How to achieve 6DoF compression?

Joel Jung

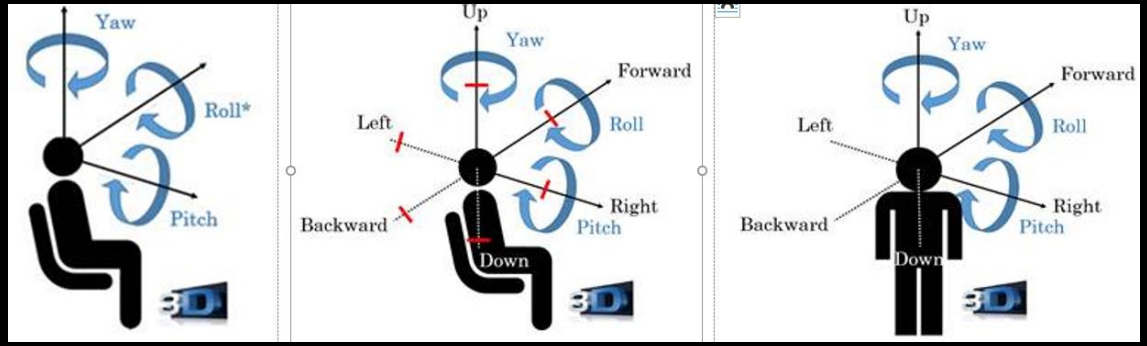
Orange Labs

Workshop on standard coding technologies for immersive visual experiences

July 10th, 2019, Gothenburg, Sweden

What is 6DoF?

6DoF: ability to move in the 6 directions (3 rotations + 3 translations)



3-DoF 3-DoF+ 6-DoF

Vocabulary: 6DoF cannot be compressed…

How to achieve 6DoF compression?

Proposition to rephrase the question:

How to design compression to achieve 6DoF immersion?

to render the perfect pixel according to any motion

What is the light-field?

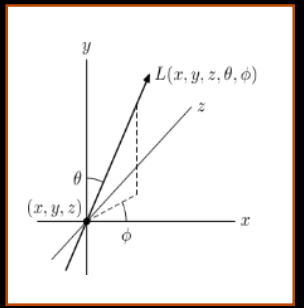
Definition: function describing the amount of

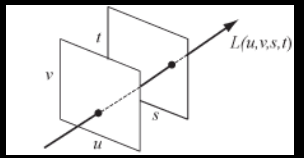
“light flowing in every direction through each point of space”

Represented by a 7 parameters function:

- Position in space (x, y, z)
- Viewing direction $(\theta, \emptyset)$
- Light intensity (t, λ)

5 parameters
(plenoptic function)





4 parameters
(pairs on 2 planes)

Assuming the light-field is available: immersive video becomes a compression problem:

- All pixels are available, for any point of view, from any point of view.
- No more motion sickness: no shift between what is displayed and the expectation of the brain.

How to design compression to achieve 6DoF immersion?

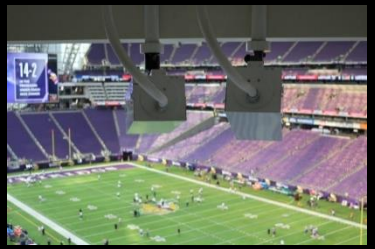
Proposition to rephrase the question:

How to design light-field compression to achieve 6DoF immersion?

Can we capture the light-field?

Plenoptic camera

Omni-directional cameras
(divergent)



Camera arrays
(convergent)

Cameras: capture sub-sampled light-field.

Lenslet, point-clouds, meshes, multi-view, 360, 2D: different discrete representation formats of the sub-sampled light-field

(with different advantages and drawbacks for immersive video)

The (full) light-field cannot be captured

What is the real problem?

Goal : render a dense light-field (from the transmitted light-field)
under constraint of:

- Sparse capture
- Constrained bit-rate

- Not only about compression of 2D images
- Rather about what to compress and transmit to enable correct synthesis
  (pixels, but not only)

How to design light-field compression to achieve 6DoF immersion?

Proposition to rephrase the question:

How to design light-field compression and rendering to achieve 6DoF immersion?

Outline

Introduction

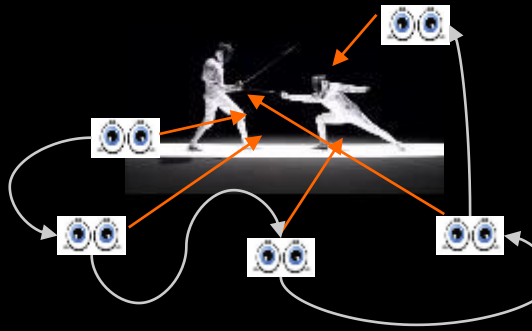
Target and current status of 6DoF immersive video in MPEG-I Visual

Challenges and current bottlenecks of 6DoF immersive video in MPEG-I Visual

Insight on how to achieve light-field compression and synthesis for immersive video

Conclusion

6DoF activity: goal



TMIV under construction

related to 3DoF+ 3DoF+ is 6DoF in a restricted volume

- Goal of 6DoF activity:
  - Extend the bounding box (most likely not unlimited)
  - Address natural and CG content
  - Consider video sequences, full scenes (not only objects). Ex: navigation in a sport event
  - Increase QoE (rendering quality at a given bit-rate)
  - Reduce pixel-rate
  - Reduce bit-rate

6DoF activity is as a v2 of the 3DoF+ activity

MPEG-I Visual current status

MPEG-I Visual inherits from FTV ad-hoc group

Former FTV group:
- set ground basis for immersive activities
- designed useful reference software (DERS, VSRS)

New actors:
- Pushed the activity one step further
- More practical approach to the problems
- Faster pace

Compression but not only:
- + depth estimation (not part of the standard)
- + view synthesis (not part of the standard, but included in encoder (3D-HEVC, TMIV)

MPEG-I
Part 2: OMAF
Part 5: V-PCC
Part 7: IMM (MPEG-I Visual)
Part 9: G-PCC

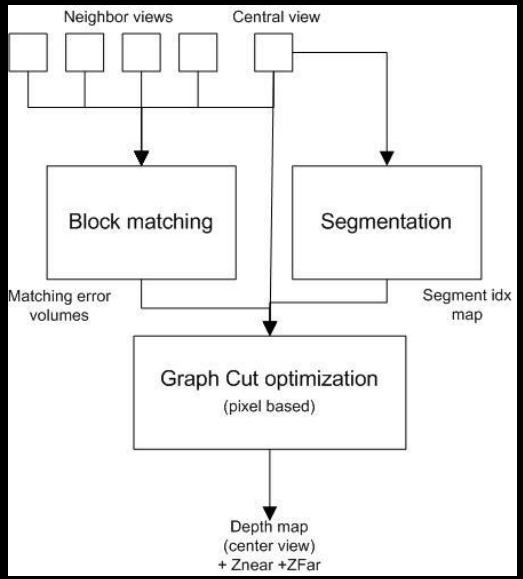
New players
2015: Univ. of Brussels, Univ. of Zhejiang, Orange
2016: Technicolor
2017: Philips
2018: Intel, Nokia

Recent boost of the activity

Current status: depth estimation

- Depth estimation: critical topic. Not sufficiently addressed so far (group busy with compression and synthesis)

- Current status: DERS8.0 (Depth Estimation Reference Software)
  - Efficient, but unstable
    - From one sequence to another
    - From one view to another



Principle:
- Up to four neighboring views
- Block matching + segmentation
- Graph cuts algo. to find correspondences between views on pixel-by-pixel basis
  - smoothing coefficient: smooth depth within a segment
  - basic temporal enhancement
- Outputs central depth map and zNear / zFar (to normalize the disparity)

- **Needs to be designed in a more practical way**
- **Robustness needs to be increased by following strict CTCs, and no sequence dependent tuning**

Current status: view synthesis (1/2)

- Most of the displayed views are synthesized.

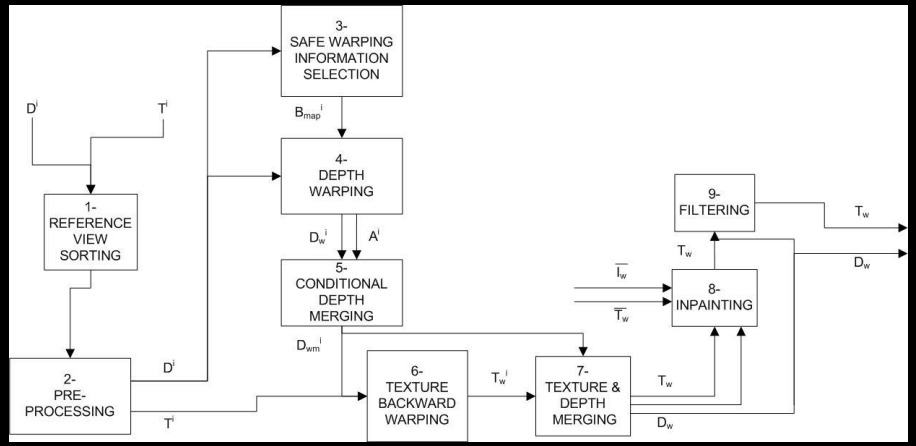
- Current status: 2 tools have replaced former VSRS (View Synthesis Reference Software)

- RVS (version 3.1) – Reference View Synthesizer
    - Developed by Université Libre de Bruxelles and Philips
    - Adopted for 3DoF+ activity in July 2018
    - Outperforms VSRS4.2 by 110% (1.1 dB) on CTCs.

Principle:

- Original, significantly differs from VSRS
- Reference views warping using a computed disparity
- References partitioning in triangles that are warped using computed translation and rotation, and filled with tri-linear interpolation.
- Views synthesized from each references are blended
- Inpainting applied on the blended view, to fill the dis-occlusions.

Current status: view synthesis (2/2)



- VVS (version 2.1) – Versatile View Synthesizer
  - Developed by Orange
  - Adopted for 6DoF activity in October 2018
  - Outperforms VSRS4.2 by 163% (1.6 dB) on CTCs.

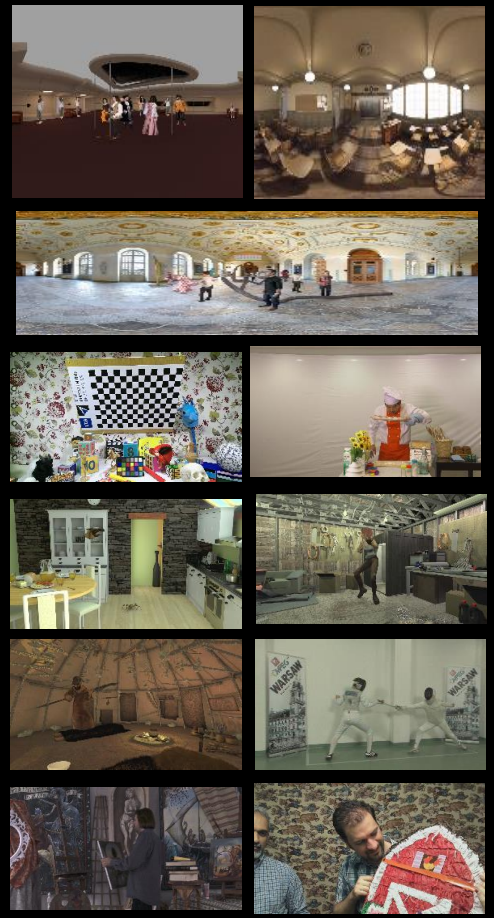
Principle:

- Designed to deal with coded contents
- Original steps of
  - Reference view selection (not based on camera positions but on warping quality)
  - Safe warping information selection (borders of objects)
  - Conditional depth merging
  - Spatio-temporal inpainting
  - Edge filtering (remove cartoon effect)

Major progress achieved by RVS and VVS on view synthesis over the last 18 months

Current status: common test conditions

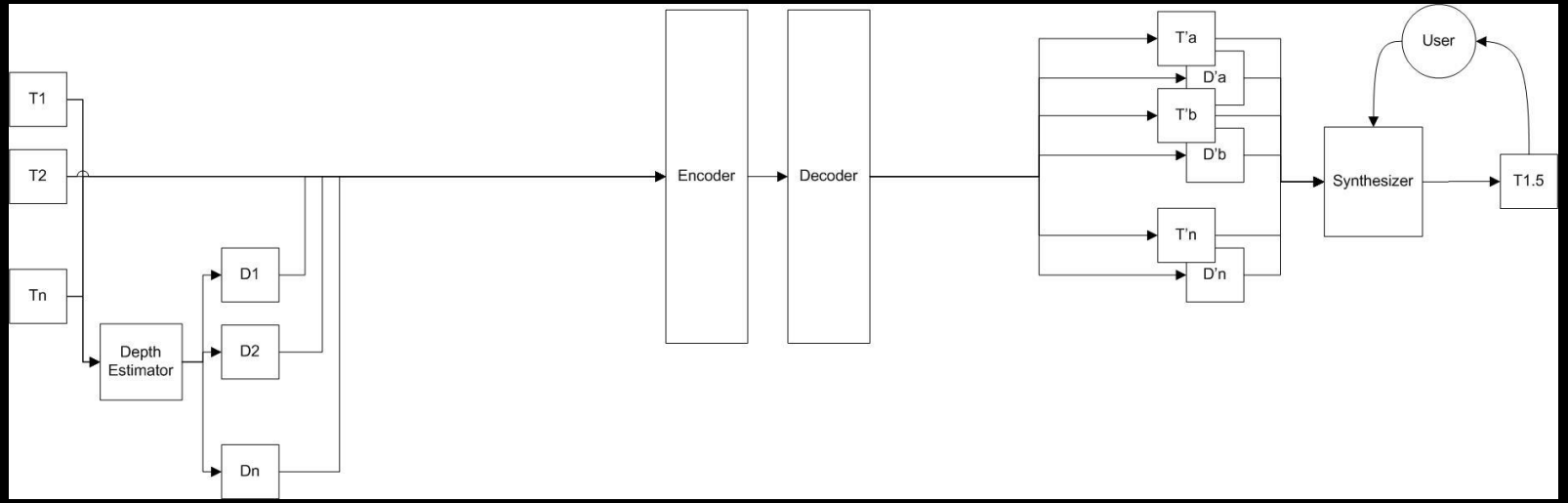
- Former 3DoF+ and 6DoF CTCs (nearly) aligned
- Total test material: 12 various sequences (not enough, though…)
  - 6 CG, 6 natural
  - 3 omni-directional, 9 perspective
  - 2 arc configuration, 3 random configuration, 7 planar configuration
  - 1 1D-array, 8 2D-arrays, 3 random arrays
- Ways of evaluating the new tools or frameworks:
  - Objectively: PSNR, MS-SSIM, VIF, VMAF
  - Subjectively: navigation paths/pose traces
- 2 anchors for the 6DoF framework: TMIV and MV-HEVC based



Clear common conditions available to compare new methods/tools/frameworks

Current status: compression, MV-HEVC anchor (using VVS)

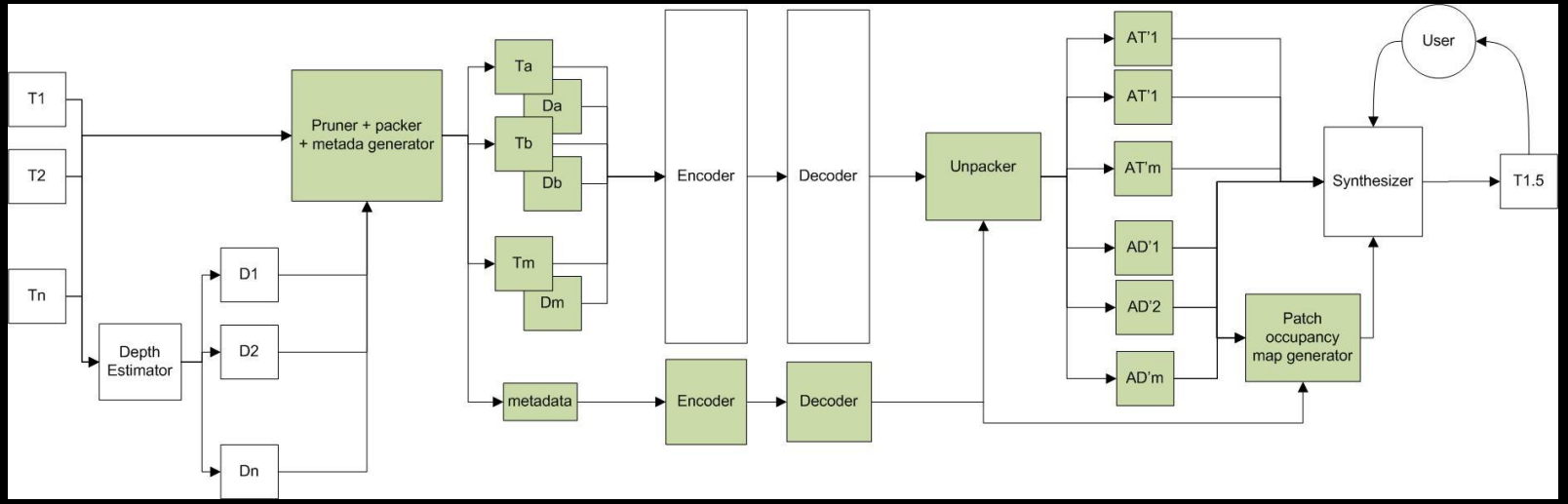
MV-HEVC



- Simple approach (HEVC + high level syntax, no pre- post-processing)
- Not specifically designed for 6DoF activity (existing standard)
- High pixel rate (all views transmitted)

Current status: compression, TMIV anchor (using RVS)

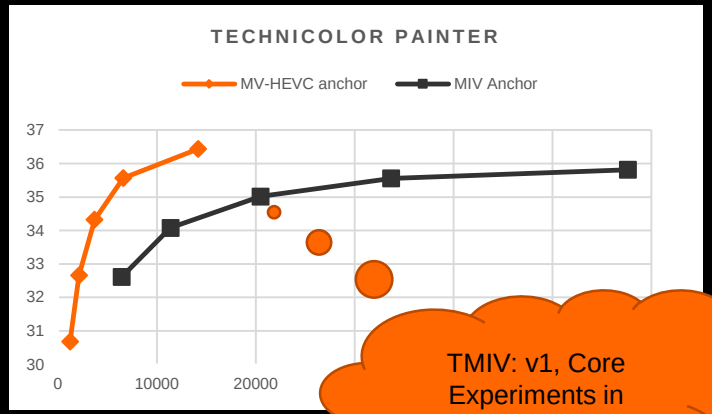
TMIV



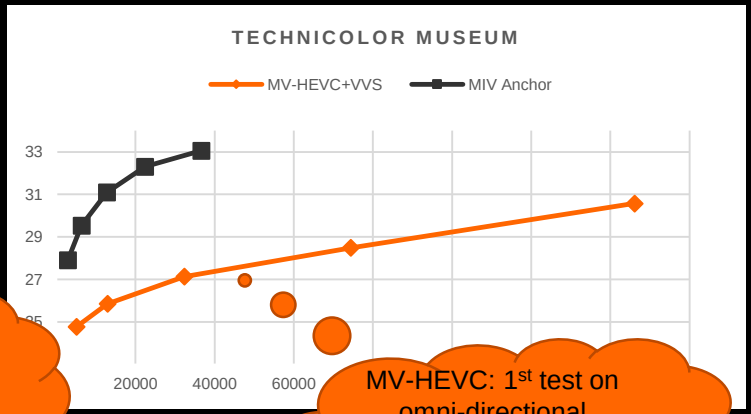
- TMIV proves that sending full views (texture + depths) is not needed
- TMIV proves that metadata created by the encoder can help synthesis
- TMIV is a first example of codec "compatible" and "friendly" with synthesis

Current status: comparison of TMIV and MV-HEVC anchors

Preliminary tests
Snapshot of current status
Results evolving quickly



TMIV: v1, Core Experiments in progress



MV-HEVC: 1st test on omni-directional content deltaQP set randomly…

- TMIV already efficient for what it is designed for (3DoF+)
- Interesting starting base for 6DoF activity

- MV-HEVC not designed for omni-directional content
- Improvements are easy: 20% improvement on perspective content by just changing the QPs [m49149]

Both anchors can be improved very quickly and significantly

Outline

Introduction

Target and current status of 6DoF immersive video in MPEG-I Visual

Challenges and current bottlenecks of 6DoF immersive video in MPEG-I Visual

Insight on how to achieve light-field compression and synthesis for immersive video

Conclusion

Technical challenges

- Dense light-field to be recovered from sparse light-field: capture / synthesis challenge

  Challenging because → Not only compression of pixels needs to be considered
  Depends partly on the progress of capturing devices

- Current quality of depth maps is too low: depth estimation challenge

  Challenging because → Depth map estimator needs to be designed with the framework (incl. the synthesis) in mind. No search for ground truth depth maps.

- Current quality of displayed views (synthesized) is too low: synthesis challenge

  Challenging because → Synthesis quality relies on depth maps quality
  Different synthesizer can be used (RD choices at the encoder side, rendering side)

- A lot of pixels needs to be sent: compression challenge

  Challenging because → A pruning of the data to transmit is needed
  Additional data (not only pixels) can be sent

Non-technical challenges

- Covers several research areas
- Several different representation formats exist with pros and cons
- Capture and display devices are not mature enough, content is missing
- Different approaches are possible, and starting from a 2D codec and adding inter-view prediction is not sufficient

Challenge to work on a topic that is not perfectly delimited
With an outcome/result not perfectly known in advance

Outline

Introduction

Target and current status of 6DoF immersive video in MPEG-I Visual

Challenges and current bottlenecks of 6DoF immersive video in MPEG-I Visual

Insight on how to achieve light-field compression and synthesis for immersive video

Conclusion

For non-technical challenges

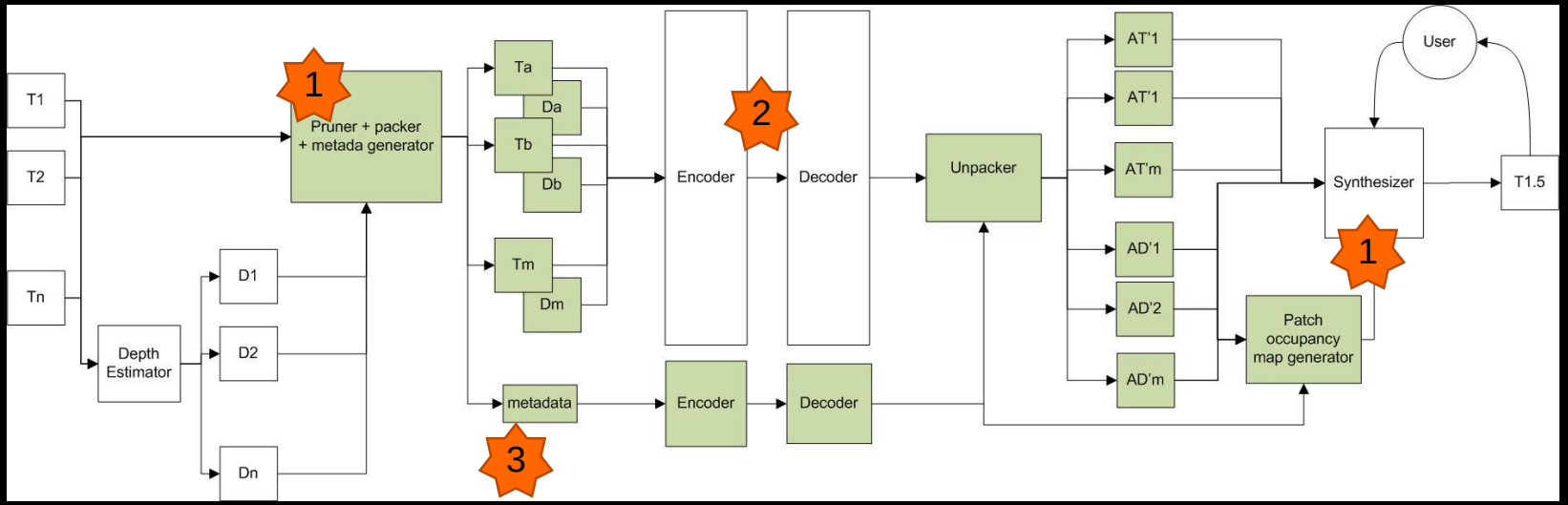
- Have some guess and do some bets
- Develop key generic technology to be used in multiple different scenarios / use cases (business driven)

For technical challenges

Driving idea: Codec design: not only “compatible”, but also “friendly” with the synthesis

By sharing metadata, TMIV is a good first example

Possible improvements of current anchors



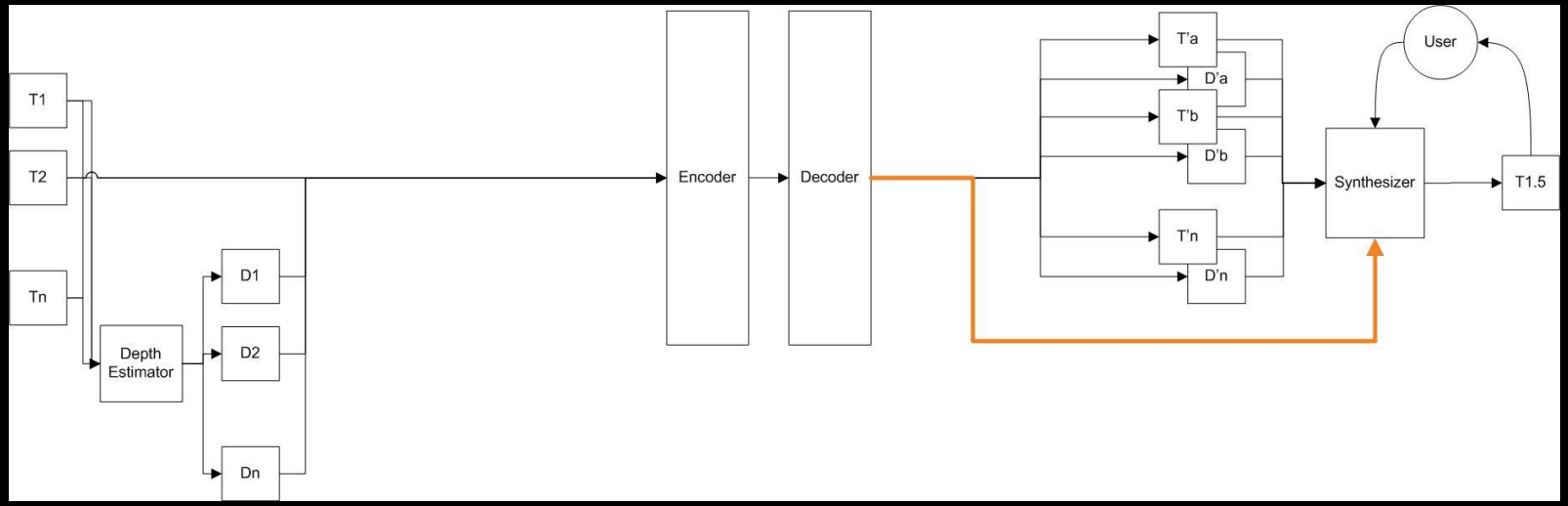
1 1- Improve synthesis and pruning steps (tightly linked)?

2 2- Improve the codec (of views or atlases)?

- Add some inter-view prediction and other selected 3D-HEVC coding tools?
- Improved RD decisions (include more synthesis)?

3 3- Add new metadata to better drive the synthesis?

Linking decoder and synthesis?



Decoder knows / can share important characteristics of the scene:

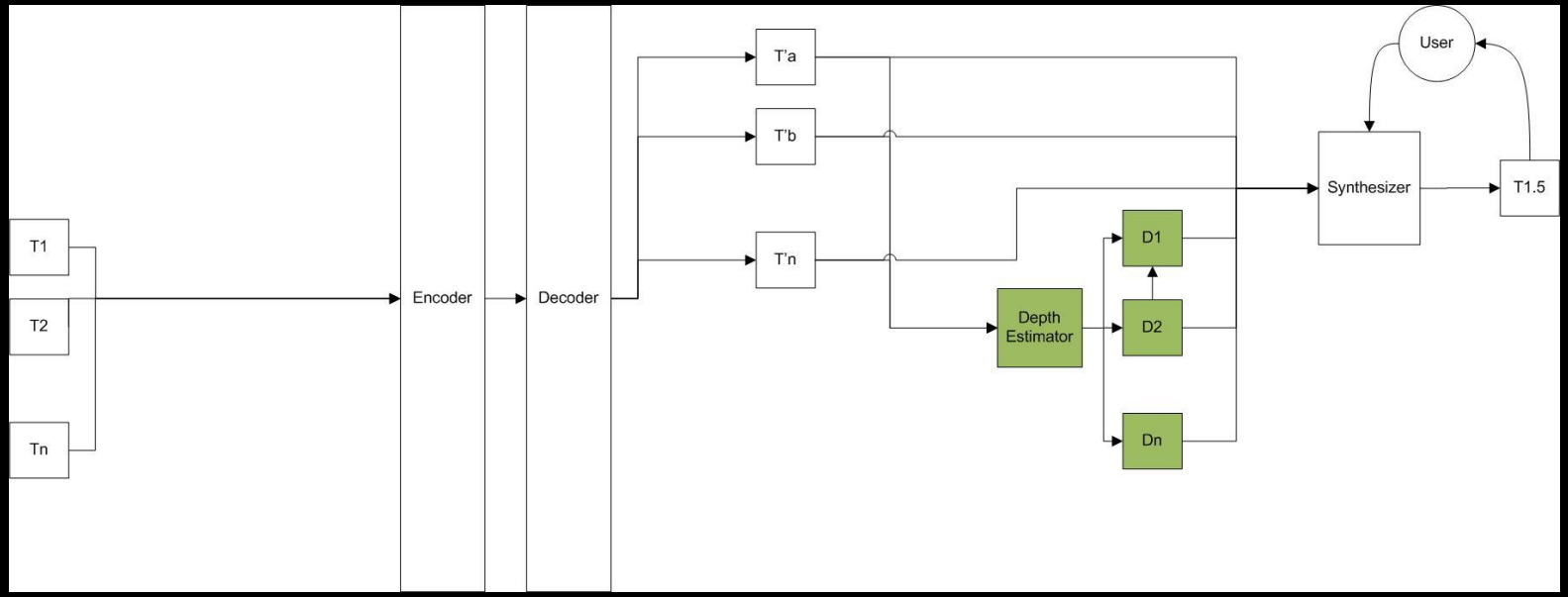
Merged areas Motion vectors Coding modes Intra directions

Decoder can compute and share new information from available ones:

Histograms Global motion and directions Statistics Level/areas of inter-view prediction (dealing with occlusions, etc.)

Decoder can help and simplify the synthesis

Moving the depth estimation after the decoder? (1/2)



Decoder Side Depth Esimation (DSDE) allows:

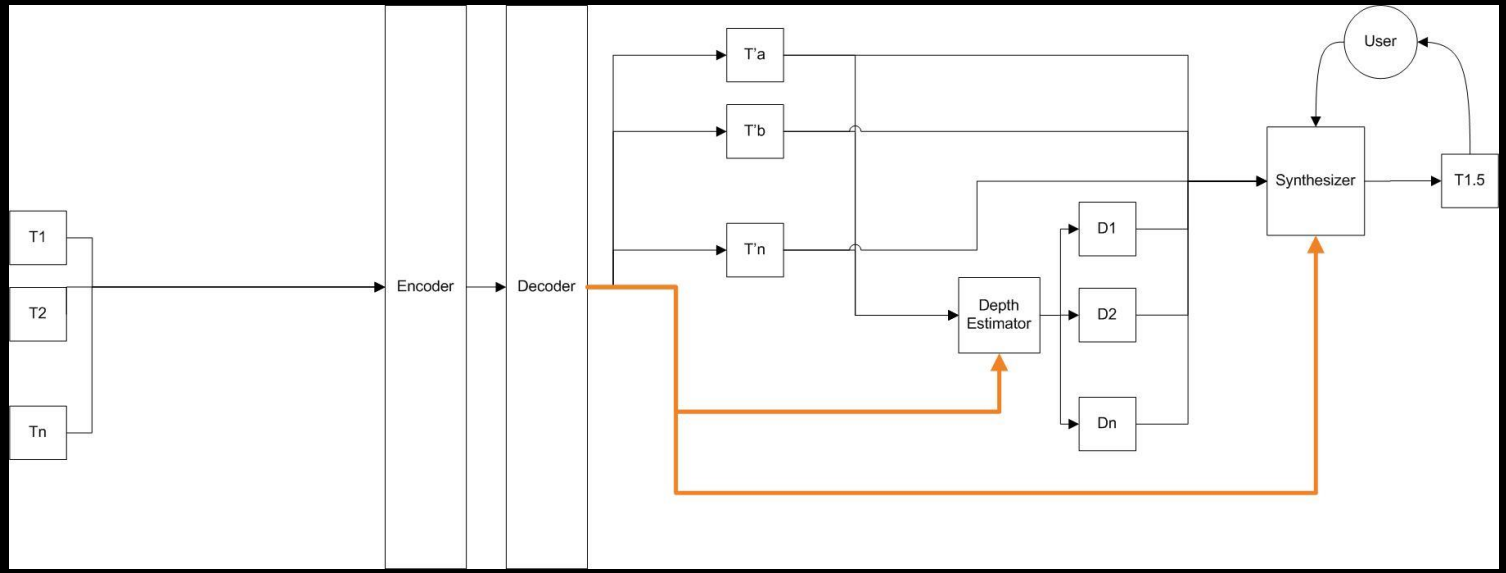
Pixel rate divided by 2

35% BD-rate improvement for MV-HEVC anchor

Check m49153 for 1st results

No more tuning of the delta QP between texture and depth

Moving the depth estimation after the decoder? (2/2)

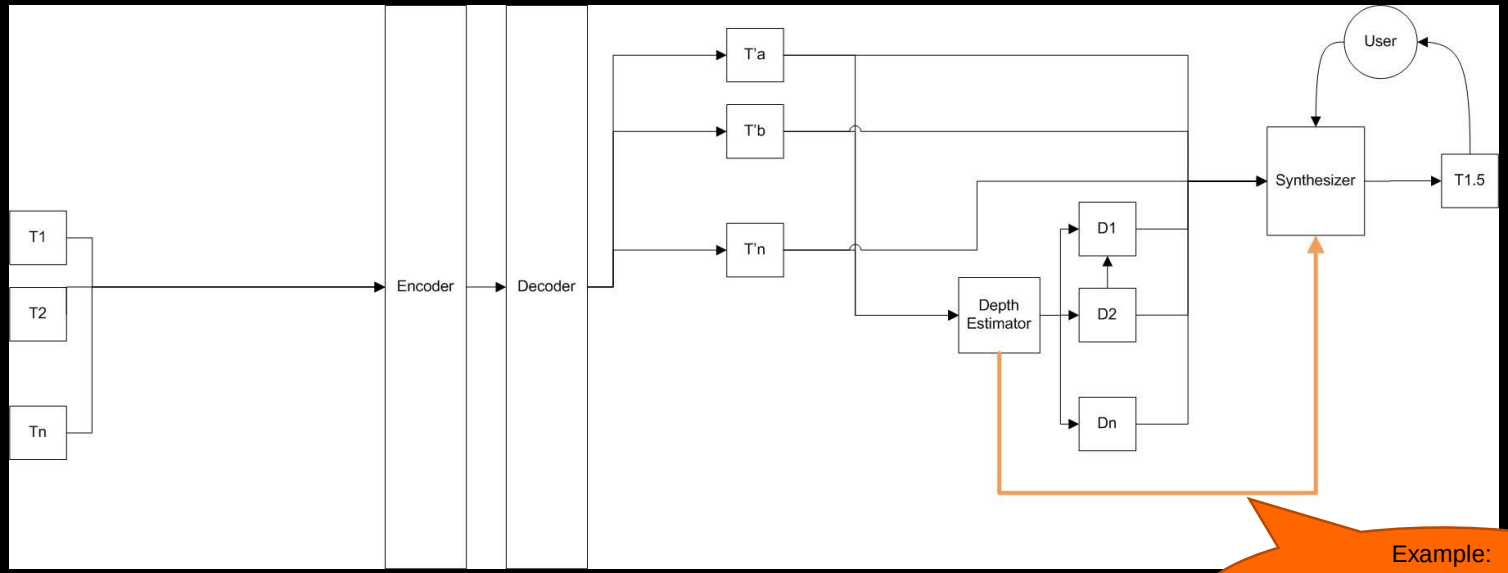


Decoder can still discuss with synthesis algorithm

DSDE: Decoder can discuss with depth estimator too

Decoder can help depth estimation process

Linking depth estimation and synthesis?



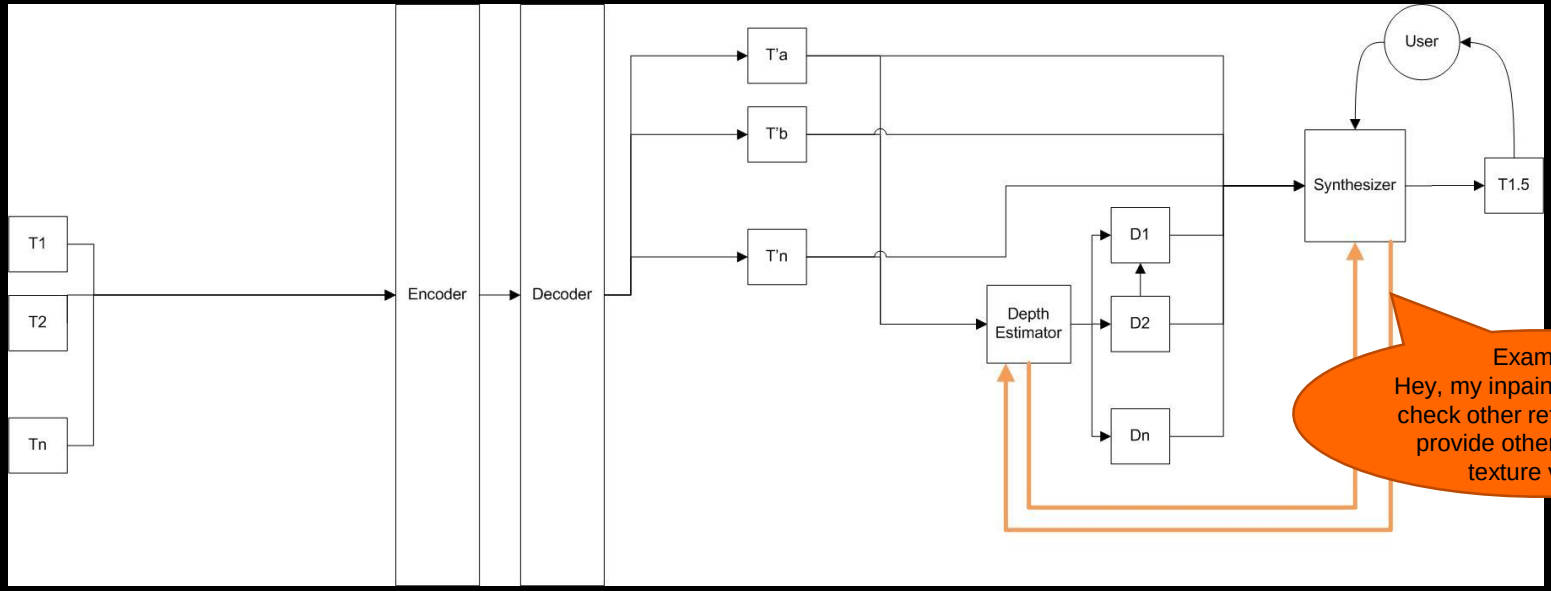
Example:
Hey, for this area, I'm really not sure about my depth estimation. Take it into account for your synthesis

Decoder can still discuss with synthesis algorithm

DSDE: Decoder can discuss with depth estimator too
Depth estimator can share information with the synthesizer

Confidence map
No information to code/transmit

Linking depth estimation and synthesis?



Decoder can still discuss with synthesis algorithm

DSDE: Decoder can discuss with depth estimator too
Depth estimator can share information with the synthesizer
Synthesis can make requests to the depth estimator

Is a full depth estimation needed?

Solves complexity issues of DSDE…

Normative vs. Non-normative?

Most of those interactions / discussions can be simulated at the encoder:

Encoder can mimic the depth estimation process and the synthesis to make the correct decisions on what to send and how to encode
(not only pixels)

What should be standardized in such a framework – with decoder / synthesis / depth estimation sharing information?

May be a bit more than a "classical" pixel decoder and meta-data?

Sensitive topic…
Think about the loop filter story in MPEG-4 AVC…

Other ideas?

Those were just example of possible ideas/frameworks
(applicable to TMIV and/or MV-HEVC anchors)

Several other options…

- Take the best of the two anchors
- Apply a different frameworks…
- Consider recent progress of CNN-based approaches?

  Recent promising results on depth estimation and synthesis

  Yet on limited/simplified test conditions (no robustness)

- Etc.

Outline

Introduction

Target and current status of 6DoF immersive video in MPEG-I Visual

Challenges and current bottlenecks of 6DoF immersive video in MPEG-I Visual

Insight on how to achieve light-field compression and synthesis for immersive video

Conclusion

Conclusion (1/2)

Goal: allow 6DoF immersion

Means rendering the correct point according to the exact motion

Under constraint of sparse capture of the light-field and reasonable bit-rate

"How to design light-field compression and rendering to achieve 6DoF immersion?"

Rendering:

- Not only about coding 2D images (decoded views are not displayed)
- About how to make the compression compatible but also "friendly" with the synthesis.

6DoF activity: v2 of 3DoF+ activity (with increased QoE / reduced bit-rate and pixel-rate)

Recent boost of the activity

- CTC defined to assess possible improvements
- Depth estimation "ready to progress"
- Synthesis already made big steps forward
- Compression + synthesis: 2 anchors are defined, TMIV / MV-HEVC

Conclusion (2/2)

Several challenges:

- Not only technical ones.
  - Accept to work on something not completely delimited
    - with target (gain) not 100% predictable

- Many possible approaches to improve the 2 anchors.
  - May be mix of them for 6DoF?
  - TMIV under CEs
  - MV-HEVC anchor easy to improve

- Other ways /frameworks are possible (CNN, etc)

What to standardize? May be not just a classical decoder

- Links can be created between the 3 major component of the immersive framework (compression depth estimation, synthesis)

Many interesting technical perspectives
With huge amount of possible improvements

A lot of work here for a lot of video coding experts

We are at a very preliminary stage of immersive video coding…
(equivalent of MPEG-1 for 2D?)

Thank you for your attention