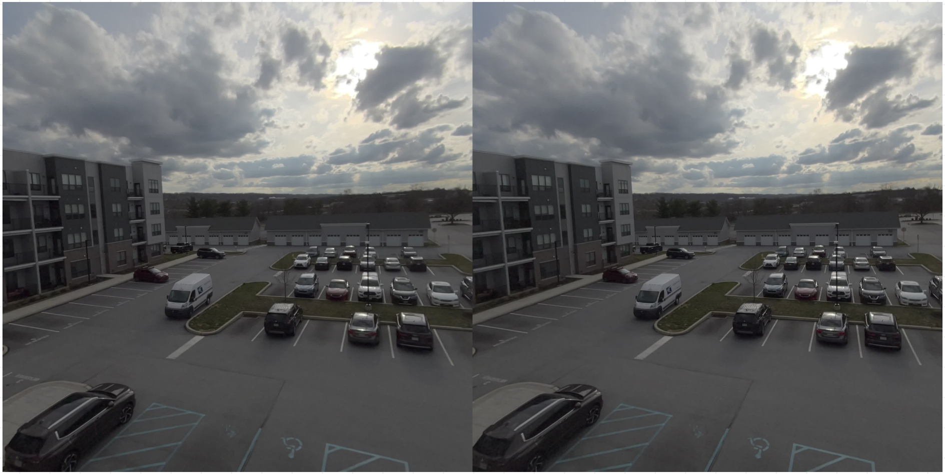
Multiview decoding in LIBAVCODEC and FFMPEG CLI

Anton Khirnov

FFlabs

2025-02-02
FOSDEM

What is multiview video



What is MV-HEVC

- a descendant of MVC (multiview AVC/H.264)
- a way of packing multiple semi-independent substreams within a single HEVC bitstream
- based on multi-layer extensions
- besides multiview can be used for scalable encoding, alpha, …
- up to 63 layers, a non-base layer may depend on (i.e. predict from) other layers
- complex dependency graphs possible
- we only support 2 layers, second depending on the first

Why might you care about MV-HEVC

- stereoscopic 3D
- other multi-layer features — e.g. alpha
- multiview challenges existing assumptions, requires updating some plumbing
    - multiple output frames for a single input packet
    - multiple output streams from a single decoder

implementation challenges — HEVC decoder

- a lot of state that used to be decoder-global is now per-layer
- common approach:

```
typedef struct FooContext {
    int some_state;
    …
} FooContext;
```

⇒

```
typedef struct FooContext {
    int some_state;
    struct FooContext *children;
    int              nb_children;
    …
} FooContext;
```

- obfuscates code
- unclear which fields are meaningful in the parent and which in the children
- only saves a tiny bit of work
- EVIL

implementation challenges — HEVC decoder II

- identify decoder-global vs. per-layer state
- add per-layer context
- move per-layer state to per-layer context
- update frame output logic
    - HEVC has frame reordering
    - need to interleave frames from the decoded layers
    - ability to output multiple frames at once makes it a lot simpler
- all the views/layers are in one packet - frame threading not very efficient for multiview

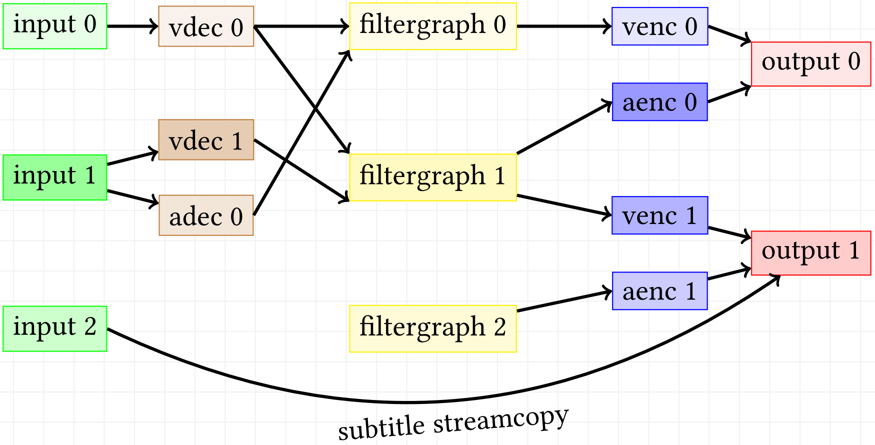
implementation challenges — AVCODEC generic code

- need multiple output frames per input packet
- decoders implement one of two APIs — older “simple” API, and newer `receive_frame`
- `receive_frame` supports arbitrary M:N packets $\longrightarrow$ frames mappings
- frame threading only supported the “simple” API
- there was a WIP patch from 2017 switching frame threading to `receive_frame`
- FFV1 decoder abused frame threading API
  - `refstruct` is great

implementation challenges — public API

- all multilayer properties are (in principle) per-sequence
- need to communicate them to the caller
    - view IDs
    - view positions
- need to allow the caller to select views to output — done via the `get_format()` callback
- array-type `AVOptions`
- frames produced by decoder have side data that indicates the view they belong to
- output layer sets — no semantics, do not seem to be useful

FFMPEG CLI general transcoding pipeline



2022-2024 multithreading project goals

- bring code structure in alignment with actual data flow
- separate every major component into its own standalone object with a clearly defined public interface
- every major component in its own thread
- advantages:
  - easier to reason about, understand, and maintain
  - more flexible structure opens the way to new features
  - improved throughput and CPU utilization

2022-2024 multithreading project status

- bulk of the work merged for 7.0
- some fallout, (hopefully) not too much
- standalone decoders
- filtergraph chaining

multiview support in the FFMPEG CLI

- native multiview support
- use view specifiers (i.e. with `-map` or complex filtergraph link labels) to select views by ID, index, or position
  - e.g. `-map 0:v:0:vpos:right`
- decoder objects in the CLI can now have multiple output streams
- 1 specifier = 1 output
- could be generalized to support e.g. splitting out closed captions