Automatic parameter selection

• Depth quality assessment
• Calculating the size of the atlas

Requirements:
• Maximum 4 decoder instances
• Maximum 8 MPix per one video

Solution:
• 2 atlases (2 × attributes + 2 × geometry)
• Atlas width: width of the widest input view
• Height of the atlas: the largest possible (meeting previous requirements)

\[ W_{atl} = W_{input} \quad H_{atl} = \frac{8 \text{Mpix}}{W_{atl}} \]
MIV – encoding of one group

- Source views
  - View parameters
    - Geometry component
    - Attribute components
    - Entity map (opt.)
- Automatic parameter selection
- Separate in entity layers (opt.)
- Prune pixels
- Aggregate pruning masks
- Cluster active pixels
- Split clusters
- Pack patches
- Patch attr. average value modification
- Color correction (opt.)
- Generate video data
- Quantize geometry
- Scale geometry
- Scale occupancy (opt.)
- View parameters list
- Parameter set

Color code:
- Prepare source material
- Pruning processes
- Atlas processes
- Video processes

MPEG Immersive video
Pixel pruning

• Most of elements visible in a view are also visible in other views
Pixel pruning

- Most of elements visible in a view are also visible in other views

- Pruning – removing inter-view redundancy and selecting the smallest amount of elements required to compress whole scene
**Pixel pruning: pruning graph**

1. Insert basic views into the pruning graph (as root nodes).
2. Project all pixels of all basic views to each additional view.
3. Create the pruning mask for each additional view.
4. Select the additional view with maximum number of preserved pixels (to prefer larger patches).
5. Insert selected additional view into the pruning graph (as a child node of all nodes already in graph) and stop if all the views are assigned to nodes in the pruning graph.
6. Project all preserved pixels of selected view to remaining additional views.
7. Update the pruning mask for each remaining additional view.
8. Go to 4.
Pixel pruning: pruning mask creation

Conditions for removing points:

- The difference of the **depth** of the transferred point and the point corresponding to it must be higher than a threshold (10%)

- The difference of the **luma** of the transferred point and all points in the corresponding 3 × 3 block must be higher than the threshold
  - Threshold value depends on the noise level in the sequence, the base value of 4% multiplied by the standard deviation of the Gaussian noise in the 1st frame

Final binary mask filtration (0: pruned pixel, 1: preserved pixel):

- Iterative erosion (3×3 window)
- Iterative dilation (3×3 window)
Pixel pruning

Geometry-based pruning

Texture- and geometry-based pruning
MIV – encoding of one group

Source views (selected for this group):
- View parameters (incl. basic/editional label)
- Geometry component
- Attribute components
- Entity map (opt.)

View parameters list
Parameter set

Automatic parameter selection
Separate in entity layers (opt.)
Prune pixels

Aggregate pruning masks
Cluster active pixels
Split clusters

Pack patches
Patch attr. average value modification
Color correction (opt.)

Atlas data

Generate video data
Quantize geometry
Scale geometry

Geometry video data (raw)
Attribute video data (raw)
Occupancy video data (raw)

Scale occupancy (opt.)

Color code:
- Prepare source material
- Pruning processes
- Atlas processes
- Video processes

MPEG Immersive video
Pruning mask aggregation

Temporal aggregation of pruning masks (in one GOP)
MIV – encoding of one group

Source views
(selected for this group)

View parameters
(incl. basic/additional label)

Geometry component

Attribute components

Entity map (opt.)

Automatic parameter selection

Separate in entity layers (opt.)

Prune pixels

Aggregate pruning masks

Cluster active pixels

Split clusters

Pack patches

Patch attr. average value modification

Color correction (opt.)

Generate video data

Quantize geometry

Scale geometry

Scale occupancy (opt.)

Atlas data

Geometry video data (raw)

Attribute video data (raw)

Occupancy video data (raw)

View parameters list

Parameter set

Color code

Prepare source material

Pruning processes

Atlas processes

Video processes
Pixel clustering
Pixel clustering
Pixel clustering

x, y, width, height
Pixel clustering: cluster merging

Fragment of (pruned) input view

Patch containing merged clusters a and b

Patch containing cluster c
MIV – encoding of one group
Cluster splitting

cluster(s) in src view → patch(es) in src view → patch(es) in atlas

L-cluster splitting
Cluster splitting

cluster in src view

C-cluster splitting
Cluster splitting
Cluster splitting

No splitting
Cluster splitting

No splitting
Cluster splitting

No splitting

Splitting enabled
MIV – encoding of one group

Source views
(selected for this group)

View parameters
(incl. basis/additional label)

Geometry component

Attribute components

Entity map (opt.)

Automatic parameter selection

Separate in entity layers (opt.)

Prune pixels

Aggregate pruning masks

Cluster active pixels

Split clusters

Pack patches

Patch attr. average value modification

Color correction (opt.)

Generate video data

Quantize geometry

Scale geometry

Scale occupancy (opt.)

View parameters list

Parameter set

Atlas data

Geometry video data (raw)

Attribute video data (raw)

Occupancy video data (raw)

Color code

- Prepare source material
- Pruning processes
- Atlas processes
- Video processes

MPEG Immersive video
Patch packing

8 possible positioning variants:
• 4 rotations (0°, 90°, 180°, 270°)
• Optional mirroring

Atlas frame
Patch packing

For each cluster:
   For each atlas:
      Push the cluster in “Used Space” (0° rotation first, 90° otherwise)
      If the push failed:
         Push the cluster into “Free Space” (0° rot. first, 90° otherwise)
         If the push failed:
            Split the cluster into 2 parts by its largest border
            For each resulting 2 parts:
               If smaller than MinPatchSize:
                  Discard the patch
               Else:
                  Put the part in the cluster priority list
Patch packing

For each cluster:
  For each atlas:
    Push the cluster in “Used Space” (0° rotation first, 90° otherwise)
    If the push failed:
      Push the cluster into “Free Space” (0° rot. first, 90° otherwise)
      If the push failed:
        Split the cluster into 2 parts by its largest border
        For each resulting 2 parts:
          If smaller than MinPatchSize:
            Discard the patch
          Else:
            Put the part in the cluster priority list

List of clusters sorted by area (1 square: 8 × 8 pixels):
Patch packing

For each cluster:
   For each atlas:
      Push the cluster in "Used Space" (0° rotation first, 90° otherwise)
      If the push failed:
         Push the cluster into "Free Space" (0° rot. first, 90° otherwise)
         If the push failed:
            Split the cluster into 2 parts by its largest border
            For each resulting 2 parts:
               If smaller than MinPatchSize:
                  Discard the patch
               Else:
                  Put the part in the cluster priority list

List of clusters sorted by area (1 square: 8 × 8 pixels):
Patch packing

For each cluster:
  For each atlas:
    Push the cluster in “Used Space” (0° rotation first, 90° otherwise)
    If the push failed:
      Push the cluster into “Free Space” (0° rot. first, 90° otherwise)
      If the push failed:
        Split the cluster into 2 parts by its largest border
        For each resulting 2 parts:
          If smaller than MinPatchSize:
            Discard the patch
          Else:
            Put the part in the cluster priority list

List of clusters sorted by area (1 square: 8 × 8 pixels):
Patch packing

For each cluster:
  For each atlas:
    Push the cluster in “Used Space” (0° rotation first, 90° otherwise)
    If the push failed:
      Push the cluster into “Free Space” (0° rot. first, 90° otherwise)
      If the push failed:
        Split the cluster into 2 parts by its largest border
        For each resulting 2 parts:
          If smaller than MinPatchSize:
            Discard the patch
          Else:
            Put the part in the cluster priority list

List of clusters sorted by area (1 square: 8 × 8 pixels):
Patch packing

For each cluster:
   For each atlas:
      Push the cluster in “Used Space” (0° rotation first, 90° otherwise)
      If the push failed:
         Push the cluster into “Free Space” (0° rot. first, 90° otherwise)
         If the push failed:
            Split the cluster into 2 parts by its largest border
            For each resulting 2 parts:
               If smaller than MinPatchSize:
                  Discard the patch
               Else:
                  Put the part in the cluster priority list

List of clusters sorted by area (1 square: 8 × 8 pixels):
Patch packing

For each cluster:

For each atlas:

Push the cluster in "**Used Space**" (0° rotation first, 90° otherwise)

If the push failed:

Push the cluster into "Free Space" (0° rot. first, 90° otherwise)

If the push failed:

Split the cluster into 2 parts by its largest border

For each resulting 2 parts:

If smaller than MinPatchSize:

Discard the patch

Else:

Put the part in the cluster priority list

List of clusters sorted by area (1 square: 8 × 8 pixels):
Patch packing

For each cluster:
  For each atlas:
    Push the cluster in “Used Space” (0° rotation first, 90° otherwise)
    If the push failed:
      Push the cluster into “Free Space” (0° rot. first, 90° otherwise)
      If the push failed:
        Split the cluster into 2 parts by its largest border
        For each resulting 2 parts:
          If smaller than MinPatchSize:
            Discard the patch
          Else:
            Put the part in the cluster priority list

List of clusters sorted by area (1 square: 8 × 8 pixels):
Patch packing

For each cluster:
   For each atlas:
      Push the cluster in “Used Space” (0° rotation first, 90° otherwise)
      If the push failed:
         Push the cluster into “Free Space” (0° rot. first, 90° otherwise)
         If the push failed:
            Split the cluster into 2 parts by its largest border
            For each resulting 2 parts:
               If smaller than MinPatchSize:
                  Discard the patch
               Else:
                  Put the part in the cluster priority list

List of clusters sorted by area (1 square: 8 x 8 pixels):
Patch packing

Source view  After pruning  Packed into atlas

Basic view

Additional view

Additional view
MIV – encoding of one group
Patch attribute value modification

![Graph showing distribution with average 230 over a range of values from 0 to 1023.]

MPEG Immersive video
Patch attribute value modification

count

avg: 230

value

0 512 1023

count

avg: 512

value

0 512 1023

MPEG Immersive video
Patch attribute value modification

count

avg: 260

value

avg: 480

avg: 512

32

count

value

MPEG Immersive video
Patch attribute value modification

Without mean patch value modification

Mean patch value modification enabled

MPEG Immersive video
Patch attribute value modification

Without mean patch value modification

Mean patch value modification enabled
MIV – encoding of one group

- Source views
  - (selected for this group)
  - View parameters
    - (incl. basic/additional label)
  - Geometry component
  - Attribute components
  - Entity map (opt.)

- View parameters list
  - Parameter set

- Automatic parameter selection
- Separate in entity layers (opt.)
- Prune pixels

- Aggregate pruning masks
- Cluster active pixels
- Split clusters

- Pack patches
- Patch attr. average value modification
- Color correction (opt.)

- Generate video data
- Quantize geometry
- Scale geometry

- Geometry video data (raw)
- Attribute video data (raw)
- Occupancy video data (raw)

- Scale occupancy (opt.)

Color code:

- Prepare source material
- Pruning processes
- Atlas processes
- Video processes

MPEG Immersive video
**Color correction**

- Alignment of color characteristics of patches from different source views
- Offset averaged independently for each patch
- The ability to compensate for local changes in lighting

*oc* – offset calculation
Color correction

Without color correction

Color correction enabled
MIV – encoding of one group

Source views
(selected for this group)

- View parameters
  (incl. basic/additional label)
- Geometry component
- Attribute components
- Entity map (opt.)

View parameters list
Parameter set

Automatic parameter selection

Separate in entity layers (opt.)

Prune pixels

Aggregate pruning masks

Cluster active pixels

Split clusters

Pack patches

Patch attr. average value modification

Color correction (opt.)

Atlas data

Generate video data

Quantize geometry

Scale geometry

Geometry video data (raw)

Attribute video data (raw)

Occupancy video data (raw)

Scale occupancy (opt.)

Color code
- Prepare source material
- Pruning processes
- Atlas processes
- Video processes

MPEG Immersive video
Video data generation: temporal redundancy removal

• Initially area within whole bounding box of a patch is packed into the atlas

• Temporal redundancy removal uses data from pixel pruning to send only data required in the current frame

• Occupancy of patch sent in its geometry

Without temporal redundancy removal

Temporal redundancy removal enabled
MIV – encoding of one group

Source views
(selected for this group)

View parameters
(incl. basis/optional/label)

Geometry component

Attribute components

Entity map (optional)

Automatic parameter selection

Separate in entity layers (opt.)

Prune pixels

Aggregate pruning masks

Cluster active pixels

Split clusters

Pack patches

Patch attr. average value modification

Color correction (optional)

Generate video data

Quantize geometry

Scale geometry

Scale occupancy (optional)

View parameters list

Parameter set

Color code

Prepare source material

Pruning processes

Atlas processes

Video processes

MPEG Immersive video
Geometry quantization

Filling the entire available dynamic range independently for each cluster:

\[
\begin{align*}
    d_{\text{near}}^{-1} &= \max \limits_{\text{all pixels of cluster in GOP}} d^{-1}(\text{pixel}) \\
    d_{\text{far}}^{-1} &= \min \limits_{\text{all pixels of cluster in GOP}} d^{-1}(\text{pixel})
\end{align*}
\]
Geometry quantization

Before scaling

After scaling
MIV – encoding of one group

Source views
- View parameters
- Geometry component
- Attribute components
- Entity map (opt.)

View parameters list
- Parameter set

Automatic parameter selection
- Separate in entity layers (opt.)
- Prune pixels

Aggregate pruning masks
- Cluster active pixels
- Split clusters

Pack patches
- Patch attr. average value modification
- Color correction (opt.)

Generate video data
- Quantize geometry
- Scale geometry

Scale occupancy (opt.)

Color code
- Prepare source material
- Pruning processes
- Atlas processes
- Video processes

MPEG Immersive video
Geometry downscaling

Attribute atlas: 1920 × 4640
Geometry atlas: 1920 × 4640
Geometry downscaling

Attribute atlas: $1920 \times 4640$
Geometry atlas: $960 \times 2320$

- Less points
- Smaller bitstream
- Smaller QP for texture with the same bitrate
Video sub-bitstream packing

Format Bitstream
(V3C sample stream with M1V extensions)

VVC encoded Attribute
Video Data (1 subpicture per picture)

VVC encoded Geometry
Video Data (1 subpicture per picture)

Packing using SubpicMergeApp

VVC bitstream 1

VVC bitstream 2

Multiplex
V3C multiplexing to V3C_PVD units

Bitstream (one file)

packing parameters (json)

Color code
- Bitstream formatting
- Video encoding
- Packing parameters
Video sub-bitstream packing
MIV – bitstream generation and multiplexing

Source views
- View parameters
- Geometry component
- Attribute components
- Entity map (opt.)

Encode groups (Figure 2)

Split source in groups

Label views (basic, additional)

Parameter set
- View parameters list
- Atlas data

Geometry video data (raw)

Attribute video data (raw)

Occupancy video data (raw)

Encode video sub bitstreams (HM / VTM)

Format Bitstream (V3C sample stream with MIV extensions)

SEI messages

Multiplex

Bitstream (one file)
MIV tutorial

• MIV website: https://mpeg-miv.org
• Public software:
  – Test model: https://gitlab.com/mpeg-i-visual/tmiv
  – IV-PSNR: https://gitlab.com/mpeg-i-visual/ivpsnr
  – IVDE: https://gitlab.com/mpeg-i-visual/ivde
  – RVS: https://gitlab.com/mpeg-i-visual/rvs
• Demo videos:
  – Freeport player: https://youtu.be/UeT_Xm1jBGs
  – Multi-plane images: https://youtu.be/DPMMzj2l6Yw
  – Real-time RVS: https://lisaserver.ulb.ac.be/rvs/
• Test material (14+ sequences) available on request
Q&A

End of tutorial