A MULTI-GPU SCALABLE SDK FOR REAL-TIME STEREO STITCHING OF 360 VIDEO AND AUDIO

Thomas True

AGENDA

Stereo Panoramas

VRWorks 360 Video SDK Overview

Calibration

Video Stitching

Audio Processing

Multi-GPU Optimization

VRWorks SDK Usage Examples

Q&A

STEREO PANORAMAS

STEREO PANORAMAS

Stereo Video in VR

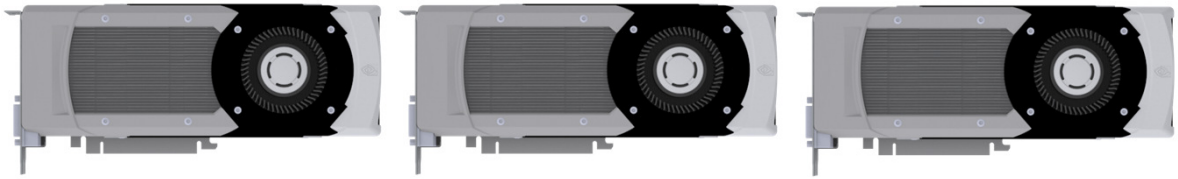
- 360 Video
- Immersive
- Provide sense of depth and scale
- Capture reality

STEREO PANORAMAS

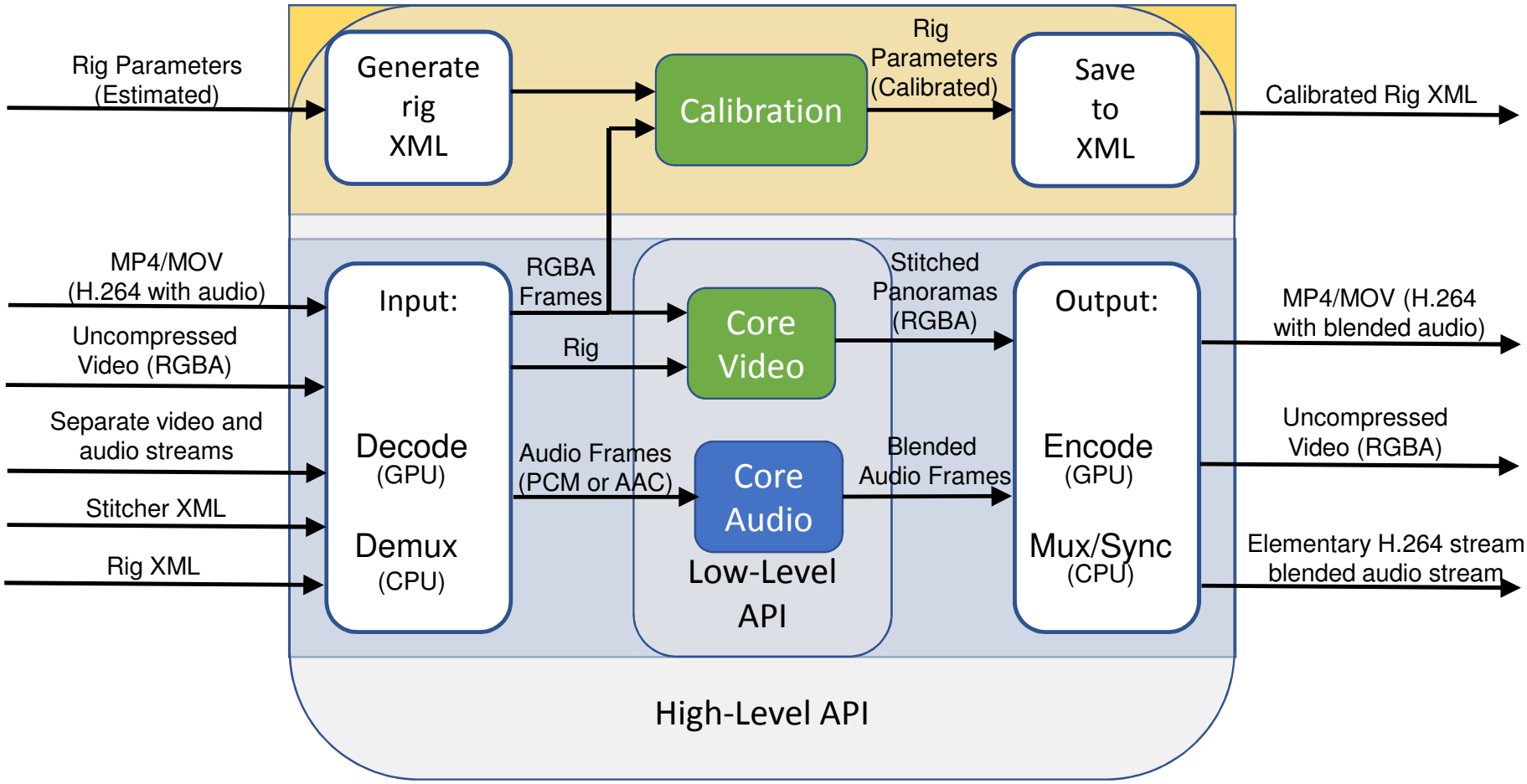
Workflow

VRWORKS 360 VIDEO

Ingest → Decode → Calibrate → Stitch → Encode → Output



360 STEREO STITCHING PIPELINE



VRWORKS 360 VIDEO SDK

VRWORKS 360 VIDEO SDK

Features

- GPU-optimized stereo pipeline
- GPU-accelerated calibration
- GPU-accelerated encode and decode
- Audio mix down and stretch
- Audio encode / decode
- Audio resampling
- Realtime and offline
- Stereo and mono stitching
- Compressed and uncompressed inputs and outputs
  - MP4, H.264 and RGBA8
  - PCM and AAC audio

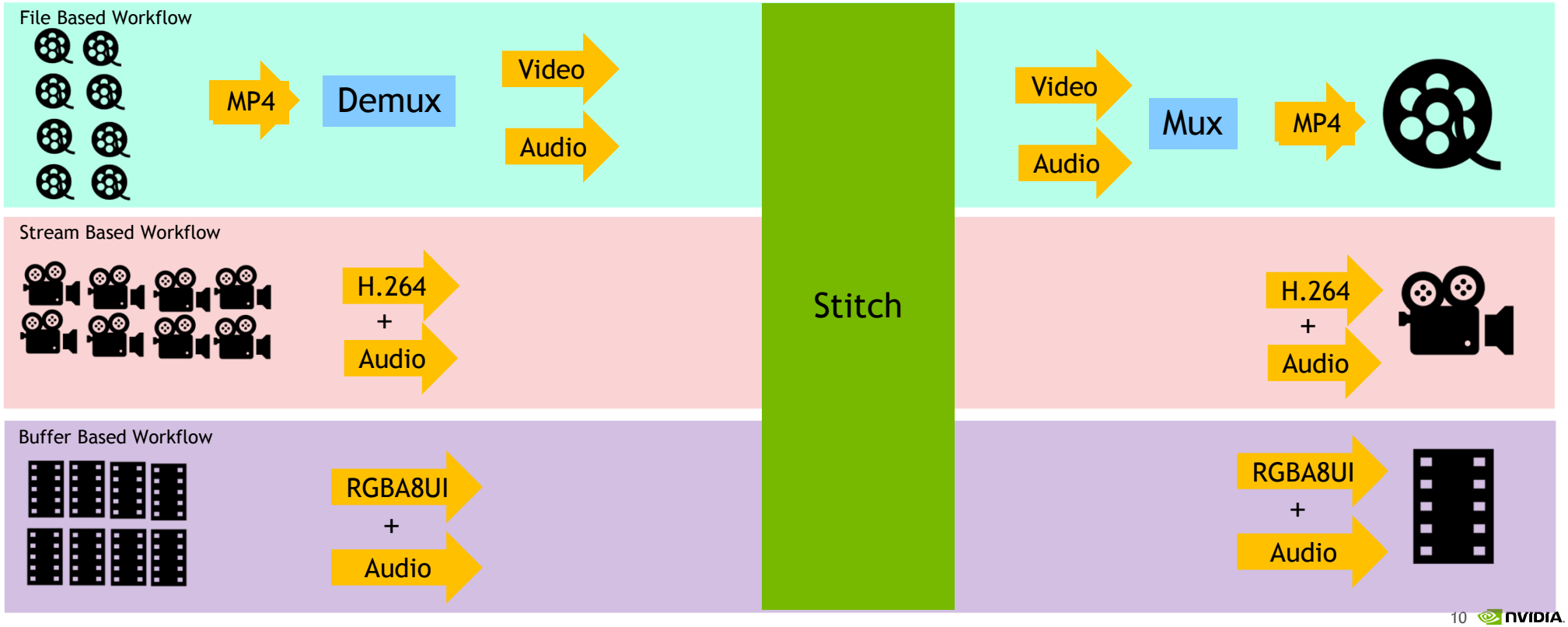
https://developer.nvidia.com/vrworks/vrworks-360video

VRWORKS 360 VIDEO SDK

Works With Multiple Rigs

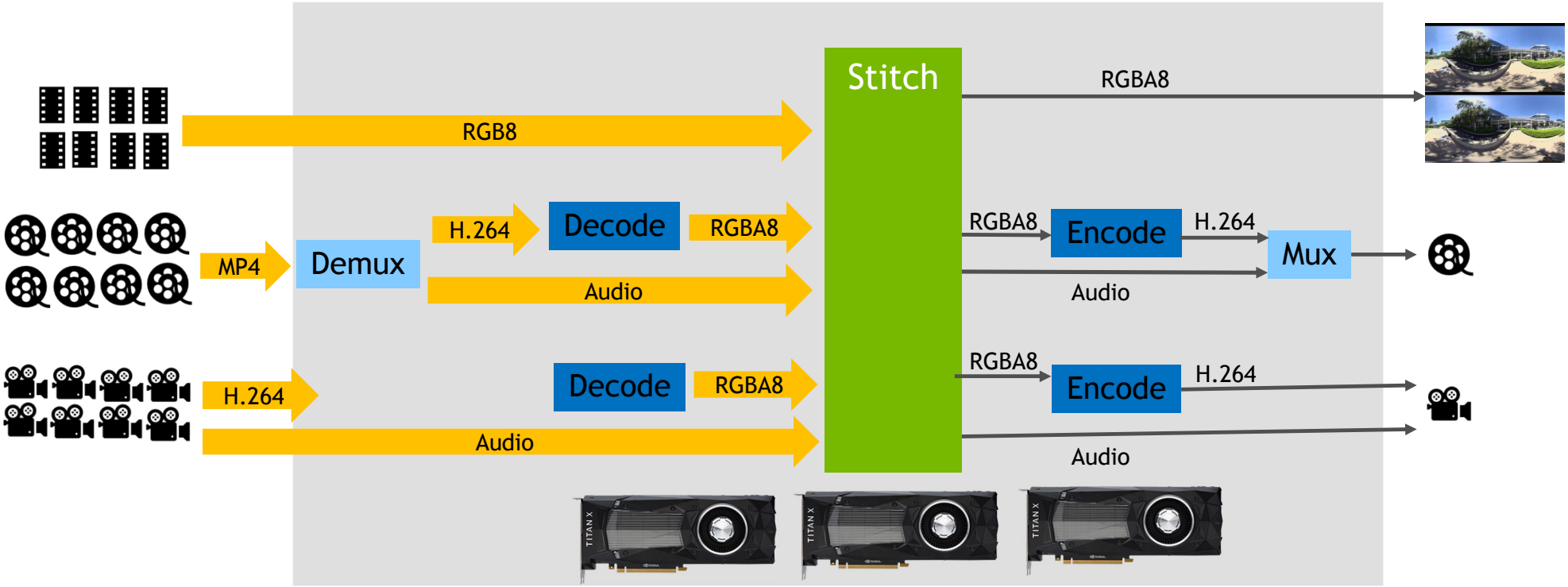
VRWORKS 360 VIDEO SDK

Workflow Flexibility



VRWORKS 360 VIDEO SDK

High-Level Video and Audio Pipeline



VRWORKS 360 VIDEO SDK

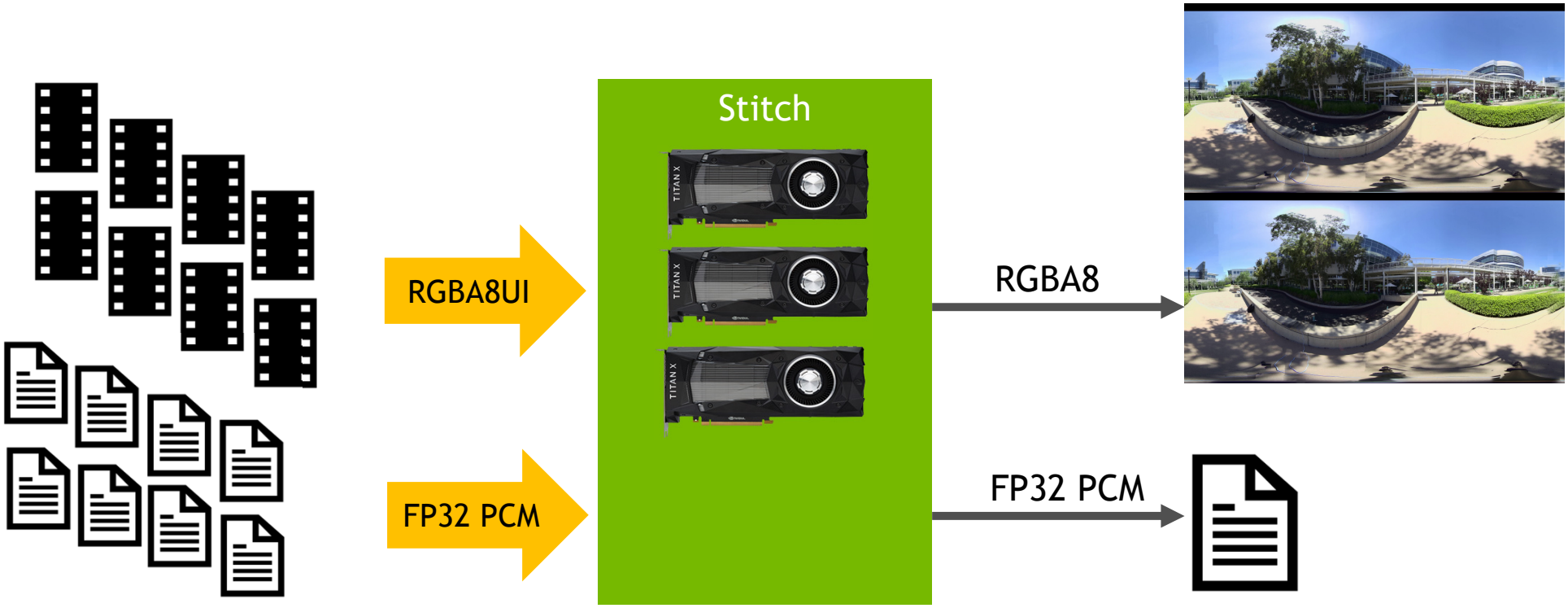
High-Level Video and Audio Pipeline

Advantages:

- GPU-accelerated demux, mux, decode and encode
- File and stream oriented API
- Includes calibration API
- Support for additional audio and video formats

VRWORKS 360 VIDEO SDK

Low-Level Video and Audio Pipeline



VRWORKS 360 VIDEO SDK

Low-Level Video and Audio Pipeline

Advantages:

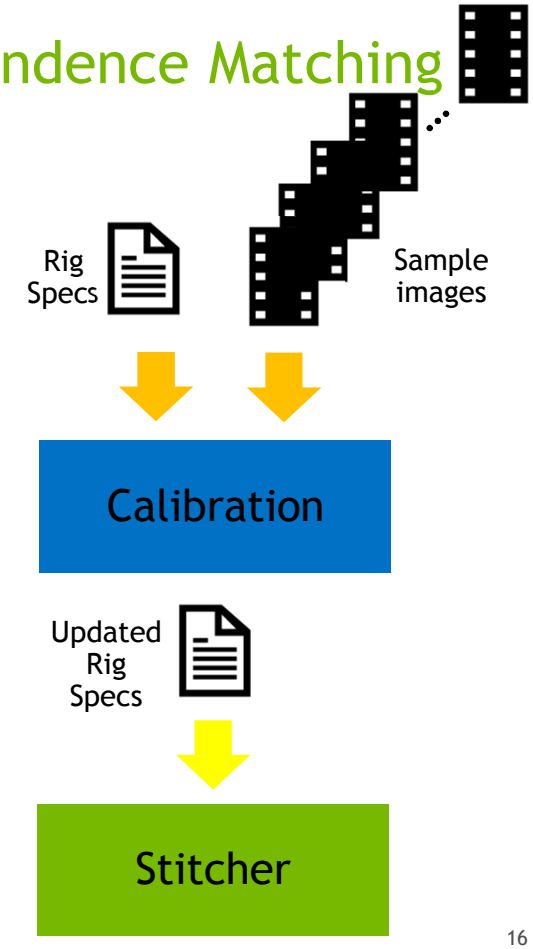
- Works directly with CUDA device memory
- Plugs directly into a larger CUDA pipeline
- Interops with graphics API to keep frames on GPU
- Application flexibility to control threading and data transfers

CALIBRATION

CALIBRATION

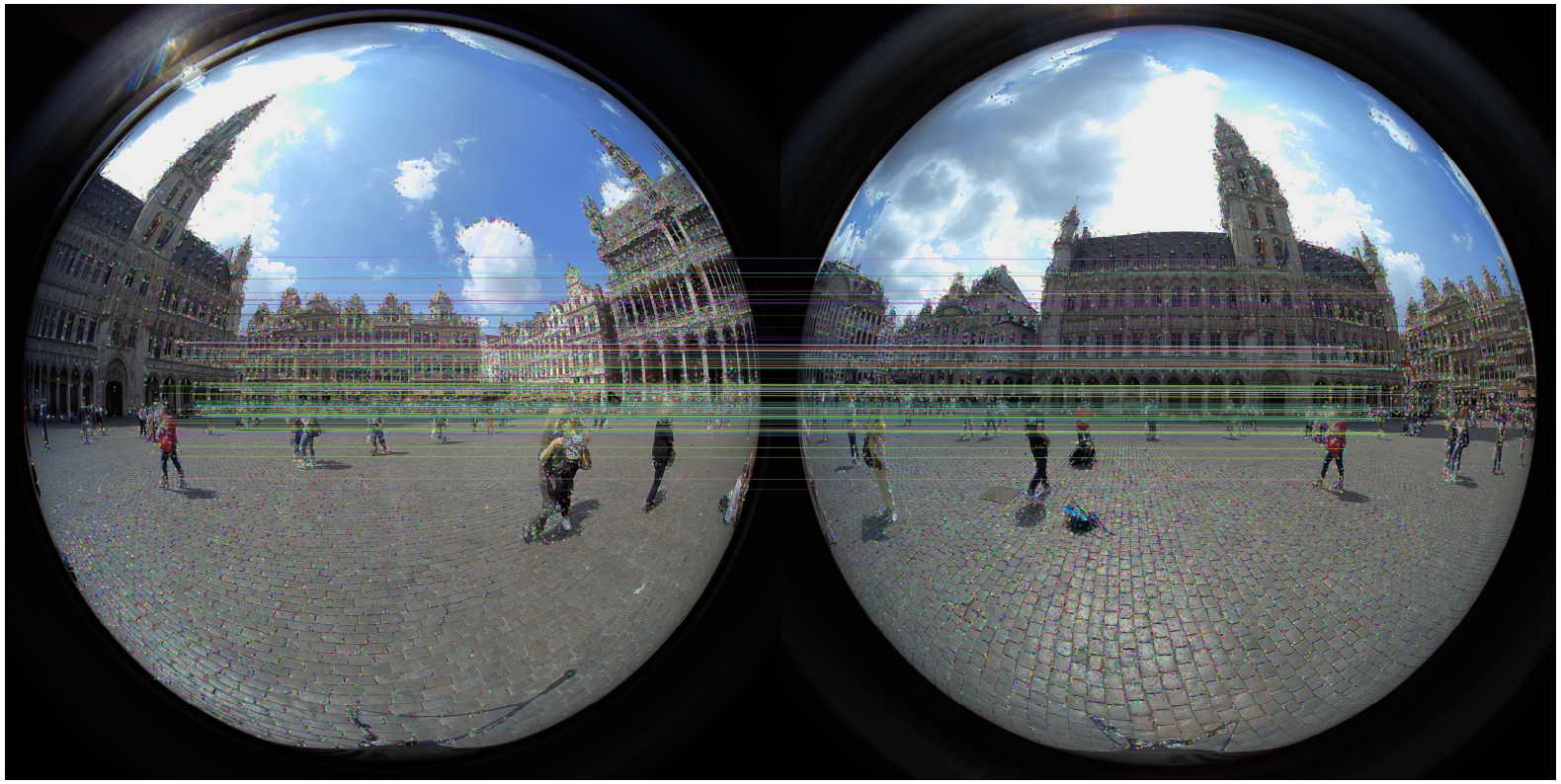
GPU-Based Feature Extraction and Correspondence Matching

1. Input initial rig specs and a set of images from the cameras
2. GPU-accelerated feature extraction
3. Construct feature correspondence pairs from overlapped regions
4. Optimize rig parameters to minimize the correspondence distances(on the virtual sphere)



CALIBRATION

GPU-Based Feature Extraction and Correspondence Matching



CALIBRATION

Features

- Works with any rig configuration
- High error tolerance in input instances.
- Accommodates lenses with up to or greater that 180° FOV
- Support standard distortion characteristics as well as custom fisheye lenses
- Provides accuracy metric based upon cross correlation [0:1]
- GPU-accelerated

VIDEO STITCHING

VIDEO STITCHING

Creating Virtual Cameras Between Physical Cameras

1. Compute pixel motion between adjacent cameras
   - Lambertian: Appearance not view dependent
   - Cameras photometrically consistent
   - Rig calibrated
   - Surface point same when viewed by adjacent cameras
   - Compute disparity
2. Interpolate to the virtual camera position, reprojection the virtual sphere and blend.
3. Apply post processing

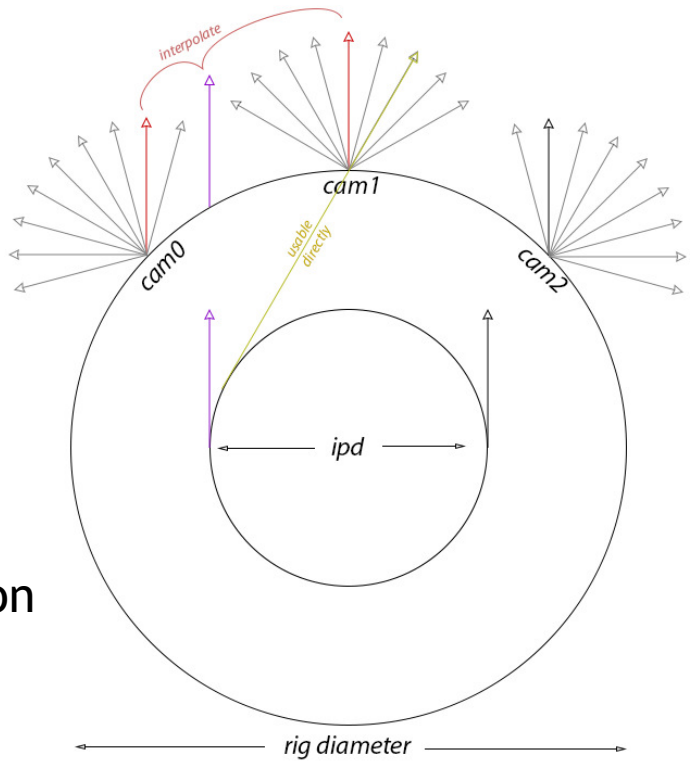


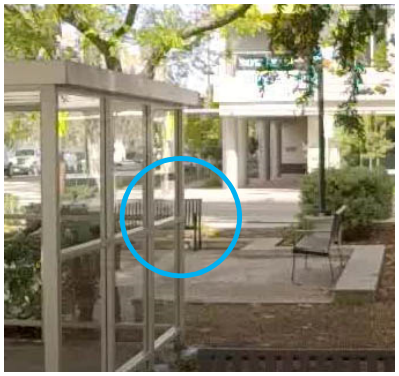
IMAGE ALIGNMENT CHALLENGES

textureless regions

periodic textures

specular surfaces

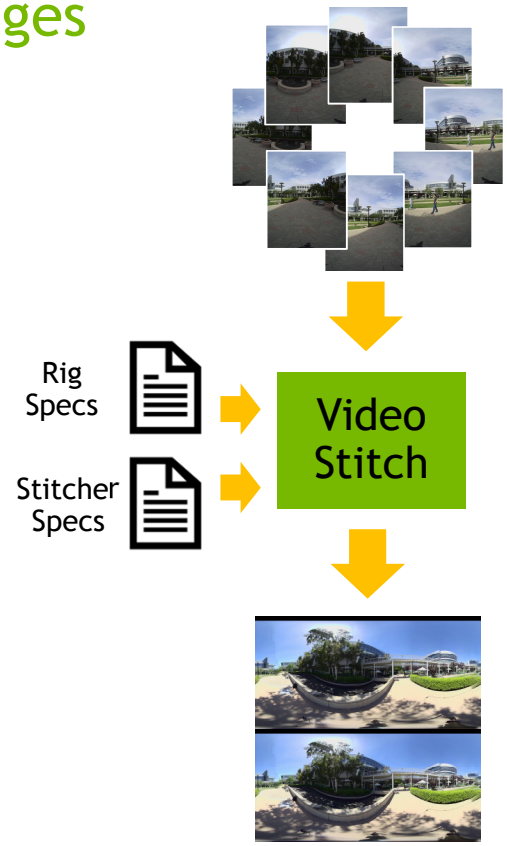
occlusion



VIDEO STITCHING

Overcoming Pixel Alignment Challenges

- Detect occlusion boundaries
- Reduce pixel motion noise between occlusion boundaries
- Fill holes (textureless regions) with plausible motion
- Enforce temporal coherence



AUDIO

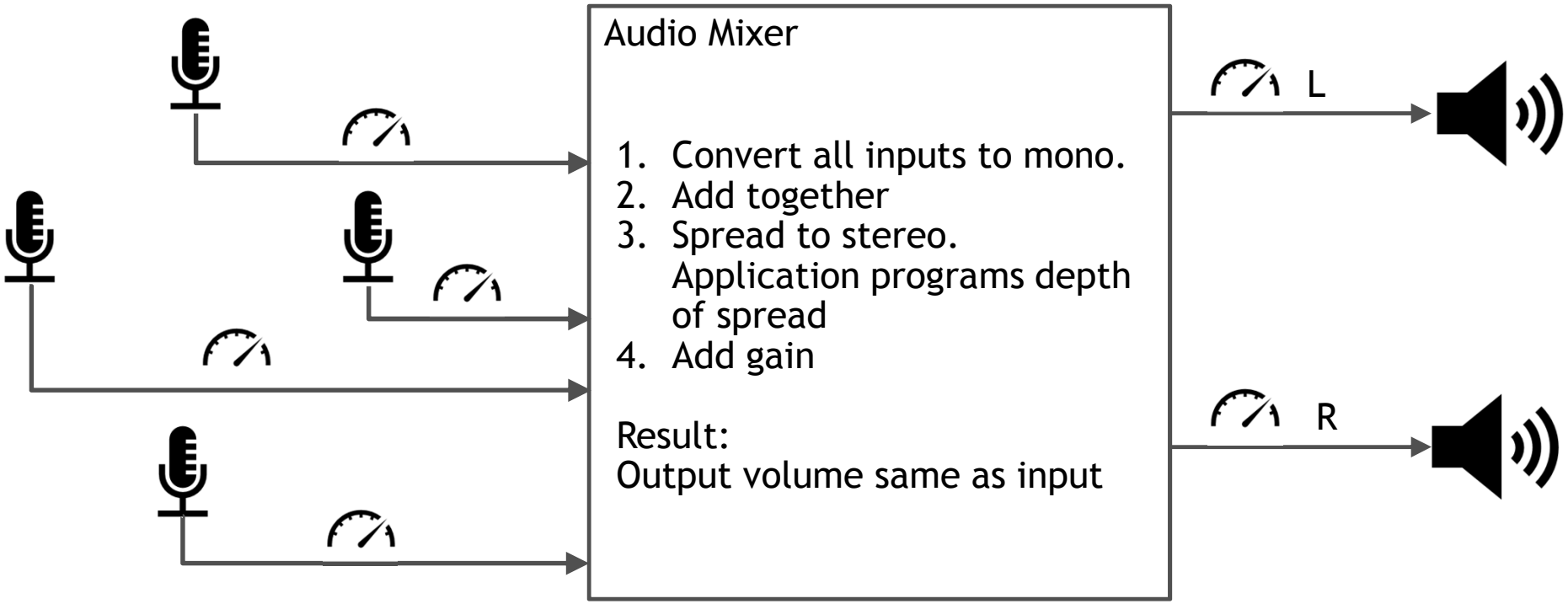
AUDIO PROCESSING

Overview

- Stereo mixdown with adjustable spread effect
- MP4 embedded audio, separate audio streams or individual audio buffers
- PCM and AAC audio streams
- Audio resampling
- Performed in parallel on the CPU while the video is stitching on the GPU
- Unlimited number of internal and external audio sources
- Push-pull model

AUDIO PROCESSING

Stereo Mixdown With Spread Effect

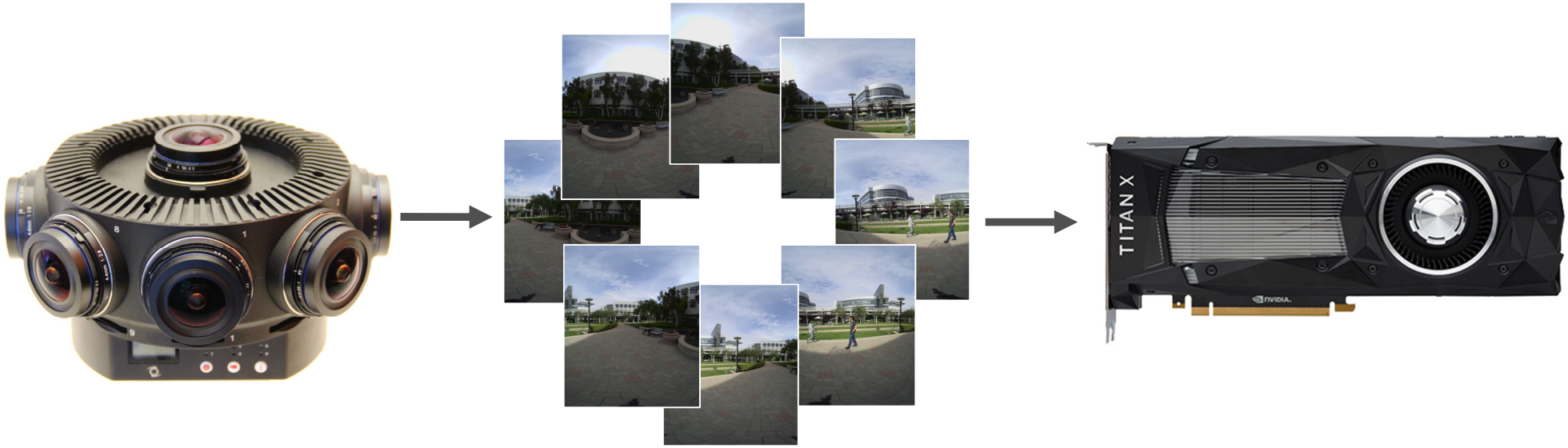


MULTI-GPU OPTIMIZATIONS

CHALLENGE

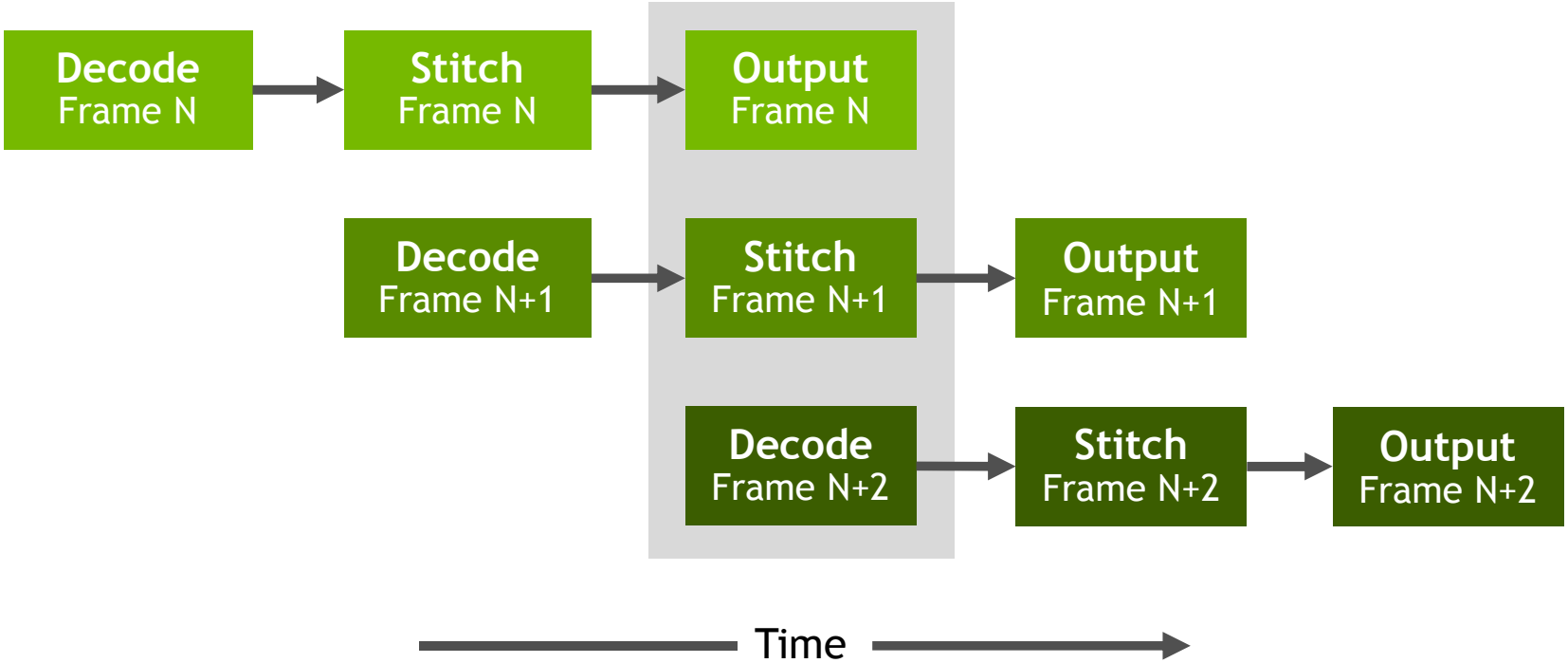
How can we decode **and** stitch in real-time?

- Need to decode 8 separate 4K streams at 30 fps (similar to 240 fps!)
- After getting frames to GPU, will we have any time left to stitch?



PIPELINING FOR PERFORMANCE

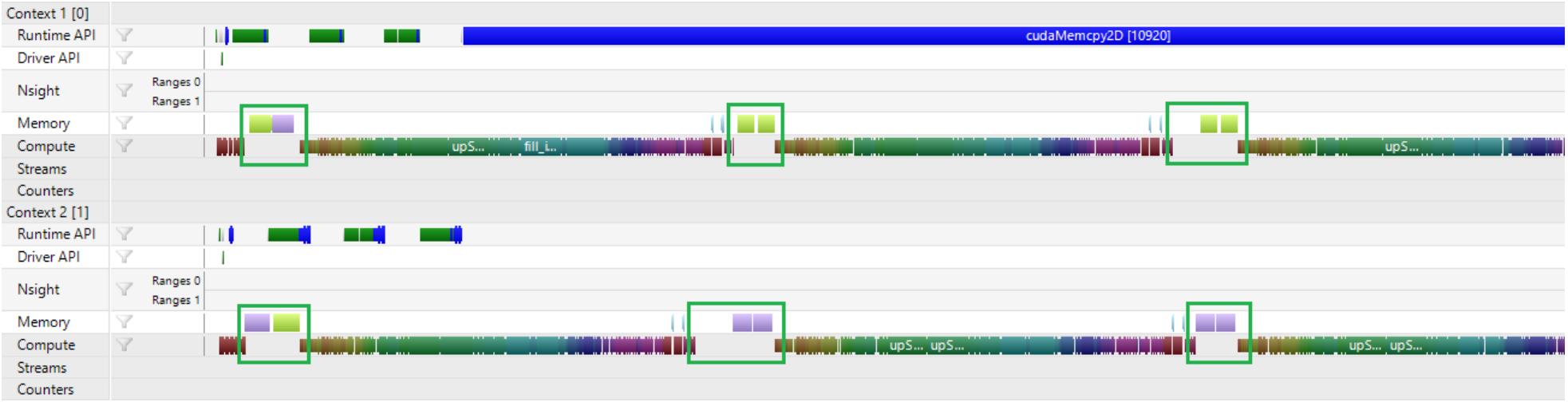
Pipelining Ingest, Stitch, and Output for Better Throughput



CHALLENGE

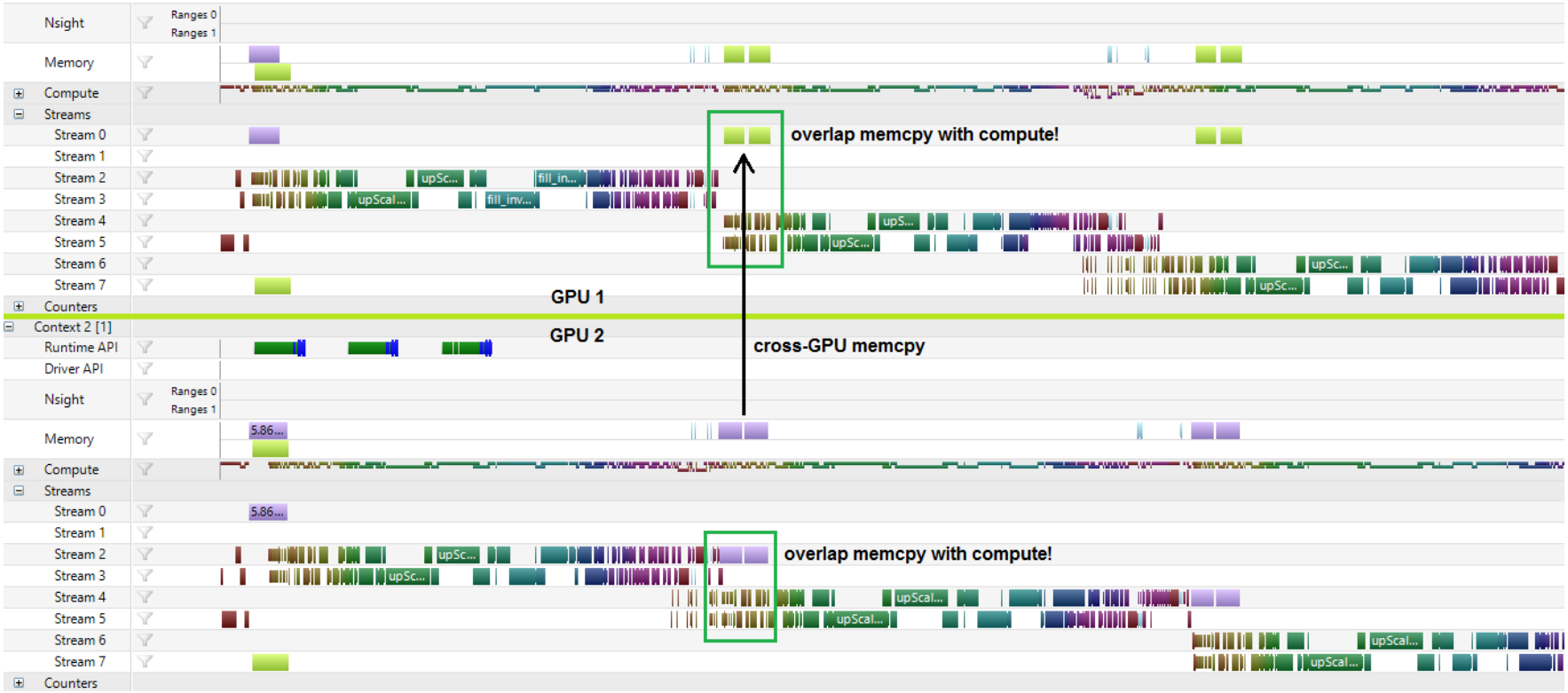
Minimizing Memory Copy Latency

- Must copy input/output between GPUs and CPU
- Synchronous memory copy injects bubbles in compute workload



HIDING MEMORY LATENCY

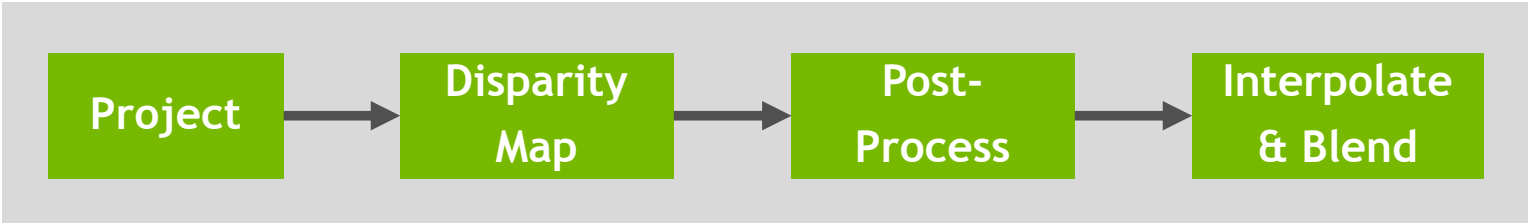
Using CUDA Streams and Asynchronous Memory Copies



CHALLENGE

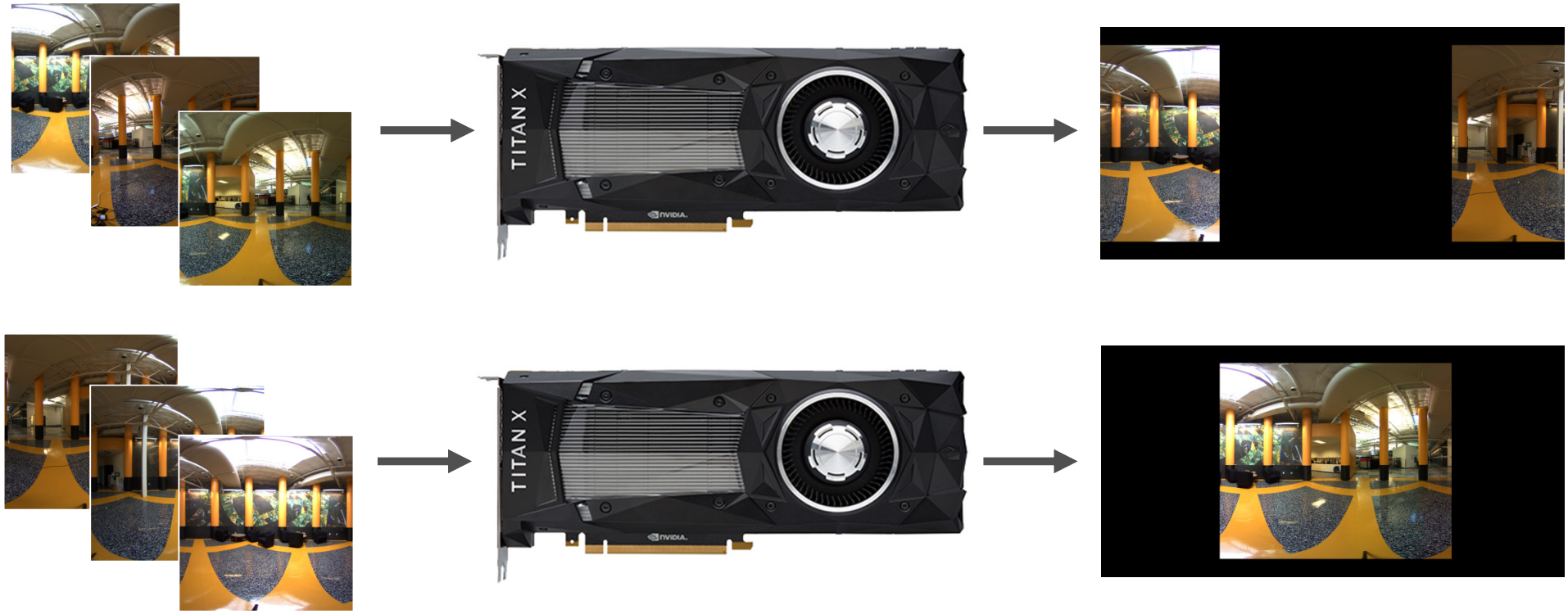
Maximum Quality Achievable in Real-Time

- We can stitch lower-quality on one GPU
- We want more cameras, higher quality, higher output resolution



MULTI-GPU SCALING

Distribute Stereo Pairs Among Available Devices



STITCHING PERFORMANCE

8x 4K input streams

STITCH TYPE	GPU	OUTPUT	STITCHING RATE (FPS)
Stereo	**2x Quadro P6000**	Uncompressed	32
Stereo	**2x Quadro P6000**	MP4	30
Mono	**Quadro P6000**	Uncompressed	40
Mono	**GTX 1060/1080**	Uncompressed	40

NVIDIA.

SAMPLE APPLICATIONS

VRWORKS 360 VIDEO SDK

Low-Level Video API Example

Create stitcher from rig parameters.

```
// Initialize stitcher instance
RETURN_NVSS_ERROR(nvssVideoCreateInstance(&stitcher_props, &params->rig_properties, &stitcher));

// Load image frames for each camera
for (uint32_t camera = 0; camera < params->rig_properties.num_cameras; camera++)
{
    nvstitchImageBuffer_t input_image;
    RETURN_NVSS_ERROR(nvssVideoGetInputBuffer(stitcher, camera, &input_image));

    // Read image from file
    getRgbaImage(image_file_path, &rgba_bitmap_ptr, image_width, image_height);

    // Copy image data to CUDA device memory
    cudaMemcpy2D(input_image.dev_ptr, input_image.pitch, rgba_bitmap_ptr, input_image.row_bytes,
                 input_image.row_bytes, input_image.height, cudaMemcpyHostToDevice);
}

// Stitch
RETURN_NVSS_ERROR(nvssVideoStitch(stitcher));
```

Get video input buffer

Copy image to CUDA device memory

Execute stitcher

VRWORKS 360 VIDEO SDK

Low-Level Video API Example (continued)

```
for (int eye = 0; eye < num_eyes; eye++)
{
    // Fetch stitched panorama(s)
    if (stereo)
        RETURN_NVSS_ERROR(nvssVideoGetOutputBuffer(stitcher, nvstitchEye(eye), &output_image));
    else
        RETURN_NVSS_ERROR(nvssVideoGetOutputBuffer(stitcher, NVSS_EYE_MONO, &output_image));

    if (out_stacked == nullptr)
        out_stacked = (unsigned char *)malloc(output_image.row_bytes * output_image.height * num_eyes);

    // Copy to host memory
    cudaMemcpy2D(out_stacked + out_offset, output_image.row_bytes, output_image.dev_ptr,
                 output_image.pitch, output_image.row_bytes, output_image.height, cudaMemcpyDeviceToHost);

    out_offset += output_image.height * output_image.row_bytes;
}
// Write to file
putRgbaImage(params->out_file, out_stacked, output_image.width, output_image.height * num_eyes);

// Clean up
RETURN_NVSS_ERROR(nvssVideoDestroyInstance(stitcher));
```

Get stitched panoramas

Copy to host memory

Destory stitcher

VRWORKS 360 VIDEO SDK

High-Level API Example

Create video rig instance

Create audio rig instance

Create stitcher

Start stitcher

Feed stitcher

Stop stitcher

Destroy stitcher

```
// Create video rig
RETURN_NVSTITCH_ERROR(nvstitchCreateVideoRigInstance(&rig_properties, &stitcher_properties.video_rig));

// Create audio rig
RETURN_NVSTITCH_ERROR(nvstitchCreateAudioRigInstance(&audio_rig_properties,
                                                     &stitcher_properties.audio_rig, NULL));

// Create stitcher instance
RETURN_NVSTITCH_ERROR(nvstitchCreateStitcher(&params->stitcher_properties, &stitcher));

// Start stitcher
RETURN_NVSTITCH_ERROR(nvstitchStartStitcher(stitcher, nullptr, nullptr));

// Load video inputs
for (uint32_t i{}; i < (uint32_t)params->rig_properties.num_cameras; ++i)
    RETURN_NVSTITCH_ERROR(nvstitchFeedStitcherAudioVideo(0, stitcher, i, i, &params->payloads[i], false));

// Stop stitcher
RETURN_NVSTITCH_ERROR(nvstitchStopStitcher(stitcher));

// Clean up
RETURN_NVSTITCH_ERROR(nvstitchDestroyStitcher(stitcher));
```

VRWORKS 360 VIDEO SDK

Calibration

Create calibration instance

```
// Create calibration instance
RETURN_NVSTITCH_ERROR(nvstitchCreateCalibrationInstance(calib_prop, &h_calib));

// Read and load input images for calibration
for (uint32_t cam_index = 0; cam_index < camera_count; cam_index++)
{
    getRgbaImage(image_file_path, &rgba_bitmap_ptr, image_width, image_height);
    nvstitchPayload_t calib_payload = nvstitchPayload_t{ calib_prop.input_form,{ width, height } };
    calib_payload.payload.buffer.ptr = rgba_bitmap_ptr;
    calib_payload.payload.buffer.pitch = rig_properties.cameras[cam_index].image_size.x * num_channels;

    RETURN_NVSTITCH_ERROR(nvstitchFeedCalibrationInput(frame_index, h_calib, cam_index, &calib_payload));
}

// Calibrate
RETURN_NVSTITCH_ERROR(nvstitchCalibrate(h_calib, &h_calibrated_video_rig));


// Destroy calibration instance
RETURN_NVSTITCH_ERROR(nvstitchDestroyCalibrationInstance(h_calib));
```

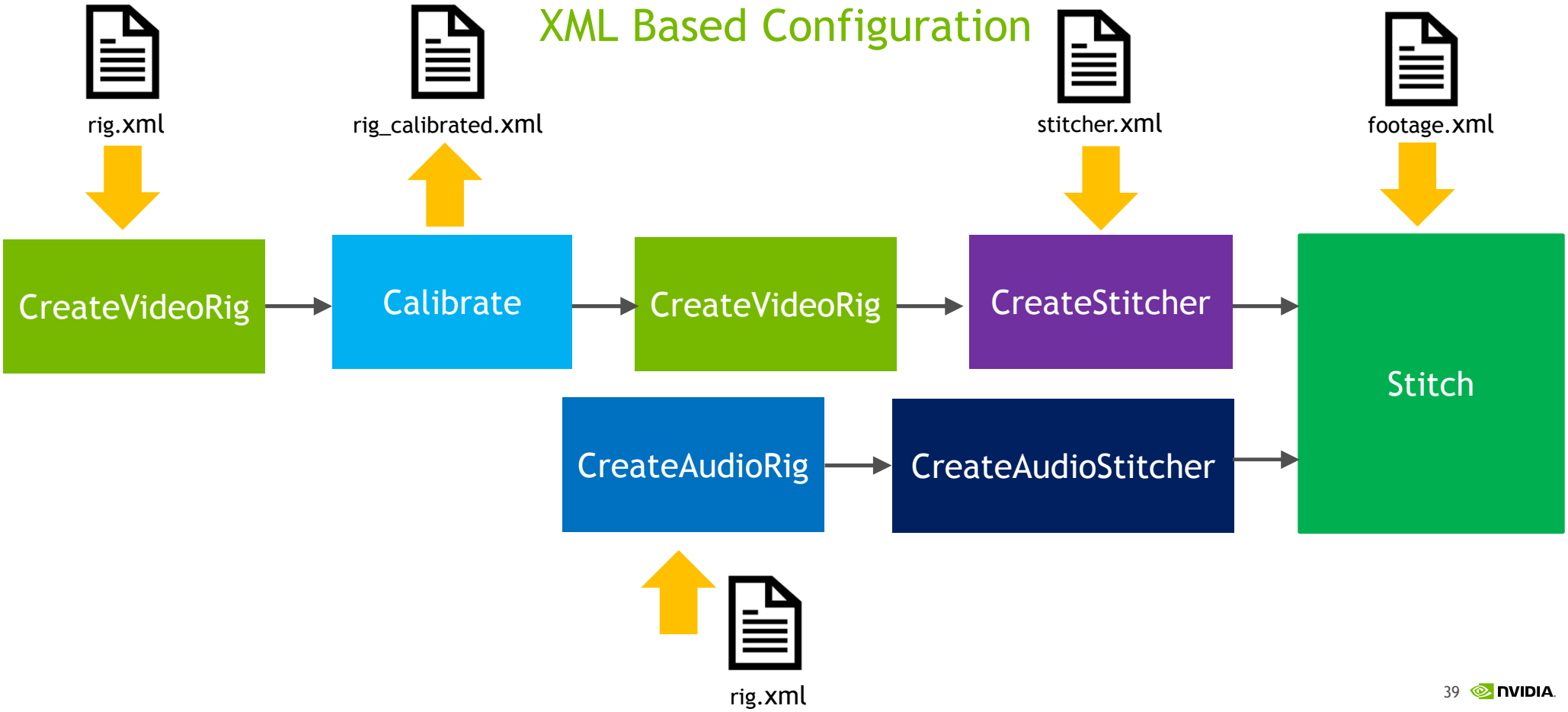
Feed images

Calibrate. Return updated video rig

Destroy calibration instance

VRWORKS 360 VIDEO SDK

XML Based Configuration



VRWORKS 360 VIDEO SDK

XML Examples

```
// Read from Xml
bool readCameraRigXml(const std::string& xmlPath,
                      std::vector<nvstitchCameraProperties_t>&cameraProperties,
                      nvstitchVideoRigProperties_t* videoRigProperties);

// Read filenames from Xml
// For nvcalib
bool readInputCalibFilenamesXml(const std::string& xmlPath,
                                std::vector<std::vector<std::string>>& filenames);

// Read media filenames Xml
bool readInputMediaFeedFilenamesXml(const std::string& xmlPath, std::vector<std::string>& filenames);

bool readInputMediaFeedXml(const std::string& inputFeedXmlPath,
                           std::vector<nvstitchPayload_t>& mediaPayloadArray, int* arrayCount);

bool readInputAudioFeedXml(const std::string& inputFeedXmlPath,
                           std::vector<nvstitchAudioPayload_t>& audioPayloadArray, int* arrayCount);

bool readStitcherPropertiesXml(const std::string& stitcherPropertiesXmlPath,
                               const std::string& inputFeedXmlPath,
                               nvstitchStitcherProperties_t* stitcherProperties);
```

VRWORKS 360 VIDEO SDK

Utilities

Image I/O

```
bool getRgbImage(const string& imgPath, unsigned char** imgRgb,
                 int& imgWidth, int& imgHeight)

bool getRgbaImage(const string& imgPath, unsigned char** imgRgb,
                  int& imgWidth, int& imgHeight)

bool putRgbaImage(const string& imgPath, unsigned char* imgRgba,
                  int imgWidth, int imgHeight)

bool showImageRgb(int width, int height, unsigned char* imgRgb)
```

EXHIBITION

Come See Our Demo in the VR Village!

VRWORKS 360 VIDEO SDK

Available Now

- GPU-optimized stereo pipeline
- GPU-accelerated calibration
- GPU-accelerated encode and decode
- Audio mix down and stretch
- Audio encode / decode
- Audio resampling
- Realtime and offline
- Stereo and mono stitching
- Compressed and uncompressed inputs and outputs
  - MP4, H.264 and RGBA8
  - PCM and AAC audio

https://developer.nvidia.com/vrworks/vrworks-360video

Q & A

NVIDIA