Automatic parameter selection

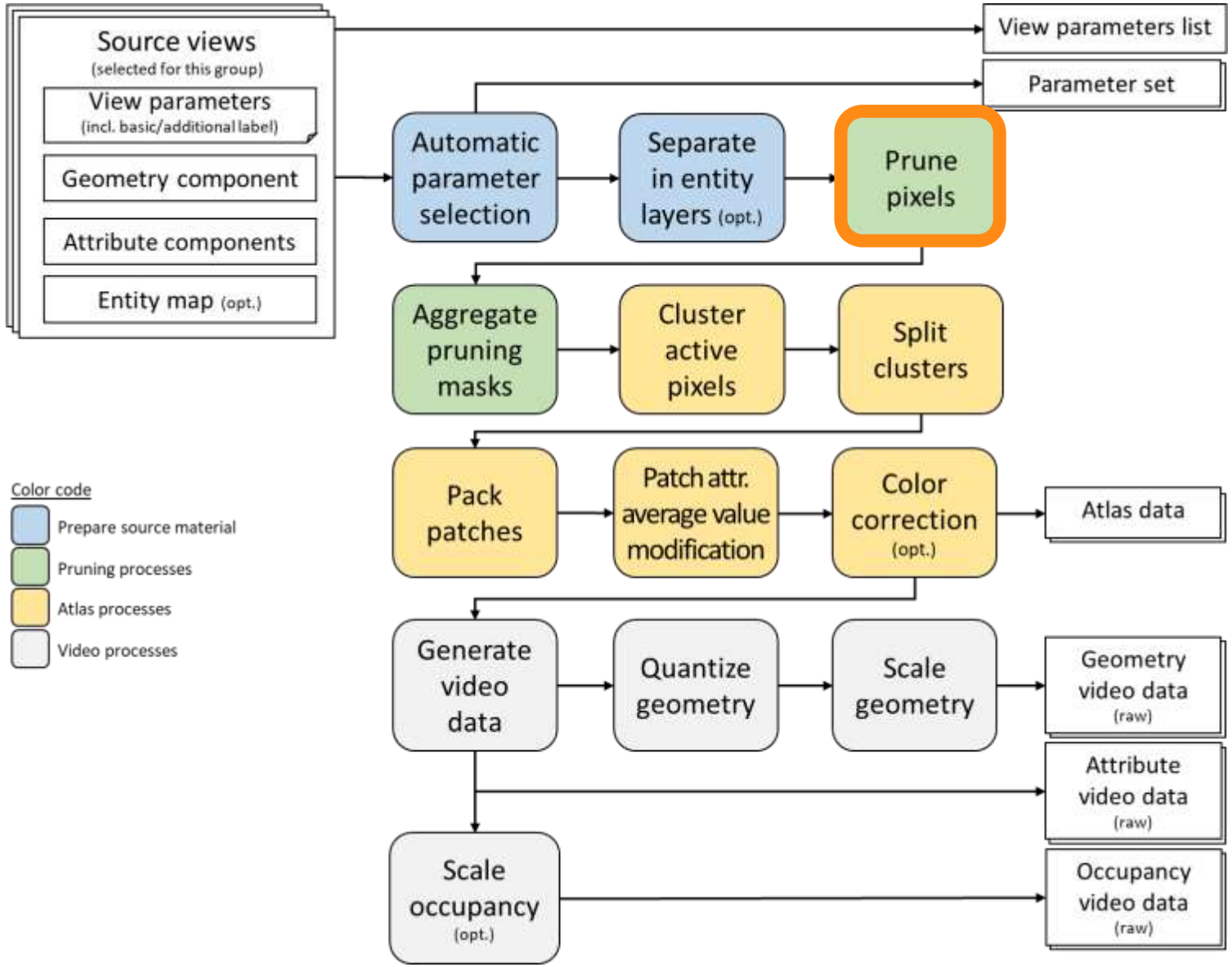
- Depth quality assessment
- Calculating the size of the atlas

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)

$$W_{atl} = W_{input} \qquad H_{atl} = \frac{8Mpix}{W_{atl}}$$

MIV – encoding of one group



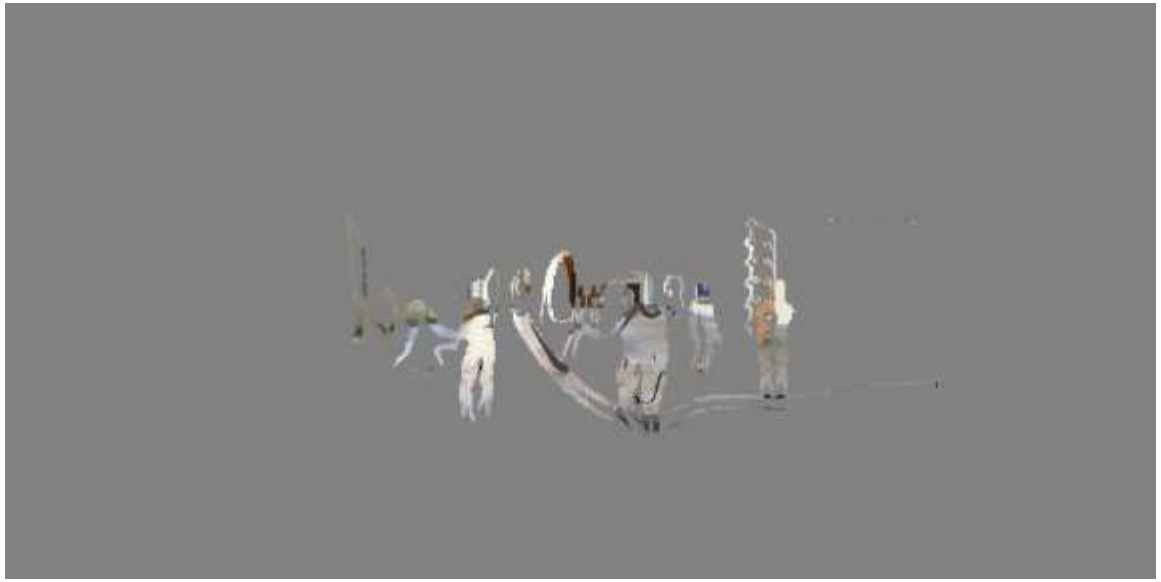
Pixel pruning

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

Pixel pruning

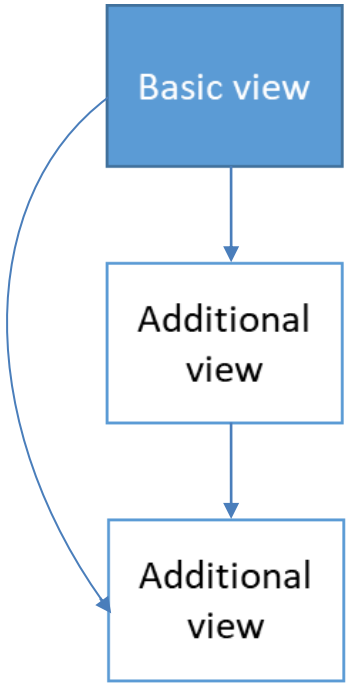
- Most of elements visible in a view are also visible in other views

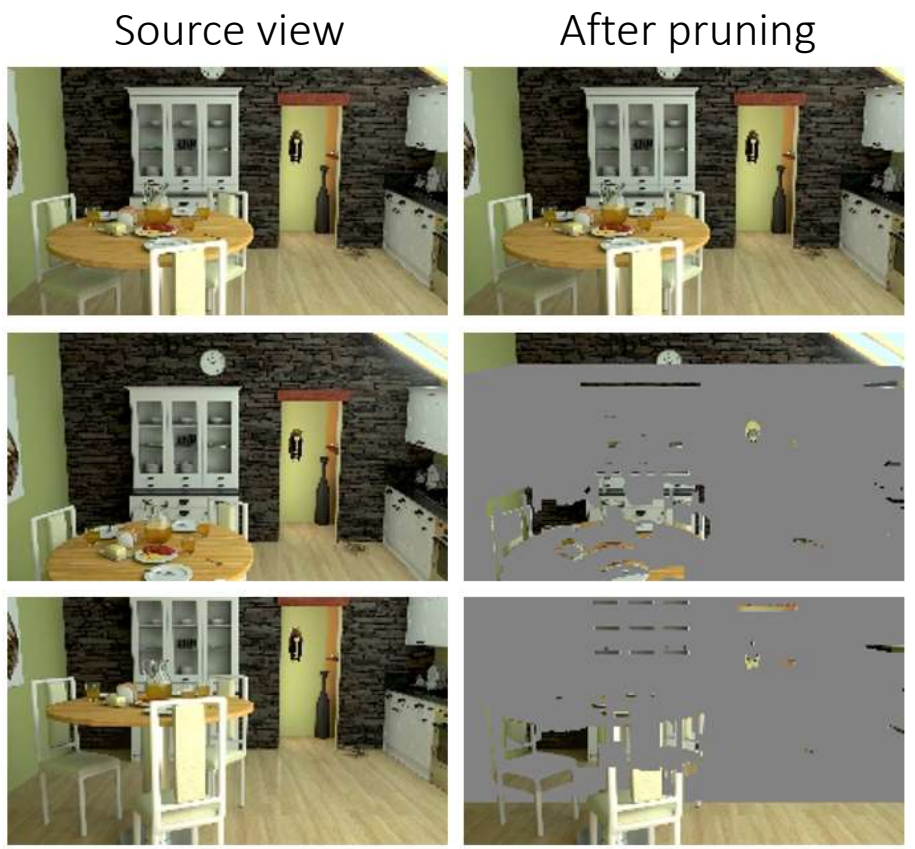
- Pruning – removing inter-view redundancy and selecting the smallest amount of elements required to compress whole scene



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



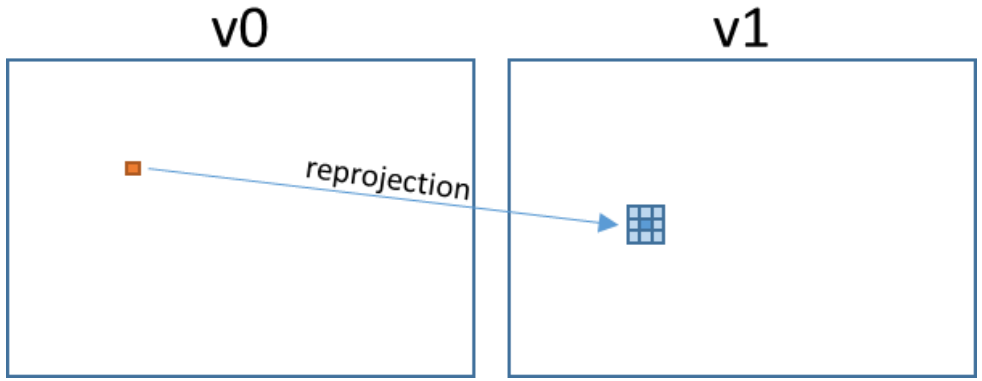


Pixel pruning: pruning mask creation

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

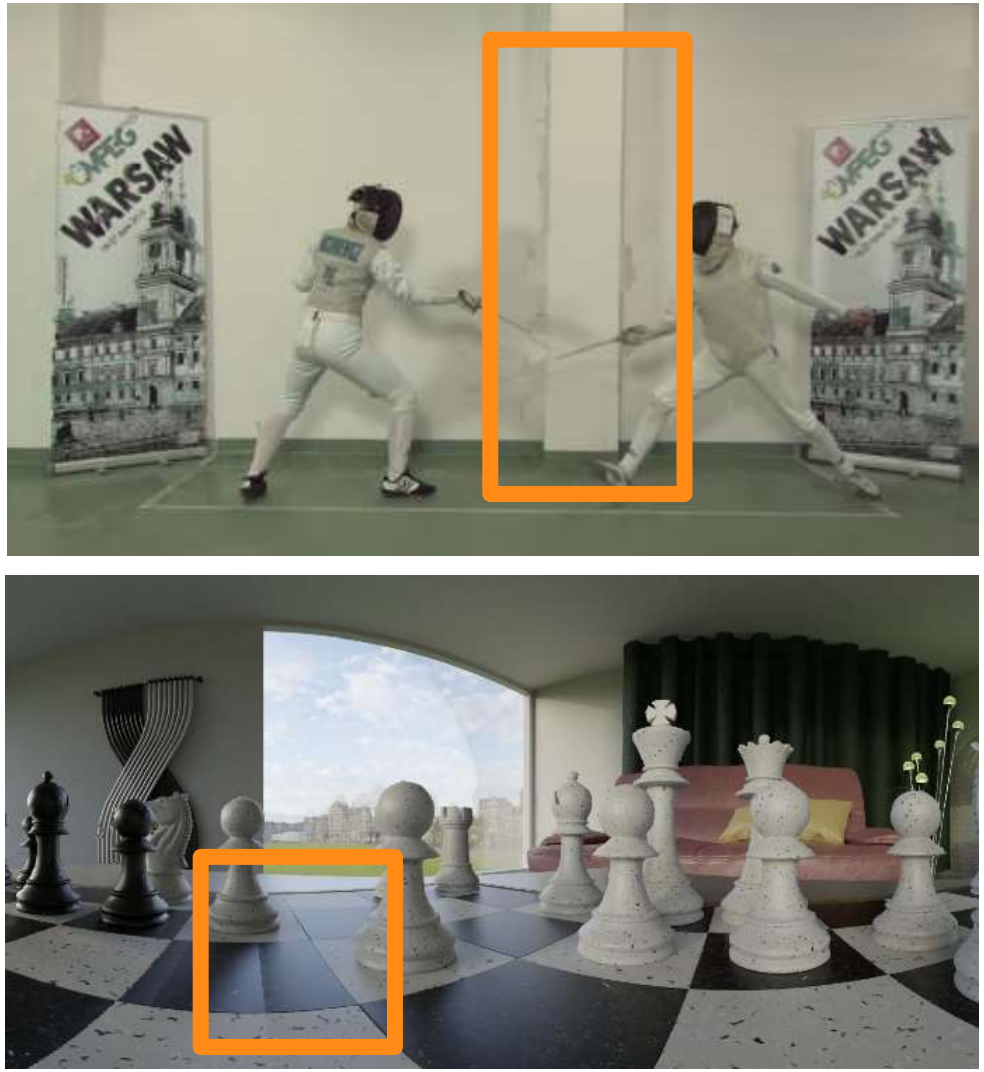


Final binary mask filtration (0: pruned pixel, 1: preserved pixel):

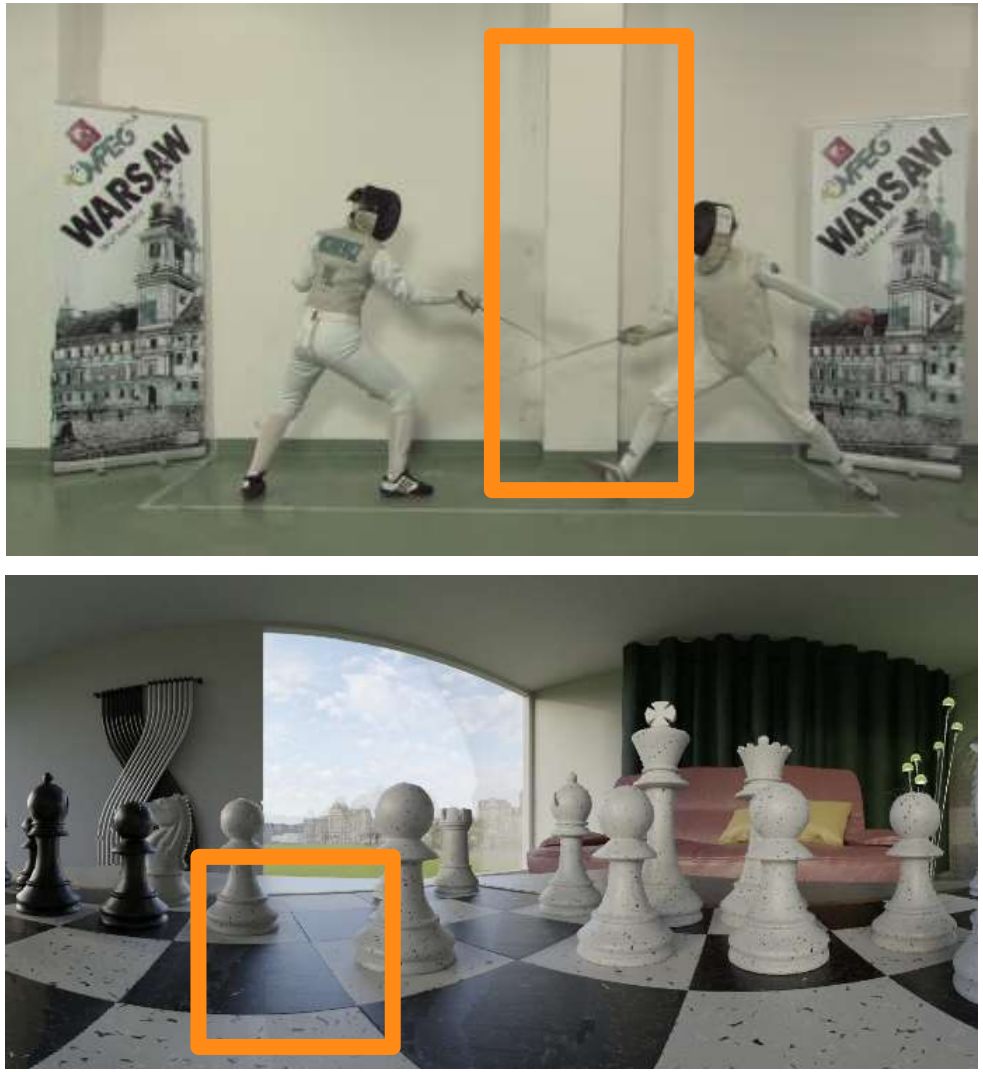
- Iterative erosion (3×3 window)
- Iterative dilation (3×3 window)

Pixel pruning

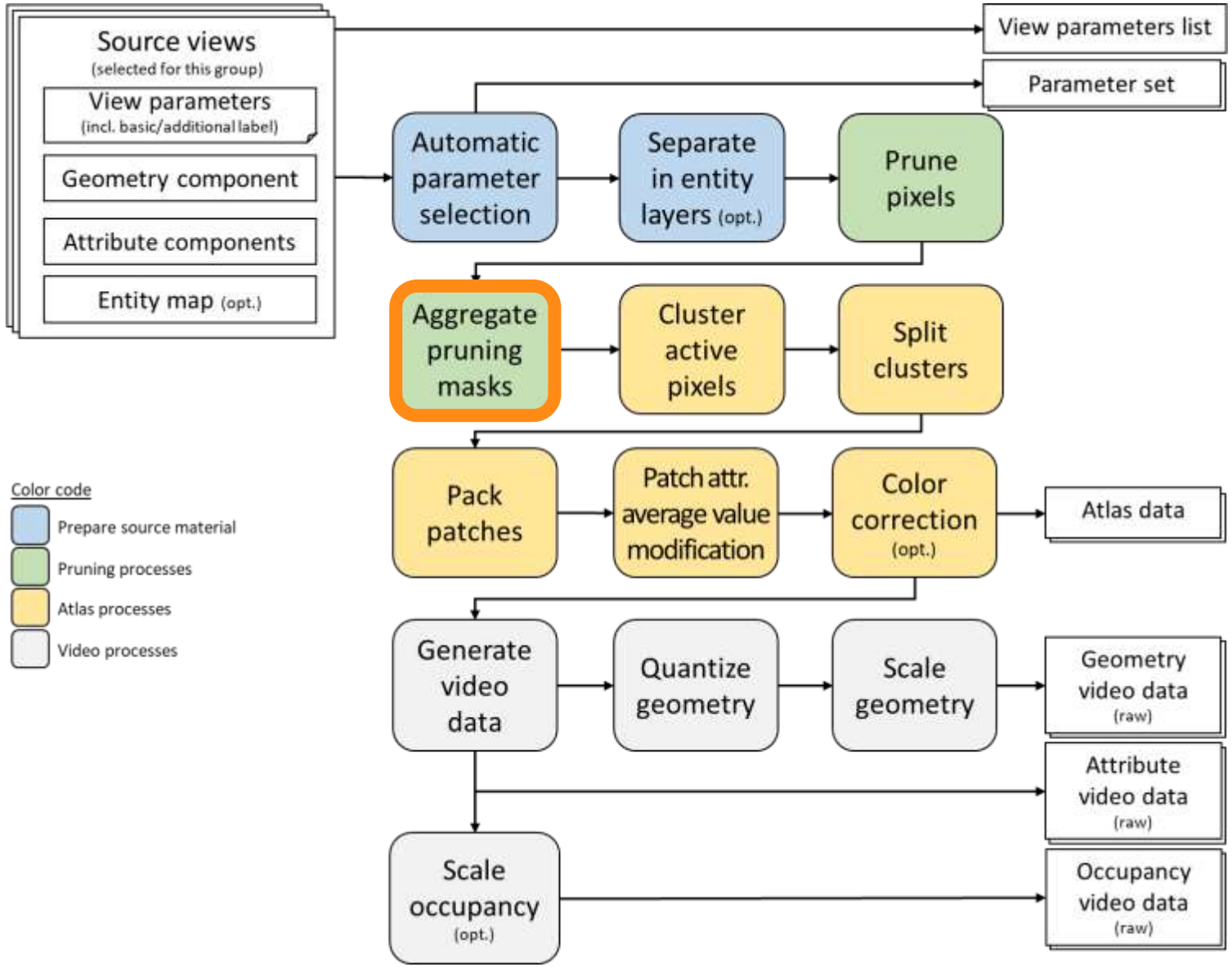
Geometry-based pruning



Texture- and geometry-based pruning

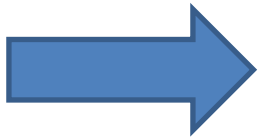


MIV – encoding of one group

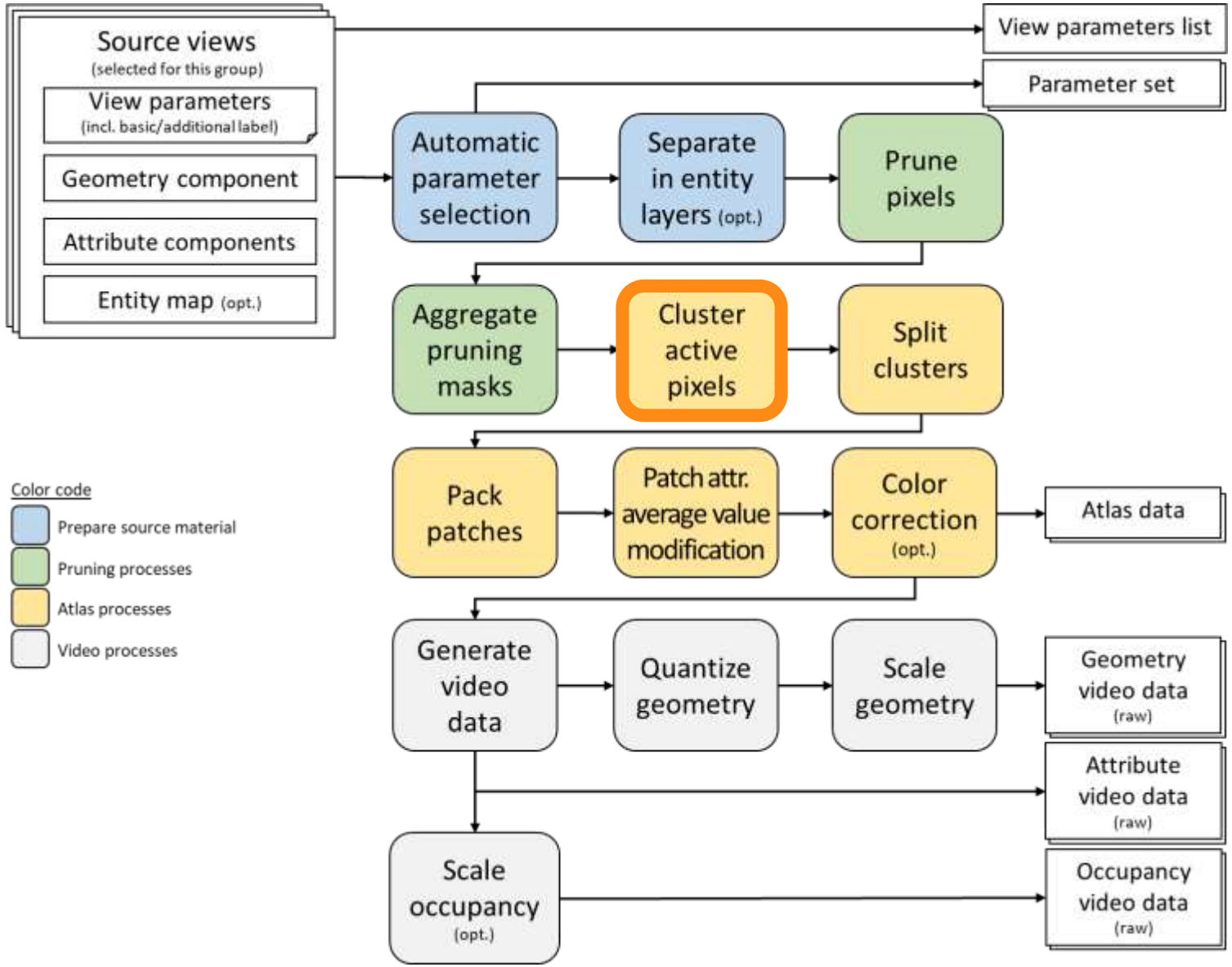


Pruning mask aggregation

Temporal aggregation of pruning masks (in one GOP)



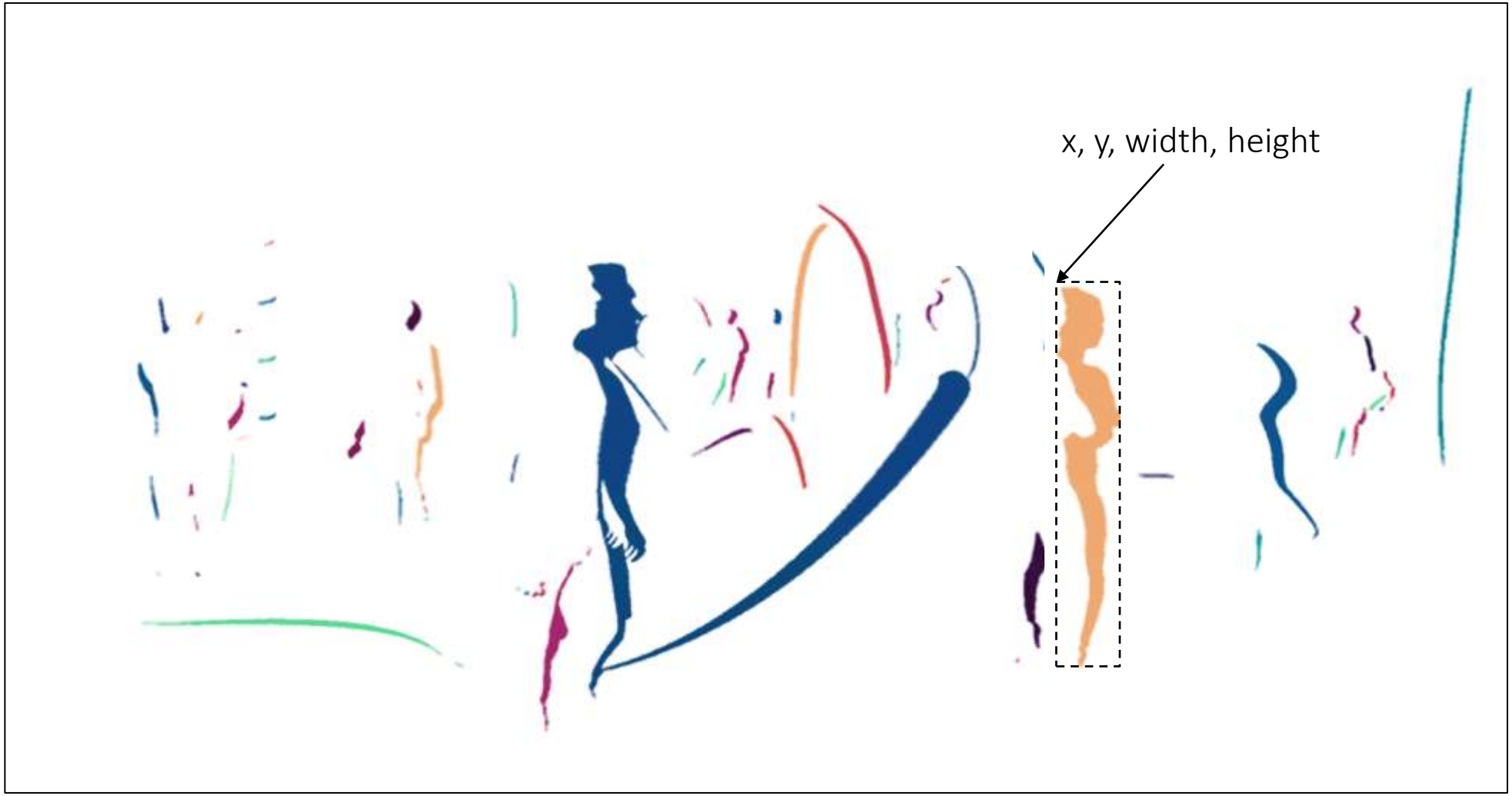
MIV – encoding of one group



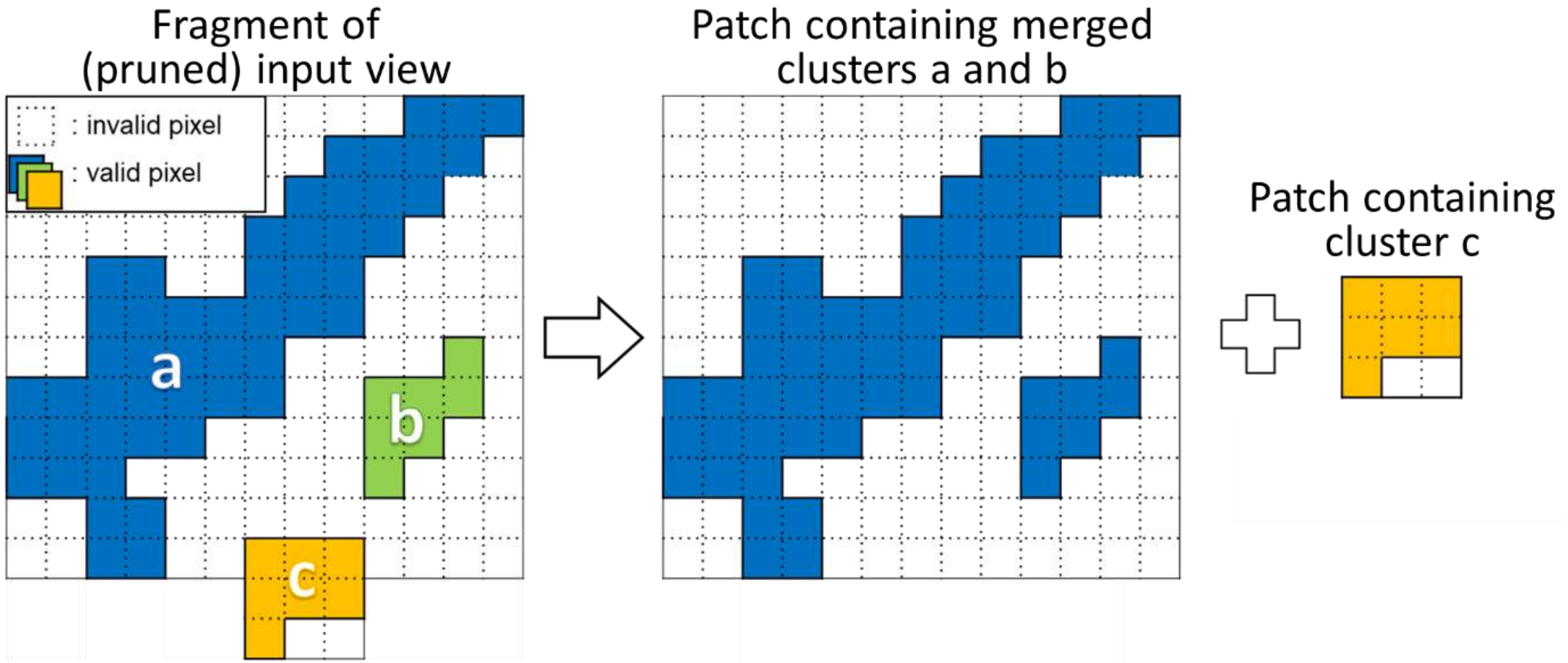
Pixel clustering

Pixel clustering

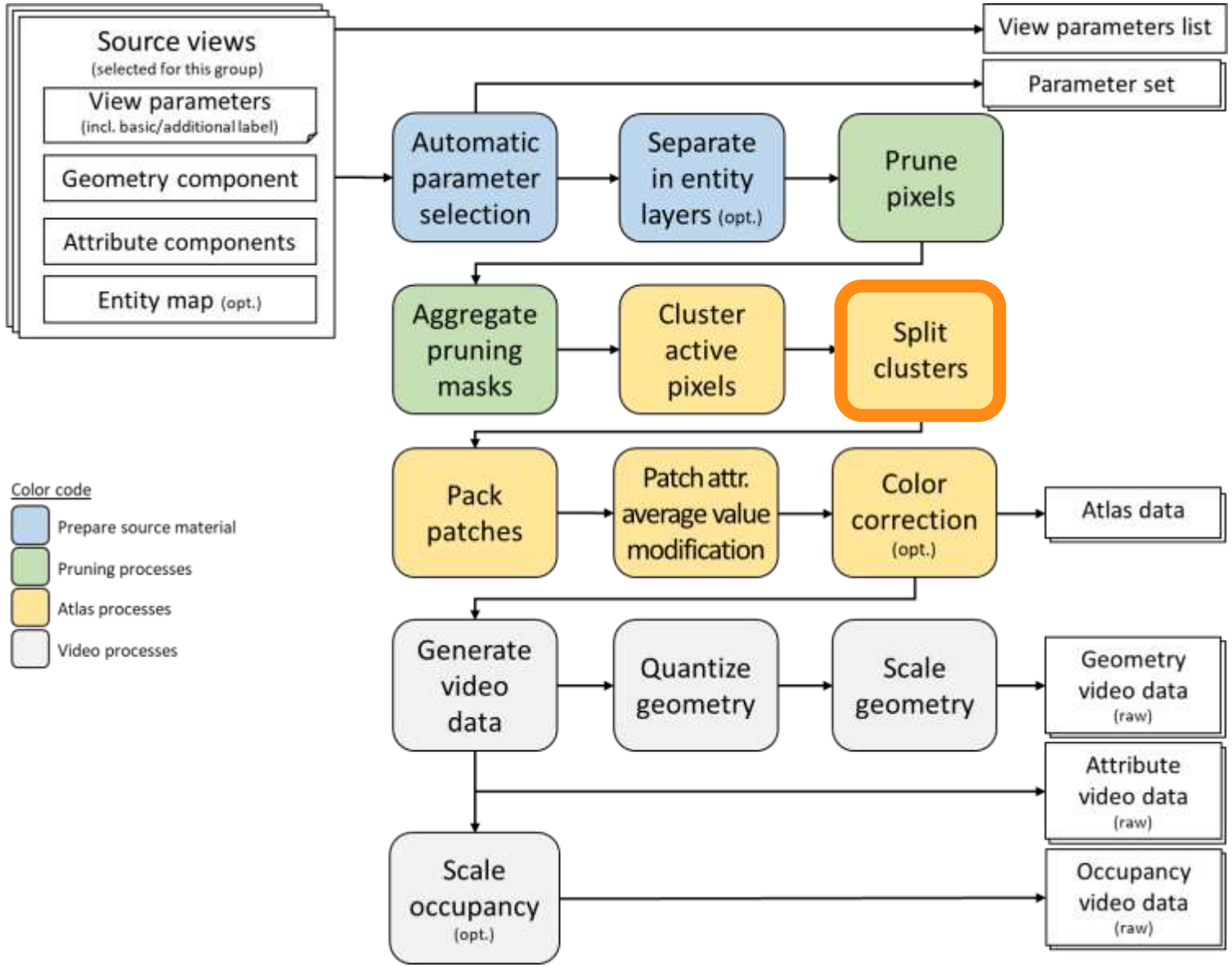
Pixel clustering



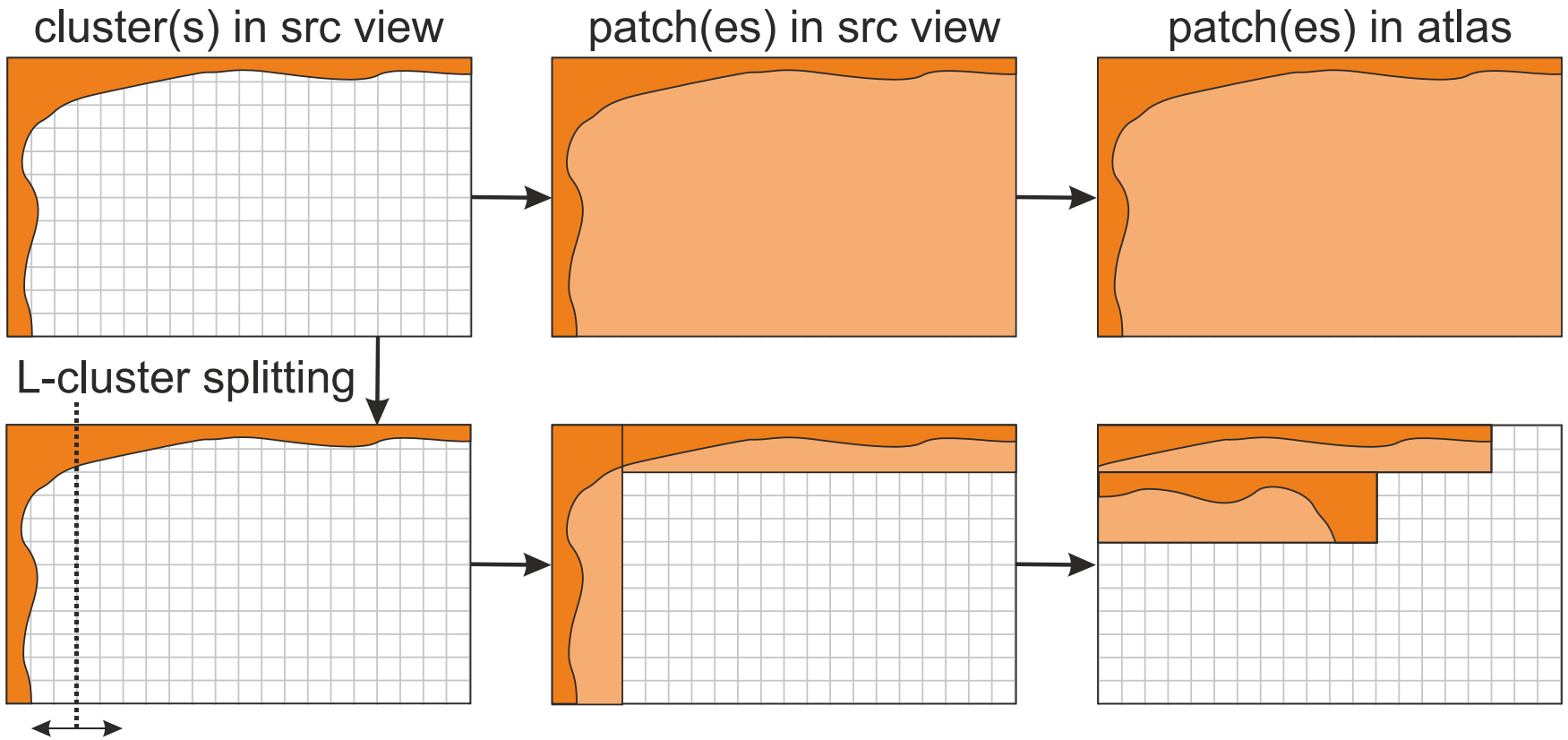
Pixel clustering: cluster merging



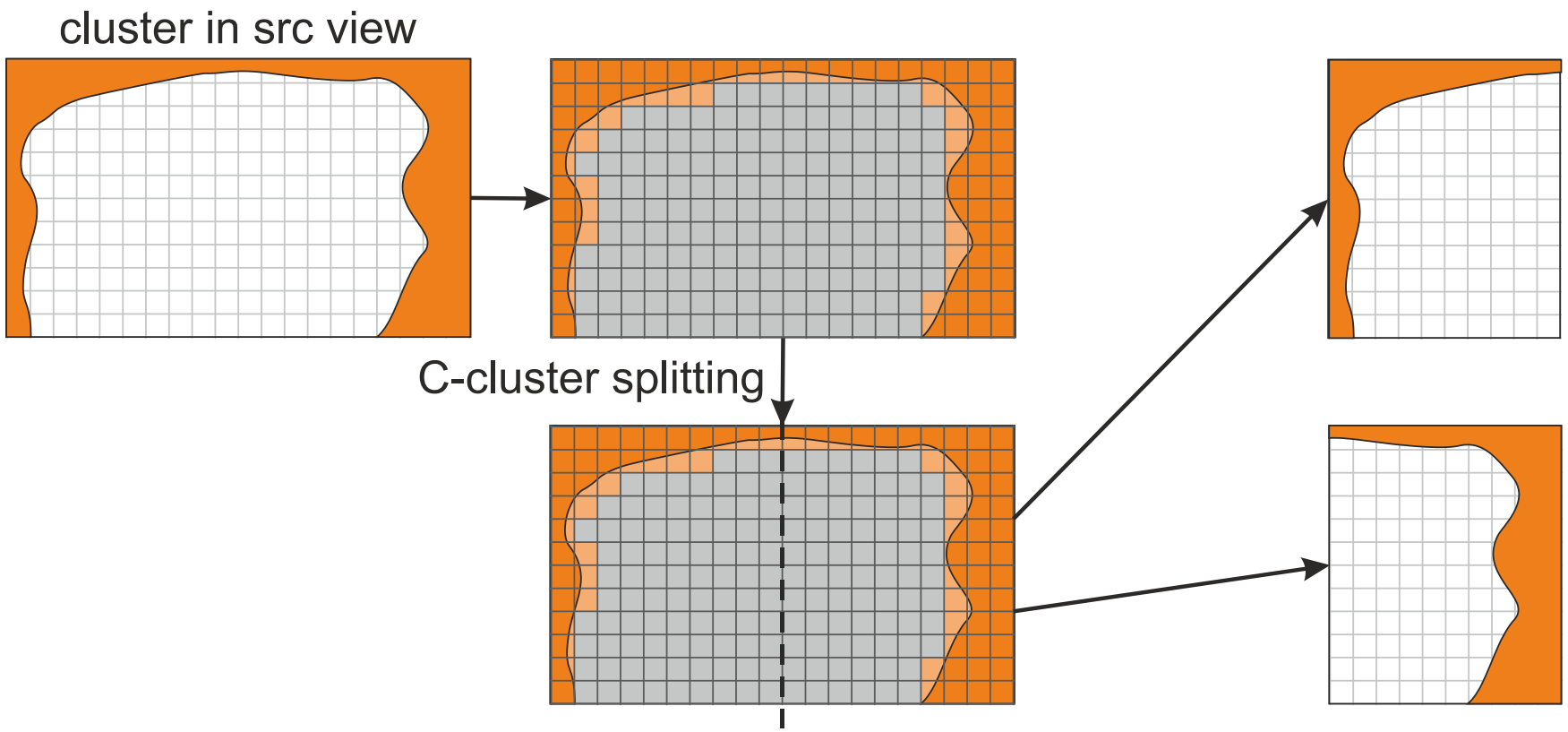
MIV – encoding of one group



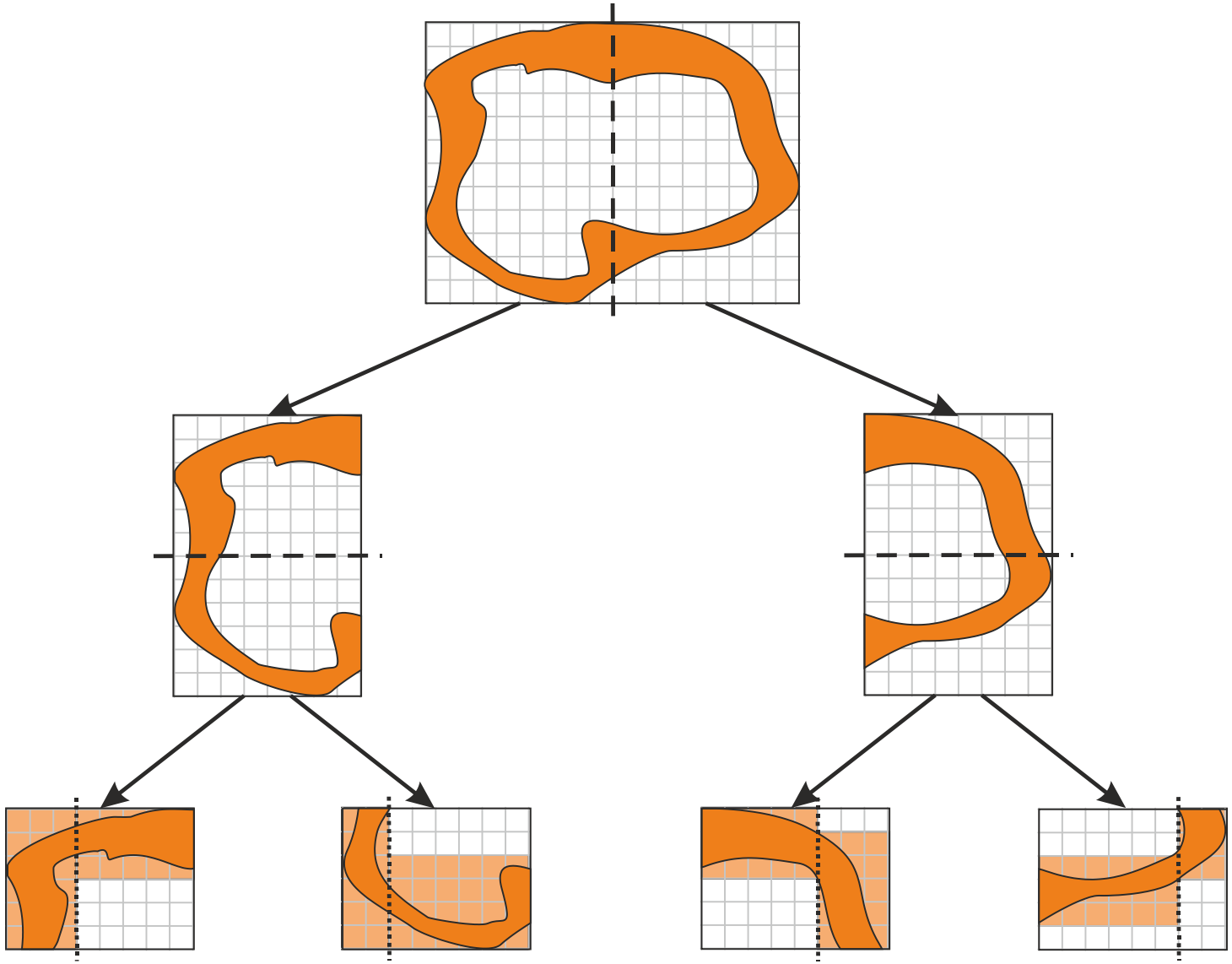
Cluster splitting



Cluster splitting



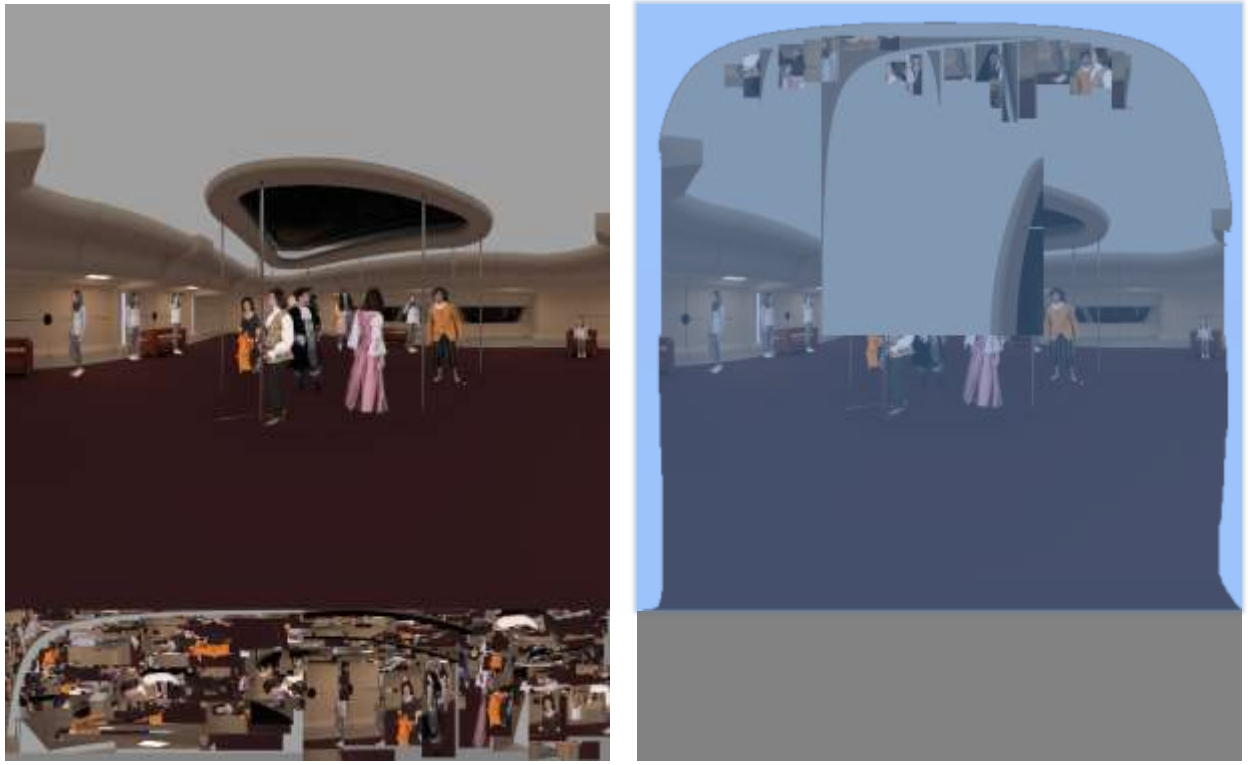
Cluster splitting



Cluster splitting

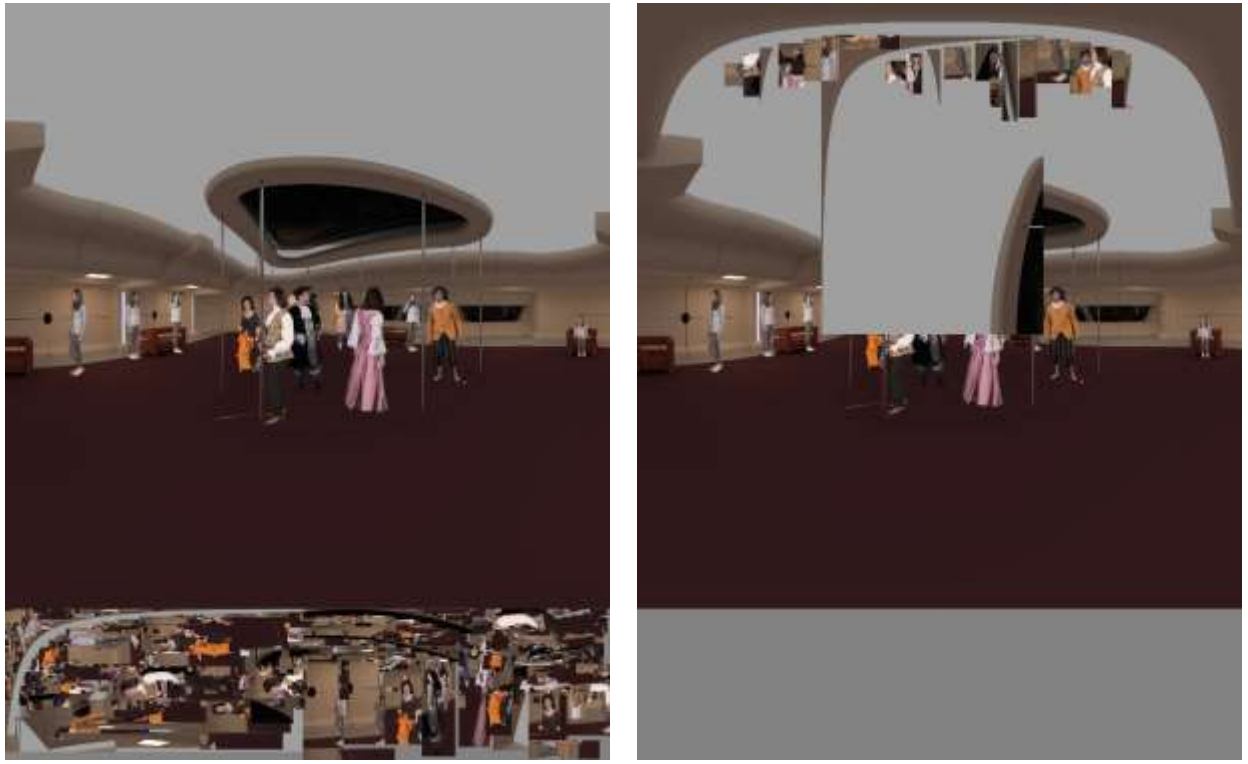
No splitting

Cluster splitting



No splitting

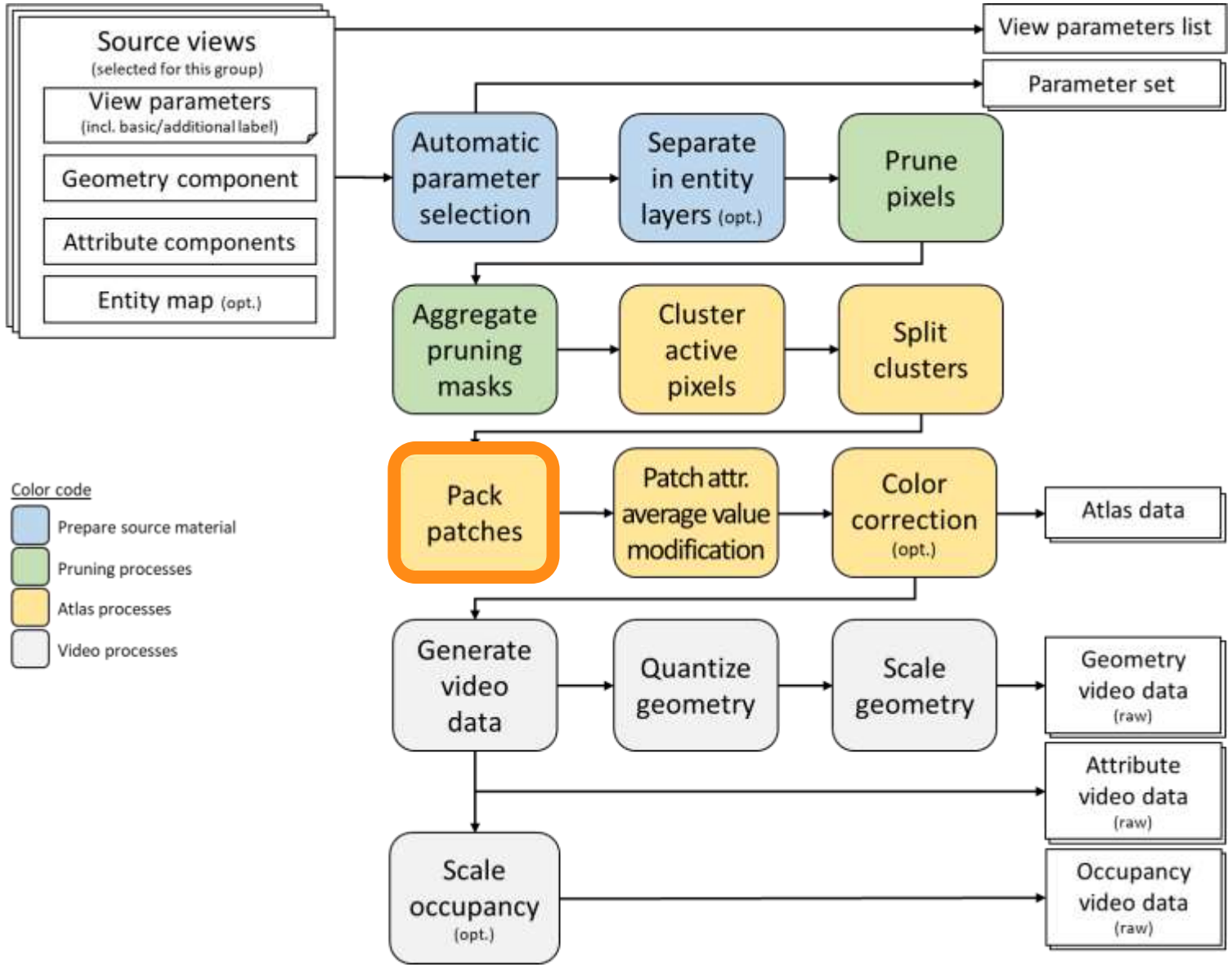
Cluster splitting



No splitting

Splitting enabled

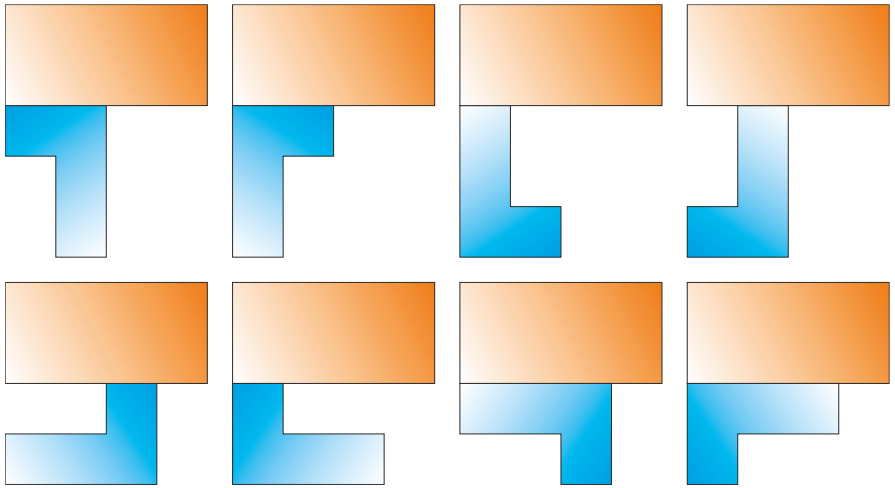
MIV – encoding of one group

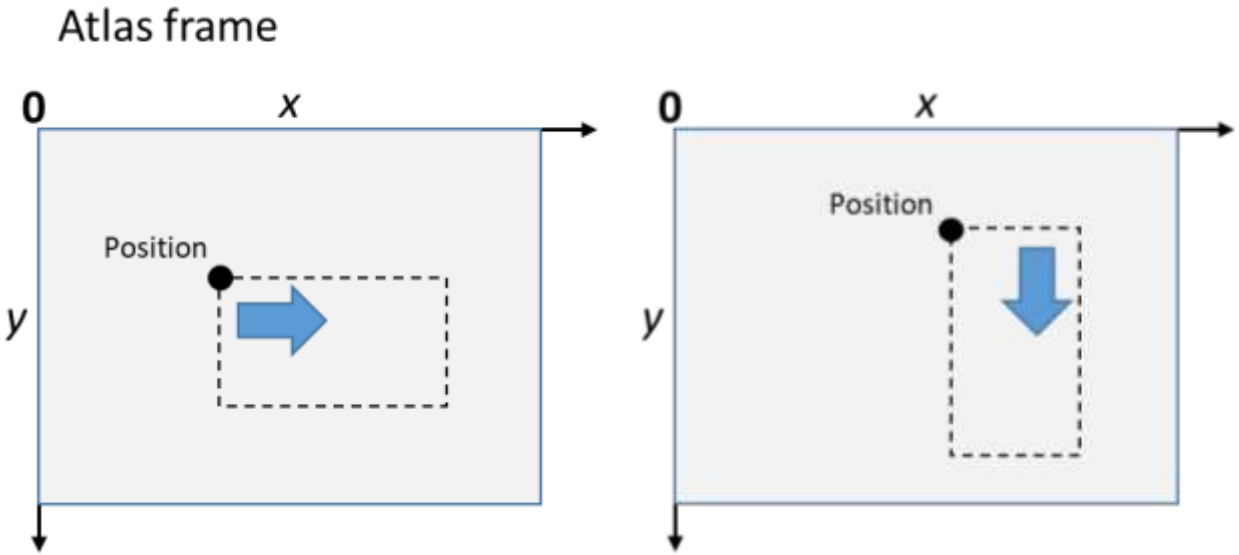


Patch packing

8 possible positioning variants:

- 4 rotations (0°, 90°, 180°, 270°)
- Optional mirroring



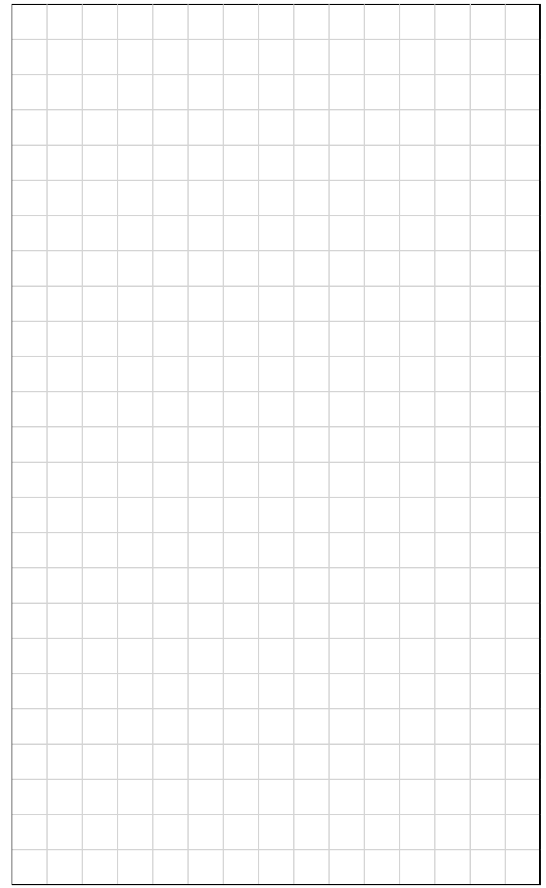


Patch packing

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

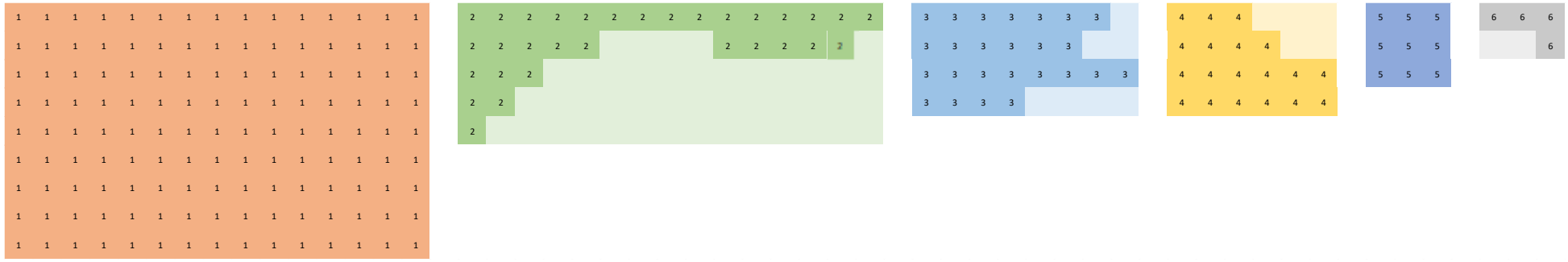
Patch packing

Atlas:



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

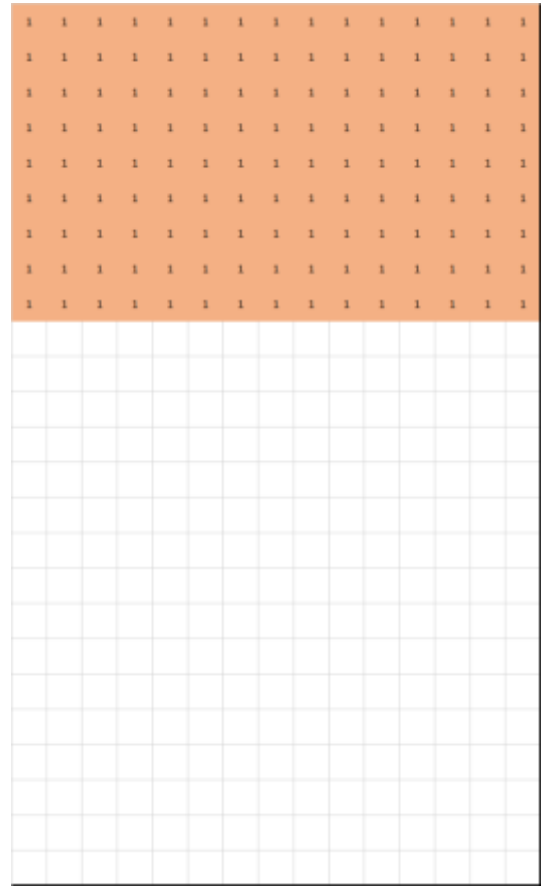
List of clusters sorted by area (1 square: 8 × 8 pixels):



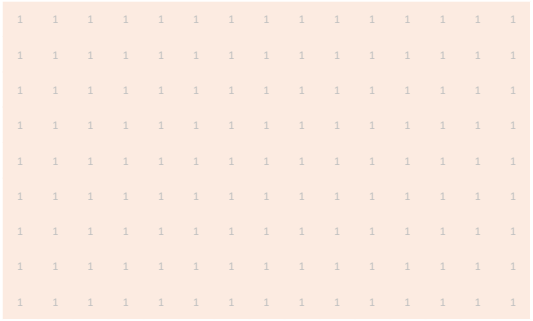
Patch packing

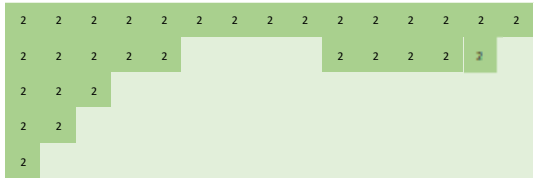
For each cluster:
    For each atlas:
        Push the cluster in “Used Space” (0° rotation first, 90° otherwise)
        If the push failed:
            Push the cluster into “Free Space” (0° rot. first, 90° otherwise)
            If the push failed:
                Split the cluster into 2 parts by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

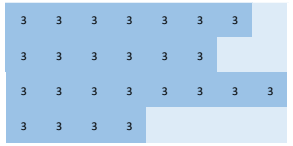
Atlas:

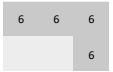


List of clusters sorted by area (1 square: 8 × 8 pixels):





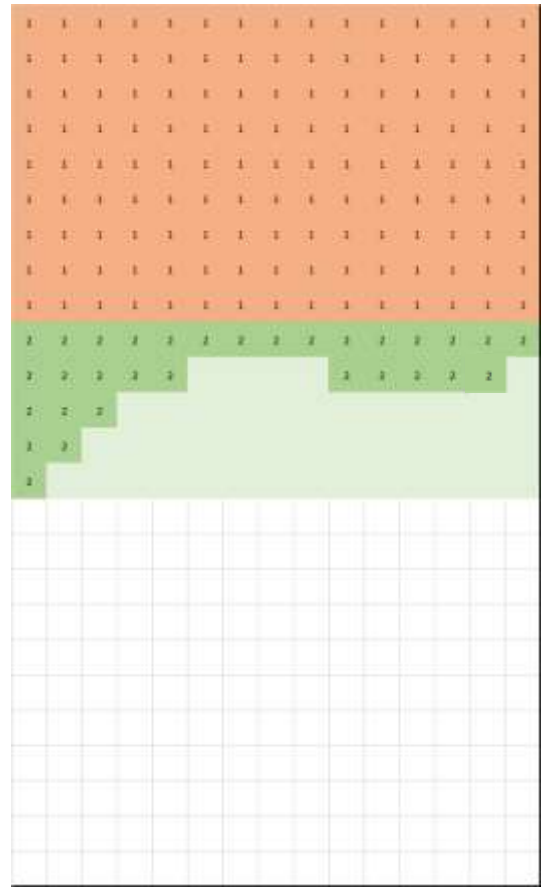




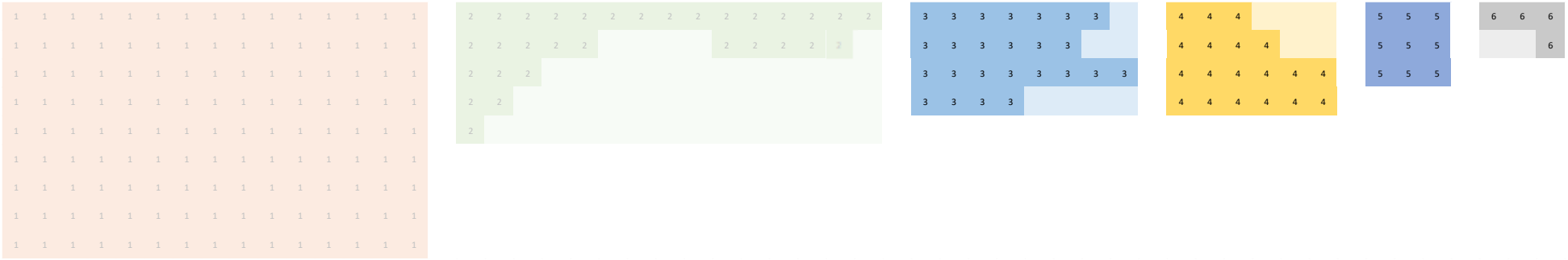
Patch packing

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

Atlas:



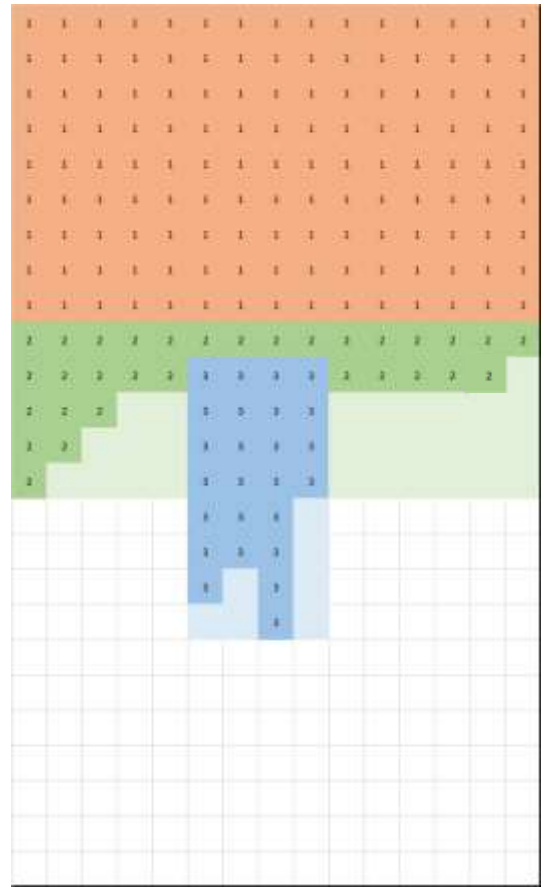
List of clusters sorted by area (1 square: 8 × 8 pixels):



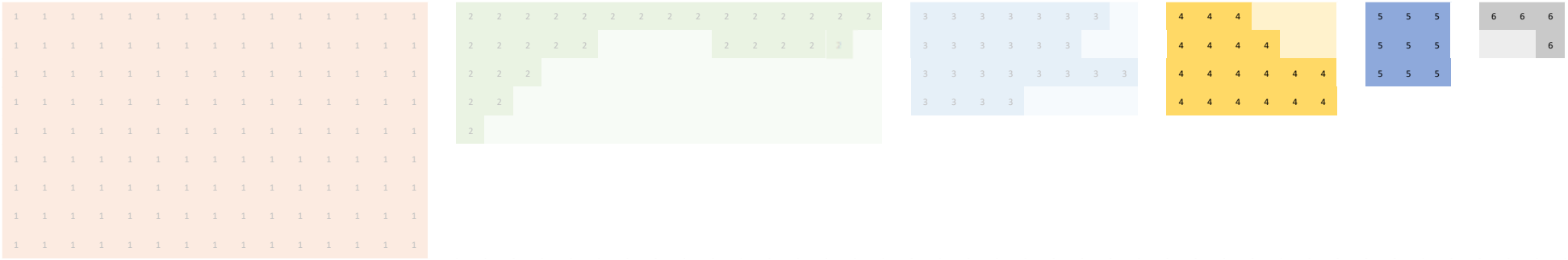
Patch packing

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

Atlas:



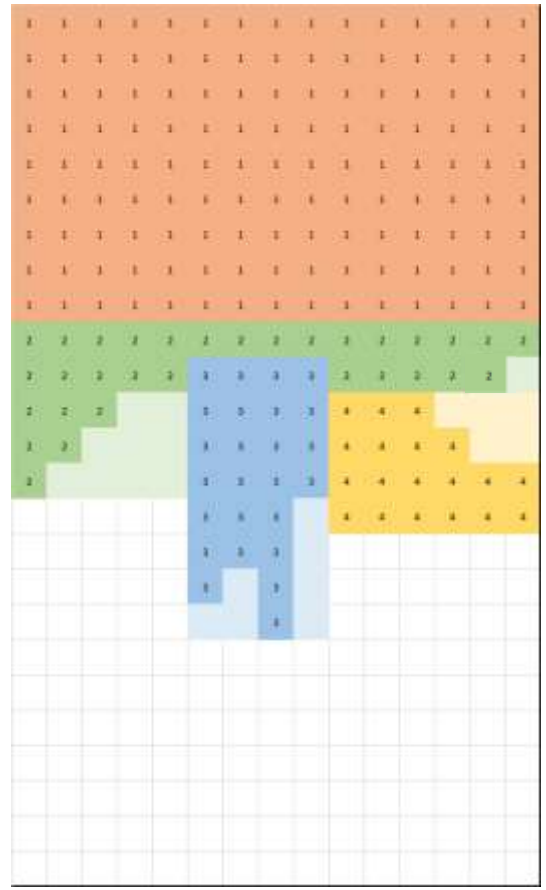
List of clusters sorted by area (1 square: 8 × 8 pixels):



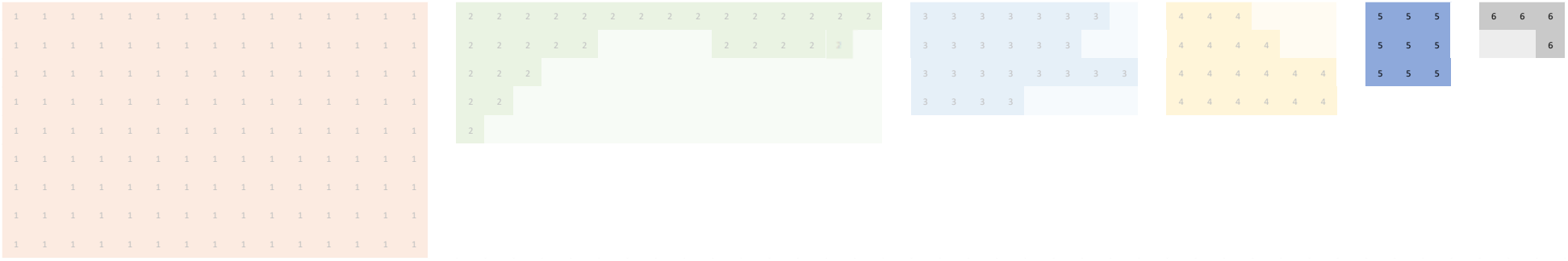
Patch packing

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

Atlas:



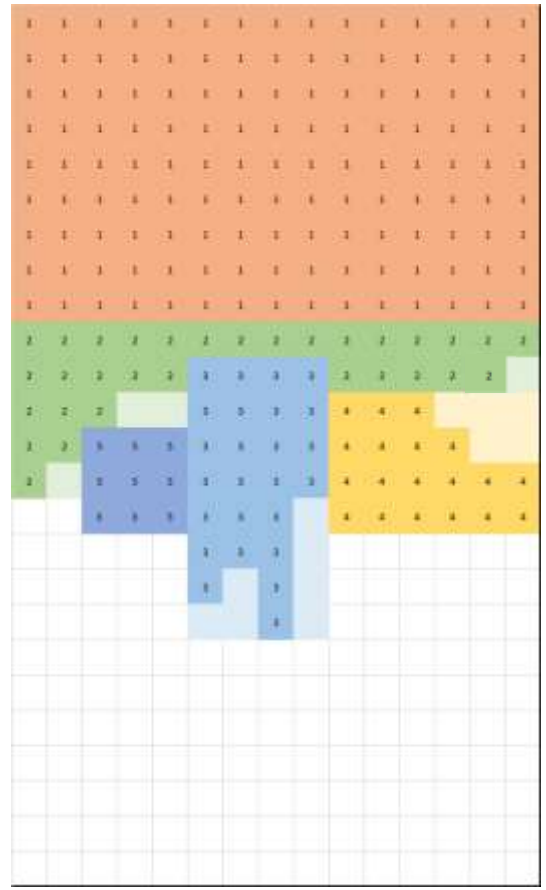
List of clusters sorted by area (1 square: 8 × 8 pixels):



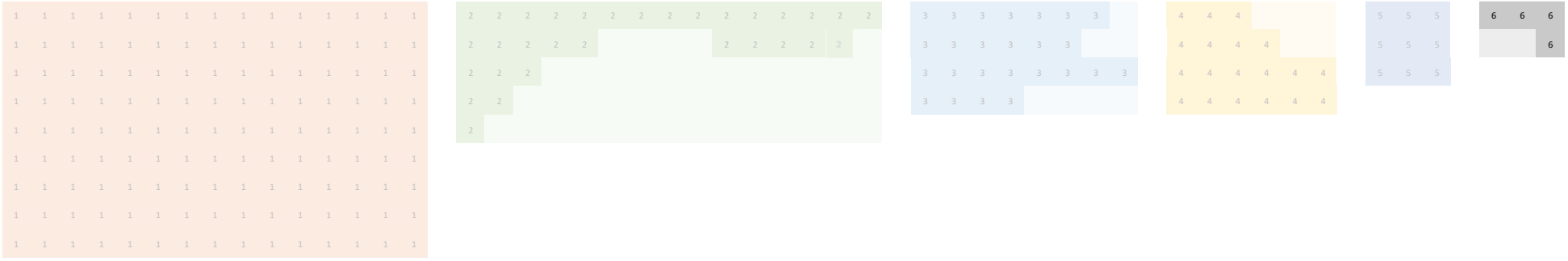
Patch packing

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

Atlas:



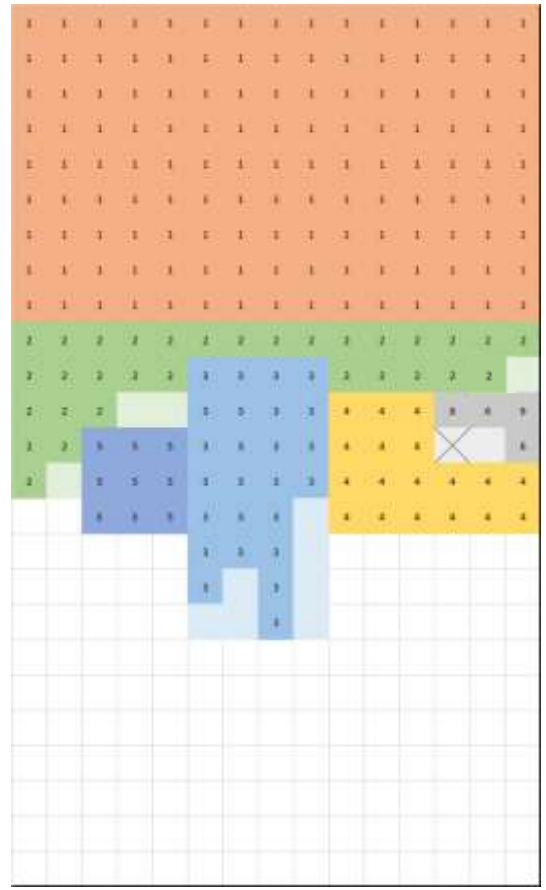
List of clusters sorted by area (1 square: 8 × 8 pixels):



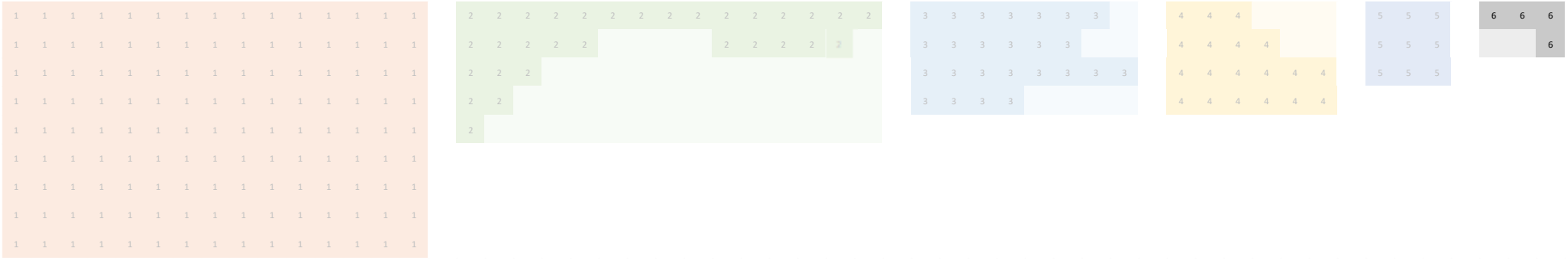
Patch packing

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

Atlas:



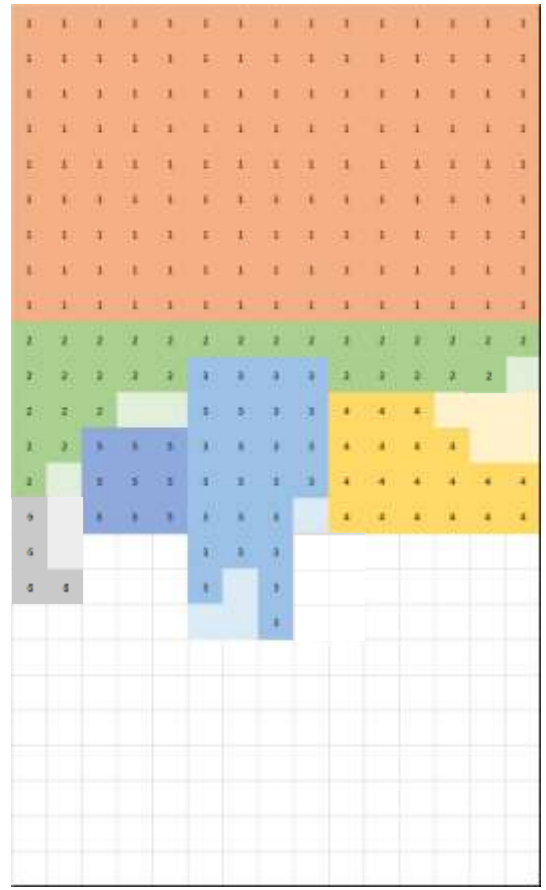
List of clusters sorted by area (1 square: 8 × 8 pixels):



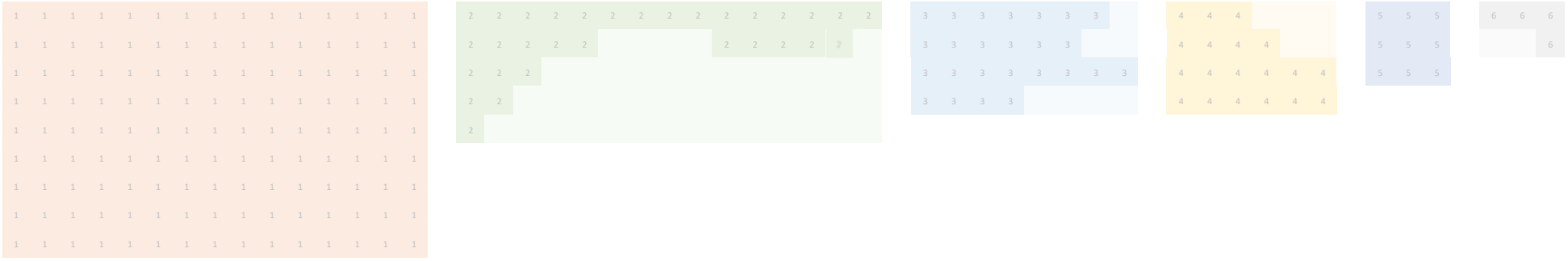
Patch packing

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

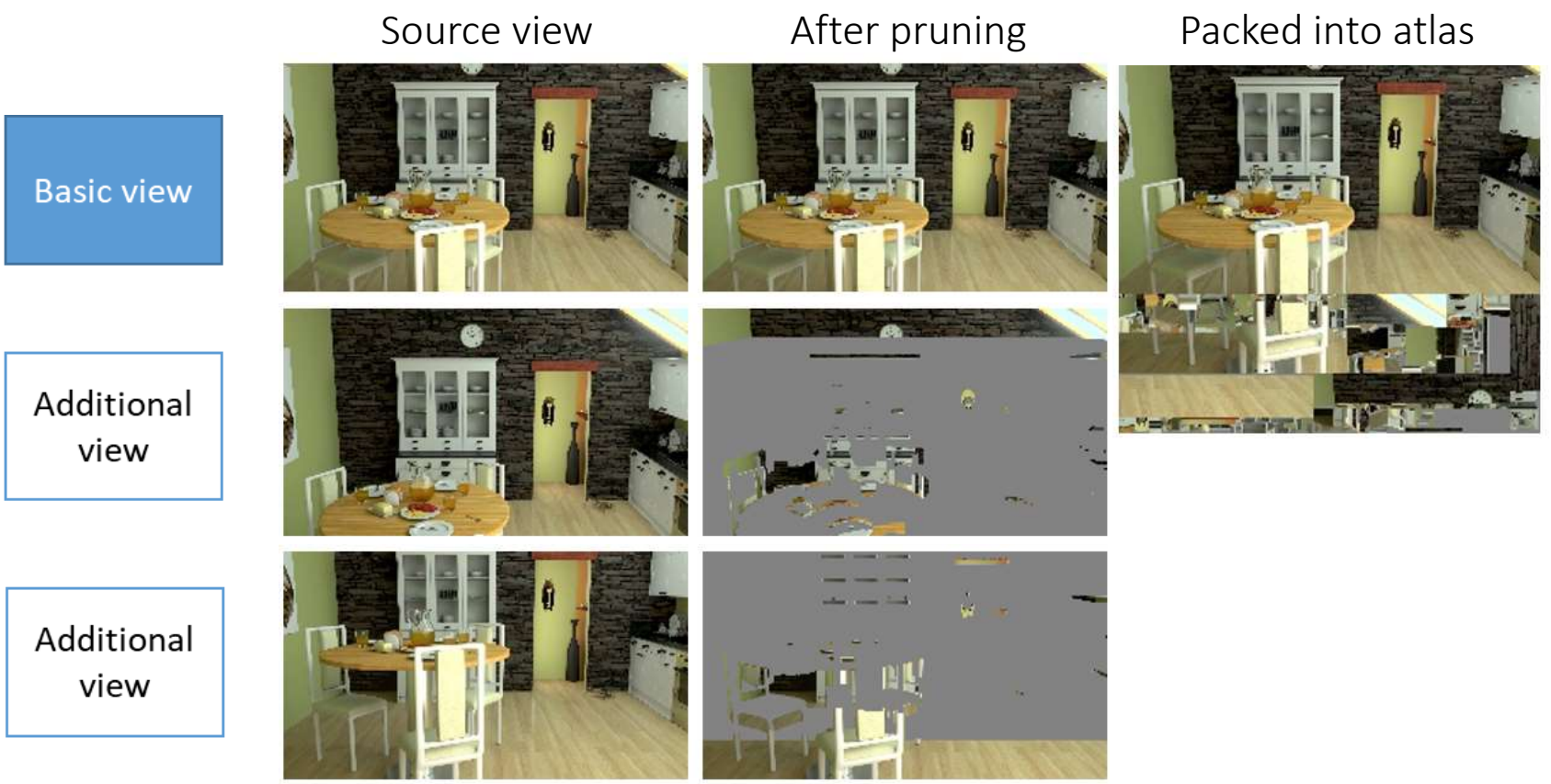
Atlas:



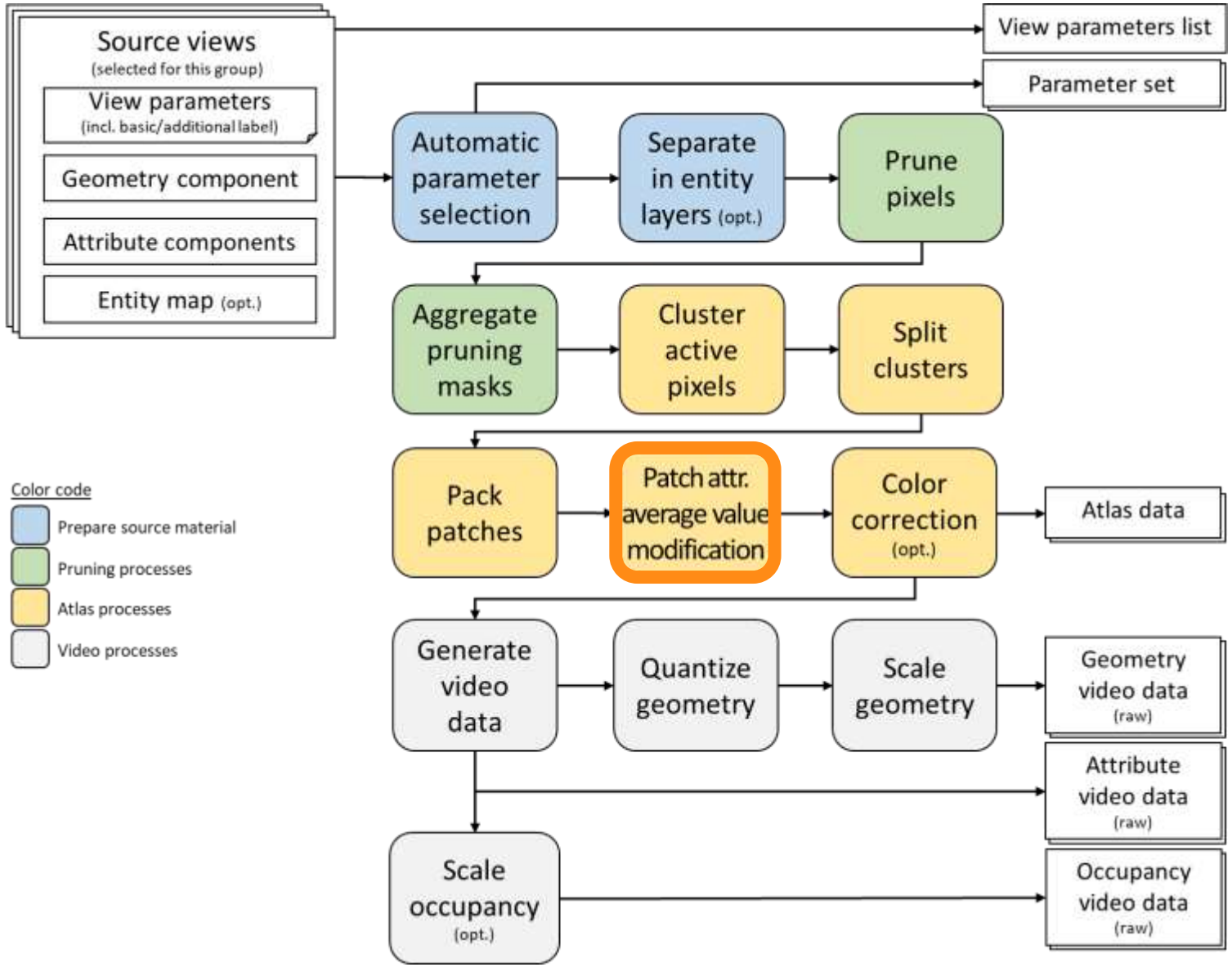
List of clusters sorted by area (1 square: 8 × 8 pixels):



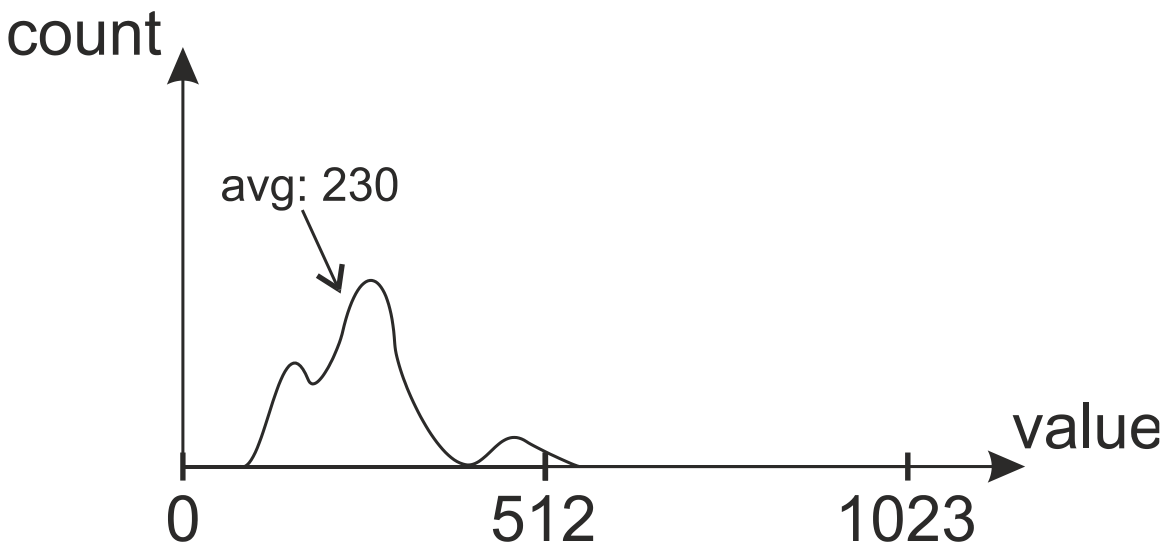
Patch packing



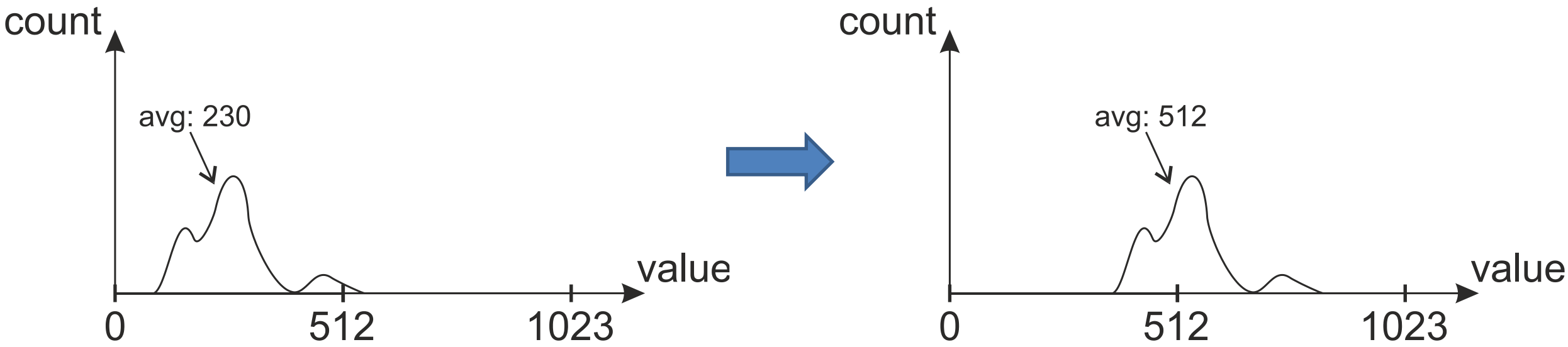
MIV – encoding of one group



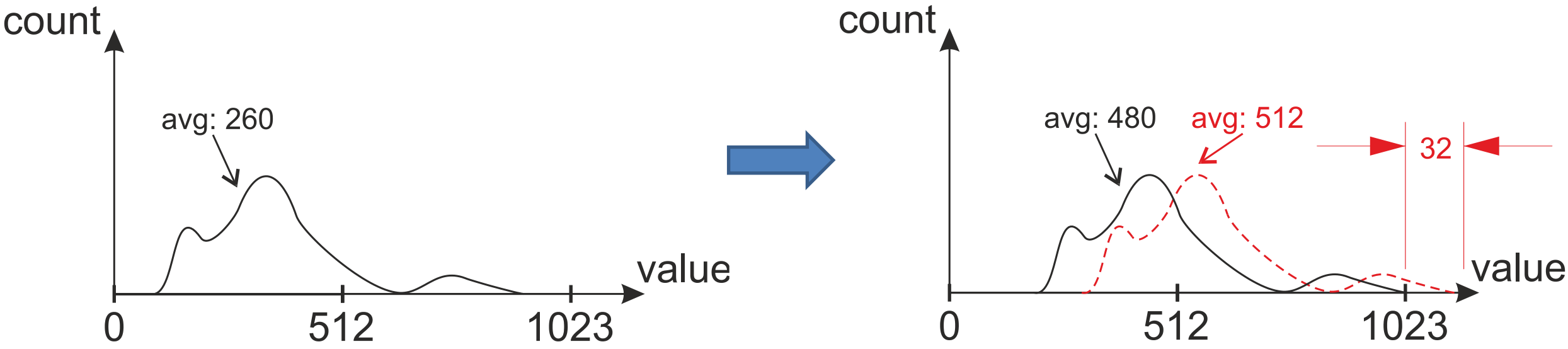
Patch attribute value modification



Patch attribute value modification

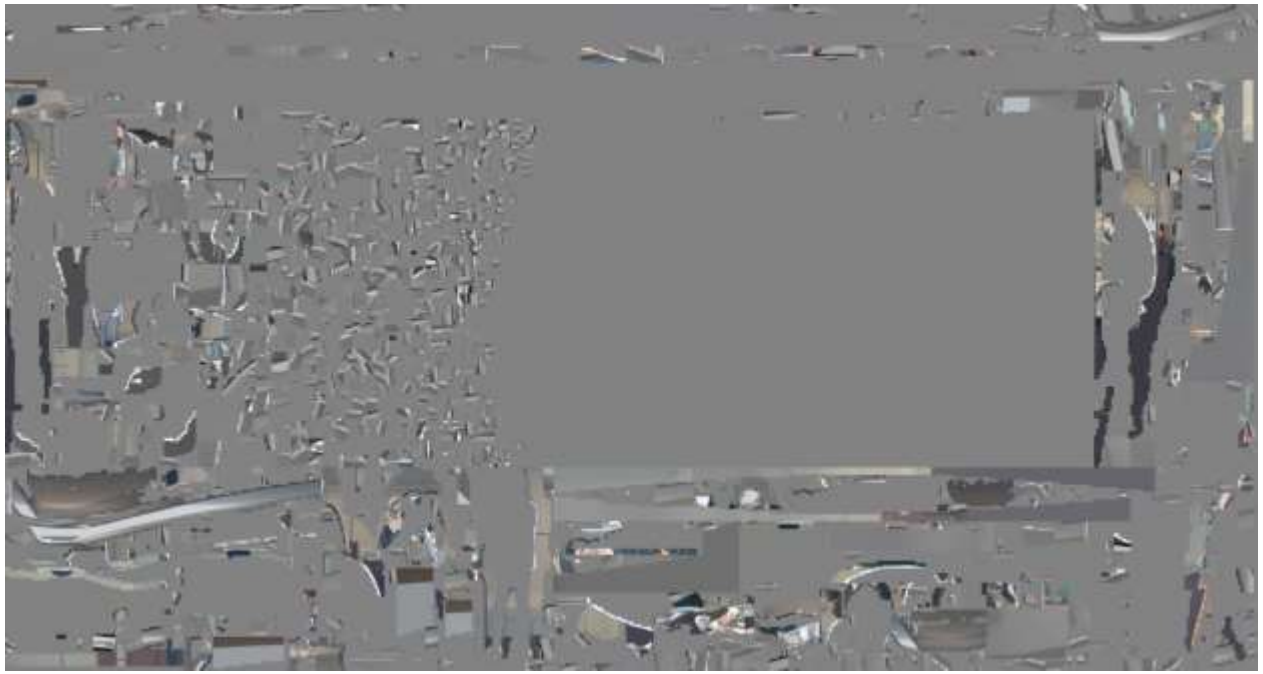


Patch attribute value modification



Patch attribute value modification

Without mean patch value modification



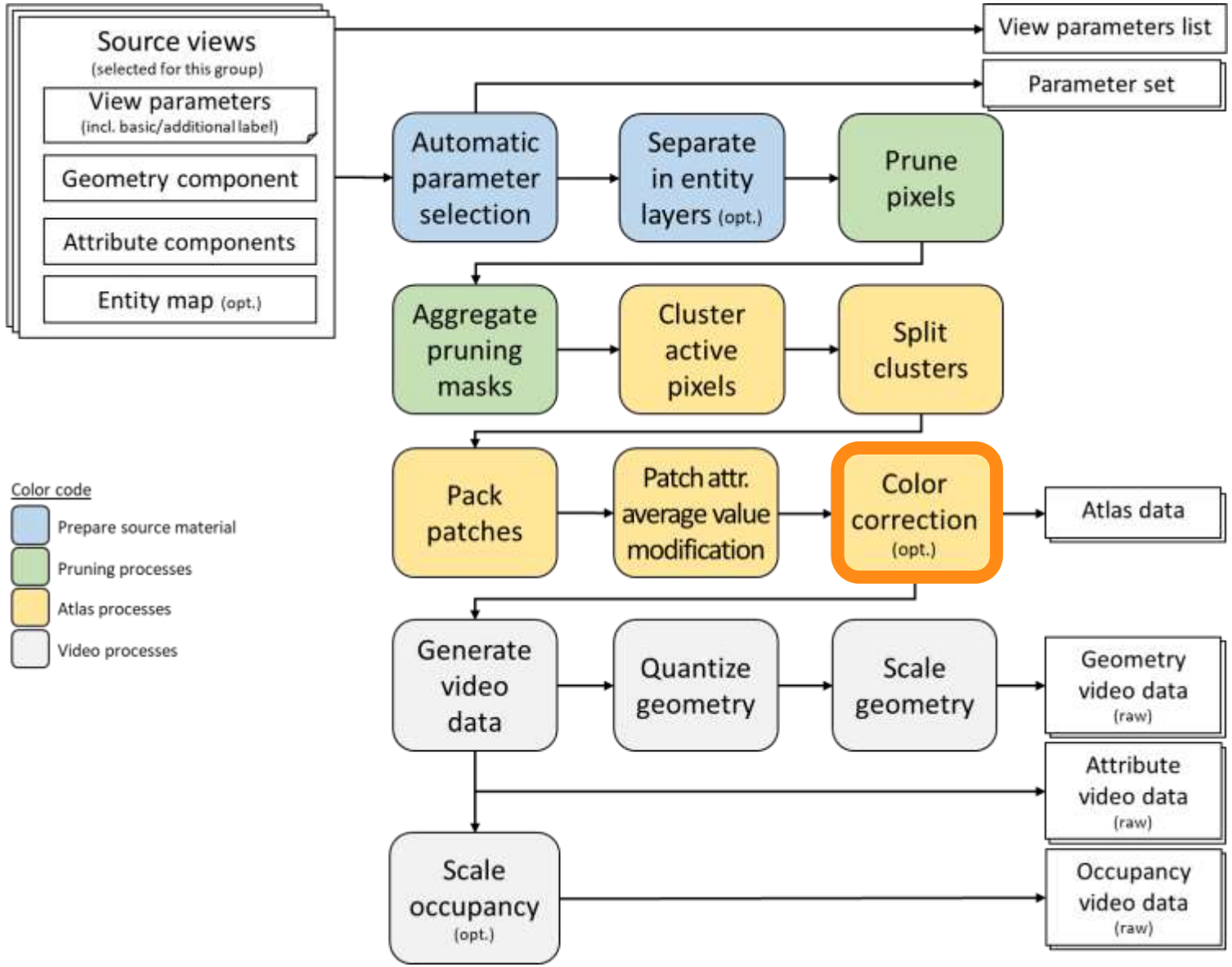
Mean patch value modification enabled

Patch attribute value modification

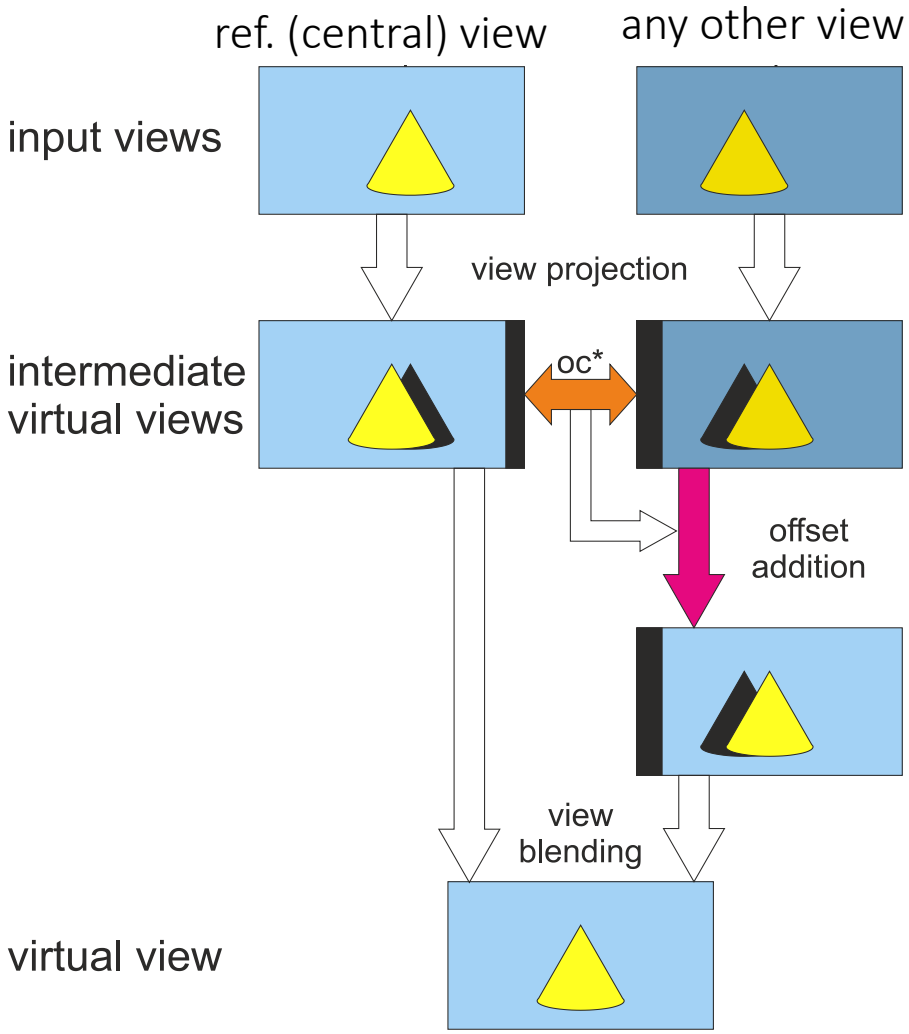
Without mean patch value modification

Mean patch value modification enabled

MIV – encoding of one group



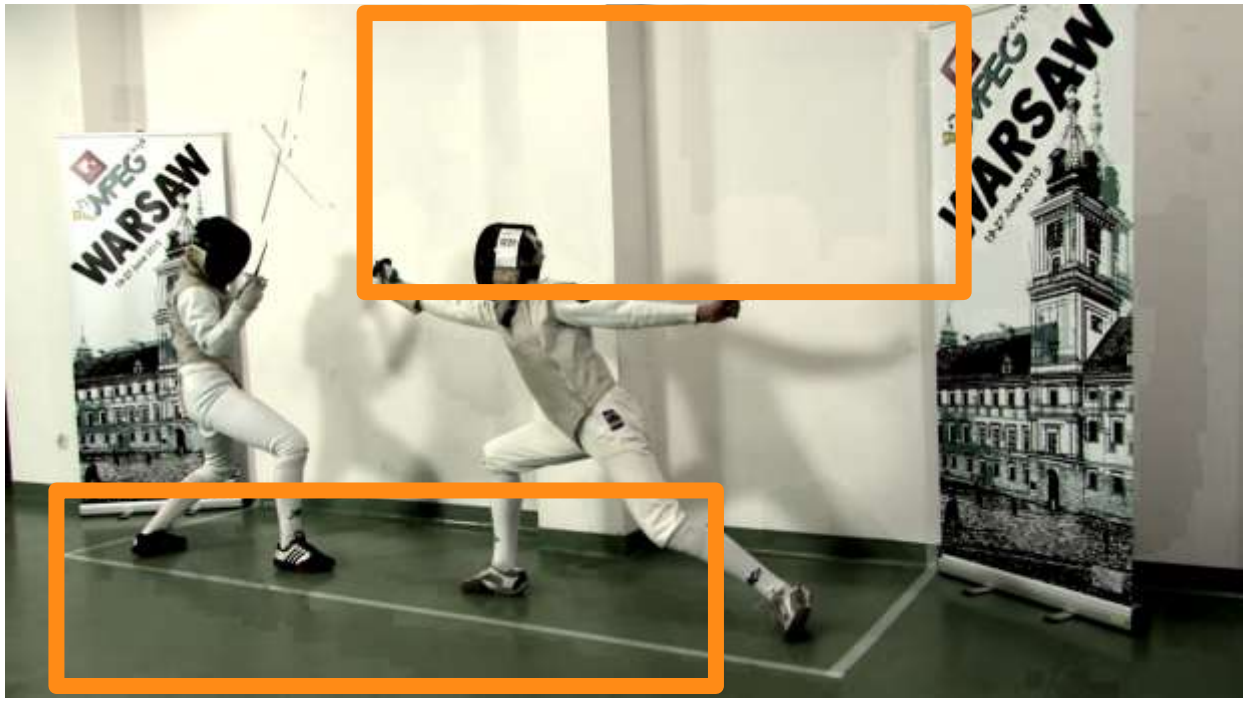
Color correction



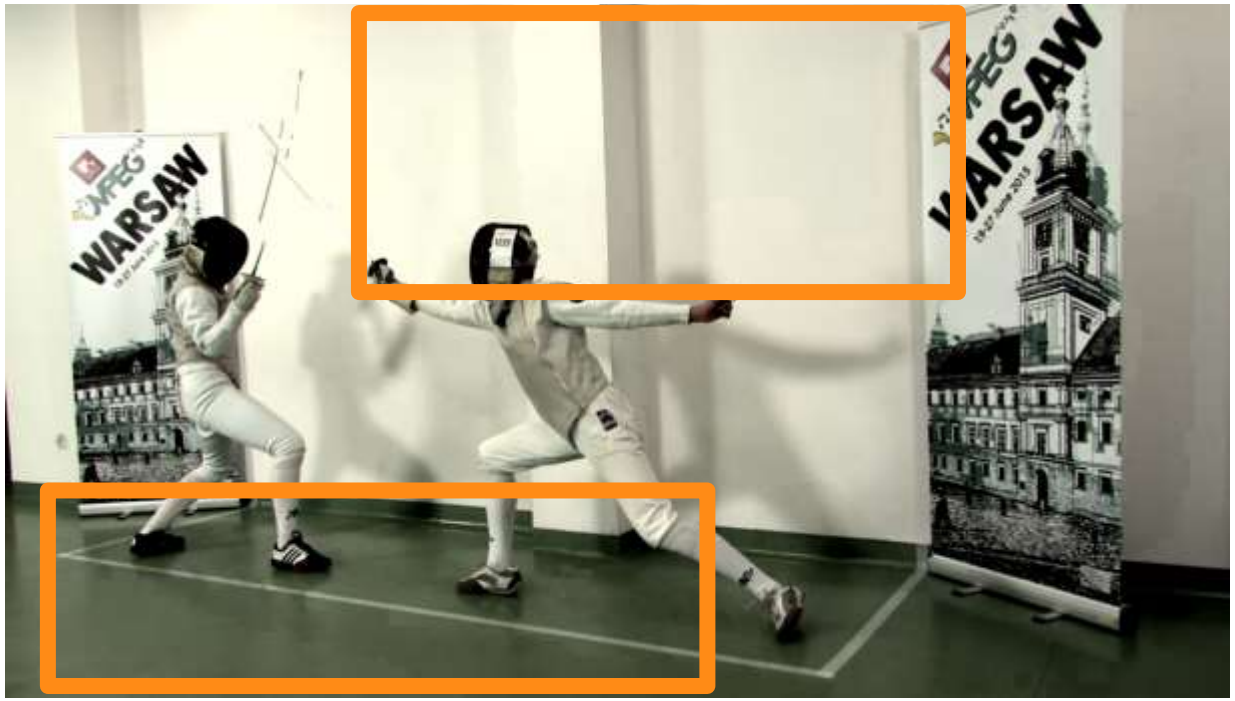
*oc – offset calculation

- 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

Color correction

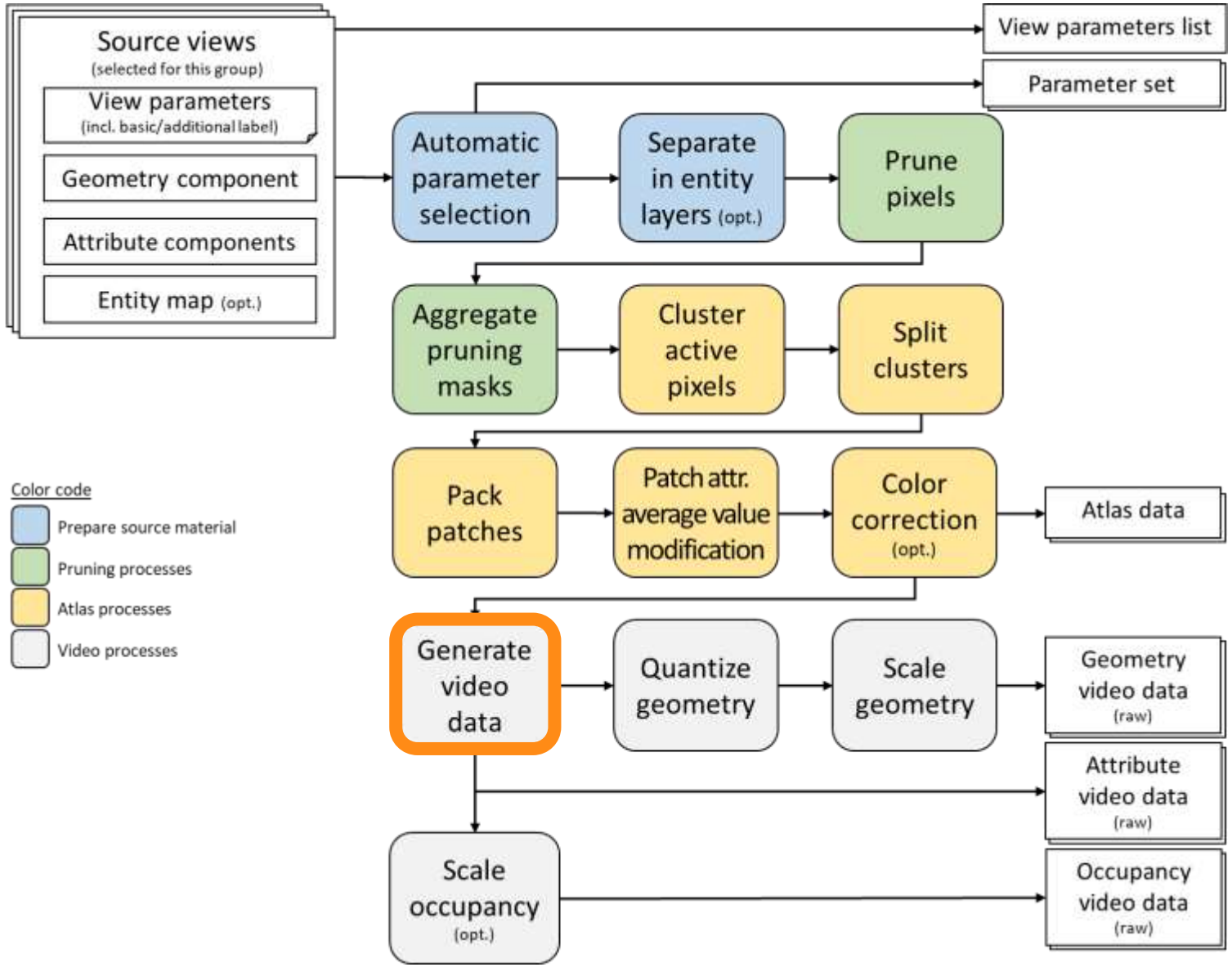


Without color correction



Color correction enabled

MIV – encoding of one group



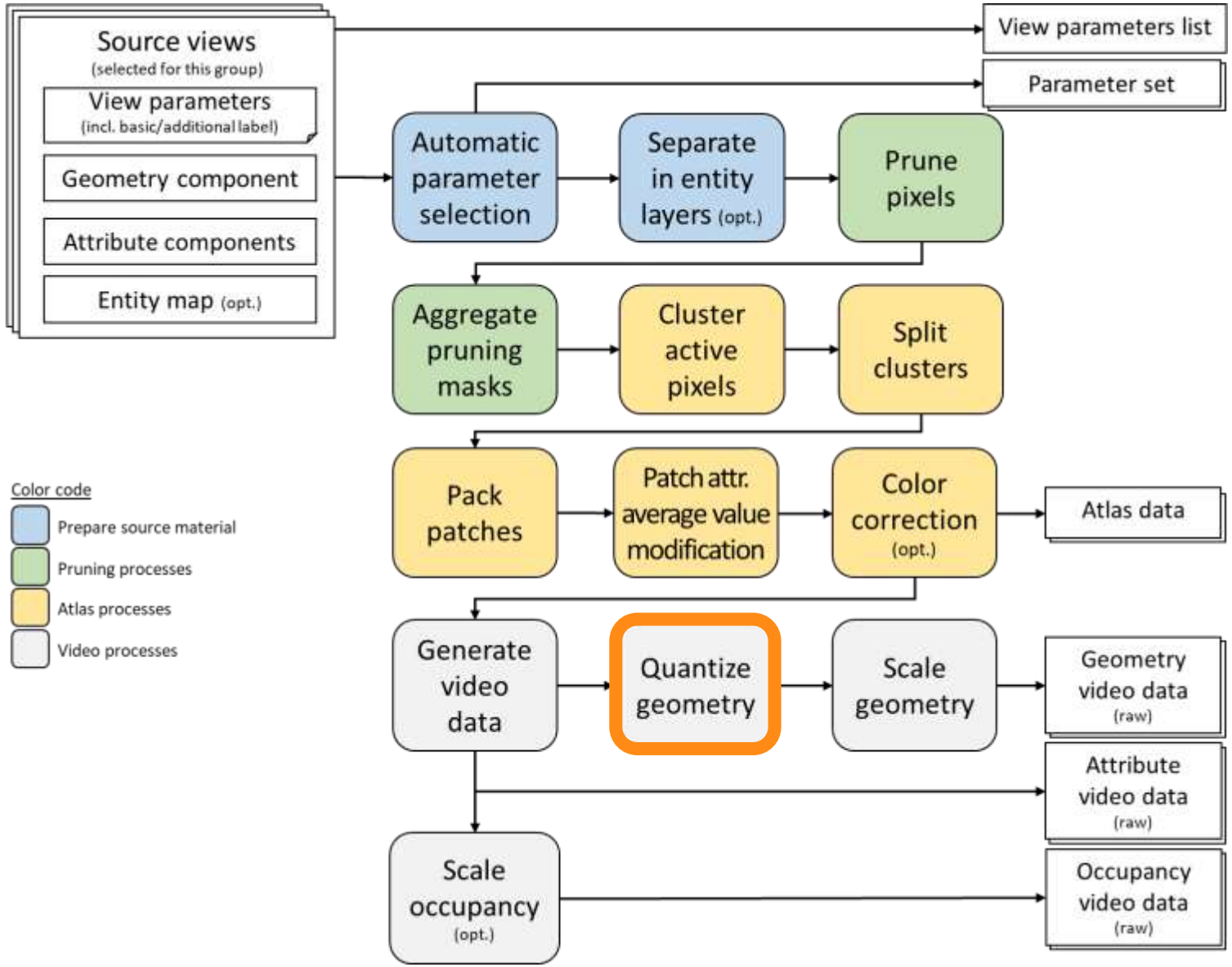
Video data generation: temporal redundancy removal

- 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

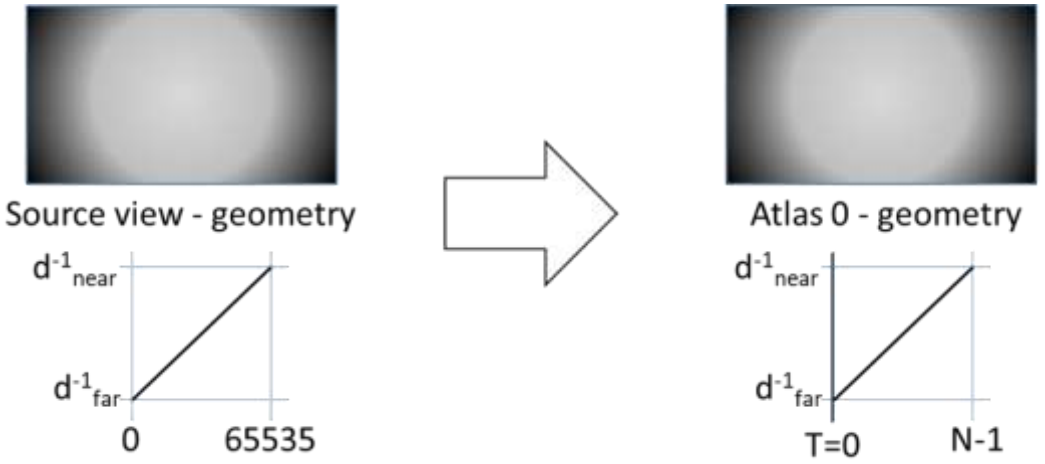
Without temporal redundancy removal

Temporal redundancy removal enabled

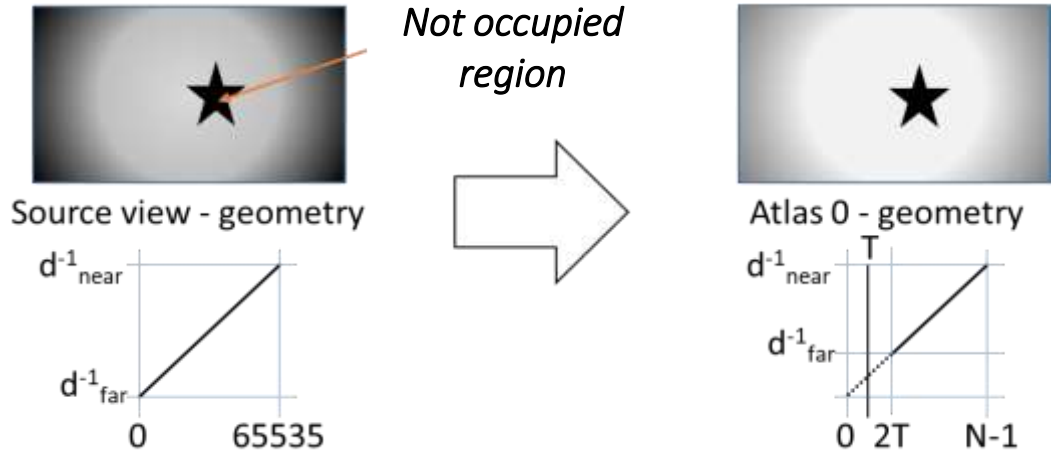
MIV – encoding of one group



Geometry quantization



base view



additional view

Filling the entire available dynamic range independently for each cluster:

$$d_{near}^{-1} = \max_{\substack{all\ pixels\ of \\ cluster\ in\ GOP}} d^{-1}(pixel)$$

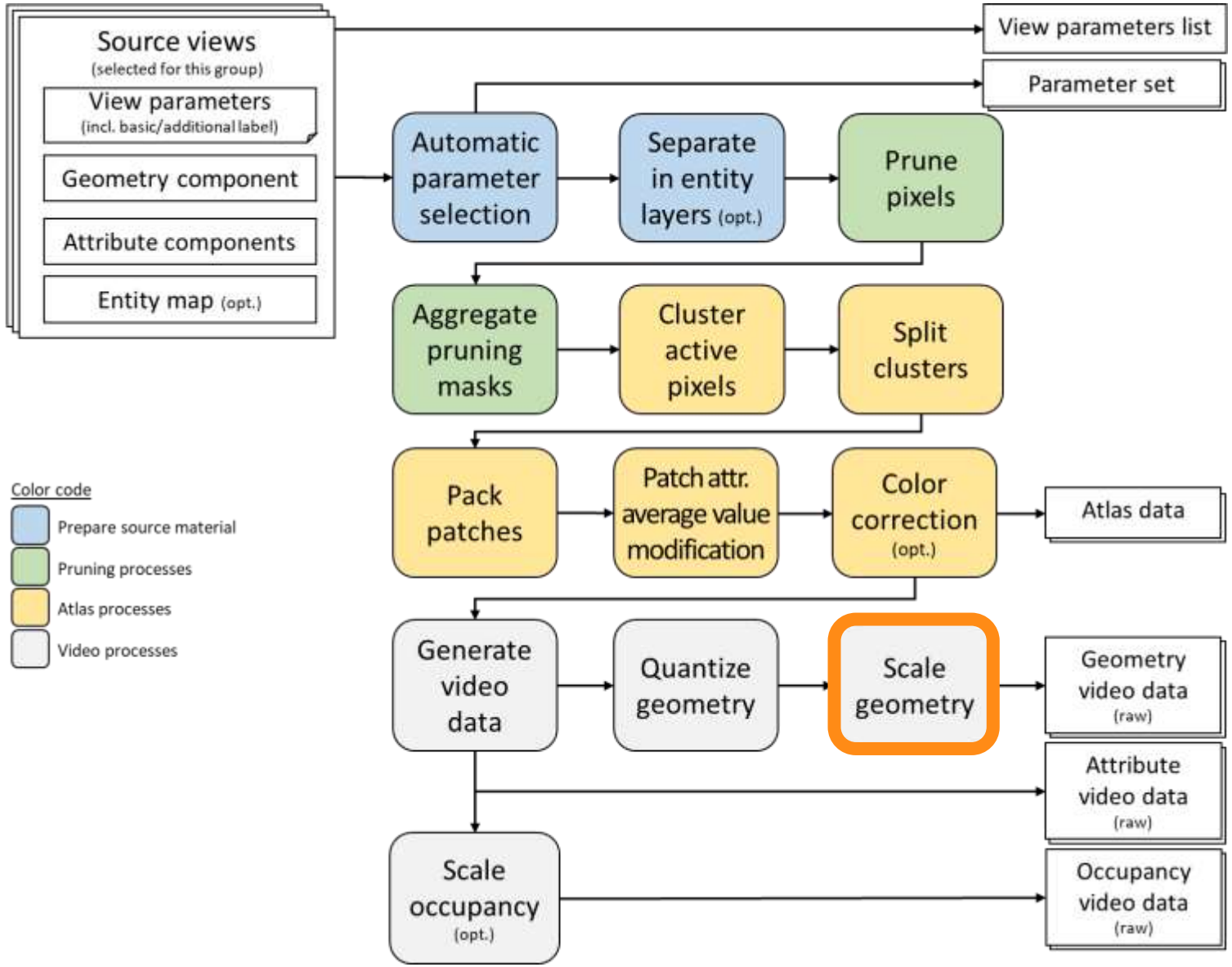
$$d_{far}^{-1} = \min_{\substack{all\ pixels\ of \\ cluster\ in\ GOP}} d^{-1}(pixel)$$

Geometry quantization

Before scaling

After scaling

MIV – encoding of one group



Geometry downscaling

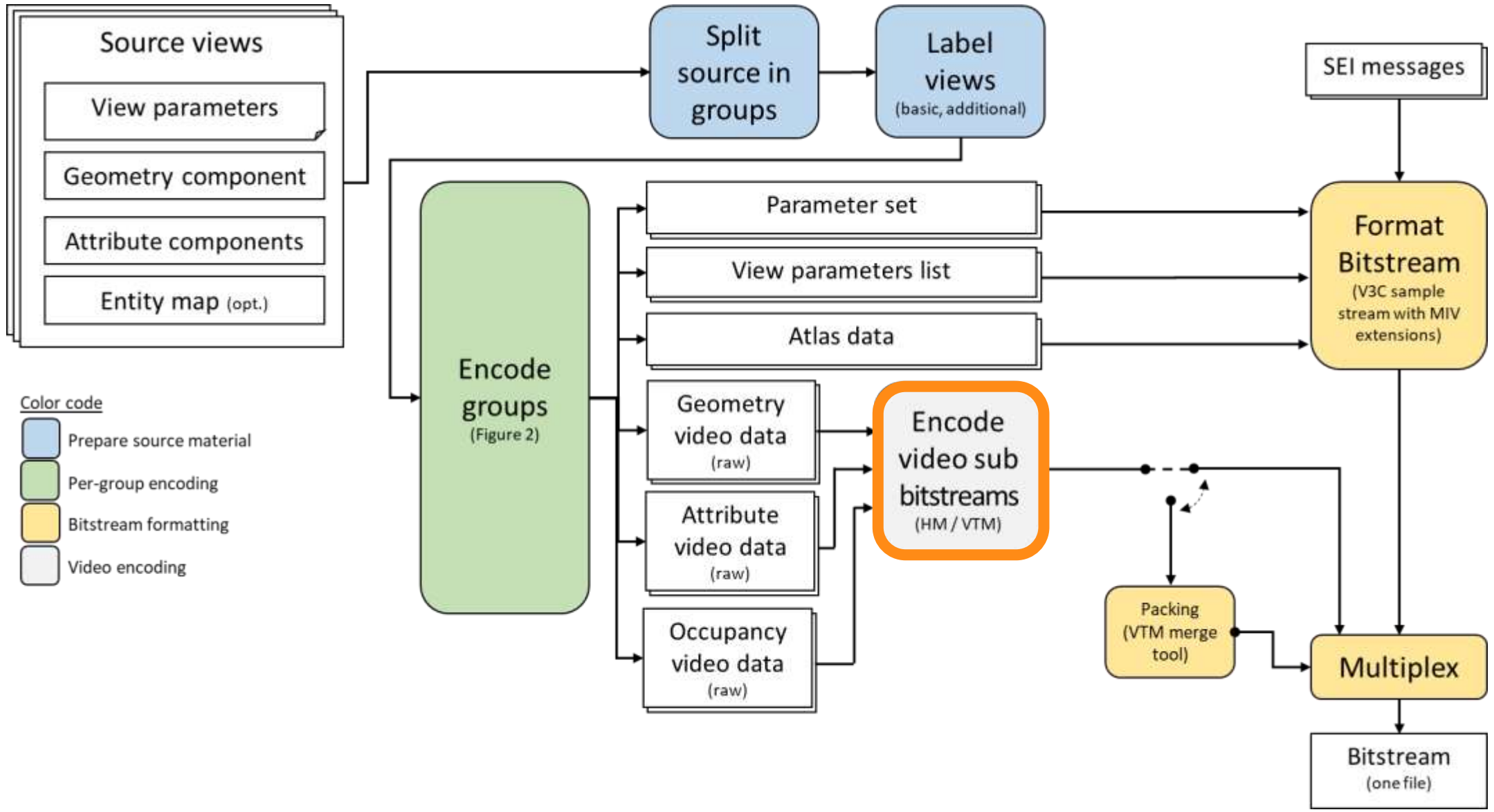
Attribute atlas: 1920 × 4640
Geometry atlas: 1920 × 4640

Geometry downscaling

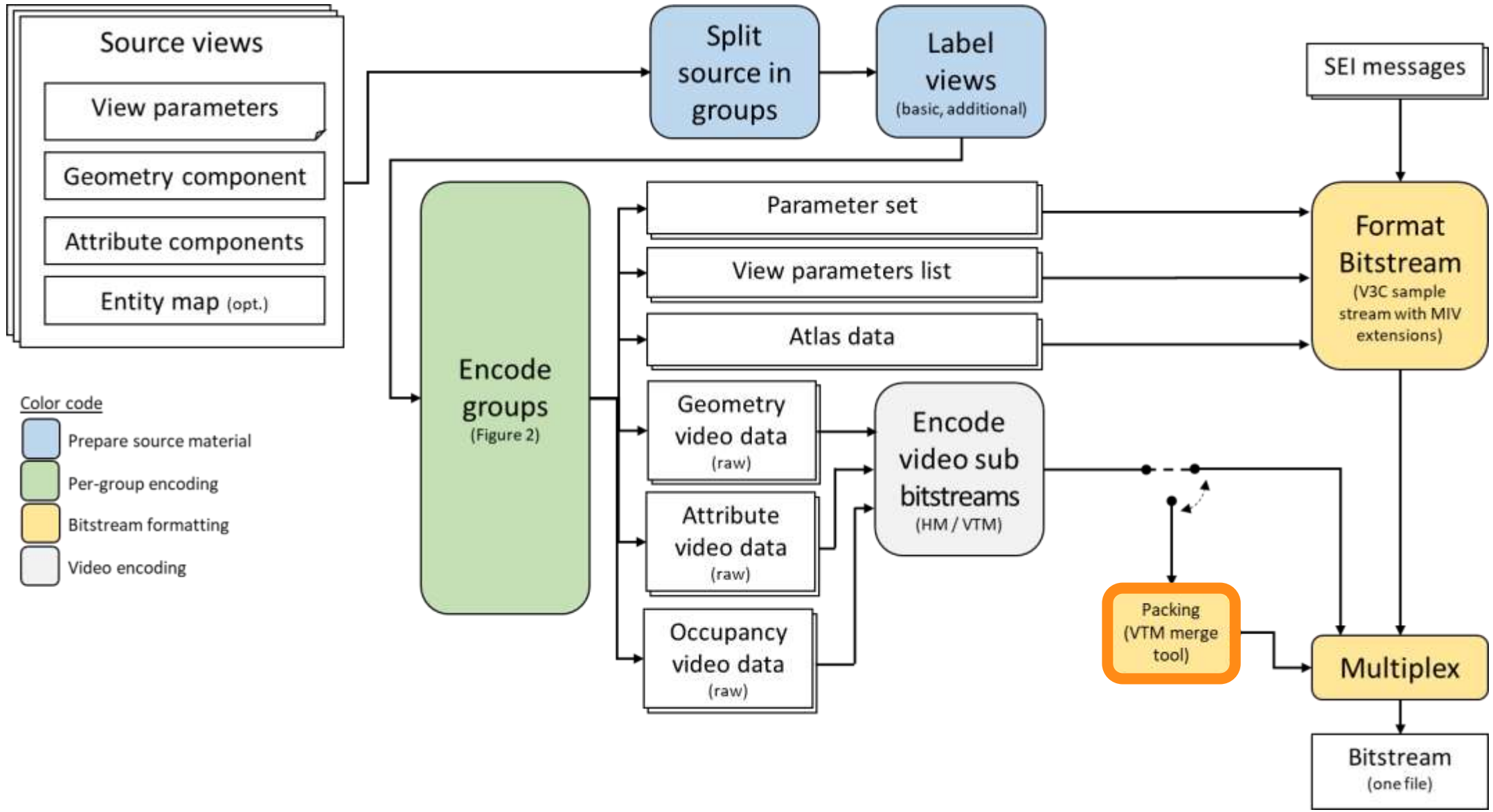
Attribute atlas: 1920 × 4640
Geometry atlas: 960 × 2320

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

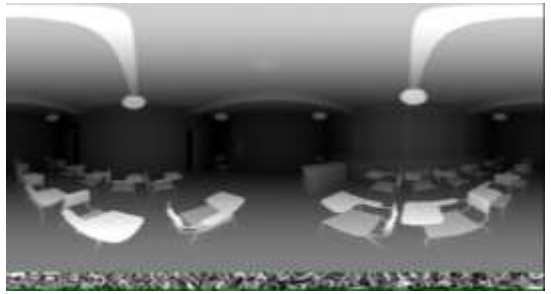
MIV encoder

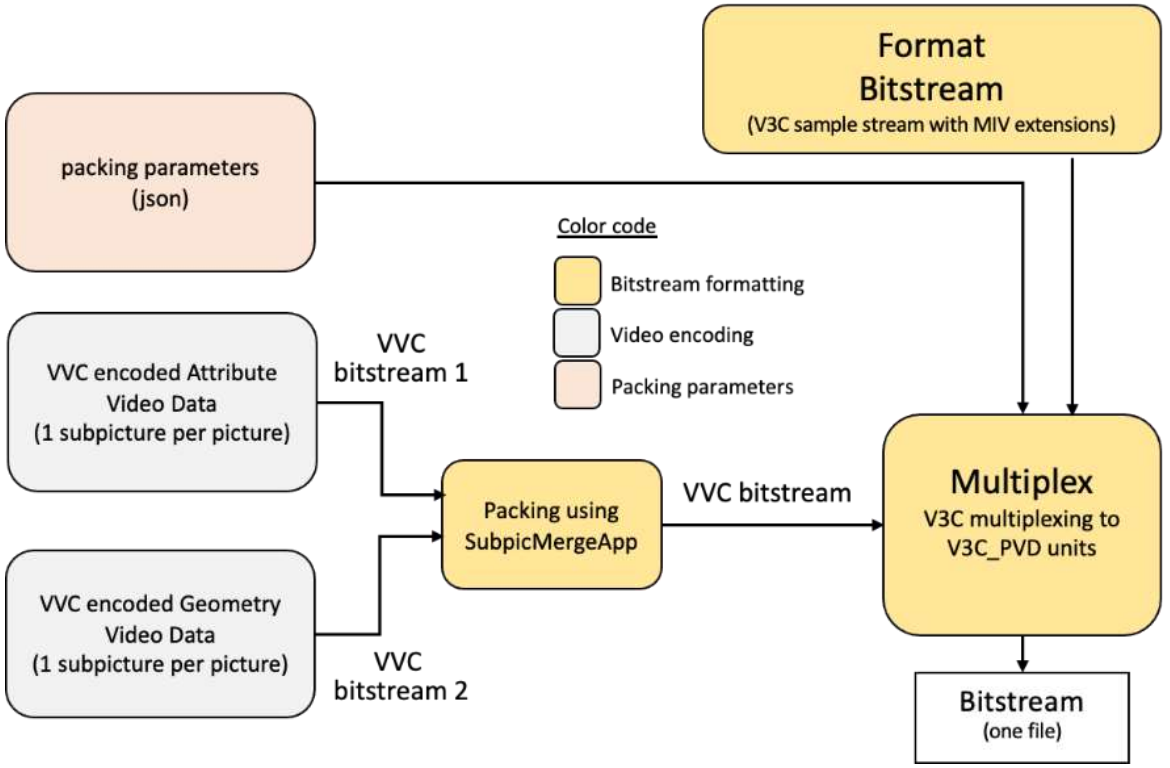

Source views
View parameters
Geometry component
Attribute components
Entity map (opt.)
Split source in groups
Label views
(basic, additional)
SEI messages
Encode groups
(Figure 2)
Parameter set
View parameters list
Atlas data
Format Bitstream
(V3C sample stream with MIV extensions)
Geometry video data
(raw)
Attribute video data
(raw)
Occupancy video data
(raw)
Encode video sub bitstreams
(HM / VTM)
Packing
(VTM merge tool)
Multiplex
Bitstream
(one file)
Color code
Prepare source material
Per-group encoding
Bitstream formatting
Video encoding

Koder MIV

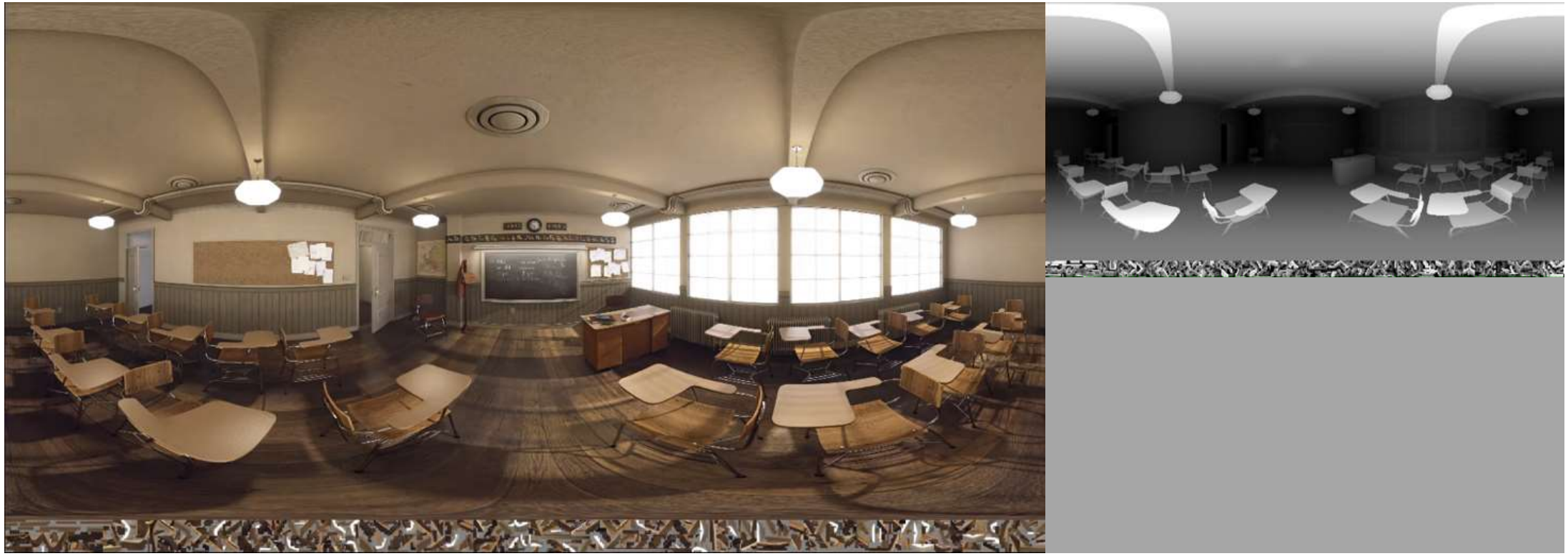

Source views
View parameters
Geometry component
Attribute components
Entity map (opt.)
Split source in groups
Label views
(basic, additional)
SEI messages
Encode groups
(Figure 2)
Parameter set
View parameters list
Atlas data
Format Bitstream
(V3C sample stream with MIV extensions)
Geometry video data
(raw)
Attribute video data
(raw)
Occupancy video data
(raw)
Encode video sub bitstreams
(HM / VTM)
Packing
(VTM merge tool)
Multiplex
Bitstream
(one file)
Color code
Prepare source material
Per-group encoding
Bitstream formatting
Video encoding

Video sub-bitstream packing

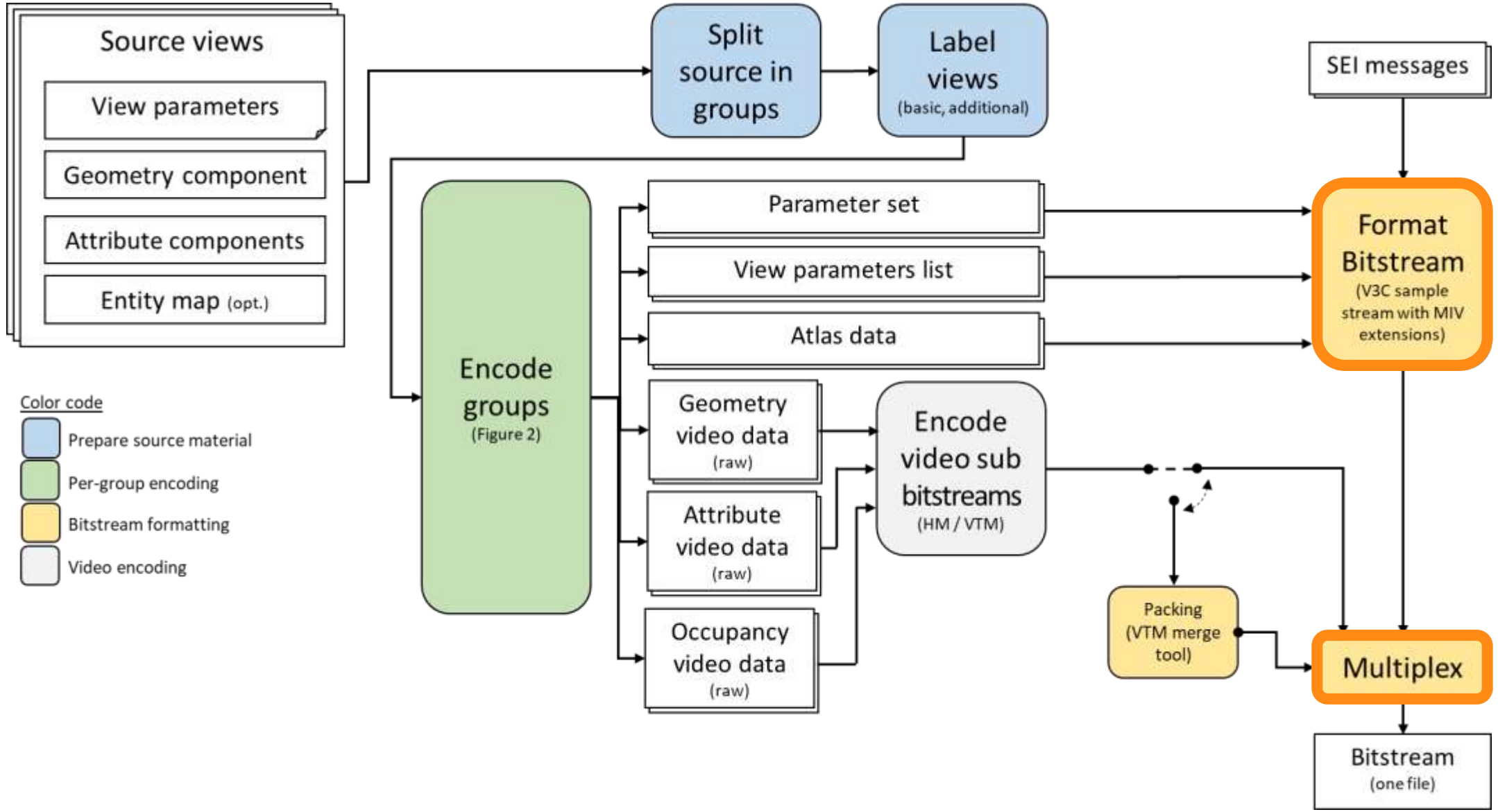




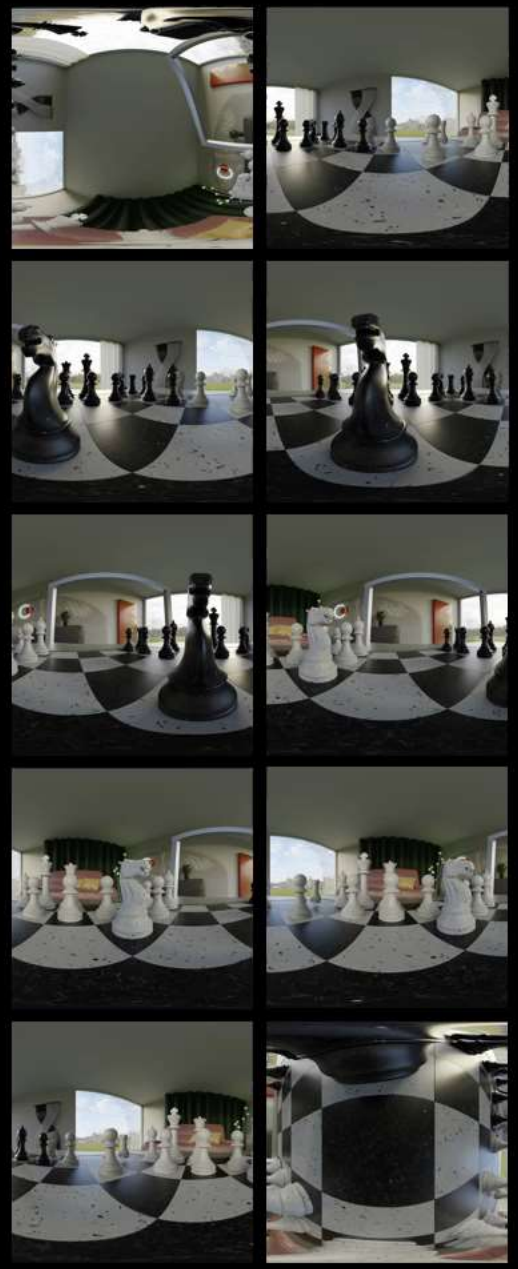
Video sub-bitstream packing



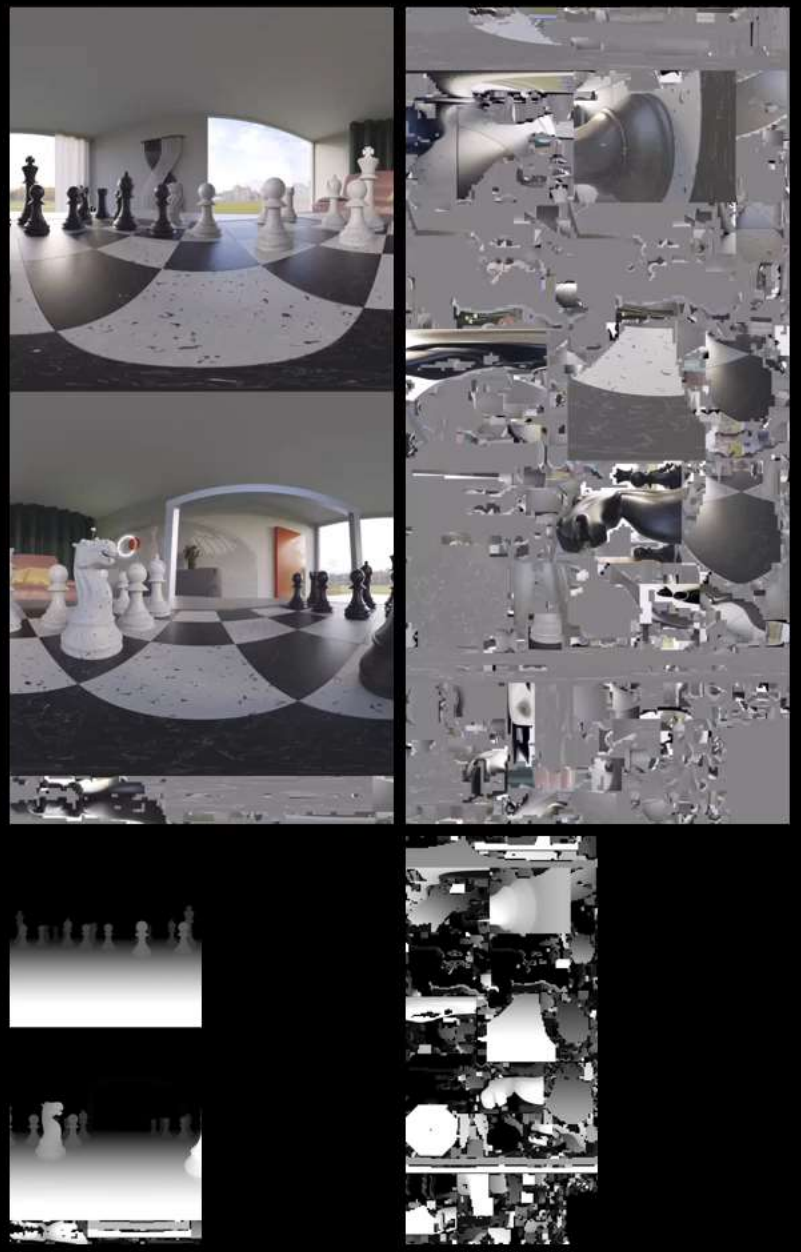
MIV – bitstream generation and multiplexing



input views
attributes + geometry + cam. params



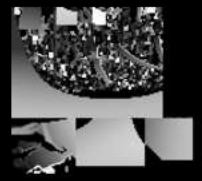
atlases
attributes + geometry + metadata sub-bitstream



viewport

input views
attributes + geometry + cam. params

atlases
attributes + geometry + metadata sub-bitstream



viewport

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/DPMMzj2I6Yw
  - Real-time RVS: https://lisaserver.ulb.ac.be/rvs/
- Test material (14+ sequences) available on request

Q&A

End of tutorial