Filling Disocclusions in Extrapolated Virtual Views Using Advanced Texture Synthesis Methods

vorgelegt von

Dipl.-Ing. Martin Köppel
geb. in Berlin

von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
- Dr.-Ing. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr.-Ing. Thomas Sikora
1. Gutachter: Prof. Dr.-Ing. Thomas Wiegend
2. Gutachter: Prof. Dr.-Ing. Peter Eisert
3. Gutachter: Prof. Dr. Ir. Peter H. N. de With

Tag der wissenschaftlichen Aussprache: 05. Juli 2017

Berlin 2017

Danksagung

Diese Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Fachgebiet Bildkommunikation der TU Berlin und als Gastwissenschaftler am Fraunhofer Heinrich-Hertz-Institut (HHI). An dieser Stelle möchte ich mich bei allen bedanken, die mich auf diesem langen und nicht immer sehr einfachen Weg unterstützt haben.

An erster Stelle möchte ich Dr.-Ing. Patrick Ndjiki-Nya danken, der mir die Möglichkeit gab, in dem Themengebiet "Textursynthese" zu arbeiten und der mich über viele Jahre unterstützt hat. Dank Dr.-Ing. Patrick Ndjiki-Nyas, war es überhaupt erst möglich, diese Doktorarbeit durchzuführen. Des Weiteren, möchte ich Prof. Dr.-Ing. Thomas Wiegend danken, der mich als wissenschaftlicher Mitarbeiter in seinem Fachgebiet aufnahm und zudem meine Doktorarbeit betreut.

Für die langjährige, erfolgreiche Zusammenarbeit möchte ich mich ganz besonders bei Dimitar Doshkov bedanken, durch dessen eigene Arbeiten und Anregungen sind erst viele neue Ideen entstanden. Wir hatten eine tolle gemeinsame Zeit am Fraunhofer HHI. Des Weiteren möchte ich Christoph Hoffmann, Mohan Liu, Dr. Fabien Racape, Dr. Haricharan Lakshmann, Oskar Jottrand, Xi Wang, Mehdi Ben Makhlouf, Dr.-Ing Karsten Müller, Marcus Müller, Dr.-Ing. Detlef Runde, Thomas Meiers, Marcus Aust, Antje Linnemann und Sebastian Gerke für die tolle Teamarbeit und die kollegiale Unterstützung danken. Für die konstruktiven fachlichen Diskussionen und Hinweise möchte ich mich darüber hinaus bei Dr.-Ing Karsten Müller bedanken. Für das Korrekturlesen der gesamten Arbeit danke ich Dr.-Ing. Detlef Runde und Dr.-Ing. Karsten Müller. Ein ganz lieber Dank richtet sich auch an Gabriela Thiele, die in dem Verwaltungschaos der TU Berlin immer den Überblick behalten hat.

Mein ganz besonderer und herzlicher Dank gilt meiner Frau Mandy Köppel. Sie hat mich über all die Jahre immer unterstützt und stand mir liebevoll zur Seite.

Kurzfassung

Stereoskopische 3-D Technologien haben sich mittlerweile im Mainstream etabliert. Viele Kinos zeigen bereits Filme in 3-D. Aufgrund des hohen visuellen Informationsgehaltes werden 3-D Video Technologien zudem immer häufiger in medizinische und logistische Applikationen integriert. Die Notwendigkeit, eine zusätzliche Brille tragen zu müssen, um einen 3-D Eindruck beim Betrachter zu erzeugen, wird jedoch als großes Hindernis für die Etablierung von 3-D Video im Heimbereich betrachtet. Ähnliches gilt für die Medizintechnik, wo Operationsgeräte oder Schutzbrillen die Verwendung zusätzlicher 3-D Brillen behindern. Neue Technologien, wie autostereoskopische Displays, ermöglichen es dem Zuschauer mittlerweile einen 3-D Eindruck zu vermitteln, ohne dass dieser eine zusätzliche Brille tragen muss. Hierbei werden mehrere Ansichten (derzeit 5-32) einer Szene aus leicht verschobenen Blickwinkeln ausgestrahlt. Da oft nur wenige (1-3) originale Kamerapositionen vorliegen, müssen die Ansichten für die fehlenden Positionen errechnet werden. Hierfür können Depth Image-based Rendering (DIBR) Verfahren verwendet werden. Diese synthetisieren eine Anzahl von unterschiedlichen Perspektiven der gleichen Szene, beispielsweise für das Multiview-Video-plus-Tiefe (MVD) Format. Das MVD Format besteht aus einer begrenzten Anzahl von Videosequenzen derselben Szene und deren zugehörigen Tiefenkarten. Ein Kernproblem beim Rendern mit wenigen Ansichten und den zugehörigen Tiefenkarten besteht jedoch darin, dass in den virtuellen Ansichten Bereiche sichtbar werden, die in allen Originalansichten verdeckt sind. Die vorgestellten Ansätze synthetisieren diese aufgedeckten Bereiche. Die Synthesizer berechnen die neuen Texturen unter Berücksichtigung räumlicher und zeitlicher Kohärenzen. Es werden Syntheseverfahren für Sequenzen mit statischen und dynamischen Hintergründen vorgestellt. Detaillierte Experimente zeigen, dass die vorgestellten Verfahren erhebliche objektive und subjektive Gewinne im Vergleich zu Verfahren erzielen, die dem aktuellen Stand der Technik entsprechen.

Abstract

Stereoscopic Three Dimensional (3-D) video technologies have been established in the mainstream. Many cinemas already show movies in 3-D. Due to the higher visual information, 3-D video technology is increasingly used in other application areas, e.g. medical and logistical applications. However, the need to wear additional glasses to create a 3-D impression for the viewer is regarded as a major obstacle for 3-D video in home environments. The same applies to medical technology where surgery devices or safety goggles hinder the use of additional 3-D glasses. New technologies such as autostereoscopic displays, however, allow the viewer to receive a 3-D impression without the need to wear additional glasses by showing a number of slightly different views (currently 5-32) simultaneously. Since usually only a few real cameras (1-3) are available, the missing views must be calculated. For this purpose, Depth Image-based Rendering (DIBR) can be used to synthesize a number of different perspectives of the same scene, e.g., from a Multiview Video plus Depth (MVD) representation. This MVD format consists of video and depth sequences for a limited number of original camera views of the same natural scene. An inherent problem of the view synthesis concept is the fact that image information which is occluded in the original views may become visible, especially in extrapolated views beyond the viewing range of the original cameras. The presented approaches synthesize these disoccluded textures. The synthesizers achieve visually satisfying results by taking spatial and temporal consistency measures into account. For this purpose, solutions for sequences with both static and dynamic backgrounds are presented. Detailed experiments show significant objective and subjective gains of the proposed methods in comparison to state-of-the-art approaches.

List of Figures

List of Tables

Glossary

General

Scalar values x, y are written in italic lowercase. Coordinate values are scalars and denoted as (x, y). Vectors $\mathbf{v}$ are bold lowercase and matrices $\mathbf{M}$ are bold capitals.

Basic Definitions

(x, y)	Position in an image or a depth map
F	A single frame
F_n	Frame with frame number n of a sequence, with $n \in \mathbb{N}$
$F_{c,n}$	A frame of a sequence at the spatial camera position c and with the frame number n, with $c \in \mathbb{R}$ and $n \in \mathbb{N}$
Ω	Unknown area in a frame
$\delta\Omega$	The outer boundary of the hole in the frame
F^o	Original texture in a frame with $F^o \subset F \backslash \Omega$
F^s	Synthesized texture in a frame
$disp$	Disparity value
D	A single depth map
D_n	Depth map with frame number n of a sequence, with $n \in \mathbb{N}$
$D_{c,n}$	A depth map of a sequence at the spatial camera position c and with the frame number n, with $c \in \mathbb{R}$ and $n \in \mathbb{N}$
Γ	Unknown area in a depth map
$\delta\Gamma$	The outer boundary of the hole in the depth map
D^o	Original depth values in a depth map with $D^o \subset D \backslash \Gamma$
b	Camera baseline
l	Focal length
N	Number of frames in a sequence

View Synthesis Method for Sequences with Static Background

m	Size of the squared neighborhood ($m \times m$) used in the k-means depth map clustering
S	Background sprite
G	Depth map sprite
β	Parameter to allow some variance in the local background depth value comparison
f^*	A known scalar function over the domain F
f	An unknown scalar function defined over Ω
R	Texture source to be (partially) mapped onto Ω
g	A function defined over the texture source R to be (partially) mapped onto Ω
Δ	Laplace operator
γ	Parameter to identify small holes
$P(x, y)$	Patch Priority of the patch centered at (x, y)
$T_{\text{Conf}}(x, y)$	Confidence Term of the patch centered at (x, y)
$T_{\text{Data}}(x, y)$	Data Term of the patch centered at (x, y)
$\mathbf{u}(x, y)$	Unit vector at position (x, y) orthogonal to $\delta\Omega$
$\nabla^{\perp}$	Direction of the isophotes
$\mathbf{\Psi}_{(x,y)}$	Patch with unknown sample positions and its center in $\delta\Omega$
$\mathbf{\Psi}_{(\widehat{x},\widehat{y})}$	Patch with the highest priority
$\mathbf{\Psi}_{(u,v)}$	Candidate patches centered at (u, v)
$\boldsymbol{\psi}^{(u,v)}$	Vectorized version of $\mathbf{\Psi}_{(u,v)}$
$\boldsymbol{\psi}^{(\widehat{x},\widehat{y})}$	Vectorized version of $\mathbf{\Psi}_{(\widehat{x},\widehat{y})}$
$\mathbf{\Psi}_{(\widehat{u},\widehat{v})}$	The best candidate patch with its center in $(\widehat{u}, \widehat{v})$
A	Source area
s	Sub-sampling factor

View Synthesis Method for Sequences with Global Background Motion

N_p	Number of frames in a sequence that can be computed using the Group-of-Pictures structure
z	Number of frames in a Group-of-Pictures
$h(x, y, t)$	Image registration: source image
$h(\hat{x}, \hat{y}, t-1)$	Image registration: target image
$\varphi_1, \varphi_2, \varphi_3, \varphi_4$	Affine parameters
φ_5, φ_6	Translation parameters
φ_7, φ_8	Parameters that model the change of contrast and brightness
ι	Small spatial neighborhood

Hybrid View Synthesis Method

$\mathscr{D}$	Total variation in a window
$\mathscr{L}$	The overall spatial variation
ν	A very small value to prevent divisions by zero
H	The pre-processed image
$\widehat{F}(x, y)$	Synthesized sample at position (x, y)
y_{min}, y_{max}, x_{min}, x_{max}	Order of the model
$\alpha_{i,j}$	The prediction coefficients with $j \in [y_{min}, y_{max}]$ and $i \in [x_{min}, x_{max}]$
ϵ	White noise process with zero mean and variance
σ^2	Variance
Ω_y	Height of the hole
b_x, b_y	Training area divided into blocks of size $b_x \times b_y$
N_{b_y}	Number of blocks in vertical direction
μ_{block}	Mean value of the pixels in the block
σ^2_{block}	Variance of the pixels in the block
C	Causal neighborhood
S	Number of samples in the sub-training area

Experimental Results

M_n	Motion difference between frame n and $n-1$
$\max_{time}$	Maximum over a set of frames
std_{space}	Standard deviation over space
κ_n	Average changes of the inter-frame samples in the hole regions in frame n
κ	Flickering of the whole sequence
$\lvert\Omega\rvert$	Total number of hole pixels in a frame

Abbreviations

2-D	Two Dimensional
3-D	Three Dimensional
3DV	3-D Video
AR	Autoregressive
ARMA	Autoregressive Moving Average
DIBR	Depth Image-based Rendering
EM	Expectation-Maximization
FDF	Frame Differential Flicker
GOP	Group-of-Pictures
IBR	Image-based Rendering
ITU	International Telecommunication Union
MA	Moving Average
MMTC	Multimedia Communications Technical Committee
MPEG	ISO/IEC Moving Picture Experts Group
MRF	Markov Random Field
MVD	Multiview Video plus Depth
NCAR	Non-Causal Autoregressive
PDE	Partial Differential Equation
PSNR	Peak Signal-to-Noise Ratio
RTV	Relative Total Variation
SCM	Spatial Consistency Metric

SSD Sum of Squared Differences

SSIM Structural Similarity

TI Temporal perceptual Information

VCEG ITU-T Video Coding Experts Group

VSRS View Synthesis Reference Software

Contents

1 Introduction

1.1 Motivation

In the last years, 3-D Video (3DV) has become popular in the mainstream. Nowadays, most cinemas show stereoscopic Three Dimensional (3-D) movies and they are well accepted by the audience. In recent years, visual 3-D devices, such as displays and sensors, are also frequently utilized in logistic and medical devices [ZRDdWPHN12]. In order to produce a 3-D impression for the viewers, it is necessary for observers to receive different views for each eye. These views have to show the scene from slightly different viewpoints. The 3-D displays emit two or more different images captured from distinct viewpoints, which are often separated by a baseline corresponding to the human eye distance. In order to separate the two views for the left and the right eye, anaglyph 3-D glasses have been used in the past. Nevertheless, such glasses only provide an inadequate quality since they only utilize a color subset for each image. Nowadays, shutter or polarized glasses are used, because they provide full-colored images. However, the need to wear additional glasses is often seen as a disadvantage for a broader acceptance of stereoscopic 3DV in a home environment. This is also true for medical or logistic use cases, since surgery equipment or safety goggles can hinder the usage of additional 3-D glasses. To overcome this disadvantage, new multiview autostereoscopic displays have been developed in recent years.

Autostereoscopic displays provide a 3-D impression to several observers simultaneously without the necessity to wear additional glasses. They emit many views of one scene from slightly different viewpoints, e.g. 8, 9, 28 or 32 [Ali13, Tos13, Dim13] different views at the same time. Nevertheless, it is expected that future autostereoscopic displays will provide significantly more views simultaneously (e.g. 50 or more views) [DSW+13]. Thus, the viewer can observe the scene from different viewpoints having a realistic viewing experience.

There exist two main ways of realizing the spatial separation of the views for autostereoscopic displays (cf. Fig. 1.1) (1) An array of cylindrical lenslets is placed in front of the pixel raster, directing the light from adjacent pixel columns to different viewing slots at the ideal viewing distance such that each of the observers eyes sees light only from every second pixel column [cf. Fig. 1.1 (a)]. (2) A parallax barrier mask is placed in front of the pixel raster so that each eye sees light only from every second pixel column [Dod05] [cf. Fig. 1.1 (b)].

However, transmitting or storing all of the required videos would be extremely inefficient. Therefore, a generic transmission format, the MVD format, was specified [ISO11], comprising video data and their associated depth information for a few views (usually two to three views).

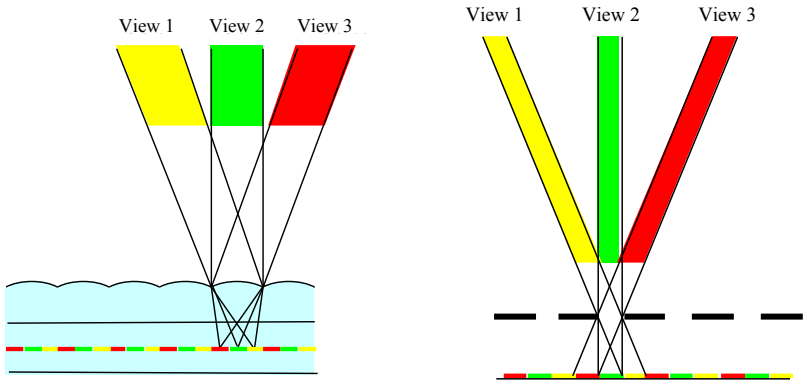


Figure 1.1: Methods of manufacturing autostereoskopic displays. (a) An array of cylindrical lenslets is placed in front of the pixel raster. (b) Parallax barrier. A barrier mask is placed in front of the pixel raster.

Joint video and depth coding methods for the MVD format were standardized in the joint JCT-3V group of ISO/IEC Moving Picture Experts Group (MPEG) and the ITU-T Video Coding Experts Group (VCEG). Here, additional virtual views for each individual display need to be calculated from the limited set of original cameras.

Recently, Depth Image-based Rendering (DIBR) techniques have become popular for this purpose. Based on the principles of projective geometry, arbitrary virtual views can be generated via 3-D projection or 2-D warping. The position of a warped virtual camera can be between (interpolation) or beside (extrapolation) the viewing range of the original cameras (cf. Fig. 1.2 and 1.3). Nevertheless, DIBR methods still show some shortcomings. One of the most significant problems in DIBR is the handling of uncovered areas (holes) in the virtual views, especially in extrapolated views beyond the viewing range of the original cameras (cf. Fig. 1.2 and 1.3). In the extrapolated views, large image regions may become visible, which are covered by foreground objects or are out-of region in the original cameras. However, the usage of extrapolated views is of fundamental importance, because, if the virtual views are only interpolated within the original views, the 3D impression is flattened. By also using extrapolated views, the full stereo sensation can be supported for both current and future viewing devices [Vid10, MJ11]. Hence, in this thesis, different approaches are proposed which can be used to synthesize the large uncovered texture areas in the extrapolated virtual views in a visually plausible manner.

1.2 Main Contributions to the State-of-the-Art

This thesis presents new methods for handling uncovered areas in synthesized views for 3-D video. Hereby, solutions for one of the most challenging problem in DIBR are presented. It is shown that disocclusions, which especially occur in extrapolated views with large baselines, can be

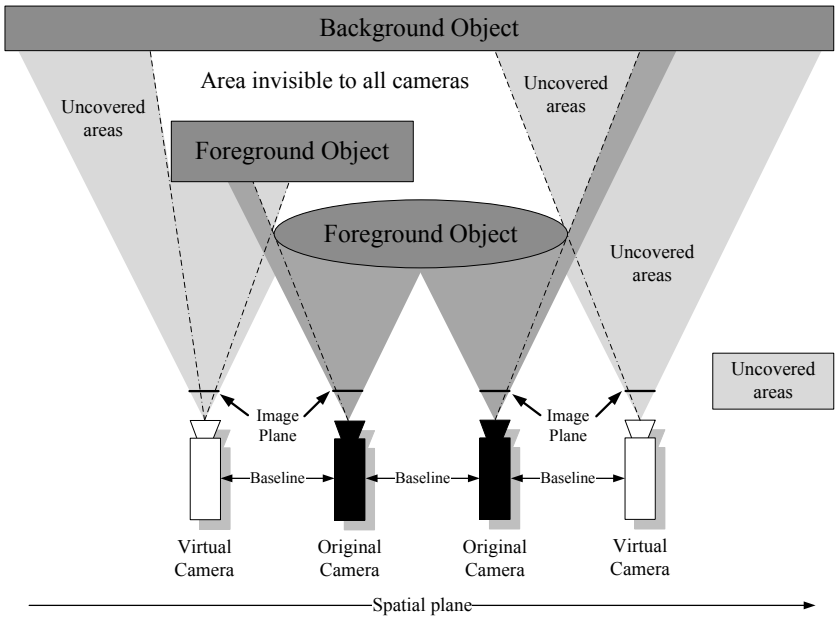


Figure 1.2: Extrapolated virtual camera views from original views. © [2012] IEEE.

successfully reconstructed in a visually plausible manner. In the proposed approaches, uncovered areas in extrapolated virtual views are specifically targeted because the uncovered areas are large. In an interpolation scenario where the virtual view is placed in-between two original cameras, the uncovered areas are rather small and original texture information is usually available from both sides, i.e. from a left and right original view. In the following the main contributions to the state-of-the-art achieved in this thesis are summarized:

- For sequences with a static background, a new method based on sprites is presented that takes image information from preceding frames into account. By using the sprite technology, temporally consistent synthesis results are obtained. The sprite stores background information from processed frames and is updated with the background image information from the current frame. The holes in the actual processed picture are updated from the background sprite. In order to account for illumination variation, the covariant cloning method is utilized to fit the background sprite samples to the intensity distribution in the relevant neighborhood of the current picture. The boundary condition of the cloning method is modified in such a way that only background samples are considered (cf. Sec. 3.3).

- An effective method to fill the uncovered areas in the virtual depth map is presented. Hereby, the spatial neighborhood of the unknown samples is clustered in order to enable an appropriate depth value selection. This ensures a consistent recovery of the uncovered areas in the depth map. During the texture synthesis stage, the virtual depth map is used to steer the filling procedure. By using this method, both the virtual depth map and the

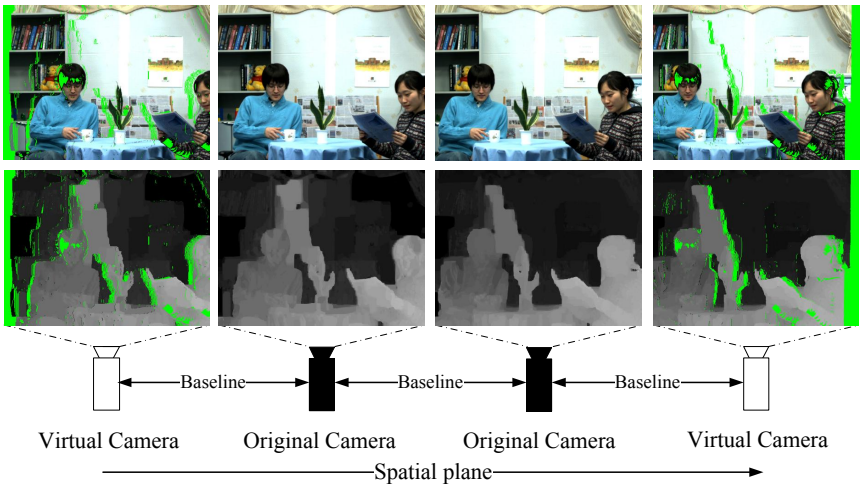


Figure 1.3: Disoccluded areas. The virtual cameras are warped from the original ones using the associated depth values. The disoccluded areas are marked green. © [2016] IEEE.

synthesized view contain fewer synthesis artifacts (cf. Sec. 3.2).

- To fill uncovered areas in the virtual view, a new texture synthesis method is presented. First, the missing textures are roughly estimated using an initialization method, which is based on the statistical properties of known samples in the vicinity of the hole. Then, patch-based texture synthesis is utilized to refine the initialized areas with optimal patches from background regions. To ensure smooth transitions between adjacent patches, an efficient post-processing method, based on covariant cloning, is utilized. The post-processing approach is adapted to the synthesis method in such a new manner that foreground objects are not considered as boundary samples. The new synthesis algorithm ensures an appropriate filling of unknown textures and reduces artifacts such as garbage-growing and blocking artifacts (cf. Sec. 3.4 and 3.5).

- A new warping structure is proposed, which improves the spatial consistency between adjacent virtual views. First, the original views are extrapolated to the left and to the right outermost virtual positions. Then, the uncovered areas in these outermost views are synthesized using a complex approach to achieve visually pleasing results. The remaining virtual views beyond the original camera range are interpolated between the outermost view and the original view utilizing classical, fast, synthesis methods. By using the new warping structure, the spatial consistency between adjacent extrapolated views is substantially improved and the overall complexity is reduced (cf. Sec. 4.2).

- For sequences which contain global background motion, a novel method to access the available image information from neighboring frames is presented. Temporal consistency in virtual sequences is achieved by considering a set of frames in a Group-of-Pictures (GOP)-like

structure derived from video coding. The global background motion is compensated utilizing a robust image registration method. High objective and subjective gains are achived by the proposed method compared to the state-of-the-art (cf. Sec. 4.4).

- To fill uncovered areas in the virtual view, a new hybrid texture synthesis approach is developed. This method combines the advantages of parametric and non-parametric (patch-based) algorithms. Patch-based texture synthesis is computationally expensive but is able to reconstruct a wide range of texture classes, while parametric approaches are faster but can reconstruct only a particular texture with a trained parameter set. Hence, an optimized patch-based texture synthesis method is used to separate different texture classes. Then, a fast Autoregressive (AR) parametric synthesis approach reconstructs each separate texture class. The proposed hybrid synthesis method computes a frame much faster than state-of-the-art patched-based texture synthesis providing similar subjective gains. Additionally, artifacts such as garbage-growing, caused by patch-based texture synthesis, are reduced (cf. Sec. 5.3).

- An effective method to select an appropriate texture area to train the parameters of the AR parametric synthesis method is presented. In order to select the optimal training area, a stationarity criterion, which discards unreliable parts, is developed. It is shown that significant visual improvements can be achieved, when gross instationarities are discarded (cf. Sec. 5.3.3).

- A new method for assessing the synthesis quality of parametrically reconstructed textures is presented. In case of detected errors, patch-based texture synthesis is used as fallback (cf. Sec. 5.3.7).

The approaches and results described in this thesis have been published in well known conference proceedings [NNKDW08, KDNN09, LNNK+09, LKNNW10, KNND+10, NNKD+10, BPLC+11b, BKP+11, KWD+12b, KWD+12a, KMNN13, KMMNN13, RKDNN14, RDKNN14] and journals [NNKD+11, BPLC+11a, KDR+15, KMW16]. The publication [KWD+12a] was presented in 2012 at the IEEE International Workshop on Multimedia Signal Processing and was counted as one of the best contributions of the conference. Therefore, the work was honored with the “Top 10 % paper award”. Furthermore, the IEEE Multimedia Communications Technical Committee (MMTC) recognized the journal paper [NNKD+11] as an outstanding contribution. Thus, the publication received the “MMTC Best Journal Paper Award” in 2013.

1.3 Overview

This thesis is organized as follows. In Ch. 2 state-of-the-art view synthesis and texture synthesis methods are presented.

The proposed framework for virtual view generation with time-consistent texture synthesis for sequences with static background is outlined in Ch. 3. The goal of the new view synthesis algorithm is to fill the disocclusions in both the virtual images and the virtual depth maps. The temporally consistency is achieved by using a background sprite which stores the original and the synthesized textures and depth values from previous frames.

In Ch. 4 a new spatial and temporal consistent view synthesis method is presented. The temporal consistency is achieved, by utilizing image information from previous and subsequent frames. By incorporating a robust image registration method into the framework, global background motion between temporally neighboring frame can be compensated. Additionally, a new method to improve the spatial consistency in the MVD format is developed.

A new hybrid view synthesis framework is presented in Ch. 5. In this approach, the advantages of parametric and non-parametric texture synthesis approaches are combined in a hybrid method. Patch-based texture synthesis is used to separate different classes of textures from each other. Then, a fast parametric method is applied to all separated textures individually. This method shows run-time, objective and subjective gains compared to state-of-the-art methods.

2 State-of-the-art

In this chapter, state-of-the-art technologies are described. First, inpainting and texture synthesis methods are presented in Sec. 2.2. Then, Image-based Rendering (IBR) and DIBR techniques are reviewed in Sec. 2.2. Finally, the warping procedure used in this thesis is described in Sec. 2.3.

2.1 Texture Completion Methods

The aim of texture completion algorithms is whether to fill an unknown region in an image/video (cf. Fig. 2.1) or to compute a large texture from a small input sample [EL99, WL00, KEBK05, TLD07, BCMS12, WLKT09]. Texture completion algorithms can be divided into three main categories: (1) Parametric, (2) Partial Differential Equation (PDE)-based, and (3) non-parametric algorithms. An overview of texture completion categories is given in Tab. 2.1 [NNDK$^+$12].

Parametric completion approaches approximate the probability density function of the texture source using a compact model with a fixed parameter set [PS00, HB95, DCWS03, CK85, Deg86, Tug94, JBS09, Kok04, SP96, KMA05, CSS08]. These methods extract statistics from the given input texture that are modeled based on a compact parameter set. Such approaches also provide information of the underlying texture properties, which can be relevant in identification and recognition applications. Some of the most commonly used parametric methods are based on the AR, Moving Average (MA) and the Autoregressive Moving Average (ARMA) models. The disadvantage of these methods is that they can only be applied to reconstruct texture classes on which the parameter set is trained on. This limits the usage to uniform texture classes. Texture transitions, boundaries or arbitrary texture classes can not be synthesized with a visual pleasing outcome.

The second texture completion methods category, termed PDE-based algorithms, employs a diffusion process to fill the missing image parts in a visually plausible manner. These techniques commonly use non-linear or high order PDEs to propagate information from the boundary towards the interior of the unknown area. Several approaches based on PDE have been developed in the last decade [BSCB00, BBC$^+$01, LZW03, BVSO03, BCMS12]. The advantage of these methods is that they consider gradients impinging on the hole. However, PDE based approaches introduce blur into the new texture. Hence, these methods are mainly used to reconstruct small scratches or homogenous textures [NN08].

The last class of methods, non-parametric completion approaches, do not explicitly model the

(a) (b) (c)

Figure 2.1: Generic texture synthesis. (a) Input image. (b) The area to be removed is marked black. (c) The missing region is synthesized using texture from the known sample positions.

Category	models	Completion of texture classes	Limitations	Complexity
Parametric	AR, MA, ARMA	Rigid and non-rigid	Structures	Medium
PDE-Based	PDE	Rigid, thin, elongated regions	Structures, smooth results	Medium
Non-parametric	MRF	Rigid and non-rigid	Prone to error	High

Table 2.1: Overview on texture completion approaches [NNDK$^+$12], visual quality and complexity limitations.

probability density function, but instead measure it from an available texture sample. In general, in this completion category, a best match is determined from a source region and copied to a target region [WL00, CPT04, Ash01, KSE$^+$03, NNSW07, NNKDW08, KA11, SYJS05]. The texture to be filled into the unknown area can be taken from the original texture in the same image or from an image database [HE07].

2.1.1 Texture Completion Problem

The major problems that have to be tackled in any texture completion process are roughly two-fold. The first one relates to the proper estimation of the underlying stochastic process of a given texture based only on a finite sample of it. The second task refers to the formulation of an efficient procedure (model) for generating new textures from a sample [WL00]. The former challenge steers the accuracy of the synthesized textures. The latter challenge determines the computational complexity of the texture generation procedure, also referred to as probability density function sampling.

A variety of texture models have been developed in the last years, the most successful models for imaging applications are based on the Markov Random Field (MRF) assumptions. The MRF model is characterized by statistical inter-relations within local vicinities. Thus the underlying generative stochastic process of the texture source is assumed to be both local and stationary. That

means, each sample of a texture pattern is characterized by a small set of spatially neighboring samples, and this characterization is the same for all samples. This can be formalized as:

$$p(T_i|I_{-i}) = p(T_i|\Psi_i) \tag{2.1}$$

The assumption is that any texture pattern T_i extracted from a given sample I, i.e. the image area, at location i can be predicted from the corresponding neighborhood Ψ_i and is independent of the rest of the texture. The homogeneity property of the MRF assumes that the conditional probability $p(T_i|\Psi_i)$ is independent of the pixel location.

2.1.2 Autoregressive Modelling

Three decades ago, the AR model, traditionally used on temporal signals, started being utilized for image processing, e.g. in the area of image and video texture completion. In the work of Chellappa et al. [CK85], a Two Dimensional (2-D), Non-Causal Autoregressive (NCAR) model was used to synthesize different texture samples, sized 64×64 with several neighbor sets and parameters. The authors show that the AR model can reproduce natural textures. A similar contribution by Deguchi [Deg86] focuses on texture characterization and completion of gray-level textures, using the same NCAR model as [CK85]. The basic properties of the model, the algorithm and the model identification problem are discussed. Furthermore, the work of Deguchi was at the time an innovative texture segmentation approach, where blocks with similar AR parameters were merged iteratively. In [Tug94], Tugnait investigated the applicability of 2-D NCAR models with asymmetric support for the completion of 128×128 real life textures. Here, the AR model is fitted to textures with abstracted mean value, i.e. with zero mean. The removed mean is finally added back into the synthetic image. In [JBS09], the authors used causal and non-causal neighborhoods for AR parameter estimation and texture pattern generation. Thus, different image textures were successfully synthesized using a given set of models and parameters.

In computer vision, the AR model has also been used in image and video reconstruction applications. A statistical framework for filling gaps in images is presented in [Kok04]. The method proposed in that paper relies on an iterative algorithm with a block based model estimation and pixel based filling. Visual results show effective reconstructions of images with thin elongated holes. In [KP94], Kokaram and Rayner proposed a 3-D AR model which is utilized to remove blotches in old film material. They developed an interpolation method which considers all texture areas in the extended boundary of the hole. In such a way the hole is filled with data that corresponds to the boundary. Furthermore, an improved version of this interpolation method [KP94] was proposed in [Kok98]. Janssen et al. [JVV86] developed a deterministic approach to fill missing samples given the AR coefficients. To determine the coefficients, Janssen et al. [JVV86] considered all known samples in the boundary region of the hole. In the work of Szummer [SP96], the temporal textures were modeled by a spatio-temporal AR model. The

authors show that the AR model can be used to synthesize video sequences, using large causal neighborhoods containing over 1000 parameters. The approach reproduces the dynamics of most input sequences effectively. The algorithm proposed in [SP96] is also used to recognize the type of content of temporal textures, by applying a purely spatial AR model. During the recognition tests, textures were correctly categorized at 95% as belonging to a certain texture class.

2.1.3 Non-Parametric Texture Synthesis

Non-parametric texture synthesis is based on the seminal work on texture synthesis proposed by Efros et al. [EL99] and Wei et al. [WL00]. The aim of the methods was to improve the recovery of missing texture in images. Non-parametric texture synthesis methods have been largely inspired by local region growing methods that grow a texture one pixel or one patch per step while maintaining coherency with the neighboring samples [GLM14]. Most non-parametric synthesis techniques rely on MRFs. But instead of running a complex probabilistic inference on the graphical model of the MRF, Efros and Leung [EL99] proposed a simpler and faster approximate solution. They rely on two assumptions, i.e. the color of the sample depends only on its neighborhood and not on the whole image and second that the dependency is independent of the pixel location (cf. Sec. 2.1.1). Then, the missing samples are obtained by sampling and copying the central pixel of a patch from the sample texture that best matches the known neighborhood of the input samples to be synthesized, according to a certain distance [EL99]. The most widely used distance metric to search for similar patches is the Sum of Squared Differences (SSD). However, as observed in [BBCS10] SSD introduces some bias towards uniform regions. This means, that SSD favors the copy of pixels from uniform texture regions. Hence, a weighted combination of SSD and the Bhattacharya distance have been proposed in [BBCS10]. The Bhattacharya distance is a statistics-based metrics and measures the similarity of two probability distributions. An example for patch-based texture synthesis is shown in Fig. 2.2.

Similarly, in a pixel-based texture synthesis method [WL00], the output image is generated pixel-per-pixel in a raster scan order choosing at each step a pixel from the known input texture of the neighborhood, which is most similar to the actually considered neighborhood of the sample to be filled. Pixel-based methods perform usually better than PDE based methods. However, they suffer from synthesis error propagation and repetitive patterns, especially in case of stochastic textures. On the other hand, approaches that synthesizes entire patches mostly overcome the drawbacks of pixel-based methods.

The missing regions in an image are often composed of both textures and structures. Criminisi et al. observed in [CPT04] that it is important to consider these two components separately in the filling routine, starting with the structures. They determine the processing order based on a patch priority measure which consists of a data and a confidence term. The data term reflects the presence of structures in the frame, while the confidence term accounts for the amount of known samples in the patch (for more details see Sec. 3.5.1). In [XS10] Xu and Sun propose a sparsity-

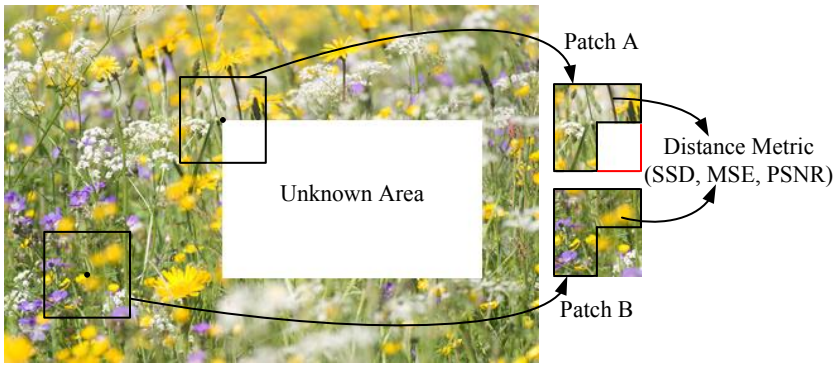


Figure 2.2: The center of the squared window, named patch, is placed at the border of the hole (cf. Patch A). The patch covers both, an unknown and a known sample position and the size of the patch is a free parameter that specifies how stochastic the user believes this texture to be. Next, the available texture is examined. For that, a patch is placed at several/all positions in the original texture (cf. Patch B). Then, the known samples in Patch A are compared to the corresponding positions in Patch B. For this purpose a comparative measure can be used. The unknown areas in Patch A are then copied from the corresponding positions of the continuation patch (cf. Patch B) that minimizes a cost function.

based term to measure the structural confidence. They assume that the structural patches have sparser nonzero similarities with its neighboring patches compared to textural patches. This assumption is derived from the observation that structures (corners, edges) are usually sparsely distributed in an image.

Filling the unknown parts of the input patch can lead to inconsistencies in terms of color and contrast since the new pixels may not fit into the existing texture. In [EF01], Efros and Freeman introduced a quilting method to find an optimal path in the overlapping region of the existing texture and the new samples. An energy function is defined to evaluate the contrast of the current pixel with respect to its neighbors. The best path is then searched using dynamic programming [EF01] or graph cuts [AS07], [KSE+03]. Furthermore, blending methods such as feathering, alpha blending, pyramid blending and cloning methods [Geo04] can also be used to seamlessly merge new patches.

The outcome of the texture synthesis approach can be further improved by introducing a priori knowledge in the patch search routine. In [DDY03], Drori et al. approximate the missing region using some guidance from coarse to fine levels.

2.2 View Synthesis Methods

The goal of view synthesis techniques is to generate virtual views from original camera perspectives. These methods can be used in applications such as 3DV, free viewpoint video, 2-D to 3-D conversion, and virtual reality. Given a set of captured images of a real scene, the synthesis of

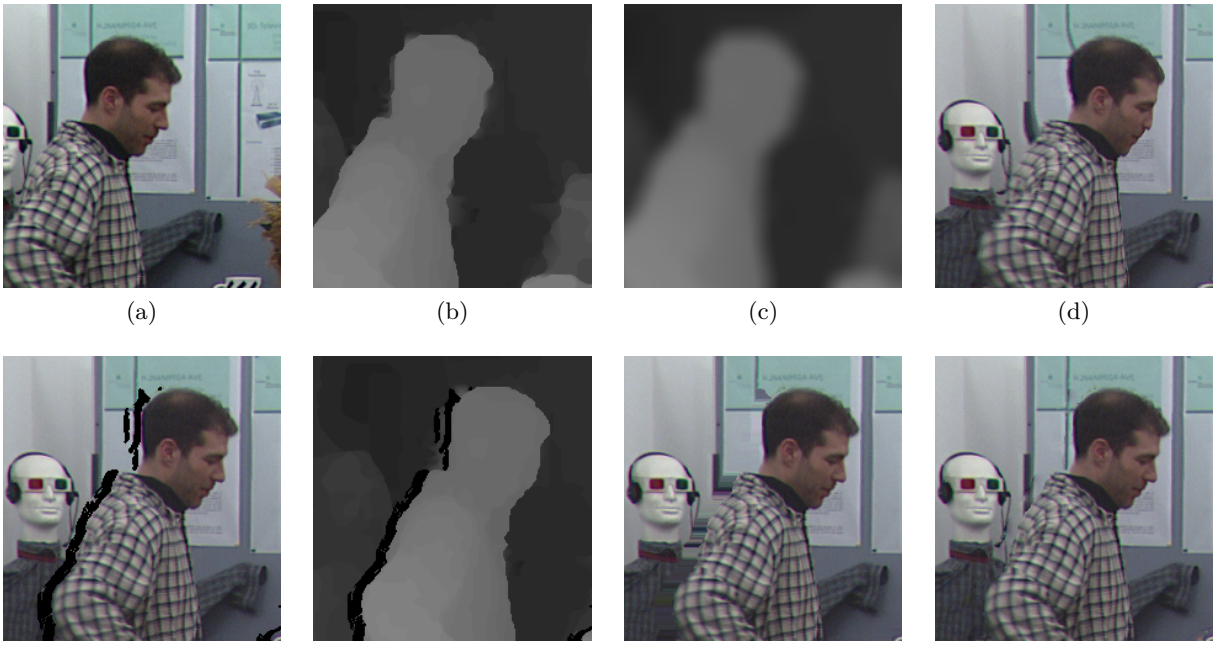
photo-realistic virtual views of the same scene at slightly different viewpoints processed from the original images is also referred to as IBR [BM95]. State-of-the-art 3-D world IBR representation methods can be classified into three categories according to the amount of geometric information used [SK00]: (1) rendering without geometry, (2) rendering with implicit geometry and (3) rendering with explicit geometry. Methods belonging to category (1) utilize several aligned images from different viewing angles in a scene to generate virtual views using ray-space geometry, without requiring geometric information [LH96]. The methods belonging to category (2) rely on implicit geometry. Such implicit geometries are typically expressed in terms of feature correspondences among the known images [CW93]. The methods belonging to category (3) utilize explicit geometry information. Such information is often available in form of depth maps or 3-D geometry [DSF+12]. Methods of category (3) usually offer the highest flexibility in view synthesis, as they allow to compute almost any virtual view independently of camera position and camera angle. If depth information is used as explicit 3-D geometry, the methods of category (3) are also called DIBR. The methods proposed in this thesis (cf. Ch. 3-5) utilize dense depth information as explicit 3-D geometry and thus belong to the DIBR methods. The depth information of the scene is usually stored as inverted real world depth data in a depth map. The depth map is represented as an 8-bit gray-scale image with values between 0 and 255 [MMW10, SSN07, SS02, Bha12]. The advantage of this storage method is that nearby objects achieve a high resolution, while objects that are farther away only receive a coarse depth resolution. This directly corresponds to the perception of the human visual system [Whe38].

A fundamental problem in the DIBR concept is the fact that not every sample in the virtual view necessarily exists in the original textured image. Therefore, unknown image regions become uncovered in the virtual view, especially in the extrapolation scenarios. But due to the enhancement of the depth experience through extrapolation in 3DV, experiments for exploring the extrapolation capabilities of DIBR algorithms have been carried out [MJ11, Vid10] by MPEG.

Several methods have been proposed in the literature in order to address the disocclusion problem. They can be classified into three main categories: (1) depth map pre-processing [DSF+12, ZT05], (2) image domain warping [FWL+11], and (3) disocclusion/hole filling [MFY+08, MFY+09, TFS08, MSD+08].

Methods belonging to the first category pre-process the depth maps in a way that no disocclusions occur. Usually, the depth map is smoothed, using a symmetric [Feh04] or asymmetric filter [ZT05], to reduce depth gradients in the depth map. These methods show good results when small baselines need to be compensated. On the other hand, geometrical distortions can be observed in both foreground and background texture regions. In order to minimize filter-induced distortions, new adaptive filter methods have been proposed [DSF+12, LH09, KMNN13, KMMNN13]. These filters smooth the depth map only in the vicinity of strong depth gradients that are able to uncover holes in the virtual view. Hence, filter-induced artifacts are diminished.

Methods belonging to the second category utilize image domain warping to overcome the disoc-



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

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

Figure 2.3: Depth image-based rendering results for frame 95 of the "Book Arrival" sequence. A baseline of 130 mm is used. (a) Original view. (b) Corresponding depth map. (d) Virtual camera view computed with (c) the Gaussian filtered depth map. (e) Warped virtual view generated based on the original depth map in (b). (f) Corresponding warped depth map with disocclusions (marked black). (g) Result of line-wise filling approach (see artifacts at the person's back). (h) Result of the proposed view synthesis method for static backgrounds (cf. Ch. 3)

clusion problem. Plath et al. [PKGS13], Farre et al. [FWL+11] and Stefanoski et al. [SWL+13] superimpose and deform a regular structured grid over an image and solve an optimization problem in a way that the holes in a synthesized view are covered. However, these technologies can only be used to compensate small baseline shifts in an extrapolation scenario and they further introduce distortion artifact in the virtual view.

Methods belonging to the third category fill the disocclusions with plausible, known image information. Müller et al. [MSD+08] utilize line-wise filling to synthesize the uncovered areas. For each hole in the image, the background pixels at the hole boundaries are identified and copied line-wise into the unknown area. The drawback is that this filling method can only reconstruct uniform textures and horizontal edges. Ye et al. [YYH+14] propose a low-rank matrix restoration model to inpaint disocclusion regions. Mori et al. [MFY+08, MFY+09, TFS08] and Bang et al. [BKY+11] utilize classical image inpainting methods [BSCB00, Tel03] to cover the unknown areas. Lee et al. [LH11] and Ko et al. [KKY13, KY14] applied weighted average filtering in order to fill the uncovered areas. Do et al. [DBZdWPHN11, DBZdWPHN12] proposed a weighted interpolation which considers the background samples alone. The drawback of simple inpaint-

ing [MFY+08, MFY+09, TFS08], average filtering [KKY13, KY14], and interpolation methods [DBZdWPHN11, DBZdWPHN12] is that the synthesized textures are blurred. However, these algorithms are run-time efficient. Wenxiu et al. [WAX+14] consider the filling of the uncovered areas as an optimization problem and use a Gauss-Seidel-based iterative approach to minimize an energy function. Hsu et al. [HAXH14] proposed a method that enforces the spatial and temporal consistency in the disocclusion regions by formulating the hole filling task as an energy minimization problem in a MRF framework. However, both approaches [HAXH14, WAX+14] are very complex and can just be utilized for holes within the virtual view and not for the large out-of-region area at the border of the virtual frame.

Ahn et al. [AK13], Ma et al. [MDdWPHN12], Daribo et al. [DS11] and Xi et al. [XWY+13] utilize patch-based texture synthesis [CPT04, KDNN09, DKNNW10, NNKDW08, GLM14] to fill the unknown areas in the virtual view. However, appropriate pre- and post-processing steps are not applied [AK13, MDdWPHN12, XWY+13, DS11]. Hence, these filling approaches can lead to garbage-growing and blocking artifacts. Nevertheless, patch-based texture synthesis methods usually result in visually pleasing synthesis outcome but are rather complex and time-consuming.

Another problem of hole filling methods for DIBR is to maintain the temporal consistency in the virtual views, particularly in the uncovered areas. In extrapolation scenarios large portions of the texture may become uncovered and the temporal consistency in the computed region is important for a visually pleasing outcome. Therefore, methods that tackle this issue are presented in the following. Yao et al. [YTZ+14] first compute a stable background image using the Gaussian Mixture Model (GMM). This background image is then used to fill the disocclusions. Areas that cannot be covered from the background image are filled using a regular texture synthesis method [CPT04]. However, Yao et al. [YTZ+14] report, that the GMM introduces blur into the background. Xi et al. [XWY+13] and Schmeing et al. [SJ10] use a mosaic/sprite to store background information from neighboring frames for further reuse during the filling process. The drawback of the methods proposed in [YTZ+14, SJ10, XWY+13] is that they are restricted to sequences with static background. Furthermore, the method proposed by [SJ10] et al. requires manual disocclusion correction. Chen et al. [CTL+10] assume that the original views are encoded with H.264/AVC and use the motion vectors from the bit stream to find appropriate information in temporally shifted frames. However, the motion vectors in H.264/AVC are sparse and encoder optimized. This can yield motion vectors that are different from the real motion. Hence, only small objective and subjective gains are reported in [CTL+10]. Stefanoski et al. [SWL+13] apply a temporal smoothness constraint in their image-domain warping approach to minimize temporal artifacts. Hsu et al. [HAXH14] use a temporal term in their energy function while they solve a minimization problem in a MRF. Ko et al. [KKY13, KY14] consider the image information of a previous frame in their weighted average algorithm. In such a way, temporal artifacts are reduced but the synthesized textures become blurrier.

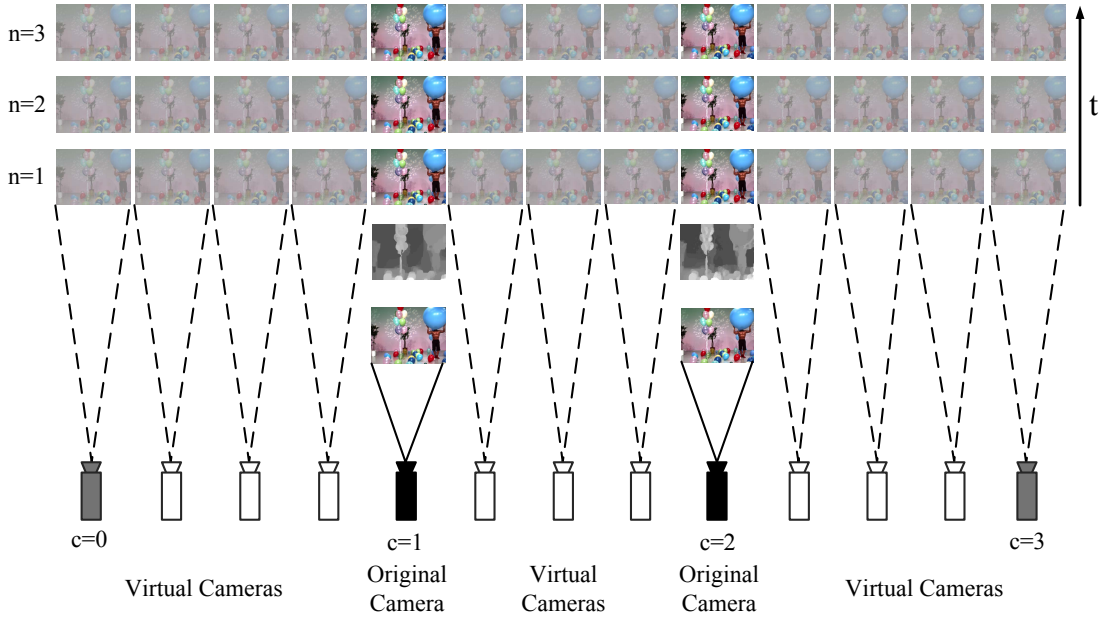


Figure 2.4: Extrapolated and interpolated virtual camera views from two original views.

2.3 General Formulation of the View Synthesis

In this section, the general warping is described for rectified MVD data material as provided by MPEG for 3D video standardization. A detailed description for other setups and arbitrary view synthesis can be found in [AK13, Feh03, HZ03]. The MPEG test data provides several textured views and the associated depth maps showing a scene from slightly different viewing points (cf. Appx. A).

In the following, the image to be filled is denoted as $F_{c,n}$ and the associated depth map as $D_{c,n}$. The subscript c denotes the spatial camera position (cf. Fig. 2.4), with $c \in \mathbb{R}$ and n denotes the actual frame number of the sequence with $n \in \mathbb{N}$. (c.f. Fig 2.4). Hence, $F_{c,n}$ refers to the nth frame at the cth spatial camera position in a sequence. In the proposed frameworks the two view MVD format is utilized. The outermost virtual left camera position (cf. Fig. 2.4, left grey camera) is set to $c = 0$. The baseline between two adjacent original views corresponds to an increment of c by one. Consequently, c is set to one, two and three for the remaining views, i.e. the original left (cf. Fig. 2.4, left black camera), the original right (cf. Fig. 2.4, right black camera) and the outermost virtual right camera (cf. Fig. 2.4, right grey camera). Holes in a textured image and in a depth map are referred to as Ω and Γ, respectively. The original texture in a frame is referred to as F^o with $F^o \subset F \backslash \Omega$ and the synthesized texture in a frame is referred to as F^s. A pixel position in a textured image or in a depth map is denoted as (x, y).

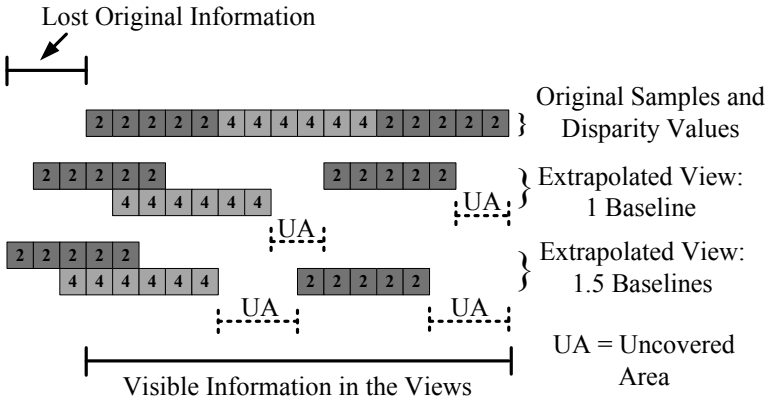


Figure 2.5: Disparity shifts for the extrapolation scenario. A line of pixels in the reference view (top line) is shifted to the virtual views (middle line, bottom line) using the disparity values (2,4). The lower the disparity values, the higher the real distance. © [2016] IEEE.

2.3.1 Rectified Camera Setup

The sample shifts which are necessary for the warping routine can be obtained by computing disparity values *disp* from the inversely quantized depth data stored in D:

$$disp(x,y) = l \cdot b \frac{D(x,y)}{255} \cdot \left(\frac{1}{z_{min}} - \frac{1}{z_{max}} \right) + \frac{1}{z_{max}}, \tag{2.2}$$

where the focal length l and the camera baseline b have to be known. b represents the spatial distance between two original cameras. The variable $disp(x,y)$ is the disparity value which specifies the distances of a sample in the first camera to the same sample in the second camera. For virtual intermediate or extrapolated views the disparity values have to be adapted according to the distance of the virtual view to the original view. z_{min} and z_{max} represent the original minimum and maximum depth values which have to be signaled with the 3DV format. For a half-baseline distance, the disparity values need to be scaled to $0.5 \cdot disp(x,y)$. For rectified camera setups, the vertical position in the virtual view is the same as in the original image.

Considering two original cameras $F_{1,n}$ and $F_{2,n}$ with a baseline distance of one, then the sample position in an intermediate virtual view $F_{c,n}$ at position $c = 1 + k$ and $k \in [0,1]$ are related to the original views as follows [MMW10]:

$$F_{c,n}(x + k \cdot disp(x,y), y) = (1-k) \cdot F_{1,n}(x,y) + k \cdot F_{2,n}(x + disp(x,y), y_2), \tag{2.3}$$

where the variable $disp(x,y)$ describes the disparity value from a sample in the left camera (x,y) to the same sample in the right original (x_r, y_r) in a way that $x + disp(x,y) = x_r$.

Considering the extrapolation scenario, where the virtual view is located outside the range of the two original views than the virtual views $F_{0,n}$ and $F_{3,n}$ (cf. Fig. 2.4, grey cameras) can be

related to the original cameras as follows (cf. Fig. 2.5):

$$F_{0,n}(x - k \cdot disp(x, y), y) = F_{1,n}(x, y), \tag{2.4}$$

$$F_{3,n}(x + k \cdot disp(x, y), y) = F_{2,n}(x, y). \tag{2.5}$$

By choosing a value for the parameter k, the distance between the original and the virtual camera is selected. Usually, k is set to the viewing distance between original views, so that $k = 1$ represents the original camera baseline (c.f. Fig. 2.5). Accordingly, for classical view interpolation between original views, k ranges between 0 and 1. In an extrapolation scenario, the parameter k can receive higher values than one or even negative values. For example, if $k = 1.5$, the distance is set to one and a half baseline (cf. Fig. 2.5).

3 View Synthesis Method for Sequences with Static Background

In this chapter, a new view synthesis method is described. This method is designed to synthesize temporally consistent virtual views from sequences with a static background and their associated depth maps.

The proposed view synthesis method for sequences with static background [NNKD+11] is outlined in Sec. 5.1. The new depth map filling is explained in Sec. 3.2. The novel sprite and image updating routine is presented in Sec. 3.3. In Sec. 3.4 and 3.5, the new initialization and texture synthesis approaches are outlined. In Sec. 6, experimental results are shown.

3.1 Proposed View Synthesis Framework for Sequences with Static Background

The proposed framework for virtual view generation with time-consistent texture synthesis is outlined in Fig. 3.1. The textured images and the associated depth maps of a MVD sequence are taken as input. Depth maps are provided with the test data. Next, the original views are warped towards the virtual positions using the information in the depth maps. For this, an algorithm similar to [TFS08] is utilized. The warped image shows holes in the disoccluded background areas [cf. in Fig. 2.3 (e)]. In addition, the depth maps are also projected [cf. Fig. 2.3 (f)] for a foreground-background separation in the texture synthesis stage. According to the original scene capturing setup, background motion in all views may occur. These cases are tackled in Ch. 4. However, this framework is focused on sequences with static backgrounds similarly to Schmeing and Jiang [SJ10]. Note, that the proposed algorithm is fully automatic, i.e. in comparison to [SJ10] no manual disocclusion correction is required and illumination changes can be seamlessly compensated. The goal of the new view synthesis algorithm is to fill the disocclusions (holes) resulting from the warping process. They become visible in both the virtual depth map and the textured image and must be filled in a visually plausible manner. For video sequences this includes a temporally stable synthesis process, i.e. information from temporally neighboring frames should be taken into account. For minimizing the processing delay, only causal neighbors are considered in this chapter. Temporal consistency is achieved with a background sprite, which stores background information from processed frames. In a first step, the disoccluded areas in the depth map are filled with a new method as shown in Sec. 3.2. The background sprite is

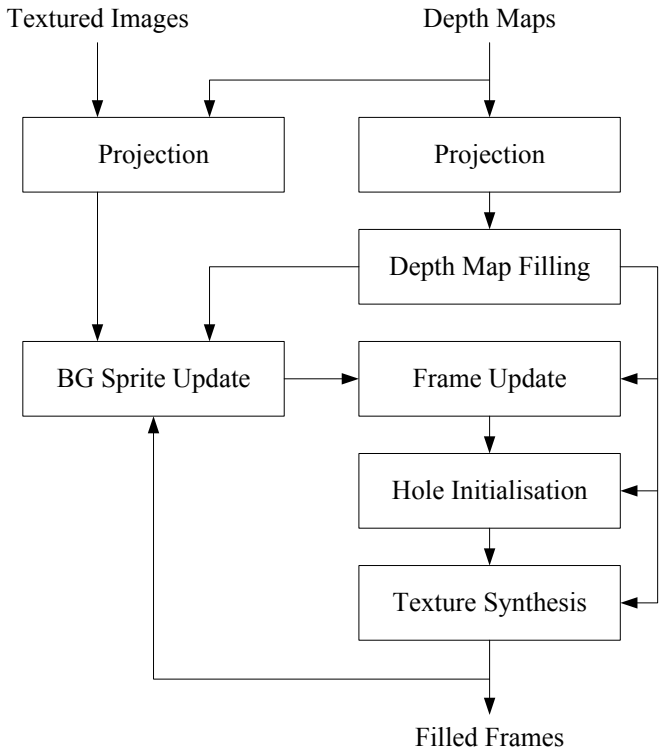


Figure 3.1: Block diagram of the proposed view synthesis framework for sequences with static backgrounds.

then updated with known background information from the current picture. Next, the holes in the current picture are updated from the background sprite (cf. Sec. 3.3). The remaining holes are first initializing from spatially adjacent original texture, providing an estimate of the missing information (cf. Sec. 3.4). In the next step, patch-based texture synthesis is used to refine the initialized areas (cf. Sec. 3.5). The background sprite is finally updated with the synthesized image information for temporal consistency during the filling of holes in the subsequent pictures.

3.2 Filling Disocclusions in the Depth Map

Given the properties of the depth-based image warping, larger uncovered areas mostly belong to background objects. The depth map is represented as an 8 bit gray scale image, denoted as D in the following (cf. Sec. 2). The continuous depth range of the scene is quantitized to the discrete depth values, assigning the value 255 to the point that is closest to the camera and 0 to the most distant point. In Fig. 3.2 (a) the uncovered area in the depth map is denoted as Γ and the corresponding boundary is denoted as $\delta\Gamma$. $\delta\Gamma$ corresponds to the outer boundary of Γ and consists of known background depth values. Due to inaccuracies in depth estimation, foreground object boundary samples may be warped into Γ [denoted as "blobs" in the following, Fig. 3.2 (a)]. One possibility to proceed is to fill the last known background depth value $D(x, y)$, with

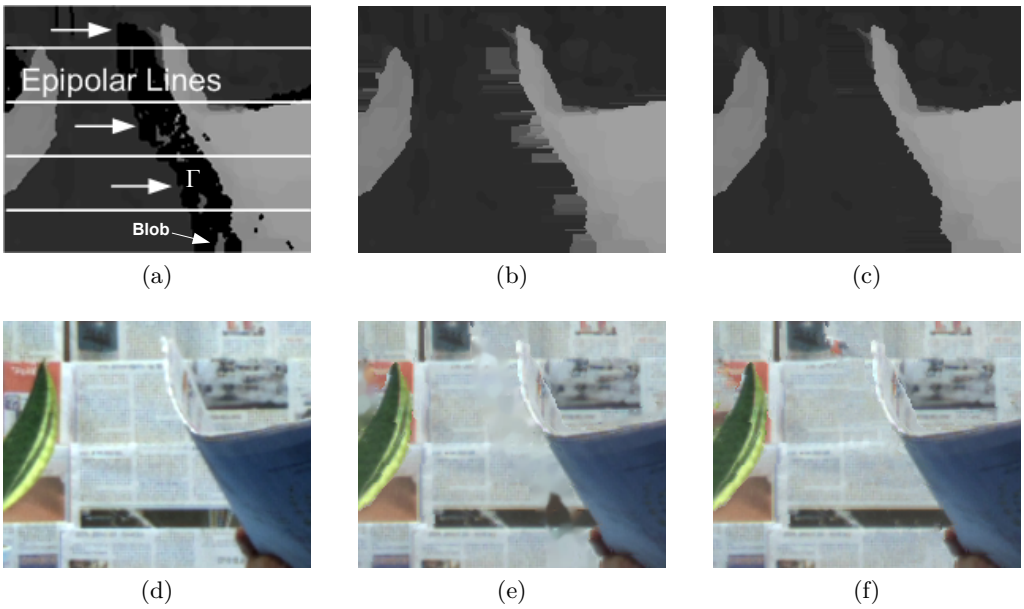


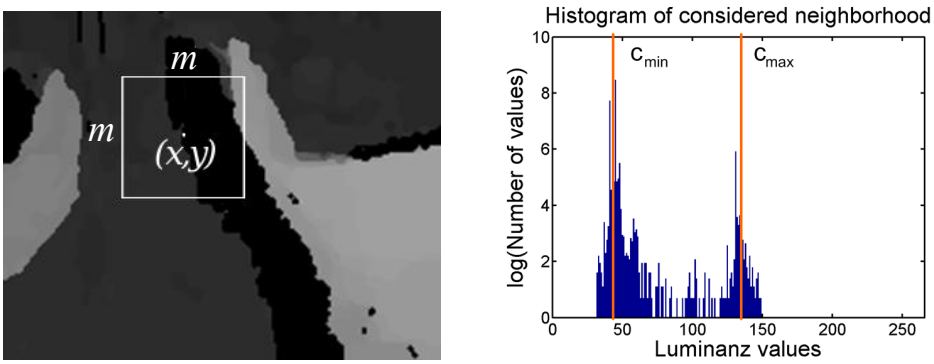
Figure 3.2: Results for picture 1 of the "Newspaper" sequence for the proposed depth map and texture filling approach. (a) Depth map with disoccluded area marked black (filling direction given by white arrows). (b) Line-wise filling of depth map without blob removal. (c) Result of proposed depth map filling approach. (d) Original reference image. (e) Result of MPEG VSRS. (f) Result of the proposed approach.

$(x, y) \in \delta\Gamma$, line-wise into $\delta\Gamma$, as proposed in [MSD+08] [cf. Fig. 3.2 (c)].

In this work a different approach is proposed. First, small blobs in Γ are assigned to Γ, as they are assumed to correspond to noise and may otherwise lead to noticeable inaccuracies in the filled depth map [cf. Fig. 3.2 (b) and (c)]. Subsequently, a verified $D(x, y)$ value is copied line-wise into Γ. It is assumed that relying on a single value of $D(x, y)$ can be error-prone. Hence, the spatial neighborhood surrounding location (x, y) is clustered into two depth classes, whose centroids are represented by $c_{min}^{(x,y)}$ and $c_{max}^{(x,y)}$. They represent foreground and background depth values respectively (cf. Fig. 3.3). The neighborhood is given by a squared area of $m \times m$ samples ($m \in \mathbb{N}$) and centered at location (x, y). $c_{min}^{(x,y)}$ and $c_{max}^{(x,y)}$ are computed via k-means clustering [Bis06] determining two different clusters. After $c_{min}^{(x,y)}$ and $c_{max}^{(x,y)}$ are estimated, the depth information at locations $(u, v) \in \Gamma$ are extrapolated along the row (i.e. $y = v$). The selection criterion for the depth values to be filled at locations (u, v) is defined as follows:

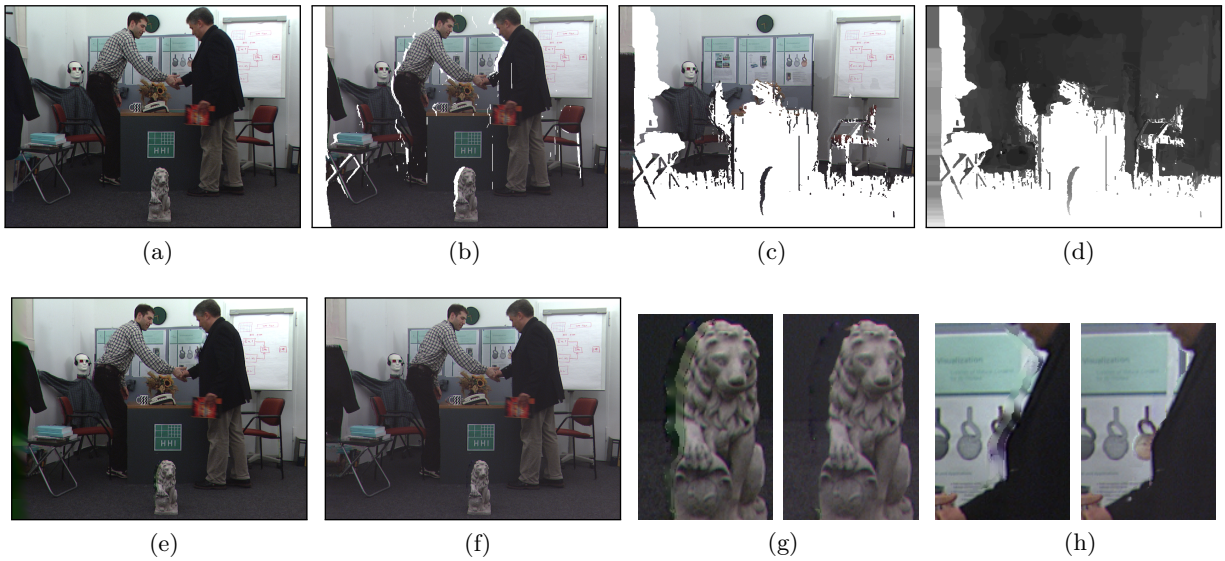
$$D(u,v) = \begin{cases} D(x,y), & \text{if } D(x,y) \leqslant c_{min}^{(x,y)} \\ c_{min}^{(x,y)}, & \text{otherwise} \end{cases}, \quad (u,v) \in \Gamma \wedge v = y, \ (x,y) \in \delta\Gamma, \tag{3.1}$$

where y and v correspond to the row coordinates of locations (u, v) and (x, y) respectively. Background-foreground clustering and subsequent line-wise filling is done for all $(x, y) \in \delta\Gamma$. By



(a) (b)

Figure 3.3: (a) Depth map with highlighted neighborhood (square) centered at (x, y). (b) Histogram of considered neighborhood with the two centroids $c_{min}^{(x,y)}$ and $c_{max}^{(x,y)}$, clustered via k-means. Please note that hole samples [black in (a)] are not considered in the histogram.

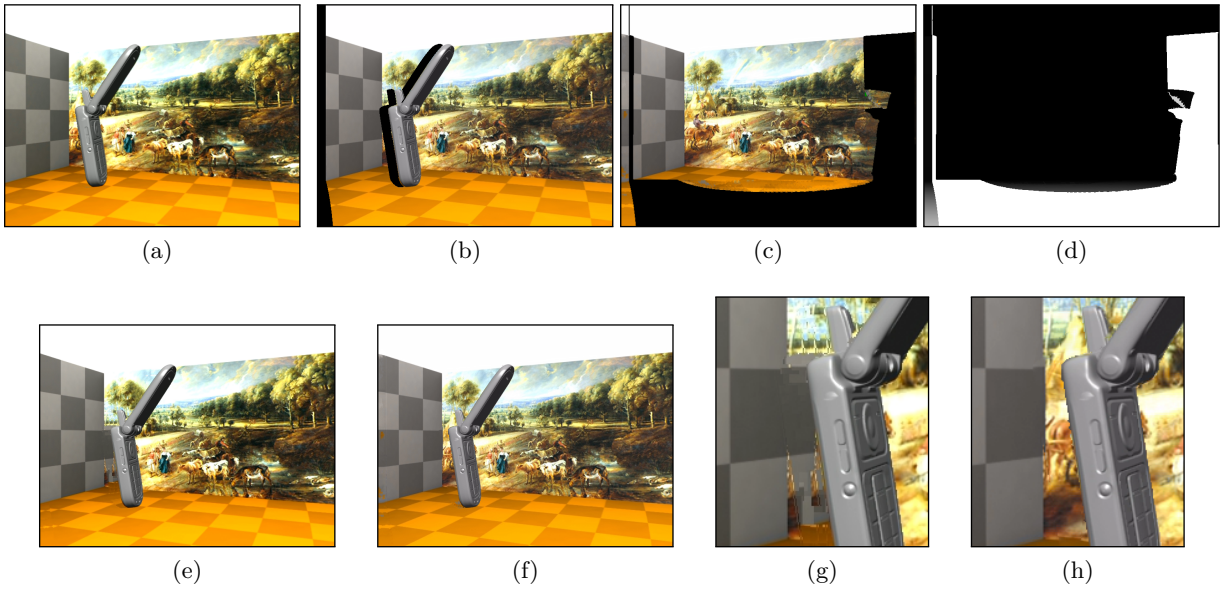


(a) (b) (c) (d)

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

Figure 3.4: DIBR results for the "Book Arrival" sequence. (a) Original reference image 52. (b) Rendered image with disoccluded area marked white. (c) Final background sprite with unknown areas marked white and its associated depth map (d). (e) Result of VSRS_alpha_ETRI [BKY+11]. (f) Result of the proposed approach. (g) and (h) Magnified results. *Left*, VSRS_alpha_ETRI [BKY+11] and *right*, the proposed approach.

using the proposed depth map filling method, the robustness to artifacts in depth map filling is increased. The computed $\left\{c_{min}^{(x,y)}\right\}$ values are stored in order to be used for image and sprite updating as explained in the next section.



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

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

Figure 3.5: DIBR results for the "Mobile" sequence. (a) Original reference image 185. (b) Rendered image with disoccluded area marked black. (c) Final background sprite with unknown area marked black and its associated depth map (unknown area marked white) (d). (e) Result of Ahn et al. [AK13]. (e) Result of the proposed approach. (g) and (h) magnified results. (g) Ahn et al. [AK13] and (h) proposed approach.

3.3 Sprite and Image Updating

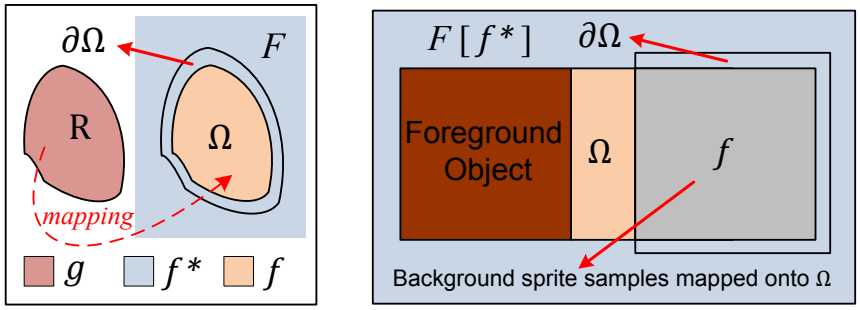
The background image information and its associated depth values are stored in a background sprite, denoted as S [cf. Fig. 3.4 (c) and Fig. 3.5 (c)] and a depth map sprite, denoted as G [cf. Fig. 3.4 (d) and Fig. 3.5 (d)]. These sprites accumulate valuable information for rendering textured images. In fact, by referencing the sprite samples for filling unknown area in the current picture, the synthesis is temporally stabilized.

3.3.1 Sprite Update

For each new picture, denoted as F, the depth values of all sample positions $(m, n) \in D \backslash \Gamma$ are examined to determine the samples that can be considered for the sprite update. For that, the following content-adaptive threshold is computed:

$$\overline{c_{min}} = \begin{cases} c_{min}^{(x,y)}\left(\frac{|\delta\Gamma|+1}{2}\right), & \text{if } |\delta\Gamma| \text{ is odd} \\ \frac{1}{2}\left[c_{min}^{(x,y)}\left(\frac{|\delta\Gamma|}{2}\right) + c_{min}^{(x,y)}(\frac{|\delta\Gamma|}{2}+1)\right], & \text{if } |\delta\Gamma| \text{ is even} \end{cases}, \tag{3.2}$$

where $\overline{c_{min}}$ is the median value of the sorted $c_{min}^{(x,y)}$ values denoted as $c_{min}^{(x,y)}(1)$ ... $c_{min}^{(x,y)}(|\delta\Gamma|)$. Hence, all samples with a depth value below $\overline{c_{min}}$ are eligible for a sprite update.



(a) (b)

Figure 3.6: (a) Seamless cloning principle. (b) Cloning application of the view synthesis framework.

Depth values below $\overline{c_{min}}$ are assumed to describe the background, while the remaining values are assigned to the foreground. Due to the mentioned inaccuracies in the depth estimation step, depth estimates along background-foreground transitions and within the uncovered area in P, denoted as Ω, are considered as being unreliable. Therefore, a two sample wide area around the unreliable regions is not considered for a sprite update. The remaining locations with $D(m,n) < \overline{c_{min}}$ are stored in the background sprite S, and depth map sprite G, respectively. Previously assigned color or depth information is overwritten in S and G. After the synthesis step (cf. Sec. 3.4 and 3.5), novel synthesized textures and depths are incorporated into the sprites as well.

3.3.2 Textured Image Update

The disoccluded regions of every frame F, are updated from the background sprite S. Sample positions corresponding to samples in the background sprite with unknown background information are ignored. The sample positions in S to be used for the update of the current frame F, are selected as follows:

$$F(x,y) = \begin{cases} S(x,y), & \text{if } D(x,y) < G(x,y) + \beta \\ F(x,y), & \text{otherwise} \end{cases}, \forall (x,y) \in \Omega, \tag{3.3}$$

where $F(x,y)$ and $S(x,y)$ represent the intensity value at location (x,y) in the current picture and the background sprite respectively. $D(x,y)$ and $G(x,y)$ represent the depth value at location (x,y) in the extrapolated depth map and the depth map sprite respectively. The parameter β allows some variance in the local background depth value. β is evaluated in Ch. 6. Note that Eq. 3.3 is applied to the chroma channels in the same way.

In order to take illumination variations into account, the covariant cloning method [Geo04, Geo06, NNKDW08] is utilized to fit the background sprite samples to the intensity distribution of the relevant neighborhood of the current picture. The term “cloning” or “seamless cloning” denotes the process of replacing a region of a given picture by another content (often from a

different picture), such that subjective impairments are minimized. In [SYJS05], Poisson cloning is used in texture synthesis to reduce the photometric seams in the gradient domain.

In order to explain the cloning principle, a known scalar function f^* is define over the domain F ($F \in \mathbb{R}^2$). As indicated in Fig. 3.6 (a), $\delta\Omega$ represents the boundary of the unknown area $\Omega \in F$. g is a function defined over the texture source R to be (partially) mapped onto Ω. f is an unknown scalar function defined over Ω. The aim is to find f using the source function g and the information available in $\delta\Omega$. With covariant derivatives, the interpolated version f of f^* can be determined by minimizing the following cost function:

$$\min_f \int\int_\Omega \left(\left(\frac{\partial}{\partial x} + \mathbf{A}_x \right)^2 f^2 + \left(\frac{\partial}{\partial y} + \mathbf{A}_y \right)^2 f^2 \right) dxdy, \tag{3.4}$$

where $\frac{\partial}{\partial x} + \mathbf{A}_x$ and $\frac{\partial}{\partial y} + \mathbf{A}_y$ are called covariant derivatives [Geo06]. $\mathbf{A}_x$ and $\mathbf{A}_y$ represent matrices that model adaptation properties of the human visual system.Solutions of Eq. 3.4 also satisfy the Euler-Lagrange equation:

$$\Delta f = \Delta g \frac{f}{g} \tag{3.5}$$

with the Dirichlet boundary condition:

$$f|_{\delta\Omega} = f^*|_{\delta\Omega} \tag{3.6}$$

where Δ represents the Laplacian operator:

$$\Delta . = \frac{\partial^2 .}{\partial^2 x^2} + \frac{\partial^2 .}{\partial^2 y^2} = 0 \tag{3.7}$$

In this way, information on the boundary $\delta\Omega$ is diffused into Ω, such that the transition between the source function g and F is smooth. Please note that for simplifying the cloning approach, the quotient f/g from Eq. 3.5 is approximated by a constant. The quotient is set to $f/g = 1$, which transforms Eq. 3.5 exactly to the corresponding one in the work by Pérez et al. [PGB03]. The notations of covariant cloning, in the context of the proposed view synthesis framework, are illustrated in Fig. 3.6 (b). It can be seen that boundaries of the area covered by the background sprite, when mapped onto Ω in the current frame F, are either adjacent to the non-reconstructed area Ω or adjacent to the foreground object [not shown in Fig. 3.6 (b)]. In this case the background sprite area is represented by g. The cloned background sprite area then corresponds to f. As can be seen in Fig. 3.6 (b), Ω may remain partially unknown. The current image is denoted as f^* and the boundary $\delta\Omega$ comprises the spatial neighbors of the background sprite samples. Due to the presence of uncovered areas (foreground objects), the boundary conditions Eq. 3.6 for the region Ω are incomplete, i.e. $\delta\Omega$ is undefined at these edges. Therefore the cloning method is adapted to the given view synthesis framework by ensuring that only background samples in the current

picture are considered as valid boundary conditions:

$$\begin{cases} f|_{\delta\Omega} = f^*|_{\delta\Omega} & \text{default} \\ f|_{\delta\Omega} = g|_{\delta\Omega} & \text{if } \delta\Omega \text{ undefined} \end{cases}. \tag{3.8}$$

This modified boundary condition implies that the color information is only diffused into the background sprite samples from those boundaries for which $\delta\Omega$ is defined. This diffusion process is also called photometric correction.

The advantages of using information from previous frames is illustrated in Fig. 3.4 (e)-(h) and Fig. 3.5 (e)-(h). The textures patterns remain their details.

3.4 Initialization of Textured Images

The remaining disocclusions after sprite and image updating are pre-processed with a new texture initialization algorithm. First of all, the Laplacian equation [PGB03] is used to fill small holes in the current image [cf. Fig. 3.7 (a) and (c)]. For the reconstruction of smooth regions this method gives satisfactory results [cf. Fig. 3.7 (b) and (d)]. Good visual results are observed for holes smaller than γ samples (e.g. γ is set to $\gamma = 50$ samples), where Laplace cloning is about 10 times faster than patch-based texture synthesis (cf. Sec. 3.5). Hence, after Laplace cloning, small holes are regarded as finally filled and are not considered in the texture refinement step. For holes larger than γ samples, the visual results of texture synthesis can be improved by using an initial estimate of sample values (cf. Sec. 6.2.2). In this thesis, a new initialization method is proposed that is based on the statistical properties of known samples in the vicinity of Ω. Generally, the known samples constitute valid background samples, but in some cases the depth values at the foreground-background transition are not reliable. Hence, the probability distribution of known background sample values in the spatial neighborhood of the hole area is observed to be skewed. In order to determine the background value from spatially adjacent samples, the median estimator is used, which is the standard measure of end value location used in case of skewed distributions. A window of samples sized 32×32 and centered around the sample to be filled is considered. For each unknown sample, a measure ζ_{BG} is set equal to the number of known samples that are classified as background in the current window. The unknown samples are considered in decreasing order of ζ_{BG}. A 2D median filter operates on the background samples in the current window and the filtered output is used to initialize the unknown sample. The filtering operation can be viewed as the process of extracting a valid background value from the spatially neighboring samples. This serves as a coarse estimate that can be used at the texture synthesis stage to recover the details in the unknown region. Using the described initialization scheme, the sensitivity of the patch-based texture synthesis to outliers is fundamentally reduced (cf. Sec. 6.2.2).

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

Figure 3.7: Results for hole filling with Laplacian cloning. (a) (c) disoccluded areas black (a) or white (c). (b) (d) Filled disocclusions. © [2011] IEEE.

3.5 Texture Refinement via Synthesis

In texture synthesis techniques the unknown region is synthesized by copying content from the known parts ($F^o = F - \Omega$) to the missing parts (Ω) of the image. Patch-based texture synthesis is used in this work to refine the initialized areas. The patch filling order criterion introduced by Criminisi et al. [CPT04] is utilized and extended in this work. Hence, the texture synthesis method proposed by Criminisi et al. [CPT04] is reviewed in the next section.

3.5.1 The Texture Synthesis Method of Criminisi, Perez and Toyama

Criminisi et al. [CPT04] suggested to fill the unknown texture according to the content of the image. They noticed that the success of the structure propagation highly depends on the filling order. A patch is symbolized as $\mathbf{\Psi}_{(x,y)}$ centered at (x, y). The patch filling priority is computed for all sample positions in $\delta\Omega$ and defined as the product of two terms:

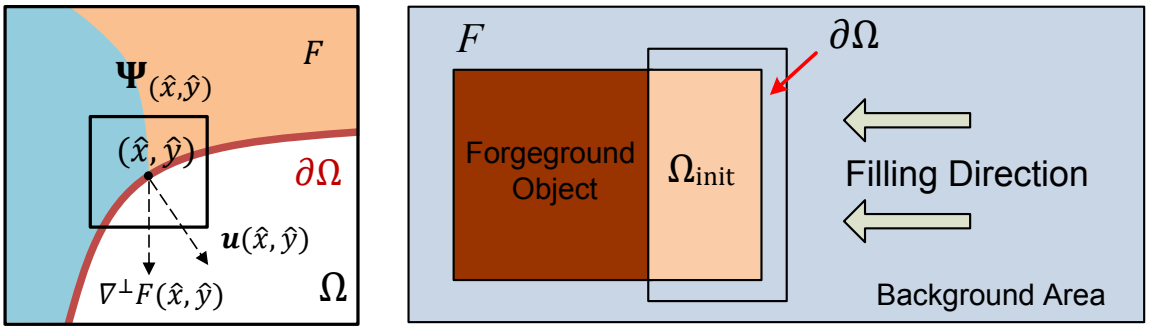
$$P(x, y) = T_{\text{Conf}}(x, y) \cdot T_{\text{Data}}(x, y). \tag{3.9}$$

The priority P of a sample (x, y) is the product of the confidence term $[T_{\text{Conf}}(x, y)]$ and the data term $[T_{\text{Data}}(x, y)]$. The confidence term indicates the reliability of the current patch and enforces a concentric filling order while the data term encourages linear structures to be synthesized first. The terms are computed as follows [cf. Fig. 3.8 (a)]:

$$T_{\text{Conf}}(x, y) = \frac{1}{|\mathbf{\Psi}_{(x,y)}|} \cdot \sum_{(u,v)\in\mathbf{\Psi}_{(x,y)}\cap(F-\Omega)} T_{\text{Conf}}(u, v), \tag{3.10}$$

$$T_{\text{Data}}(x, y) = \frac{\left|\nabla^{\perp} F(x, y) \cdot \mathbf{u}(x, y)\right|}{\eta}, \tag{3.11}$$

where $|\mathbf{\Psi}_{(x,y)}|$ is the area of $\mathbf{\Psi}_{(x,y)}$. η is a normalization factor and set to $\eta = 255$ for an 8-bit image. $\mathbf{u}(x, y)$ is a unit vector orthogonal to $\delta\Omega$ at the sample position (x, y). $\nabla^{\perp} = (-\delta_y, \delta_x)$ is



(a) (b)

Figure 3.8: Texture synthesis. (a) Notation, (b) Hole filling order from background data at the texture refinement step.

the direction of the isophotes [cf. Fig. 3.8 (a)] and derived from the gradients in x and y direction.

The confidence term represents a measure of the amount of reliable pixels in the patch and is initialized as follows: $T_{\text{Conf}}(x,y) = 1, \ \forall (x,y) \in F^o$ and $T_{\text{Conf}}(x,y) = 0, \ \forall (x,y) \in \Omega$. The data term is a function of the strength of the isophothes hitting the boundary of the hole. Hence, broken lines tend to reconnect and thus realizing the "Connectivity Principle" of vision psychology [CS01, Kan79]. Once all the priorities for the sample values in $\delta\Omega$ have been determined, the patch with the highest priority (denoted as $\mathbf{\Psi}_{(\hat{x},\hat{y})}$) is selected for filling. Then, a block-matching algorithm determines the best candidate patch (denoted as $\mathbf{\Psi}_{(\hat{u},\hat{v})}$) from the source region by minimizing a distance measure (SSD) for the known sample positions in $\mathbf{\Psi}_{(\hat{x},\hat{y})}$. After finding the best continuation patch, the unknown sample positions in $\mathbf{\Psi}_{(\hat{x},\hat{y})}$ are filled with the sample values from the corresponding known position in the continuation patch. Finally, the confidence term is updated in the unknown region delimited by $\mathbf{\Psi}_{(\hat{x},\hat{y})}$:

$$T_{\text{Conf}}(x,y) = T_{\text{Conf}}(\hat{x},\hat{y}), \qquad \forall (x,y) \in \mathbf{\Psi}_{(\hat{x},\hat{y})} \cap \Omega. \tag{3.12}$$

The data term is updated by copying the data values from the best candidate patch to the corresponding unknown sample positions in the highest priority patch.

3.5.2 Proposed Texture Synthesis Method

The approach of Criminisi et al. [CPT04] is enhanced in two ways in this work in order to meet the new requirements of the DIBR framework. First, the gradients are calculated for the original as well as the initialized samples (cf. Fig. 3.8 (b), Ω_{init}). This leads to a better isophote direction compared to [CPT04]. Secondly, the filling order is steered in such a way that the synthesis starts from the background area towards the foreground objects. For steering the filling direction, only the border sample positions located in the background are assigned filling priorities according to [CPT04] (cf. Fig. 3.8 (b), $\delta\Omega$ sample positions). When all priorities on $\delta\Omega$ are computed, a block

matching algorithm determines the best exemplar patch $\mathbf{\Psi}_{(\hat{u},\hat{v})}$ to fill the missing samples in the highest priority patch $\mathbf{\Psi}_{(\hat{x},\hat{y})}$. $\mathbf{\Psi}_{(\hat{u},\hat{v})}$ is selected from all possible candidate patches $\mathbf{\Psi}_{(u,v)}$ in a source area. An area A around the patch with the highest priority centered at $(\hat{x},\hat{y})$ is defined to be the source area. In the matching routine only the luminance channel is considered. Given the filled depth map, the depth value of $\mathbf{\Psi}_{(x,y)}$'s center is always known. All sample positions in A with depth values higher than $D_{(x,y)} + \beta$ are excluded from the source area, i.e. they are not considered as center position (u,v) of the candidate patches $\mathbf{\Psi}_{(u,v)}$ Therefore, the patches are selected from corresponding depth regions, excluding foreground objects. To speed-up the matching procedure, the source area is sub-sampled by a factor s. A and s are determined in Sec. 6.2.4. The remaining source positions are used as center positions of the candidate patches $(\mathbf{\Psi}_{(u,v)})$. The best continuation patch out of all candidate patches in the source area is obtained by minimizing the following distance measure:

$$\mathbf{\Psi}_{(\hat{u},\hat{v})} = \underset{\mathbf{\Psi}_{(u,v)} \in F^0}{\arg\min}\, d\left(\mathbf{\Psi}_{(\hat{x},\hat{y})}, \mathbf{\Psi}_{(u,v)}\right). \tag{3.13}$$

Assuming that $\boldsymbol{\psi}^{(u,v)}$ and $\boldsymbol{\psi}^{(\hat{x},\hat{y})}$ are vectorized versions of $\mathbf{\Psi}_{(u,v)}$ and $\mathbf{\Psi}_{(\hat{x},\hat{y})}$, then $d\left(...,...\right)$ is defined as follows:

$$d\left(\mathbf{\Psi}_{(\hat{x},\hat{y})}, \mathbf{\Psi}_{(u,v)}\right) = \sum_{i=1}^{K} \left(\boldsymbol{\psi}_i^{(u,v)} - \boldsymbol{\psi}_i^{(\hat{x},\hat{y})}\right)^2 + \omega_\Omega \sum_{i=1}^{K_{\Omega_{init}}} \left(\boldsymbol{\psi}_i^{(u,v)} - \boldsymbol{\psi}_i^{(\hat{x},\hat{y})}\right)^2, \tag{3.14}$$

where K is the number of original and K_Ω the number of initialized samples in $\boldsymbol{\psi}^{(x,y)}$. ω_Ω is a weighting factor that regularizes the influence of the initialized samples on the overall cost. ω_Ω is determined in Sec. 6.2.2. To ensure smooth transitions between adjacent patches, an efficient post-processing method, based on covariant cloning and similar to the photometric correction method described in Sec. 3.3.2, is utilized. This post-processing approach is adapted to the framework in such a manner that foreground objects are not considered as boundary samples (cf. Sec. 3.3.2). Hereby, the selection of valid boundary samples based on the depth information is novel.

3.6 Chapter Summary and Limitations

In this chapter a new hole filling approach which uses advanced texture synthesis methods for DIBR is proposed. The algorithm works for large baseline extensions and generates spatio-temporally consistent rendering results for 3-D sequences with a static camera setup. Each virtual view image that shows disocclusions is compensated using image information from a causal picture neighborhood via a background sprite. According to the scene capturing setup, background motion may occur. However, this setup cannot be considered since the framework is lacking a motion estimation stage. This issue is tackled in the framework presented in Ch. 4. The

residual uncovered areas that cannot be filled from the sprite are initially coarsely estimated and then refined using a novel texture synthesis method. The visual outcome of the new synthesis method highly depends on the quality of the depth map. Hence, depth estimation inconsistencies especially at foreground-background transitions can lead to obvious degradation of the rendering results.

4 View Synthesis Method for Sequences with Global Background Motion

In the following, an extension of the view synthesis method described in Ch. 3 is described. In comparison to the approach outlined in Ch. 3, the method proposed in this chapter can also be applied for sequences with global background motion.

The proposed view synthesis method for sequences with global background motion [KWD+12a] is outlined in Sec. 4.1. In Sec. 4.2 the novel warping routine is presented. The new method to compensate global background motion is explained in Sec. 4.4 and 4.5. Experimental results are shown in Sec. 6.

4.1 Proposed View Synthesis Framework for Sequences with Global Background Motion

The proposed framework for DIBR with spatially and temporally consistent texture synthesis is outlined in Fig. 4.1. The textured images and the associated depth maps are taken as input. Then a new warping routine is applied which enhances both the spatial consistency and the run-time efficiency. First, the original views and the associated depth maps are warped to the outermost left and right positions beyond the original camera range. For this, an algorithm similar to [TFS08] is used [cf. Fig. 4.2 (a)]. When the outermost views are synthesized, the remaining virtual view in-between the available original and virtual camera positions, i.e. $c = 0$ (virtual view), $c = 1$ (original view), $c = 2$ (original view), $c = 3$ (virtual view), are interpolated by applying an extended version of the warping method proposed in [TFS08]. Hence, only the outermost views need to be synthesized with a complex approach and the remaining virtual positions beyond the original camera range can be interpolated using simple methods, since disocclusions are typically reflected by small holes.

The goal of the new synthesis algorithm applied to the outermost views is to synthesize the uncovered areas in a visually pleasing manner. In a first synthesis step, the virtual depth maps are filled using the method proposed in Sec. 3.2. The filled depth information is used during image registration, frame update and texture synthesis. Then, the uncovered areas in the textured frames are tackled. In order to maintain the temporal consistency the texture information of neighboring frames at the same spatial position is referenced. Since global background motion may be present, the available information is registered to the coordinate system of the current frame

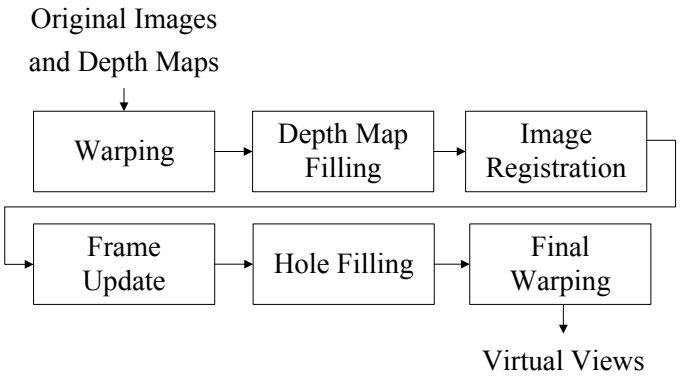


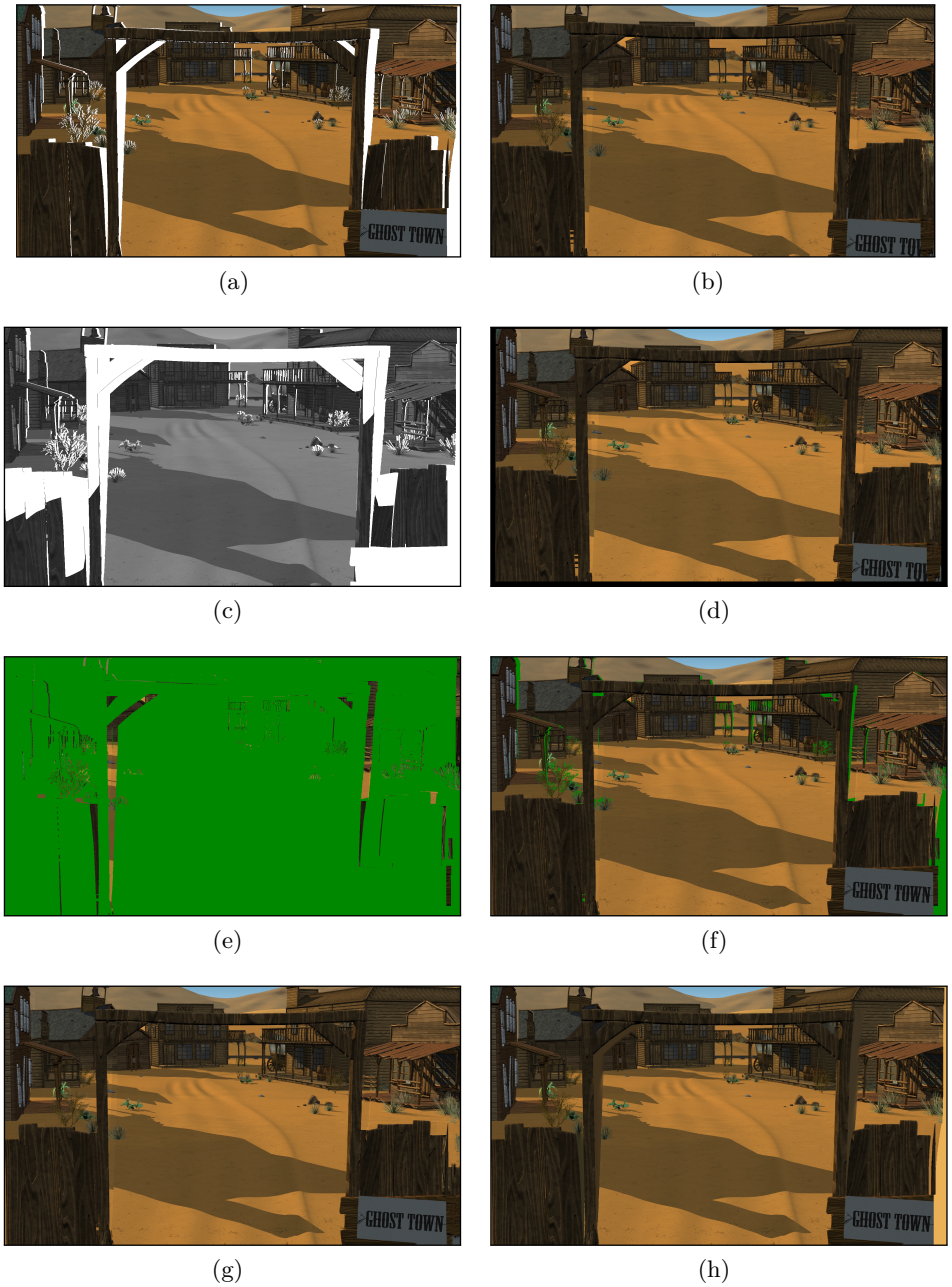
Figure 4.1: Block diagram of the proposed view synthesis framework for sequences with global background motion.

to be processed using a general-purpose registration method [PF06] [cf. Fig. 4.2 (d)]. When the neighboring frames are registered to the new coordinate system, the holes in the current textured frame are updated with reliable image information from the registered background textures [cf. Fig. 4.2 (e)]. Hence, the temporal consistency in the unknown areas is improved, compared to frame-wise filling. The incorporation of image registration into a view synthesis application to compensate global motion is a new concept. Furthermore, a new hierarchical pattern is introduced to determine both the processing order and the background texture source images registered to the current coordinate system of the frame to be processed.

Since, the left and right outermost views are synthesized accordingly, the spatial camera position of $F_{c,n}$ (cf. Sec. 2.3) can be set to $c = 0$ or $c = 3$ during the descriptions in Sec. 4.4 and 4.5. Furthermore, the original texture in a frame is referred to as F^o with $F^o \subset F \backslash \Omega$ and the synthesized texture in a frame is referred to as F^s. $F_{c,np}$ and $F_{c,nf}$ denote a previous and a subsequent frame (in processing order) of $F_{c,n}$ with its associated depth maps $D_{c,np}$ and $D_{c,nf}$, respectively (cf. Fig. 4.5). Note that $F_{c,np}$ and $F_{c,nf}$ are not necessarily adjacent to $F_{c,n}$.

4.2 Warping Routine

In Fig. 4.3 the new warping routine, subdivided into initial and final warping, is outlined. Initially, the original views and the associated depth maps are warped to the outermost left and right camera positions (Fig. 4.3, gray cameras). Then, the disocclusions in these views are individually filled applying the novel method presented in Sec. 4.4.2-4.5. The remaining virtual position (cf. Fig. 4.3, white cameras) are interpolated during a final warping step from the available synthesized (Fig. 4.3, gray cameras) and original cameras (cf. Fig. 4.3, black cameras). For that reason, the work of Tanimoto et al. [TFS08] is extended in a way that it takes original (cf. Fig. 4.3, black cameras) as well as virtual views (cf. Fig. 4.3, gray cameras) and the associated depth maps as input. Employing the new warping routine improves the spatial consistency among



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

Figure 4.2: Frame 153 of the “GT-Fly" sequence, virtual view 1 rendered from camera 5. (a) Current frame 153 to be processed, the disocclusions are marked white. (b) Filled source frame 154. (c) Frame information that is used to register frame 154 to frame 153. Unused foreground regions are marked white. (d) Registered result of frame 154 with a PSNR value of 23.42 dB. (e) Information from frame 154 used to update 153 (green area is not used). (f) Frame 153 filled from the registered source frame. Remaining disocclusions are marked green. (g) Final synthesized result of frame 153. (h) Result of frame 153 rendered with VSRS-Alpha-Gist. © [2012] IEEE.

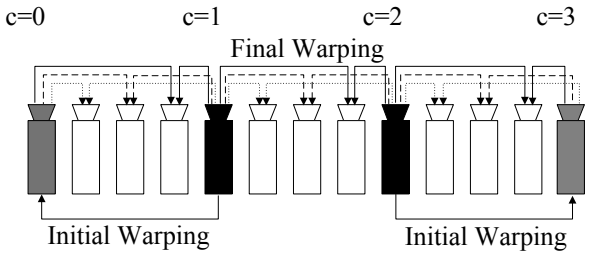


Figure 4.3: Illustration of initial and final warping.

adjacent views, especially in the synthesized areas. Due to the fact that only the outermost virtual cameras (cf. Fig. 4.3, gray cameras) need to be rendered with a complex approach to achieve visually pleasing results, the overall complexity decreases. Note that the complexity stems from the fact that time-consuming patch-based texture synthesis operations are used for rendering of the outermost views (cf. Sec. 4.5).

In Fig. 4.3 three intermediate cameras (cf. Fig. 4.3, white cameras) are interpolated. Nevertheless, depending on the application scenario, the number of interpolated views can be adjusted.

4.3 Depth Map Filling

The uncovered areas in the outermost virtual depth maps are filled using the method proposed in Sec. 3.2. During the depth map filling procedure, an adaptive threshold ($\overline{c_{min}}$) is automatically computed. This threshold constitutes valuable information used to separate foreground and background regions in the image, based on local depths (cf. Sec. 3.2).

4.4 Image Registration Pattern

The outermost views are utilized as anchor cameras for the computation of the remaining extrapolated virtual views (cf. Sec. 4.2). They are thus tremendously important for the overall visual outcome. Therefore, the outermost views are processed with powerful but complex methods. During the synthesis step, image registration is used to compensate global camera motion. The incorporated image registration tool is outlined in the following section.

4.4.1 Image Registration with Partial Data

Image registration is a process of transforming different sets of data onto one common coordinate system through geometrical mapping. The registration method used in this thesis is a general-purpose registration, which can be applied in a broad application field [PF06]. The method works intensity-based, which avoids the problem of choosing a proper feature descriptor and yields a consistent transformation across the entire image. A local affine transformation with a global

smoothness constrain is used to model the transformation between source and target images. Furthermore, this registration framework allows content with missing or partial data.

Denote $h(x, y, t)$ and $h(\hat{x}, \hat{y}, t-1)$ as the source and the target image with a temporal parameter t. The basic local affine transformation is then modeled as:

$$h(x, y, t) = h(\varphi_1 x + \varphi_2 y + \varphi_5, \varphi_3 x + \varphi_4 y + \varphi_6, t - 1), \tag{4.1}$$

where φ_1, φ_2, φ_3, φ_4 are the linear affine parameter and φ_5 and φ_6 are the translation parameters. In order to estimate $\varphi_1, ..., \varphi_6$, a quadratic energy function is minimized:

$$E_b(\boldsymbol{\varphi}) = \sum_{x,y \in \iota} [h(x, y, t) - h(\varphi_1 x + \varphi_2 y + \varphi_5, \varphi_3 x + \varphi_4 y + \varphi_6, t - 1)]^2, \tag{4.2}$$

where $\boldsymbol{\varphi} = (\varphi_1, ..., \varphi_6)^T$ and ι denotes a small spatial neighborhood.

Inherent to the model shown in Eq. 4.1 is the assumption that the sample intensities between the source and the target image are similar. In order to take intensity variations into account an explicit change of local contrast and brightness is incorporated into the original model (cf. Eq. 4.1):

$$\varphi_7 h(x, y, t) + \varphi_8 = h(\varphi_1 x + \varphi_2 y + \varphi_5, \varphi_3 x + \varphi_4 y + \varphi_6, t - 1), \tag{4.3}$$

where φ_7 and φ_8 are new parameters that model the change of contrast and brightness. The vector $\boldsymbol{\varphi}$ is modified to $\boldsymbol{\varphi} = (\varphi_1, ..., \varphi_8)^T$.

Until now it is assumed, that the parameter are constant within a small spatial neighborhood. However, there is a natural trade-off in choosing the size of this neighborhood. A solution for this is to replace the assumption of constancy with a smoothness assumption. This means, that the model parameter $\boldsymbol{\varphi}$ vary smoothly across the space. Hence, the smoothness constraint $[E_s(\boldsymbol{\varphi})]$ is integrated in the energy function:

$$E(\boldsymbol{\varphi}) = E_s(\boldsymbol{\varphi}) + E_b(\boldsymbol{\varphi}), \tag{4.4}$$

where the smoothness constraint is defined as follows:

$$E_s(\boldsymbol{\varphi}) = \sum_{i=1}^{8} \lambda_i \left[\left(\frac{\partial \varphi_i}{\partial x} \right)^2 + \left(\frac{\partial \varphi_i}{\partial y} \right)^2 \right]. \tag{4.5}$$

λ_i is a positive constant that controls the relative weight given to the smoothness constraint on parameter φ_i. Subsequently, $\boldsymbol{\varphi}$ can be calculated in an iterative manner. Finally, a local affine and smooth transformation is achieved.

Inherent to the model explained above is the assumption that each region in the source image has a corresponding region in the target image. However, this is not always the case. In the following, the image registration method is extended in order to handle partial or missing data.

First, it is assumed that pixels in the source and target image are either related according to Eq. 4.3, denoted as model H_1, or cannot be explained by this transformation and therefore belong to an outlier model H_2. Since there is no clear classification for each pixel at the beginning, the likelihood function is applied in this model. Assuming that the pixels are spatially independent and identically distributed the likelihood of observing a pair of images is given by:

$$L(\boldsymbol{\varphi}) = P(q(x, y)), \tag{4.6}$$

where, $q(x, y)$ denotes the image intensities of the source $\varphi_7 h(x, y, t) + \varphi_8$ and the target $h(\varphi_1 x + \varphi_2 y + \varphi_5, \varphi_3 x + \varphi_4 y + \varphi_6, t - 1)$. To simplify the optimization of the likelihood function, the log-likelihood function is considered with the priors on the models $P(H_1)$ and $P(H_2)$:

$$\log\left[L(\boldsymbol{\varphi})\right] = \sum_{x,y\in\iota} \log\left[w_r P(q(x, y)|H_1)P(H_1) + P(q(x, y)|H_2)P(H_2)\right], \tag{4.7}$$

where the weight w_r is proportional to the likelihood of pixels which belong to model H_1. Finally, the Expectation-Maximization (EM) algorithm is used to estimate $\boldsymbol{\varphi}$ and proceeds as follows:

1. Initialization: initialize the estimated affine transformation matrix $\boldsymbol{\varphi}$.
2. E-step: compute the weights w_r.
3. M-step: estimate the model parameters $\boldsymbol{\varphi}$.
4. Repeat steps 2 and 3 until the difference between successive estimates of $\boldsymbol{\varphi}$ is below a specified threshold.

The EM algorithm allows a simultaneous segmentation and registration. Therefore, this model still works even if image data is only partially available or missing. A heavy computational load is one of the main issues of area-based approaches. Hence, a differential multi-scale framework is implemented to improve it. Furthermore, a course-to-fine scheme is adopted in order to contend with large motion.

In the method proposed in this chapter, global motion in the background areas should be compensated. An area-based method is appropriate for this application. Another reason for choosing this registration method is its ability of dealing with missing data, since partial information in the images is unavailable after the warping procedure.

4.4.2 Hierarchical Prediction Structures

A pattern derived from the hierarchical prediction structure originally utilized in video coding [SMW06] is used to determine the temporal frame processing order in the method proposed in this chapter. Therefore, the general hierarchical prediction structure is outlined in this section and the novel proposed pattern is explained in Sec. 4.4.3.

A general hierarchical prediction structure has several hierarchical stages depending on the GOP size, i.e. four for a GOP size of eight frames [SMW06] (cf. Fig. 4.4) and three in the proposed framework (cf. Fig. 4.5). In Fig. 4.4, the black frames represent the first, the blue frames the second, the red frames the third, and the green frames the fourth hierarchical stage. In regular intervals, pictures of a sequence are assigned as key pictures (cf. Fig. 4.4, black frames). All pictures that are located in-between the current key frame and the previous key frame are used to build a GOP in [SMW06] (cf. Fig. 4.4). All pictures of a GOP, except the key frame, are computed using the bi-hierarchical syntax and each of them has two reference pictures [SMW06] (cf. Fig. 4.5).

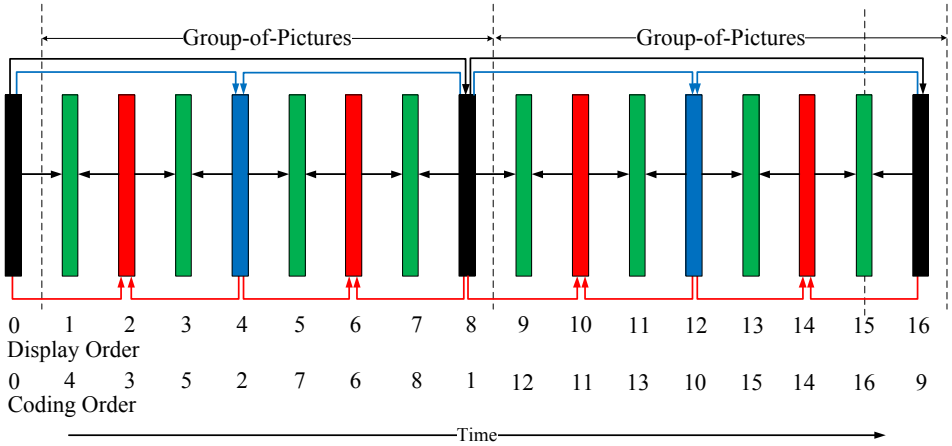


Figure 4.4: Hierarchical coding structure with four temporal levels [SMW06].

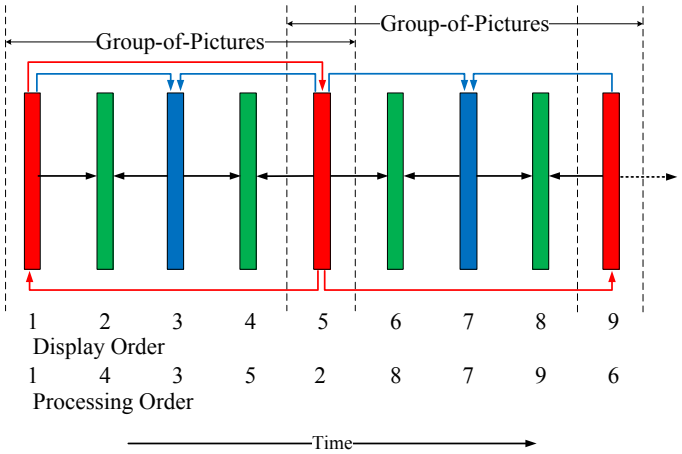


Figure 4.5: Proposed hierarchical pattern used to determined both the temporal processing order and the neighboring frames considered for the texture update routine.

4.4.3 Proposed Image Registration Pattern

In coding scenarios, the key frames have the highest quality of the GOP, i.e. the lower the stage, the lower the reconstruction quality. In the proposed method, this quality separation is not considered.

Here, the hierarchical pattern is applied to determined both the temporal processing order and the neighboring frames considered for the texture update routine (cf. Sec. 4.5). As an example, Fig. 4.5 shows the processing order for the first nine frames of a sequence. In the proposed framework, an overlapping GOP structure is introduced to connect the individual GOPs, leading to global consistency within a virtual sequence. Each GOP structure considers five frames and the associated depth maps loaded and processed as a unit. Images in one GOP are classified into three temporal levels, as shown by three different colors in Fig. 4.5, i.e. the red frames represent the first, the blue frames the second, and the green frames the third hierarchical stage. Images marked in the same color belong to the same temporal level. The frames that point to the current image to be processed (cf. Fig. 4.5) are registered to the actual coordinate system in order to use existing original and/or synthesized textures. The key frame is the first frame of each GOP as well as the last frame in previous GOP. The key frames are processed first as they are directly or indirectly used as references. Furthermore, it is supposed that frames with large temporal distance also show larger spatial differences, i.e. more image information can be used to fill the holes.

According to the size of one GOP, each time a predefined number of frames are loaded and processed. The number of frames in a sequence that can be computed using the GOP structure (N_p) is thus determined as follows:

$$N_p = \begin{cases} \left\lfloor \frac{N}{z-1} \right\rfloor \cdot (z-1)+1, & \text{if } \left\lfloor \frac{N}{z-1} \right\rfloor \cdot (z-1)+1 \leqslant N \\ \left(\left\lfloor \frac{N}{z-1} \right\rfloor - 1 \right) \cdot (z-1)+1, & \text{otherwise} \end{cases}, \; N_p, N \in \mathbb{N}, \tag{4.8}$$

where N is the number of all frames in the sequence. z is the number of frames in a GOP and set to $z = 5$ (cf. Fig. 4.5). Furthermore, it is assumed that $z < N$. The remaining frames (F_{c,N_p+1}, ..., $F_{c,N}$) are filled frame-wise utilizing patch-based texture synthesis (cf. Ch. 3).

In general, the disoccluded areas belong to the background textures (cf. Fig. 1.2). Therefore, the affine transformation matrix is estimated in the background regions of the luminance signal (Y) and then applied to all channels (YUV). Background and foreground textures are separated, examining the depth values in the depth map with the automatically computed threshold $\overline{c_{min}}$ (cf. Sec 3.2)

To decide whether the image registration was successful, the PSNR between $F_{c,n}$ and the registered frame $F_{c,np}$ or $F_{c,nf}$ is determined for the image region used in the registration step [cf. Fig. 4.2 (c)]. If the PSNR measured in the luminance channel lies above a chosen threshold (t_{psnr}), the registered frame is considered for updating frame $F_{c,n}$.

However, in a set of objective evaluations, it was found that at least 80% percent of the image information needs to be present to compute a reliable affine transformation matrix. Thus, if necessary $\overline{c_{min}}$ is refined until 80% percent of the image information is available [cf. Fig. 4.2 (c)]. This means, that in some rare cases image locations from foreground/background transitions or parts of foreground objects are considered during the transformation estimation.

4.5 Frame Update and Texture Filling

The registered frames [cf. Sec. 4.4.3 and Fig. 4.5, Fig. 4.2 (e)] provide valuable information for frame updating. By referencing the registered samples for filling unknown areas in the current picture, the synthesis is temporally stabilized.

The uncovered frames are processed in the temporal order depicted in Fig. 4.2. The pseudo code of this procedure is shown in Algorithm 1. The outcome of the **Update** function, is a synthesized frame. The **Update** function takes the current frame to be processed and one or two registered images, i.e. $F_{c,np}$, $F_{c,nf}$, as input. The pseudo code for the **Update** function is shown in Algorithm 2.

Algorithm 1: Pseudo code for GOP update

```
Data: Frames and associated depth maps.
Result: Synthesized frames.
begin
    for i ← 1 : 4 : N_p do
        if i == 1 then
            F_{c,i} ← Update(F_{c,i}, [], F_{c,i+4}, D_{c,i}, [], D_{c,i+4});
        F_{c,i+4} ← Update(F_{c,i+4}, F_{c,i}, [], D_{c,i+4}, D_{c,i}, []);
        F_{c,i+2} ← Update(F_{c,i+2}, F_{c,i}, F_{c,i+4}, D_{c,i+2}, D_{c,i}, D_{c,i+4});
        F_{c,i+1} ← Update(F_{c,i+1}, F_{c,i}, F_{c,i+2}, D_{c,i+1}, D_{c,i}, D_{c,i+2});
        F_{c,i+3} ← Update(F_{c,i+3}, F_{c,i+2}, F_{c,i+4}, D_{c,i+3}, D_{c,i+2}, D_{c,i+4});
```

The samples from the registered frames corresponding to the hole positions in the actual frame are considered as candidates for the update (cf. Algorithm 2). Original texture is used primarily and synthesized data as fall-back. An available pixel position $F_{c,np}(x,y)$ or $F_{c,nf}(x,y)$ in a registered frame is used to fill $F_{c,n}(x,y)$, $(x,y) \in \Omega$ in $F_{c,n}$, if the associated registered depth value $D_{c,np}(x,y)$ or $D_{c,nf}(x,y)$, is in the depth range of $D_{c,n}(x,y)$ (cf. Algorithm 2). The depth values indicate the relative distance of an object to the camera. Therefore, they can be considered as a valid criterion to decide whether the texture belong to the same depth range. This is formalized as follows:

$$D_{c,n}(x,y) - \beta < D_{c,np/nf}(x,y) < D_{c,n}(x,y) + \beta, \tag{4.9}$$

where β is a parameter to account for small variations in different depth maps (cf. Sec. 3.3.2). If two values $F_{c,np}(x,y)$ and $F_{c,nf}(x,y)$ qualify for updating $F_{c,n}(x,y)$ then the following applies

Algorithm 2: Pseudo code for the **Update** function

Data: $F_{c,n}$, $F_{c,np}$, $F_{c,nf}$, $D_{c,n}$, $D_{c,np}$ and $D_{c,nf}$.
Result: Updated and subsequently synthesized frame $F_{c,n}$.
begin
  for *each pixel position p in* Ω *in* $F_{c,n}$ **do**
    if $F^o_{c,np}(x,y)$ ***and*** $F^o_{c,nf}(x,y)$ *are in depth range (cf. Eq. 4.9)* **then**
    Compute new texture value (cf. Eq. 4.10);
    else if $F^o_{c,np}(x,y)$ ***or*** $F^o_{c,nf}(x,y)$ *is in depth range (cf. Eq. 4.9)* **then**
      Choose texture value in depth range;
    else
      Take no action;
  for *each remaining pixel position p in* Ω *in* $F_{c,n}$ **do**
    if $F^s_{c,np}(x,y)$ ***and*** $F^s_{c,nf}(x,y)$ *are in depth range (cf. Eq. 4.9)* **then**
    Compute new texture value (similar to Eq. 4.10);
    else if $F^s_{c,np}(x,y)$ ***or*** $F^s_{c,nf}(x,y)$ *is in depth range (cf. Eq. 4.9)* **then**
      Choose texture value in depth range;
    else
      Take no action;
  Fill remaining holes in $F_{c,n}$ using texture synthesis.

(cf. Algoritm 2):

$$F_{c,n}(x,y) = \frac{F_{c,np}(x,y) + F_{c,nf}(x,y)}{2}, \quad F_{c,n}(x,y) \in \Omega. \tag{4.10}$$

In order to fill the remaining disocclusions $F_{c,n}$ that can not be copied from $F_{c,np}$ or $F_{c,nf}$, the method proposed in Sec. 3.4-3.5 is applied. First, small blobs are closed with Laplace PDE. Then, the large holes are initialized and subsequently optimized with patch-based texture synthesis [cf. Fig. 4.2 (g)]. To account for intensity variations between adjacent patches, covariant cloning is used.

4.6 Chapter Summary and Limitations

In this chapter a new method to handle the disocclusion problem in DIBR is introduced, especially for the extrapolation scenario. First, a new warping scheme is presented, which is divided in two steps, i.e. initial and final warping. In the initial warping step, the outermost virtual cameras beyond the viewing range of the original cameras are synthesized with a new approach. Then, all remaining views are computed in the final warping step between the original and rendered outermost cameras using a fast state-of-the-art method. Hence, the overall complexity is minimized and spatial consistency between adjacent extrapolated views is maintained, particularly in the synthesized areas.

The new synthesis approach uses image information from previous and subsequent frames to

reduce artifacts in unknown areas. To compensate global motion in a sequence, image registration is incorporated into the framework. Hereby, a novel strategy derived from the general GOP structure is utilized to take the image content from temporally surrounding frames into consideration. The image registration method estimates affine global transformation parameters between the source and the target image. However, if several distinct motions are present in the background of the sequence, the registration tool is unable to represent all of them, leading to falsely estimated transformation parameters.

5 Hybrid View Synthesis Method

In the previous chapters and in several recent publications [AK13, MDdWPHN12, DS11, KNND+10, KWD+12b, KWD+12a, NNKD+10, NNKD+11, XWY+13], it was shown, that patch-based non-parametric texture synthesis is a proper technique to synthesize missing regions in virtual views in a visually plausible manner. However, these methods are very complex and thus time-consuming. In this chapter, a novel hybrid synthesis framework is described that overcomes this complexity issue providing a similar or even better visual outcome. A new patch-based and a novel fast parametric texture synthesis method are combined in an innovative way to compute the uncovered image areas.

The proposed hybrid framework [KMW16] is presented in Sec. 5.1. The pre-processing is explained in Sec. 5.2. The hybrid texture synthesis approach is outlined in Sec. 5.3. Finally, in Sec. 5.4 this chapter this summarized. In Ch. 6, experimental results are shown.

5.1 Proposed Framework

The proposed framework is outlined in Fig. 5.1. The framework takes the original frames and the associated depth maps as input signals (cf. Fig. 1.3). A state-of-the-art warping routine, similar to the method proposed in [TFS08] is utilized to shift the original images, the pre-processed images and the depth maps to the extrapolated virtual camera position (cf. Fig. 1.3). To fill the uncovered areas in the virtual views, a new hybrid texture synthesis method is proposed. The proposed approach combines the advantages of parametric and patch-based texture synthesis. As explained in Sec. 2.1, parametric texture synthesis is faster than patch-based synthesis, but is efficient only for textures with similar appearance and fails in the presence of a heterogeneous texture segment, i.e. when the segment is composed of several texture sources. Hence, a new patch-based texture synthesis is first used at texture boundaries to split the hole into several homogeneous texture segments. Then, a fast parametric approach is applied to synthesize the separated texture regions with an improved likelihood to generate better results. Furthermore, new quality assessment tools are introduced to identify an appropriate training area and to rate the quality of the parametrically synthesized area. To utilize the hybrid texture synthesis method properly, it is important to identify appropriate structures and object boundaries. This is a challenging task, especially when texture patterns appear. Such patterns could be regular, near regular or irregular [XYXJ12] and can be falsely identified as boundaries. In order to improve the detection of the relevant texture transitions, a pre-processing step [XYXJ12] is incorporated into the framework and applied on the

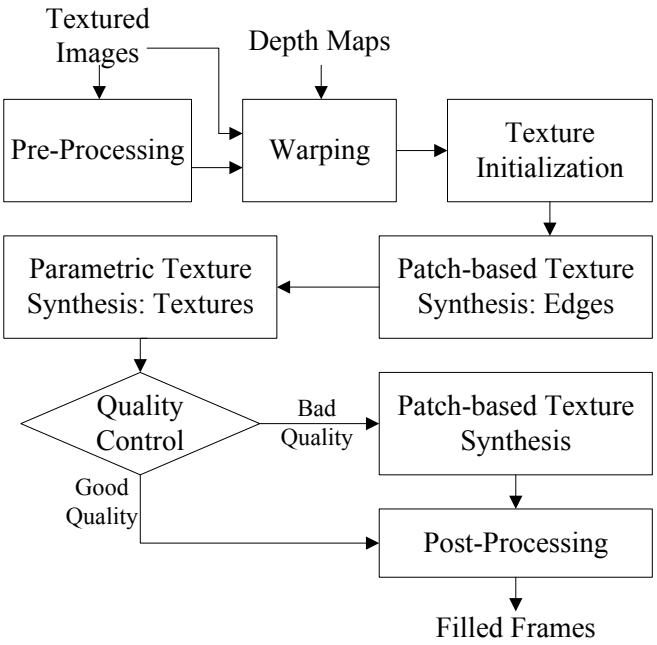


Figure 5.1: Block diagram of the proposed hybrid view synthesis framework. © [2016] IEEE.

textured images prior to the warping routine. The pre-processing step improves the distinction of heterogeneous texture segment tremendously. In detailed experiments (cf. Ch. 6), it is shown that the proposed hybrid view synthesis approach is faster than patched-based state-of-the-art methods and provides similar or even better subjective and objective results.

5.2 Pre-Processing

In the pre-processing step, texture separation is used to separate the structure and object boundaries from texture areas. The aim is to preserve the main borders of objects and edges, while smoothing the remaining parts. This is due to the fact that in the synthesis step the texture transitions are determined based on the isophotes. The isophotes are derived from the gradients in X and Y direction. Hence, a simple low-pass filter can not be applied to meet the requirements. Therefore, an edge preserving filtering method is incorporated. The approach proposed by Xu et al. [XYXJ12] is chosen due to the fact that the authors outperform alternative state-of-the-art methods and provide promising results. However, other edge preserving smooth filters may further improve the performance of the new hybrid method. The filter proposed by Xu et al. [XYXJ12] is applied frame-by-frame. But, in experiments (cf. Sec. 6.4.1) it was figured out that the filter separates the texture classes better in a bi-cubic down-sampled image [cf. Fig. 6.7 (e)]. The proposed pre-processing method is shown in Fig. 5.2 and evaluated in Sec. 6.4.1. The filtered image is denoted as H with its sample locations $\mathbf{p} = (x, y)$. The filtering approach, applied to the down-sampled frames, is presented in the following.

Figure 5.2: Pre-processing method used in the hybrid texture synthesis framework. The input image is bi-cubically, down-sampled to half resolution. Then the filter proposed by Xu et al. is applied [XYXJ12] to the image. The filtered outcome is subsequently bi-cubically up-sampled to full resolution. © [2016] IEEE.

5.2.1 Relative Total Variation Smoothing

Xu et al. proposed a method for the extraction of contours from images by taking the Relative Total Variation (RTV) into account [XYXJ12]. RTV is defined as follows:

$$\frac{\mathscr{D}_x(\mathbf{p})}{\mathscr{L}_x(\mathbf{p}) + \nu} + \frac{\mathscr{D}_y(\mathbf{p})}{\mathscr{L}_y(\mathbf{p}) + \nu}, \tag{5.1}$$

where

$$\mathscr{D}_{(.)(\mathbf{p})} = \sum_{\mathbf{p} \in R(\mathbf{p})} g_{(\mathbf{p},\mathbf{q})} \cdot \|(\partial_{(.)} H_{\mathbf{q}})\| \tag{5.2}$$

is the total variation in a window R centered at $\mathbf{p}$ and

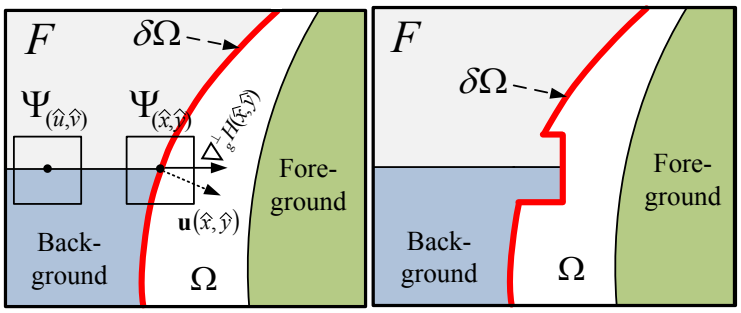
$$\mathscr{L}_{(.)} = \| \sum_{\mathbf{p} \in R(\mathbf{p})} g_{(\mathbf{p},\mathbf{q})} \cdot (\partial_{(.)} H_{\mathbf{q}})\| \tag{5.3}$$

measures the overall spatial variation. $g_{(\mathbf{p},\mathbf{q})}$ is a weighting function defined according to the spatial affinity. ν is a very small value to prevent divisions by zero. The structure layer of the separation is supposed to have a small RTV, leading to the following function:

$$\arg\min_{H} \left\{ \sum_{\mathbf{p}} (H_{\mathbf{p}} - F_{\mathbf{p}})^2 + \lambda \cdot \frac{\mathscr{D}_x(\mathbf{p})}{\mathscr{L}_x(\mathbf{p}) + \nu} + \frac{\mathscr{D}_y(\mathbf{p})}{\mathscr{L}_y(\mathbf{p}) + \nu} \right\}. \tag{5.4}$$

This function is non-convex. Hence, the authors suggested an iterative numerical solution.

In the next step, the pre-processed frame, the original frame and the depth map are warped to the extrapolated virtual position. In the virtual frames, unknown regions become uncovered [cf. Fig. 1.3 and Fig. 6.12 (a)].



(a) (b)

Figure 5.3: Utilized notation. F denotes the frame. The area to be filled is indicated by Ω, and its contour is denoted as $\delta\Omega$. $\nabla^{\perp}H(x,y) = (-\partial_y, \partial_x)$ represents the strength of the isophotes measured in the pre-processed image H. $\boldsymbol{\Psi}_{(\hat{x},\hat{y})}$ is the patch to be filled having the highest priority. $\boldsymbol{\Psi}_{(\hat{u},\hat{v})}$ is the best candidate patch.

5.3 Synthesizing Textures

As mentioned in Sec. 2.2, an appropriate technology for filling missing image regions with known information is texture synthesis [CPT04, NNKDW08]. As proposed in the former chapters, the unknown texture regions in the shifted pre-processed image and in the virtual frame are initialized. For that, the initialization method proposed in Sec. 3.4 is utilized. The virtual depth maps are filled applying the method outlined in Sec. 3.2.

In the following, a detailed description of the new hybrid synthesis approach will be given.

5.3.1 Proposed Dominant Structure Synthesis

In the proposed hybrid approach, a new patch-based texture synthesis method is utilized to synthesize the dominant structures only. The method is a modification of the texture synthesis approach outlined in Sec. 3.5.1. It is assumed that different classes of textures are separated by dominant edges in the image. Hence, the remaining areas, which mostly contain texture patterns, remain unfilled and can be subsequently computed using fast parametric texture synthesis. The filling direction is steered from the background towards the foreground areas (cf. Fig. 5.3). Based on the warping direction, the holes are either on the left (warping to the left) or on the right (warping to the right) side of foreground objects, such that the background area can be determined accordingly (cf. Fig. 1.3). Only background samples are considered as elements of $\delta\Omega$ (cf. Fig. 5.3). In the proposed work, the filling priority (P) of a sample $(x,y) \in \delta\Omega$ is computed with the data term [CPT04] alone. Furthermore, the strength of the isophotes is measured in the pre-processed image. In such a way, texture patterns are not considered as boundaries (cf. Fig.

6.7). Hence, the priority can be computed as follows:

$$P(x,y) = \frac{\left|\nabla_g^{\perp} H(x,y) \cdot \mathbf{u}(x,y)\right|}{\eta}, \tag{5.5}$$

where $\mathbf{u}(x,y)$ is a unit vector orthogonal to $\delta\Omega$ at the sample location (x,y) and $\nabla_g^{\perp} C$ is the strength of the isophotes in the pre-processed image. η is a normalization factor, which is set to 255 for an 8-bit image. In contrast to [CPT04], the isophote strength $\nabla^{\perp} H(x,y) = (-\partial_y, \partial_x)$ [cf. Fig. 5.3 (a)] is computed separately for the RGB channels $\left(\nabla_R^{\perp}, \nabla_G^{\perp}, \nabla_B^{\perp}\right)$ and only the highest isophote magnitude is selected for a specific sample location (x,y):

$$\nabla_g^{\perp} H(x,y) = \max\left[\nabla_R^{\perp} H(x,y), \nabla_G^{\perp} H(x,y), \nabla_B^{\perp} H(x,y)\right]. \tag{5.6}$$

Note, that experimental results were also carried out using the luminance information, i.e. the Y-channel in YUV color space. However, they did not lead to satisfying results, as also color edges play an important role. Hence, by using the highest isophote strength selected in the RGB color space, the texture transitions are detected more reliably.

After computing all priorities of the sample positions in $\delta\Omega$, a block matching algorithm determines the best exemplar patch $\mathbf{\Psi}_{(\widehat{u},\widehat{v})}$ to fill the missing samples in the highest priority patch $\mathbf{\Psi}_{(\widehat{x},\widehat{y})}$. $\mathbf{\Psi}_{(\widehat{x},\widehat{y})}$ is a squared patch centered at $(\widehat{x},\widehat{y})$ with $(\widehat{x},\widehat{y}) \in \delta\Omega$ [cf. Fig. 5.3 (a)]. $\mathbf{\Psi}_{(\widehat{u},\widehat{v})}$ is then selected from all possible reference patches $\mathbf{\Psi}_{(u,v)}$ by minimizing a distance measure according to Eq. 3.13. Assuming that $\boldsymbol{\psi}^{(u,v)}$ and $\boldsymbol{\psi}^{(\widehat{x},\widehat{y})}$ are vectorized versions of the patches $\mathbf{\Psi}_{(u,v)}$ and $\mathbf{\Psi}_{(\widehat{x},\widehat{y})}$, the cost function to find the minimal distance $d\left(\mathbf{\Psi}_{(\widehat{x},\widehat{y})}, \mathbf{\Psi}_{(u,v)}\right)$ is defined as follows:

$$\begin{aligned} d\left(\mathbf{\Psi}_{(\widehat{x},\widehat{y})}, \mathbf{\Psi}_{(u,v)}\right) = &\sum_{i=1}^{Q} \left(\boldsymbol{\psi}_i^{(u,v)} - \boldsymbol{\psi}_i^{(\widehat{x},\widehat{y})}\right)^2 + \omega_{\Omega} \sum_{i=1}^{Q_{\Omega}} \left(\boldsymbol{\psi}_i^{(u,v)} - \boldsymbol{\psi}_i^{(\widehat{x},\widehat{y})}\right)^2 \\ &+\omega_{\text{prev}} \sum_{i=1}^{Q+Q_{\Omega}} \left(\boldsymbol{\psi}_{n-1,i}^{(u,v)} - \boldsymbol{\psi}_{n-1,i}^{(\widehat{x},\widehat{y})}\right)^2, \end{aligned} \tag{5.7}$$

where Q is the number of original and Q_{Ω} the number of initialized samples in $\boldsymbol{\psi}^{(\widehat{x},\widehat{y})}$. ω_{Ω} is a weighting factor that regularizes the influence of the initialized samples on the overall cost. $\boldsymbol{\psi}_{n-1,i}^{(u,v)}$ and $\boldsymbol{\psi}_{n-1,i}^{(\widehat{x},\widehat{y})}$ are the vectorized versions of the patches centered at (u,v) and $(\widehat{x},\widehat{y})$ in the previous frame $n-1$. ω_{prev} is a weighting factor that regularizes the influence of the samples of the previous image. By including the third term in Eq. 5.7, the temporal consistency of the synthesized outcome is improved.

When the best candidate patch is found, the unknown positions in $\mathbf{\Psi}_{(\widehat{x},\widehat{y})}$ are filled from the corresponding positions in $\mathbf{\Psi}_{(\widehat{u},\widehat{v})}$ [cf. Fig. 5.3 (b)]. To smooth the transitions between adjacent patches the post-processing step outlined in Sec. 3.5 is applied. P is updated accordingly by copying the isophote magnitudes from the positions in $\mathbf{\Psi}_{(\widehat{u},\widehat{v})}$ to the corresponding hole positions in $\mathbf{\Psi}_{(\widehat{x},\widehat{y})}$. The filling process is conducted until all isophote magnitudes $[P(x,y)]$ of the samples

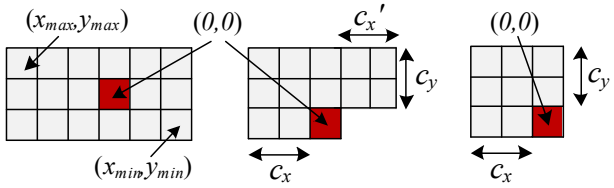


Figure 5.4: Examples of AR models with different neighborhoods: (a-left) non-causal, (b-middle) semi-causal and (c-right) causal.

in $\delta\Omega$ lie below a threshold t_p. The value for t_p is selected empirically in a way that all main structures in the virtual view are synthesized. After computing the dominant isophotes, it is assumed that the remaining texture segments to be filled are homogeneous and separated from each other [cf. Fig. 6.7 (f)].

5.3.2 Spatial Auto Regressive Model

To reconstruct the remaining homogeneous texture areas, an auto-regressive model is used. The AR model is a linear prediction model that attempts to express each value of a sequence as a linear combination of preceding terms. In this thesis a two dimensional AR model is applied. The general definition of a 2-D AR model can be expressed as follows:

$$\begin{aligned} \widehat{F}(x,y) = \sum_{i=y_{min}}^{y_{max}} \sum_{i=x_{min}}^{x_{max}} \alpha_{i,j} \cdot F(x-i, y-j) + \epsilon(x,y) \\ \text{with } \epsilon(x,y) \approx N(0,\sigma^2) \text{ and } (i,j) \neq (0,0), \end{aligned} \tag{5.8}$$

where $\widehat{F}(x,y)$ represents a synthesized sample at location (x,y) in the current frame F. $F(x,y)$ corresponds to known spatial neighbors. y_{min}, y_{max}, x_{min} and x_{max} characterize the order of the model (cf. Fig. 5.4 and 5.5) and $\alpha_{i,j}$ are the prediction coefficients with $j \in [y_{min}, y_{max}]$ and $i \in [x_{min}, x_{max}]$. The function $\epsilon(x,y)$ is a white noise process with zero mean and variance σ^2, i.e. $N(0,\sigma^2)$, and denotes the innovation signal which drives (innervates) the AR model. Commonly, white noise models can be represented by Gaussian, Laplacian, Poisson, Cauchy and Uniform noises [HHH09], among others. Due to the fact that the (additive) Gaussian noise provides a good noise approximation of many real-world applications (e.g. imaging systems) and generates mathematically tractable models, the innovation term $\epsilon(x,y)$ is typically represented by white Gaussian noise. However, the white noise driven AR process is only a subset of a general set of AR models [Kas80]. Hence, for other AR models different noise classes may also be a good choice [Kok98], [LL89] (cf. Fig. 5.4). Fig. 5.4 illustrates AR models (non-causal, semi-causal and causal) with different neighborhood structures. The pixel $\hat{F}(x,y)$ to be estimated is depicted in red. Note that the "semi-causal" neighborhood can be extended in horizontal [cf. Fig. 5.4 (b)]

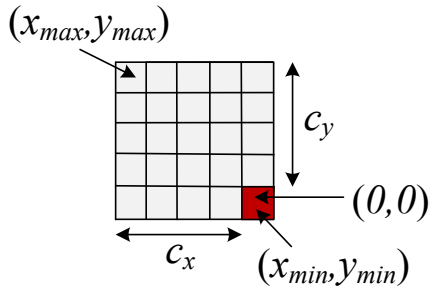


Figure 5.5: 2-D causal coefficient model used in the proposed framework. © [2016] IEEE.

as well as in vertical directions. In the view synthesis application the filling is driven from the background towards the foreground. Therefore, the image information is only available either on the right or on the left side of the hole. Furthermore, the shape of the hole can vary, i.e. if the hole is synthesized in a raster-scan order only the image information above the pixel to be synthesized is certainly available. Hence, the causal AR model is chosen (cf. Fig. 5.5). With respect to Fig. 5.5 and Eq. 5.8, $y_{min} = x_{min} = 0$, $x_{max} = c_x$ and $y_{max} = c_y$, where c_x, c_y represent the horizontal and vertical orders of the AR model used in this thesis.

Before Ω can be filled, the model parameters have to be determined. According to Eq. 5.8, there are three sets of unknown parameters: the samples $\widehat{F}(x, y)$, the coefficients $\alpha_{i,j}$ and the variance σ^2 of the innovation process ϵ. $\widehat{F}(x, y)$ will be obtained in the final completion step (cf. Sec. 5.3.6). The first objectives are (1) to define a training area, (2) to estimate the optimal model coefficients and (3) to calculate the variance σ^2 of the white Gaussian noise.

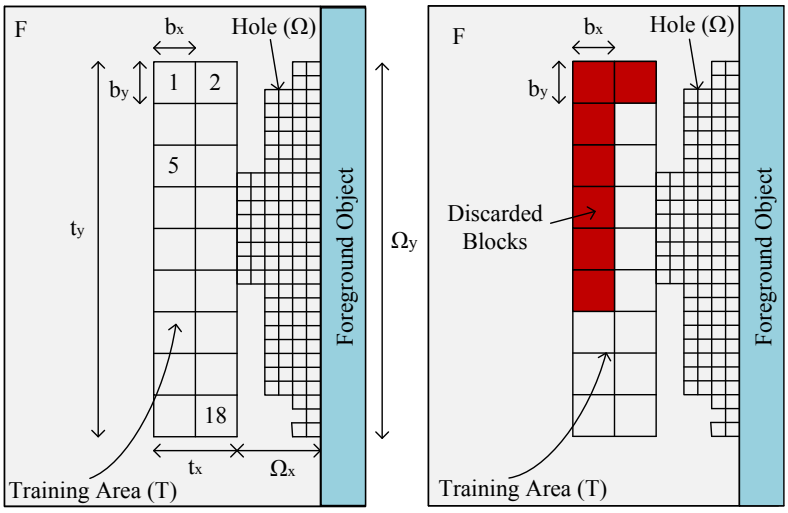
5.3.3 Defining the Training Area

In order to select the optimal training area, a stationnarity criterion, which discards unreliable parts, is presented. Unreliable textures are defined as causes of instationarities in a given training area. The training area is defined adjacent to the hole in the background area (cf. Fig. 5.6) and should be large enough. The training region is divided into blocks of size $b_x \times b_y$ (cf. Fig. 5.6). Two blocks are considered in the horizontal direction ($t_x = 2 \cdot b_x$). In the vertical direction, the number of blocks (N_{b_y}) is adapted to the hole size and thus calculated as follows:

$$N_{b_y} = \left\lfloor \frac{\Omega_y}{b_y} \right\rfloor, \tag{5.9}$$

where Ω_y represents the height of the hole [cf. Fig. 5.6 (a)]. The height of the training area can then be computed as follows: $t_y = b_y \cdot N_{b_y}$. The first block of the training area starts at the same height (y position), as the hole occurs in the image.

The mean μ_{block} and variance σ^2_{block} of the samples in each block is calculated. $\mu_{\text{block},j}$ and $\sigma^2_{\text{block},j}$ ($j = 1, ..., 2 \cdot N_{b_y}$) are used for outlier identification. If the absolute mean and variance between the sample values of the two blocks is smaller than $t_{\mu_{\text{block}}}$ and $t_{\sigma^2_{\text{block}}}$ respectively, these



(a) (b)

Figure 5.6: Illustration of the stationarity criterion for a training area adjacent to the unknown area Ω. (a) The training region is divided into blocks (numbered from 1 to 18 in this example) of size $(b_x \times b_y)$. (b) Using our stationarity criterion, certain blocks are discarded from the training area. © [2016] IEEE.

two blocks are considered as neighbors, which contain statistically similar textures. $t_{\mu_{\text{block}}}$ and $t_{\sigma^2_{\text{block}}}$ are selectable parameters (cf. Sec. 6.4). As a result, a set of segments is obtained. The largest region is chosen as the validated training area [cf. Fig. 5.6 (b)]. Synthesis results are shown in Fig. 5.7. It can be seen that significant visual improvements become visible, when gross instationarities are discarded.

5.3.4 Estimating the AR Coeffecients

The optimal AR coefficients can be estimated as the solution to the following least square problem:

$$\boldsymbol{\alpha}_{C\times 1} = \arg\min_{\alpha} \|\mathbf{y}_{S\times 1} - \mathbf{X}_{S\times C}\boldsymbol{\alpha}_{C\times 1}\|^2 \tag{5.10}$$

where $\boldsymbol{\alpha}$ $\left(\boldsymbol{\alpha} \in \mathbb{R}^C\right)$ is a vector containing the AR coefficients (cf. Fig. 5.5).

$$\boldsymbol{\alpha} = [\alpha_{1,0}, \alpha_{2,0}, \cdots, \alpha_{c_x,c_y}]^T. \tag{5.11}$$

$\boldsymbol{y}$ $(\boldsymbol{y} \in \mathbb{R}^S)$ denotes the known samples F in the sub-training area [cf. Fig. 5.8 (b)]

$$\boldsymbol{y} = [F(x_0, y_0), \cdots, F(x_0 + s_x - 1, y_0 + s_y - 1)]^T. \tag{5.12}$$

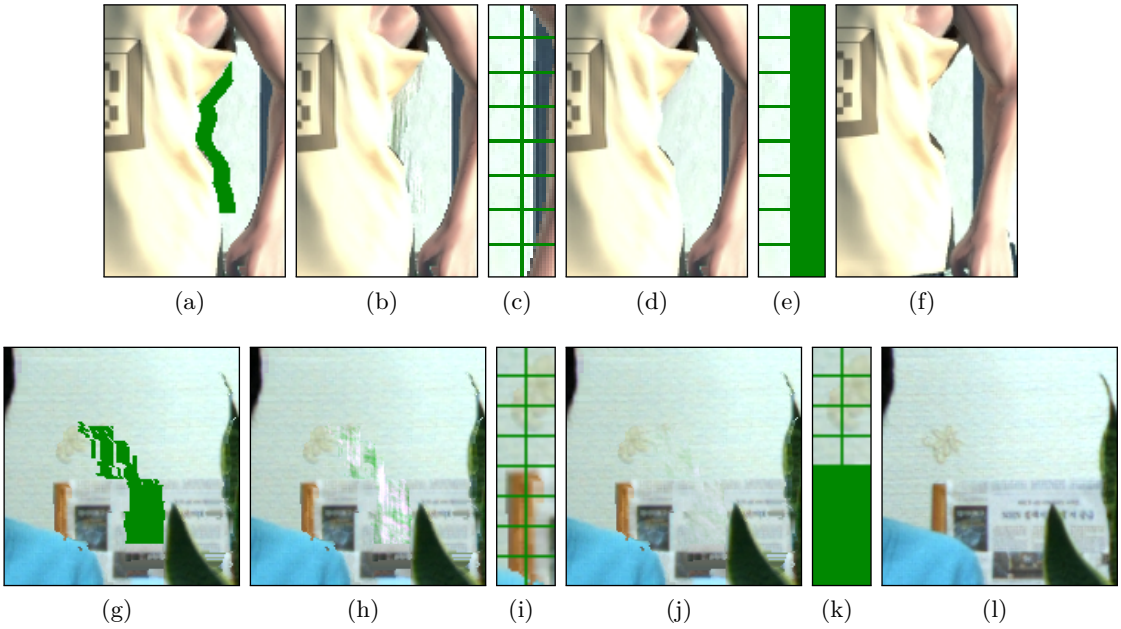


Figure 5.7: Undo Dancer (virtual view 9 rendered from original view 5), Frame 1 (a)-(f) and Newspaper (virtual view 4 rendered from original view 6) Frame 6 (g)-(l). (a), (g) Ω is marked in green. (b), (h) The synthesis result, if the complete training area (c), (i) is used. (d), (j) Synthesis result, if the validated trainings area (e), (k) is used. (f), (l) The original reference image. © [2016] IEEE.

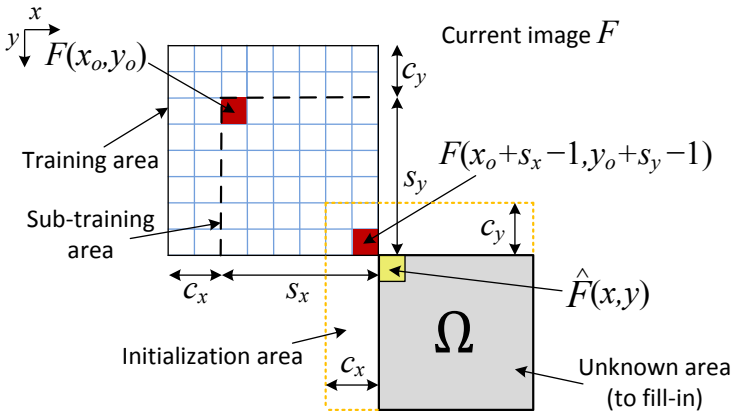


Figure 5.8: Notation of the 2D-AR approach. For a better visualization the training area is placed in the top-left corner.

and $\boldsymbol{X}$ $\left(\mathbf{X} \in \mathbb{R}^{S \times C}\right)$ represents the neighboring sample matrix for each of the samples in $\boldsymbol{y}$:

$$\boldsymbol{X} = \begin{bmatrix} F(x_o - 1, y_0) & \cdots & F(x_o - c_x, y_0 - c_y) \\ \vdots & \ddots & \vdots \\ F(x_o + s_x - 2, y_0 + s_y - 1) & & F(x_o + s_x - c_x - 1, y_0 + s_y - c_y - 1) \end{bmatrix} \tag{5.13}$$

Furthermore, the subscripts in Eq. 5.10 represent the dimension of the vectors and matrices, where C is the number of prediction coefficients. The causal neighborhood C is determined as follows (cf. Fig. 5.5):

$$C = (c_x + 1)(c_y + 1) - 1. \tag{5.14}$$

S denotes the number of samples in the sub-training area (the number of linear equations), e.g. using the neighborhood example in Fig. 5.8: $S = s_x s_y$.

Hence, Eq. 5.10 can be solved with the closed-form solution:

$$\boldsymbol{\alpha} = (\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{y}). \tag{5.15}$$

As the set of coefficients $\boldsymbol{\alpha}$ minimizes the model error in a least-square sense, samples that are unsuitable for completion in the current training area are assigned smaller coefficients, i.e., the AR model adapts to the local texture characteristics. In case Eq. 5.15 cannot be solved due to non-invertible matrices $\mathbf{X}^T\mathbf{X}$ that are bound to appear, a pseudo inverse of the matrix $\mathbf{X}^T\mathbf{X}$ can be determined [GK65]. The optimal coefficients are determined in the luminance signal of an image. During the completion of the hole (cf. Sec. 5.3.6) these optimal coefficients are then applied to all channels of the image. However, there exist also other efficient ways to estimate the AR coefficients. Alternatively, the covariance method (Yule-Walker equations) can be used [SP96].

5.3.5 Estimation of the Innovation Term

Once the AR coefficients have been estimated, the standard deviation σ^2 of the innovation term $\epsilon(x, y)$ can be calculated using the completion error:

$$Err = \left\| \boldsymbol{y}_{S\times 1} - \boldsymbol{X}_{S\times C}\boldsymbol{\alpha}_{C\times 1} \right\|^2 \tag{5.16}$$

which is normalized by the size of the training area [SP96]:

$$\sigma^2 = \frac{Err}{S} \tag{5.17}$$

Note that Err is estimated on the same area as the one used to learn the AR coefficients, i.e. the validated training area. The error term Err is determined by simulating a completion $\boldsymbol{X\alpha}$ of the validated training area [cf. Fig. 5.6 (b)]. During the simulation, F is not modified. This is done to ensure that the completion error only stems from the imperfectly predicted AR coefficients and not from the use of synthesized samples in the completion (simulation) procedure.

5.3.6 Completion of the Missing Area

Finally, all the model parameters have been estimated and Ω can be completed in a raster-scan way. Hence, Eq. 5.8 is applied to each sample $(x, y) \in \Omega$.

5.3.7 AR Verification Criterion and Fall-Back

Completion methods always have to deal with the issue that the example texture surrounding Ω is finite. Hence, the best AR settings may still be an unsatisfactory compromise. In fact, it may happen that no good fit (training area) is found for the current neighborhood. Furthermore, it is possible that the estimated AR coefficients overfit the training data. If such wrong AR parameters are used to complete the unknown area, an erroneous extrapolation of the existing texture will be the consequence. Hence, an AR verification criterion is proposed in this work. Since the completion process is derived from the initialization samples, i.e. the samples adjacent to Ω, the statistical properties of the extrapolated and the training texture should be similar. If F_{min} and F_{max} are the lowest and highest sample intensity values in the background neighborhood of the hole, the synthesis is considered as unsuccessful if:

$$\begin{cases} F_{min} - \tau > \widehat{F}(x,y) \\ \text{or} \qquad\qquad\qquad\qquad \text{with } (x,y) \in \Omega \\ F_{max} + \tau < \widehat{F}(x,y) \end{cases} \tag{5.18}$$

where τ is a threshold value that allows a small deviation from F_{min} and F_{max}. This effective criterion is motivated by the observation that AR distortions typically lead to gross luma and chroma variations that extremely deviate from the spatial context. In case the criterion Eq. 5.18 detects a failed synthesis, the patch-based texture synthesis approach proposed in Sec. 3.4 and 3.5 is used as a fall-back to reconstruct the missing area. The main difference of the method proposed in Sec. 3.4 and 3.5 and the new hybrid synthesis approach (cf. 5.3.1) is that the former method (cf. Sec. 3.4 and 3.5) considers both structure and texture in the patch-based filling routine.

5.3.8 Post-Processing

Finally, a specific post-processing filter is applied. In the texture synthesis steps (cf. Sec. 5.3), the texture filling starts from the background towards the foreground objects. Due to this fact, obvious transitions may occur between the boundary of the foreground object and the synthesized area. To conceal these transitions, a Gaussian filter (window size: 3×3, sigma: 0.5) is applied on the foreground and synthesized background transitions. A transition boundary of four pixels is considered. By applying the post-processing method, unnaturally sharp edges that occasionally appear in the background/foreground transitions are blurred and thus invisible to observers.

5.4 Chapter Summary and Limitations

In this chapter, a new method to fill uncovered areas in a DIBR framework is introduced. The holes in the virtual views are reconstructed using a new hybrid texture synthesis approach combining the advantages of patch-based and parametric methods. First, a new patch-based texture synthesis is utilized to separate different texture classes, where only the transitions between several homogeneous texture patterns are synthesized. To improve the detection of the important texture transitions, an enhanced pre-processing module is incorporated into the framework. Subsequently, a fast autoregressive parametric synthesis approach reconstructs the separated homogeneous texture classes by training its parameters in validated known texture regions. To select an appropriate training area, a new stationarity criterion is proposed that is based on statistical texture similarity. The parametric synthesis results are subsequently evaluated with a new quality criterion. In case of identified errors in the parametrically synthesized areas, the texture synthesis method proposed in Sec. 3.4 is used as a fall-back. Finally, a post-processing step is applied to improve the background-foreground transitions in the virtual view. The proposed method computes the uncovered textures frame-wise. This can lead to temporal inconsistencies. Hence, this framework should be combined with the motion compensation methods proposed in Ch. 3 and Ch. 4.

6 Experimental Results

In this chapter, detailed experiments are described. In Sec. 6.1, the data set as well as the used evaluation measures are defined. Then, the individual parts of the proposed frameworks presented in Ch. 3, 4 and 5 are evaluated in Sec. 6.2, 6.3 and 6.4, respectively. Finally, in Sec. 6.5 the overall experiments are shown.

6.1 Data Set and Quality Measures

For evaluating the proposed algorithms, eight MVD test sequences (textured images and depth maps), provided by MPEG are used: "Book Arrival" (S1, 100 frames), "Lovebird1" (S2, 150 frames), "Newspaper" (S3, 200 frames), "Mobile" (S4, 200 frames), "Undo Dancer" (S5, 250 frames), "Ghost Town Fly" (S6, 250 frames), "Poznan Hall2" (S7, 198 frames) and "Balloons" (S8, 300 frames). S1, S2, S3 and S8 have a resolution of 1024×768 samples, S5, S6 and S7 have a resolution of 1920×1088 samples and S4 has a resolution of 720×540 samples (cf. Appx. A). The data set consists of real world (S1, S2, S3, S7 and S8) and computer graphic scenes (S4, S5, S6) captured with a stationary (S1-S4) or a dynamic camera (S5-S8). The depth maps of the real world scenes are estimated while the depth maps of the computer graphic scenes are computer generated. The texture in the sequences varies from simple (S1, S7) to complex (S2, S3, S6, S8) to very complex (S4, S5) (cf. Appx. A).

For each sequence, the rectified videos of several views with slightly different camera perspectives are available. The baseline between two adjacent cameras corresponds to the eye distance ($\approx$ 65 mm) or the double eye distance ($\approx$ 130 mm) for all test sequences. To assess the performance of the proposed approaches one or two original, but not necessarily adjacent, cameras (left and right view) are considered. The following two scenarios are evaluated, which are relevant for multiview content:

- View extrapolation with a small baseline ($\approx$ eye distance).
- View extrapolation with a large baseline ($\approx$ double eye distance).

The extrapolation capabilities of the proposed approaches are evaluated as follows: an outermost virtual view is rendered from an original sequence with its associated depth map. The outermost views are evaluated by measuring the PSNR and the Structural Similarity (SSIM) [WBSS04] in the luminance channel (Y-channel) [WBSS04] between the rendered frames and the

original data. SSIM is provided in addition to PSNR, since PSNR is not always a reliable measure to judge the quality of texture completion results [WBSS04]. A higher PSNR value is related to a higher similarity between the original and the synthesized image. SSIM is normalized to values between 0-1. Here, a value of one corresponds to identical images and thus maximum similarity. In order to determine the temporal consistency of the virtual sequences, the temporal consistency measure proposed by Schmeing and Jiang [SJ11] is used (cf. Sec. 6.5.2).

The proposed approaches are compared to five state-of-the-art methods (M1-M5) that have been especially developed for the filling of uncovered areas in extrapolated virtual views:

- Two extensions of the MPEG View Synthesis Reference Software (VSRS) that have been proposed for the extrapolation of virtual views: VSRS-alpha-Gist [LH11] (M1) and VSRS-alpha-Etri [BKY+11] (M2). In M1 a weighted average filtering method is used to fill the unknown areas while in M2 an inpainting method is applied.
- Two frameworks developed by Daribo et al. that either use patch-based texture synthesis [DS11] (M3) or pre-process the depth maps [DSF+12] (M4) to fill the disocclusions.
- A framework proposed by Ahn et al. that utilizes patch-based texture synthesis to fill the holes in the virtual view [AK13] (M5).

In the following, the methods proposed in this thesis (cf. Ch. 3-5) are abbreviated as follows:

- The view synthesis method for sequences with static background (cf. Ch. 3) is denoted as P1.
- The view synthesis method for sequences with global background motion (cf. Ch. 4) is denoted as P2.
- The hybrid view synthesis framework (cf. Ch. 5) is denoted as P3.

6.2 Assessment of the View Synthesis Method for Static Backgrounds

In Sec. 6.2.1, 6.2.2 and 6.2.3, relevant modules of the proposed view synthesis framework approach for sequences with static background (cf. Ch. 3, P1), i.e. the depth map filling algorithm, the texture synthesis method and the sprite updating module, are evaluated to assess their contribution to the overall system performance. Since P1 was developed for sequences with a static camera setup, only S1-S4 are used for evaluation.

6.2.1 Assessment of the Depth Map Filling Algorithm

First, the new depth map filling method as described in Sec. 3.2 is analyzed. As mentioned in Sec. 3.2, the most important depth map filling parameter is the k-means clustering window

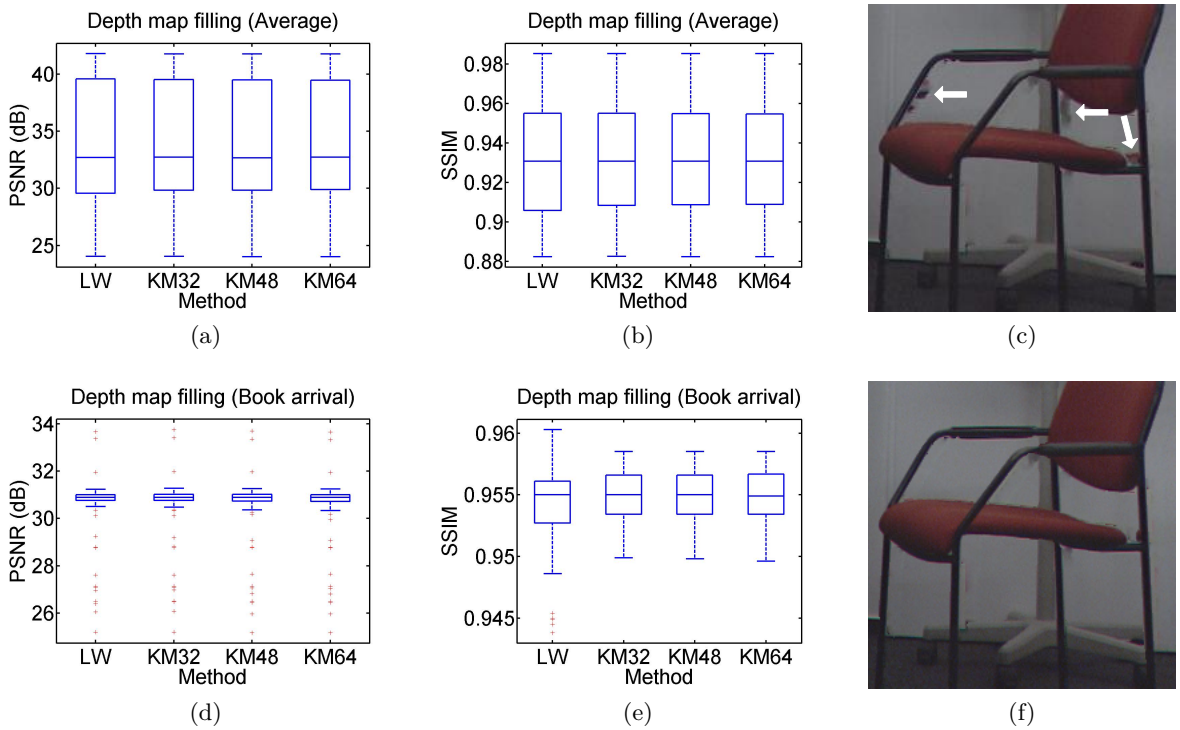


Figure 6.1: Influence of the proposed depth map filling method on the filling results. Average values for (a) PSNR and (b) SSIM, over the whole test set. Objective results for the sequence "Book Arrival" with (d) PSNR and (e) SSIM. Subjective differences between (c) line-wise method without blob removal (LW) and (f) k-means clustering with $M = 32$ (KM32). © [2011] IEEE.

sized $m \times m$. Experiments were conducted for all video sequences assuming a square window. Furthermore, all tests were performed using a large baseline ($\approx$ double eye distance) in order to access larger disoccluded areas. Please note that PSNR and SSIM are measured in the luminance channel of the textured images. Fig. 6.1 (a), (b) depict the average values that were achieved for PSNR and SSIM over the whole test set. It can be seen that all filling methods [line-wise without blob removal (LW) and k-means clustering with different window sizes (32×32, 48×48, 64×64)] perform similarly in terms of PSNR and SSIM. Some objective results for the "Book Arrival" (S1) sequence are shown in Fig. 6.1 (d), (e). In the next step, visual results are taken into consideration to find the optimal filling method. As shown in Fig. 6.1 (c), (f), for "Book Arrival" (S1), distortions can be observed for the LW approach, while k-means clustering generates good results. Therefore, visual results of the test sequences leaded to the conclusion that the k-means clustering method with window size $m = 32$ produces the best results [cf. Fig. 6.1(c), (f)].

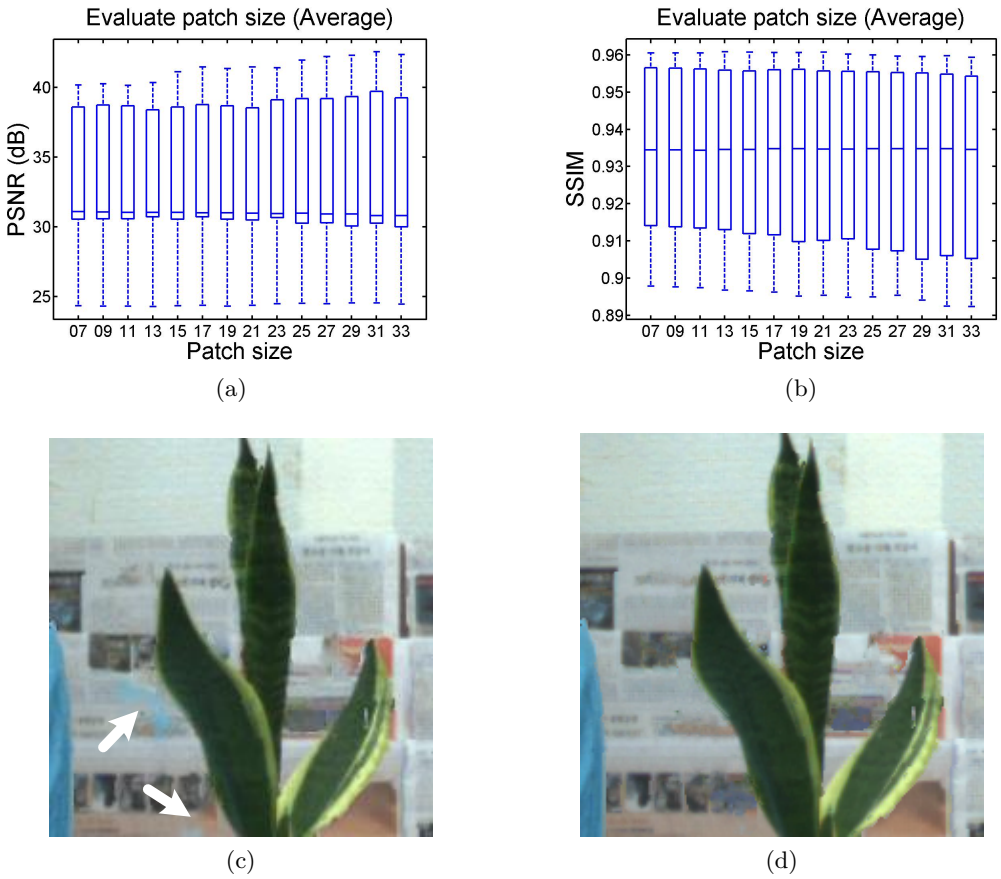


Figure 6.2: Influence of the patch size on the view synthesis accuracy. Overall values for (a) PSNR and (b) SSIM, over the whole test set. Subjective differences between the results after filling the disoccluded area with a patch of size (c) 25×25 and (d) 9×9. © [2011] IEEE.

6.2.2 Assessment of the Texture Synthesis Algorithm

In the following, the parameter used during the texture synthesis routine are evaluated. The most important texture synthesis parameters (cf. Sec. 3.4 and 3.5) are:

- The search area A and its corresponding sub-sampling factor s
- The patch size q
- The weighting factor ω_Ω (3.13)

Again, all tests were performed using the scenario with twice the regular baseline in order to have larger disoccluded areas. For reducing the complexity of estimating the texture synthesis parameters, a set of five key frames from each sequence and view (left and right) is used. The key frames were selected manually to ensure that all of the relevant scene content as well as

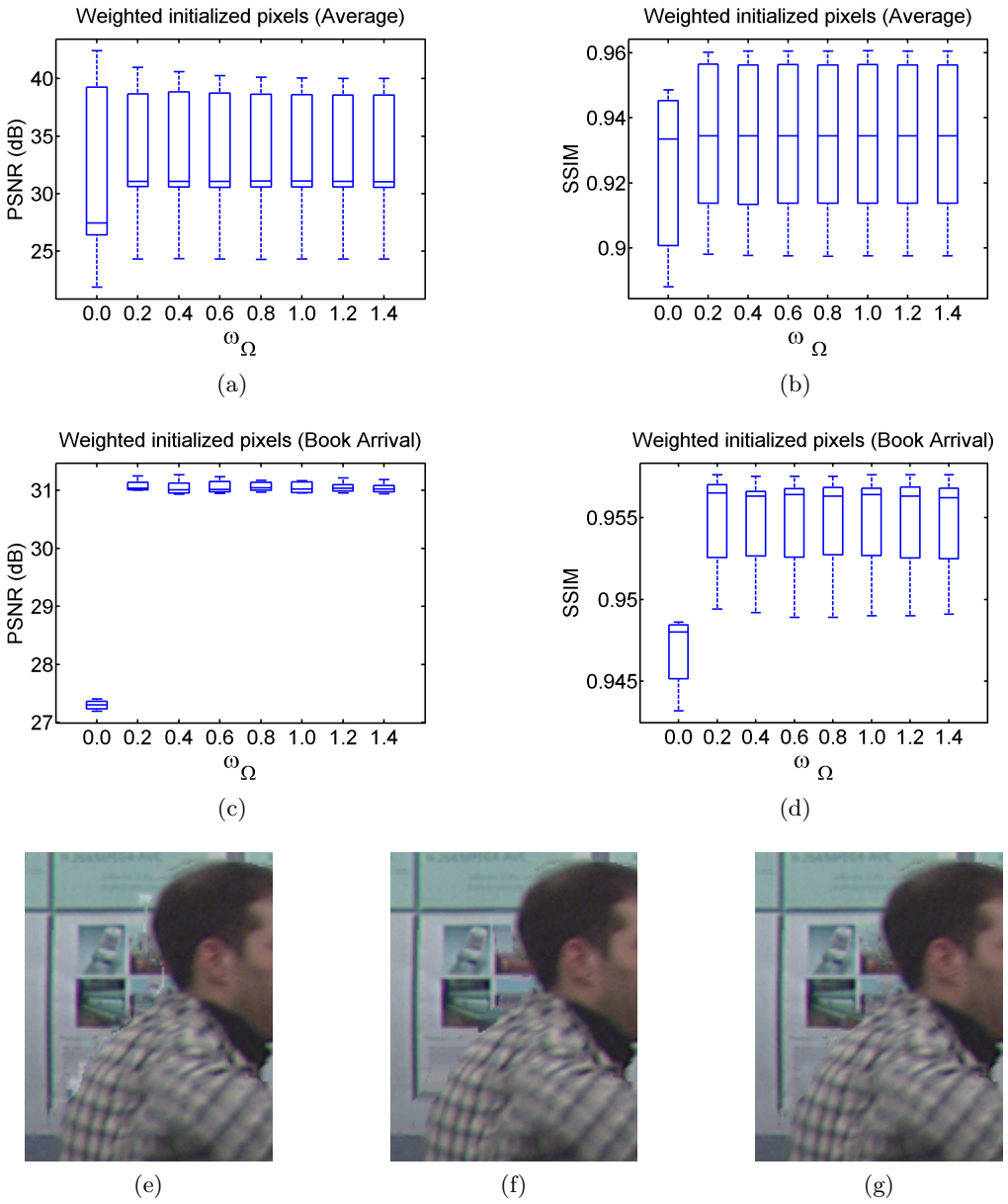


Figure 6.3: Influence of the initialization step on the view synthesis accuracy. Overall values for (a) PSNR and (b) SSIM, for patch size of 9×9 over the entire test set. The objective results for the sequence "Book Arrival" with (c) PSNR and (d) SSIM. Result after filling the disoccluded area (e) without initialization. (f) Result using the initialization step (ω_Ω=0.2) without texture synthesis and (g) result using the initialization step (ω_Ω=0.2) before texture synthesis. © [2011] IEEE.

large disocclusions were considered. Furthermore, the depth map was filled using the optimized k-means clustering settings determined in the previous section.

The search area A is an important parameter, which mainly depends on the content of the

considered image. It has been observed that for the test sequences analyzed in this work, the view synthesis performance is not very sensitive to the size of the search area. This may, however, be different for other sequences. It is possible to decrease the search complexity by sub-sampling A with a factor $s \in \mathbb{N}$. Increasing s decreases run-time and reduces the quality of the results. It was found that it is adequate to set $A = 80 \times 80$ samples and $s = 2$, so that a reasonable compromise between complexity and sufficient quality is achieved.

Next, the optimal patch size is determined. Patches are assumed to be squares to simplify the evaluation. Moreover, the initialization step is disabled in this experiment, i.e. only the texture synthesis approach is taken into account in this experiment. The influence of the patch size on the view synthesis results is shown in Fig. 6.2 (a), (b). No significant difference can be observed between the diverse patch size selections. However, visual results of synthesized views, show that a patch size of 9×9 ($q = 9$) yields better subjective results than larger patches, e.g. 25×25. The larger the block size is, the more likely artifacts occur. Fig. 6.2 (c) and (d) illustrate the difference between the results obtained after the disoccluded area was filled with a patch size of 9×9 and 25×25. It can be seen that foreground colors have been copied into the background area with a patch size of 25×25.

In the next experiment, the texture initialization step is evaluated (cf. Sec. 3.4). The impact of the initialization of the uncovered regions in the textured images is depicted in Fig. 6.3. Note that the texture synthesis step realized after the initialization is performed with the optimized patch size and search area $q = 9$, $A = 80 \times 80$ and $s = 2$. It can be seen that for both measures [cf. Fig. 11 (a) and (b)], a quality improvement can be achieved by setting $\omega_\Omega \neq 0$ (3.13). When ω_Ω is set to 0.2 instead of 0.0, the PSNR is increased by approximately $3dB$ on average, while SSIM rises by 0.02. The gains are even larger for "Book Arrival" (S1) as shown in Fig. 6.3 (c), (d). Increasing ω_Ω further does, however, not yield further gains. Therefore the final setting for this parameter is selected to be 0.2. Fig. 6.3 (e)-(g) further show visual results for "Book Arrival" (S1).

6.2.3 Assessment of the Sprite Updating Algorithm

By using a background sprite, the temporal consistency is improved in the virtual view. The impact of the sprite is further evaluated in Sec. 6.5.2. Nevertheless, the updating process highly depends on the quality of the depth map (cf. Sec. 3.3). If unreliable depth maps are used, inappropriate image information can be falsely copied into the sprite and propagated to subsequent frames. Therefore, the quality of the results can suffer. Vice versa, in the case of high quality depth maps, the sprite updating works efficiently and leads to accurate temporally consistent synthesis results.

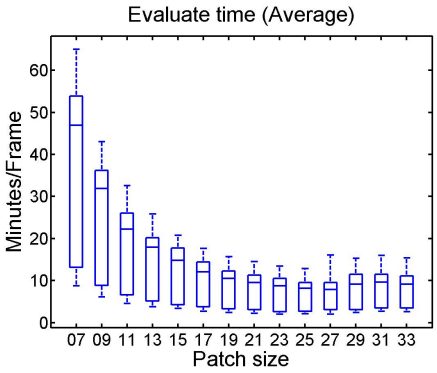


Figure 6.4: Evaluation of the run-time using different patch sizes. © [2011] IEEE.

Parameter	Value
A	80×80
s	2
β	15
m	32
q	9
ω_Ω	0.2

Table 6.1: Parameter settings for P1.

6.2.4 Complexity Assessment

The complexity and thus the run-time of the proposed algorithm is mainly dominated by the following three aspects:

- The search area A with the corresponding sub-sampling factor s, of the texture synthesis approach.
- The patch size used in the texture refinement step.
- The utilized cloning method used in the updating process.

The other functions are less time consuming and their contribution to the overall complexity is rather small. A PC with an Intel Xeon CPU and 4 GB RAM was used in the experiments to evaluate P1 (cf. Sec. 6.2). The software is currently implemented in MATLAB. To evaluate the complexity of the optimized settings for β, m, q and ω_Ω given in TABLE 6.1 are used. According to the results obtained, varying the search area A and the sub-sampling factor s, strongly influences the complexity of the proposed approach. The complexity increases by a factor of approximately 1.5 when A is doubled. On the other hand, when s is increased from 1 to 2, the complexity reduces by a factor of approximately 1.31. Increasing s from 1 to 4 yields a complexity reduction of approximately 3.24. To update the current frame from the background sprite, the covariant cloning is used to fit the background data into the frame (cf. Sec. 6.2.1). For every sample

position to be updated from the background, a linear equation has to be solved. Hence, the complexity is proportional to the number of samples which are copied from the background sprite to the actual frame, which corresponds to a linear growth of complexity Fig. 6.4 depicts the results obtained by varying the patch size. Furthermore, the results are generated with the key frames used in Sec. 6.2.2, i.e. the run-time represents the mean processing time of the different single images (no time consistency is available) evaluated with the same patch size. It can be seen that the complexity is approximately inversely proportional to the patch size growth. This relates to the fact that larger patches cover more unknown pixels. Hence, fewer search iterations have to be run. If texture synthesis with time consistency (using sprite update) is applied with a patch of size 9×9, the run-time decreases by a factor of ≈ 3.2 compared to texture synthesis without time consistency (cf. Fig. 6.4). However, in applications where run-time is of more importance than quality, a patch size of 25×25 appears to be the better choice.

6.3 Evaluation of the View Synthesis Method for Sequences with Global Background Motion

In Sec. 6.3.4 and 6.5.2 relevant modules of the proposed view synthesis framework approach for sequences with global background motion (cf. Ch. 4, P2), i.e. the new warping routine and the motion compensation method, are evaluated to assess their contribution to the overall system performance. In section 6.3.4, the new warping routine of P2 (cf. Sec. 4.2, P2) is evaluated. First, the proposed metric used to judge the spatial consistency of neighboring views is outlined in Sec. 6.3.3-6.3.3.

6.3.1 Measuring of the Spatial Consistency

Autostereoscopic displays support parallax head motions. Therefore, the spatial consistency of the virtual views is important for the visual 3-D experience and the received quality. However, classical full-reference quality assessment tools cannot be applied for this purpose, since only a limited number of original camera views is available. In this work, a no-reference assessment tool [ITU00] proposed by the International Telecommunication Union (ITU) is considered for this purpose. Originally, the measure is used to judge the temporal consistency. Here, the metric is extended for measuring the spatial consistency between neighboring views

First, the no-reference metric, namely the Temporal perceptual Information (TI) measure, as proposed in [ITU00] is outlined in Sec. 6.3.2. Subsequently, the proposed Spatial Consistency Metric (SCM) derived from the TI is explained in Sec. 6.3.3.

6.3.2 Temporal Perceptual Information Measurement

The TI metric [ITU00] was developed to determine the temporal perception of a sequence. The TI is based on the motion difference $M_n(x, y)$ between the sample values at the same spatial location

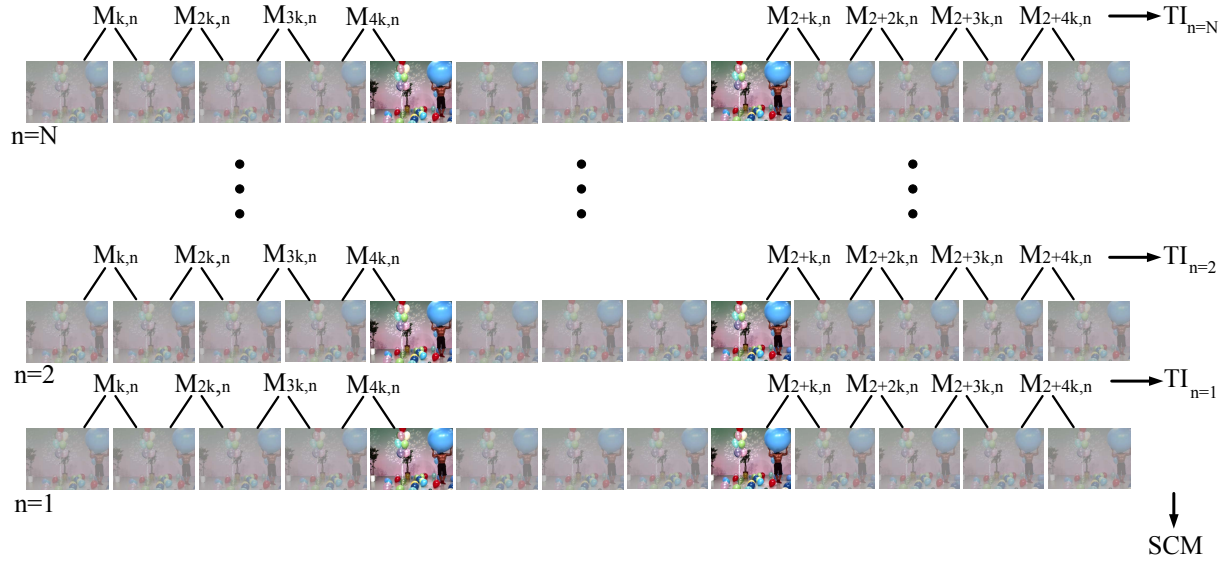


Figure 6.5: Measuring of the spatial consistency.

at successive frames, i.e. $F_n(x, y)$ and $F_{n-1}(x, y)$. The motion difference is then computed as follows:

$$M_n(x, y) = F_n(x, y) - F_{n-1}(x, y), \tag{6.1}$$

The TI is then computed as the maximum over a set of frames ($\max_{time}$) of the standard deviation over space(std_{space}):

$$\text{TI} = \max_{time}\{\text{std}_{space}[M_n(x, y)]\}. \tag{6.2}$$

Higher TI values indicate that a higher motion flow is present between successive frames.

6.3.3 Proposed Spacial Consistency Measure

In the test scenario, two original cameras are utilized to synthesize all remaining virtual views (cf. Fig. 4.3, 2-view MVD). In interpolated virtual views in-between the original cameras, only small areas become uncovered, since the missing textures can either be filled from the left or the right original camera. Therefore, the spatial consistency is measured in the extrapolated views alone. Here, large portions of the texture are synthesized. The main assumption is that the spatial consistency is higher when the changes in the textures of spatial neighboring frames are small. Therefore, these changes are measured in-between spatially adjacent images. Fig. 6.5 shows the functionality of the proposed SCM. First, the motion differences between slightly parallax shifted virtual and/or original camera locations at a specific frame-position n is measured using Eq. 6.1 (cf. Fig. 6.5), i.e. from $M_{k,n}$ to $M_{1,n}$ and from $M_{2+k,n}$ to $M_{3,n}$ (cf. Fig. 6.5), where k represents the distance between two adjacent views (cf. Sec. 2.3.1). For the scenarios illustrated in Fig. 4.3 and Fig. 6.5, k is set to $k = 0.25$ since three intermediate views are chosen. Then, the TI_n of a frame number n is determined by applying Eq. 6.2 considering all measured motion difference

Parameter	Value
z	5
t_{dr}	15
t_{psnr}	21

Table 6.2: Parameter settings for P2.

Seq.	SCM	
	P2Ex	**P2**
S1	19.6617	**19.0060**
S2	9.5626	**8.8739**
S3	29.0741	**28.4993**
S4	18.8547	**18.6808**
S5	20.3011	**20.1300**
S6	8.7785	**8.7448**
S7	10.2127	**10.1736**
S8	15.1206	**15.1201**

Table 6.3: SCM results by a state-of-the-art warping method (P2Ex) and the proposed novel warping method (P2).

(M) at the time instance as shown in Fig. 6.5. The spatial consistency for the whole sequence, is then computed as follows:

$$\text{SCM} = \frac{1}{N} \sum_{n=1}^{N} \text{TI}_n. \tag{6.3}$$

where smaller SCM values indicate a higher spatial consistency of neighboring views.

6.3.4 Evaluation of the new Warping Routine

The improved spatial consistency of the new warping method is evaluated using the following cameras setups for the sequences S1-S8: S1: 12, 10, 6, 8. S2: 4, 6, 8, 10. S3: 2, 4, 6, 8. S4: 3, 5, 5, 7. S5: 1, 3, 5, 7. S6: 7, 5, 3, 1. S7: 4, 5, 6, 7. S8: 1, 3, 5, 7. The first number represents the virtual left camera (c=0), the second number, the original left camera (c=1), the third number, the original right camera (c=2) and the last number, the virtual right camera (c=3). The distance between the original cameras corresponds to a large baseline (double eye distance). Note that only the extrapolated outermost cameras are computed from an original view. The remaining virtual views are computed utilizing two anchor cameras, i.e. an outermost virtual and an original cameras (cf. Sec. 4.2). The distance k between two spatial neighboring views is set to $1/4 \cdot b$ for S1, S2, S3, S4, S5, S6 and S8 and to $1/8 \cdot b$ for S7 [ISO11]

For evaluation purposes all extrapolated virtual views have been additionally computed from the original views alone, using the background motion compensation and texture synthesis methods of P2. This method is denoted as P2Ex in the following. This means, the new warping routine

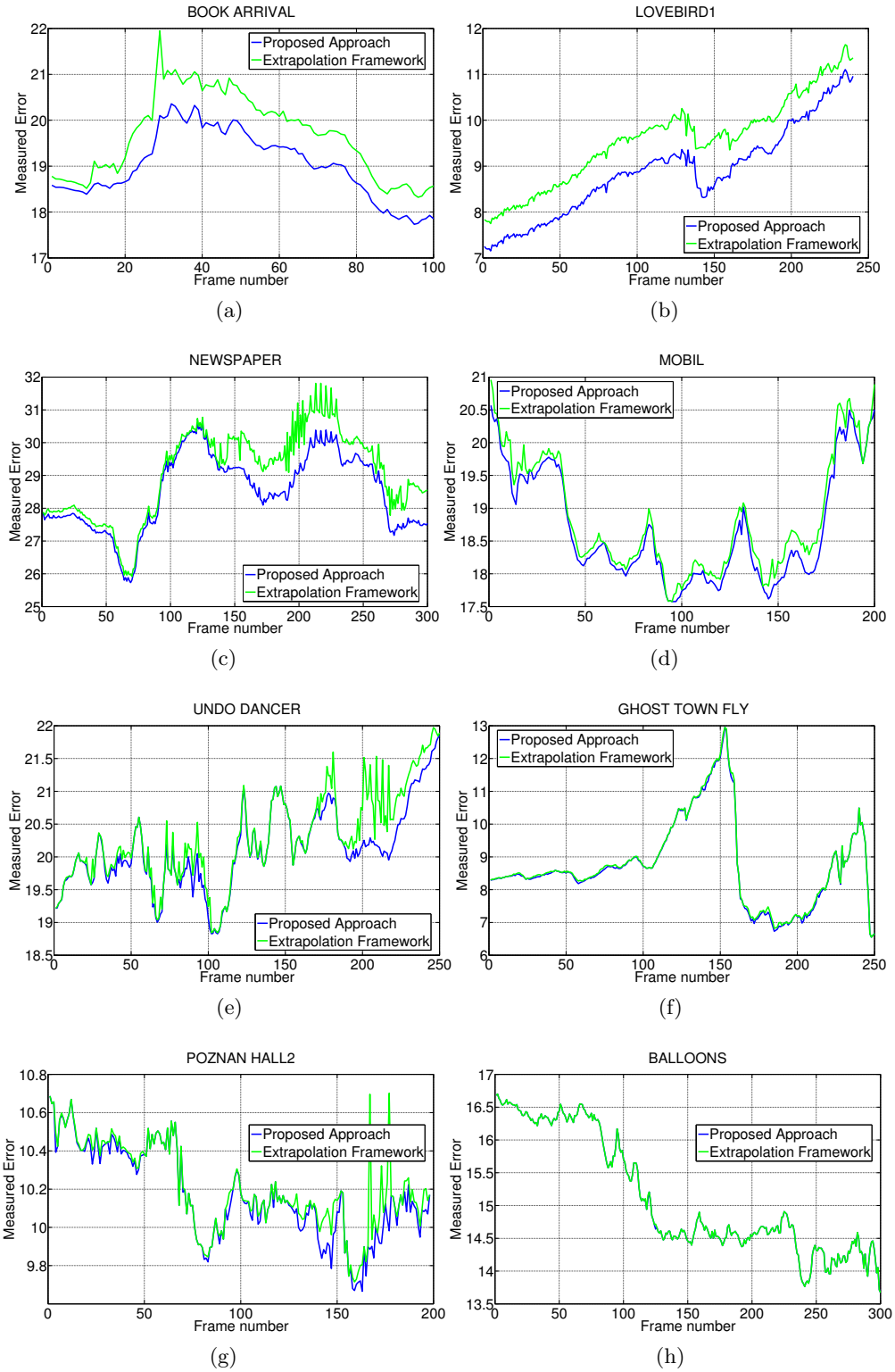


Figure 6.6: Objective results for the Spatial Consistency Measure (SCM).

Parameter	Value
t_p	0.09
ω_Ω	0.2
$b_x = b_y$	10
$c_x = c_y$	4
$t_{\mu_{\text{block}}}$	10
$t_{\sigma^2_{\text{block}}}$	10
τ	30

Table 6.4: Parameter settings for P3.

is not incorporated in P2Ex. In this way the improvements of the new warping scheme can be determined by comparing P2 with P2Ex. To compute the results of P2 and P2Ex, the same parameter settings shown in TABLE 6.2 have been used. The patch size of the patch-based texture synthesis is chosen according to the resolution of the sequences. In Sec. 6.2.2 the patch size was evaluated and set to 9×9 for videos with a resolution of up to 1024×768 samples. However, for 3-D video sequences with higher resolution, i.e. 1920×1088 samples, the patch size is increased to 21×21.

The objective results for the SCM are given in Table 6.3. The spatial consistency between adjacent views are measured using SCM. Smaller SCM values indicate a higher spatial consistency. The best result for every sequence among the two different warping methods is highlighted through bold face type. For all sequences, the proposed warping routine performs better than the state-of-the-art warping method (cf. Table 6.3), where all views are extrapolated from the original cameras. The SCM differences shown in Table 6.3 are quite small. However, considering the fact that the differences between the proposed method and the state-of-the-art approach are present in the uncovered area of the frame, only small variations can be expected since the SCM is determined for the entire image. In Fig. 4.2 (a), for example, the disoccluded areas cover only $\approx 6\%$ of the entire image. Frame-wise objective results are shown in Fig. 6.6. As can bee seen, the out-performance is achieved constantly over time [cf. Fig. 6.6].

6.4 Evaluation of the Hybrid View Synthesis Framework

In Sec. 6.4.1 and 6.4.2, relevant modules of the proposed hybrid view synthesis framework (cf. Ch. 5, P3) are evaluated, i.e. the pre-processing method and the AR texture synthesis method. The complexity of the proposed hybrid view synthesis framework is then assessed in Sec. 6.4.3. The patch size of the patch-based texture synthesis is chosen according to the resolution of the sequences (cf. Sec. 6.2.2 and 6.3.4).

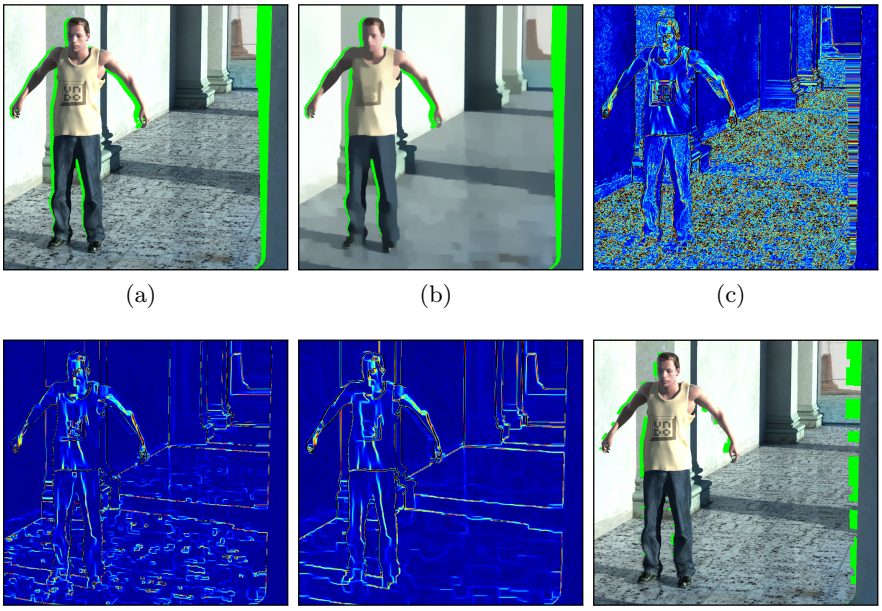


Figure 6.7: Undo Dancer, Frame 17 (virtual view 1 rendered from original view 5), (a) The unknown area Ω is marked in green. (b) The warped pre-processed frame. (c) $X - Y$ gradients of the original warped frame (a). (d) $X - Y$ gradients of the regularly (without sub-sampling), pre-processed warped frame. (e) $X - Y$ gradients of the pre-processed warped frame (b). (f) The dominant structures are synthesized using patch-based texture synthesis. The strength of the isophotes is determined based on (e). Unfilled regions are marked green in (f). © [2016] IEEE.

6.4.1 Evaluation of the Pre-Processing Method

In this thesis, texture separation is utilized to smooth texture patterns while maintaining dominant boundaries. In Fig. 6.7 (c)-(e), gradients in X and Y direction are pictured. Based on the gradients, it is decided which areas are computed using a complex patch-based synthesis method. The gradients in Fig. 6.7 (c) are computed on the warped image with initialized texture areas (without pre-processing). As can be seen in Fig. 6.7 (c), all texture patterns are recognized as strong gradients, and a separation of different classes of textures is impossible. Hence, the texture patterns on the floor are entirely computed with the time-consuming patch-based texture synthesis. Separating the different texture labels from the image filtered in full-resolution [XYXJ12] is still challenging [cf. Fig. 6.7 (d)]. However, using the proposed pre-processing procedure (cf. Sec. 5.2) the main boundaries, i.e. the transitions between homogeneous texture patterns, can be separated reliably [cf. Fig. 6.7 (b), (e), (f)].

6.4.2 Evaluation of the Auto Regressive Parameter Settings

In this section the optimal parameter settings for the AR synthesis framework are determined. In order to cover a wide variety of textures the AR synthesis framework is evaluated on an extended more general texture database [DVGNK99].

6.4.2.1 Data Set and Quality Measures for Assessing the Auto Regressive Parameters

To evaluate the proposed AR algorithm, 20 test images are used: *rough plastic, plaster, rough paper, artificial grass, cork, sponge, lettuce leaf, loofa, limestone, ribbed paper, straw, corduroy, stones, corn husk, white bread, soleirolia, orange peel, peacock feather, tree bark and moss.* The AR model can be used to synthesize a class of texture with a parameter set that is trained in the same texture class. The data set is therefore chosen to cover a broad spectrum of different texture characteristics. All images have a resolution of 180×180 (cropped from the original resolution 640×480) and are publicly available at the Columbia Utrecht Reflectance and Texture Database (CUReT) [DVGNK99]. Furthermore, all tests are conducted with two different hole sizes ($\Omega = 20 \times 20$ and 40×40, i.e. $1,2\%$ and 5% of the image size).

The performance of the proposed AR completion algorithm is assessed with PSNR and SSIM. For the presented results, PSNR is computed locally only for Ω, while SSIM is determined for the entire image as it is not suitable for arbitrarily small regions.

6.4.2.2 Assessment of the Training Area

As mentioned in Sec. 5.3.3, the first step of the AR process is to identify an appropriate training area adjacent to Ω to be filled. Hence, two main parameters need to be assessed: (1) the size of the sub-training area and (2) the stationarity of the training texture.

Assessment of the Size of the Sub-Training Area First, a proper value for size S of the sub-training area needs to be selected. S corresponds to the number of linear equations (cf. Sec. 5.3.4). For this investigation, experiments were conducted for all test images assuming a square training window, i.e. $c_x = c_y$, $s_x = s_y$ ($S = s_x s_y$) at the top-left corner of Ω (cf. Fig. 5.8) without loss of generality. Furthermore, all tests were performed using a causal model [cf. Fig. 5.4 (c) and 5.5] with three different settings of C ($C = (c_x + 1)(c_y + 1) - 1$, cf. Eq. 5.14) and the two hole sizes (Ω) in order to draw reliable conclusions. Note that the PSNR and SSIM results are measured using the original texture from Ω. Fig. 6.8 (a) and (b) depict the average values that were achieved for PSNR and SSIM (blue lines) over the whole test set with $\Omega = 40 \times 40$ depending on the number of linear equations (S). Similar results are observed for $\Omega = 20 \times 20$. Furthermore, the averaged run-time to synthesize the missing region in an image is also taken into consideration and depicted on the y sub-axis [green lines in Fig. 6.8 (a) and (b)]. It can be seen that all configurations of C ($C = 15, 63, 143$) perform similarly although different training

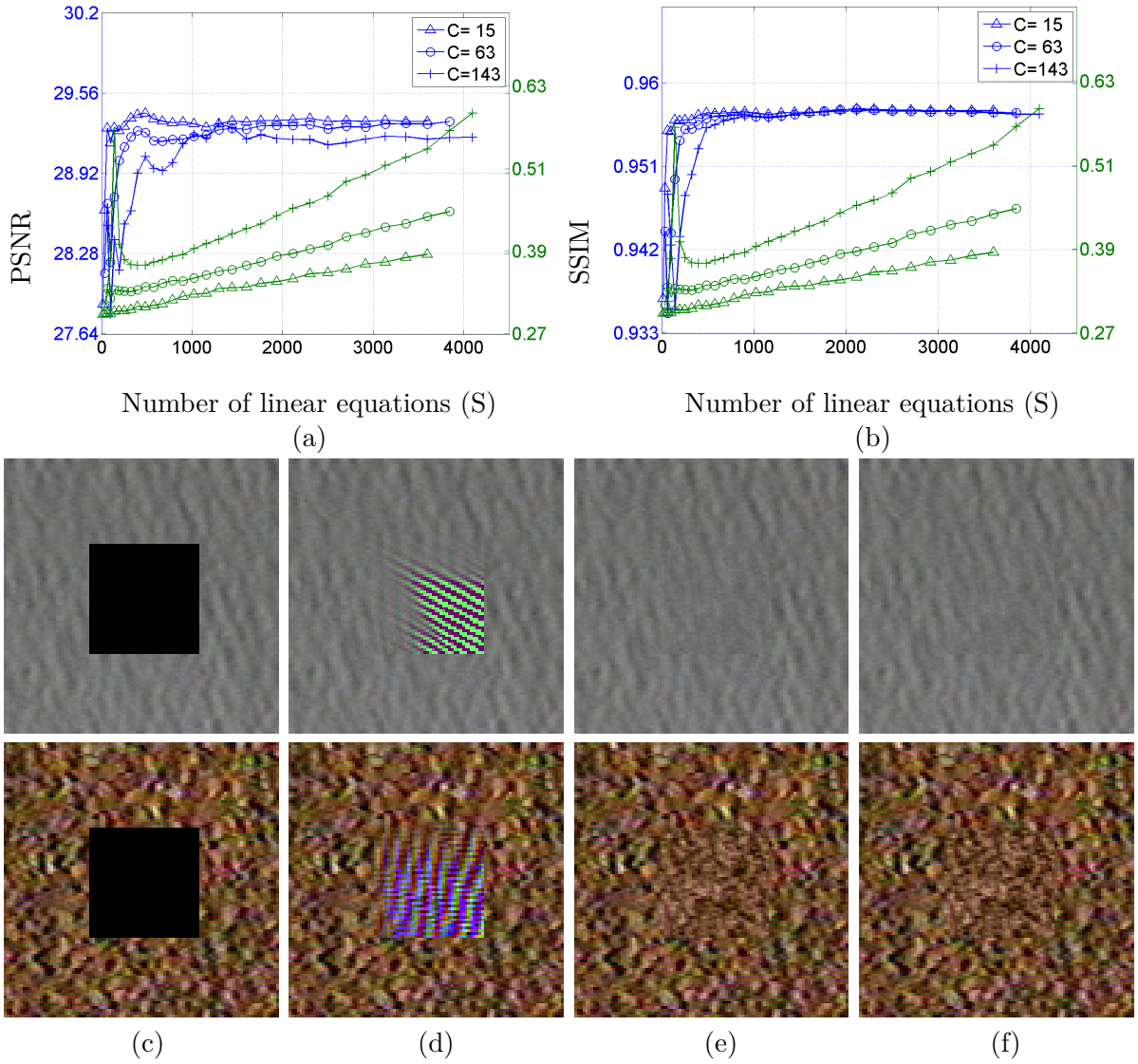
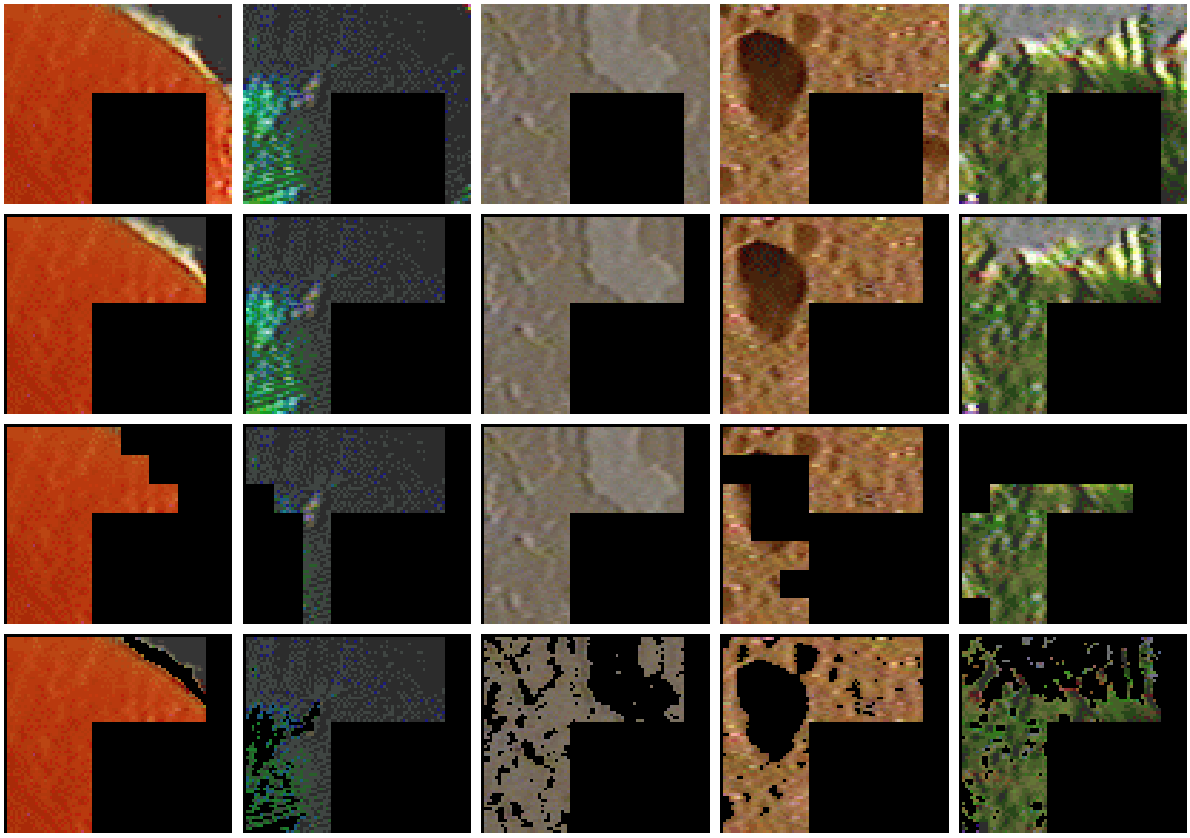


Figure 6.8: Influence of the sub-training area size [number of linear equations (S)] on the filling results. Average values for (a) PSNR vs. run-time (measured in seconds) and (b) SSIM vs. run-time (measured in seconds) over the whole test set with $\Omega = 40 \times 40$. (c-d) Visual differences for the test images *Rough paper* (top) and *Cork* (bottom). (c) Input with $\Omega = 40 \times 40$. Results of the AR texture completion (without post-processing) with $C = 15$ and (d) $S = 36$ (e) $S = 841$ (f) $S = 3025$



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

Figure 6.9: Pruning of the training area for (a) orange peel, (b) peacock feather, (c) lettuce leaf, (d) sponge, and (e) moss. Examples of (top to down) input with $\Omega = 40 \times 40$; the non-stationary training area; the training area after applying the new block-based clustering criterion; and the training area after applying k-means clustering.

sizes (from $S = C + 1$ "to" ~ 5000) are considered. Small values of S yield low PSNR and SSIM values. It appears that S should be larger than 300 for the utilized test set [cf. Fig. 6.8 (a), (b)] to contain a sufficient amount of texture information to fit the model. On the other side, if $S > 1000$, there is a clear saturation of the quality of the final results although the computational costs and thus the run-time progressively increase [green lines in Fig. 6.8 (a), (b)]. In general this means that the training area should be sized to $\approx 53\% - 75\%$ of the hole to be filled. The computational time in Fig. 6.8 increase due to the rising number of linear equations that have to be solved. Furthermore, in terms of visual evaluations, it was found that S should be approximately ten times larger than C in order to have satisfying results. The visual influence of the sub-training size is shown in Fig. 6.8 (d), (e) and (f). Considering the AR test set, $S \approx 800$ ($s_x = s_y = 29$) is a reasonable compromise between complexity and quality. However, for smaller hole sizes, smaller values for S lead also to satisfying results.

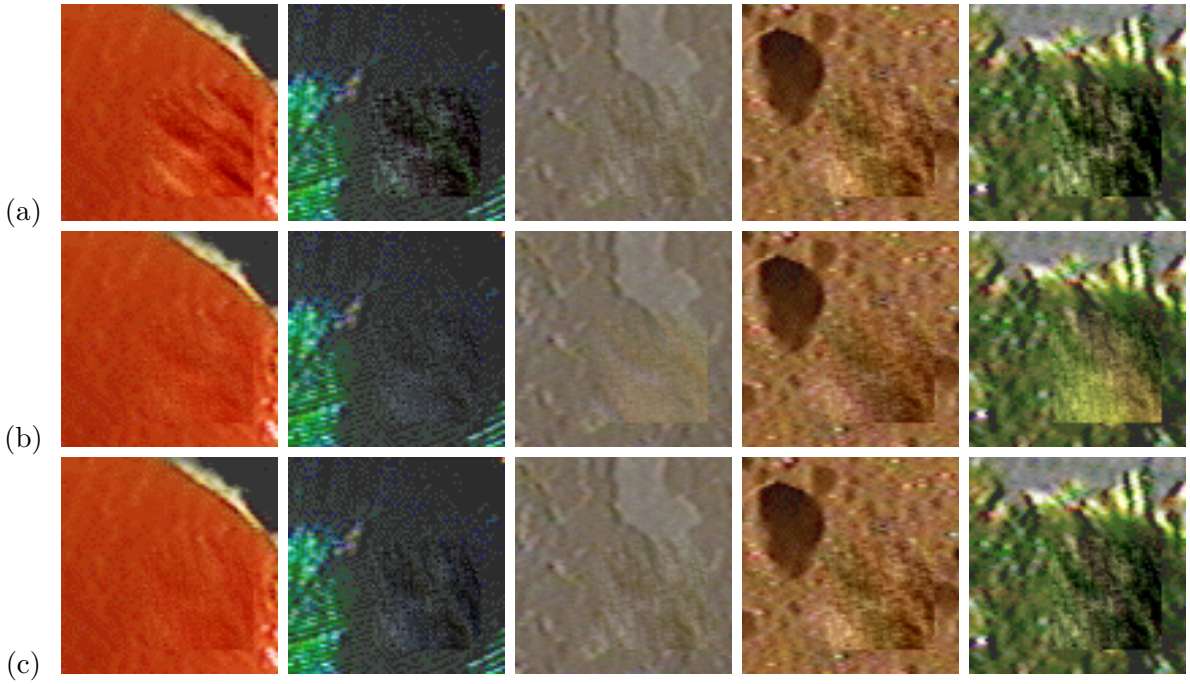


Figure 6.10: Stationarity of the training area. Results using (a) the non-stationary training area. (b) The results achieved using k-means clustering and (c) using the block-based clustering to detect an appropriate training area. (cf. Fig. 6.9). Note that all results are generated without a post-processing step.

6.4.2.3 Pruning of the Training Area

In this section, the proposed method to find the appropriate training area is evaluated (cf. Sec. 5.3.3). The impact of the content of the training area on the completion results is an important investigation, which highly depends on the texture characteristics of the considered image. In general, if the texture information in the training area is stationary and correlates with the unknown texture in Ω, the missing texture will be more likely to be completed well. If the training area is not well chosen, the method proposed in Sec. 5.3.3 can be applied. Fig. 6.9 illustrates the effect of processing the training area using the new block-based stationarity criterion in comparison to the k-means clustering approach. It can be seen that both methods can successfully recognize and remove the unstationary texture locations. K-means works sample wise and is an iterative approach [Bis95], whereas the proposed method operates block wise and non-iteratively. Therefore, the excluded regions of the final results have irregular boundaries after applying the k-means method (cf. Fig. 6.9, bottom row). An essential disadvantage of k-means consists in the problem of finding the global minimum, since it tends to converge towards a local minimum. Commonly, this problem can be solved by a careful choice of starting conditions. Using several replicates with random starting points typically results in a solution that is a global minimum. In run-time evaluations, it was found that the new method is approximately three times faster

Seq.	P3	Daribo et al. [DS11]	Ahn et al. [AK13]
S1	337 sec	783 sec	1685 sec
S2	522 sec	1740 sec	3480 sec
S3	279 sec	1992 sec	845 sec
S4	155 sec	738 sec	281 sec
S5	56 sec	186 sec	103 sec

Table 6.5: Complexity assessment: Run-times in seconds (sec) of the proposed hybrid approach, Daribo et al. [DS11] and Ahn et al. [AK13]. © [2016] IEEE.

than k-means clustering. In a set of different simulations, it was found that the proposed criterion works well within the following parameter range $10 \leqslant t_\mu \leqslant 20$ and $10 \leqslant t_\theta \leqslant 20$. Furthermore, the computation of the training area with k-means (two clusters) was optimized. After applying the clustering procedure, small blobs (smaller than 20 samples) were removed. Hence, only large texture segments were kept. Fig. 6.10 illustrates the influence of the stationarity criteria on the filling results.Note that all results are generated with $C = 15$, $t_\mu = t_\theta = 20$ and without post-processing. It can be seen that the quality of the completion results increases when an appropriate texture [cf. Fig. 6.10 (d) vs. (b),(c)] is selected for the training process. Hence, it can be concluded, that pruning the training area is recommended when the texture consists of a complex pattern.

6.4.3 Hybrid vs. Patch-based Texture Synthesis

In this section, the complexity of the novel hybrid view synthesis framework is compared with the entirely patched-based texture synthesis methods proposed by Daribo et al. [DS11] (M3) and Ahn et al. [AK13] (M5). A PC with an Intel Xeon CPU 3.33 GHz and 24 GB RAM was utilized in all experiments presented in Sec. 6.4. The mean run-times in seconds of the proposed method and patch-based view synthesis methods on the state-of-the-art are are given in Tab. 6.5. Please note that all frameworks are prototypes developed in MATLAB. An implementation of the frameworks in C/C++ can significantly accelerate all methods. However, MATLAB implementations have been tested since the reference methods (M3 and M5) where also provided as MATLAB code. The framework proposed by Daribo et al. [DS11] utilizes the same patch search routine as P3. Hence, the run-time improvements by P3 can be evaluated directly. As can be seen in Tab. 6.5 (second and third row), the new hybrid texture synthesis method requires less processing time, i.e. 14%-54%, of pure patch-based view synthesis frameworks. Additionally, in Fig. 6.12, 6.13 and Tab. 6.6, 6.7 it is shown that the proposed hybrid view synthesis method provides a similar or better visual and objective outcome compared to entirely patch-based view synthesis methods.

Seq.	Cam.	PSNR							
		M1	M2	M3	M4	M5	P1	P2	P3
S1	8 → 10	19.24	**30.24**	27.22	25.87	24.99	**29.58**	28.42	**29.12**
S1	8 → 9	23.11	**33.30**	30.28	29.48	28.66	**33.06**	31.21	**32.45**
S1	10 → 8	23.97	**28.86**	27.44	25.84	28.32	**29.16**	28.24	**28.49**
S1	10 → 9	28.77	**31.97**	30.29	29.39	31.46	**32.39**	31.93	**32.11**
S2	6 → 8	27.32	27.52	27.57	27.57	28.25	**28.27**	**28.50**	**28.27**
S2	6 → 7	26.17	26.17	26.53	26.61	**26.64**	**26.71**	**26.68**	26.60
S2	8 → 6	25.97	26.13	27.36	27.33	27.52	**27.90**	**27.73**	**27.76**
S2	8 → 7	**28.48**	**28.41**	27.42	27.15	27.45	**27.59**	27.51	27.54
S3	4 → 6	23.45	23.19	22.79	21.83	20.49	**23.75**	**23.56**	**23.63**
S3	4 → 5	27.65	**28.66**	26.58	26.31	23.37	**28.51**	28.08	**28.76**
S3	6 → 4	**25.83**	26.22	25.00	23.47	24.99	**26.95**	25.43	**27.21**
S3	6 → 5	28.77	**28.93**	26.93	26.55	27.34	**29.25**	27.91	**29.62**
S4	5 → 3	**35.30**	34.39	27.78	29.83	28.29	**36.54**	35.19	**36.11**
S4	5 → 4	**37.46**	36.67	28.41	33.30	28.81	**38.38**	37.26	**38.48**
S4	5 → 6	34.17	**35.08**	27.74	31.84	27.76	**35.88**	34.58	**36.06**
S4	5 → 7	31.80	**31.57**	26.59	28.13	26.35	**33.05**	31.32	**32.43**
Avg. S1-S4		27.96	**29.83**	27.24	27.53	26.91	**30.43**	29.59	**30.29**
S5	5 → 1	31.34	30.51	29.46	26.84	**31.98**	-	**32.74**	**32.63**
S5	5 → 9	29.80	28.66	26.00	26.16	**30.79**	-	**31.25**	**31.77**
S5	3 → 1	32.56	33.32	32.00	29.81	**34.98**	-	**36.00**	**35.68**
S6	5 → 1	32.91	35.61	33.44	32.01	**36.13**	-	**37.99**	**35.65**
S6	5 → 9	34.88	35.64	33.29	31.84	**36.29**	-	**38.24**	**35.60**
S6	3 → 1	35.12	37.21	35.30	34.76	**38.05**	-	**39.71**	**37.42**
S7	5 → 4	32.57	**32.74**	31.93	31.66	32.41	-	**32.75**	**32.84**
S7	6 → 7	32.94	**33.72**	32.74	31.93	33.23	-	**33.76**	**33.64**
S8	3 → 5	28.14	28.98	28.80	27.46	**29.33**	-	**29.65**	**30.05**
S8	3 → 4	32.33	**33.74**	31.97	31.26	32.24	-	**32.59**	**33.52**
S8	5 → 3	29.08	**31.16**	29.00	28.52	29.64	-	**31.11**	**31.16**
S8	5 → 4	31.95	**34.33**	31.91	31.63	32.62	-	**34.46**	**34.45**
Avg. S5-S8		31.96	32.96	31.32	30.32	**33.14**	-	**34.18**	**34.11**

Table 6.6: PSNR Results. The best result is highlighted through bold face type colored **red**. The second best result is highlighted through bold face type colored **blue** and third best result is highlighted through bold face type colored **green**.

6.5 Overall Experimental Results

Given the experiments conducted in the previous sections, optimized parameter settings have been derived and summarized in Tab. 6.1, 6.2 and 6.4. The objective results given in Table 6.6 and 6.7 correspond to the mean PSNR and SSIM over all pictures of the rendered sequences. The projection configuration (original and the virtual camera) is given in the second column (Cam.) in Table 6.6, 6.7 and 6.8. Here, "3 → 1" means: virtual camera 1 is rendered from the original camera 3. The average results for S1-S4 and S5-S8 for all methods are given in the rows "Avg. S1-S4" and "Avg. S5-S8", respectively. The best result for each sequence is highlighted through bold face type colored **red**. The second best result for each sequence is highlighted through bold face type colored **blue** and third best result for each sequence is highlighted through bold face type colored **green**.

For S7 the distance between two original cameras corresponds to to a large baseline (≈ double eye distance). Hence, only two virtual camera positions are computed.

6.5.1 Objective Results

On average the proposed approaches perform better than state-of-the-art methods in terms of PSNR and SSIM (cf. Table 6.6 and 6.7).

Seq.	Cam.	SSIM							
		M1	M2	M3	M4	M5	P1	P2	P3
S1	8 → 10	0.9412	**0.9651**	0.9473	0.9046	0.9469	0.9598	**0.9599**	**0.9626**
S1	8 → 9	0.9666	**0.9803**	0.9721	0.9570	0.9734	0.9790	**0.9794**	**0.9806**
S1	10 → 8	0.9524	**0.9642**	0.9524	0.9152	0.9626	0.9603	**0.9634**	**0.9664**
S1	10 → 9	0.9745	0.9785	0.9713	0.9588	**0.9793**	0.9777	**0.9796**	**0.9812**
S2	6 → 8	0.9107	0.9128	0.9283	0.9211	**0.9335**	**0.9352**	**0.9351**	0.9334
S2	6 → 7	0.8929	0.8933	0.9086	0.9078	0.9104	**0.9113**	**0.9110**	**0.9105**
S2	8 → 6	0.8898	0.8907	0.9372	0.9292	**0.9416**	**0.9422**	**0.9421**	0.9414
S2	8 → 7	**0.9598**	**0.9592**	0.9401	0.9334	0.9424	0.9424	0.9425	**0.9425**
S3	4 → 6	**0.8971**	**0.9060**	0.8871	0.8684	0.8865	0.8881	0.8936	**0.9051**
S3	4 → 5	**0.9598**	**0.9659**	0.9543	0.9424	0.9526	0.9544	0.9570	**0.9645**
S3	6 → 4	0.9150	**0.9220**	0.9054	0.8812	0.9057	**0.9171**	0.9062	**0.9278**
S3	6 → 5	**0.9622**	**0.9653**	0.9562	0.9423	0.9582	0.9609	0.9568	**0.9667**
S4	5 → 3	0.9885	0.9861	0.9775	0.9516	0.9802	**0.9911**	**0.9887**	**0.9905**
S4	5 → 4	**0.9936**	0.9922	0.9830	0.9694	0.9854	**0.9948**	0.9932	**0.9948**
S4	5 → 6	**0.9938**	0.9923	0.9810	0.9681	0.9841	**0.9945**	0.9921	**0.9940**
S4	5 → 7	**0.9861**	0.9830	0.9684	0.9438	0.9724	**0.9870**	0.9820	**0.9845**
Avg. S1-S4		0.9490	0.9536	0.9481	0.9309	0.9510	**0.9560**	**0.9552**	**0.9592**
S5	5 → 1	0.9793	0.9742	0.9700	0.9556	**0.9809**	-	**0.9832**	**0.9821**
S5	5 → 9	0.9814	0.9777	0.9651	0.9576	**0.9850**	-	**0.9861**	**0.9867**
S5	3 → 1	0.9886	0.9886	0.9845	0.9794	**0.9921**	-	**0.9939**	**0.9930**
S6	5 → 1	0.9884	0.9914	0.9830	0.9774	**0.9924**	-	**0.9945**	**0.9894**
S6	5 → 9	0.9899	**0.9909**	0.9815	0.9753	**0.9921**	-	**0.9944**	0.9892
S6	3 → 1	0.9943	**0.9954**	0.9895	0.9876	**0.9955**	-	**0.9968**	0.9935
S7	5 → 4	0.9570	**0.9575**	0.9542	0.9511	**0.9579**	-	0.9567	**0.9591**
S7	6 → 7	0.9641	**0.9659**	0.9607	0.9559	0.9632	-	**0.9649**	**0.9646**
S8	3 → 5	0.9447	0.9468	0.9452	0.9284	**0.9505**	-	**0.9532**	**0.9560**
S8	3 → 4	0.9765	**0.9787**	0.9732	0.9677	0.9754	-	**0.9765**	**0.9786**
S8	5 → 3	0.9611	**0.9680**	0.9496	0.9351	0.9611	-	**0.9680**	**0.9682**
S8	5 → 4	0.9799	**0.9825**	0.9742	0.9631	0.9796	-	**0.9826**	**0.9828**
Avg. S5-S8		0.9754	0.9765	0.9692	0.9612	**0.9771**	-	**0.9792**	**0.9786**

Table 6.7: SSIM Results. The best result is highlighted through bold face type colored **red**. The second best result is highlighted through bold face type colored **blue** and third best result is highlighted through bold face type colored **green**.

Sequences with Static Background For sequences with static background, P1 performs best in terms of PSNR and SSIM on average. P3 shows improvements concerning SSIM and PSNR compared to M1-M5, too. For the majority of S1-S4, P2 performs worse than P1 and P3. The reason for this is that the registration tool does not always work reliably. Local motion in the background or a false foreground separation leads to small transformations of the whole image and thus to poor objective results. Nevertheless, P2 reached higher gains then P1 and P3 in terms of PSNR for S2 "6 → 8". For some virtual views of S1, M2 achieves the highest gains in terms of PSNR (S1 "8 → 10", S2 "8 → 9") and SSIM (S1 "8 → 10"). This is due to the fact that the partly smooth background textures in these sequences can be synthesized reliably with classical image inpainting methods (M2). For the sequence "Newspaper" (S3) P1 and P2 perform worse

Seq.	Cam.	FDF							
		M1	M2	M3	M4	M5	P1	P2	P3
S1	8 → 10	0.87	**0.63**	6.34	**0.57**	15.32	1.34	**0.42**	1.41
S1	8 → 9	0.69	**0.46**	6.42	**0.56**	10.17	1.09	**0.65**	1.32
S1	10 → 8	3.11	**1.23**	5.06	**1.42**	4.39	1.65	1.46	**1.20**
S1	10 → 9	3.61	1.41	4.53	**1.18**	3.56	1.75	**1.31**	**0.96**
S2	6 → 8	**0.52**	1.53	11.84	1.15	3.64	**0.59**	**0.90**	1.33
S2	6 → 7	**0.51**	1.41	11.61	1.31	2.74	**0.93**	0.98	**0.82**
S2	8 → 6	**0.35**	1.79	12.14	1.77	4.56	**0.95**	**0.53**	2.41
S2	8 → 7	**0.52**	2.09	12.08	2.37	4.65	**0.88**	**0.64**	2.46
S3	4 → 6	0.80	**0.29**	15.43	**0.48**	23.53	**0.74**	1.71	3.38
S3	4 → 5	0.82	**0.23**	13.44	**0.36**	21.92	**0.74**	2.15	3.15
S3	6 → 4	**0.78**	**1.21**	11.84	1.51	15.15	**1.09**	1.95	2.66
S3	6 → 5	**0.95**	**0.89**	10.93	1.14	10.49	**0.91**	1.69	1.84
S4	5 → 3	**0.36**	2.24	6.02	**2.20**	4.20	**0.45**	3.47	2.44
S4	5 → 4	**0.46**	2.04	4.37	2.37	3.09	**0.58**	2.77	**1.66**
S4	5 → 6	**0.39**	3.22	10.65	6.17	6.73	**0.64**	6.01	**2.66**
S4	5 → 7	**0.35**	**3.10**	12.30	4.76	10.49	**0.50**	7.47	3.65
Avg. S1-S4		**0.94**	**1.48**	9.68	1.83	9.03	**0.92**	2.13	2.08
S5	5 → 1	2.73	**1.45**	6.40	**1.20**	4.78	-	2.80	**1.65**
S5	5 → 9	**3.19**	**1.86**	5.72	3.23	5.96	-	4.40	**2.48**
S5	3 → 1	4.56	**1.43**	5.93	**1.19**	4.10	-	2.02	**1.58**
S6	5 → 1	3.65	2.79	3.74	**1.51**	**1.84**	-	**1.18**	3.43
S6	5 → 9	3.36	1.99	3.18	**1.49**	**1.71**	-	**1.19**	3.17
S6	3 → 1	3.99	3.13	4.21	**1.67**	**2.26**	-	**1.01**	3.30
S7	5 → 4	1.90	**1.54**	3.36	**1.04**	3.47	-	**1.55**	1.64
S7	6 → 7	1.21	1.24	2.85	**0.63**	2.59	-	**0.96**	**0.83**
S8	3 → 5	1.52	**1.44**	8.86	**1.28**	7.01	-	**1.39**	1.63
S8	3 → 4	1.32	**0.66**	7.40	1.16	5.78	-	**1.30**	**1.20**
S8	5 → 3	**1.32**	**0.66**	9.75	1.76	8.19	-	1.90	**1.38**
S8	5 → 4	1.10	**0.59**	8.81	1.33	5.80	-	**1.05**	**1.02**
Avg. S5-S8		2.47	**1.56**	5.85	**1.45**	4.45	-	**1.72**	1.94

Table 6.8: FDF Results. The best result is highlighted through bold face type colored **red**. The second best result is highlighted through bold face type colored **blue** and third best result is highlighted through bold face type colored **green**.

than P3, M2 and/or M1 for “4 → 5", “6 → 4" and “6 → 5". As all of the modules of P1 and P2 rely on the depth map and the depth map of “Newspaper” (S3) is particularly unreliable, visual and objective losses occur for P1 and P2. On the other hand, P3 outperforms state-of-the-art methods for these virtual views.

Sequences with Global Background Motion For sequences with global background motion (S5-S8), both P2 and P3 perform better than the state-of-the-art in terms of PSNR and SSIM. However, on average, P2 achieves slightly better results than P3. For the sequences “Balloons” (S8), P3 performs better than P2. The reason for this is that the registration tool does not work reliably for this sequence due to local movement in the background that is different from the global background motion. Furthermore, for S8, “3 → 4” M2 achieves better results than P2 and

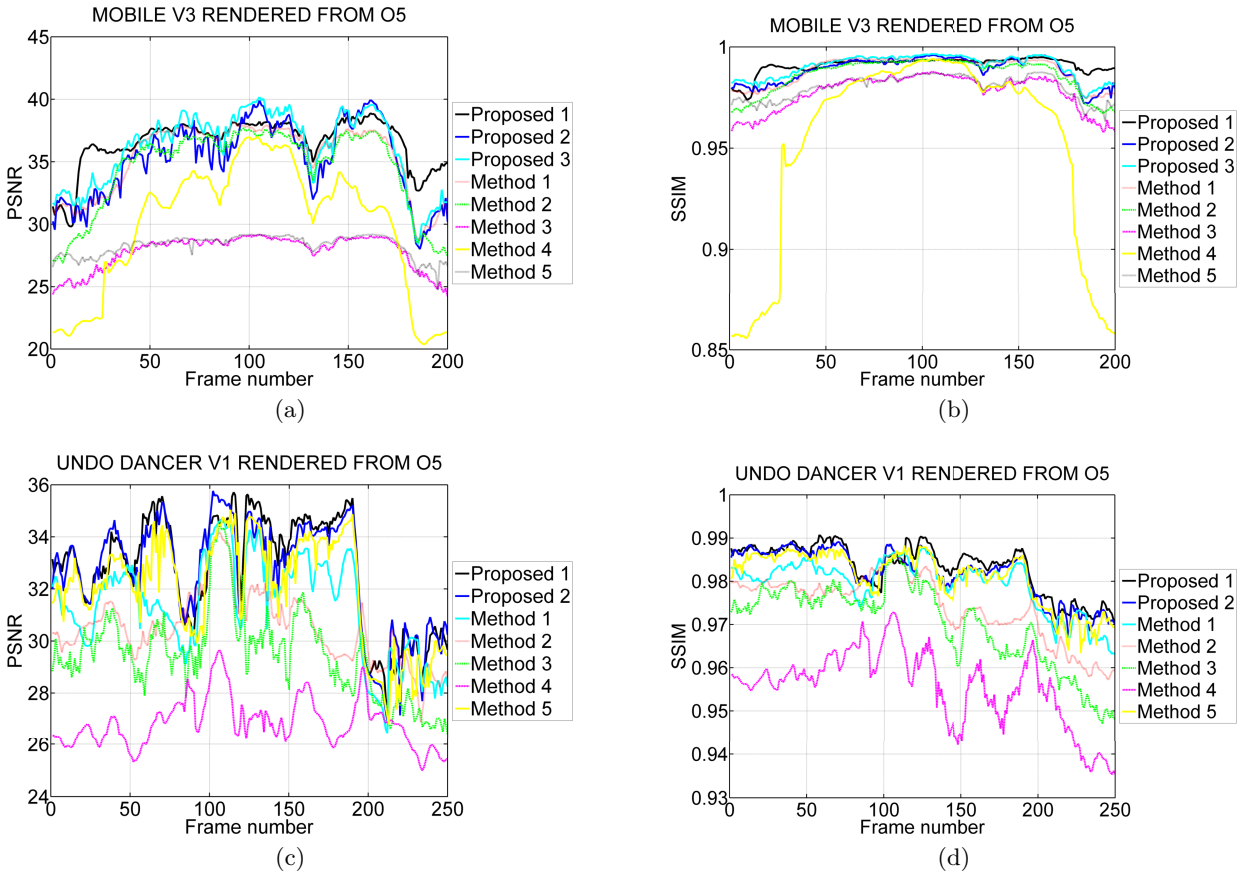


Figure 6.11: Frame-wise objective results for "Mobile" (S4) (virtual camera 3 from original camera 5). (a) PSNR Results. (b) SSIM Results. Objective Results for "Undo Dancer" (S5) (virtual camera 1 from original camera 5). (c) PSNR Results. (d) SSIM Results.

P3 for PSNR and SSIM. This is due to the fact, that S8 partly show smooth textures which can be reliably synthesized using classical inpainting methods. For the sequence "Poznan Hall2" S7, "6 → 7" M2 outperforms both P2 and P3 in terms SSIM and P3 in terms of PSNR. The reason for this is that the depth map is partly unreliable and the background texture is mostly smooth

Fig. 6.11 shows frame-wise objective results for "Mobile" S4, "6 → 7" and for "Undo Dancer" S5, "5 → 1".

6.5.2 Temporal Quality Evaluation

In this section, the temporal consistency is evaluated. The unknown areas in P1 and P2 are filled from a background sprite or from registered temporally surrounding frames. P3 only considers the previous frame in the patch matching criterion (cf. Sec. 5.3.1). Hence, temporal correlations are considered in all of the proposed frameworks P1-P3.

The human eyes are very sensitive to temporal inconsistencies, i.e. flickering artifacts. Therefore, the flickering is measured using the Frame Differential Flicker (FDF) metric proposed in

[SJ11]. This metric is computed as follows:

$$\kappa_n = \frac{1}{|\Omega|} \sum_{(x,y)\in\Omega} |F_n(x,y) - F_{n-1}(x,y)|, \tag{6.4}$$

where $|\Omega|$ represents the total number of hole pixels in a frame. The κ_n value measures the average changes of the inter-frame samples in the hole regions of frame n. The average flickering for the whole sequence is then computed as follows:

$$\kappa = \frac{1}{N} \sum_{n=2}^{N} \kappa_n, \tag{6.5}$$

where κ measures the absolute amount of flickering in the disoccluded areas of a video. Since different reference views also exhibit different a priori flickering (noise, compression artifacts, ...) the κ value is furthermore normalized using the original view at the same position as the virtual view. Then, the final FDF can be computed as follows:

$$\text{FDF} = |\kappa_{synth} - \kappa_{orig}|, \tag{6.6}$$

where κ_{synth} and κ_{orig} represent the κ result of the synthesized view and the original view at the same spatial positions. Note that lower values for the FDF metric indicate fewer flickering artifacts.

The FDF results are shown in Table 6.8. All proposed methods (P1-P3) perform better than the state-of-of-the-art patch-based texture synthesis methods M3 and M5. This is due to the fact that the proposed frameworks consider temporal correlations in the filling process, while M3 and M5 synthesize the new textures frame-by-frame. However, P3 considers previous frames only in the patch searching routine (cf. Sec. 5.3.1), thus P3 shows a higher amount of flickering artifacts than P1 and P2. Since M1 and M2 mostly compute smooth new textures, flickering artifacts are reduced. Therefore, M1 and M2 often provide the least amount of flickering in the synthesized regions. However, the proposed methods are often close to these results. Furthermore, for S1, "8 $\rightarrow$ 10" and for all virtual views of S6, P2 shows the lowest amount of flickering artifacts. For S1, "10 $\rightarrow$ 8" and "10 $\rightarrow$ 9", P3 shows the lowest amount of flickering artifacts. For the sequence S4 the proposed method P2 has low temporal correlations between the inter-frame samples. Here, the registration results of neighboring frames are often not considered for updating the synthesized areas due to an unreliable estimated transformation matrix (cf. Sec. 4.4.3). This leads to partially frame-by-frame filling. Hence, the FDF outcome for S4 is only slightly better than the FDF results of M3 and M5.

6.5.3 Visual Results

Visual results for some sub-frames are shown in Fig. 6.12 and 6.13. The first column in Fig. 6.12 shows results for S1 (“10 → 8", frame 10), the second column for S2 (“8 → 6", frame 106), the third column for S3 (“6 → 4", frame 1), and the last column for S4 (“5 → 7", frame 32). The first column in Fig. 6.13 shows results for S5 (“5 → 1", frame 181), the second column for S6 (“5 → 9", frame 145), the third column for S7 (“5 → 4", frame 177), and the last column for S8 (“5 → 3", frame 52).

In Fig. 6.12 and 6.13 (a), the warped sub-frames with the unknown areas marked green are presented. Fig. 6.12 (b)-(f) shows the synthesis results for M1-M5. The results of P1-P3 are shown in Fig. 6.12 (g)-(i) while Fig. 6.13 (g)-(h) show the results of P2 and P3. Finally, the original sub-frames are depicted in Fig. 6.12 (j) and 6.13 (i)

M1 and M2 [cf. Fig. 6.12 and 6.13 (b) and (c)] smooth the synthesized textures. This is visually annoying especially for sequences with complex texture patterns in the current background (cf. S1, S2, S4-S6). On the other hand, the proposed methods, i.e. P1-P3, can maintain these texture patterns. M3 [cf. Fig. 6.12 and 6.13 (d)] does not use an appropriate handling of the filling routine. Hence, foreground texture is inserted into the unknown image regions. Since the proposed methods consider the depth values and furthermore steer the filling routing from the background to the foreground areas, only background textures are used to occlude the holes. M4 smooths the depth maps in order to prevent holes in the virtual frame. Therefore, strong distortions can appear in the results [cf. Fig. 6.12 and 6.13 (f)], while such distortions are prevented using P1-P3 [cf. Fig. 6.12 (g)-(i) and 6.13 (g),(h)]. The results of M5 are visually plausible, especially for the very complex sequences. However, since M5 does not use appropriate post-processing methods, blocking artifacts can appear (cf. Fig. 6.12 and 6.13 (f), S1, S3, S7 and S8).

P1 and P2 utilize the texture information of temporally neighboring frames. Therefore, uncovered areas that require information from other time instances can be recovered reliably. On the other hand, P3 utilizes only the texture from the frame to be filled. Hence, the subjective results can be slightly worse (cf. Fig. 6.12, S1, the poster in the background, S2, the stairs and Fig. 6.13, S6, the house in the background).

Figure 6.12: Subjective results: static sequences (S1-S4, left to right). (a) Sub-frame with holes marked green. (b) Results of VSRS-alpha-Gist [LH11] (M1) (c) Results of VSRS-alpha-Etri [BKY+11] (M2) (d) Results of Daribo et al. [DS11] (M3) (e) Results of Daribo et al. [DSF+12] (M4) (f) Results of Ahn et al. [AK13] (M5) (g) Results of P1 (h) Results of P2 (i) Results of P3 (j) Original sub-frame.

Figure 6.13: Subjective Results: sequences with global motions (S5-S8, left to right). (a) Sub-frame with holes marked green. (b) Results of VSRS-alpha-Gist [LH11] (M1) (c) Results of VSRS-alpha-Etri [BKY+11] (M2) (d) Results of Daribo et al. [DS11] (M3) (e) Results of Daribo et al. [DSF+12] (M4) (f) Results of Ahn et al. [AK13] (M5) (g) Results of P2 (h) Results of P3 (i) Original sub-frame.

7 Conclusion and Future Work

7.1 Conclusion

Autostereoscopic multi-view displays provide 3-D depth perception without the need to wear additional glasses by showing many different views of a scene from slightly different viewpoints simultaneously. Nevertheless, only a limited number of original views can be recorded, stored and transmitted. Consequently, the need to render additional virtual views arises, in order to support autostereoscopic multi-view displays. DIBR is an appropriate technology for synthesizing virtual images at a slightly different view perspective, using a textured image and its associated depth map. A critical problem is that regions occluded by foreground objects in original views may become visible in synthesized views. This is particularly problematic in the case of extrapolation beyond the baseline of the original views, as there is no additional information from another original camera.

In this thesis, new methods to fill disoccluded areas in a DIBR framework are proposed, especially for the extrapolation scenario. The algorithms are designed to compensate large baseline extensions between the original and the virtual view and generate spatial-temporal consistent rendering results for 3-D sequences. All of the presented methods utilize advanced texture synthesis methods to compute unknown image regions.

The first proposed approach (P1) utilizes a static background sprite to recover background textures using image information from a causal picture neighborhood. Unknown texture regions that cannot be filled from the background are roughly estimated and subsequently refined using novel advanced texture synthesis. The proposed new patch-based texture synthesis method utilizes both depth and luminance information to determine the filling order and to find the best continuation patches. It could be shown that the synthesized results are fundamentally improved, if the texture is initialized since garbage-growing due to misplaced textures is consequently diminished. Finally, a novel post-processing module based on cloning methods is applied, in order to conceal transitions between the synthesized and the original textures. The second framework (P2) recovers missing background texture from temporally surrounding frames. In comparison to P1, P2 can also be applied for videos which contain global motion. To compensate the global background motion, image registration is incorporated into the framework. The frames are furthermore considered as a GOP-structure for the computation. Hereby, a new processing order derived from the GOP structure in video coding is presented. Disocclusions that can not be filled from the known texture are synthesized using the patch-based completion method proposed in P1. Furthermore, a new

warping routine is presented which improves the spatial consistency between adjacent virtual-virtual or virtual-original views. The methods proposed in P1 and P2 synthesize missing regions in the virtual views in a visual plausible manner using patch-based non-parametric texture synthesis. However, these techniques are very complex and therefore time-consuming. Hence, the third method (P3) describes a novel hybrid synthesis framework that overcomes this complexity issue while providing a similar or even better visual outcome. First, a new patch-based synthesis method is applied to synthesize the object-texture boundaries alone. Then, a fast parametric method is utilized to recover the remaining texture segments. For this method, a new way to find an appropriate texture region to train the parameters of the model is presented. It can be shown that visual improvements become visible, when gross instationarities are discarded. Furthermore, a new criteria to evaluate the parametrically synthesized textures is proposed. Finally, a post-processing step is applied to improve the background-foreground transitions in the virtual view. Due to the hybrid synthesis framework, P3 method requires less processing time, i.e. only 14%-54% of pure patch-based view synthesis methods.

In detailed experiments, it could be shown that all of the methods in average outperform state-of-the-art approaches. High objective and subjective gains could be achieved. However, since P1 was designed for sequences with static background and P2 for sequences with global background motion, the highest gains for P1 and P2 are especially shown for their respective application scenario. The outcome of P3 is on average slightly worse than for P1 and P2. This is due to the fact, that large texture portions of temporally surrounding frames are not considered during the filling routine. However, all of the proposed methods (P1-P3) provide a better temporal consistency in the synthesized views than state-of-the-art patch-based texture synthesis methods. Additionally, it was figured out in the experiments that the highest gains can be achieved for sequences with complex or very complex background textures. For videos with simple or uniform background areas, recent approaches also provide promising results due to the fact that smooth background textures in such sequences can be synthesized reliably with classical inpainting methods.

7.2 Future Work

In future work items the proposed hybrid synthesis approach (P3) could be incorporated into the frameworks of P1 and P2 which also consider neighboring textures. Since P1 and P2 are designed for different video contents, i.e. static backgrounds and backgrounds with global motion, a decision criterion could be included, to decide whether to use P1 or P2 for synthesizing uncovered areas.

In [BPLC+11a] Bosc et al. show that human observers sometimes rate the quality of synthesized virtual views different than the objective evaluation measures suggest. Furthermore, in [KMNN13, KMMNN13] Köppel et al. show that the complexity of the view synthesis framework can be reduced while providing a similar quality, if the depth map is partially smoothed. Hence, small holes are closed, while the size of large holes can be reduced. However, the outcome can not

be evaluated with classical full-reference metrics such as PSNR and SSIM. Therefore, new full-reference or no-reference metrics should be investigated in order to evaluate the quality of texture synthesis approaches with slight image distortions according to the human visual system.

Appendix

A 3-D Test Sequences

In Tab. A.1, the characteristics of the utilized MVD data set, provided by MPEG, are presented. The data set consists of real world and computer graphic scenes captured by a stationary or a dynamic camera. The depth maps of the real world scenes are estimated while the depth maps of the computer graphic scenes are computer generated (Ground truth). The texture in the background varies from simple to very complex. One frame of each sequence is shown in Fig. A.1 and Fig. A.2.

Name	**Resolution**	**Computed Frames**	**Scene Type**	**Depth Type**	**Camera Motion**	**Background Texture**
Book Arrival	1024 × 768	1-100	Real world	Estimated	Stationary	Simple
Lovebird1	1024 × 768	1-150	Real world	Estimated	Stationary	Complex
Newspaper	1024 × 768	1-200	Real world	Estimated	Stationary	Complex
Mobile	720 × 540	1-200	Computer Graphics	Ground truth	Stationary	Very Complex
Undo Dancer	1920 × 1088	1-250	Computer Graphics	Ground truth	Dynamic	Very Complex
Ghost Town Fly	1920 × 1088	1-250	Computer Graphics	Ground truth	Dynamic	Complex
Poznan Hall2	1920 × 1088	1-198	Real world	Estimated	Dynamic	Simple
Balloons	1024 × 768	1-300	Real world	Estimated	Dynamic	Complex

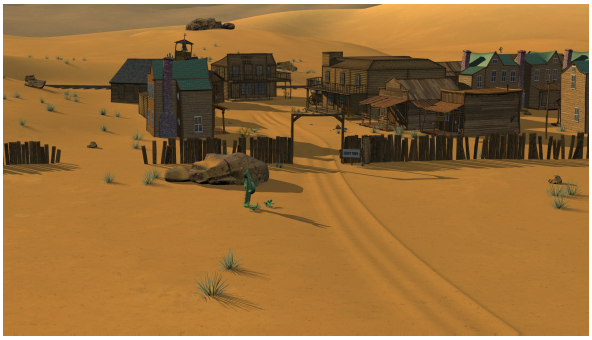
Table A.1: Characteristics of the MVD data set.

(a) (b)

(c) (d)

Figure A.1: Test sequences with static background. (a) “Newspaper”, camera 4, frame 29. (b) “Book Arrival”, camera 10, frame 27. (c) “Lovebird1”, camera 6, frame 18. (d) “Mobile”, camera 5, frame 20

(a)



(b)

(c)

(d)

Figure A.2: Test sequences with global background motion. (a) “Balloons”, camera: 3, frame: 13. (b) “Poznan Hall2”, camera: 5, frame: 60. (c) “Ghost Town Fly”, camera: 5, frame: 91. (d) “Undo Dancer”, camera: 5, frame:21

Bibliography

[AK13] I. Ahn and C. Kim. A novel depth-based virtual view synthesis method for free viewpoint video. *IEEE Trans. on Broadcasting*, 59(4):614–626, 2013.

[Ali13] Alioscopy 3D. http://www.alioscopy.com, 2013.

[AS07] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. In *Proc. Conference and Exhibtion on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2007.

[Ash01] M. Ashikhmin. Synthesizing natural textures. In *Proc. Conference and Exhibtion on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 217–226. ACM, 2001.

[BBC+01] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera. Filling-in by joint interpolation of vector fields and gray levels. *IEEE Trans. on Image Processing*, 10(8):1200–1211, August 2001.

[BBCS10] A. Bugeau, M. Bertalmio, V. Caselles, and G. Sapiro. A comprehensive framework for image inpainting. *IEEE Trans. on Image Processing*, 19(10):2634–2645, 2010.

[BCMS12] M. Bertalmio, S. Caselles, S. Masnou, and G. Sapiro. *Encyclopedia of Computer Vision*, chapter Inpainting, pages 1–22. Springer, 2012.

[Bha12] A. Bhatti. *Current Advancements in Stereo Vision.* InTech, 2012.

[Bis95] C. M. Bishop. *Neural networks for pattern recognition.* Oxford University Press, Oxford, 1995.

[Bis06] C. Bishop. *Pattern Recognition And Machine Learning.* Information Science and Statistics, 2006.

[BKP+11] E. Bosc, M. Köppel, R. Pepion, M. Pressigout, P. Ndjiki-Nya, and P. Le Callet. Can 3D synthesized views be reliabily assessed through usual subjective and objective evaluation protocols? In *Proc. Int. Conf. Imag. Process.*, 2011. © [2011] IEEE.

[BKY+11] G. Bang, M. S. Ko, J. Yoo, W.-S. Cheong, G.M. Um, and N. Hur. Boundary noise removal and common hole filling method for VSRS 3.5. *ISO/IEC JTC1/SC29/WG11 Doc. m19356*, Jan. 2011.

[BM95] G. Bishop and L. McMillan. Plenoptic modeling: An image-based rendering system. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, Chapel Hill, NC, USA, 1995.

[BPLC+11a] E. Bosc, R. Pepion, P. Le Callet, M. Köppel, P. Ndjiki-Nya, M. Pressigout, and L. Morin. Towards a new quality metric for 3-D synthesized view assessment. *IEEE J. Sel. Topics Signal Process.*, 5(7):1332–1343, November 2011. © [2011] IEEE.

[BPLC+11b] E. Bosc, R. Pepion, P. Le Callet, M. Köppel, P. Ndjiki-Nya, M. Pressigout, L. Morin, and M. Pressigout. Perceived quality of DIBR-based synthesized views,. In *Proceedings of SPIE Optical Engineering and Applications*, San Diego, USA, Aug. 2011.

[BSCB00] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, Minnesota, Minneapolis, 2000.

[BVSO03] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting. *IEEE Trans. on Image Processing*, 12(8):882–889, 2003.

[CK85] R. Chellappa and R.L. Kashyap. Texture synthesis using 2-d noncausal autoregressive models. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 33(1):194–203, 1985.

[CPT04] A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. on Image Processing*, 13(9):1–13, September 2004.

[CS01] T. F. Chang and J. Shen. Non-texture inpainting by curvature-driven diffusions (CDD). *ELSEVIER Journal Visual Communication and Image Representation*, 4(12):436–449, 2001.

[CSS08] R. Costantini, L. Sbaiz, and S. Süsstrunk. Higher order SVD analysis for dynamic texture synthesis. *IEEE Trans. on Image Processessing*, 17(1):42–52, January 2008.

[CTL+10] K.-Y. Chen, P.-K. Tsung, P.-C. Lin, H.-J. Yang, and L.-G. Chen. Hybrid motion/depth-oriented inpainting for virtual view synthesis in multiview applications. In *Proc. 3DTV-CON The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, pages 1–4, Tampere, Finland, June 2010.

[CW93] S. Chen and L. Williams. View interpolation for image synthesis. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 279–288, USA, Aug. 1993.

[DBZdWPHN11] L. Do, G. Bravo, S. Zinger, and de With P. H. N. Real-time free-viewpoint DIBR on GPUs for large base-line multi-view 3DTV videos. In *Proc. IEEE Conference on Visual Communications and Image Processing (VCIP)*, Tainan, Taiwan, Nov. 2011.

[DBZdWPHN12] L. Do, G. Bravo, S. Zinger, and de With P. H. N. GPU-accelerated real-time free-viewpoint DIBR for 3DTV. *IEEE Trans. Consumer Electronics*, 58(2):633–640, May 2012.

[DCWS03] G. Doretto, A. Chiuso, Y. Wu, and S. Soatto. Dynamic textures. *Int. J. Comput. Vision*, 51(2):91–108, 2003.

[DDY03] I. Drori, Cohen-Or D., and H. Yeshurun. Fragment-based image completion. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2003.

[Deg86] K. Deguchi. Two-dimensional auto-regressive model for analysis and sythesis of gray-level textures. In *Proc. of the 1st Int. Sym. for Science on Form, General Ed. S. Ishizaka, Eds. Y. Kato, R. Takaki, and J. Toriwaki*, pages 441–449, 1986.

[Dim13] Dimenco Displays. http://www.dimencodisplays.com, 2013.

[DKNNW10] D. Doshkov, M. Köppel, P. Ndjiki-Nya, and T. Wiegand. Complexity and quality evaluation of structure extrapolation methods within a fully automatic inpainting framework. In *Proc. Conf. Signals, Syst. Comput. (Asilomar)*, 2010.

[Dod05] N. Dodgson. Autostereoscopic 3D displays. *IEEE Computer Society*, 38(8):31–36, August 2005.

[DS11] I. Daribo and H. Saito. A novel inpainting-based layered depth video for 3DTV. *IEEE Trans. on Broadcasting*, 57(2):533–541, Jun. 2011.

[DSF+12] I. Daribo, H. Saito, S. Furukawa, S. Hiura, and N. Asada. *3D-TV Systems with Depth-Image-Based Rendering*, chapter Hole Filling for View Synthesis, pages 169–189. Springer, 2012.

[DSW+13] M. Damański, O. Stankiewcz, K. Wegner, M. Kurc, J. Konieczny, J. Siast, R. Stankowski, R. Ratajczak, and T. Grajek. High efficiency 3D video coding using new tools based on view synthesis. *IEEE Trans. on Image Processing*, 22(9):3517–3527, 2013.

[DVGNK99] K. J. Dana, B. Van Ginneken, S. K. Nayar, and J. J. Koenderink. Reflectance and texture of real-world surfaces. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1999.

[EF01] A. Efros and W. Freeman. Image quilting for texture synthesis and transfer. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2001.

[EL99] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, CA, USA, 1999.

[Feh03] C. Fehn. A 3D-TV approach using depth-image-based rendering (DIBR). In *Proc. of Visualization, Imaging, and Image Processing VIIP*, Berlin , Germany, 2003.

[Feh04] C. Fehn. Depth-image-based rendering (DIBR), compression and transmission for a new approach on 3D-TV. In *Proc. SPIE Conf. Stereoscopic Displays and Virtual Reality Systems X*, San Jose, USA, January 2004.

[FWL+11] M. Farre, O. Wang, M. Lang, N. Stefanoski, A. Hornung, and A. Smolic. Automatic content creation for multiview autostereoscopic displays using image domain warping. In *Proc. IEEE International Conference on Multimedia & Expo (ICME)*, pages 1–6, Barcelona, Spain, July 2011.

[Geo04] T. G. Georgiev. Photoshop healing brush: a tool for seamless cloning. In *Proc. of European Conf. on Comp. Vision (ECCV)*, pages 1–8, Prague, Czech Republic, May 2004.

[Geo06] T. G. Georgiev. Covariant derivates and vision. In *Proc. of European Conf. on Comp. Vision (ECCV)*, Graz, Austria, 2006.

[GK65] G. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. 1965.

[GLM14] C. Guillemot and O. Le Meur. Image inpainting: Overview and recent advances. *IEEE Trans. on Sinal Processing Magazine*, 31:127–144, 2014.

[HAXH14] H.-A. Hsu, O. C. Au, L. Xu, and W. Hu. Spatio-temporally consistent view synthesis from video-plus-depth data with global optimization. *IEEE Trans. on Circuits and Systems for Video Technology (CSVT)*, 24:74–84, 2014.

[HB95] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *Proc. of the 22nd annual conference on Computer graphics and interactive techniques*, pages 229–238. ACM, 1995.

[HE07] J. Hays and A. Efros. Scene completion using millions of photographs. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2007.

[HHH09] A. Hyvèarinen, J. Hurri, and P. O. Hoyer. *Natural Image Statistics*, volume 39. Springer, Feb., 2009.

[HZ03] R. Hartley, , and A. Zisserman. *Multiple View Geometry*. Cambridge University Press, 2003.

[ISO11] ISO. Applications and requirements on 3D video coding. *ISO/IEC JTC1/SC29/WG11 Doc. N11829*, Mar. 2011.

[ITU00] TELECOMMUNICATION STANDARDIZATION SECTOR OF ITU. Subjective video quality assessment methods for multimedia applications. *ITU-T Recommendation P.910*, P.910:4–5, 2000.

[JBS09] M. S. Joshi, P. Bartakke, and M. S. Sutaone. Texture representation using autoregressive models. In *IEEE International Conference on Advances in Computational Tools for Engineering Applications (ACTEA)*, 2009.

[JVV86] A. J. E. M. Janssen, R. N. Veldhuis, and L. B. Vries. Adaptive interpolation of discrete-time signals that can be modeled as autoregressive processes. *IEEE Trans. on Acoustics Speech and Signal Processing*, 34(2):317–330, 1986.

[KA11] S. Korman and S. Avidan. Coherency sensitive hashing. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1607–1614. IEEE, 2011.

[Kan79] G. Kanizsa. *Organization in Vision: Essays on Gestalt Perception*. Greenwood Press, 1979.

[Kas80] R. L. Kashyap. Univariate and multivariate random field models for images. *ELSEVIER Computer Graphics and Image Processing*, 12(2):257–210, 1980.

[KDNN09] M. Köppel, D. Doshkov, and P. Ndjiki-Nya. Fully automatic inpainting method for complex image content. In *Int. Workshop Imag. Analysis Multimedia Interactive Services*, 2009.

[KDR+15] M. Köppel, D. Doshkov, F. Racape, P. Ndjiki-Nya, and T. Wiegand. On the usage of the 2d-ar-model in texture completion scenarios with causal boundary

conditions: A tutorial. *ELSEVIER Signal Processing: Image Communication*, 32:106–120, 2015.

[KEBK05] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra. Texture optimization for example-based synthesis. *ACM Transactions on Graphics (TOG)*, 24(3):795–802, July 2005.

[KKY13] M. S. Koo, D. W. Kim, and J. Yoo. A new common-hole filling algorithm for virtual view synthesis with a probability mask. In *Proc. IEEE IVMSP workshop*, Seoul, Korea, Jun. 2013.

[KMA05] A. Kaup, K. Meisinger, and T. Aach. Frequency selective signal extrapolation with applications to error concealment in image communication. *Int. J. Electron. Commun. (AEÜ)*, 59:147–156, 2005.

[KMMNN13] M. Köppel, M. B. Makhlouf, M Müller, and P. Ndjiki-Nya. Temporally consistent adaptive depth map preprocessing for view synthesis. In *Proc. IEEE Visual Communications and Image Processing (VCIP)*, Kuching, Malaysia, Nov. 2013. © [2013] IEEE.

[KMNN13] M. Köppel, M. B. Makhlouf, and P. Ndjiki-Nya. Optimized adaptive depth map filtering. In *Proc. IEEE International Conference on Image Processing ICIP*, Melbourne, Australia, Sep. 2013. © [2013] IEEE.

[KMW16] M. Köppel, K. Müller, and T. Wiegand. Filling disocclusions in extrapolated virtual views using hybrid texture synthesis. *IEEE Trans. On Broadcasting*, 62(2):457 – 469, 2016. © [2016] IEEE.

[KNND+10] M. Köppel, P. Ndjiki-Nya, D. Doshkov, H. Lakshman, P. Merkle, K. Müller, and T. Wiegand. Temporally consistent handling of disocclusions with texture synthesis for depth-image-based rendering. In *Proc. IEEE International Conf. on Image Process. (ICIP)*, Hong Kong, China, September 2010. © [2010] IEEE.

[Kok98] A. Kokaram. *Motion Picture Restauration*. Springer, London, 1998.

[Kok04] A. Kokaram. A statistical framework for picture reconstruction using ar models. *Image Vision Comput.*, 22(2):83–172, February 2004.

[KP94] A. Kokaram and Rayner. P. Detection and interpolation of replacement noise in motion picture sequences using 3d autoregressive modellin. In *Proc. of the IEEE International Symposium on Circuits and Systems*, 1994.

[KSE+03] V. Kwatra, A. Schödl, I. Essa, G. Turk, and Aaron Bobick. Graphcut textures: image and video synthesis using graph cuts. In *ACM Transactions on Graphics (TOG)*, 2003.

[KWD+12a] M. Köppel, X. Wang, D. Doshkov, T. Wiegand, and P. Ndjiki-Nya. Consistent spatio-temporal filling of disocclusions in the multiview-video-plus-depth format. In *Proc. IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2012. © [2012] IEEE.

[KWD+12b] M. Köppel, X. Wang, D. Doshkov, T. Wiegand, and P. Ndjiki-Nya. Depth image-based rendering with spatio-temporally consistent texture synthesis for 3-D video with global motion. In *Proc. IEEE International Conf. on Image Process. (ICIP)*, 2012. © [2012] IEEE.

[KY14] S. M. Ko and J. Yoo. Virtual view generation by a new hole filling algorithm. *JEET Journal of Electrical Engineering & Technology*, 9:1023–1033, 2014.

[LH96] M. Levoy and P. Hanrahan. Light field rendering. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 31–42, New York, USA, Aug. 1996.

[LH09] S.-B. Lee and Y.-S. Ho. Discontinuity-adaptive depth map filtering for 3d view generation. In *Proc. International ICST Conference on Immersive Telecommunications IMMERSCOM*, Brussels, Belgium, Jun. 2009.

[LH11] C. Lee and Y.-S. Ho. Results of view synthesis using modified hole filling methods. *ISO/IEC JTC1/SC29/WG11 Document m19281*, January 2011.

[LKNNW10] H. Lakshman, M. Köppel, P. Ndjiki-Nya, and T. Wiegand. Image recovery using sparse reconstruction based texture refinement. In *Int. Conf. Acoustics, Speech, Signal Process.*, 2010. © [2010] IEEE.

[LL89] J. S. Lim and J. S. Lim. *Two-Dimensional Signal and Image Processing*. Prentice Hall, 1989.

[LNNK+09] H. Lakshman, P. Ndjiki-Nya, M. Köppel, D. Doshkov, and T. Wiegand. An automatic structure-aware image extrapolation applied to error concealment. In *Proc. of the 16th IEEE international conference on Image processing*, Cairo, Egypt, 2009. © [2009] IEEE.

[LZW03] A. Levin, A. Zomet, and Y. Weiss. Learning how to inpaint from global image statistics. In *Proc. International Conference on Computer Vision (ICCV)*, 2003.

[MDdWPHN12] L. Ma, L. Do, and de With P. H. N. Depth-guided inpainting algorithm for free viewpoint video. In *Proc. IEEE International Conference on Image Processing (ICIP)*, Florida, USA, Sep. 2012.

[MFY+08] Y. Mori, N. Fukushima, T. Yendo, T. Fujii, and M. Tanimoto. View generation with 3D warping using depth information for FTV. In *Proc. 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, Nagoya, Japan, May 2008.

[MFY+09] Y. Mori, N. Fukushima, T. Yendo, T. Fujii, and M. Tanimoto. View generation with 3D warping using depth information for FTV. *ELSEVIER Signal Processing: Image Communication*, 24:65–72, 2009.

[MJ11] P. Merkle and F. Jäger. Description of core experiments for HEVC-based 3DV. *ISO/IEC JTC1/SC29/WG11 Document n12354*, November 2011.

[MMW10] K. Müller, P. Merkle, and T. Wiegand. 3-D video representation using depth maps. *IEEE Trans. on IEEE*, 99(4):643–656, April 2010.

[MSD+08] K. Müller, A. Smolic, K. Dix, P. Merkle, and P. Kauff. View synthesis for advanced 3d video systems. *J. on Image. Video. Process.*, 2008:1–11, November 2008.

[NN08] P. Ndjiki-Nya. *Mid-Level Content-Based Video Coding using Texture Analysis and Synthesis*. PhD thesis, Technischen Universität Berlin, 2008.

[NNDK+12] P. Ndjiki-Nya, D. Doshkov, H. Kaprykowsky, F. Zhang, D. Bull, and T. Wiegand. Perception-oriented video coding based on image analysis and completion: A review. *ELSEVIER Signal Processing: Image Communication*, 27(6):579–594, 2012.

[NNKD+10] P. Ndjiki-Nya, M. Köppel, D. Doshkov, H. Lakshman, P. Merkle, K. Müller, and T. Wiegand. Depth image based rendering with advanced texture synthesis. In *Proc. IEEE International Conference on Multimedia & Expo (ICME)*, Singapore, July 2010. © [2010] IEEE.

[NNKD+11] P. Ndjiki-Nya, M. Köppel, D. Doshkov, H. Lakshman, P. Merkle, K. Müller, and T. Wiegand. Depth image-based rendering with advanced texture synthesis for 3-D video. *IEEE Trans. On Multimedia*, 13(3):453–465, June 2011. © [2011] IEEE.

[NNKDW08] Patrick Ndjiki-Nya, Martin Köppel, Dimitar Doshkov, and Thomas Wiegand. Automatic structure-aware inpainting for complex image content. In *Proc. of the 4th International Symposium on Advances in Visual Computing (ISVC)*, pages 1144–1156, Las Vegas, USA, Dec. 2008.

[NNSW07] P. Ndjiki-Nya, C. Stueber, and T. Wiegand. Texture synthesis method for generic video sequences. In *Proc. Int. Conf. Image Process.*, 2007.

[PF06] S. Periaswamy and H. Farid. Medical image registration with partial data. *Elsevier Medical Image Analysis*, 10:452–464, 2006.

[PGB03] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 313–318, San Diego, USA, Jul. 2003.

[PKGS13] N. Plath, S. Knorr, L. Goldmann, and T. Sikora. Adaptive image warping for hole prevention in 3D view synthesis. *IEEE Trans. on Image Processing*, 22(9):3420–3432, Sep. 2013.

[PS00] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *IEEE Int. J. Comput. Vision*, 40(1):49–71, 2000.

[RDKNN14] F. Racape, D. Doshkov, M. Köppel, and P. Ndjiki-Nya. 2D+T autoregressive framework for video texture completion. In *Proc. IEEE International Conference on Image Processing (ICIP)*, Paris, France, Oct. 2014. © [2014] IEEE.

[RKDNN14] F. Racape, M. Köppel, D. Doshkov, and P. Ndjiki-Nya. Adaptive 2d-ar framework for texture completion. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Florence, Italy, May 2014. © [2014] IEEE.

[SJ10] M. Schmeing and X. Jiang. Depth image based rendering: A faithful approach for the disocclusion problem. In *Proc. 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, Münster, Germany, 2010.

[SJ11] M. Schmeing and X. Jiang. Time-consistency of disocclusion filling algorithms in depth image based rendering. In *Proc. 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, May 2011.

[SK00] H-Y. Shum and S. Kang. Review of image-based rendering techniques. In *Proc. Visual Communications and Image Processing*, 2000.

[SMW06] H. Schwarz, D. Marpe, and T. Wiegand. Analysis of hierarchical b pictures and mctf. In *Proc. IEEE International Conference on Multimedia & Expo (ICME)*, 2006.

[SP96] M. Szummer and R. Picard. Temporal texture modeling. In *Proc. on International Conference on Image Processing (ICIP)*, 1996.

[SS02] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42, April-June 2002.

[SSN07] A. Saxena, J. Schulte, and A. Ng. Depth estimation using monocular and stereo cues. In *Proc. Int. Joint Conf. Artificial Intelligence*, 2007.

[SWL+13] N. Stefanoski, O. Wang, M. Lang, P. Greissen, S. Heinzle, and A. Smolic. Automatic view synthesis by image-domain-warping. *IEEE Trans. on Image Processing*, 22(9):3329–3341, Sep. 2013.

[SYJS05] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum. Image completion with structure propagation. In *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 861–868, New York, NY, USA, 2005.

[Tel03] A. Telea. An image inpainting technique based on the fast marching method. *J. Graphics Tools*, 9(1):25–36, May 2003.

[TFS08] M. Tanimoto, T. Fujii, and K. Suzuki. View synthesis algorithm in view synthesis reference software 2.0 (VSRS 2.0). *ISO/IEC JTC1/SC29/WG11 Doc. m16090*, Feb. 2008.

[TLD07] Z. Tauber, Z-N. Li, and M. Drew. Review and preview: Disocclusion by inpainting for image-based rendering. *IEEE Trans. on Systems, Man, and Cybernetics: Applications and Reviews.*, 37(4):527–540, July 2007.

[Tos13] Toshiba 55ZL2. http://www.toshiba.com, 2013.

[Tug94] Jitendra K Tugnait. Estimation of linear parametric models of nongaussian discrete random fields with application to texture synthesis. *IEEE Trans. on Image Processing*, 3(2):109–127, 1994.

[Vid10] Video. Description of exploration experiments in 3d video coding. *ISO/IEC JTC1/SC29/WG11 Document n11630*, October 2010.

[WAX+14] S. Wenxiu, O. C. Au, L. Xu, Y. Li, and W. Hu. Seamless view synthesis through texture optimization. *IEEE Trans. on Image Processing*, 23(1):342–355, Jan. 2014.

[WBSS04] Z. Wang, A. C. Bovic, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on Image Processing*, 13(4):600–612, 2004.

[Whe38] C. Wheatstone. Contributions to the physiology of vision.-part the first. on some remarkable, and hitherto unobserved, phenomena of binocular vision. *"Philosophical Transactions" of the Royal Society of London*, 128:371–394, 1838.

[WL00] L-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proc. of Special Interest Group on GRAPHics and Interactive Techniques (SIGGRAPH)*, CA, USA, 2000.

[WLKT09] L-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk. State of the art in example-based texture synthesis. In *Proc. Eurographics*, 2009.

[XS10] Z. Xu and J. Sun. Image inpainting by patch propagation using patch sparsity. *IEEE Trans. on Image Processessing*, 19(5):1153–1165, May 2010.

[XWY+13] M. Xi, l.-H. Wang, Q.-Q. Yang, D.-X. Li, and M. Zhang. Depth-image-based rendering with spatio and temporal texture synthesis for 3DTV. *EURASIP Journal on Image and Video Processing*, 9(1):1–18, 2013.

[XYXJ12] Li Xu, Qiong Yan, Yang Xia, and Jiaya Jia. Structure extraction from texture via natural variation measure. *Proc. ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH Asia)*, 2012.

[YTZ+14] C. Yao, T. Tillo, Y. Zhao, H. Xiao, H. Bai, and C. Lin. Depth map driven hole filling algorithm exploiting temporal correlation information. *IEEE Trans. on Broadcasting*, 60(2):394–404, 2014.

[YYH+14] X. Ye, J. Yang, H. Huang, C. Hou, and Y. Wang. Computational multi-view imaging with Kinect. *IEEE Trans. on Broadcasting*, 60(3):540–554, 2014.

[ZRDdWPHN12] S. Zinger, D. Ruijters, L. Do, and de With P. H. N. View interpolation for medical images on autostereoscopic displays. *IEEE Trans. on Circuits and Systems for Video Technology*, 22:128–137, 2012.

[ZT05] L. Zhang and W. Tam. Stereoscopic image generation based on depth images for 3D TV. *IEEE Trans. on Broadcasting.*, 51(2):191–199, 2005.