

# ***MPEG 3D Graphics Coding updates***

**September 2021**

**Marius PREDA**

**MPEG 3D Graphics Convenor**

**Institut Polytechnique de  
Paris, FRANCE**



**What is MPEG 3D Graphics Coding and why it exists**

**What MPEG 3D Graphics Coding did in the past**

**What MPEG 3D Graphics Coding is doing now**

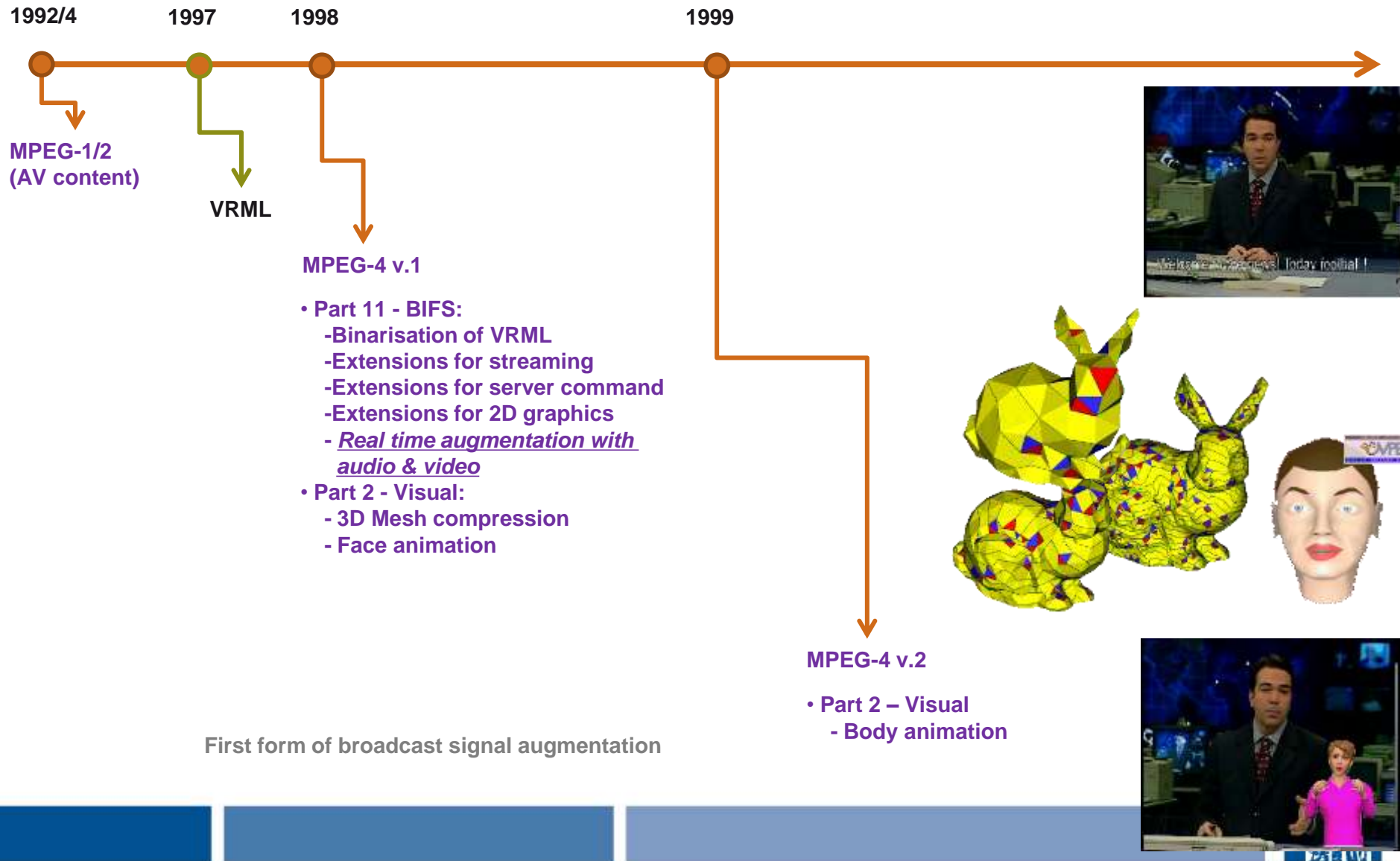
# What is MPEG 3D Graphics Coding and why it exists

- **MPEG 3D Graphics Coding : an ISO working group (within SC 29) aiming at developing standards for graphics **compression****
  - Application agnostic
  - For a variety of raw data representation models
  - Applying traditional signal processing (prediction, frequency transformations, quantisation, entropy encoding)

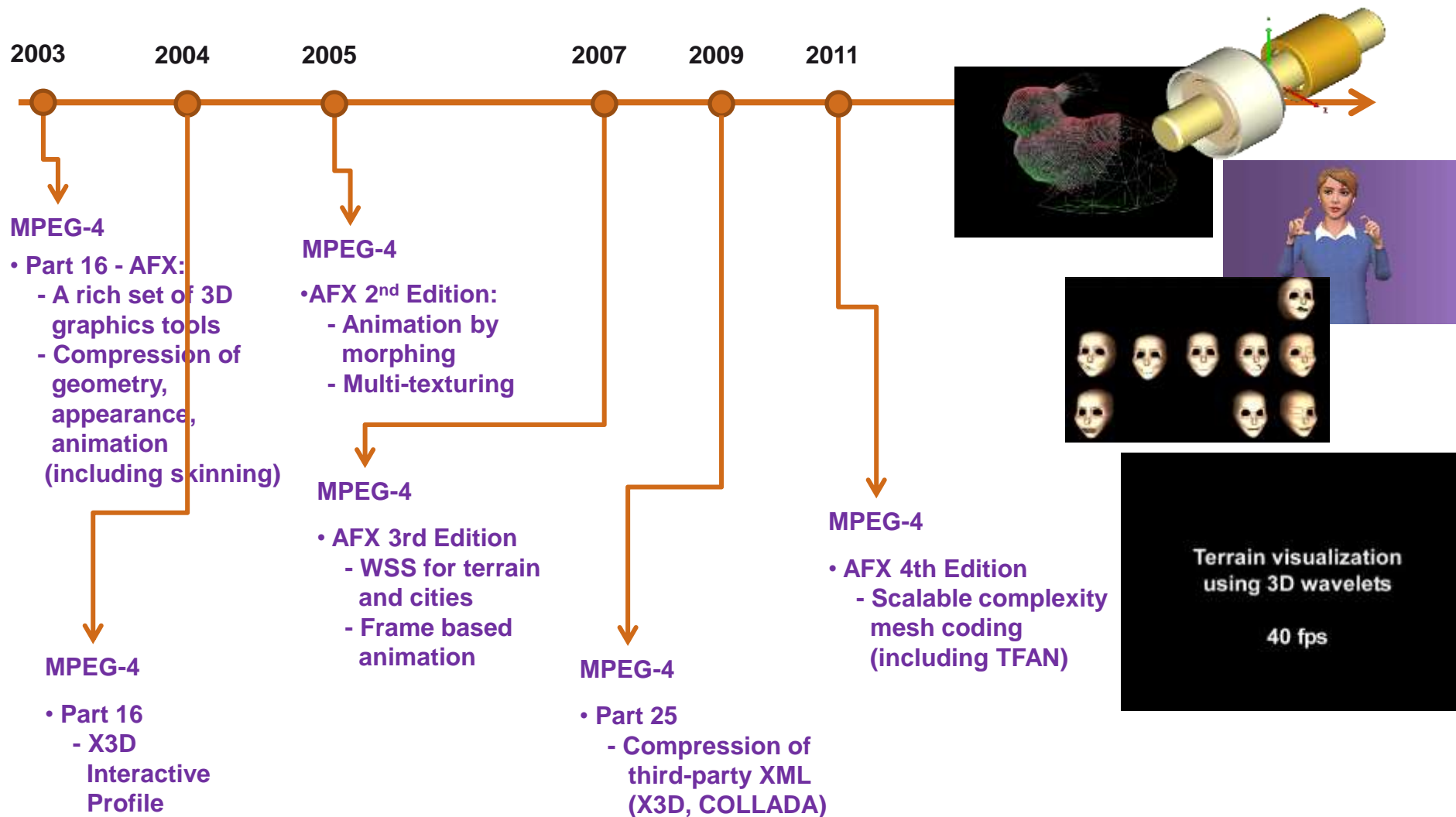
# What is MPEG 3D Graphics Coding and why it exists

- **Because like any other media (e.g. audio, images, videos) 2D and 3D graphics need to be (efficiently) stored and transmitted**
- **However, graphics relies on (mathematical) models**
  - Compression of the model(s) or compression of the results of model interpretation?
- **Difficulties**
  - Variety of the models (different applications, different requirements therefore different representation models)
  - Variety of data type (geometry, material, animation, ...)

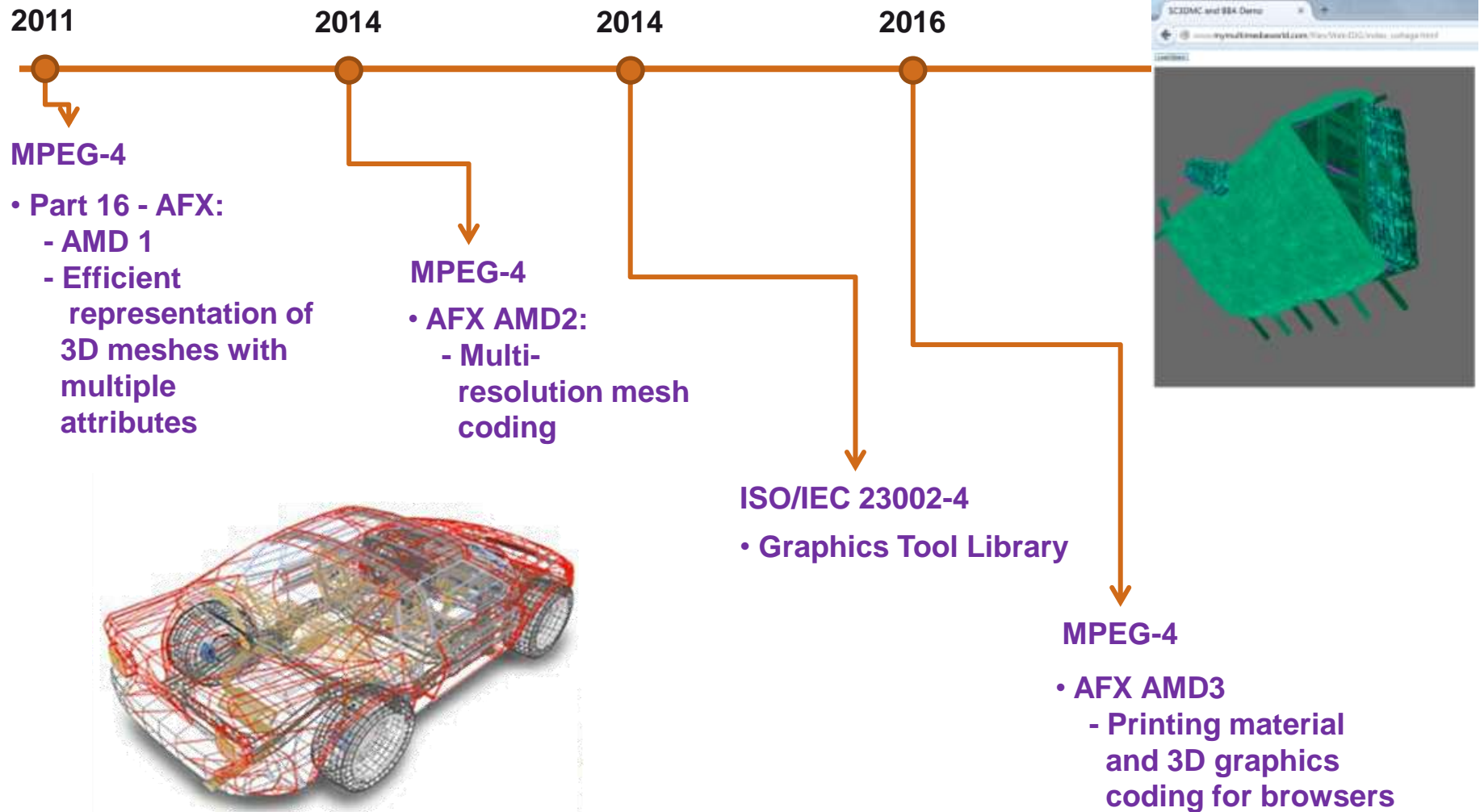
# What MPEG 3D Graphics Coding did in the past



# What MPEG 3D Graphics Coding did in the past



# What MPEG 3D Graphics Coding did in the past



# What MPEG 3D Graphics Coding did in the past

## MPEG-4 Part 16 - AFX (*Animation Framework eXtension*)

### Shapes

- ☐ IFS surfaces
- ☐ Patches
- ☐ Subdivision surf.
- ☐ Wavelet SS
- ☐ Mesh Grid
- ☐ Solids
- ☐ SC-3DMC

### Textures

- ☐ VTC
- ☐ Synthesized texture
- ☐ Procedural texture
- ☐ DIBR
- ☐ Point Texture

### Animation

- ☐ Interpolators
- ☐ Bone-Based
- ☐ Morphing
- ☐ FAMC
- ☐ Remote and programmatic

MPEG 3D Graphics: Encoded binary format for each item

- highly efficient representation
- transmission through various networks and terminal devices
- streaming capabilities



# What MPEG 3D Graphics Coding is doing now

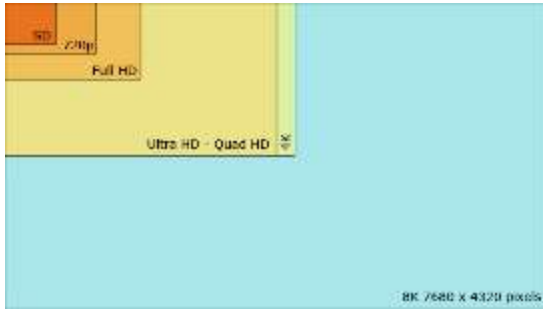
**Focus on applications where  
compression is REALLY needed**

**\*If zip is good enough, then it is out of the scope of  
MPEG 3DGC**

**Focus on applications where  
compression is REALLY needed**

**1<sup>st</sup> use case: “Realistic” graphics or better called  
graphics-based representation of the Reality**

# Capturing more and more from the reality



HD, Full HD, 4K, 8K



LDR, HDR



Multi-camera



Stereoscopy



What is the most appropriate representation form?

- Video + depth
- Many videos + depths
- Point clouds
- Meshes

# What is at the frontier?

## Point Cloud – a convergence between 2 worlds

**Visual  
capture**

- Easy to produce
- High quality

- Interactivity
- Immersion



**Visual  
synthesis**

# Point Cloud

- A set of 3D points
  - not ordered,
  - without relations between them
- Each point is defined by
  - $(X, Y, Z)$
  - $(R, G, B)$  or  $(Y, U, V)$
  - reflectance, transparency, ...

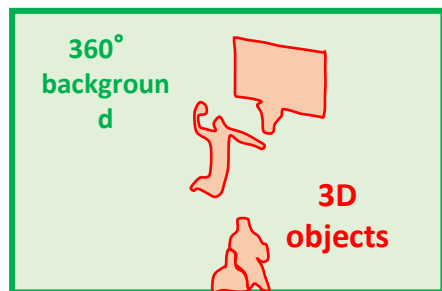


# Point Clouds





# Sport viewing with point clouds



1-3 Gbps per object



# Presence in Augmented Reality



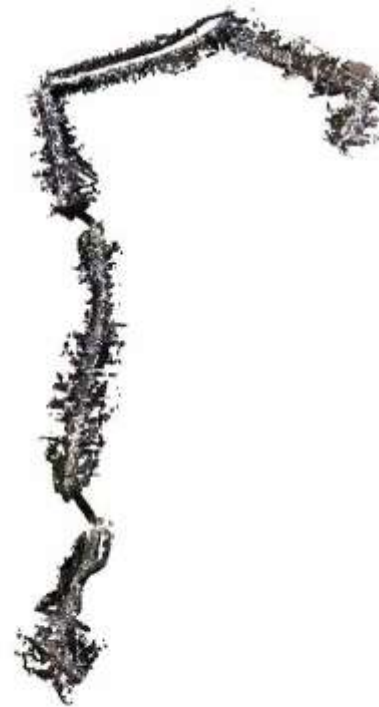
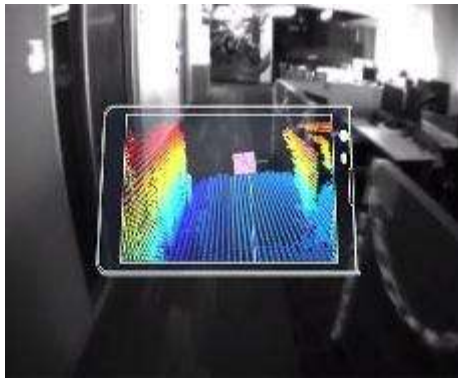
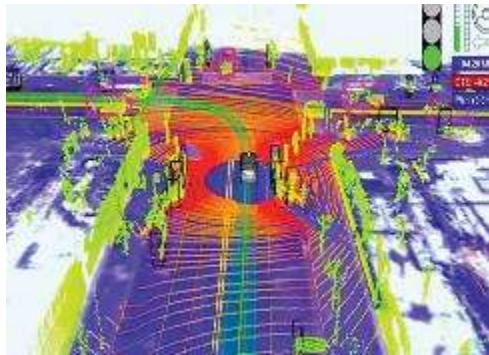
- **Realistic content**
  - spatially collocated,
  - interactive





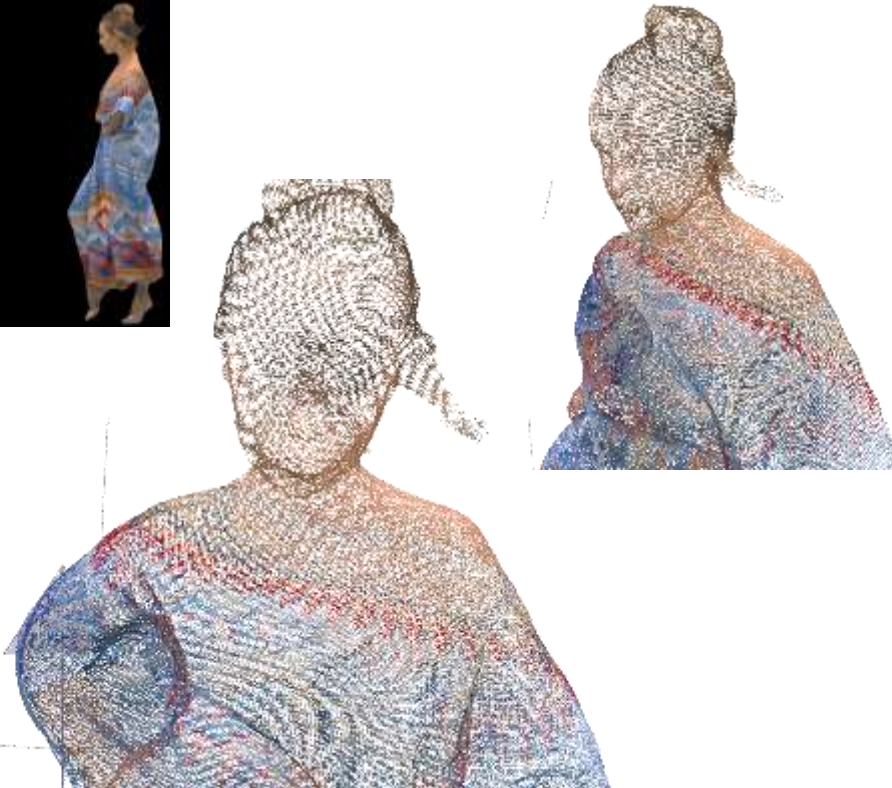
# Environment mapping for autonomous driving

- ~20 million points
  - 2,020,734,515 bytes



# Point Clouds

**800,000 points -> 1 000 Mbps (uncompressed)**

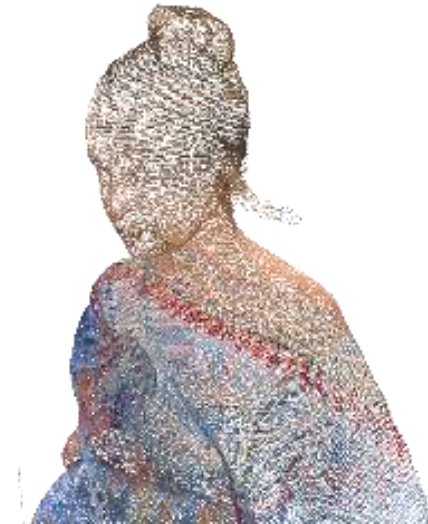
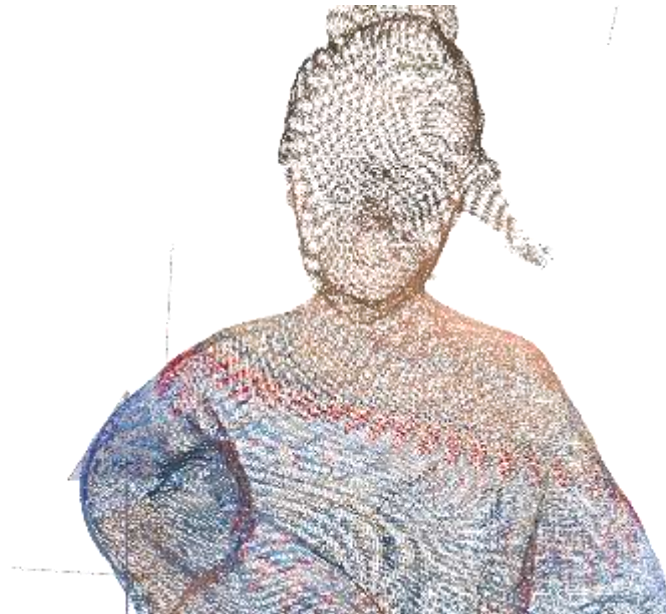


**Compression is required in order to make PC useful**

# Point Clouds Compression – basic principles

Very sparse occupancy of the 3D space

- (usually) the objects are represented by their surface and not by volumes
- In 2D a pixel has 8 neighbors, in 3D - 26 and many of them are transparent

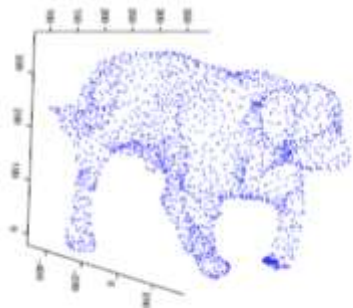




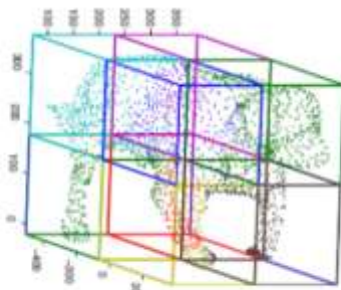
# Point Clouds Compression – basic principles

Special constructs are needed: octree or KD-tree

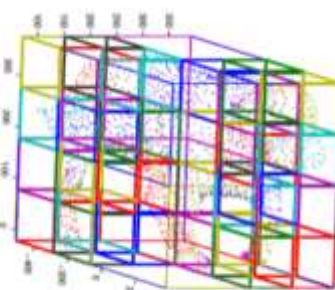
- points grouped into hierarchical structure of branches and leaves
- better difference/residual coding between a representative point and its direct neighbors in a group



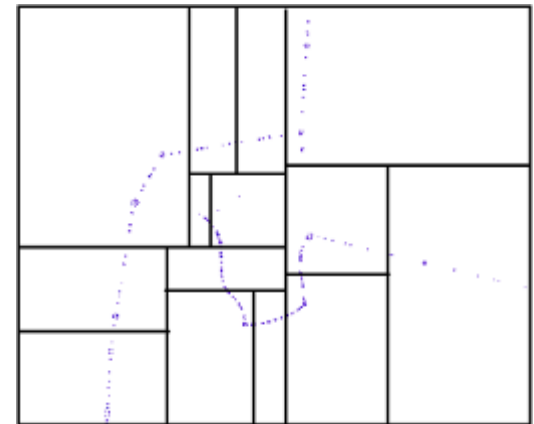
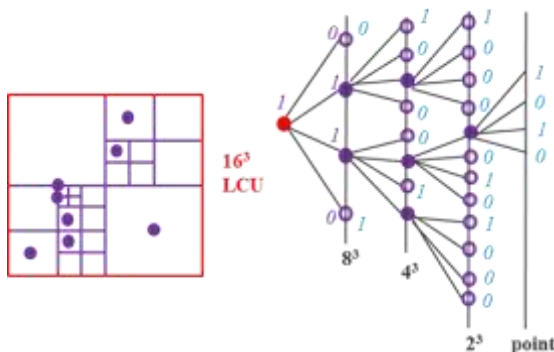
(a) Original point cloud



(b) Depth 1



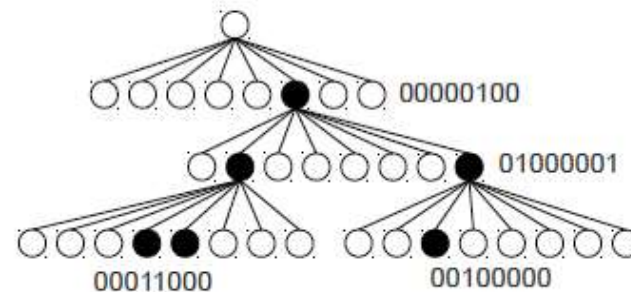
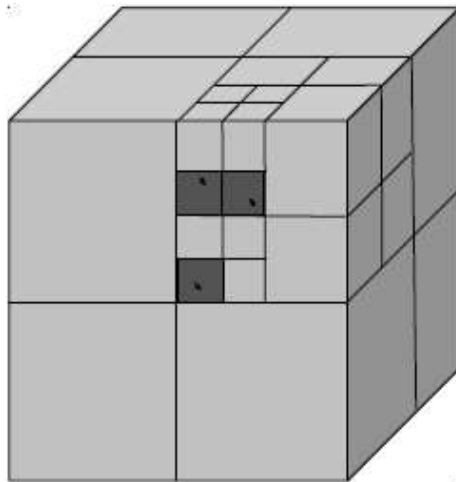
(c) Depth 2



# Point Clouds Compression – basic principles

Octree and KD-tree are great for static scenes

- very difficult to extend their performances to the temporal axis
- leaves that jump from one branch to another in the octree, even after a simple motion



Serialized Octree:  
00000100 01000001 00011000 00100000

# Point Clouds Compression in MPEG

V-PCC  
04/2020

G-PCC  
7/2020

2014

2015

2016

2017

2018

2019

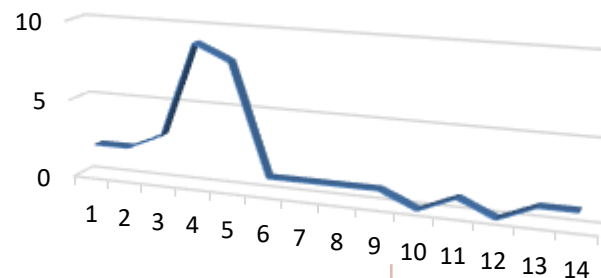
2020

MPEG initiated  
the work on  
PCC

In April 2017 MPEG  
issued a Call for  
Proposals

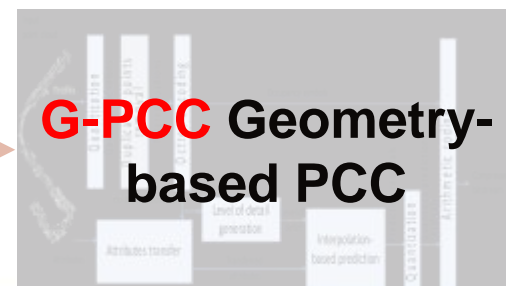
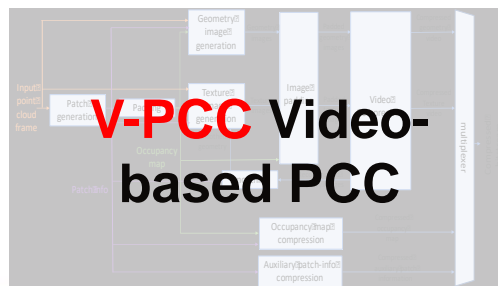
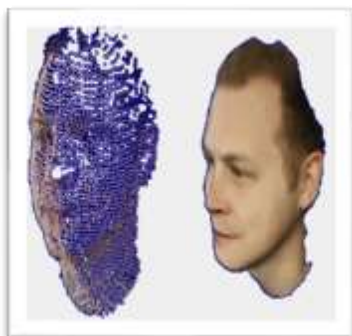
First Committee Draft  
issued in October 2018

9 technology leading companies  
responded and MPEG evaluated them in  
October 2017



**V-PCC Video-**  
**based PCC**

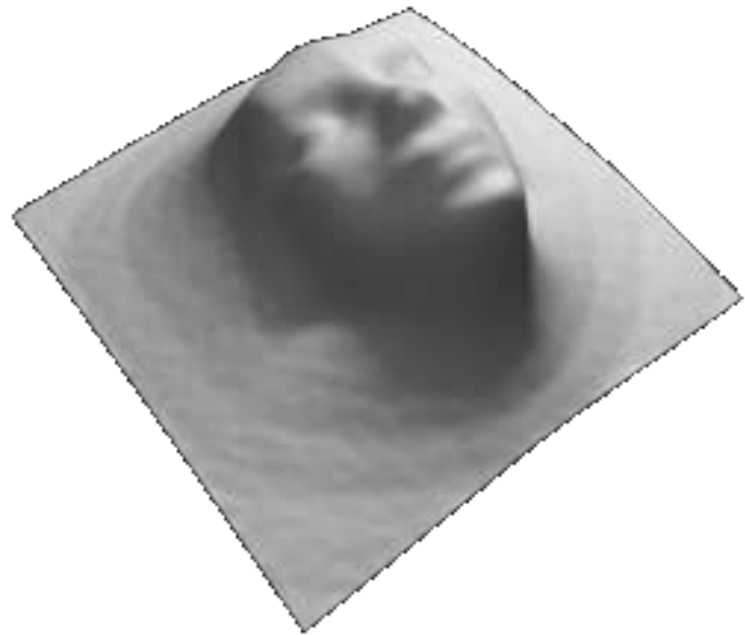
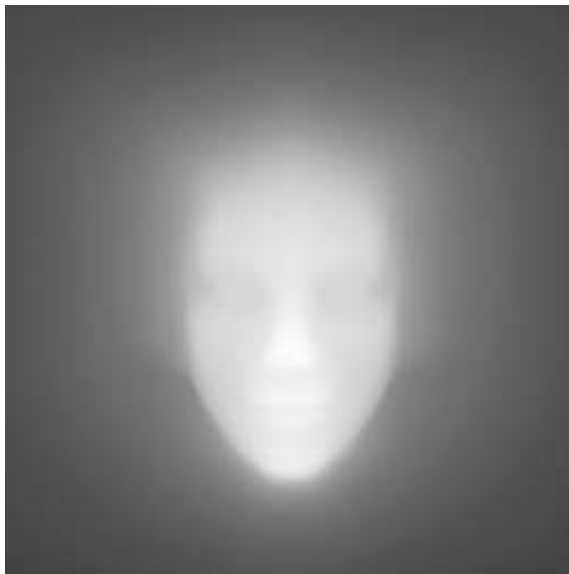
**G-PCC Geometry-**  
**based PCC**



# Video-based Point Clouds Compression

Main ideas:

(1) a point coordinate is encoded as a distance with respect to a particular plane – inspired from the “displacement mapping” in Graphics

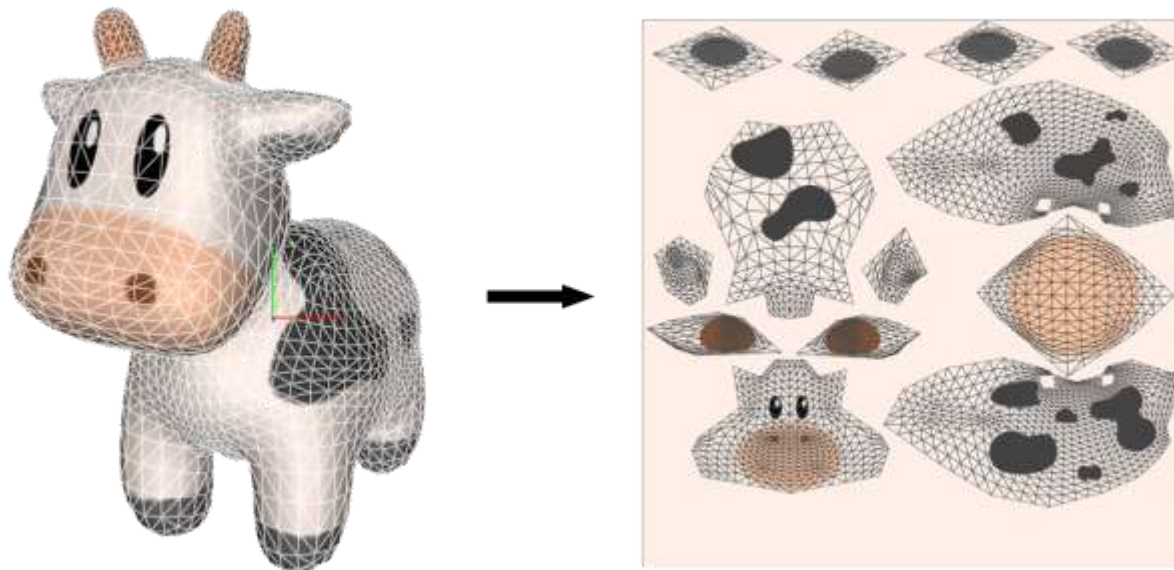


Pixel intensity → Vertex Height

# Video-based Point Clouds Compression

Main ideas:

(2) the color (or any attribute) associated to a 3D vertex is encoded in a 2D texture – inspired from the “texture mapping” in Graphics

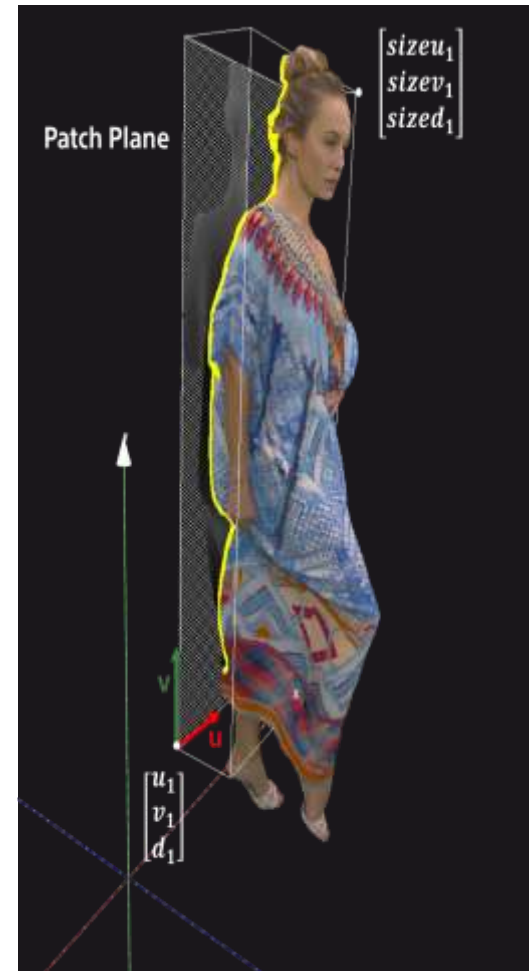


Vertex color ← Pixel color



# Video-based Point Clouds Compression

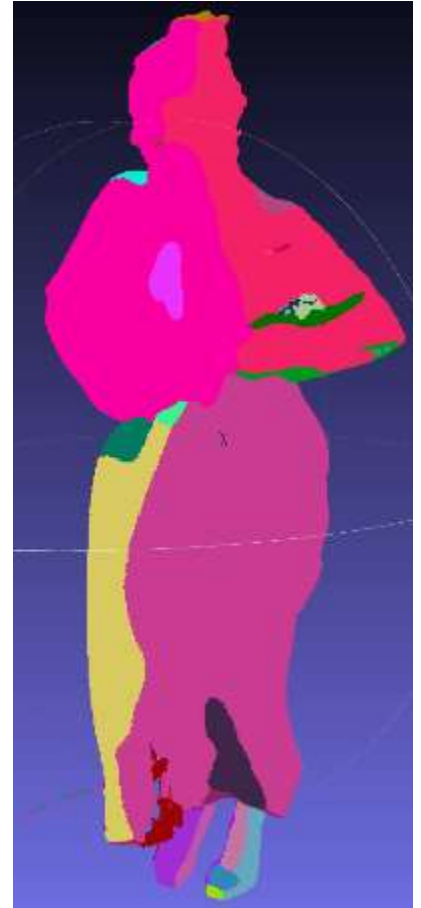
Projecting all the points on a single plane would result to several 3D points having the same 2D projection - > several depth values should be stored per pixel



# Video-based Point Clouds Compression

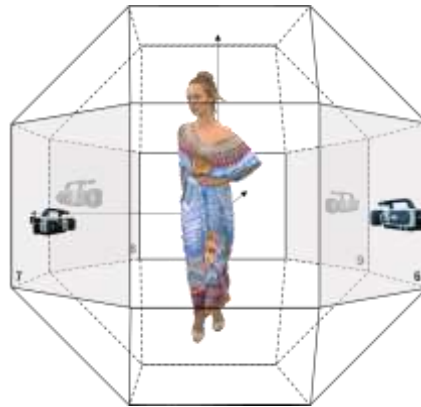
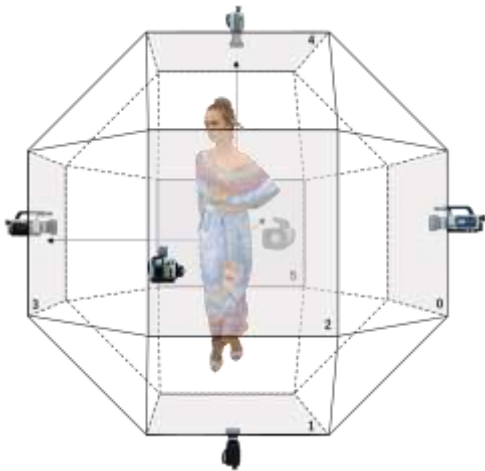
Projecting per patch is preferred:

- A set of points (patch) in a small neighborhood is projected on the same plane
- The set of projection planes is very limited
  - 6 faces of the cube
  - 4 additional diagonal planes

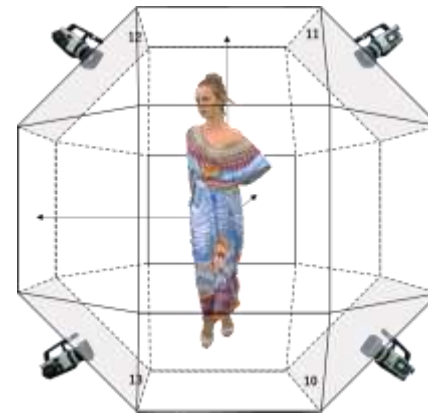


# Projection Orientation

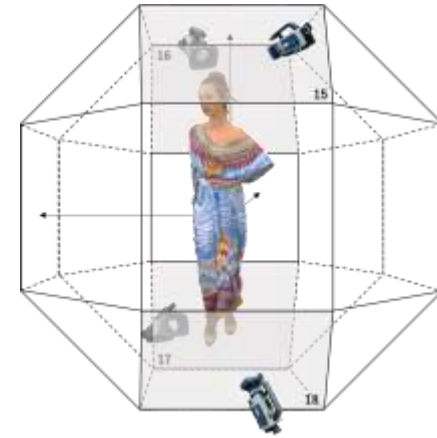
- In V-PCC, 6 orientations are defined: ( $\pm x$ ,  $\pm y$ ,  $\pm z$ )
- Additionally, the projection axis may be rotated by 45 degrees around each direction, into 12 new orientations



Rotation\* around Y-axis



Rotation\* around Z-axis

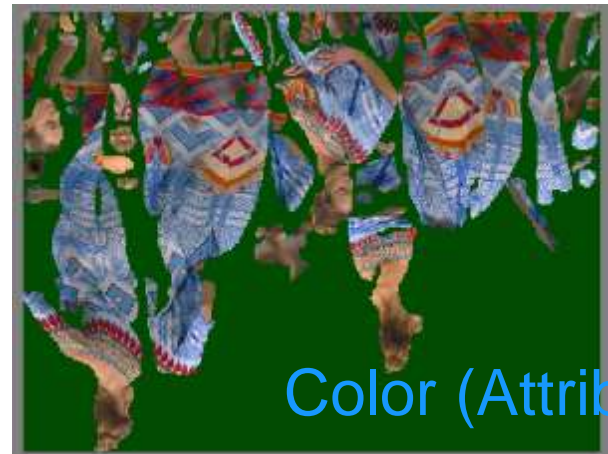
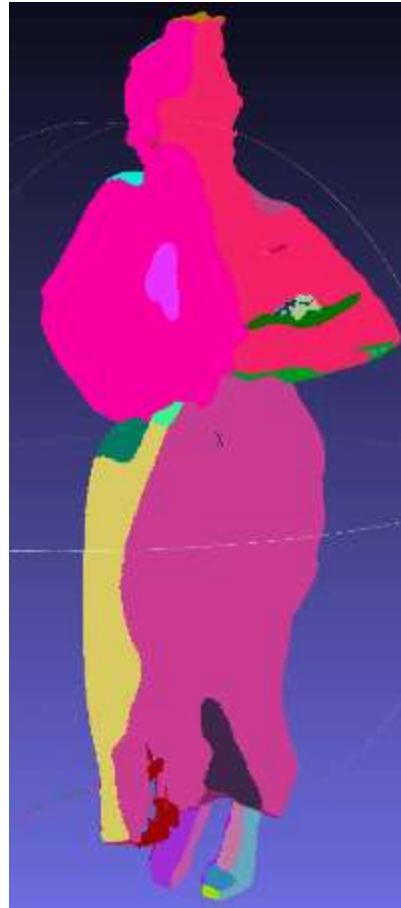


Rotation\* around X-axis

\*Rotations can be done lossless in integers, using shearing and scaling.

# Video-based Point Clouds Compression

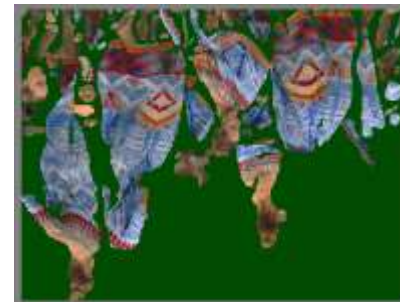
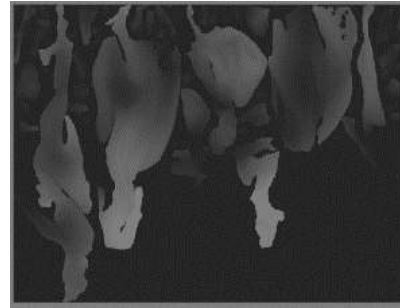
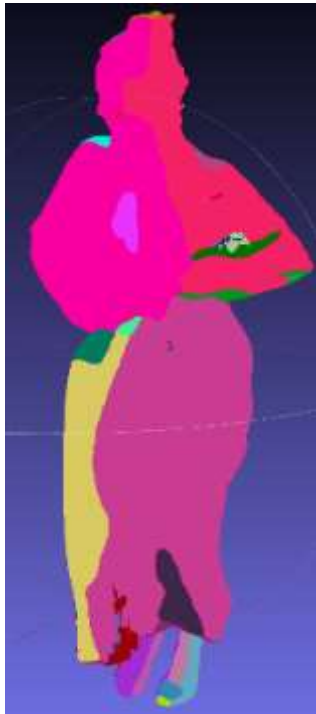
Encoding the 3D point clouds as a set of 2D patches



# Video-based Point Clouds Compression

Encoding the 3D point clouds as a set of 2D patches

- not all the pixels in the image are used for reconstruction

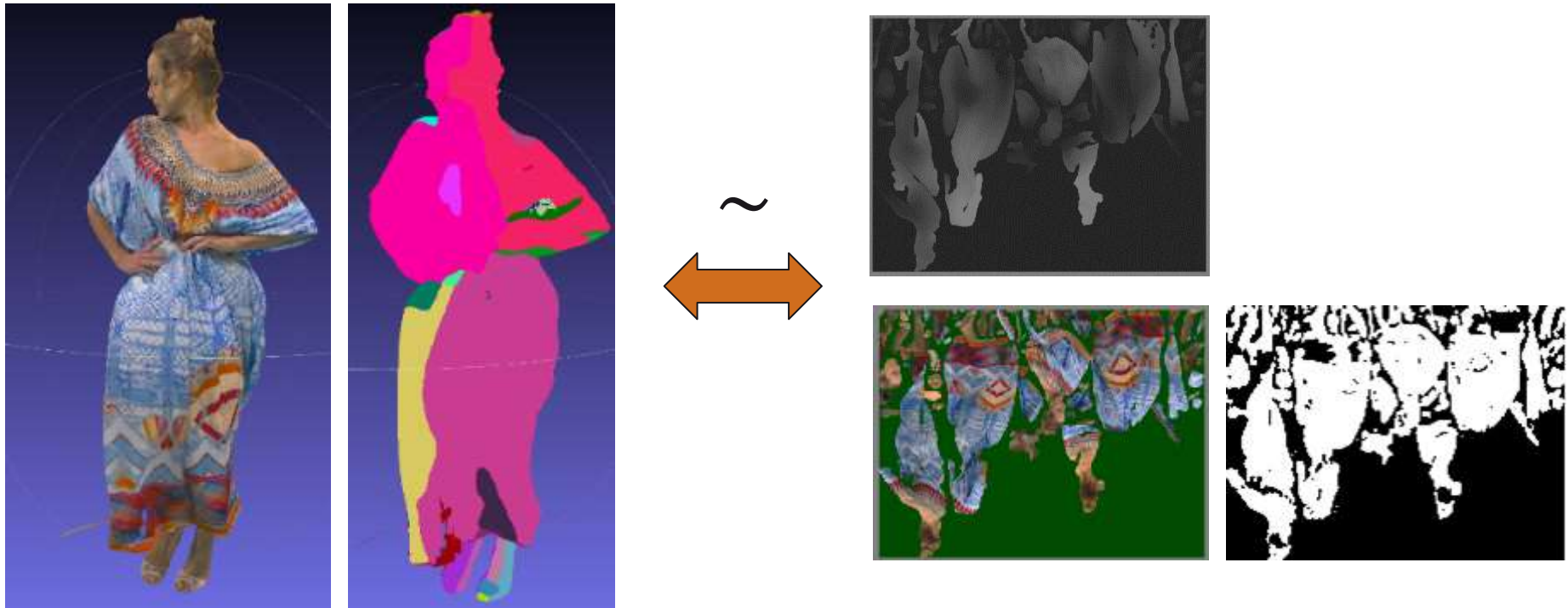




# Video-based Point Clouds Compression

Encoding the 3D point clouds as a set of 2D patches

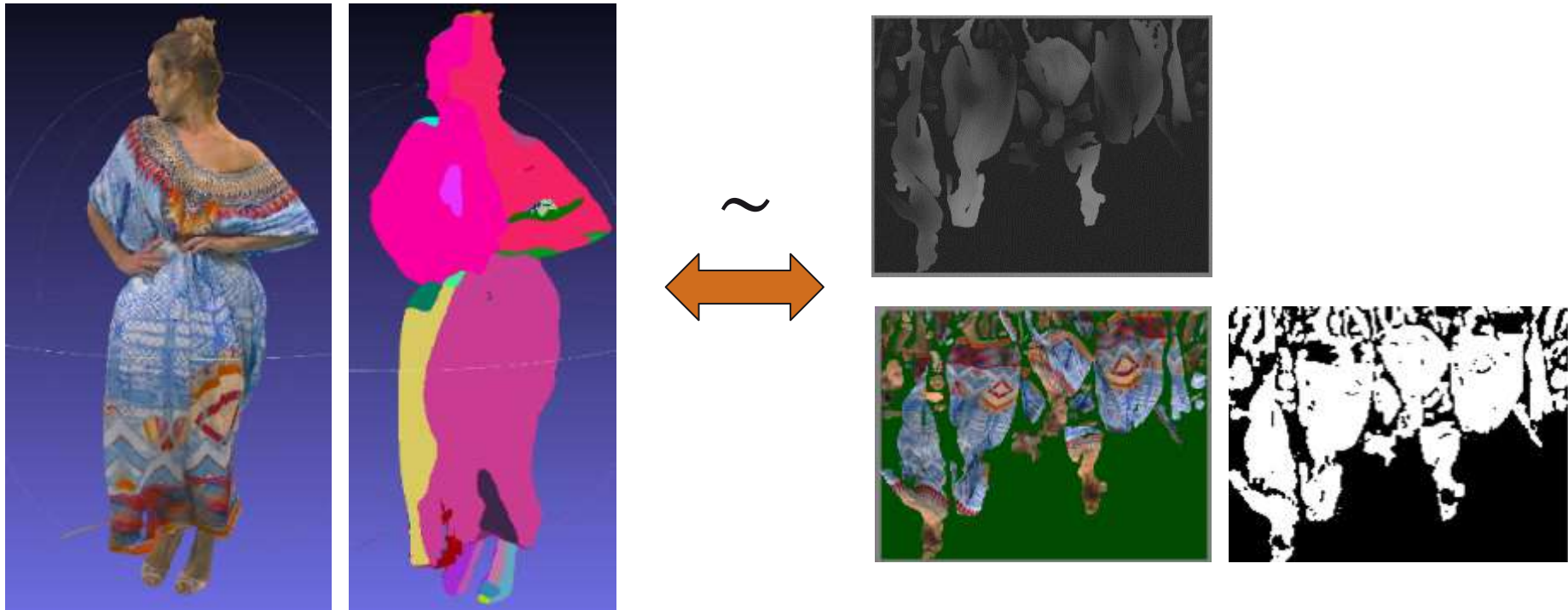
- not all the pixels in the image are used for reconstruction, an occupancy map indicates which ones should be used



# Video-based Point Clouds Compression

Encoding the 3D point clouds as a set of 2D patches

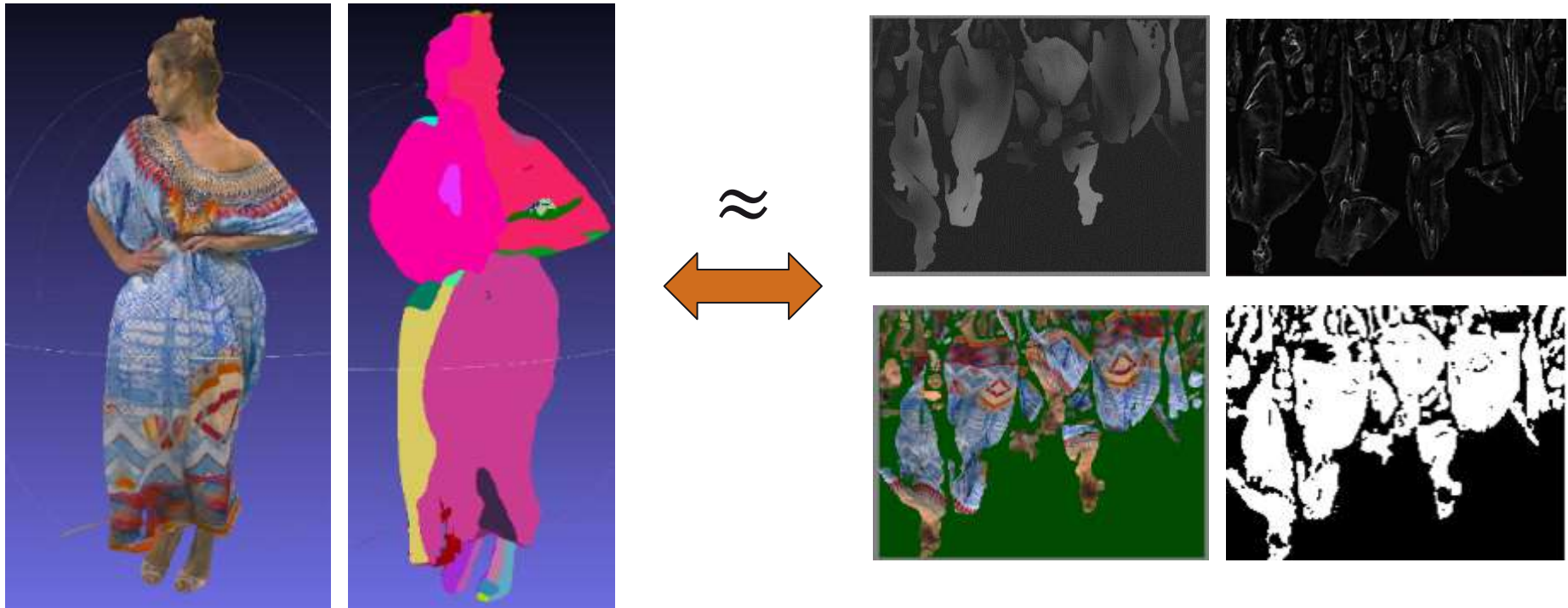
- Some points are still projected on already occupied pixels



# Video-based Point Clouds Compression

Encoding the 3D point clouds as a set of 2D patches

- Some points are still projected on already occupied pixels, an additional depth map is used

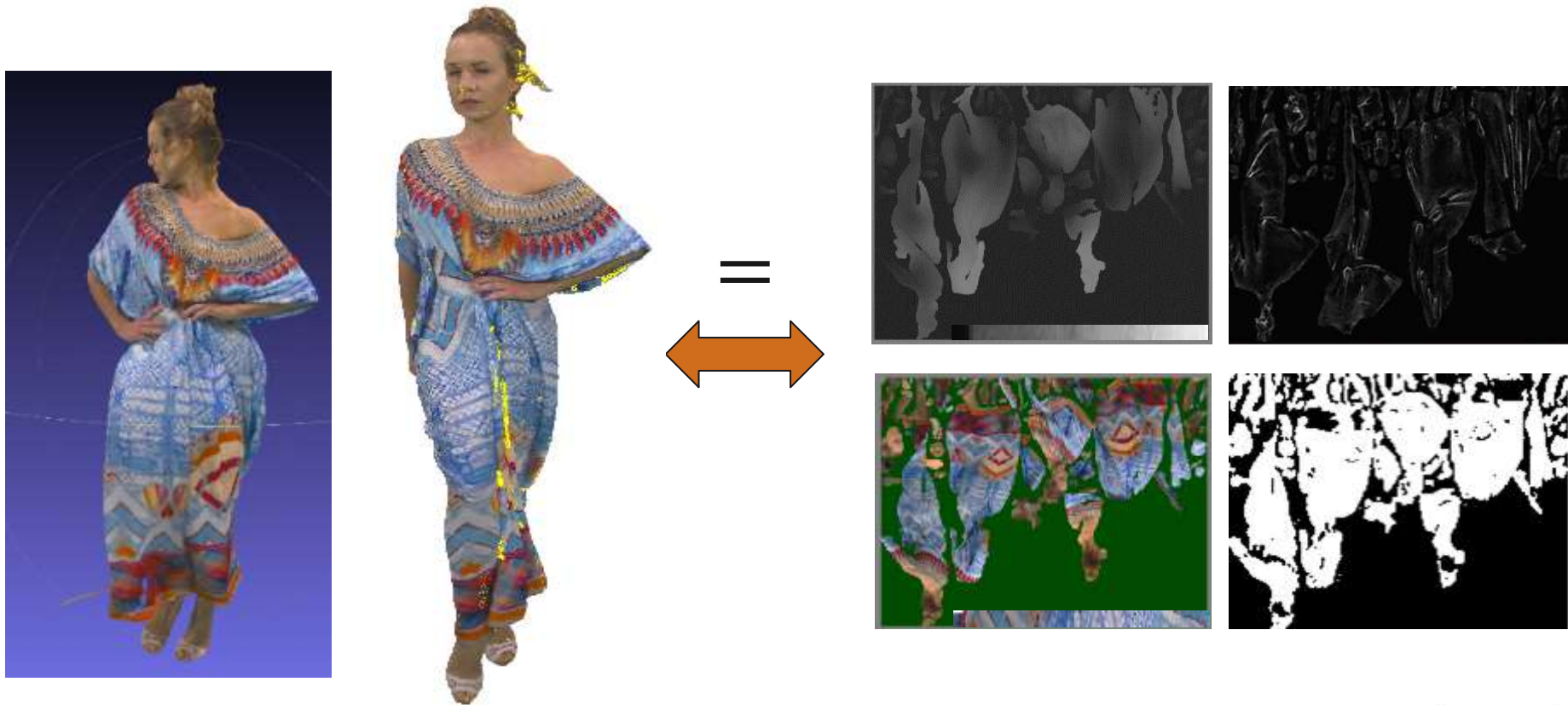




# Video-based Point Clouds Compression

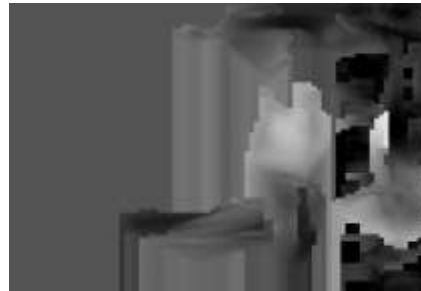
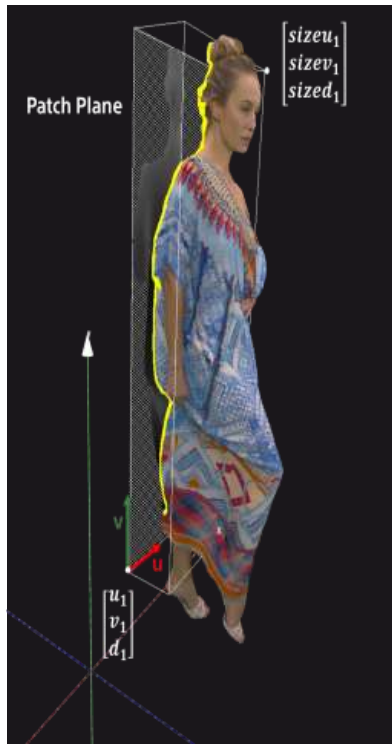
Encoding the 3D point clouds as a set of 2D patches

- For enforcing lossless, the missed points are encoded separately



# Video-based Point Clouds Compression

Encoding the 3D point clouds as a set of 2D videos: depth, color and occupancy maps



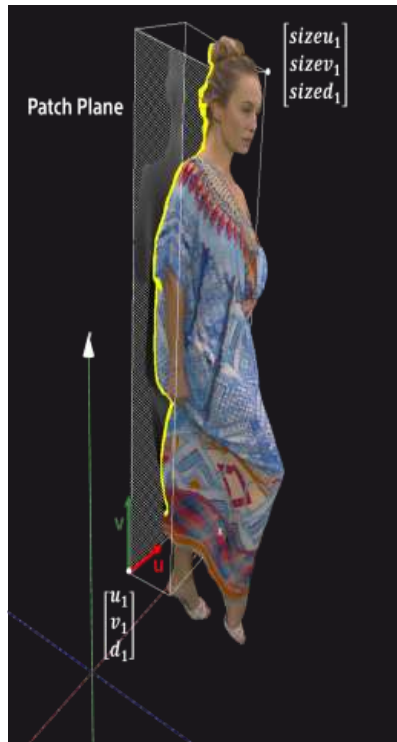
MPEG is very good in video coding!

Problem (almost) solved 😊

# Video-based Point Cloud Compression

Encoding 3D point clouds as a set of 2D videos: color, depth and occupancy map

100,000 points @ 30fps → 360 Mbps (uncompressed)  
→ **1 Mbps** (MPEG PCC 2019)



**7 Mbps**



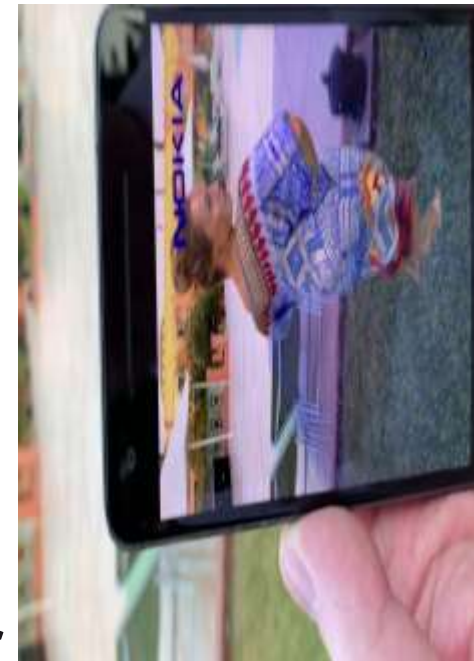
**4.4 Mbps**

# Video-based Point Cloud Compression

## ■ V-PCC implementations publicly available



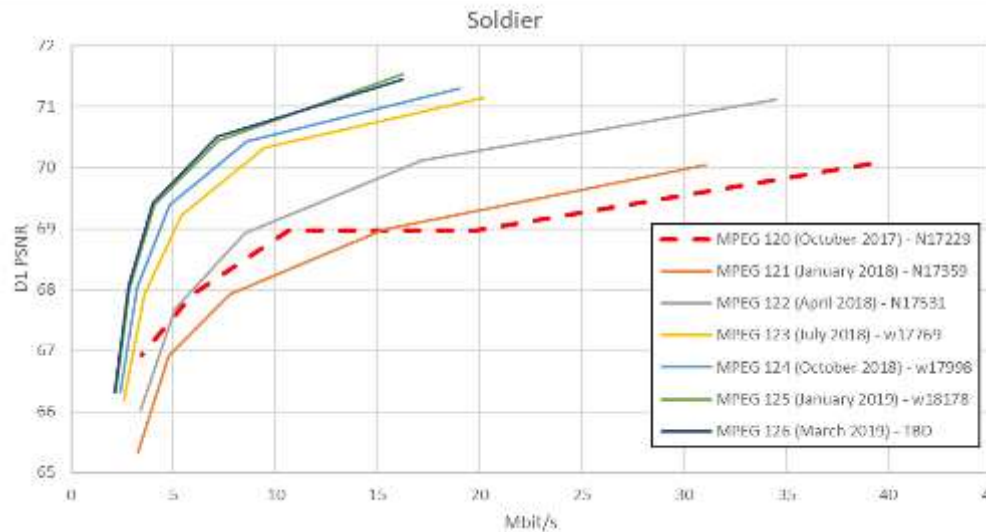
[www.mpeg-pcc.org](http://www.mpeg-pcc.org)



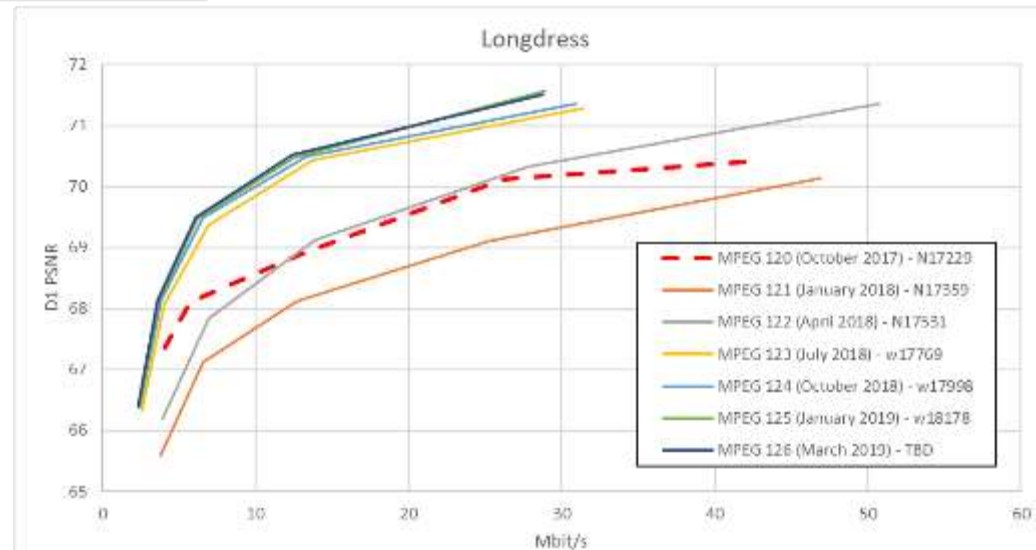
## ■ Integrated real-time decoder and renderer is also available for Android

<https://github.com/nokiatech/vpcc>

# About collaborations and environnement



- V-PCC progress was fast because video coding expertise was accessible
- No other Graphics community has this privilege

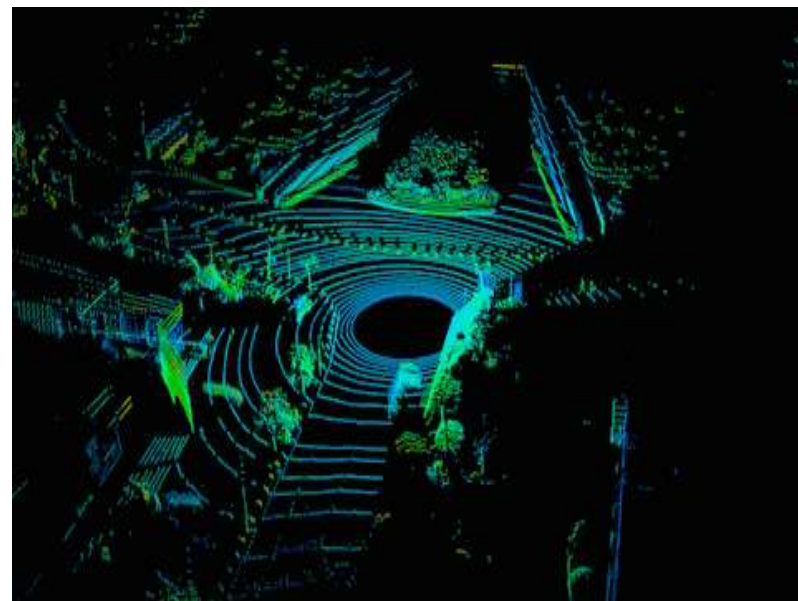
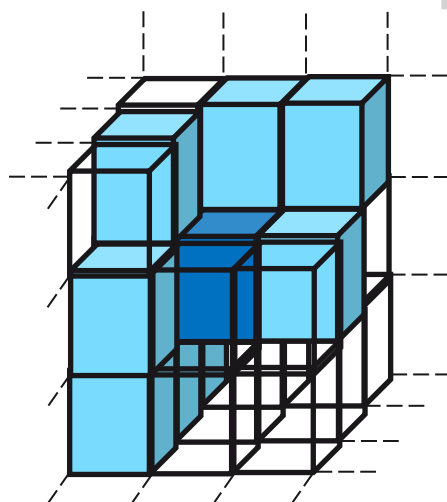
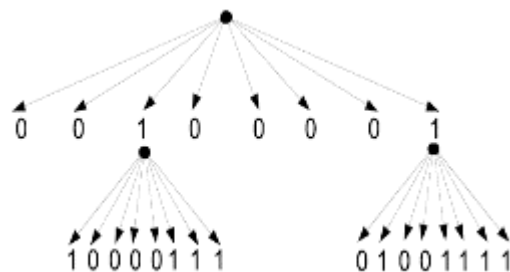
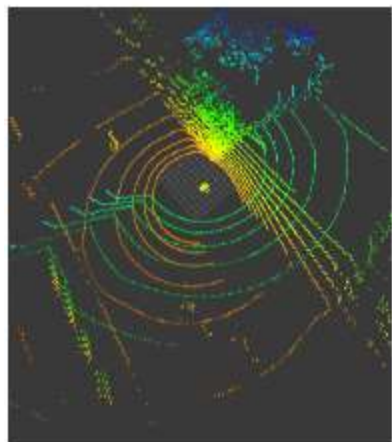




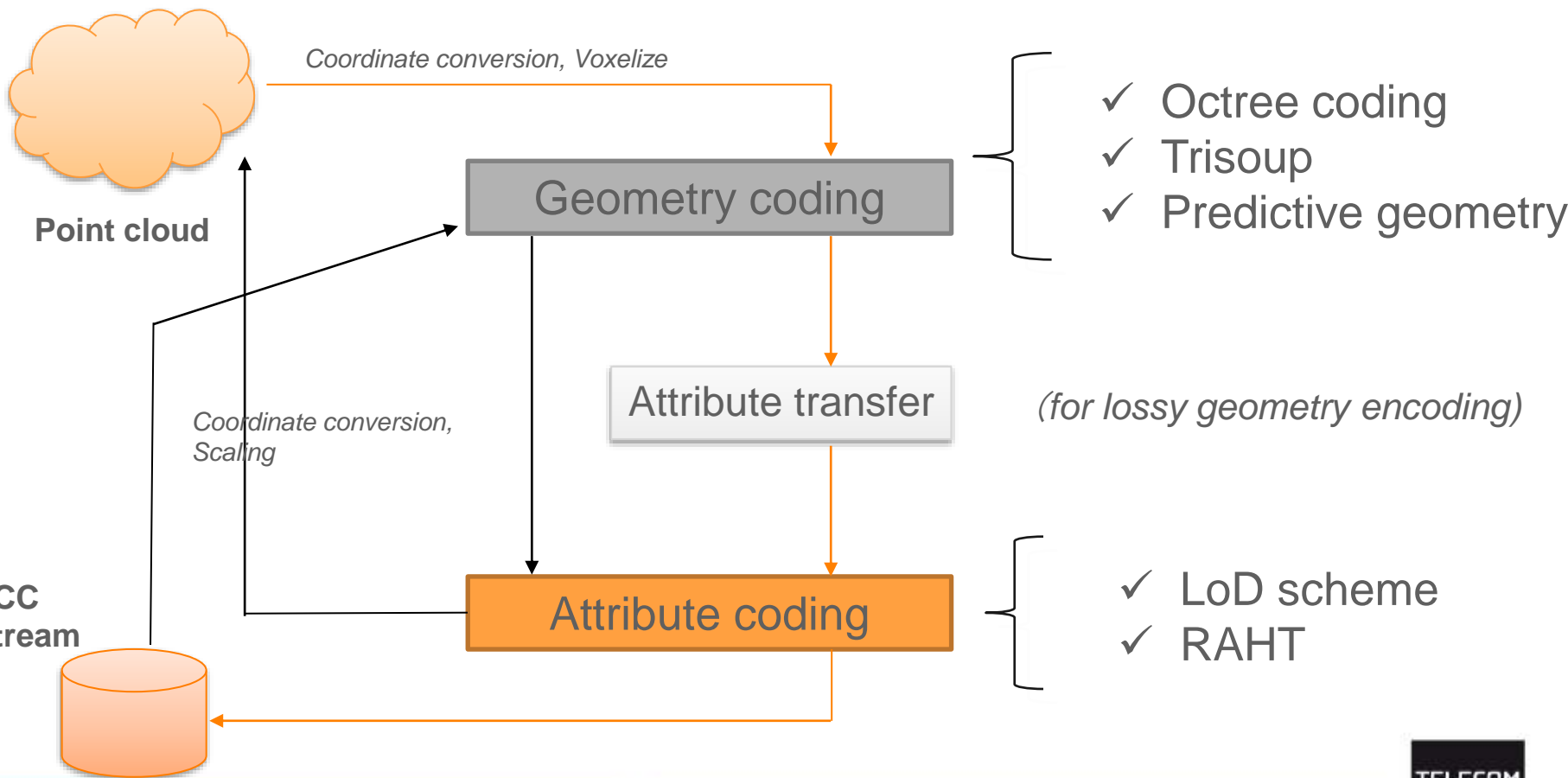
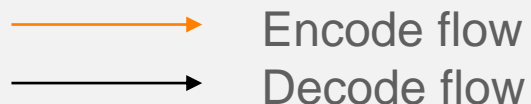
# Geometry-based Point Cloud Compression

## ■ Encoding 3D point clouds in their native format

100,000 points @ 10 fps  $\rightarrow$  112 Mbps (uncompressed)



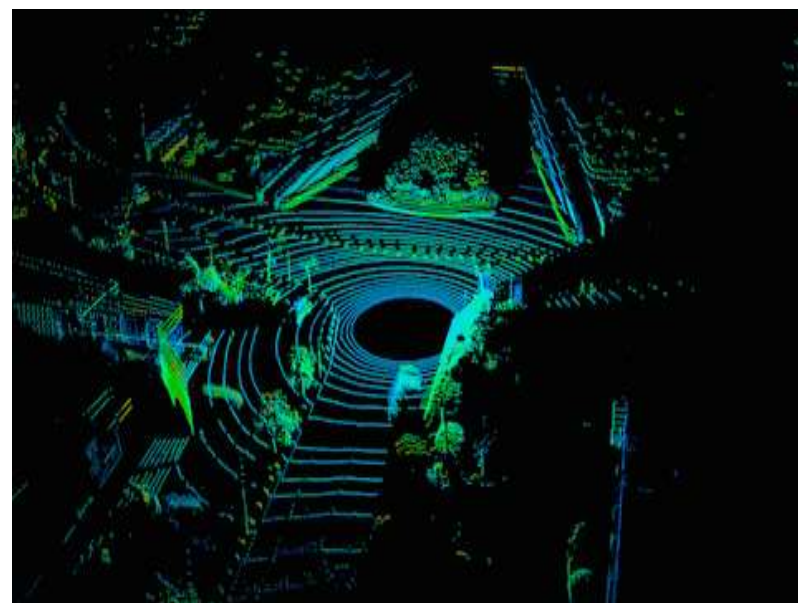
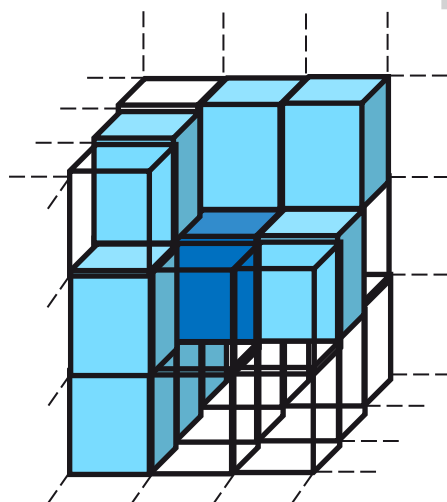
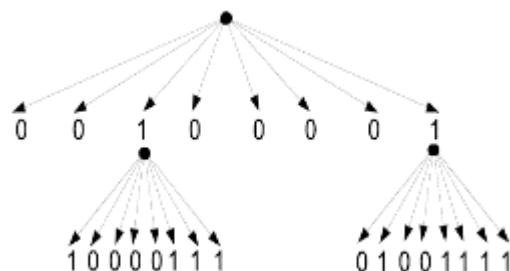
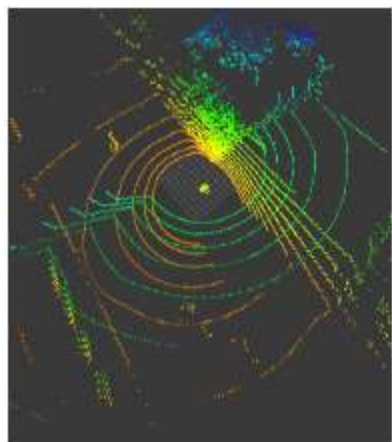
# Geometry-based Point Cloud Compression



# Geometry-based Point Cloud Compression

## ■ Encoding 3D point clouds in their native format

100,000 points @ 10 fps  $\rightarrow$  112 Mbps (uncompressed)



**24 Mbps (lossless) (80kpoints)**



# Geometry-based Point Cloud Compression

G-PCC handles various content categories, while offering state-of-the-art RD performance

Coding tool	Solid	Dense	Sparse	Scant	Lidar-Fused	Lidar-Frame	Improvements
IDCM			✓	✓	✓	✓	Complexity
Planar			✓	✓			RD
Neighbor Dependent Entropy Context	✓	✓					RD
Intra Occupancy Prediction	✓	✓					RD
Angular						✓	RD
Predictive					✓	✓	RD & Complexity
Trisoup	✓						RD
LoD	✓	✓	✓	✓			RD
RAHT			✓	✓	✓	✓	RD

What is next?

Revisiting the past!

**Visual  
capture**



**Point Cloud –**

a convergence between 2 worlds

**Mesh –**

A surface approximation of the point cloud



**Visual  
synthesis**

# Introducing D-Mesh (Dynamic Mesh)

- A set of 3D points
  - ordered,
  - connected to form polygons – **varying in time**
- A D-Mesh is defined by
  - $(X_t, Y_t, Z_t)_n$
  - $(v_1^t, v_2^t, v_3^t)_m$
  - $(R, G, B)$  – ~~still image~~ **video**
  - mapping from **video** texture to geometry
  - reflectance, transparency, ...
- Automatically captured

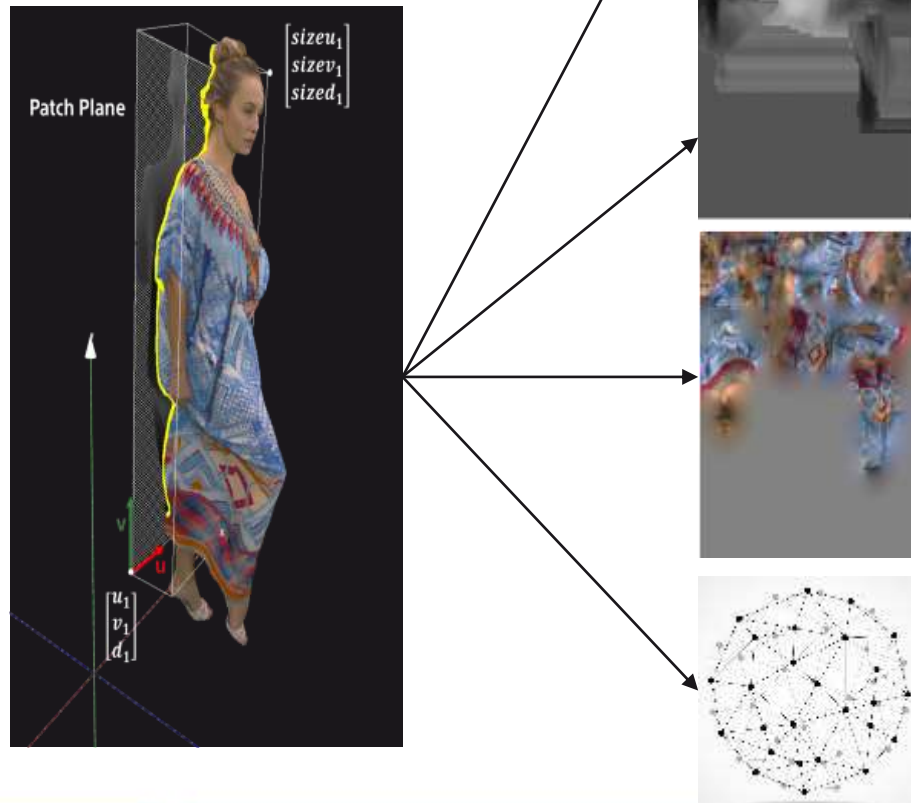


# V-PCC for the D-Mesh object

- V-PCC like approach for vertex positions and colors
- TFAN (or other) connectivity per frame

Ongoing  
exploration in  
MPEG

CfP to be issued in  
October 2021



# MPEG 3DGC and Khronos

- **Several discussions in the past but not yet a concrete modus operandi**
- **MPEG 3DGC is not developing standards for graphics formats but compression for (existent) graphics formats**
- **MPEG 3DGC can use the Vendor Extension mechanism of glTF to add compression for some data types (meshes, point clouds, animation)**



# Conclusions

- Point Cloud Compression enables interactive high quality 3D content by providing **manageable bitrates** and also reducing requirements in creation, transmission and (rendering) of 3D content
- V-PCC leverages the existing hardware and software infrastructure for **rapid deployment** of new immersive experiences. G-PCC has the potential to do better (supposing a similar expertise as in video coding is possible to build)
- PCC provides a **solid framework** for the convergence between natural and synthetic 3D graphics. Mesh extension of PCC improves rendering and (potentially) saves bandwidth

# Conclusion

- We are at the beginning of a new era when humanity will re-gain its third dimension in the digital space!



# Disclaimer

- Several pictures and videos used in this presentation are provided by
  - 8i, Owli, Sony, Intel RealSense, Microsoft Hololens, Institut Mines Telecom
  - The huge Internet image data bases

Thank you!