Representation and coding of 3D video data

Josselin Gautier, Emilie Bosc, Luce Morin

Projet PERSEE
« Schémas Perceptuels et Codage vidéo 2D et 3D »
n° ANR-09-BLAN-0170

Livrable **D4.1** *17/11/2010*

Representation and coding of 3D video data

Josselin	GAUTIER	IRISA
Emilie	BOSC	INSA
Luce	MORIN	INSA

Contents

List of Figures

Chapter 1

Introduction

3D video is meant to provide the immersion feeling to the user or interactivity depending on the application. Recently, consumer markets have just began to emerge, and the need for extending the visual sensation to the third dimension is believed to grow, as the entertainment market (IMAX cinemas, gaming) initiated it. Indeed, two main applications may refer to 3D video: 3DTV and FTV (Free Viewpoint Video).
3DTV consists in 3D relief rendering through the use of stereopsis: the human vision system assesses depth by fusing two displayed images (one for each eye but from two slightly different viewpoints).
FTV provides interactivity because the user is free to select a desired viewpoint of the displayed scene.
Those applications require specific processing chains, from the acquisition step to the display step. To make 3DTV and FTV possible, many issues appear. A large number of videos are captured, synchronized, and processed. This supposes accurate calibration of the cameras. Storing 3D video data then induces other challenges such as the appropriate representation and the compression of the data. Furthermore, it turns out that every single step of the processing chain is linked to the others: the choice of a representation of the data controls the requirements for acquisition, the coding process choices, the rendering process design, and the possible application.
Currently, two standards deal with 3D-video: MPEG-C part 3 (developed by the Stereoscopic Video group) that compresses one color video and its associated depth map video; and MVC (Multi-View Coding) that encode multiple views by exploiting the spatial (i.e. between views) and temporal (i.e. within a single view) redundancies. But, those methods still do not provide significant gain and seem not to exploit sufficiently the available 3D information from the videos.
Many representations are candidates for 3D video and should be appropri-

ately chosen depending on the targeted application. Otherwise, new representations and compression methods must be investigated. This document present the existing representations and coding methods for 3D video sequences.

Chapter 2

Representation

2.1 Introduction

Today, a 3D video system provides 3D content in different forms, depending on the input source, the type of acquisition and scene geometry, the desired level of quality, the type of application and the bandwidth. Then, the 3D-scene representation is the key technology between acquisition (or content generation), compression, transmission and display (figure 2.1). Especially, the compression stage will be interdependent of the representation used, so both stages have to be considered. Indeed the requirements for each stages may vary from one to another, and may not always be compatible. Representation will be partly conditioned by the acquisition, but from the choice of representation will depends the compression but also the transmission, rendering, degree and mode of interactivity.

A survey of existing representations will be introduced. Keeping in mind the 3DTV and FTV applications and their requirements, they will be considered in terms of feasibility, compression efficiency or compactness, interactive rendering, level of detail and perceptual quality.

2.2 Requirements

Requirements for 3DTV 3DTV means that, whatever the type of display, a set of multiple videos should be displayed in real time. Indeed, the stereoscopic effect is realized by the displaying of two slightly shifted views to each viewer eye. The usage of more views guarantees the accessibility of more viewers on certain display technologies, but also a better viewing comfort (the user can move freely without necessity of glasses), then increasing the viewing experience.

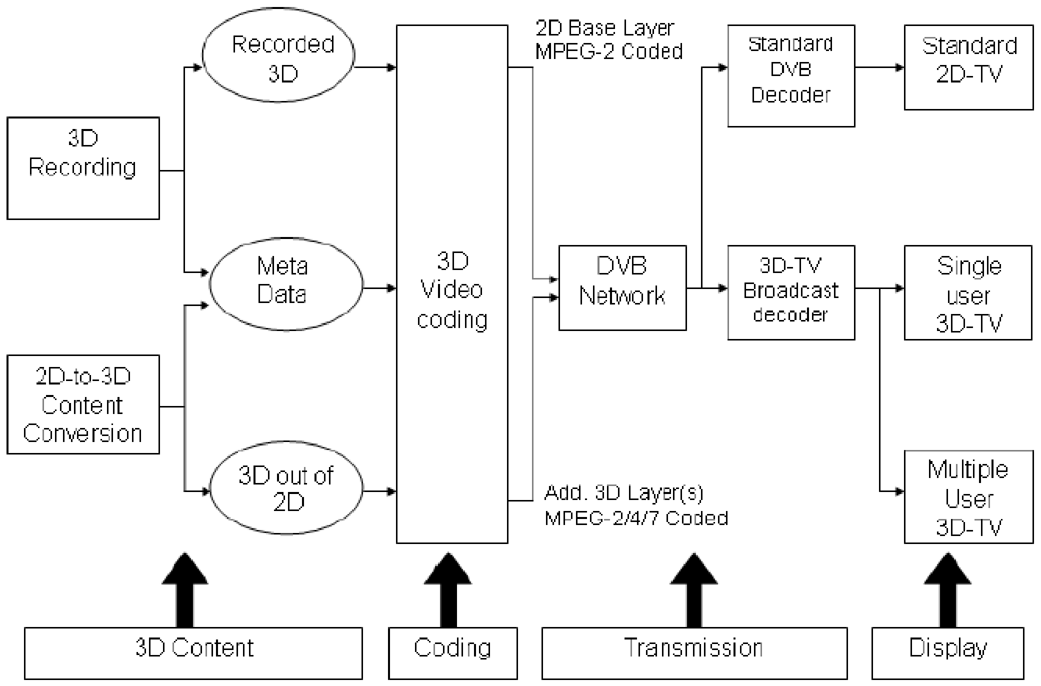


Figure 2.1: The ATTEST 3-D video processing chain

Quality, consistency between the multiple views has to be guaranteed over a higher level of quality on existing standards.Indeed, it is now widely accepted that 3DTV could not be deployed if the quality perceived by viewer do not exceed the existing 2D quality standards, such as High Definition.

The choice of the representation-compression methods, should also consider the **progressivity**, regarding broadcasting methods in limited or noisy environment (cable, satellite, antennas, Internet). Finally, **backward compatibility**, i.e. the capacity of the codec -software- to include a former one, or the capacity of the hardware (set top box are now widely used) to decode this new bitstream, is necessary for deployment. Typically, the end-to-end broadcast architecture should support for one channel the diffusion of multi view video and guarantee the display in real-time without jiggs or any block effect, and this whatever the terminal computation capabilities.

Requirements for Free Viewpoint TV FTV (sometimes called FVV) is a functionality for viewing and interactively control the viewpoint in a natural video scene. (For synthetic video, it is commonly called virtual reality). The viewer choose where to move in the scene, and the new desired viewpoint -virtual camera- is displayed interactively. So considered, the focus of attention will be partly controlled by the viewers rather than a director, so each viewer may observed a unique viewpoint. Thus, the wider and denser the acquisition cameras are placed on an axis, the better the reconstruction quality will be. However, a correct trade-off depending of the application, should be defined between amount of camera -data- at capture, transmission capacity and complexity for real-time rendering.

Considering this, the FTV requires especially **level of detail (LoD) scalability** -the viewer may want to move freely in the scene and zoom on parts of the texture- different real time processing such as interpolation by inpainting at user side may be preferred. This means that high computational capabilities realized today by Graphics Processor Units should be support on rendering device if FTV functionalities are expected. Efficient **compression** of this accurate 3D scene are also needed for transmission and broadcasting. **Flexibility and capability** of FTV is also an important issue, space-time manipulation by the viewer involves real-time rendering but again high computation capabilities. Then, a **low complexity** 3D model will be expected.

Mutual requirements Multiview video acquisition can range from partial (about 30 degrees) to complete (360 degrees) coverage of the scene. Stereoscopic views can then be rendered and used both for 3DTV and FTV application, once virtual views different from acquired views can be synthesized. 3DTV and FTV functionality are not only compatible but mutually usable. Indeed, 3DTV and FTV share common requirements of flexibility, compression efficiency, and quality. In the case of FTV as an extra application of 3DTV where the free viewpoint navigation is realized commonly with stereo display, high level of requirements will then be expected : progressivity, scalability, quality, and flexibility.

2.3 Image based representations

The representations based only on color images can be divided in two types, either describing the data in the image domain, through array of pixels, either representing it as a light flow.

2.3.1 Image domain

Two existing solutions represent the information in the image domain only. The first one displays directly from cameras to back-end the same original images, either in stereoscopic or in multiview mode:

Conventional Stereoscopic Video The stereoscopic mode consists in simply providing a 3D impression with a pair of left and right videos i.e. with a stereo camera system for the acquisition and with a stereoscopic screen for the display, as illustrated in figure 2.2:

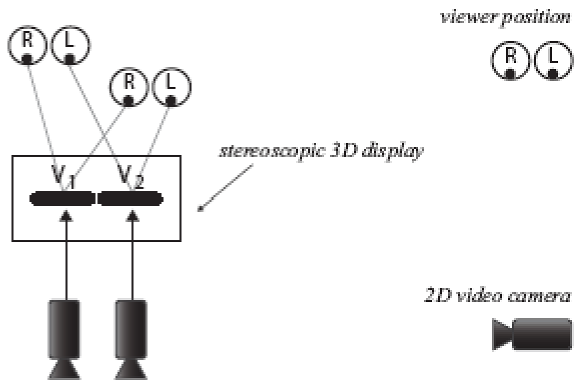


Figure 2.2: Efficient support of stereoscopic display based on stereo video content (from [1])

A common way to represent and transmit these two video streams is to multiplex them temporally or spatially. Within the time multiplexed format, left and right pictures are interleaved temporally, as alternating frames. Whereas with spatial multiplexing, left and right pictures are squeezed along the horizontal or vertical axis to fit in the original picture dimension, at the cost of a spatial resolution lost along this axis.

The main limitation of the stereo representation is the hardware acquisition dependency. The conditions of acquisition, especially the fixed baseline between the two cameras, are optimized for one type of stereoscopic display (regarding its size, type). Without much information than two 2D views, occlusion-disocclusion and new view synthesis can not be supported.

Multiview Video As described in the 3DTV requirements, we focus on multiview enabling representation, and its promising results of viewing comfort and immersion. A first general representation is the Multi View Video (MVV) [4] representation. It commonly describes a set of consecutive views which acts like local stereo pairs to guarantee stereoscopy to the viewer (figure 2.3). The Head motion parallax viewing can then be supported within practical limits. Without any intermediate representation, MVV minimizes the image transformation, but suffers of flexibility and of its high capacity channels requirements.

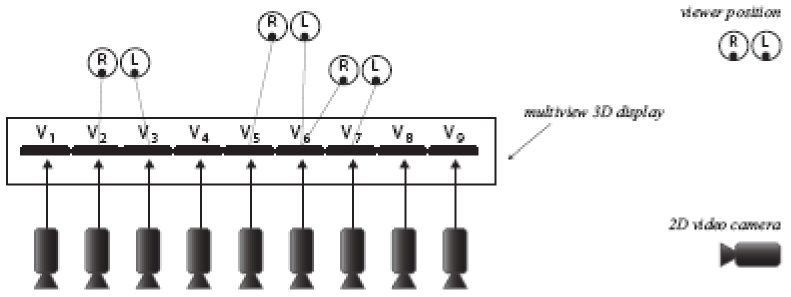


Figure 2.3: Efficient support of multiview autostereoscopic displays based on MVV content (from [1])

The second one is based on a real-time estimation of an approximate geometry of the scene in order to generate virtual viewpoints [Zhang 04, Nozick 06, Taguchi 08]. In the first case, the acquisition and display processes are linked, flexibility is impossible. In the second case, the estimated geometry is likely to contain inaccuracies, resulting in rendering artifacts if the density of the sampling is sparse.

2.3.2 Plenoptic function and light field

An alternative to the input 2D images -pixel array- from camera consists in describing the flow of light : the plenoptic function [5]. It describes the light rays received in different direction of space, generally in a 7 dimension space. The light intensity I is received at a 3D space viewpoint (x, y, z), under a certain viewing direction (θ, ϕ), with a certain wavelength λ and at a considered time t. The information acquired by camera give certain discrete values of this unknown function and an interpolation could then be applied on this known value. But acquiring the full plenoptic function is not feasible in practice due to the heavy processes and the huge amount of data required. The question is then on how to reduce the dataset while keeping the rendering quality.

Levoy et al. [6] proposed to make different assumptions, by ignoring wavelength and time dimensions first, which reduce the function to 5 dimensions. The light field, the radiance is expressed as a function of position and direction, in regions of space assumed free of occluders. Consequently it does not change along a line in free space, and the light field in this *free space* is then a 4D and not a 5D function.

An image is then considered as a two dimension slice of this 4D light field: creating a light field from images corresponds to inserting each 2D slice into the 4D light field representation. At the opposite, generating new views corresponds to extracting and resampling a slice. Figure below presents two visualizations of a light field from two slices called camera plane (u, v) and from focal plane (s, t).

Precisely, lights rays are stored by the intersection of one plane with coordinate (u, v), the other with coordinates (s, t). During the rendering, each ray r_i passes through the two planes and generates a particular sample (u_i, v_i, s_i, t_i) If this sample already exists in the database, the color value is applied, if not, the nearest ones are selected and interpolated.

The advantages of plenoptic function and light field is their capacity to render photo-realistic images, but at a cost of an high camera density and a necessary high bitrate.

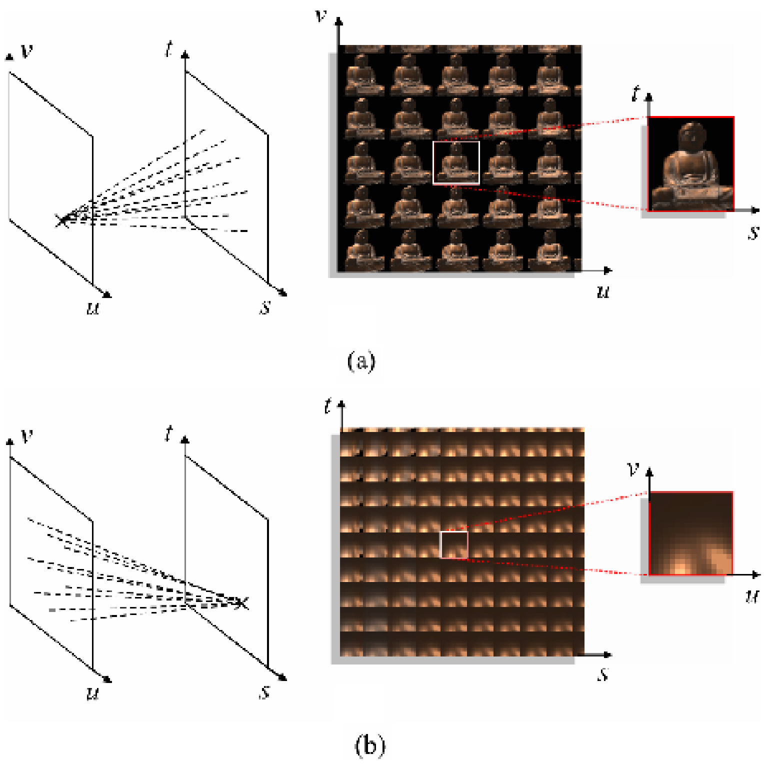


Figure 2.4: Two visualizations of a light field: (a) each image in the array represents the rays arriving at one point on the uv plane from all points on the st plane, as shown on left.(b) each image represents the rays leaving one point on the st plane bound for all points on the uv plane. The images in (a) are off-axis perspective views of the scene, while the images in (b) look like reflectance maps. The latter occurs because the object has been place astride the focal plane, making sets of rays leaving point on the focal plane similar in character to sets of rays leaving points on the object.

2.4 Depth image based representations

A representation of a point in 3D-space can consist in a three dimension vector (or four in homogeneous coordinates). The depth, or distance to the referential -here the camera- expresses this third coordinate often called z. Now considering the projection of all the z coordinates of an object in space into an image plane viewed from a given camera, we obtain a 2D image called “depth map” or “Z-map”. There exist various way to obtain those maps. Either by real sensor acquisition, like laser scanner or Z-cam: it is possible to get a set of points of a real object in 3D space, with their coordinates x,y,z. Either by stereo calculation (or stereo correspondence): the projection of pixels displacement from one view to another reflect, under assumptions and uncertainty, the disparity between these two views and then the depth map from one view. The depth map is consequently estimated. The common usage of a depth map with its associated 2D color (or sometimes called “texture”) image in the same coordinates enables to build a coherent 3D-like representation in one view : a 2D+Z representation.

Knowing the position of a pixel in a 3D space, it can then be projected to another location in an image to render an arbitrary view of the scene through image warping. This is particularly useful for free-viewpoint TV and 3DTV scenario where novels views can be generated with this depth information. This is call Depth Image Based Rendering (DIBR). Then, compared to a configuration with a number of cameras equal to the number of possible view, the density of camera over an axis can be subsequently reduced, or the viewing angle enlarged. We can consider that the larger the viewing angle, the better the 3D experience. In the case where the depth map is not transported but computed at the receiver side, this computational cost for rendering will limit the real time capabilities. Again, using depth maps in the representation constitutes another advantage. Finally the format of depth map -2D one component map at the same resolution as video- make the depth image based representation backward compatible to existing 2D TV digital coders.

In the next subsections, the evolution of the depth based representation will be described, from the 2D+Z format to the DES multi-layered multi-view based one, thanks to the gain of additional occlusion information and views.

2.4.1 2D+Z

A 2D+Z representation consists, for one given viewpoint, in a 2D image and its associated depth map. This pair allows to generate a novel viewpoint for stereoscopy but in a relative short range limited by the disocclusion appearance.

Two approaches are given to cope with the intrinsic limitation of a single texture view plus a depth map, the first one by multiplying the number of views and depth maps to transmit, the second approach using additional layers to limit the dissoclusions.

2.4.2 MVD

Considering the necessity of a larger viewing angle, both for free viewpoint applications and for 3DTV DIBR issues, a Multi-View video + Depth (MVD) representation can be considered (figure 2.5). This is a combination of previous 2D+Z with MVV representations: multiple 2D videos are used with their associated depth video.

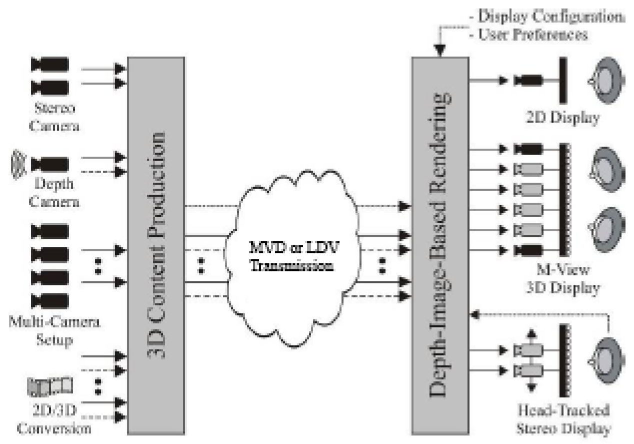


Figure 2.5: Possible scenario of a future 3DTV service, relying on MVD representation and transmission (from [2])

Multitexturing -i.e. multiple camera views combining texture of the 3D scene- permits theoretically to increase the resolution and so the quality of the rendered images. The drawback is the relative rendering complexity and the high correlation of information between different views, leading to huge input view data volumes to be compressed. Depth video has to be acquired or estimated for N views , then N 2D videos and N depth video have to be transmitted, and finally multiple virtual views have to be rendered, depending on the device, as illustrated in fig.2.6 with an autostereoscopic display. Scalability and progressivity in MVD can however be considered, where a base layer is accessible for low complexity devices. Concerning the

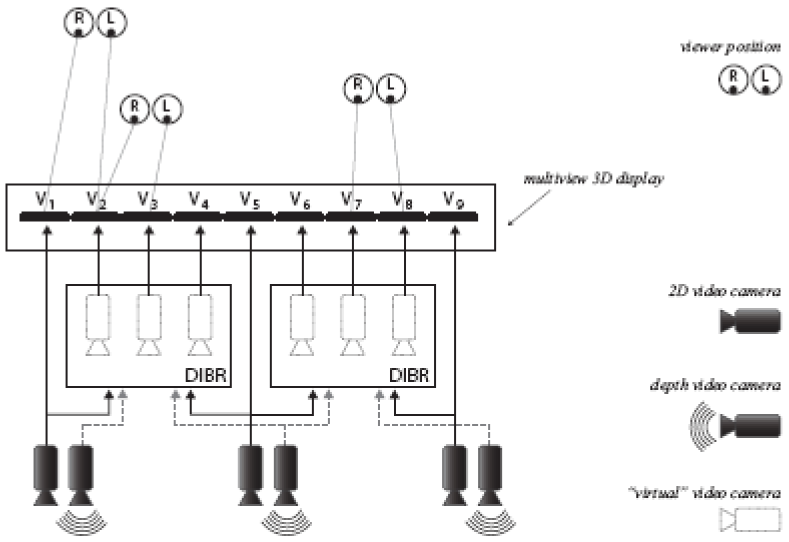


Figure 2.6: Support of multiview stereoscopic displays based on MVD content

rendering quality of MVD, some efforts have been put to improve it either during acquisition (camera calibration issues), representation, coding (like MVC, please report to chapter 3) and displays. In [7] a MVD representation is enhanced with matting information at depth discontinuitues. This matting information increases the rendering quality at the objects boundaries where pixels color values are usually mixed between background and foreground.

2.4.3 LDI

The LDI representation [8] consists in representing color and associated depth pixels in their consecutive position along some depth layers. Then, a set of layers, a layered depth image, could store the repartition of relevant texture onto layer. It then avoid the MVD limitation of storing redundant identical textures obtained from different views: the common textures are all fused and expressed in one common view.

Concretely, a LDI is then a 3D matrix of visible and occluded pixels viewed from a reference camera. Each LDI pixel, i.e. Layered Depth Pixel (LDP), is composed of different Depth Pixels (DPs) carrying both color and depth information. Then, the main advantage of LDI lies in the reduction of the correlated data over the multi-view videos sequences, at the expense of a computational cost of projection. Historically, the idea was to filter the depth values of the warped LDP, using a depth threshold Δ_z, both for avoiding warping inaccuracy during construction [8] or for compression efficiency [9].

The impact of this depth threshold on the trade-off between the layer filling rate and the quality of the synthesized view is still an open issue. Cheng et al. [10] introduce a clustering over the depth pixels to avoid matting or ghosting effects.

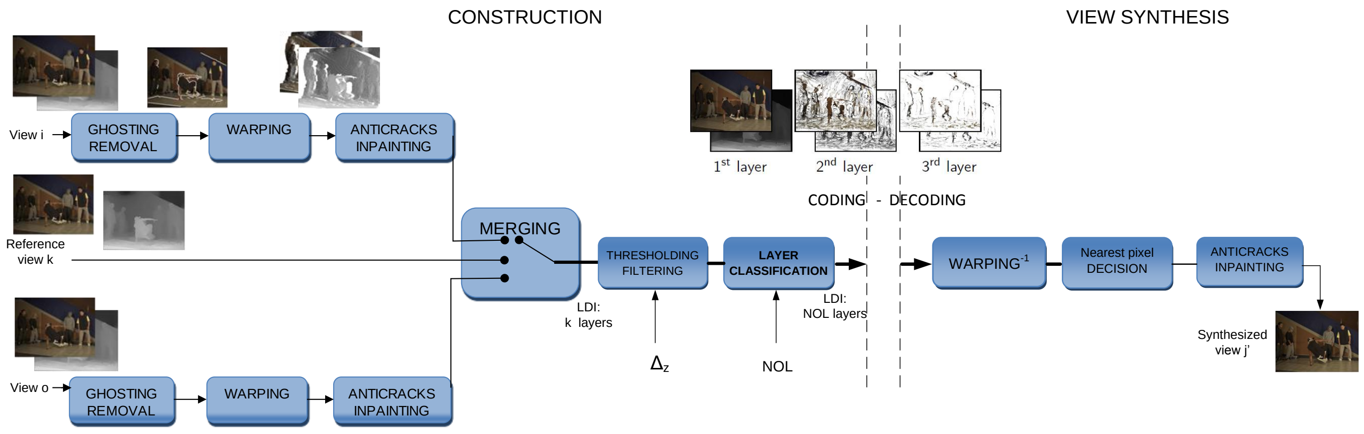


Figure 2.7: LDI Construction and Rendering scheme

LDI extensions In the next sections, different variants and extensions of the LDI over the time, space (among the views), -or over the way to organize those layers- are presented.

2.4.4 I-LDI

An alternative approach to reduce pixel redundancy between the layer - hence to reduce filling rate - was presented by Jantet et al [11]. The layers were decorrelated using an incremental construction: a logical exclusion between a real-view and a virtual view obtained for a temporary LDI enabled to compute occluded areas that could be added to the reference viewpoint in the I-LDI.

While the LDI can contain many but partially empty layers, the I-LDI incremental approach, based on a mutual-exclusive construction, is supposed to better support the necessary information, especially the occluded areas. Secondary texture and depth layers contains effectively a better pixel distribution. Studies show that a clustered-based segmentation of classical LDI can also help to reduce the layers completion rate and decrease the spreading of the layer's pixels.

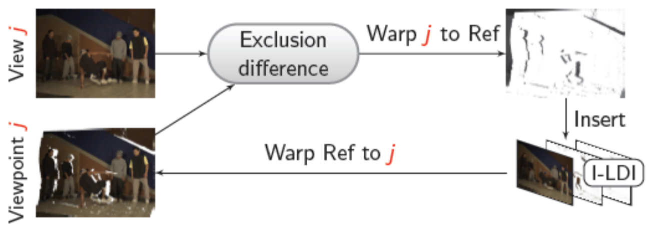


Figure 2.8: I-LDI exclusion-based construction scheme

2.4.5 LDV-R

Among the variants in the supports of the LDI over the time, i.e Layered Depth Video (LDV), Bruls et al. [12] proposed to add a 2nd color view to a classical LDI on 2 layers, i.e color and depth of one view (first layer), background color and background depth (second layer). Then, this Left layered Depth Video + Right format -LDVR- is compared with stereo MVC and show a bitrate increase of a 1.25 factor, while containing additional background color and depth. In this continuity, they proposed a modified MVC encoder, that use the spatial prediction from pre-synthesized Right view from the Left LDV part. This scheme while showing good result, might be an hard trade-off between the bitrate saving and the high complexity on the decoder. On the output of MVC, this LDVR rendering scheme first decode the left view and synthesized the predictor of the right view to help finally to decode the right view. Then, the LDVR layers are used to synthesized novel views.

2.4.6 DES

Another extension to those LDV has been proposed by Smolic, Mueller and Merkle in [13]. To overcome the high variety of 3D video formats, they proposed “a generic, flexible and efficient format” combining the capabilities of the basic 3D video formats : the concept of what they call Depth Enhanced Stereo (DES).

This format, that can be seen as a container format, extends the conventional stereo, with the LDI capabilities. This double-LDI representation provide stereo backward capabilities, but also enable depth-based view synthesis for autostereoscopic rendering with an improved quality over simple LDI. No implementation neither results have been proposed yet, but we can bet this generic solution has a promising future in standardization.

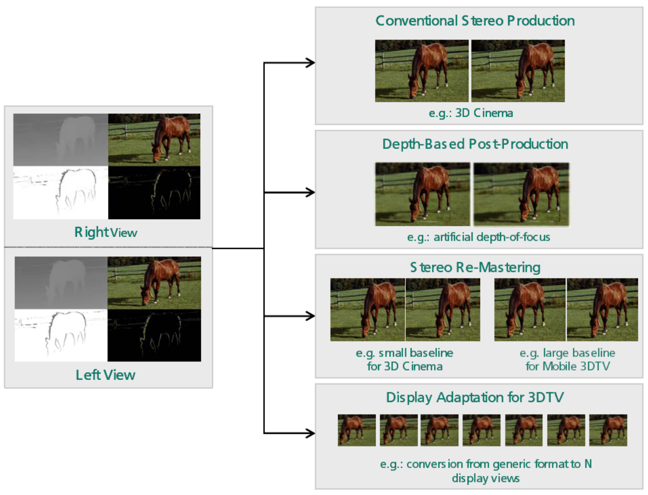

Conventional Stereo Production
e.g.: 3D Cinema
Depth-Based Post-Production
e.g.: artificial depth-of-focus
Stereo Re-Mastering
e.g. small baseline for 3D Cinema
e.g. large baseline for Mobile 3DTV
Display Adaptation for 3DTV
e.g.: conversion from generic format to N display views
Right View
Left View

Figure 2.9: Depth enhanced stereo (DES), extending high quality stereo with advanced functionalities based on view synthesis.

2.5 Surface-based representation

In this section we discuss four different representations: polygonal meshes, NURBS, subdivision surfaces and polygon soup that could address the 3DTV requirements.

2.5.1 Polygonal meshes

Polygons are widely used in the computer graphic community (from entertainment to manufacturing), as they are the primitives of hardware rendering technologies. Today's graphic card processes more than millions of polygons, enabling to render realistic scene with complicated objects. But such complex meshes are expensive to store, transmit and render. Many mesh simplification and compression techniques (please report to section compression of meshes) lead to different flexible representations with different level of detail.

The *progressive meshes* is a seminal technique, where an arbitrary triangular mesh can be stored as a coarser mesh with a sequence of mesh refinement operations called *vertex splits*. It consists of a local elementary mesh transformation that adds a single vertex to the mesh. Then a continuous sequence of meshes can be represented with increasing accuracy depending on the viewpoint. Indeed, polygonal artifact can appear along the silhouette boundaries, especially for a close viewpoint or for low resolution representation. A view-dependent rendering combined to an associate transmission strategy can selectively refine a progressive mesh along object boundaries for a given viewpoint by using such vertex split operations [14].

The *progressive time-varying meshes*: A static connectivity along the time, i.e for all frames of the animation, with transmission of the vertex positions, is an efficient and space-saving technique, but which often leads to inadequate modeling of deformable surface. A progressive scheme based on edge splits (contractions) to refine (or simplify) the geometry of a given mesh has been recently proposed [15]. The edge contractions are clustered according to a base hierarchy that gives a LoD (Level of Detail) approximation for the first frame. The hierarchy is then incrementally adapted to the geometry of the next frame b using edge swap operation (see figure 1.9). The whole deforming surface is thereby coded by initial vertex positions with the base hierarchy, the swap sequence and the vertex displacements for each frame.

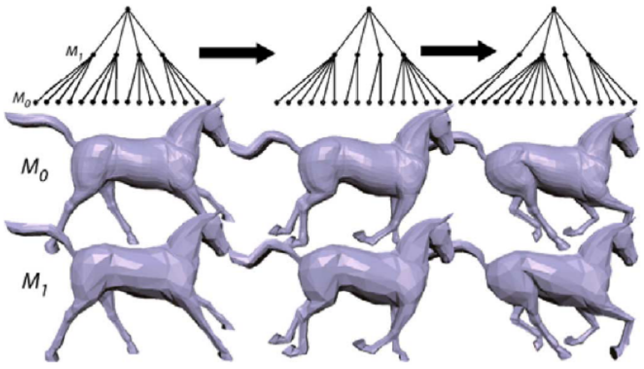


Figure 2.10: Progressive Time-Varying Meshes. The 3D Horse is presented at two levels of detail. ©ACM Inc.

2.5.2 NURBS

Nonuniform rational B-spline Surface (NURBS), is the most common representation of parametric surface based representation, especially in CAD and CAM domains. A B-spline surface is a continuous piecewise polynomial surface defined like the union of surface patches of fixed degrees.

The NURBS are a generalization of these B-spline functions, where a weight w_{ij} is associated for each control point (see figure.2.12).

This coefficient is a tension parameter: increasing the weight of a control point pulls the surface toward that point. A point on a NURBS surface S is defined by:

$$S(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{ip}(u)N_{jq}(v)w_{ij}\mathbf{B}_{ij}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{ip}(u)N_{jq}(v)w_{ij}B_{ij}}, 0 \leq u,v \leq 1$$

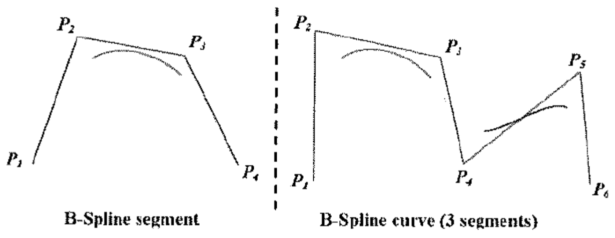


Figure 2.11: B-spline curve.©3D image processing

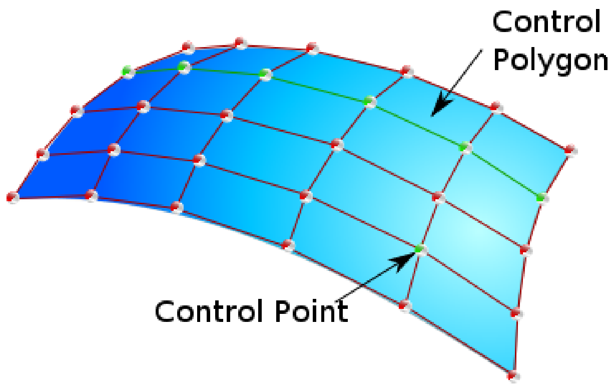


Figure 2.12: Illustration of a NURBS surface with control point and control polygon

where $\{N_{ir}\}$ denotes the B-spline basis functions, p, q the degrees (order) of the surface in u and v directions, $\mathbf{B}_{i,j}$ a mesh of $n \times m$ control points. The two knots U and V are vectors specifying the domain over which the B-spline basis functions are defined. It consists of two nondecreasing sets of real numbers (knots) that partition the parameterization domain into subintervals : $U = u_1, ...u_n$ and $V = v_1, ...v_n$.

The NURBS representation as a tensor product surface, can represent planar surfaces or quadrics (spheres, cylinders) but also surfaces having sharp edges. The benefits of these surface are that they are mathematically complete and easy to sample or to digitize in voxels or triangles. As they don't depend on a scale factor, NURBS have theoretically an infinite resolution.

In practice, it is sampled into triangular or quadrilateral representation for GPU-based rendering. The Level of Detail control of NURBS surfaces can be constructed only if the complete representation is available, which is rarely the case in a compact representation goal.

The second limitation to represent fine detail comes from the NURBS construction itself. Local refinement of a NURBS surface necessitates large-scale modification. To add a single control point within a patch, an entire column or row of control points must be split to preserve the desired quadrilateral grid structure. Two solutions consists of either using displacement maps, where a model stored fine details as if there were a kind of texture information, and then map it onto the surface during rendering. The other rely on the use of hierarchical B-spline, but are not sufficiently generalized to work on arbitrary complexity.

2.5.3 Polygon soup representation

From the input MVD data, [16] introduced a polygon soup representation, which take the advantage of polygonal primitives: compactness, surface continuity, and graphic processor compatibility. They remain defined in 2D for compression efficiency and possible backward compatibility, but replace depth maps.

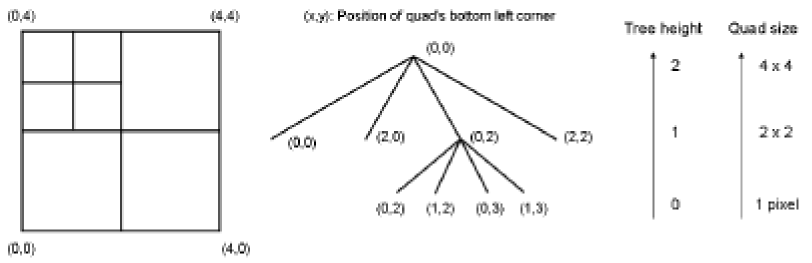


Figure 2.13: Example of image decomposition and quadtree structure. Each level of the quadtree fives the size od the quads and each node gives the position of the bottom left corner of the quads (courtesy of T.Colleu)

The 3D polygon soup is composed of polygons stored in 2D with their depth values at each corner. These 2D polygons are extracted from the depth maps using a quadtree decomposition method for each view (fig.2.13).

This decomposition per view provides an accurate model of associated depth map, while preserving discontinuities : created quads are recursively divided if they contains depth discontinuity or a bad approximation of the original depth. From this preliminary soup of polygon, an inter-view redun-

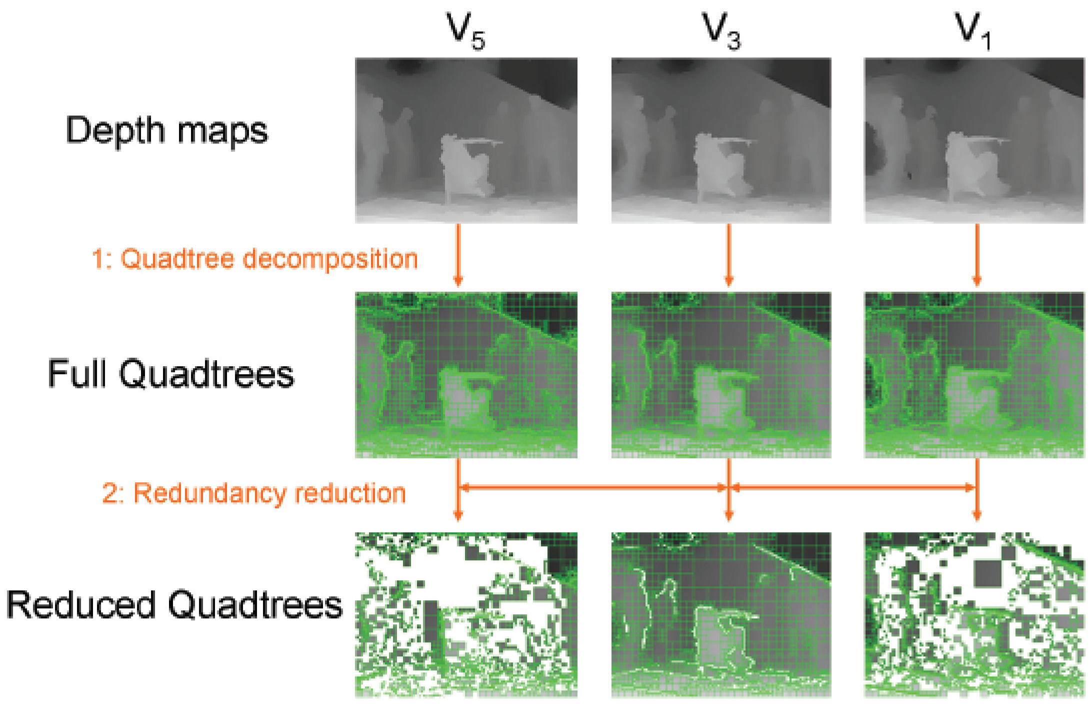


Figure 2.14: Overview of the Polygon soup construction method (courtesy of T.Colleu)

dancy reduction step eliminates identical or inadequate quads, while increasing the compactness (as illustrated in fig.2.14).

At the rendering stage of synthesized view, a triangulation and quadtree restriction enables crack elimination, while a multiview adaptive blending (based on view-dependent texture mapping) smooth color and geometry inconsistencies. Ghosting artifacts having been eliminated during the quadtree reduction process, the virtual view is finally inpainted on unknown pixels and edge-filtered on object boundaries to provide a natural appearance. The polygon soup representation while preserving rendering quality, transfer the complexity to the construction of the representation. The compactness and low complexity synthesis make from this representation a good candidate for 3DTV.

2.6 Point-based representation

We have seen that triangles (or polygons) can be used in an efficient way for surface representation. Surface points, particles or surfels can also be used as a more basic display primitive for surface representation.

The topology or connectivity isn't explicitly stored, but a set of points sampled from the surface and their surface normals and colors. Then the point-based schemes are not limited by the topology, and can easily be used to represent any complex 3D scene. The first point-based representation has been proposed by [17], but it recently regained attention due to its rendering complexity capacity. As the abilities to acquire more and more meshes put the classic polygonal models to the limit of graphic card capabilities, the rendering of individual points instead of polygon rasterization becomes much more efficient.

Also, the splatting is a common technique to avoid visual artifact like holes (that appears due to projection and grid positionning), aliasing and undersampling effects. Each surface point is associated with an oriented tangential disc: a surfel. The size and shape is changing depending on the type of surface and the local density. The shade or color of the point is warped so that intensity decays in the radial direction from the center. When a single image pixel is influenced by several overlapping splats, the shade of the pixel results in the intensity-weighted average of the splat colors. Different 2D gaussian based filtering finally achieve an high quality rendering of point-based surface models at an interactive frame rate.

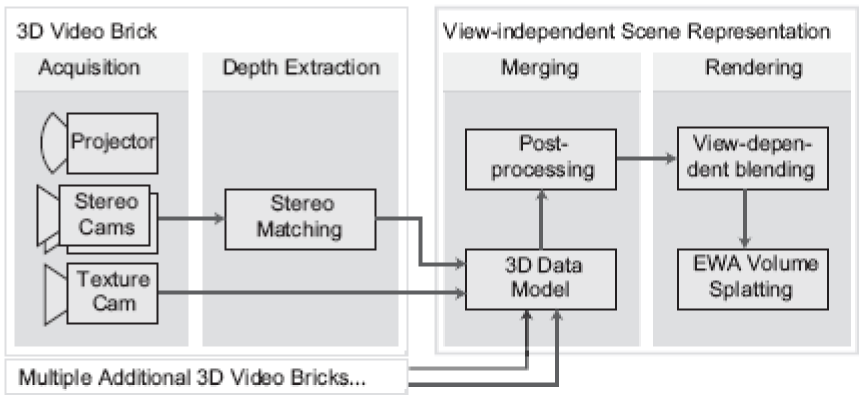
A recent example of 3D video framework relying on point-based representation has been presented by [3]. The acquisition part is composed of multiple 3D video bricks containing a projector, two grayscale cameras and a high-resolution color camera (as illustrated in figure 2.15.

The depth calculation is aided by structured light patterns projected on the surfaces of the scene. Textures images and pattern-augmented views of the scene are acquired simultaneously by time multiplexed projections of patterns and camera exposures.

Then, the depth maps are extracted using stereo matching on the acquired pattern images. Each surface sample corresponding to depth value are merged into a view-independent point-based 3D data structure. Each point is modeled by a 3D gaussian ellipsoid, and the resulting point-cloud is post-processed to remove outliers and artifacts. At the rendering stage, enhanced probabilistic EWA volume splatting and view-dependent blending lead to high quality synthesized views.

2.7 Volumetric representation

This consists in decomposition in volume units or primitives of the 3D space, thee discrete model and primitive-based model respectively. The primitive based models are the extension of surface parameterization to volume, and



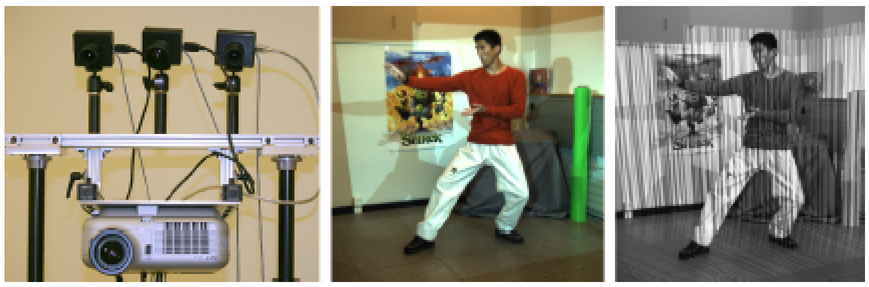


Figure 2.15: Overview of the Waschbüch 3D video framework (top) and illustration of a brick(bottom left), simultaneously acquiring textures (middle and structured light patterns(right), from [3]

rely on cylinders, superquadrics, supershapes or hyperquadrics and other polynomial models combined with different operations such as graph to form a coherent representation. Despite their advantage in compactness, it is inherently difficult to model arbitrary and natural object through primitives; even if flexibility has been considered (deformation: torsion, etc..), this primitive based representation is more dedicated to CAD, object extraction or object modeling.

Discrete model decompose and segment the 3D space into volume unit called a voxel. The set of voxel constitutes the reconstructed volume in a world reference frame, but there exists different ways to organize this set. Each voxel contains the properties of a surface segment within it, and empty voxel can be either flagged as empty, either not represented at all. This last method results in the **octree** structure [18],2.16 . Each octree codes the areas that contain surfaces only. The data structure is organized as a tree growing in depth at the nodes that correspond to occupied voxels. Then octrees show efficiency in term of memory space and are a common method to store voxel in term of compression and ease of implementation.

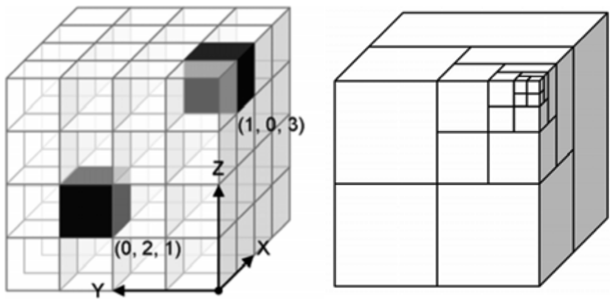


Figure 2.16: Illustration of data structures implementing volumetric representations, voxel buffer(left) and octree (right)

The main advantages of the voxel representation in the multiview video context is that it provides a common reference frame where the surface information obtained from different views are combined. It also enables direct -and then fast- access to neighboring voxels for the rendering, as this neighborhood is coded into the tree. This facilitates the computation of voxel visibility, the absence of potential voxel between camera and image surface. This also helps in 3D convolution operations and detection of connected components, used for local geometric properties calculation, and noise-filtering operations.

In conclusion, advantages of voxel-based representations are numerous, linear access time to the structured data, and then complexity-independent rendering. But the cube unity results in low quality rendering when the camera become too close to the cube based surface, which is not the case with polygons. The conversion from a volumetric representation to a mesh-based representation at the rendering stage is under investigation by the computer vision community. Radial basis function (RBF), [19]could be applied on voxel representations, as it shows promising results on surface interpolation.

Chapter 3

Compression

3.1 Introduction

In this section, we describe the coding standards for encoding 3D video data whose different representations have been introduced above. Previous work on video based rendering has shown the need of multiple views in order to achieve the best image quality, especially because of the occlusion problem, for 3D video services. This makes the duty of compression even more challenging. Although many approaches are focusing on capturing, modeling and rendering, the task of efficiently representing the large amount of 3D video data in order to efficiently coding is less emphasized.

We will first present the fundamental principles of a video codec because it is of relevant importance since most of the standards for encoding 3D video data are inspired from 2D video codecs. Then the coding standards of the previously introduced representations will be developed.

3.2 Principles

Most of the conventional video codecs [20] reduce the amount of data to be transmitted by a combination of temporal and spatial prediction, also called DPCM/DCT design. The temporal prediction, called DPCM (Differential Pulse Code Modulation) uses previous samples to encode a residual. The spatial transformation operator is the Discrete Cosinus Transform (DCT) which uses the values of a single block for encoding. The need for a combination of both temporal and spatial prediction can easily be explained: neighboring pixels can be very similar, in homogeneous areas while temporally adjacent frames are often very correlated. After a quantization step, the bit stream is generated following an entropy encoding.

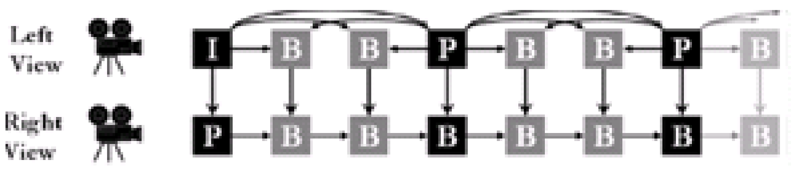


Figure 3.1: Prediction Structure in MPEG-2 MVP

The aim of compression is to reduce the amount of data to be transmitted at the cost of a minimum deterioration of the information. So, we need to quantify this distortion. The most reliable measure is still the subjective evaluation by a set of observers. Yet, the human visual system is very complex, and thus, an objective model of the perceived quality is very difficult to process. Though it has shown some limitations (because it does not express correctly the human perception), the most used metric is the PSNR (Peak-Signal-to-Noise-Ratio).

3.3 Stereoscopic Video

2D video encoders use motion estimation and motion compensation as tools for temporal prediction. In the case of stereoscopic video, disparity estimation and compensation are used. Indeed, geometry information can be found thanks to the application of projective geometry on both images of the stereoscopic pair. The displacement or disparity between the images of the stereoscopic pair is equivalent to a dense motion field between two images of a conventional video sequence. For this reason, encoders use this disparity, for image prediction.

A standard specification has been defined in ITU-T Rec. H.262/ISO/IEC 13818-2 MPEG-2 Video: the multi-view Profile (MVP)[21].. As seen on the illustration (fig. 3.1), the left eye is encoded as a key sequence, i.e. with no reference to the right eye sequence. Then, the right eye is predicted thanks to the left eye sequence. This structure uses conventional temporal prediction, and inter-view prediction. However, there is no significant gain compared to simulcast coding (separate encoding of each view), because temporal prediction is much efficient than inter-view prediction (i.e. disparity-compensated prediction) as observed in [21].

Also, more recently, the H.264/MPEG-4 AVC standard [22]can be used. It achieves higher coding efficiency thanks to many improvements: variable block sizes for the temporal prediction, intra prediction, multiple reference frames, etc. This standard allows the use of both temporal and inter-view prediction, in a hierarchical way.

3.4 2D+Z

As explained in 2.4.1, the video-plus-depth data representation has some advantages, compared to the stereoscopic video: a stereo pair can be rendered at the receiver side thanks to the geometry information from the depth signal. MPEG specified a standard for this type of data, known as MPEG-C part 3(or ISO/IEC 23002-3, [23]).There is no novel algorithm since the texture video and the depth video are encoded as MPEG-2 streams, or MPEG-4 streams, and with an auxiliary container for depth information (such as Zfar and Znear). This way, the decoder interprets the two incoming video streams as color and depth. It has been claimed, from the European ATTEST project that this depth signal can be efficiently compressed. Being smoother and less textured than the color signal, the depth signal needs only 10 to 20% of the bit rate of the texture video to be encoded at a good quality ([24]).

3.5 MVV

Section 2.4 presented MVV as a set of multiple conventional video sequences of the same scene, captured from different viewpoints. Compared to stereoscopic video, the amount of data to process is thus significantly higher. Hence, the use of the temporal and inter-view correlations cannot be ignored to design an efficient compression. The JVT is currently developing an extension of H.264/MPEG-4 AVC standard, known as MVC extension (multi-view Video Coding). It exploits all statistical dependencies within the multi-view video data set and thus uses temporal and inter-view prediction structures (see Figure 3.2). Previous work on analysis on temporal and inter-view dependencies was carried out and showed that on a RD criterion, the temporal prediction mode was the most chosen [25].

MVC has been proved to outperform simulcast coding, with objective and subjective measurements as supporting evidence. Yet, the studies have also shown that this gain is very dependent to the content of the sequences (complexity due to motion, details) and the configuration of the acquisition [26].

Given N views at the receiver side, N-1 pairs of stereoscopic views can be rendered on an auto-stereoscopic display. Thus, in this case, head motion parallax is supported. But the price of such an advantage is the complexity of the MVC standard. This implies computational complexity, memory re-

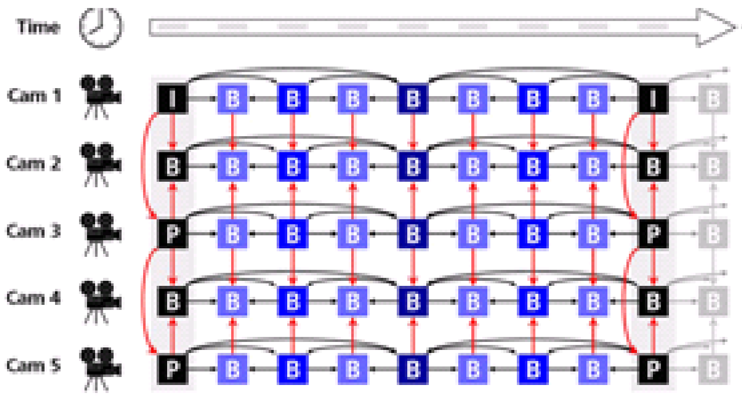


Figure 3.2: MVC Structure

quirements and delay, [27]. Currently, it is the reference standard for MVV coding.

3.6 MVD

For multi-view video plus depth data (MVD), MVC can be used for the compression of the texture sequences and the depth sequences, as two set of sequences. Thus, the combined information of the depth and the texture is not used. The relationship between texture and depth is still under investigation, and work has been done in this field to evaluate the influence of depth coding on view synthesis in [28] and to improve the coding performance of MVC, in [1] and [29].

In [29], considering the results from [30] (the use of temporal prediction and spatial prediction, i.e. inter-view depends on the multi-view sequences properties like the baseline-camera distance), Morvan proposes a prediction structure that exploits either temporal or spatial prediction, according to a rate-distortion criterion. It uses view synthesis as a prediction tool. The method is used for texture compression and depth compression. Compared to H.264/MPEG-4 AVC using only a block-based disparity-compensated prediction, there is no significant gain for texture encoding; but the proposed method outperforms H.264 (in the configuration mentioned above) for depth encoding.

In [28], the authors investigate the effects of MVD compression on the rendered views. The adaptive used method (platelets) show better results than H.264/MVC method because it preserves better the depth maps edges.

In [1], an H.264-based algorithm improves rate-distortion (RD) performances by using a dense motion/disparity estimation framework by a RD driven segmentation and coding (in order to replace the block-based motion/disparity estimation step).

3.7 LDV

As explained in the previous chapter, layered depth video (LDV) is an alternative representation of MVD. Because they are made as images, the data stored can be processed by a video codec. Most of the algorithms are MVC based, and often variant of H.264/ AVC video coding standard, as in [31] where after generating the LDI, three types of data are encoded (the colour, the depth and the image of number of layers per pixels). This representation can achieve significant gain while it uses the texture and depth information altogether. Because of the special characteristics of the LDI, the number of layers may be lower than the number of existing sequences, which is an advantage compared to MVD. And the more layers we have (whose maximum corresponds to the maximum number of views), the less pixels are stored in the back layers. This can allow a big difference of gain compared to an MVC algorithm processing MVD data.

Some encoders, would have difficulties when processing a holed image, like the back layers ones. In [31], two alternatives have been tested: one consists in filling the holes with the pixels of the first layer and then eliminate then automatically, knowing the number of layers per pixel (with the image of number of layers); the other suggests to aggregate the pixel horizontally, and then aggregate both layers.

3.8 Mesh compression

3D meshes can be used in 3D video representations. An object's surface is approximated by a set of connected triangles. Depending on the desired resolution, i.e. level of accuracy and thus level of realism, the number of triangles may be consequent and make it expensive to store and transmit. Dynamic meshes represent a sequence of static meshes. Static meshes can be seen as a connectivity (triangulation of the mesh vertices) or a geometry (each vertex is assigned a 3D location). Dynamic meshes require an efficient compression method that exploits the spatial and temporal geometrical dependencies. Many algorithms have been proposed to compress 3D meshes efficiently since early 1990s

Single-rate compression designs the fact that all connectivity and geometry data are compressed and decompressed as a whole. The original mesh is not rendered until the whole stream is received. Connectivity and geometry data are separately encoded. Geometry data is more bit-demanding than connectivity.

Progressive compression allow coarse to fine decoding, while the stream is received.

3.8.1 Static meshes

3D meshes can be represented in VRML[32] ASCII format and efficiently compressed by the method proposed by Taubin et al. in [33], based on the the topological surgery (TS) algorithm [34]. Then Taubin et al. introduced progressive decoding by the Progressive Forest Split (PFS) to describe connectivity in different resolution levels. TS and PFS are the basis of the MPEG-4- 3DMC (3D Mesh Coding) standard. The mesh connectivity can be described by a state machine [35]: each time a vertex is visited, one out of five symbols is emitted. The symbol represents the the configuration of this vertex according to the traversed region. the stream of symbols is entropy encoded. In [36], a predictor of vertex location is proposed: the parallelogram predictor. Touma and Gotsman also introduced the notion of valence by the valence-based technique (TG) for mesh connectivity compression. The processed vertices are assigned valences. For geometry compression there exist methods based on the wavelet approach [37] or on spectral decomposition [38]. Then we can cite the image-based techniques that were introduced for more coding efficiency of the geometry. They are called Geometry Images [39]. It is a lossy coding method based on the remeshing of the triangular mesh on a grid, the image. The RGB channels of the image represent the geometry. The image is encoded by common image codecs.

3.8.2 Dynamic meshes

Most of the coding methods for dynamic meshes exploits the dependencies in spatial domain and temporal domain. The methods are classified into two categories: transform dominated and prediction dominated. The transform dominated methods models vertices trajectories by approximating the global behavior of a group of vertices through time. In was first introduced by Lengyel in 1999 [40]. Other methods uses principal component analysis and linear predictors [41, 42]. As for static meshes, wavelet methods are also employed for compression. As well the extension of Geometry Images is Geometry Videos [43]. The method of skinning animation [44] should be pointed out. It is based on vertex clustering and weighted affine transforms. For predicition dominated methods, the parallelogram predictor is widely used. AFX-IC method encode only a few frames and interpolates the missing ones [45]. D3DMC (octree based motion vector clustering) and D3DMC-RD (D3DMC with rate distortion criterion) [46] are coding methods based octree clustering for grouping motion vectors.

Chapter 4

Conclusion

It is generally accepted that the coding gain can be obtained by exploiting images correlation. Disparity-compensated prediction is typically as an exploitation of inter-view correlation. Most of the proposed algorithms are 2D-video-codecs-inspired. It seems that 3D information could be exploited in both texture and depth sequences, in order to achieve a better 3D perceived quality.

Methods are still under investigation for MVD data compression. From the proposed algorithms, compression performances are still dependent on video contents.

Moreover, because normal 2D TV and HDTV applications are still dominating the market, the development of compression algorithm can be restrained by the backward compatibility constraint.

Another issue concerns the bit-rate allocation for each type of data. The problem of measuring and optimizing the visual quality of encoded 3D video is still an open question. In the case of data represented by texture and depth data, the bit-rate to be allocated to depth data and texture data should be chosen as appropriately as possible to ensure a good quality of the virtual generated views at the decoder side. Studies investigating this issue have been published but numbers are controversial.

In [47], the distortion of compressed depth has been studied by compressing multi-view plus depth data at different bit rates, with the MVC method. Then a known view is generated and compared with a view generated from uncompressed data. Comparisons are based on the PSNR score. Results show that depth data seriously impacts the quality of the rendered intermediate view. With lower bit rate (and higher distortion) coding of the depth maps the distortion of rendered intermediate views increases. Distorted areas are especially located around depth discontinuities at the borders of objects with different scene depth. On the other hand the quality of compressed color video influences the quality of rendered intermediate views much less, basically showing well-known effects like a loss of sharpness.

Yet, in [48], a similar experiment (compression of video plus depth data, by varying the bit rate of each type of data and computing an intermediate view from compressed data) assessed the video quality by objective and subjective measures. Objective metrics (VSSIM and PSNR), show different tendencies: VSSIM showed that highest quality can be achieved when 15 to 20% of the total bit budget is used for coding the depth map while PSNR favored higher depth quality. Also, the subjective evaluation show that depth distortion is less important than color distortions, which suggests that the quality estimation method is not perfect.

In numerous studies such as [49], authors propose a joint depth/texture bit-allocation algorithm for the compression of multi-view video. It combines the depth and texture rate-distortion (RD) curves to obtain a single R-D surface that allows the optimization of the joint bit-allocation problem in relation to the obtained rendering quality.

In [50], the bit-rate budget is fixed but images are down-sampled prior to encoding. This is based on the fact that down-sampling an image to a low resolution, then coding at the lower resolution and subsequently interpolating the result to the original resolution can improve the overall PSNR performance of the compression process. results show a gain of 33% of the total rate.

Bibliography

[1] Ismael Daribo, *Codage et rendu de sequence video 3D; et applications Ã la television tridimensionnelle (TV 3D) et a la television Ã base de rendu de videos (FTV).*, Ph.D. thesis, Telecom ParisTech, Nov. 2009. 5, 12, 13, 32

[2] P. Kauff, M. Müller, F.Zilly, A.Smolic, and C.Varekamp, “Deliverable d.2.1.2 : Requirements on post-production and formats conversion,” Tech. Rep., 3D4YOU consortium, 2008, public deliverable D1.1.2 available on the projects website: www.3d4you.eu. 5, 17

[3] M. Waschbüsch, S. Würmlin, D. Cotting, and M. Gross, “Point-sampled 3d video of real-world scenes,” *Signal Processing: Image Communication*, vol. 22, no. 2, pp. 203–216, 2007. 6, 26, 27

[4] W. Matusik and H. Pfister, “3d tv: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes,” in *ACM SIGGRAPH 2004 Papers*. ACM, 2004, pp. 814–824. 13

[5] E.H. Adelson and J.R. Bergen, “The plenoptic function and the elements of early vision,” *Computational models of visual processing*, vol. 1, 1991. 14

[6] M. Levoy and P. Hanrahan, “Light field rendering,” in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 1996, p. 42, light field. 14

[7] C.L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, “High-quality video view interpolation using a layered representation,” in *International Conference on Computer Graphics and Interactive Techniques*. ACM New York, NY, USA, 2004, pp. 600–608, LDI. 18

[8] J. Shade, S. Gortler, L. He, and R. Szeliski, “Layered depth images,” in *Proceedings of the 25th annual conference on Computer graphics and*

interactive techniques. ACM New York, NY, USA, 1998, pp. 231–242, LDI. 18

[9] S.U. Yoon, E.K. Lee, S.Y. Kim, Y.S. Ho, K. Yun, S. Cho, and N. Hur, “Coding of layered depth images representing multiple viewpoint video,” in *Proc. of Picture Coding Symposium (PCS) SS3-2*, 2006, LDI. 18

[10] X. Cheng, L. Sun, and S. Yang, “Generation of layered depth images from multi-view video,” in *IEEE International Conference on Image Processing, 2007. ICIP 2007*, 2007, vol. 5, LDI. 19

[11] V. Jantet, L. Morin, and C. Guillemot, “Incremental-ldi for multi-view coding,” 2008, LDI. 19

[12] WHA Bruls and R.K. Gunnewiek, “Options for a new efficient, compatible, flexible, 3d standard,” in *IEEE International Conference on Image Processing, 2009. ICIP 2009*, 2009, LDVR. 20

[13] A. Smolic, K. Mueller, P. Merkle, P. Kauff, and T. Wiegand, “An overview of available and emerging 3d video formats and depth enhanced stereo as efficient generic solution,” in *Proc. PCS*, 2009, 3D rep. 20

[14] H. Hoppe, “View-dependent refinement of progressive meshes,” in *Proc. Siggraph*, 1997, vol. 97, pp. 189–198. 22

[15] S. Kircher and M. Garland, “Progressive multiresolution meshes for deforming surfaces,” in *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation.* ACM, 2005, pp. 191–200. 22

[16] T. Colleu, S. Pateux, L. Morin, and C. Labit, “A polygon soup representation for free viewpoint video,” in *Proceedings of SPIE*, 2010, vol. 7526, p. 75260H. 24

[17] M. Levoy and T. Whitted, “The use of points as a display primitive,” *Tech. Report85-022, University of North Carolina at Chapel Hill*, 1985, Point based. 26

[18] M. Gervautz and W. Purgathofer, “A simple method for color quantization: Octree quantization,” in *Graphics Gems.* Academic Press Professional, Inc., 1990, pp. 287–293. 27

[19] JC Carr, RK Beatson, JB Cherrie, TJ Mitchell, WR Fright, BC McCallum, and TR Evans, “Reconstruction and representation of 3D objects

with radial basis functions," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 2001, p. 76. 28

[20] Peter D. Symes, *Digital video compression*, McGraw-Hill Professional, 2004. 29

[21] J. R Ohm, "Stereo/multiview video encoding using the MPEG family of standards," *Invited Paper, Electronic Imaging*, vol. 99, 1999. 30

[22] ITU-T Recommendation H.264, "Advanced video coding for generic Audio-Visual services," http://ip.hhi.de/imagecom_G1/assets/pdfs/h264_multi_view.pdf, 2009. 30

[23] "ISO - ISO standards - JTC 1/SC 29 - coding of audio, picture, multimedia and hypermedia information," http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_tc_browse.htm?commid=4531 31

[24] A. Smolic, K. Mueller, N. Stefanoski, J. Ostermann, A. Gotchev, G. B Akar, G. Triantafyllidis, and A. Koz, "Coding algorithms for 3DTV survey," *IEEE transactions on circuits and systems for video technology*, vol. 17, no. 11, pp. 1606–1620, 2007. 31

[25] P. Merkle, K. Muller, A. Smolic, and T. Wiegand, "Efficient compression of multi-view video exploiting inter-view dependencies based on h. 264/MPEG4-AVC," in *Proc. ICME*, 2006, pp. 9–12. 31

[26] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Efficient prediction structures for multiview video coding," *IEEE Transactions on circuits and systems for video technology*, vol. 17, no. 11, pp. 1461–1473, 2007. 31

[27] Y. Chen, Y. K Wang, K. Ugur, M. M Hannuksela, J. Lainema, and M. Gabbouj, "The emerging MVC standard for 3D video services," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, no. 1, 2009. 32

[28] P Merkle, Yannick Morvan, Aljoscha Smolic, Dirk Farin, Karsten Muller, and Thomas Wiegand, "The effects of multiview depth video compression on multiview rendering," *Singal Processing: Image Communication*, vol. 24, no. 1-2, pp. 73–88, 2009. 32

[29] Yannick Morvan, *Acquisition, compression and rendering of depth and texture for multi-view video*, Ph.D. thesis, 2009. 32

[30] A. Kaup and U. Fecker, “Analysis of multi-reference block matching for multi-view video coding,” in *Proc. 7th Workshop Digital Broadcasting*, pp. 33–39. 32

[31] S. U Yoon and Y. S Ho, “Multiple color and depth video coding using a hierarchical representation,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1450–1460, 2007. 33

[32] R. Carey, G. Bell, and C. Marrin, “Iso/iec 14772-1: 1997 virtual reality modeling language (vrml97),” *The VRML Consortium Incorporated*, 1997. 34

[33] G. Taubin, W. P Horn, F. Lazarus, and J. Rossignac, “Geometry coding and VRML,” *Proceedings of the IEEE*, vol. 86, no. 6, pp. 1228–1243, 1998. 34

[34] G. Taubin and J. Rossignac, “Geometric compression through topological surgery,” *ACM transactions on Graphics*, vol. 17, no. 2, pp. 84–115, 1998. 34

[35] J. Rossignac, “Edgebreaker: Connectivity compression for triangle meshes,” *IEEE transactions on visualization and computer graphics*, vol. 5, no. 1, pp. 47–61, 1999. 34

[36] C. Touma and C. Gotsman, *Triangle mesh compression*, Google Patents, Dec. 2000, US Patent 6,167,159. 34

[37] A. Khodakovsky, P. Schroder, and W. Sweldens, “Progressive geometry compression,” in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, p. 278. 34

[38] Z. Karni and C. Gotsman, “Spectral compression of mesh geometry,” in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 279–286. 34

[39] X. Gu, S. J Gortler, and H. Hoppe, “Geometry images,” *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 355–361, 2002. 34

[40] J. E Lengyel, “Compression of time-dependent geometry,” in *Proceedings of the 1999 symposium on Interactive 3D graphics*, 1999, pp. 89–95. 35

[41] Z. Karni and C. Gotsman, “Compression of soft-body animation sequences,” *Computers & Graphics*, vol. 28, no. 1, pp. 25–34, 2004. 35

[42] M. Sattler, R. Sarlette, and R. Klein, “Simple and efficient compression of animation sequences,” in *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2005, p. 217. 35

[43] H. M Brice, P. V Sander, L. McMillan, S. Gortler, and H. Hoppe, “Geometry videos: a new representation for 3D animations,” in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2003, p. 146. 35

[44] K. Mamou, T. Zaharia, and F. PrÃªteux, “A skinning approach for dynamic 3D mesh compression,” *Computer Animation and Virtual Worlds*, vol. 17, no. 3-4, pp. 337–346, 2006. 35

[45] E. S Jang, J. D.K Kim, S. Y Jung, M. J Han, S. O Woo, and S. J Lee, “Interpolator data compression for MPEG-4 animation,” *IEEE transactions on Circuits and Systems for Video Technology*, vol. 14, no. 7, pp. 989–1008, 2004. 35

[46] N. Stefanoski, X. Liu, P. Klie, and J. Ostermann, “Scalable linear predictive coding of time-consistent 3d mesh sequences,” in *3DTV Conference, 2007*, 2007, pp. 1–4. 35

[47] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, “Multi-view video plus depth representation and coding,” in *Proceedings of ICIP*, 2007, pp. 201–204. 37

[48] A. Tikanmaki, A. Gotchev, A. Smolic, and K. Muller, “Quality assessment of 3D video in rate allocation experiments,” in *IEEE Int. Symposium on Consumer Electronics (14-16 April, Algarve, Portugal)*, 2008. 38

[49] Y. Morvan, D. Farin, and P.H.N. de With, “Joint depth/texture bit-allocation for multi-view video compression,” *Picture Coding Symposium*, vol. 10, no. 1.66, pp. 4349, 2008. 38

[50] E. Ekmekcioglu, S. T Worrall, and A. M Kondoz, “Bit-rate adaptive down-sampling for the coding of multi-view video with depth information,” in *3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video, 2008*, 2008, pp. 137–140. 38