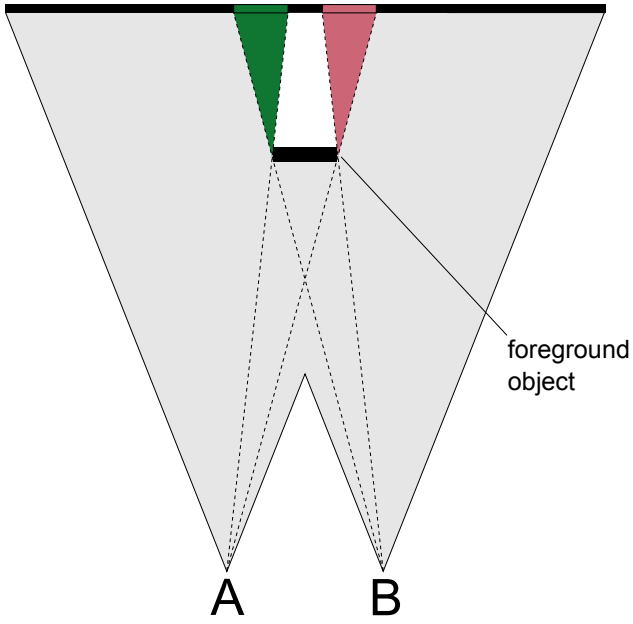


Figure 2.3.: Occlusion during DIBR. There is no color information for the red region from view A in the synthesized view B.

vision have been developed by several authors (Section 2.2.2). The powerful idea of Sparse Representations which became very popular in the domain of signal processing in recent years will be related to the visual perception system (Section 2.2.3).

2.2.1. Visual Pathway

Before trying to get a deeper understanding of how the visual system solves problems like hole filling and image completion, a brief summary of the visual pathway up to the visual cortex (Fig. 2.4) as presented by Schwabe et al. [24]:
When a mammalian perceives a natural scene, it will focus on the part that attracts the most attention. The light that falls into the eye, is refracted by cornea, pupil and lens and eventually reaches the retina. In binocular vision, an inverted representation of the natural scene is projected on each retina. The retina consists of approximately 10^8 photo receptor cells and transforms the incoming light intensity distribution in chemical and electrical events that trigger nerve impulses. These are transmitted by optic nerves (with approximately $1.2 * 10^6$ fibres) from every eye to the brain. Half of the nerves are going from each eye to the Lateral Geniculate Nucleus (LGN) on the opposite side of the brain according to the visual field crossing the Optic Chiasm. The other half connects every eye to the LGN on the same side. Subsequently the LGN on each side is connected to the primary visual cortex (V1) through the optic radiation. From the V1, there are two major output streams: one is going to higher visual areas in the cortical hierarchy (V2, V3, V4 and V5/MT) and the other one loops back to the LGN and other deep structures.

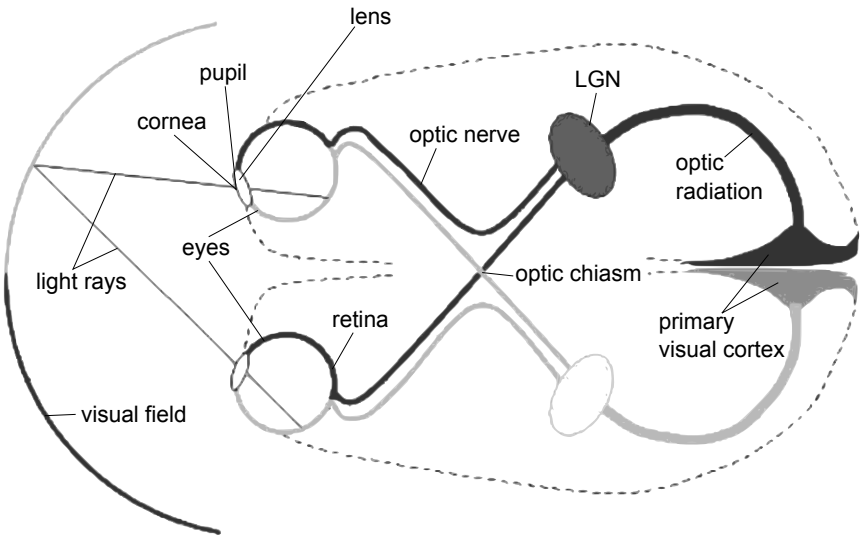


Figure 2.4.: The visual pathway up to the visual cortex from Schwabe et al. [24].

The encoding process of visual information of the retina and the LGN has interesting properties. The major part of photo reactive cells are "rods" (>90%) which are responsible for black-and-white vision in dim light. Furthermore there are "cones" that support daytime and color vision. After some first processing steps, the visual information is transmitted to the optic nerves through the gangilon cells. As there is only a fraction of optic nerves compared to the number of photo receptor cells on the retina, the information has to be compressed. This is done by the spatial combination of ON- and OFF-center cells, the so called receptive fields. A common representation of the receptive fields is a center cell (ON-center cell) surrounded by an annulus cell (OFF-center cell). The gangilon cells which transmit the information from the photoreactive cells to the optic nerves have a weak response on diffuse illumination whereas they respond strongly when light hits only the center or only the surround of the receptive field. This process is illustrated in Fig. 2.5.
The result is that the visual system responds selectively to edges and corners. This interesting property will be taken into account in the development of the inpainting algorithm.

2.2.2. Perceptual Filling and Completion in Human Vision

After having discussed the particular challenges of the inpainting process in 3-D view synthesis, the next step is to find out how the human visual perception system solves these problems.
An interesting phenomenon in human vision is the filling-in process of the blind spot on the retina. The blind spot is the region on the retina where the optic nerve is connected and has no photo receptors. However the blind spot is never perceived, even not under monocular viewing. This can easily be experienced in Fig. 2.6: When closing the left eye, the big gray dot on the left side that lies in the area of the blind spot should disappear and be filled

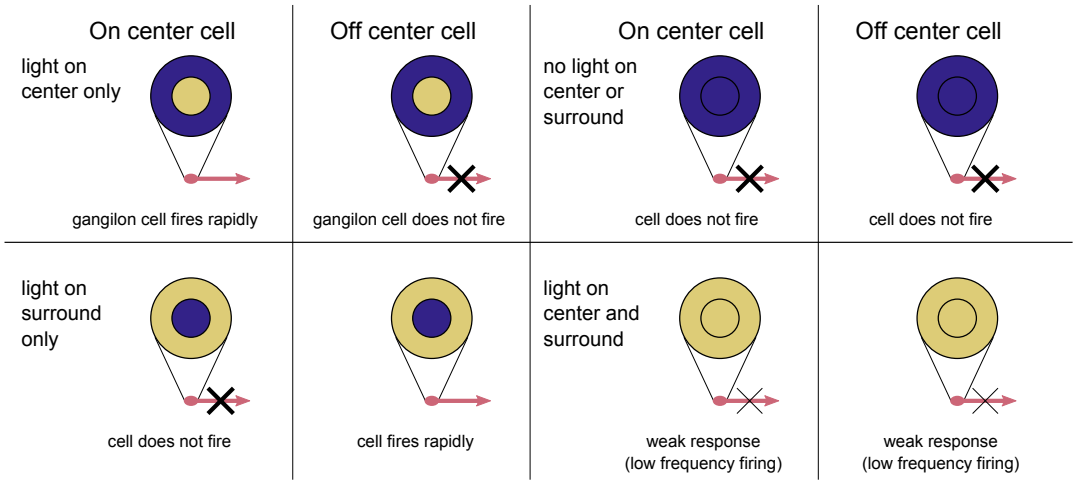


Figure 2.5.: ON/OFF ganglion cells respond differently to illumination configurations of the center and the surround.

in with the surrounding texture. The filling-in process happens within a few seconds and can be divided in 3 steps: First the big gray dot will be filled-in with the surrounding color information without the small black dots. Then the black dots start to appear in a flickering process. They stabilize and the viewer can not perceive any difference in color and texture in the area of the gray dot anymore. The authors of [25] suggest that there may be separate fill mechanisms for color and texture that could correspond to different areas of the visual cortex. There are many debates among visual scientist if there really is neural filling-in but many support this idea of a perceptual completion process which uses brain information to make up for an absence [26]. Others see the filling-in process as a result of neural activity that mis-attributes perceived signals at the border of the gray dot [25].

The above identified filling mechanisms can be seen as "featural completion" [26]. The idea behind this concept is that there are several distinguishable "visual primitives" out of which visual perception is composed. Furthermore there is the concept of boundary completion. The coexistence of both concepts can be experienced in the Kanizsa triangle (Fig. 2.7): There is boundary completion in form of illusionary contours form a triangle outline as wells as featural completion in form of an illusionary brightening of the triangle compared to the background.

In boundary completion two different types of illusory contours can be identified: Edge-induced illusory contours are collinear with the inducing edges (for example the edges induced through the partial disks in Kanizsa triangle). Line-induced illusory contours are perpendicular to the inducing lines which are typically thin. This can also be experienced in Kanizsa triangle.

All above cited filling mechanisms should be kept in mind for later use in the inpainting algorithm.

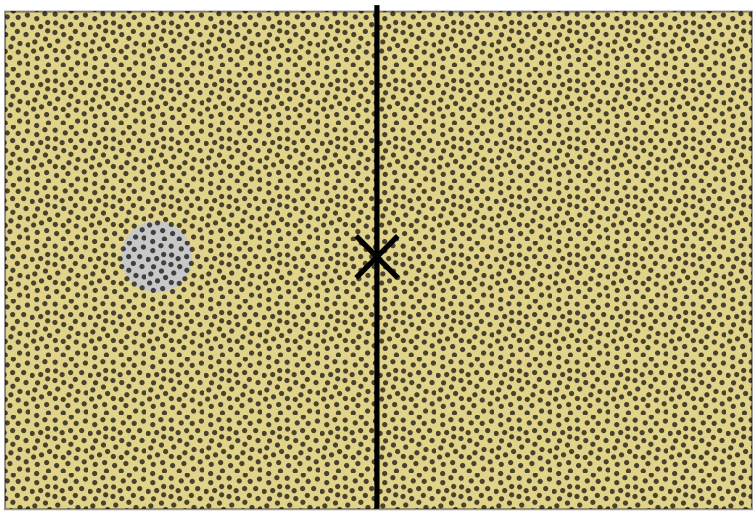


Figure 2.6.: Blind spot demonstration: Fix the cross and close your left eye. In a distance of approximately 20 cm to the page, the big gray dot on the side of the open eye should disappear and be filled in with the surrounding texture.

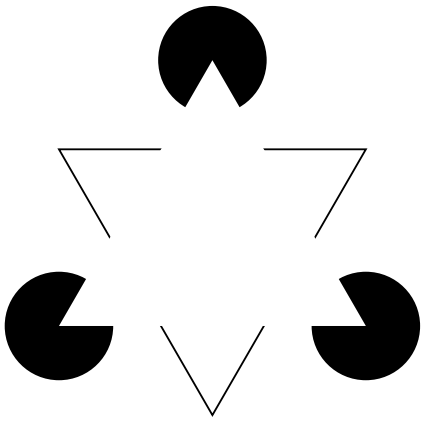


Figure 2.7.: The Kanizsa triangle. An illusory triangle can be perceived trough two different types of boundary completion: line-induced illusory contours with the thin lines and edge-induced illusory contours with the partial disks. Furthermore there is featural completition in form of an illusionary brightening of the triangle compared to the background.

2.2.3. Sparse Coding in the Visual Cortex

The information of natural images in the brain are coded in an efficient way after millions of years of evolution. Every item of information is represented by a pattern of active neurons (which can be seen as basis functions). Only a fraction of neurons are strongly active at any time. If this fraction is relatively small when representing items in a natural image it is called Sparse Coding (see Fig. 2.8). The concept of Sparse Coding applies to several layers of the visual cortex, for example in higher-level visual cortex, sparser codes could be

defined for higher probable items (e.g. faces) [27].
Advantages as well as experimental evidences of Sparse Coding in the visual perception system will be presented in the following.

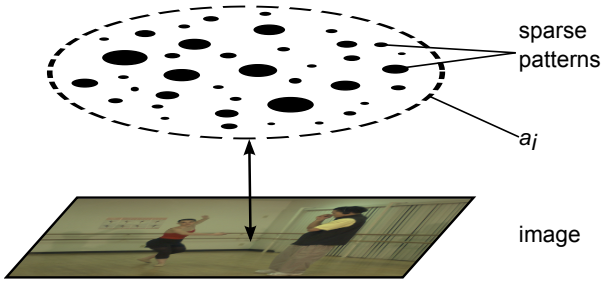


Figure 2.8.: Sparse coding. An image is only represented by a small number of neural patterns out of a bigger set a_i depending on the input image.

Advantages of Sparse Coding in Human Vision

The advantages of Sparse Coding were discussed in several publications ([28], [27], [24] and [29, Chapter 10.3]).
In Sparse Coding the sensory input is recoded in a way that preserves most of the information. At the same time, redundancy is reduced due to the statistical dependencies of the visual input signal. This way, an image may be described as a collection of independent events. The probabilistic model used to reduce the information of an image will be presented in Section 2.3.2. Sparse Coding is efficient in the sense that the code for an information is as short as possible. This way, it maximizes memory capacity and at the same time it minimizes the energetic cost in the biological systems as well as it allows faster reaction times.
Furthermore Sparse Coding makes the structure in natural images more explicit. Sparse Representations may require a large number of basis functions but at the same time only a small number is used to represent an information item. This minimizes interferences between different input patterns.
Another advantage is that Sparse Coding represents the complex data in natural images in an easier readable and processable way [27]. The data lies along a continuous curved manifold in the high-dimensional state space of images. Sparse Coding "flattens" out this curved manifold, thus makes it easier for the visual cortex to read and process it.

Sparse Coding with an Overcomplete Code In this context it should be mentioned that in the algorithms presented in Section 2.3 the used code is "overcomplete". This means that the number of basis functions is greater than the dimensionality of the input signals so that any signal can be represented by more than one combination of different basis functions. Sparse coding with an overcomplete set of basis functions has the advantage

that basis functions which are not needed to describe an image are wiped out through the sparsification in the learning process. In general an overcomplete code is more robust towards noise [27].

Experimental Evidence

When observing the activity of neurons of macaque monkeys watching a natural scene, the distribution of spikes (= active neurons) over a certain timeframe is close to being exponential ([29, Chapter 10.3]). This means that it is most likely that only a few neurons are activate at any given time. It is rare that a large number of neurons are active at the same time. Unlike a Gaussian distribution observed when applying principal component analysis to similar natural scenes, this can be interpreted as a Sparse distribution.
Vinje et al. have demonstrated that neurons in V1 produce sparse responses when being stimulated with natural images. The sparseness of the response is higher with natural images that include both the Classical Receptive Field (CRF) - "the region of space where stimuli evoke action potentials" - as well as the non-Classical Receptive Field (nCRF) - "where stimuli can modulate the responses evoked by CRF stimulation" [30].
Another evidence for the use of Sparse Coding by the perception system is that in the primary auditory cortex single spike responses to a certain sound can be observed [27].

2.3. Sparse Coding Algorithms and Dictionary Learning

2.3.1. Prerequisites

Sparse Coding in the visual cortex has already been presented in Section 2.2.3. This section gives a more mathematical description of Sparse Coding of images and dictionaries. Furthermore an algorithm that solves the NP-hard problems that occur in this context will be introduced which is at the heart of the later presented K-SVD algorithm.

Sparse Representations of Images

Color Image as a Signal The common representation of images in the digital world are bit-mapped graphics. The visual information is stored in the form of dot matrix data structure with a fixed size (= resolution). Color images are usually represented through the additive RGB (three channels: red, green, blue) color model. That means that for every pixel in the image, a color value is stored for every channel.
In the following paragraph, Sparse Representations of two-dimensional signals will be discussed. In order to make a proper transition, an example patch of the image of a certain size (e.g. 8x8) will be reshaped to a column vector. Therefore, every column of the red channel will be stored below each other. Beneath that the same will be done for the green and the blue channel (see Fig. 2.9).

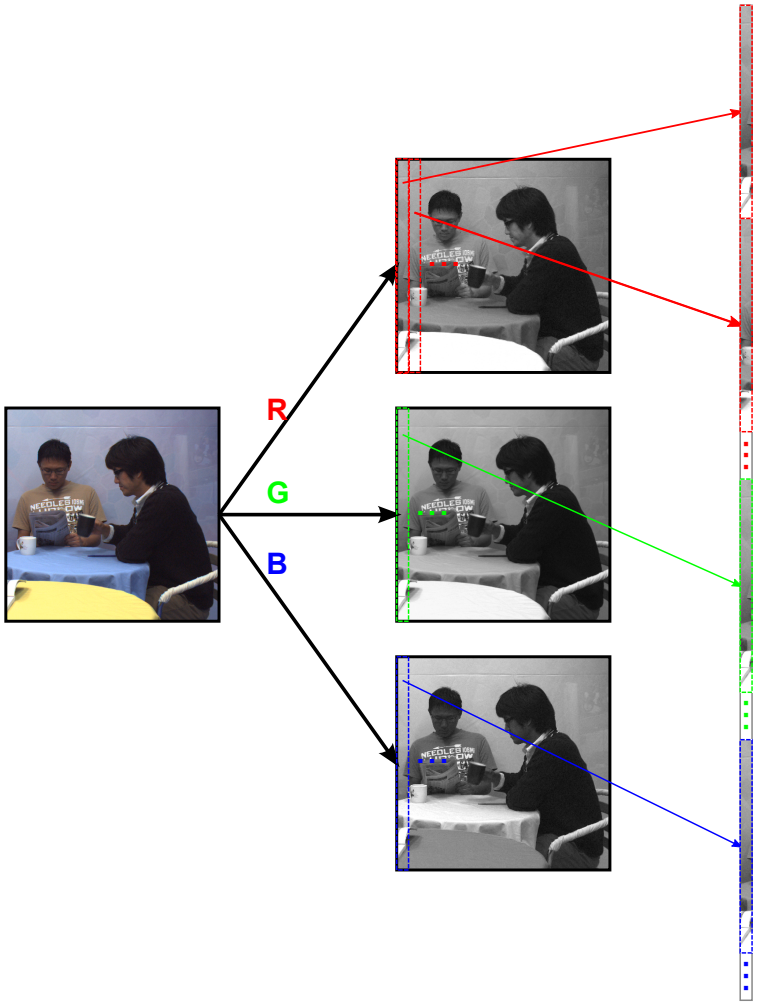


Figure 2.9.: Color image as a signal. A color image is separated in the red, green and blue channel. To obtain a vector representation, every channel is reshaped.

Sparse Representations of Signals A signal $\mathbf{y} \in \mathbb{R}^n$ can be represented by the linear combination of so called "atoms" of an overcomplete "dictionary" $\mathbf{D} \in \mathbb{R}^{n \times K}$

$$\mathbf{y} = \mathbf{D}\mathbf{x}. \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^K$ contains the representation coefficients of $\mathbf{y}$. When $n < K$ and $\mathbf{D}$ has full rank, there is an infinite number of solutions for $\mathbf{x}$. A "sparse representation" is the solution with a very small number of non-zero coefficients ($< \frac{1}{10}$ of the dictionary size) in $\mathbf{x}$. This can be formulated as a minimization problem with respect to the approximation error of $\|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2$ and a maximal sparsity L

$$\min_{\mathbf{D},\mathbf{x}} \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}\|_0 \leq L \tag{2.2}$$

where $\|\cdot\|_0$ is the ℓ^0 norm which only counts the nonzero elements of a vector.

Solving an NP-hard Problem: Orthogonal Matching Pursuit

The optimization problem from Equation 2.2 is generally NP-hard. This means that it can only be solved in superpolynomial time since there is a known polynomial-time algorithm for a non-deterministic machine to find the solution.
A solution for NP-hard problems can be found through greedy algorithms such as Orthogonal Matching Pursuit (OMP) ([31], [32, Chapter 3.1]). It finds the best matching orthogonal projection of $\mathbf{y}$ onto the dictionary $\mathbf{D}$. A step by step description of the OMP algorithm is presented in 2.10. Hereby $\mathbf{d}_j$ stands for a column vector of the dictionary matrix $\mathbf{D}$.

Task: Approximate the solution of $\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}\|_0 \leq L$.
Parameters: We are given the matrix $\mathbf{D}$, the vector $\mathbf{y}$, and the maximal sparsity L.
Initialization: Initialize $t = 0$ and set

- The initial solution $\mathbf{x}_0 = 0$.
- The initial residual $\mathbf{r}_0 = \mathbf{y}$.
- The initial measurement matrix $\mathbf{\Phi} = 0$.

Main Iteration: Increment t by 1 and perform the following steps:

- **Sweep**: Find the optimal choice of atom $\mathbf{d}_j$ which is closest to $\mathbf{r}_{t-1}$:

$$\lambda_t = \arg \max_{j=1,\dots,K} | \langle \mathbf{r}_{t-1}, \mathbf{d}_j \rangle | \quad \text{with } \|\mathbf{d}_j\|_2 = 1$$

- **Update measurement matrix**: Extend the measurement matrix by the obtained column $\mathbf{d}_{\lambda_t}$:

$$\mathbf{\Phi}_t = \left[\mathbf{\Phi}_{t-1} | \mathbf{d}_{\lambda_t}\right]$$

- **Update provisional solution**: Find a minimizer $\mathbf{x}_t$ of the measurement matrix:

$$\mathbf{x}_t = \arg \min_{\mathbf{x}} \|\mathbf{y} - \mathbf{\Phi}_t \mathbf{x}\|$$

- **Update the residual**: Compute $\mathbf{r}_t = \mathbf{y} - \mathbf{\Phi}_t \mathbf{x}_t$
- **Stopping Rule**: If $t > L$, stop.

Output: After t iterations, a solution $\mathbf{x}_t$ for $\mathbf{x}$ is obtained.

Figure 2.10.: Orthogonal Matching Pursuit. A greedy algorithm for NP-hard problems (from [31]).

2.3.2. Probabilistic Approach: Bayesian Learning

The probabilistic approach was first presented by Olshausen et al. [27]. Aharon et al. [14] took Olshausen's and several other authors' work as a basis for the development of the

K-SVD algorithm. Below is an overview:
Olshausen et al. added white, additive, Gaussian noise to the model of Equation 2.1 such that

$$\mathbf{y} = \mathbf{D}\mathbf{x} + \mathbf{v} \tag{2.3}$$

with $\mathbf{y}$ being one sample.
The goal is to find a dictionary $\mathbf{D}^*$ that "can best account for the structure in images in terms of a linear superposition of sparse statistically independent events." [27]. In probability theory this means that the idea is to match the distribution of images $P(\mathbf{Y}|\mathbf{D})$ arising from the linear model to the actual distribution of natural images $P^*(\mathbf{Y})$ where $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$ are all samples.
Three important assumptions were made:

1. The measurements are independent and therefore the probability $P(\mathbf{Y}|\mathbf{D})$ is

$$P(\mathbf{Y}|\mathbf{D}) = \prod_{i=1}^{N} P(\mathbf{y}_i|\mathbf{D}). \tag{2.4}$$

2. The probability $P(\mathbf{y}_i|\mathbf{x},\mathbf{D})$ of an image arising from certain coefficients is

$$P(\mathbf{y}_i|\mathbf{x},\mathbf{D}) = Const \cdot e^{\frac{1}{2\sigma^2}\|\mathbf{D}\mathbf{x}-\mathbf{y}_i\|_2^2}. \tag{2.5}$$

3. The prior distribution of the representation vector $\mathbf{x}$ is parameterized as

$$P(\mathbf{x}) = Const \cdot e^{\lambda S(\mathbf{x})} \tag{2.6}$$

where the $S(\mathbf{x})$ determines the shape of the distribution.

The probability of an image arising from the probabilistic model is

$$P(\mathbf{y}_i|\mathbf{D}) = \int P(\mathbf{y}_i|\mathbf{x},\mathbf{D})P(\mathbf{x})d\mathbf{x}. \tag{2.7}$$

In order to evaluate how well $P(\mathbf{Y}|\mathbf{D})$ matches $P^*(\mathbf{Y})$, Olshausen et al. use the Kullback-Leibler divergence

$$KL = \int P^*(\mathbf{Y}) \log \frac{P^*(\mathbf{Y})}{P(\mathbf{Y}|\mathbf{D})} d\mathbf{Y} \tag{2.8}$$

between the two distributions. The smaller KL is, the more similar the distributions are. As $P^*(\mathbf{Y})$ is fixed, minimizing Equation 2.8 turns into maximizing $\langle log\, P(\mathbf{Y}|\mathbf{D})\rangle = \int P^*(\mathbf{Y}) \log P(\mathbf{Y}|\mathbf{D}) d\mathbf{Y}$. From this optimization problem the dictionary

$$\mathbf{D}^* = \arg\max_{\mathbf{D}} \langle \log P(\mathbf{Y}|\mathbf{D}))\rangle . \tag{2.9}$$

is obtained. When integrating the assumptions of Equation 2.5 and 2.6 into Equation 2.7 such that

$$P(\mathbf{y}_i|\mathbf{D}) = Const \cdot \int e^{\frac{1}{2\sigma^2}\|\mathbf{D}\mathbf{x}-\mathbf{y}_i\|_2^2} \cdot e^{\lambda S(\mathbf{x})} d\mathbf{x}, \tag{2.10}$$

evaluating the integral becomes difficult. Olshausen et al. solved this by approximating the integral by its maximum.
The overall problem of Equation 2.9 becomes

$$\mathbf{D}^* = \arg\max_{\mathbf{D}} \left\langle \max_{\mathbf{x}} \log P(\mathbf{Y}|\mathbf{x},\mathbf{D}) P(\mathbf{x}) \right\rangle. \tag{2.11}$$

With the independence assumption from Equation 2.4 the problem turns into

$$\mathbf{D}^* = \arg\max_{\mathbf{D}} \sum_{i=1}^{N} \max_{\mathbf{x}_i} P(\mathbf{y}_i, \mathbf{x}_i|\mathbf{D}) = \arg\min_{\mathbf{D}} \sum_{i=1}^{N} \min_{\mathbf{x}_i} \left(\|\mathbf{D}\mathbf{x}_i - \mathbf{y}_i\|_2^2 + \lambda S(\mathbf{x}) \right). \tag{2.12}$$

The minimization problem with respect to $\|\mathbf{D}\mathbf{x}_i - \mathbf{y}_i\|_2^2 + \lambda S(\mathbf{x})$ is similar to applying the method of Lagrange multipliers to Equation 2.2 when $S(\mathbf{x}) = \|\mathbf{x}\|_0 - L$. This nested minimization problems consists of two parts:

1. In the inner part the expression is minimized over $\mathbf{x}_i$ and a fixed dictionary $\mathbf{D}$.
2. In the outer part, the expression is minimized with respect to the dictionary $\mathbf{D}$.

The same structure can be later found in the K-SVD algorithm.

2.3.3. The K-SVD Algorithm

Learning an Overcomplete Dictionary with K-SVD

The K-SVD algorithm has been first presented by Aharon et al. [14]. The starting point is the same as Equation 2.2 or the result obtained through the probabilistic approach of Section 2.3.2. The algorithm searches for the best possible dictionary for the sparse representation of $\mathbf{Y}$

$$\min_{\mathbf{D},\mathbf{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 \quad \text{subject to} \quad \forall i,\ \|\mathbf{x}_i\|_0 \leq L. \tag{2.13}$$

with $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$.
First, every representation vector $\mathbf{x}_i$ of $\mathbf{X}$ is obtained by applying the OMP algorithm of Fig. 2.10. The next step is to update the dictionary $\mathbf{D}$. Therefore $\mathbf{X}$ is fixed as well as $\mathbf{D}$ except one column $\mathbf{d}_k$. The corresponding row in $\mathbf{X}$ is $\mathbf{x}_T^k$ (T stands for transposed, as it is a row and not a column vector). This gives the possibility to introduce a penalty term $\mathbf{E}_k$ such that

$$\|\mathbf{Y} - \mathbf{DX}\|_F^2 = \left\|\mathbf{Y} - \sum_{j=1}^{K} \mathbf{d}_j \mathbf{x}_T^j\right\|_F^2 = \left\|\left(\mathbf{Y} - \sum_{j\neq k} \mathbf{d}_j \mathbf{x}_T^j\right) - \mathbf{d}_k \mathbf{x}_T^k\right\|_F^2 = \left\|\mathbf{E}_k - \mathbf{d}_k \mathbf{x}_T^k\right\|_F^2. \tag{2.14}$$

A common method to minimize the above problem and obtain an alternative $\mathbf{d}_k$ and $\mathbf{x}_T^k$ would be Singular Value Decomposition (SVD) but $\mathbf{x}_T^k$ would not be likely to be sparse. To handle this constraint, Aharon et al. introduced a matrix $\mathbf{\Omega}_k$ which applied to $\mathbf{x}_T^k$ only keeps the non-zero entries: $\mathbf{x}_R^k = \mathbf{x}_T^k \mathbf{\Omega}_k$. The matrix "points" on the values that use the $\mathbf{d}_k$ atom. Similarly, the multiplications $\mathbf{Y}_k^R = \mathbf{Y}\mathbf{\Omega}_k$ and $\mathbf{E}_k^R = \mathbf{E}_k \mathbf{\Omega}_k$ create a matrix that only includes a subset of the examples respectively of the error columns that correspond to examples that use the $\mathbf{d}_k$ atom. Thus, Equation 2.14 turns into

$$\left\|\mathbf{E}_k \mathbf{\Omega}_k - \mathbf{d}_k \mathbf{x}_T^k \mathbf{\Omega}_k\right\|_F^2 = \left\|\mathbf{E}_k^R - \mathbf{d}_k \mathbf{x}_R^k\right\|_F^2. \tag{2.15}$$

The minimization problem

$$\arg\min_{\mathbf{d}_k, \mathbf{x}_R^k} \left\|\mathbf{E}_k^R - \mathbf{d}_k \mathbf{x}_R^k\right\|_F^2 \tag{2.16}$$

to obtain an alternative $\mathbf{d}_k$ and $\mathbf{x}_R^k$ can be solved trough SVD

$$\mathbf{E}_k^R = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T. \tag{2.17}$$

The first column of $\mathbf{U}$ is the solution for $\mathbf{d}_k$ and the first column of $\mathbf{V}$ multiplied by the first eigenvalue of $\mathbf{\Sigma}$ the solution for $\mathbf{x}_R^k$.
The algorithm as a whole is presented in Fig. 2.11.

K-SVD for Color Image Inpainting

The above-mentioned algorithm has been successfully tested with black-and-white images with uniform white Gaussian noise. For the specific task of color image inpainting, the K-SVD algorithm has to be modified in two ways:

1. Adapt the algorithm to handle color images.
2. Find a way to handle non-uniform noise (in the case of this report: holes from view reconstruction).

Solutions for both challenges have been presented by Mairal et al. [2].
Going from black-and-white to color is straight forward and follows the same logic as in Section 2.3.1. A color image patch $\mathbf{y} \in \mathbb{R}^{n\times n\times 3}$ is reshaped in a column vector $\tilde{\mathbf{y}} \in \mathbb{R}^{n^2*3\times 1}$ such that every channel is below the previous one.
More modifications to the existing algorithm need to be made in order to handle holes coming from view synthesis. Mairal et al. proposed to introduce a $\boldsymbol{\beta} \in \mathbb{R}^{n^2*3\times 1}$ vector for each example patch such that

Task: Find the best dictionary to represent the data sample $\{\mathbf{y}_i\}_{i=1}^N$ as sparse compositions by solving

$$\min_{\mathbf{D},\mathbf{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 \quad \text{subject to} \quad \forall i,\ \|\mathbf{x}_i\|_0 \leq L.$$

Parameters: We are given the matrix $\mathbf{Y}$, the maximal sparsity L and an initial dictionary $\mathbf{D}^{(0)}$.
Initialization: Initialize $J = 1$ and set the dictionary matrix $\mathbf{D}^{(0)} \in \mathbb{R}^{n \times K}$ with ℓ^2 normalized columns.
Main Iteration: Repeat until convergence (stopping rule):

- **Sparse Coding Stage**: Apply OMP to compute the representation vectors $\mathbf{x}_i$ for each example $\mathbf{y}_i$ by approximating the solution of

$$i = 1, 2, ..., N \quad \min_{\mathbf{x}_i} \|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_F^2 \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \leq L$$

- **Codebook Update Stage**: For each column $k = 1, 2, ..., K$ in $\mathbf{D}^{(J-1)}$, update it by
  - Computing the overall representation error matrix, $\mathbf{E}_k$, by

$$\mathbf{E}_k = \mathbf{Y} - \sum_{j \neq k} \mathbf{d}_j \mathbf{x}_T^j$$

  - Restricting $\mathbf{E}_k$ by choosing only the columns that correspond to examples that use the $\mathbf{d}_k$ atom and obtain $\mathbf{E}_k^R$
  - Applying SVD $\mathbf{E}_k^R = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. The updated dictionary column $\mathbf{d}_k$ is the first column of $\mathbf{U}$. The updated representation vector $\mathbf{x}_R^k$ is the first column of $\mathbf{V}$ multiplied by the first eigenvalue of $\mathbf{\Sigma}$.
- **Incrementation**: Set $J = J + 1$.

Output: After $J + 1$ iterations, an updated dictionary $\mathbf{D}^{(J)}$ is obtained.

Figure 2.11.: The K-SVD algorithm. Adapted from [14]

.

$$\forall i \in 1, ..., n^2 * 3 \quad \beta_i = \begin{cases} 0, & \text{Pixel is a "hole"} \\ 1, & \text{Pixel is not a "hole"} \end{cases} \tag{2.18}$$

The idea is to mask the missing information when feeding the sample patches to the OMP algorithm. Therefore the OMP algorithm has to be modified as presented in Fig. 2.12.

The K-SVD algorithm has to be modified as well. The minimization problem of Equation 2.16 has to be modified to incorporate β

$$\arg\min_{\mathbf{d}_k, \mathbf{x}_R^k} \left\| \beta \times \left(\mathbf{E}_k^R - \mathbf{d}_k \mathbf{x}_R^k \right) \right\|_F^2. \tag{2.19}$$

Srebro et al. [33] developed an iterative algorithm which gives an approximate solution of a local minimum for this so called "weighted rank-one approximation" problem. Mairal et al. introduced an algorithm that applies Srebro's method to the minimization problem of Equation 2.19 (see Fig. 2.13).

Task: Approximate the solution of $\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}\|_0 \leq L$.
Parameters: We are given the matrix $\mathbf{D}$, the vector $\mathbf{y}$, and the maximal sparsity L.
Initialization: Initialize $t = 0$ and set

- The initial solution $\mathbf{x}_0 = 0$.
- The initial residual $\mathbf{r}_0 = \mathbf{y}$.
- The initial measurement matrix $\mathbf{\Phi} = 0$.

Main Iteration: Increment t by 1 and perform the following steps:

- **Sweep**: Find the optimal choice of atom $\mathbf{d}_j$ which is closest to $\mathbf{r}_{t-1}$:

$$\lambda_t = \arg\max_{j=1,\ldots,K} | \langle \boldsymbol{\beta} \otimes \mathbf{r}_{t-1}, \mathbf{d}_j \rangle | \quad \text{with } \|\mathbf{d}_j\|_2 = 1$$

- **Update measurement matrix**: Extend the measurement matrix by the obtained column $\mathbf{d}_{\lambda_t}$:

$$\mathbf{\Phi}_t = \left[\mathbf{\Phi}_{t-1} | \mathbf{d}_{\lambda_t}\right]$$

- **Update provisional solution**: Find a minimizer $\mathbf{x}_t$ of the measurement matrix:

$$\mathbf{x}_t = \arg\min_{\mathbf{x}} \|\boldsymbol{\beta} \otimes \mathbf{y} - \boldsymbol{\beta} \otimes \mathbf{\Phi}_t \mathbf{x}\|$$

- **Update the residual**: Compute $\mathbf{r}_t = \mathbf{y} - \mathbf{\Phi}_t \mathbf{x}_t$
- **Stopping Rule**: If $t > L$, stop.

Output: After t iterations, a solution $\mathbf{x}_t$ for $\mathbf{x}$ is obtained.

Figure 2.12.: Orthogonal Matching Pursuit for non-uniform noise. Introducing a $\boldsymbol{\beta}$ vector for missing image information.

Task: Solve the optimization problem $\arg\min_{\mathbf{x}_R^k, \|\mathbf{d}_k\|_2=1} \left\|\boldsymbol{\beta} \times \left(\mathbf{E}_k^R - \mathbf{d}_k \mathbf{x}_R^k\right)\right\|_F^2$.
Parameters: We are given the error matrix $\mathbf{E}_k^R$, the mask matrix $\boldsymbol{\beta}$, the dictionary atom $\mathbf{d}_k$ and the number of iterations *iter*.
Initialization: Initialize $\mathbf{d}_{new} = 0$, $\mathbf{x}_{new} = 0$.
Main Iteration: For *i* from 1 to *iter*, use a truncated SVD to solve:

$$(\mathbf{d}_{old}, \mathbf{x}_{old}) = (\mathbf{d}_{new}, \mathbf{x}_{new})$$

$$(\mathbf{d}_{new}, \mathbf{x}_{new}) = \arg\min_{\mathbf{x}_R^k, \|\mathbf{d}_k\|_2=1} \left\|\boldsymbol{\beta} \otimes \mathbf{E}_k^R + (\mathbb{1} \times \mathbb{1} - \boldsymbol{\beta}) \otimes \mathbf{d}_{old} \mathbf{x}_{old}^T - \mathbf{d}_k \mathbf{x}_R^k\right\|_F^2$$

Output: After *iter* iterations an updated $\mathbf{d}_{new}$ and $\mathbf{x}_{new}$ are obtained.

Figure 2.13.: Weighted rank-one approximation algorithm used for non-uniform noise from Mairal et al. [2].

Furthermore Mairal et al. proposed to use an initial dictionary learned from natural images. The reason is that "some β-coefficients equal to zero could mask parts of unnatural

atoms, which could end up being used" [2].
The inpainting process is straight forward given the learned dictionary and the image to inpaint plus the corresponding holemask. The image and the holemask are divided into several patches that can superpose each other. Using the OMP algorithm for non-uniform noise presented in 2.12, a coefficient vector x is computed. This solution together with the learned dictionary gives the solution for the image patch. When using superposed image patches, the mean has to be taken from the multiple results for every pixel.

2.4. Texture- and Structure-based Inpainting

The filling order in simple exemplar-based methods (as well as in K-SVD) usually does not play any role. The inpainting step is executed in pixelwise overlapping patches and then the mean is taken. This often results in blurred details as they may be reconstructed in only a few of the overlapping patches.
Boundary and featural completion as introduced in Section 2.2.2 are important mechanisms regarding how the human visual perception system solves inpainting problems. Criminisi et al. [1] developed an algorithm which can be seen as an artificial implementation of these mechanisms. The algorithm performs exemplar-based inpainting that conserves textures but with respect to surrounding linear structures. Furthermore the algorithm gives higher priority to more reliable regions as hole corners during the filling step.
A target region for inpainting is denoted as Ω where its contour is $\delta\Omega$ (see Fig. 2.14). The source region for image patches is indicated by Φ. The image patch with partly missing information that should be filled is denoted by $\Psi_{\mathbf{p}} \in \Omega$ and centered at the point $\mathbf{p}$, $\mathbf{n_p}$ is the normal to the contour and $\nabla I_{\mathbf{p}}^{\perp}$ the isophote (direction and intensity) at point $\mathbf{p}$.

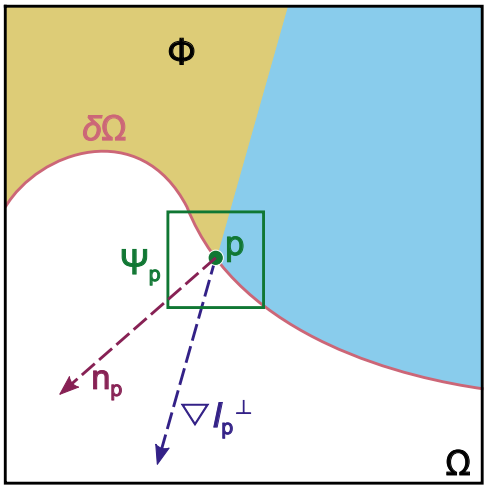


Figure 2.14.: Notation diagram for the algorithm of Criminisi et al. Ω is the target region, $\delta\Omega$ its contour, Φ the source region, $\Psi_{\mathbf{p}} \in \Omega$ the image patch to inpaint and $\mathbf{n_p}$ the normal to the contour and $\nabla I_{\mathbf{p}}^{\perp}$ the isophote at point $\mathbf{p}$.

The algorithm proceeds layer-wise with the inpainting over the fill front. In every iteration, filling priorities are assigned along the fill front and at the point **p** with the highest priority, exemplar-based inpainting is performed.
A priority at a point **p** is defined as

$$P(\mathbf{p}) = C(\mathbf{p}) \cdot D(\mathbf{p}) \tag{2.20}$$

where $C(\mathbf{p})$ is the "confidence term" and $D(\mathbf{p})$ the "data term". The confidence

$$C(\mathbf{p}) = \frac{\sum_{\mathbf{q} \in \Psi_\mathbf{p} \cap \overline{\Omega}} C(\mathbf{q})}{|\Psi_\mathbf{p}|} \tag{2.21}$$

indicates the amount of reliable information around a point **p**.
The patches that have already more pixels filled then others should be processed first. Before starting the algorithm, $C(\mathbf{p})$ is initialized as $C(\mathbf{p}) = 0 \; \forall \mathbf{p} \in \Omega$ and $C(\mathbf{p}) = 1 \; \forall \mathbf{p} \in \Phi$. The data term $D(\mathbf{p})$ is defined as

$$D(\mathbf{p}) = \frac{\left| \nabla I_\mathbf{p}^\perp \cdot n_\mathbf{p} \right|}{\alpha} \tag{2.22}$$

where α is a normalization factor. It computes the strength of isophotes at the contour $\delta\Omega$ and gives higher priority to linear structures.
After having computed the priorities of the fill front during an iteration, the patch with the highest priority is selected to be inpainted. This is done by directly copying the most similar patch $\Psi_{\hat{\mathbf{q}}}$ of the source region. The similarity is measured over the SSD between the already filled pixels of the two patches such that

$$\Psi_{\hat{\mathbf{q}}} = \arg \min_{\Psi_\mathbf{q} \in \Phi} d\left(\Psi_{\hat{\mathbf{p}}}, \Psi_\mathbf{q}\right). \tag{2.23}$$

Before continuing with the next iteration, the confidence values have to be updated.
The whole algorithm is presented in Fig. 2.15.

The idea of Criminisi et al. has become very popular and has been widely adopted throughout the research community. Many improvements have been proposed including modifying the combination of the data and the confidence term ([4]) or changing the metric of the data term ([6]). As those improvements have a limited impact on the results, methods that take advantage of depth information or use a multiscale approach are more important. Some of them will be presented in the following two sections.

2.5. Incorporating Depth Information in the Inpainting Process

A main concept of the ATTEST project and of more recent developments on 3-D-Video compression and transmission codecs is the existence of a depth map of the captured scene as described in Section 2.1. This additional information can add substantial improvements to the inpainting process.
Several researchers have already tried out different approaches to take advantage of depth

- Extract the manually selected initial fill front $\delta\Omega^0$.
- Repeat until done:
  1. Identify the fill front $\delta\Omega^t$. If $\Omega^t = \varnothing$, exit.
  2. Compute priorities $P(\mathbf{p})\ \forall \mathbf{p} \in \delta\Omega^t$.
  3. Find the patch $\Psi_{\hat{\mathbf{p}}}$ with the maximum priority
  $$\Psi_{\hat{\mathbf{p}}} | \hat{\mathbf{p}} = \arg \max_{\mathbf{p} \in \delta\Omega^t} P(\mathbf{p}).$$
  4. Find the exemplar $\Psi_{\hat{\mathbf{q}}} \in \Phi$ that minimizes $d\left(\Psi_{\hat{\mathbf{p}}}, \Psi_{\hat{\mathbf{q}}}\right)$.
  5. Copy image data from $\Psi_{\hat{\mathbf{q}}}$ to $\Psi_{\hat{\mathbf{p}}}$.
  6. Update $C(\mathbf{p})\ \forall \mathbf{p}\ | \mathbf{p} \in \Psi_{\hat{\mathbf{p}}} \cap \Omega$

Figure 2.15.: The inpainting algorithm of Criminisi et al. [1]

information. There are two main ways: The depth information can be used either to determine the filling priority or during the filling process itself.
Daribo et al. [5], [34] use it in both steps. Their method is based on Criminisi's method (see Section 2.4) and adds another term related to the depth to Equation 2.20 such that the priority $P(\mathbf{p})$ of a point $\mathbf{p}$ becomes

$$P(\mathbf{p}) = C(\mathbf{p}) \cdot D(\mathbf{p}) \cdot L(\mathbf{p}) \tag{2.24}$$

where $L(\mathbf{p})$ is defined as

$$L(\mathbf{p}) = \frac{|Z_{\mathbf{p}}|}{|Z_{\mathbf{p}}| + \sum_{\mathbf{q} \in Z_{\mathbf{p}} \cap \overline{\Omega}} \left(Z_{\mathbf{p}}(\mathbf{q}) - \bar{Z}_{\mathbf{p}}\right)^2} \tag{2.25}$$

where $|Z_{\mathbf{p}}|$ is the number of pixels of the actual patch $Z_{\mathbf{p}}$ centered in $\mathbf{p}$, $Z_{\mathbf{p}}(\mathbf{q})$ is the depth value of a point $\mathbf{q}$ in $Z_{\mathbf{p}}$ and $\bar{Z}_{\mathbf{p}}$ the mean value of the patch.
To find an appropriate match, Daribo et al. apply a modified version of SSD. To the minimization term of Equation 2.23 that is based on the texture, they added a cost function $d\left(Z_{\hat{\mathbf{p}}}, Z_{\mathbf{q}}\right)$ that measures the distance of the depth values such that the most similar patch is found through

$$\Psi_{\hat{\mathbf{q}}} = \arg \min_{\Psi_{\mathbf{q}} \in \Phi} d\left(\Psi_{\hat{\mathbf{p}}}, \Psi_{\mathbf{q}}\right) + \beta \cdot d\left(Z_{\hat{\mathbf{p}}}, Z_{\mathbf{q}}\right) \tag{2.26}$$

where β is a parameter that controls the importance given to the depth information. Daribo et al. continue working on their algorithm and an improved version will be published on the upcoming 3DTV Conference in October 2012.
Modifying Criminisi's approach and using depth information in both steps of the inpainting process has also been done by Gautier et al. [6]. To determine the filling order, they modify Equation 2.22 such that a channel with an additional image component Z that contains the depth values is added to the RGB-channels of every patch. This way the isophote $\nabla I_{\mathbf{p}}^{\perp}$ is

computed over all 4 channels. Furthermore during the patch matching process Gautier et al. define the cost function $d\left(\Psi_{\hat{\mathbf{p}}}, \Psi_{\mathbf{q}}\right)$ from Equation 2.23 as

$$d\left(\Psi_{\hat{\mathbf{p}}}, \Psi_{\mathbf{q}}\right) = \sum_{\mathbf{p},\mathbf{q} \in \Psi_{\mathbf{p},\mathbf{q}} \cap \Phi} \alpha_I \left\| \Psi_{\hat{\mathbf{p}}} - \Psi_{\mathbf{q}} \right\|^2 \tag{2.27}$$

where α_I sets the depth channel as important as the color channels ($I \in R, G, B, Z$ with $\alpha_{R,G,B} = 1$ and $\alpha_Z = 3$). The result is that the search of matching patches is restricted to a certain depth range.
The concept of using depth information for the patch matching process has also been applied by Luo et al. [4]. Their version of the cost function

$$d\left(\Psi_{\hat{\mathbf{p}}}, \Psi_{\mathbf{q}}\right) = SAD(\Psi_{\hat{\mathbf{p}}}, \Psi_{\mathbf{q}}) + \lambda \left| mean\left(depth\left(\Psi_{\hat{\mathbf{p}}}\right)\right) - mean\left(depth\left(\Psi_{\mathbf{q}}\right)\right) \right| \tag{2.28}$$

takes the difference of the means and weights it with a Lagrange multiplier λ, which is defined as

$$\lambda = \begin{cases} \lambda_b, mean\left(depth\left(\Psi_{\hat{\mathbf{p}}}\right)\right) \leq mean\left(depth\left(\Psi_{\mathbf{q}}\right)\right) \\ \lambda_f, mean\left(depth\left(\Psi_{\hat{\mathbf{p}}}\right)\right) > mean\left(depth\left(\Psi_{\mathbf{q}}\right)\right) \end{cases} \tag{2.29}$$

where λ_f is bigger then λ_b such that through Equation 2.28 patches from the background are favored.
An interesting approach that includes depth information has been developed by Oh et al. [8]: They separate foreground from background information using the depth data and "replace the boundary region bordering on foreground to the background region located on the opposite side". This method gives more natural results as there is no false propagation from foreground information to holes in the background during the inpainting process.
A depth-guided exemplar-based inpainting algorithm was part of a complete view synthesis system from Cheng et al. [35]. They only consider background information for the patch matching process as for the particular task of view synthesis this seems logical to them.
A concept that involves a different technique has been presented by Hervieux et al. [13]: They segment the color information and then with the help of the depth information approximate a 3-D plane for every color segment using RANdom Sample And Consensus (RANSAC). To obtain a better approximation of the 3-D structure of the scene, they compute another segmentation based on the depth data. The main inpainting step is realized with Criminisi's standard method.
Altogether the use of depth information for inpainting has been partially explored but there is no common method to benefit from this additional data so far. Nevertheless, the results so far achieved are promising in some scenarios and should be explored.

2.6. Multiscale and Hierarchical Inpainting Methods

For the task of inpainting of big holes, several methods using a multiscale or hierarchical approach have been proposed.

As an extension to the K-SVD algorithm presented in Section 2.3.3, Mairal et al. [15] presented a multiscale version. They establish a quadtree structure for a large image patch as shown in Fig. 2.17. For a fixed number of scales N the large image patch with n_s pixels is divided by two in height and width for each scale $s = 0...N-1$. A subpatch of scale s has $n_s = \frac{n}{4^s}$ pixels. For every scale s a dictionary $\mathbf{D}_s \in \mathbb{R}^{n_s \times K_s}$ with K_s atoms is learned. As Mairal et al. wanted to stick very close to the original K-SVD algorithm, all the atoms of the dictionaries $\mathbf{D}_s$ are embedded into one overall dictionary $\mathbf{D} \in \mathbb{R}^{n \times K}$. Smaller atoms first, then the size of the large atom of scale $s = 0$ are embedded with zero padding on every possible position. An image patch can be decomposed as shown in Fig. 2.16. The reconstruction step remains the same as in Section 2.3.3 and applies implicitly the multiscale idea. In the learning process, a modification has to be made to the dictionary update step as each atom in every scale has to be updated (see [15]).

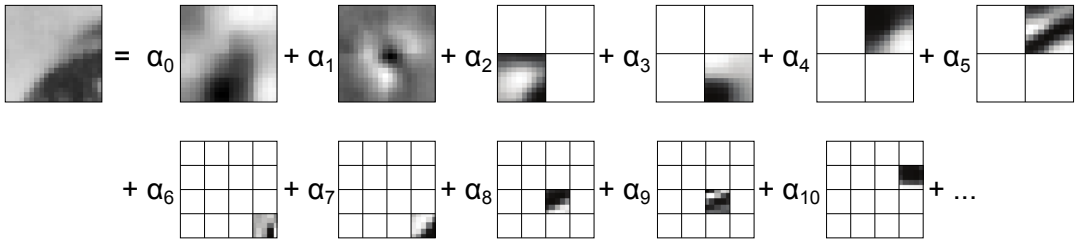


Figure 2.16.: Decomposition of a 20×20 image patch from Mairal et al. [15].

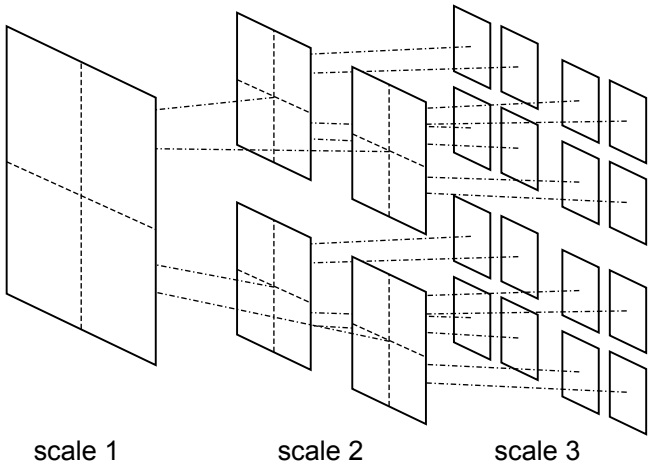


Figure 2.17.: Quadtree structure for multiscale K-SVD algorithm from Mairal et al. [15] with 3 scales.

Multiscale approaches have also been applied in exemplar-based methods. Cant et al. [36] reduced the size of the image to inpaint by factor 2,4,... and then searched for the best matching complete patch for an incomplete patch in the lowest resolution. As a next step, they took the region of the best matching patch as a search region for the same procedure in the next higher resolution (see Fig. 2.18). Besides speeding up the inpainting process,

this methods also gives the big advantage of having more general results. This way the error spread due to strong intensity changes is limited.

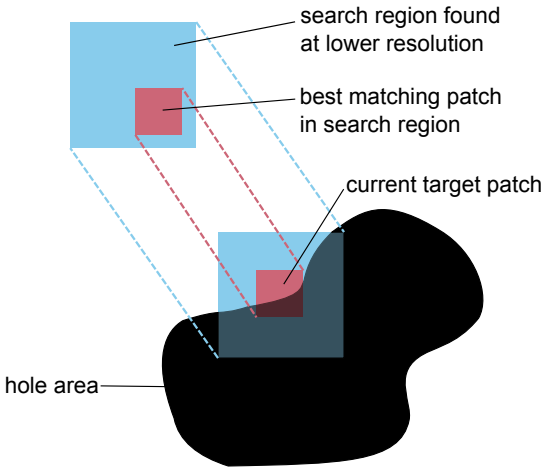


Figure 2.18.: Multiscale patch matching from Cant et al. [36]. The blue square indicates the matching patch that has been found at a lower resolution and this area is then used as a search region for the inpainting at the highest resolution.

Le Meur et al. [37] use a single low-resolution picture as a first step. The inpainting algorithm gives a combination of patches by using a non-local means approach to create a new patch for the incomplete region. When going to the original resolution, a dictionary with the correspondence between low and high resolution image patches has to be built. In the higher resolution the same linear combination as before is used but this time with the higher resolution image patches.
The image pyramid is a very popular multi-scale representation of images. It uses a Gaussian low-pass windows filter to create different scales of an image. Wang et al. [38] performed their inpainting algorithm on all scales and later fused the information using a weighted average approach.
Solh et al. [39] ,[40] even base their hole inpainting process only on image reduction using Gaussian low-pass filters and linear interpolation. First they reduce the original image subsequently by using Gaussian low-pass filtering until no visible holes can be found anymore in the image. Then they expand the image to the next higher scale using linear interpolation. The so created texture information is used for the inpainting step. Afterwards they expand the inpainted image again and inpaint the corresponding incomplete scaled image. The procedure is repeated until the original image size is reached.

2.7. Limitations of Existing Methods

Most of the so far presented methods have been developed for specific tasks which are different from the setting presented in Section 1.3: inpainting large holes from 3-D view

synthesis given the incomplete image, the holemask and the complete depth map. Therefore they all lack certain features which are presented in this section.
The powerful K-SVD algorithm from Aharon et al. [14] (Section 2.3.3) and the corresponding reconstruction step have been originally developed for image denoising with zero-mean white and homogeneous Gaussian noise. Therefore the patch order during the reconstruction did not play any role. Aharon et al. and later Mairal et al. [2] simply took pixel-wise overlapping patches going from the left to the right and from the top to the bottom of the image. Regarding hole filling this procedure does not conserve texture and structure very well - instead it blurs the image as for the final result a mean over the results of all overlapping patches is taken (see Fig. 2.19). Furthermore, this leads to two different problems when inpainting large holes: Either patches can not be inpainted when an image patch is not updated with the inpainting result before going to the next patch. Or when the image is always updated, there is a big risk of error spreading and this strongly depends on the chosen direction during the patch selection (see Fig. 2.20). This work tries to solve these disadvantages of the existing K-SVD algorithm by assigning patch priorities as in [1].

Figure 2.19.: Blurring effect during inpainting with original K-SVD. From left to the right: image area with hole, original image area, with K-SVD inpainted area.

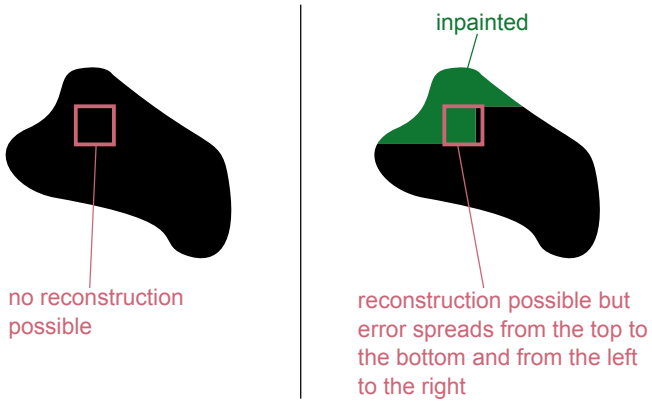


Figure 2.20.: Problems occurring during inpainting process with original K-SVD. Either reconstruction is not possible or error spreads in one direction.

Criminisi et al. [1] (Section 2.4) have presented an algorithm that became a standard reference for inpainting since then. But it was designed for two-dimensional inpainting purposes without a depth map available. In the case of view synthesis inpainting, where the

holes usually appear next to foreground objects but are associated to the background (because of the viewpoint change), this leads to false texture and structure propagation in the hole.

Figure 2.21.: Problem occurring during inpainting process with Criminisi et al. [1]. Hole (in white in the left image) associated to the background is partly inpainted on the right image with texture information from the person in the foreground.

Also the methods that use depth information from Section 2.5, have their limitations. The method of Oh et al. [8] might help the filling for a small distance, but for a wider camera baseline, it might distort the geometry (see Fig. 2.22).

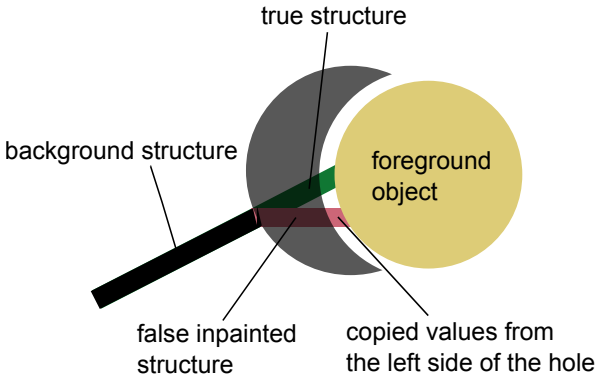


Figure 2.22.: Problem occurring during inpainting process with Oh et al. [8]. The copied values from the left do not represent the correct structure due to the viewpoint change.

The approach developed by Hervieux et al. [13] is powerful but computationally expensive and only works well for specific images. Especially for images with clear geometrical forms, for example many large planes and few rounded structures, the algorithm gives convincing results. As soon as the geometry becomes complicated or the quality of the depth map is poor, other methods should be preferred. For this work, the depth information has been used for depth layer segmentation and then, inpainting is performed within the layers (see Section 3.2.1). The computational complexity for this approach is limited and it can be

applied to a wide range of images.
A multiscale approach for K-SVD solves one of the main problems that large holes can not be inpainted. But the price for this improvement is high as the computation complexity increases radically: Mairal et al. [15] used as the smallest scale a 5×5 patch. When working with 3 scales, this results in a 20×20 patch for the largest scale, so every atom in the dictionary is of length 1200 (compared to 192 in the original single-scale K-SVD algorithm with 8×8 patches). At the same time the number of atoms in a dictionary increases. For every atom in the largest scale, 16 atoms in the next smaller scale and 256 atoms in the smallest scale have to be added to the dictionary due to zero padding (4 smaller patches per scale multiplied by 4 different possible positions). The lack of performance can be observed when running the already strongly optimized implementation of [15]. Also the other problem that pixel-wise overlapping patches lead to a blurring effect remains the same.
Applying multiscale representations of images for inpainting algorithms in general seems to be a powerful method to conserve the generality of the results. However it increases computation time strongly during the learning process and should be implemented in a lean way. To overcome this problem this work proposes to keep the size for dictionary atoms small during the learning as described in Section 3.2.1 and rescale them into a higher resolution for the filling (see Section 3.2.1). Furthermore, the multiple scales of patches for the filling are found in a smart way (see Section 3.2.2).

3. Implementation

The aim of the software developed in this work is to inpaint an image with holes given the holemask, the depth map and the input parameters. It has been entirely implemented in Matlab. As a starting point two existing toolboxes have been used:

1. KSVD-Box v13 and OMP-Box v10 from Ron Rubinstein, Michael Zibulevsky and Michael Elad [41]:
   The toolbox implements the work of Aharon et al. [14] and Elad et al. [32] and was originally developed for denoising of black-and-white images.

2. Matlab implementation of the work of Criminisi et al. [1] from Sooraj Bhat [42]:
   The implementation performs exemplar-based inpainting using patch priorities (see Fig. 2.15).

The structure of the entire inpainting software is introduced in Section 3.1. The input images and parameters are defined in Section 3.3 and Section 3.4. The functionalities of the different parts of the implementation are presented in Section 3.2.

3.1. Software Structure

The software structure of the proposed inpainting algorithm using K-SVD and patch priorities is presented in Fig. 3.1.

3.2. Implementation Details

In this section, details on the implementation are presented as well as the solutions for the challenges that arised during the development process. A documentation does not exist for every part of the software but only for the main functionalities as well as for the new developments of secondary functionalities.

3.2.1. Main Software Functionalities

msksvdinpaintingdemo.m

This is the main script that starts the execution of the learning and inpainting process given the required input. The location of the input images is provided through command line, the parameters have to be changed in the script. The execution time of the complete learning and inpainting process is measured. As the software is not necessarily run on machine with

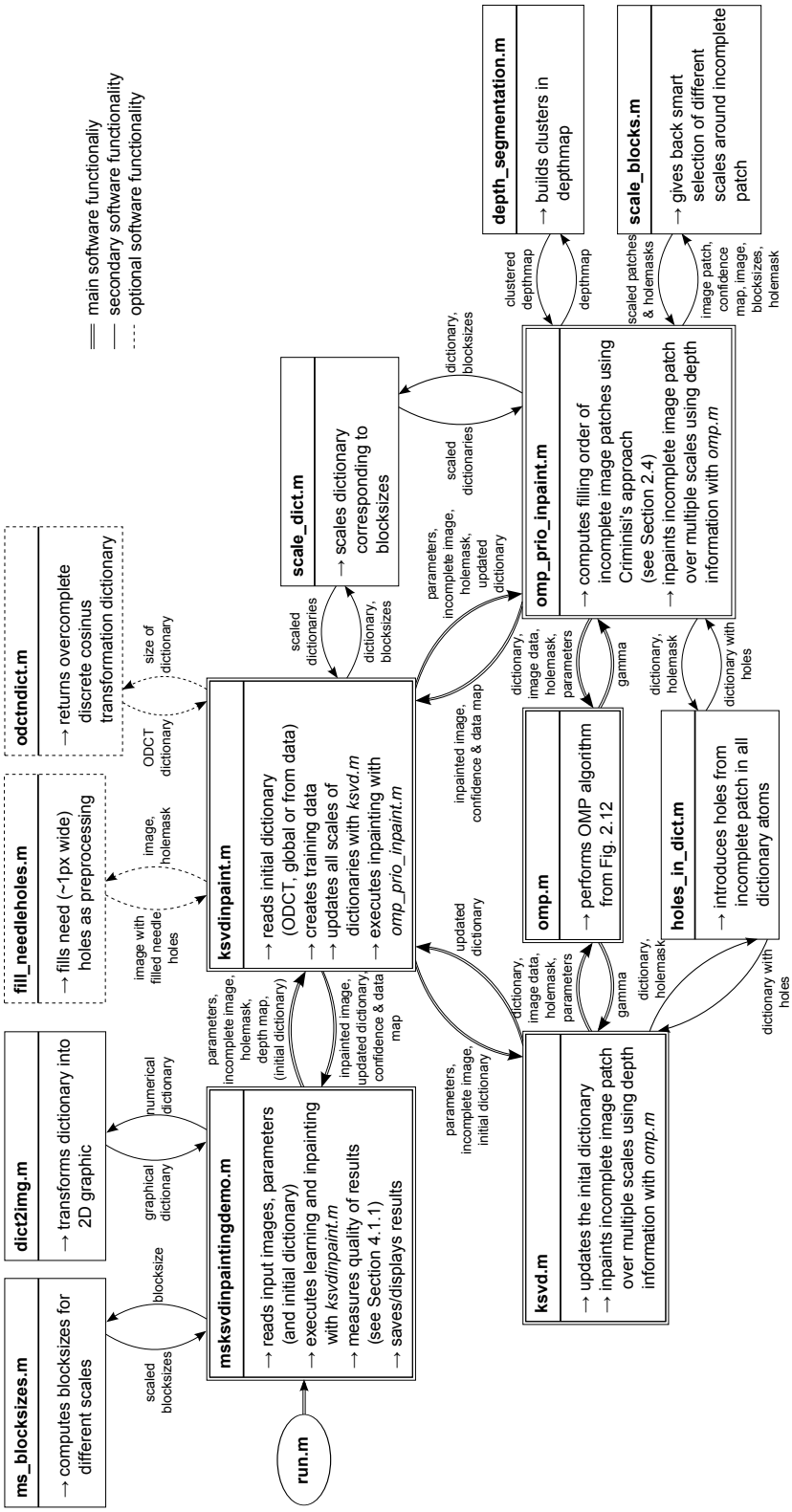


Figure 3.1.: Software structure of the proposed inpainting algorithm.

graphical Matlab interface, a flag can be set if the results should be displayed, only saved or both. The saved output consists of the

- Input images,
- Inpainted image,
- Confidence and data map,
- Dictionaries used for the inpainting,
- Structural SIMilarity (SSIM) and High Dynamic Range Visual Difference Predictor (HDR-VDP) map from the evaluation,
- Params file with the input parameters and the output values from the evaluation.

ksvdinpaint.m

Depending on the input parameters, the initial dictionary (or dictionaries when inpainting over multiple scales) is prepared:

- Overcomplete Discrete Consinus Transformation (ODCT): An overcomplete 3-D discrete consinus transformation dictionary with the size of the input parameters is created.
- data: Random samples from the incomplete input image are taken as the initial dictionary.
- global: The global dictionary that has been entered in msksvdinpaintingdemo.m is taken as the initial dictionary. This dictionary has been learned before with an additional script using K-SVD over a number of images.

In order to update the dictionaries with the information of the incomplete image, a set of training data is created. As defined in the input parameters, a number of equidistant image patches are extracted from the incomplete input image. Only patches with less than 50% of missing information are taken. Furthermore, the training data is centered in relation to the maximal color value. For the multiscale approach, the computation time should be limited. Therefore the originial size for a dictionary atom is maintained and the training data is resized to the corresponding smaller scales. In order to handle the RGB data from an image patch, it is reshaped as in Section 2.3.1.
The updating process itself takes place in *ksvd.m*. When analyzing the implementation from Mairal et al. [43] which both combines the work of [15] and [2], it was discovered that they introduce additional color atoms for every channel after updating the dictionaries. Mairal et al. also mention this in [2]: "For the color extension, we introduced one constant atom per channel, i.e., one red, one green, and one blue atom...". This modification was implemented.
The updated dictionaries is then used in the inpainting step in *omp_prio_inpaint.m*.

ksvd.m

The implementation of the K-SVD algorithm is based on the work from Mairal et al. [2], respectively Rubinstein et al. [41]. The algorithm was presented in Section 2.11. As described in Section 2.3.3, two modifications had to be implemented in order to apply the algorithm to (color) images with non-uniform noise, namely:

- Implementation of the weighted rank-one approximation algorithm (see Fig. 2.13) to solve Eq. 2.19
- Modification of the OMP algorithm as shown in Fig. 2.12

The second point was realized differently: As the training data is centered, it is possible to introduce the holes in the dictionary before running the OMP step. This reduces computation time. The β mask (=holemask) is only handed over to the OMP algorithm to update the threshold for the loop. Before handing over the dictionary to the OMP algorithm, it is normed and the norm stored for the later reconstruction in the K-SVD algorithm.

omp_prio_inpaint.m

After updating the dictionary, the inpainting step itself is executed. For the same reason as in Section 3.2.1, the data is centered before performing the inpainting. The filling order is defined through the priority assigned by the algorithm of Criminisi et al. 2.15.
The hole filling is performed depth-layer by depth-layer and loops from the back- to the foreground. This approach gave the most convincing result to take advantage of the depth information after implementing and evaluating the work of other authors (Gautier et al. [6] and Daribo et al. [5], [34]). An example how the layer-based method affects the input for the filling with OMP is given in Fig. 3.2.
As many difficulties came up when implementing the OMP inpainting, a separate script was created that only performs this step: *slim_OMP_Inpainting*. The same results as in the text removal test image from Mairal et al. [2] were obtained with pixel-wise overlapping.

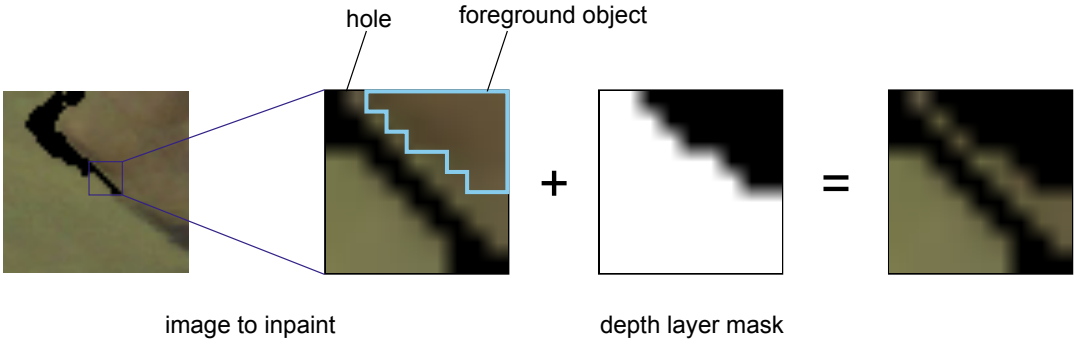


Figure 3.2.: For the inpainting of a 9x9 patch, the holemask is combined with the depth mask resulting from depth layer segmentation.

Multiscale extensions as presented in Section 2.6 can add further improvements to the inpainting process but are always subject to higher computational complexity. Therefore a lean method was implemented which gives the same advantages. For the patch that has to be inpainted, several other larger patches (in terms of pixel) are computed with *ms_blocksizes.m*. The filling process starts with the largest scale. In order to obtain a dictionary that corresponds to this scale, it is resized using bicubic interpolation. The larger patches, in which the patch that is about to be inpainted lies, are found with *scale_blocks.m* as well as the corresponding holemask (an additional image that indicates where holes are located). Given the resized dictionary, image patch, holemask and the parameters, the OMP algorithm can perform the reconstruction. For the next smaller patch size, the remaining error residual is computed and the process is repeated until it reaches the smallest patch size. The inpainting process is briefly described in 3.3.
Even if a complete reconstruction for the whole image patch is obtained, only the information that is needed to fill the hole is used for inpainting.

For every depth layer from the back- to the foreground:

- Residual = patch with maximum priority
- Cleanpatch = zeros
- For every scale from the largest to the smallest:
    1. Resize dictionary to actual scale
    2. Find a patch of the actual scale in which the patch to inpaint lies
    3. Find a reconstruction for the non-hole areas of the residual with OMP
    4. Cleanpatch = (cleanpatch + reconstruction) in size of next smaller scaled patch
    5. Residual = next smaller scaled patch - cleanpatch

Figure 3.3.: The description of the multiscale inpainting process developped in this work.

omp.m

This is a direct implementation from Fig. 2.10. The only modification is that the threshold is updated with the number of filled pixel as ϵ is a per-pixel-error.

3.2.2. Secondary Software Functionalities

depth_segmentation.m

The depth segmentation is realized by the Matlab implementation[1] of k-Means clustering [44]. For every possible cluster configuration from one cluster to the given maximum, the Davies Bouldin index (see [45]) is computed. Then the cluster with the minimal index is chosen and the depth layers are sorted from the back- to the the foreground.

[1] http://www.mathworks.de/de/help/stats/kmeans.html

scale_blocks.m

To obtain larger scales of the image patch that has to be inpainted, the area around the patch can be considered. The easiest solution would take the patches in which center the patch to inpaint is located. As more information is available, this process can be improved. The confidence map computed for the filling order gives a good measure of the quality of information in a potential patch, e.g. mostly inpainted patches will have a lower confidence value. Additionally the actual holemask and the depth layers have to be considered. The confidence value for all potential patches is computed and the one fixed with the highest confidence value. The same process is executed for the next higher scale, this time with the just found patch as the basis and so on.

ms_blocksizes.m

For inpainting over multiple scales, the size of the patches has to be fixed. This function computes the size of the larger patches with the following logic: the next larger scale is twice as large as the previous one plus one pixel.

3.2.3. Optional Software Functionalities

fill_needleholes.m

Cheng et al. [35] mentioned in their work, that in the context of view synthesis they used simple interpolation for holes that are thin ($\approx$ 1 pixel wide) and typically vertical ("needle holes"). This concept applies well as a preprocessing step before updating the dictionary. Depending on the view synthesis algorithm such holes can occur very often. Although they do not represent a significant lack of information, they can degrade the quality of the updated dictionary in the K-SVD update step. The implemented needle hole filling uses an interpolation technique developed by Do et al. [46].

3.3. Input Images

Four images with specific characteristics are requested through commandline when running the inpainting software:

- Reference image: The inpainted image is compared to a reference image through several quality measures. When evaluating the software with existing test sequences from Microsoft Research (MSR) or MPEG, a view should be synthesized for which a real image already exists.
- Image for inpainting: This is the image that should be inpainted. There is no specific requirement concerning holes as the image is later processed with the input holemask.

- Holemask for inpainting: The holemask should only have a color depth of 1 bpp. Holes are 0 (=black) whereas the complete areas are 1 (=white).
- Depth map: The depth map should have a color depth of 8 bpp - typically only gray values are used. The depth map should not have any holes - when working with the previous mentioned MSR or MPEG test sequences, there should be a real depth map for the synthesized view.

All images should be of the same size and in the PNG file format.

3.4. Parameters

A number of parameters are defined in *msksvdinpaintingdemo.m* which have an impact on the results:

- scales: The number of different sizes of patches that will be considered in the filling step in *omp_prio_inpaint.m*. If multiscale inpaiting is not used, it can be set to 1.
- dictsize: The number of dictionary atoms in a dictionary for every scale.
- maxval: The maximum intensity value of the reference and the synthesized image (typically 255 in images with 8 bpp color depth).
- trainnum: The number of training samples used for updating the initial dictionary with K-SVD.
- iternum: The number of iteration steps in the K-SVD algorithm.
- fill_needle_holes: The flag if needle hole filling is used before updating the dictionary with K-SVD.
- remove: The flag if incomplete patches are removed before updating the dictionary with K-SVD (can result in very few training data).
- max_depth_layers: The number of maximal depth layers for depth segmentation. If no depth information at all should be used, it can be set to 1.
- initdict: The type of the initial dictionary as described in Section 3.2.1: 'odct', 'global' or 'data'.
- ksvd.epsilon: The ϵ value for updating the dictionary with K-SVD.
- ksvd.sparsity: The maximal sparsity for updating the dictionary with K-SVD.
- inpaint.epsilons: The ϵ values for every scale when updating the dictionaries with K-SVD. (format: e.g. [0.74 7.4 74])
- inpaint.sparsities: The maximal sparsity for every scale when updating the dictionaries with K-SVD. (format: e.g. [5 10 15])
- muthresh: The threshold for replacing atoms in the K-SVD algorithm.

3.5. Exemplar-based Multiscale Inpainting with Patch Priorities and Depth Information

In addition to the above presented algorithm that learns a dictionary with K-SVD, another algorithm has been developed.
Exemplar-based inpainting methods such as from Criminisi et al. [1], Daribo et al. [5] and Gautier et al. [6] give good results for most inpainting cases and have been the state-of-the-art in many frameworks. They need few computation time for the SSD patch matching process and the inpainted areas often seem natural as real information from the image is copied. But they lack a good way to incorporate depth information. Furthermore an approach that uses multiple scales has not been presented so far. That is why this work presents additionally to the main focus of the K-SVD-based dictionary learning method another exemplar-based method. Several improvements that have been developed for the K-SVD-based method have been adapted to the new exemplar-based method.
The following three improvements have been added to the original algorithm of Criminisi et al. [1]:

- Needle hole filling:
  A common problem of exemplar-based methods in the context of inpainting of synthesized images is, that the search range of complete patches for SSD is limited due to the thin mostly vertical needle holes. Therefore the developed algorithm inpaints these holes as a preproccesing step using an interpolation technique developed by Do et al. [46].

- Depth-layer-based inpainting:
  For the same reason as in the previously presented method, the inpainting is executed layer-wise (the combination of holemask and depth information was presented in Fig. 3.2). Therefore the same depth segmentation step as in Section 3.2.2 has to be performed before the inpainting starts.

- SSD patch matching over multiple scales:
  The multiscale approach for the previously presented algorithm improves the inpainting result significantly. Therefore it was adapted to the exemplar-based method. It follows the same logic for finding the larger scales of patches (see Section 3.2.2). For every scale the best matching patch in the same depth layer is computed with SSD and the area of the smallest scale is stored. After having processed all scales, a combination of the patches is computed with OMP (see 2.10). This method applies well to calm images where foreground objects are placed in front of an almost uniform background. Images with a strongly changing background often do not provide appropriate patches in high scales which results in error spreading. Furthermore the number of scales has to be chosen in respect to the image and the hole size. In small images, the size of the largest patch and the number of scales is limited in order to provide enough patches for SSD. Regarding the hole size, a trade-off has to be found between choosing the number of scales big enough to cover the largest

hole and keeping the largest patch as small as possible to provide enough patches for SSD.

4. Evaluation

4.1. Evaluation Measures

To quantify the quality of the results of the developed algorithms, several objective evaluation measures have been used as they represent the results of human visual perception on different levels. Additionally a subjective image quality test has been conducted to confirm the results.

4.1.1. Objective Evaluation Measures

Peak Signal-to-Noise Ratio (PSNR)

The Peak Signal-to-Noise Ratio (PSNR) is a common method to measure the quality of a distorted image compared to a reference image. It is defined over the Mean Squared Error (MSE),

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - I(i,j)]^2 \tag{4.1}$$

such that

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \tag{4.2}$$

where MAX_I is the maximum possible pixel value of the image and it is measured in dB.

Structural Similarity (SSIM)

The SSIM index measures the similarity in terms of structural information between two images [47]. Whereas traditional objective evaluation measures quantify the visibility of errors between a reference image and a distorted image, the SSIM index focus on the differences in structural information as this more adapted to the human visual perception. The idea of structural information is that the nearer pixels are, the stronger the inter-dependencies they have are. Other than for PSNR, contrast and luminance differences should not have an impact on the SSIM index values.
The SSIM index between two windows $\mathbf{x}$ and $\mathbf{y}$ of size $N \times N$ computes as follows

$$SSIM(\mathbf{x}, \mathbf{y}) = \frac{\left(2\,\mu_{\mathbf{x}}\mu_{\mathbf{y}} + C_1\right)\left(2\sigma_{\mathbf{xy}} + C_2\right)}{\left(\mu_{\mathbf{x}}^2 + \mu_{\mathbf{y}}^2 + C_1\right)\left(\sigma_{\mathbf{x}}^2 + \sigma_{\mathbf{y}}^2 + C_2\right)} \tag{4.3}$$

where $\mu_{\mathbf{x}}$ is the average of $\mathbf{x}$, $\mu_{\mathbf{y}}$ is the average of $\mathbf{y}$, $\sigma_{\mathbf{x}}^2$ is the variance of $\mathbf{x}$, $\sigma_{\mathbf{y}}^2$ is the variance of $\mathbf{y}$, $\sigma_{\mathbf{xy}}$ is the covariance of $\mathbf{x}$ and $\mathbf{y}$, C_1 is a constant to avoid instability when $\mu_{\mathbf{x}}^2 + \mu_{\mathbf{y}}^2$ is very close to zero and C_2 the same for $\sigma_{\mathbf{x}}^2 + \sigma_{\mathbf{y}}^2$.
This gives a map with SSIM index values for every pixel in the image. In order to get an overall value for the entire image, the Mean Structural SIMilarity (MSSIM) of all SSIM values is taken:

$$MSSIM(\mathbf{X}, \mathbf{Y}) = \frac{1}{M} \sum_{j=1}^{M} SSIM\left(\mathbf{x}_j, \mathbf{y}_j\right) \tag{4.4}$$

where $\mathbf{X}$ and $\mathbf{Y}$ are the reference and the distorted image and M the number of $N \times N$ windows in the image. The MSSIM index goes from 0 to 1 with 1 being the highest value.
SSIM has been used for evaluation of this work to obtain objective repeatable measures that come closer to subjective results than PSNR.

High Dynamic Range Visual Difference Predictor (HDR-VDP)

In recent publications, it was shown "both empirically and analytically, that the [SSIM] index is directly related to the conventional, and often unreliable, mean squared error", respectively the PSNR [48]. Therefore to complete the objective quality measures, another evaluation algorithm was used: High Dynamic Range Visual Difference Predictor. Developed by Mantiuk et al. [49], [50], this metric can predict if differences between two images are visible to a human. The main advantage of HDR-VDP is that the metric works with a greater dynamic range in terms of luminance like it is the case in real world images (HDR images). It is based on the work from Daly et al. [51]. As for comparison, the Quality Mean-Opinion-Score (Q-MOS) computed by HDR-VDP (from 0 to 100, where 100 is the highest quality) is taken.

4.1.2. Subjective Evaluation Measures

The structure of the test sequences was based on the recommendations from [52] on Double-Stimulus Impairment Scale (DSIS) tests. A test block for one image can be described as follows (see Fig. 4.1):
After 2 s of a mid-gray screen (RGB-value: [127, 127, 127]) with the the test number in white and 2 s of break, the reference picture (which is the original non-synthesized image from the test sequence) is exposed for 5 s. After another 2 s of mid-gray break, the test image is exposed for 5 s. This procedure is repeated once and a block finishes with 5 s of voting time for the subject. For images smaller then Full HD resolution, a mid-gray border is added. The test of all images was realized in a way, that a block with similar image material does not follow another one as well as same algorithms. For every test image, the one with the algorithm that performed worst in the objective test is shown first.
At maximum 3 subjects performed the test at the same time in a distance of 0.7 m to the screen. The subjects voted on a scale from 0 to 10, 10 being the best vote. The evaluation sheet can be found in the Appendix A. Altogether, 20 subjects performed the test

and no vote had to be dropped during the screening process. The votes follow a Student's t-distribution with ν = 19.

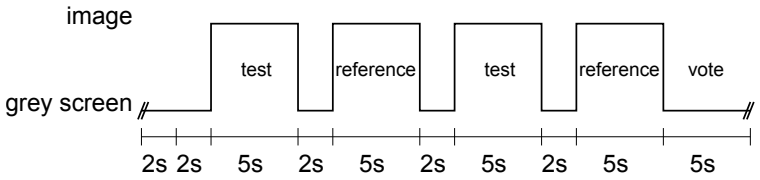


Figure 4.1.: The structure of a test block for the subjective evaluation of one image.

All tests were conducted in the video laboratory of the Institute for Data Processing at the Technische Universität München. This laboratory allows testing under all specifications given by [52]. The display used is a reference monitor with 23" and Full HD resolution (SONY BVM-L230). This display represents a standard test device.

4.2. Input

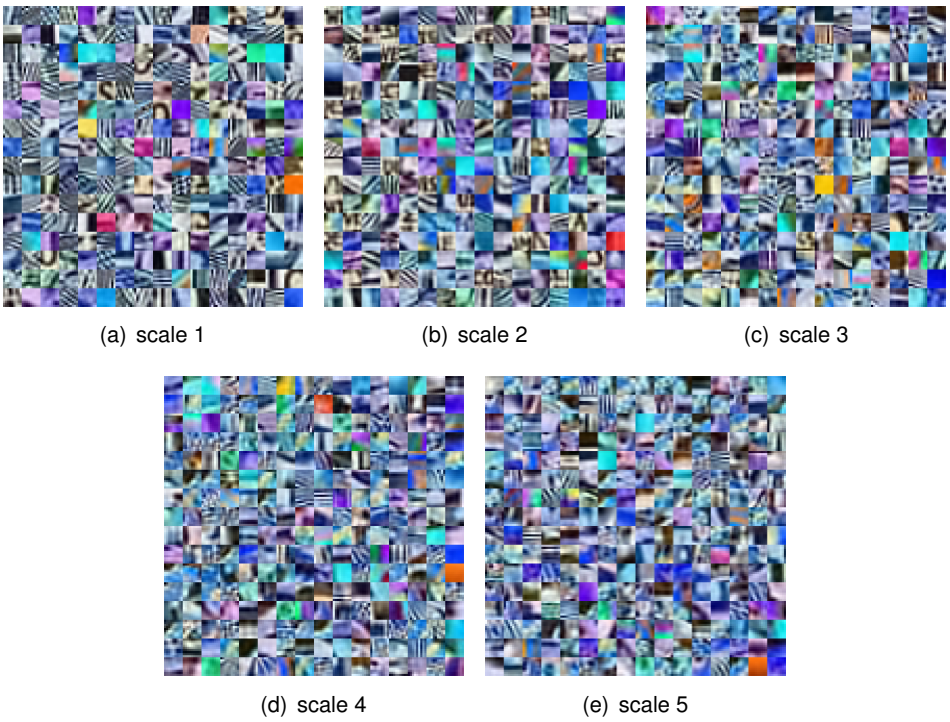
Besides the parameters that have been experimentally determined, the inpainting algorithm gets an initial global dictionary as input. Furthermore the test sequences that have been used for evaluating the algorithm, are described in this chapter.

4.2.1. Learning an Initial Global Dictionary

As described in Section 3.2.1, the inpainting algorithm can take as initial dictionary a global, an ODCT or a dictionary consisting of random samples of the input image. Whereas an initial dictionary of random samples is for experimental use, the ODCT dictionary has already been used as a baseline for denoising of black-and-white images [14]. However, Mairal et al. [2] showed with their experiments the inefficiency of ODCT dictionaries for color inpainting. That is why a global multiscale dictionary has been learned for this work (see Fig. 4.2). A separate script for this task has been created which learns a global multiscale dictionary from a number of BMP images with the following parameters:

- Number of dictionary atoms: 256
- Number of training samples: 100000
- Number of iteration steps in the K-SVD algorithm: 20
- Number of scales: 5
- ϵ for all scales: 7.444216
- Maximal sparsity L for all scales: 20

The choice of parameters has been inspired by Aharon et al. [14] and Mairal et al. [2], [15].
The used 5-scale dictionary was learned from the LIVE Image Quality Assessment Database Release 2[1].



(a) scale 1 (b) scale 2 (c) scale 3

(d) scale 4 (e) scale 5

Figure 4.2.: The global 5-scale dictionary learned with K-SVD from the LIVE Image Quality Assessment Database Release 2.

4.2.2. View Synthesis of Test Sequences

The most popular test sequences are from MSR and MPEG. The corresponding view synthesis algorithms work differently. The MPEG view synthesis algorithm is implemented in the View Synthesis Reference Software (VSRS) which is mainly based on [53] and [54]. The MSR algorithm is described in [55].
In a first step, view synthesis Matlab scripts were developed, one based on the MSR C++ implementation from [56], the other one based on the MPEG document [57]. These scripts can already be used to obtain images for the evaluation of the new inpainting algorithm.
A view synthesis algorithm with higher performance that avoids need holes in the output image is implemented in the VSRS. That is why this software has been used in this report. VSRS performs view synthesis from two views and applies the inpainting function

[1] http://live.ece.utexas.edu/research/quality/subjective.htm

from the OpenCV library to the left over holes. For the purpose of this work, the incomplete view synthesized out of only one view was needed without having the holes inpainted. The source code (*ViewSynthesis.cpp* in ViewSynLibStatic) had to be modified in order to obtain the output images from the intermediate steps. As the synthesized views from the first and the second image are computed separately and only later the information is fused, there is only one additional line needed to output the first synthesized view:

```
cvSaveImage("virtualimage.png", m_pcViewSynthesisLeft->getVirtualImage());
```

The corresponding holemask in the VSRS is code is modified with morphological transformations (erosion $\rightarrow$ dilation $\rightarrow$ dilation) such that boundary errors due to an inaccurate depth map during the inpainting process are avoided. The following line is added to obtain the processed holemask:

```
cvSaveImage("unstable.png", m_pcViewSynthesisLeft->getUnstablePixels());
```

With these modifications, the MPEG test sequences can be used for the evaluation of the new inpainting algorithm. In order to use MSR test sequences, the input parameters have to be modified: First, the vertical camera coordinates (usually v) are mirrored. Hence in the intrinsic matrix

$$\mathbf{A} = \begin{bmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.5}$$

the focal pixel length α_y has to be inverted as well as the principal point v_0 has to be modified such that $v_{0,new} = v_{max} - v_0 - 1$ where v_{max} is the vertical image resolution.
Another change is that the origin of the 3-D space in the VSRS software is different compared to the MSR coordinate system. The depth type in the input configuration file for VSRS has to be changed and the following line needs to be added(see [58]):

```
DepthType       1
```

Additionally the rotation matrix **R** has to be inverted and the translation vector **t** has to be multiplied with the inverted rotation matrix $\mathbf{R}_{new} = inv(\mathbf{R})$.
Furthermore the signs in the translation vector have to be changed as described in [54].

4.3. Experiments and Analysis

The impact of the input parameters, in particular of the new introduced multiscale and depth-layer-based inpainting solutions, on the developed algorithm MultiScale K-SVD with Depth (MS-KSVD-D) have been evaluated. In this section, the results are presented in form of the previously introduced quality measures as well as some concrete examples of the impact are shown. Not all parameters have been considered in this part. For the following parameters, existing values have been used that already have proven well [2], [15]:

- dictsize: 256

- iternum: 20
- ksvd.epsilon: 7.44
- ksvd.epsilon: 20
- muthresh: 0.99

4.3.1. Multiscale Approach

Using multiple scales as described in Fig. 3.3 can add substantial improvements for the consistency of the result over the whole hole but conserves fine texture and structure as well (see Fig. 4.3). The improvements can also be quantified in terms of evaluation measures (see Table 4.1).

Evaluation Measure	3 scales	1 scale
PSNR [dB]	33.46	28.04
MSSIM	0.905	0.864
Q-MOS	32.09	9.91

Table 4.1.: Objective evaluation measures for MS-KSVD-D inpainting with different number of scales after view synthesis from the MPEG ballet test sequence (view 2 $\rightarrow$ 3, frame 1).

(a) reference (b) image with hole (c) 3 scales (d) 1 scale

Figure 4.3.: Example of the impact of multiscale inpainting. Inpainting with 3 scales brings an advantage for the overall consistency of the result.

4.3.2. Depth-layer-based Inpainting

The other major improvement that was implemented in addition to the main idea of combining the K-SVD algorithm and the patch priorities, was to inpaint depth-layer-wise. The impact of inpainting only within a automatically determined number of depth layers was evaluated (see Fig. 4.4). Fixing the number of maximal depth layers to 1 results in non-depth-layer-based inpainting. Using multiple layers gives slightly better quality measures (see Table 4.2).

As the number of training samples strongly influences the computation time, the impact of a relatively small number of training samples was measured. The impact on the learned

Evaluation Measure	max. 6 depth layers	max. 1 depth layer
PSNR [dB]	33.46	32.03
MSSIM	0.905	0.890
Q-MOS	32.09	28.39

Table 4.2.: Objective evaluation measures for MS-KSVD-D inpainting with different maximal number of depth layers after view synthesis from the MPEG ballet test sequence (view 2 $\rightarrow$ 3, frame 1).

(a) reference (b) image with hole (c) max. 1 depth layer (d) max. 6 depth layers

Figure 4.4.: Example of the impact of depth-layer-based inpainting. Inpainting with max. 6 depth layers brings an advantage regarding the correctness of the result at foreground object borders compared to single-layer-based inpainting.

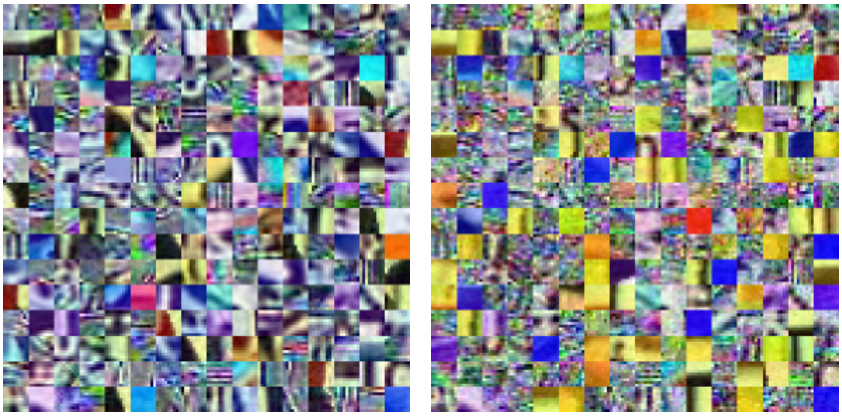
dictionary is perceivable (see Fig. 4.5): The dictionary learned with a relatively small number of samples is noisy. But also the objective image quality is worse (see Table 4.3).

Evaluation Measure	10000 training samples	1000 training samples
PSNR [dB]	33.46	30.62
MSSIM	0.905	0.903
Q-MOS	32.09	16.16

Table 4.3.: Objective evaluation measures for MS-KSVD-D inpainting with different maximal number of training samples after view synthesis from the MPEG ballet test sequence (view 2 $\rightarrow$ 3, frame 1).

4.3.3. Epsilon and Sparsity

The sparsity and ϵ are the most important parameters for OMP during the filling step. Whereas for single-scale inpainting, there are already well established values, new parameters had to be determined for the new multiscale approach. Two main conclusions have been found: Firstly, the sparsity for every level should not be chosen as high as for single scale inpainting but rather be equally distributed over all scales. Otherwise unpleasing error spreading can be observed (see Fig. 4.7). Secondly, the choice of ϵ should be limited in range because a too big ϵ can introduce color artifacts and performs bad in terms of structure conservation (see Fig. 4.6). Whereas the first statement can also be observed in terms of objective image quality, this is not the case for the second (see Table 4.4). However, a rather small ϵ that increases with the image scale is suggested to obtain better subjective results.



(a) learned dictionary with 10000 training samples

(b) learned dictionary with 1000 training samples

Figure 4.5.: Learned dictionary for 10000 and 1000 samples. To few training samples result in a noisy dictionary.

Evaluation Measure	sparsity: 7, 7, 7 ϵ: 0.074, 0.74, 7.4	sparsity: 7, 7, 7 ϵ: 740, 740, 740	sparsity: 20, 20, 20 ϵ: 0.074, 0.74, 7.4
PSNR [dB]	33.46	33.95	29.32
MSSIM	0.905	0.910	0.884
Q-MOS	32.09	38.60	15.76

Table 4.4.: Objective evaluation measures for MS-KSVD-D inpainting with different values for ϵ and sparsity after view synthesis from the MPEG ballet test sequence (view 2 $\rightarrow$ 3, frame 1).

(a) reference

(b) image with hole

(c) inpainting with appropriate ϵ

(d) inpainting with too big ϵ

Figure 4.6.: Example of the impact of the choice of ϵ for MS-KSVD-D. A too big ϵ results in color artifacts and bad structure conservation.

4.4. Comparison with Prior Work

For a final evaluation, the new developed methods have been compared to state-of-the-art algorithms in the domain of inpainting.

(a) reference (b) image with hole (c) inpainting with appropriate sparsity (d) inpainting with too big sparsity

Figure 4.7.: Example of the impact of the choice of the sparsity for MS-KSVD-D. A too big sparsity results in strong error spreading.

4.4.1. Tested Algorithms

The following algorithms have been chosen:

- Criminisi et al. [1]: With over 800 citations, this algorithm presents one of the most popular inpainting algorithms. No specific parameters are used for the evaluation. An implementation is available online [42].
- Gautier et al. [6]: Of all exemplar-based methods that also use depth information, this algorithm brings the most significant improvements. The only specific parameter that was given to the software is the direction of inpainting. The software is not publicly available - it was obtained from the author personally.
- Inpainting algorithm from VSRS [58]: As part of the reference software, the algorithm provides a robust inpainting solution. It uses the inpainting function from the OpenCV library[2], which is based on Navier Stokes PDE like in [11]. As the reference software was developed for view interpolation, for both input views the same image and depth map was given. Furthermore, view blending was deactivated in the configuration file.

The choice of parameters for the in this work developed algorithms have been the following:

- MS-KSVD-D algorithm:
  - scales: 3
  - dictsize: 256
  - maxval: 255
  - trainum: 10000
  - iternum: 20
  - fill_needle_holes: 0

[2] http://opencv.willowgarage.com/documentation/c/imgproc_miscellaneous_image_transformations.html

  - max_depth_layer: 6
  - remove: 0
  - initdict: global
  - ksvd.epsilon: 7.44
  - ksvd.sparsity: 20
  - inpaint.epsilons: [0.0744, 0.7444, 7.4442]
  - inpaint.sparsity: [7, 7, 7]
  - muthresh: 0.99

- MultiScale SSD with Depth (MS-SSD-D) algorithm:
  - scales: 3
  - maxval: 255
  - max_depth_layer: 6

4.4.2. Obtained Results

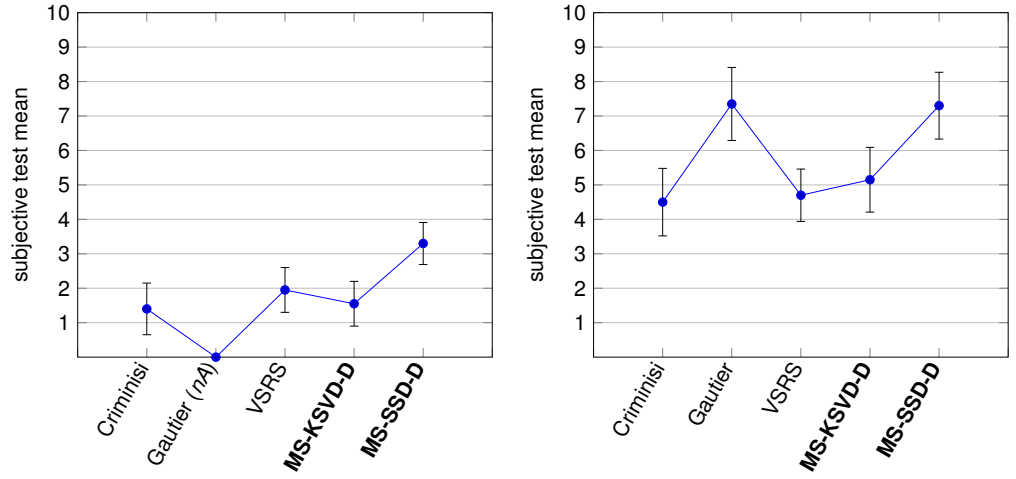
A total of 6 images have been evaluated with objective and subjective quality measures. The most demonstrative results are presented in the following paragraphs.

Test Image 1

As the main focus of this work is to inpaint large holes, a corresponding test image has been chosen (biggest single horizontal hole width >10% of image width): view 3 synthesized out of view 1 from frame 1 of the MSR ballet test sequence. The obtained objective and subjective results show a clear improvement over existing methods for the developped MS-SSD-D algorithm (see Table 4.5 and Fig. 4.8(a)). The results of the MS-KSVD-D algorithm lie within the range of other algorithms. However, when looking at the images, only the results of the MS-SSD-D algorithm come close to be acceptable for this size of holes (see Fig. 4.9). No results were obtained from the algorithm of Gautier et al. as the software was crashing with the test image.

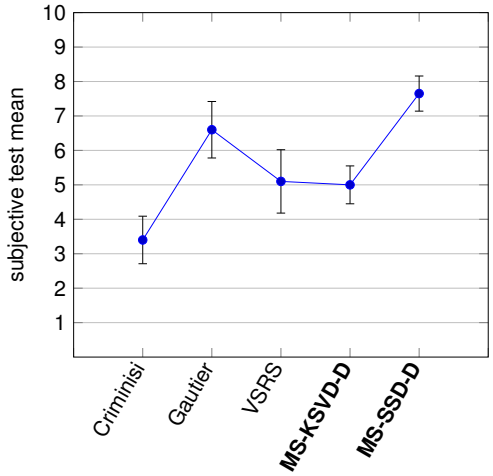
Evaluation measure	Criminisi [1]	Gautier [6]	VSRS [58]	**MS-KSVD-D**	**MS-SSD-D**
PSNR [dB]	28.53	*nA*	29.82	**28.53**	**31.39**
MSSIM	0.790	*nA*	0.821	**0.811**	**0.841**
Q-MOS	10.44	*nA*	15.90	**11.45**	**27.90**
Subjective test mean	1.40	*nA*	1.95	**1.55**	**3.30**

Table 4.5.: Comparison of objective and subjective quality measures of different inpainting algorithms (test image: view 3 synthesized out of view 1 from frame 1 of the MSR ballet test sequence).



(a) Test image 1: view 3 synthesized out of view 1 from frame 1 of the MSR ballet test sequence

(b) Test image 2: view 3 synthesized out of view 2 from frame 1 of the MSR breakdancer test sequence

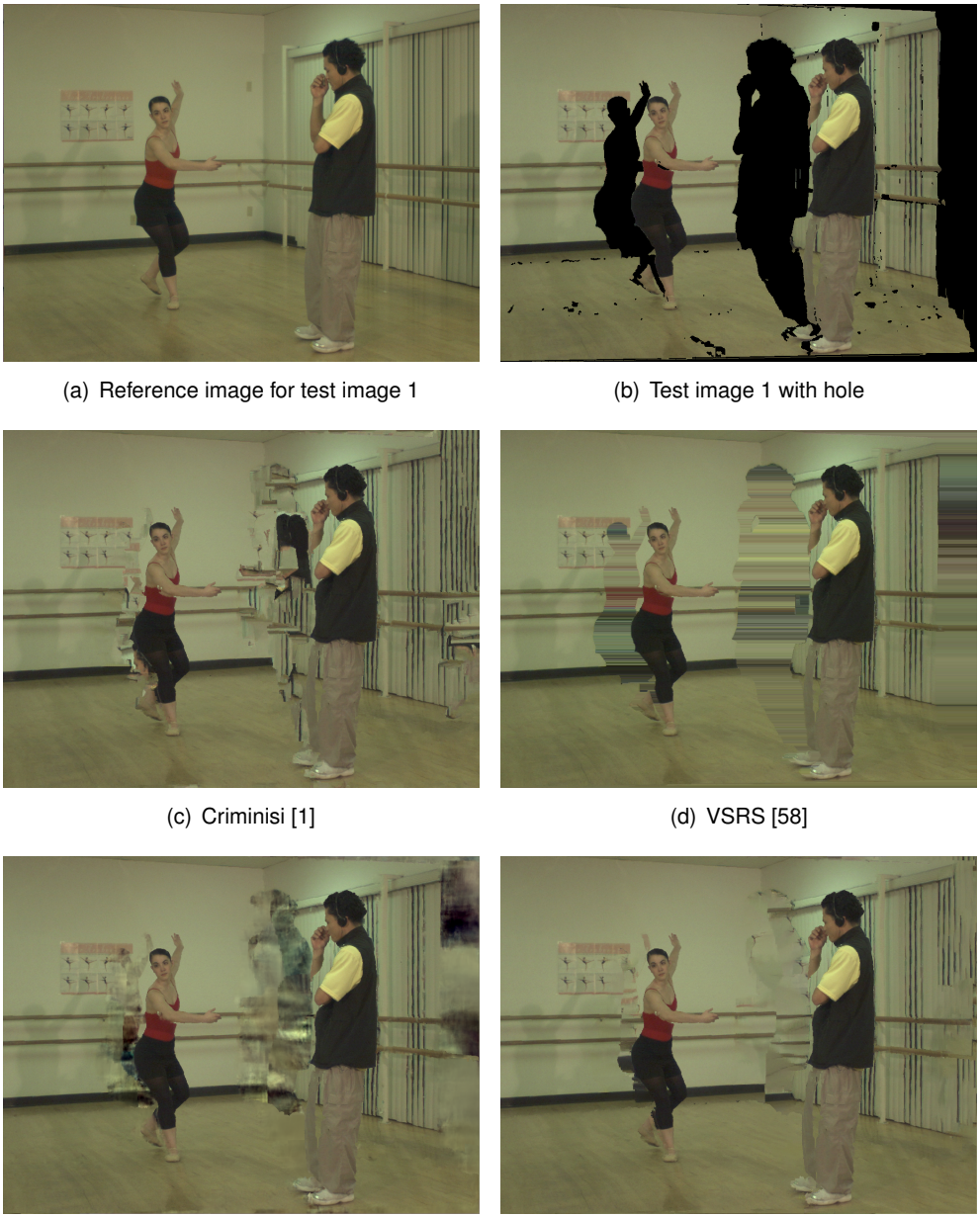


(c) Test image 3: view 3 synthesized out of view 4 from frame 1 of MPEG cafe test sequence in Full HD

Figure 4.8.: Comparison of subjective quality votes of different inpainting algorithms for the test images.

Test Image 2

The second chosen test image is from the MSR breakdancer test sequence (view 3 synthesized out of view 2 from frame 1). The size of holes is much smaller then in the previous test image (biggest single horizontal hole width ≈3% of image width) and the overall image quality is worse (see Fig. 4.10(a)). Here again, the MS-SSD-D algorithm gives the best



(a) Reference image for test image 1

(b) Test image 1 with hole

(c) Criminisi [1]

(d) VSRS [58]

(e) MS-KSVD-D

(f) MS-SSD-D

Figure 4.9.: Inpainting results from different inpainting algorithms for test image 1.

results (see Table 4.6 and Fig. 4.8(b)). The proposed MS-KSVD-D algorithm is on the lower end of the tested algorithms.

Evaluation measure	Criminisi [1]	Gautier [6]	VSRS [58]	**MS-KSVD-D**	**MS-SSD-D**
PSNR [dB]	32.67	34.85	35.83	**31.23**	**36.00**
MSSIM	0.927	0.937	0.941	**0.933**	**0.944**
Q-MOS	15.01	31.57	47.11	**13.20**	**59.13**
Subjective test mean	4.50	7.35	4.70	**5.15**	**7.30**

Table 4.6.: Comparison of objective and subjective quality measures of different inpainting algorithms (test image: view 3 synthesized out of view 2 from frame 1 of the MSR breakdancer test sequence).

(a) Test image 2 with hole

(b) Test image 2 with hole

Figure 4.10.: Test images 2 and 3 with hole.

Test Image 3

To complete the range of test images, a Full HD image of the MPEG cafe test sequence was selected (view 3 synthesized out of view 4 from frame 1). Like in test image 2, the holes are smaller (biggest single horizontal hole width ≈2.5% of image width, see Fig. 4.10(b)). The MS-SSD-D algorithm shows again the best results but also the MS-KSVD-D algorithm performs reasonably well (see Table 4.7 and Fig. 4.8(c)). An interesting comparison can be done when looking at the hole at the left side of the image: Whereas the exemplar-based methods of Criminisi et al. [1] and the MS-SSD-D algorithm perform well (see Fig. 4.11(c) and Fig. 4.11(g)), the result from VSRS does not seem realistic (see Fig. 4.11(e)). Due to fact that the inpainting direction has to be chosen in the opposite direction of the camera movement, the algorithm from Gautier et al. [6] is not able to inpaint the border (see Fig. 4.11(d)). The proposed MS-KSVD-D algorithm shows it strength when inpainting the structured background (see Fig. 4.11(f)).

Evaluation measure	Criminisi [1]	Gautier [6]	VSRS [58]	**MS-KSVD-D**	**MS-SSD-D**
PSNR [dB]	32.44	27.07	33.27	**32.74**	**32.99**
MSSIM	0.954	0.944	0.960	**0.953**	**0.955**
Q-MOS	49.38	18.72	67.01	**61.68**	**67.17**
Subjective test mean	3.40	6.60	5.10	**5.00**	**7.65**

Table 4.7.: Comparison of objective and subjective quality measures of different inpainting algorithms (test image: view 3 synthesized out of view 4 from frame 1 of MPEG cafe test sequence in Full HD).

(a) (b) (c) (d) (e) (f) (g)

Figure 4.11.: Inpainting results from different inpainting algorithms for the left border of test image 3. From left to right: reference image, image with hole, Criminisi, Gautier, VSRS, MS-KSVD-D, MS-SSD-D.

5. Conclusion and Future Work

In this report, a new inpainting algorithm was developed with the goal to produce better results for filling holes that occur during 3-D view synthesis. Two main directions were set at the beginning: the work of Mairal et al. [2] on Sparse Representations for color image inpainting (the K-SVD algorithm) and the work of Criminisi et al. [1] on exemplar-based inpainting with patch priorities.

The origin of both methods can be found in the human visual perception system. The concept of structure propagation of Criminisi et al. through filling priority assignment based on the strength of isophotes corresponds to the "boundary completion" during perceptual filling in the human vision. The exemplar-based filling step itself through SSD can be seen as "featural completion". Mairal et al. presented a concept that relies on Sparse Coding with an overcomplete code. As discussed, this efficient coding method can also be found in the visual cortex.

The probabilistic approach behind the theory of Sparse Representations was presented and out of it the algorithm from Mairal et al. was derived. As only a software implementation for the denoising of black-and-white images using sparse representation was available so far, the major modifications to adapt to the task of inpainting color images were detailed.

After having found the justifications for the use of the methods from Mairal et al. and Criminisi et al., a further detailed research and first implementation tests on the general subject of inpainting was conducted. Another two potential improvements were added to the new framework: using multiple scales for the filling process and taking advantage of the depth map required for DIBR.

The implementation of the new algorithm was realized in Matlab in form of a toolbox that takes parameters and images as input. The work of Mairal et al. and Criminisi et al. was combined and extended to the task of color image inpainting. For the proposed improvements of using multiple scales and the depth map, efficient implementations were presented: In order to reduce computation complexity related to the multiscale approach during the dictionary update, the training patches in larger scales are resized to the usual smaller patch size. As for the filling, the dictionaries are then resized to the larger patch sizes. The larger images patches around a patch to inpaint for OMP are found in a smart way such that a larger patch contains as much as hole-less information as possible. After segmentation, the filling is performed depth-layer by depth-layer. Besides the dictionary-based method, another exemplar-based method, MS-SSD-D, has been developed. It is based on the work of Criminisi et al. and the proposed improvements on multiple scales and the use of depth information was adapted.

For the evaluation, three different objective quality measures are used: classical PSNR, SSIM (that measures the similarity in terms of structural information between two images) and HDR-VDP (which has proven to give more similar results as SSIM regarding human

perception). With these metrics, the efficiency of the proposed MS-KSVD-D method (color inpainting with a K-SVD learned multiscale dictionary and depth-information) compared to single-depth-layer and -scale inpainting with K-SVD and filling priorities has been shown.
The overall inpainting quality of both developed methods (MS-KSVD-D and MS-SSD-D) has been compared to state-of-the-art inpainting algorithms through objective and subjective measures. Even though sparse coding is an essential part of the human visual system, the MS-KSVD-D algorithm is not fulfilling the expectations. The exemplar-based methods perform better in most of the tests. In particular the new developed MS-SSD-D algorithm gives the best results of all tested methods.
Finally, the starting point to inpaint the size of holes that can occur during 3-D view synthesis using a concept that learns a dictionary with K-SVD and filling priorities (MS-KSVD-D), could not be confirmed. However, a promising approach for exemplar-based inpainting (MS-SSD-D) has been proposed that takes advantage of multiple scales and depth information as it has not been done before. Both in terms of objective and subjective inpainting quality of selected test images, it outperforms available state-of-the-art algorithms.
As a future outlook, a major improvement can be proposed: The quality of the depth map has a big impact on the results. In many test images, after the view synthesis, borders from foreground objects cause wrong structure and texture propagation. Depth map refinement, for example by combining depth and color information, could be advantageous.

6. Acronyms

ATTEST Advanced Three-dimensional Television System Technologies

CRF Classical Receptive Field

DIBR Depth-Image Based Rendering

DSIS Double-Stimulus Impairment Scale

HDR-VDP High Dynamic Range Visual Difference Predictor

LASSO Least Absolute Shrinkage and Selection Operator

LGN Lateral Geniculate Nucleus

MSE Mean Squared Error

MSSIM Mean Structural SIMilarity

MSR Microsoft Research

MS-KSVD-D MultiScale K-SVD with Depth

MS-SSD-D MultiScale SSD with Depth

nCRF non-Classical Receptive Field

ODCT Overcomplete Discrete Consinus Transformation

OMP Orthogonal Matching Pursuit

PDE Partial Differential Equation

PSNR Peak Signal-to-Noise Ratio

Q-MOS Quality Mean-Opinion-Score

RANSAC RANdom Sample And Consensus

SAD Sum of Absolute Differences

SfS Structure-from-Stereo

SSIM Structural SIMilarity

SSD Sum of Squared Differences

SVD Singular Value Decomposition

VSRS View Synthesis Reference Software

A. Evaluation Sheet

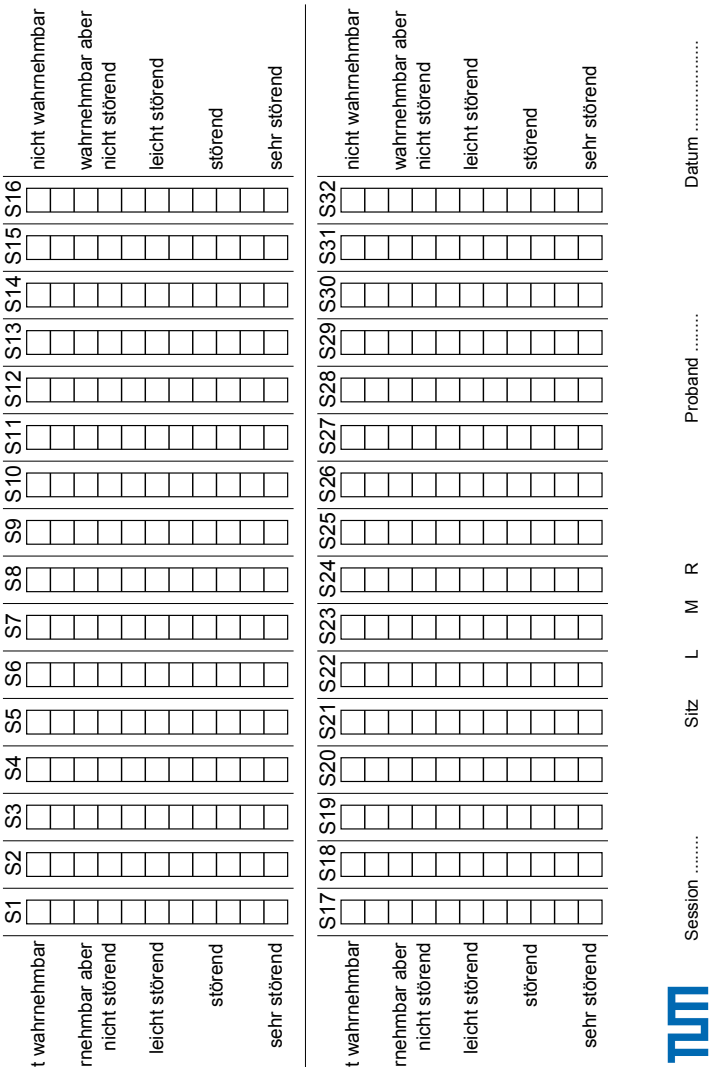


Figure A.1.: The evaluation sheet for the subjective tests.

Bibliography

[1] Antonio Criminisi, Patrick Perez, and Kentaro Toyama. Object removal by exemplar-based inpainting. Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, pages 721–8, Madison, WI, 2003.

[2] Julien Mairal, Michael Elad, and Guillermo Sapiro. Sparse representation for color image restoration. *IEEE Trans. Image Process.*, 17(1):53–69, January 2008.

[3] André Redert, Marc Op De Beeck, Christoph Fehn, Wijnand Ijsselsteijn, Marc Pollefeys, Luc Van Gool, Eyal Ofek, Ian Sexton, and Philip Surman. ATTEST : Advanced three-dimensional television system technologies. Proc. 1st Int. Symp. on 3D Data Processing Visualization and Transmission, pages 313–319, Padova, Italy, 2002.

[4] Kai Luo, Dong-xiao Li, Ya-mei Feng, and Ming Zhang. Depth-aided inpainting for disocclusion restoration of multi-view images using depth-image-based rendering. *Journal of Zhejiang University - Science A*, 10(12):1738–1749, December 2009.

[5] Ismaël Daribo and Béatrice Pesquet-Popescu. Depth-aided image inpainting for novel view synthesis. IEEE Int. Workshop on Multimedia Signal Process., pages 167–170, Saint-Malo, France, 2010.

[6] Josselin Gautier, Olivier Le Meur, and Christine Guillemot. Depth-based image completion for view synthesis. 3DTV Conf.: The True Vision - Capture, Transmission and Display of 3D Video, pages 1–4, Antalya, Turkey, 2011.

[7] Silvano Di Zenzo. A note on the gradient of a multi-image. *Computer Vision, Graphics, and Image Process.*, 33(1):116–125, January 1986.

[8] Kwan-jung Oh, Sehoon Yea, and Yo-sung Ho. Hole filling method using depth based in-painting for view synthesis in free viewpoint television and 3-D video. Picture Coding Symp., pages 1–4, Chicago, IL, 2009.

[9] Jinn-Cherng Yang. Improved novel view synthesis from depth image with large baseline. Proc. Int. Conf. on Pattern Recognition, pages 1–4, Tampa, FL, 2008.

[10] Zongben Xu and Jian Sun. Image inpainting by patch propagation using patch sparsity. *IEEE Trans. Image Process.*, 19(5):1153–1165, May 2010.

[11] Marcelo Bertalmio, Andrea L. Bertozzi, and Guillermo Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, pages 355–362, Kauai, HI, 2001.

[12] Yonatan Wexler, Eli Shechtman, and Michael Irani. Space-time completion of video. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(3):463–476, March 2007.

[13] Alexandre Hervieux, Nicolas Papadakis, and Aurélie Bugeau. Stereoscopic image inpainting using scene geometry. Proc. IEEE Int. Conf. on Multimedia and Expo, pages 1–6, Barcelona, Spain, 2011.

[14] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD : An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Process.*, 54(11):4311–4322, November 2006.

[15] Julien Mairal, Guillermo Sapiro, and Michael Elad. Multiscale sparse image representation with learned dictionaries. *Multiscale Modeling and Simulation*, 7(1):214–241, March 2008.

[16] Bin Shen, Wei Hu, and Yu-jin Zhang. Image inpainting via sparse representation. Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Process., pages 697–700, Taipei, Taiwan, 2009.

[17] Hakan Urey, Kishore V. Chellappan, Erdem Erden, and Phil Surman. State of the art in stereoscopic and autostereoscopic displays. *Proc. IEEE*, 99(4):540–555, April 2011.

[18] Karsten Müller, Philipp Merkle, and Thomas Wiegand. 3-D video representation using depth maps. *Proc. IEEE*, 99(4):643–656, April 2011.

[19] International Telecommunication Union. H264: Advanced video coding for generic audiovisual services. *ITU-T Recommendations H.264 and ISO/IEC 14496-10 (MPEG-4 AVC)*, 9, 2009.

[20] Nicole Brosch, Asmaa Hosni, Geetha Ramachandran, Liu He, and Margrit Gelautz. Content generation for 3D video/TV. *e & i Elektrotechnik und Informationstechnik*, 128(10):359–365, October 2011.

[21] Aljoscha Smolic, Peter Kauff, Sebastian Knorr, Alexander Hornung, Matthias Kunter, Marcus Müller, and Manuel Lang. Three-dimensional video postproduction and processing. *Proc. IEEE*, 99(4):607–625, April 2011.

[22] Steven M. Seitz and Charles R. Dyer. View morphing. Proc. 23rd Annu. Conf. on Computer Graphics and Interactive Techniques, pages 21–30, New Orleans, LA, 1997.

[23] Christoph Fehn. A 3D-TV approach using depth-image-based rendering (DIBR). Proc. 3rd IASTED Conf. on Visualization, Imaging, and Image Process., pages 482–487, Benalmadena, Spain, 2003.

[24] Lars Schwabe and Klaus Obermayer. Modeling the adaptive visual system: a survey of principled approaches. *Neural Networks : The Official Journal of the Int. Neural Network Society*, 16(9):1353–1371, November 2003.

[25] Vilayanur S. Ramachandran and Richard Langton Gregory. Perceptual filling in of artificially induced scotomas in human vision. *Nature*, 350(6320):699–702, April 1991.

[26] Luiz Pessoa, Evan Thompson, and Alva Noë. Finding out about filling-in: a guide to perceptual completion for visual science and the philosophy of perception. *Behavioral and Brain Sciences*, 21(6):723–748, December 1998.

[27] Bruno A. Olshausen and David J. Field. Sparse coding of sensory inputs. *Current Opinion in Neurobiology*, 14(4):481–487, August 2004.

[28] Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision Research*, 37(23):3311–3325, December 1997.

[29] Peter Dayan and Larry F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. The MIT Press, Cambridge, MA, 1st edition, 2001.

[30] William E. Vinje and Jack L. Gallant. Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*, 287(5456):1273–1276, February 2000.

[31] Joel A. Tropp and Anna C. Gilber. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Inf. Theory*, 53(12):4655–4666, December 2007.

[32] Michael Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer, New York, 1st edition, 2010.

[33] Nathan Srebro and Tommi Jaakkola. Weighted low-rank approximations. Proc. 20th Int. Conf. on Machine Learning, pages 720–727, Washington, DC, 2003.

[34] Ismaël Daribo and Hideo Saito. A novel inpainting-based layered depth video for 3DTV. *IEEE Trans. on Broadcast.*, 57(2):533–541, June 2011.

[35] Chia-Ming Cheng, Shu-Jyuan Lin, Shang-Hong Lai, and Jinn-Cherng Yang. Improved novel view synthesis from depth image with large baseline. Proc. 19th Int. Conf. on Pattern Recognition, pages 1–4, Tampa, FL, December 2008.

[36] Richard J. Cant and Caroline S. Langensiepen. A multiscale method for automated inpainting. Proc. 17th European Simulation Multiconf., pages 148–153, Nottingham, United Kingdom, June 2003.

[37] Olivier Le Meur and Christine Guillemot. Super-resolution-based inpainting. Proc. 12th European Conf. on Computer Vision, to be published, Firenze, Italy, 2012.

[38] Rui Wang, Dongbing Gu, Guangwen Liu, and Junxi Sun. A multi-scale image inpainting algorithm based on GMRF model. Proc. IEEE Int. Conf. on Robotics and Biomimetics, pages 1844–1848, Guilin, China, December 2009.

[39] Mashhour Solh. Hierarchical hole-filling (HHF): depth image based rendering without depth map filtering for 3D-TV. Proc. IEEE Int. Workshop on Multimedia Signal Process., pages 87–92, Saint-Malo, France, December 2010.

[40] Mashhour Solh and Ghassan AlRegib. Depth adaptive hierarchical hole filling for DIBR-based 3D videos. Proc. SPIE Three-Dimensional Image Process. (3DIP) and Applications, pages 829004–1–829004–11, Burlingame, CA, February 2012.

[41] Ron Rubinstein, Michael Zibulevsky, and Michael Elad. Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit. Technical report, August 2008. Last access on August 25th 2012.

[42] Sooraj Bhat. Object removal by exemplar-based inpainting, December 2004. Last access on July 12th 2012.

[43] Julien Mairal. KSVD package. Last access on September 26th 2012.

[44] James B. MacQueen. Some methods for classification and analysis of multivariate observations. *Proc. 5th Berkeley Symp. on Mathematical Statistics and Probability*, pages 281–297, 1967.

[45] David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1(2):224–227, February 1979.

[46] Luat Do, Sveta Zinger, and Peter H. N. de With. Quality improving techniques for free-viewpoint DIBR. *Proc. SPIE Stereoscopic Displays and Applications*, pages 75240I–75240I–10, 2010.

[47] Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, April 2004.

[48] Richard Dosselmann and Xue Dong Yang. A comprehensive assessment of the structural similarity index. *Signal, Image and Video Processing*, 5(1):81–91, March 2011.

[49] Rafal Mantiuk, Karol Myszkowski, and Hans-Peter Seidel. Visible difference predicator for high dynamic range images. *Proc. IEEE Int. Conf. on Systems, Man and Cybernetics*, pages 2763–2769, 2004.

[50] Rafal Mantiuk, Scott J. Daly, Karol Myszkowski, and Hans-Peter Seidel. Predicting visible differences in high dynamic range images: model and its calibration. *Proc. IS&T/SPIE 17th Annual Symp. on Electronic Imaging*, pages 204–214, 2005.

[51] Scott Daly. The visible differences predictor: An algorithm for the assessment of image fidelity. In Andrew B. Watson, editor, *Digital Image and Human Vision*, pages 179–206. The MIT Press, Cambridge, MA, 1993.

[52] International Telecommunication Union. Methodology for the subjective assessment of the quality of television pictures. *Recommendation ITU-R BT.500*, 12, 2010.

[53] Masayuki Tanimoto, Toshiaki Fujii, Kazuyoshi Suzuki, Norishige Fukushima, and Yuji Mori. Reference softwares for depth estimation and view synthesis. Technical report, MPEG, April 2008. ISO/IEC JTC1/SC29/WG11 M15377.

[54] Dong Tian, Purvin Pandit, and Cristina Gomila. Simple view synthesis. Technical report, MPEG, July 2008. ISO/IEC JTC1/SC29/WG11 M15696.

[55] Charles Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. on Graphics*, 23(3):600–608, August 2004.

[56] Sing Bing Kang and Charles Lawrence Zitnick. Projection test and results for microsoft research 3D video (breakdancing frame). Last access on September 19th 2012.

[57] MPEG Video Group. Report on experimental framework for 3D video coding. Technical report, MPEG, October 2010. ISO/IEC JTC1/SC29/WG11 W11631.

[58] MPEG 3D Video Coding Team. VSRS 3.0 software manual, May 2009.