HoloLens 2 Research Mode: API Overview

D. Ungureanu, P. Sama

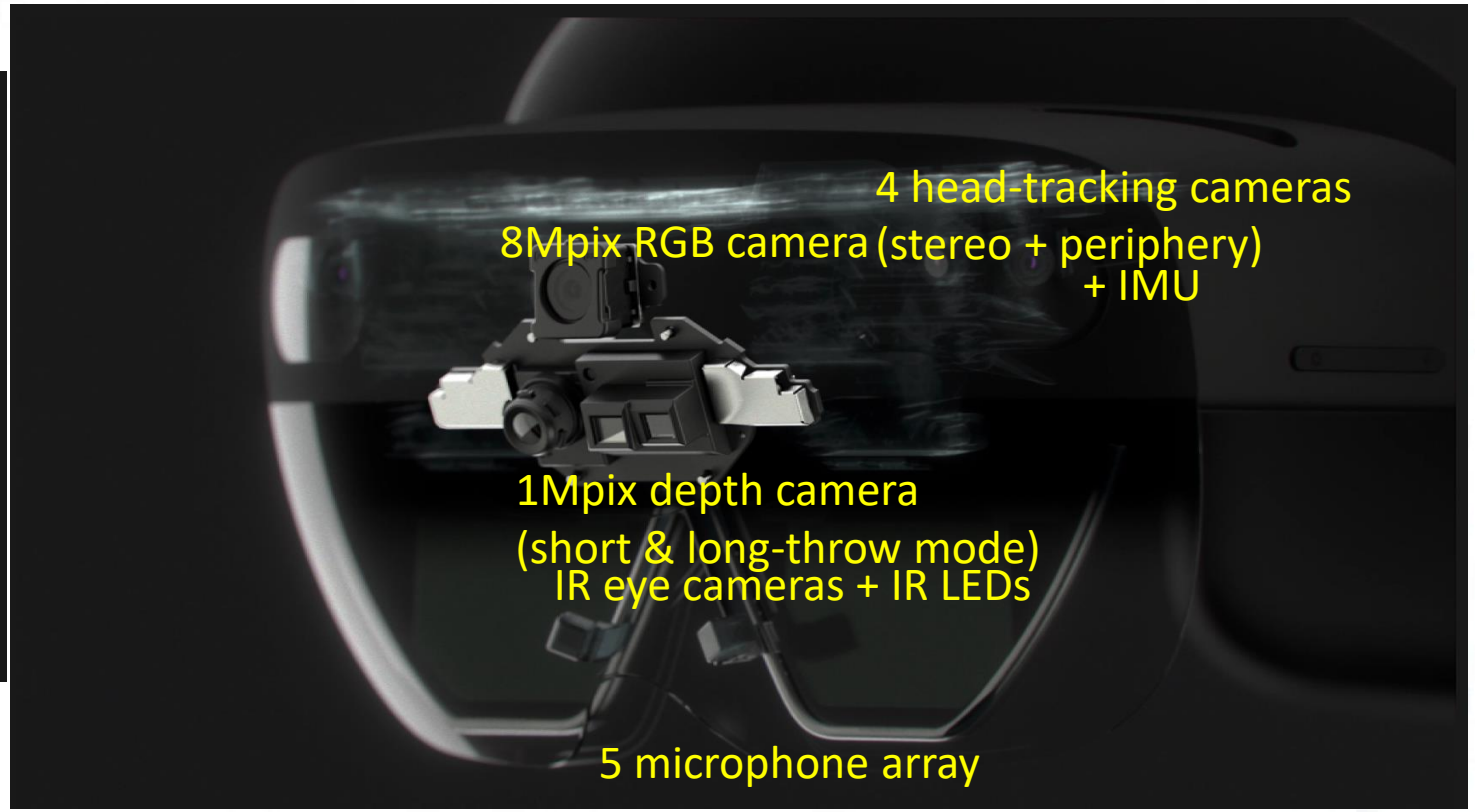
{dorinung, sasama}@microsoft.com

Outline

- Why Research Mode?
- HoloLens 2 Research Mode: What's New?
- Setup
- API overview
- Github repo and sample apps

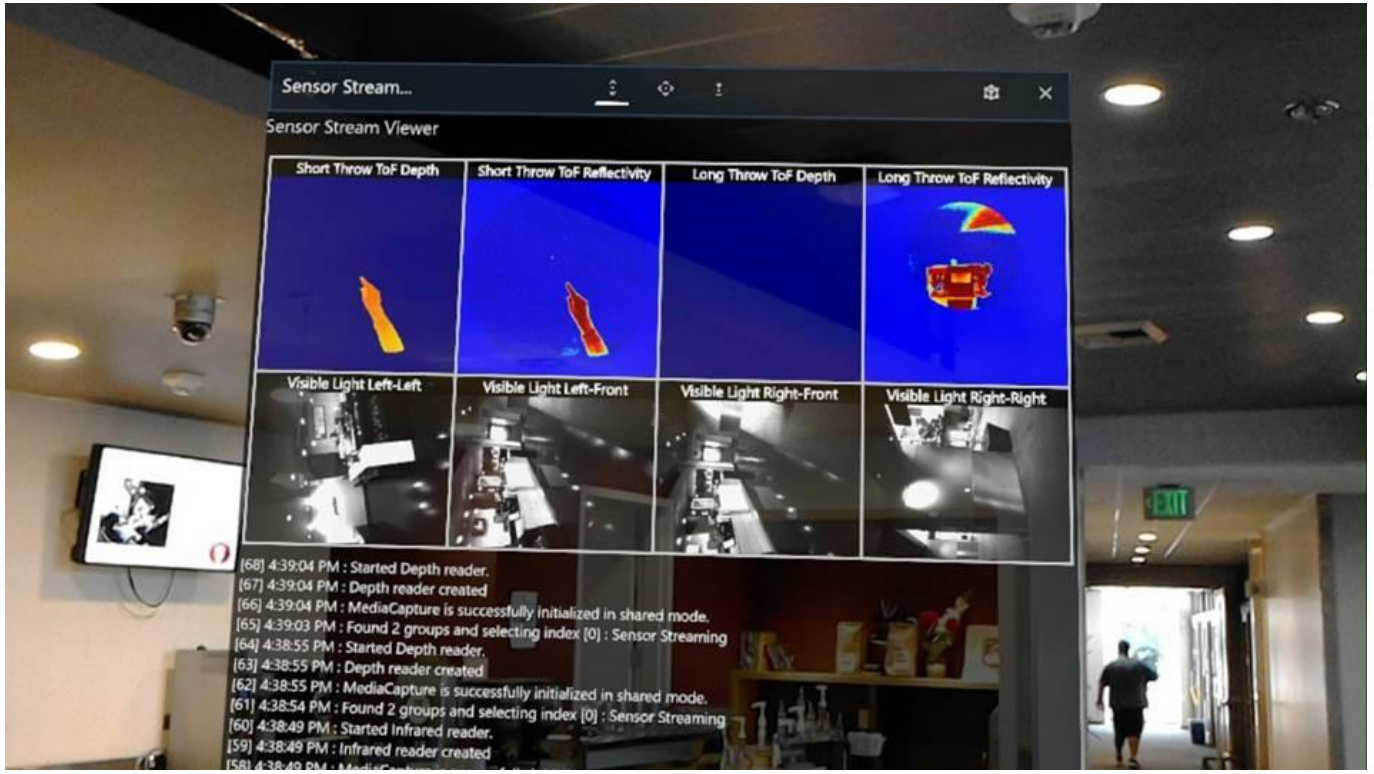
Why Research Mode

- HoloLens 2 as a powerful platform for research in computer vision / robotics



Looking Back: HoloLens 1 Research Mode

- Gave access to Depth (short/long throw), IR, VLC (grayscale cameras) streams
- Sample apps and utilities available at https://github.com/microsoft/HoloLensForCV



HoloLens 2

- New hardware! Old Research Mode code not compatible.
- Research Mode now gives access to:
  - Visible Light Environment Tracking Cameras (VLC)
  - Depth Camera (Long Throw, AHAT)
  - IMU sensors (accelerometer, gyroscope and magnetometer)
- Resources: https://docs.microsoft.com/en-us/windows/mixed-reality/research-mode#enabling-research-mode-hololens-1st-gen-and-hololens-2
- New repository: https://github.com/microsoft/HoloLens2ForCV

Requirements

- Windows 10
  - Desktop
  - HoloLens 2
    (build 19041.1364.VB_RELEASE_SVC_SYDNEY.200807-1600 or newer)

- Visual Studio requirements
  - Download free copy of Visual Studio 2017 or 2019
  - Install C++, UWP, Windows SDK components

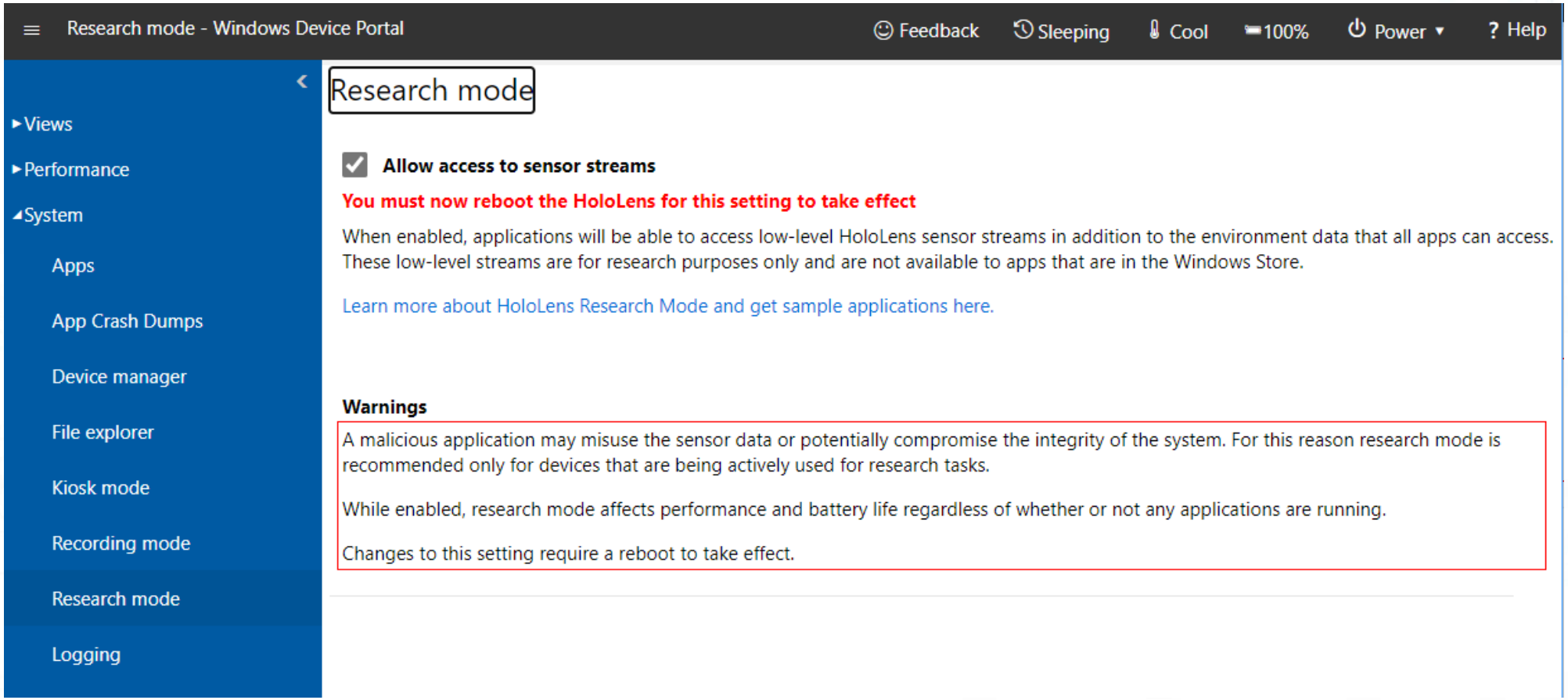
Setup

Setting up Research mode on HL2

- **Turn on Developer mode:**
  On your HoloLens 2 device, go to Settings > Update & security > For developers and Turn on Developer mode

- **Turn on device portal:**
  On your HoloLens 2 device, go to Settings > Update & security > For developers and Turn on device portal

- **Enable research mode:**
  Plug your HoloLens into your PC, open a browser, in the address bar go to http://127.0.0.1:10080:
  - Hit "Request PIN". A PIN will show in your HoloLens. Enter that PIN into the browser dialog,
  - Setup a username and password to enter device portal with that HoloLens,
  - Enter the username and password in the browser dialog,
  - From device portal, go to System > Research mode and Check "Allow access to sensor streams"


Research mode - Windows Device Portal
Feedback
Sleeping
Cool
100%
Power
Help
Views
Performance
System
Apps
App Crash Dumps
Device manager
File explorer
Kiosk mode
Recording mode
Research mode
Logging
Research mode
Allow access to sensor streams
You must now reboot the HoloLens for this setting to take effect
When enabled, applications will be able to access low-level HoloLens sensor streams in addition to the environment data that all apps can access. These low-level streams are for research purposes only and are not available to apps that are in the Windows Store.
Learn more about HoloLens Research Mode and get sample applications here.
Warnings
A malicious application may misuse the sensor data or potentially compromise the integrity of the system. For this reason research mode is recommended only for devices that are being actively used for research tasks.
While enabled, research mode affects performance and battery life regardless of whether or not any applications are running.
Changes to this setting require a reboot to take effect.

Setup

Adding the perceptionSensorsExperimental capability

You should add to the Package.appxmanifest of your app the perceptionSensorsExperimental capability:

```
<Capabilities>
  <Capability Name="internetClient" />
  <uap:Capability Name="documentsLibrary" />
  <rescap:Capability Name="perceptionSensorsExperimental" />
  <DeviceCapability Name="webcam" />
  <DeviceCapability Name="wifiControl" />
</Capabilities>
```

Research Mode Sensors

Sensor types

- Grey scale Visible Light Cameras (VLC)
- Depth Camera with two modes
  - Articulated HAnd-Tracking mode (AHAT),
  - Long-throw mode
- IMU Sensors:
  - Accelerometer,
  - Gyroscope,
  - Magnetometer

Research Mode API

Low-level Sensor Access

- Timestamps
- Camera images
- IMU samples batches
- Sensor Poses
  (relative and absolute sensor poses)
- Intrinsic sensor calibration
  (undistorted map from pixels to rays)

API Outline

A **research mode device** is the first object created. It is used to:

- Enumerate available sensors by type
- Create sensor objects
- Request access consent
- Only one sensor object per sensor type can be created

Sensor objects provide the following functionality:

- Return sensor name and type
- Start and stop streaming
- In streaming state wait for and retrieve frames
- Return extrinsics transform that gives position of the sensor relative to a device attached origin (Rig Origin),
- Return the Rig/Device Coordinate frame GUID that can be used to map the Rig coordinate frame to other perception coordinate frames,
- Sensors can be cameras or IMUs and both return frames with sensor-specific payload formats

Sensor frames specify:

- Frame timestamps
- Frame sizes
- Specialized per-sensor properties and payload formats

Main Sensor Loop

```
CreateResearchModeSensorDevice(&pSensorDevice ); // Factory function

pSensorDevice->GetSensorCount(&sensorCount);

pSensorDevice->GetSensorDescriptors(sensorDescriptors.data(), sensorDescriptors.size(), &sensorCount);

for (const auto& sensorDescriptor : sensorDescriptors)
{
    pSensorDevice->GetSensor(sensorDescriptor.sensorType, &pSensor);

    pSensor->GetSampleBufferSize(&sampleBufferSize);

    pSensor->OpenStream();

    for (UINT i = 0; i < 4; i++)
    {
        pSensor->GetNextBuffer(&pSensorFrame);

        ProcessFrame(pSensor, pSensorFrame, i);
    }

    pSensor->CloseStream();
    pSensor->Release();
}
```

Threading

- OpenStream and GetNextBuffer for a sensor need to be called from the same thread
- GetNextBuffer calls are blocking
- Run sensor frame loops for each sensor on its own thread to allow sensors to be processed at their own frame rate
- Recommended threading structure:
  - The main thread manages ResearchMode device and sensors
  - Each sensor has a worker thread, started by the main thread, that opens the sensor streams and reads buffers and processes buffers
  - The main thread retrieves buffers from the sensor worker threads

Sensors

Sensor type:

Camera Sensors

- Intrinsics (project/unproject in camera space)
- Extrinsics returns R, T transform in rig space
- Frames are specialized for cameras

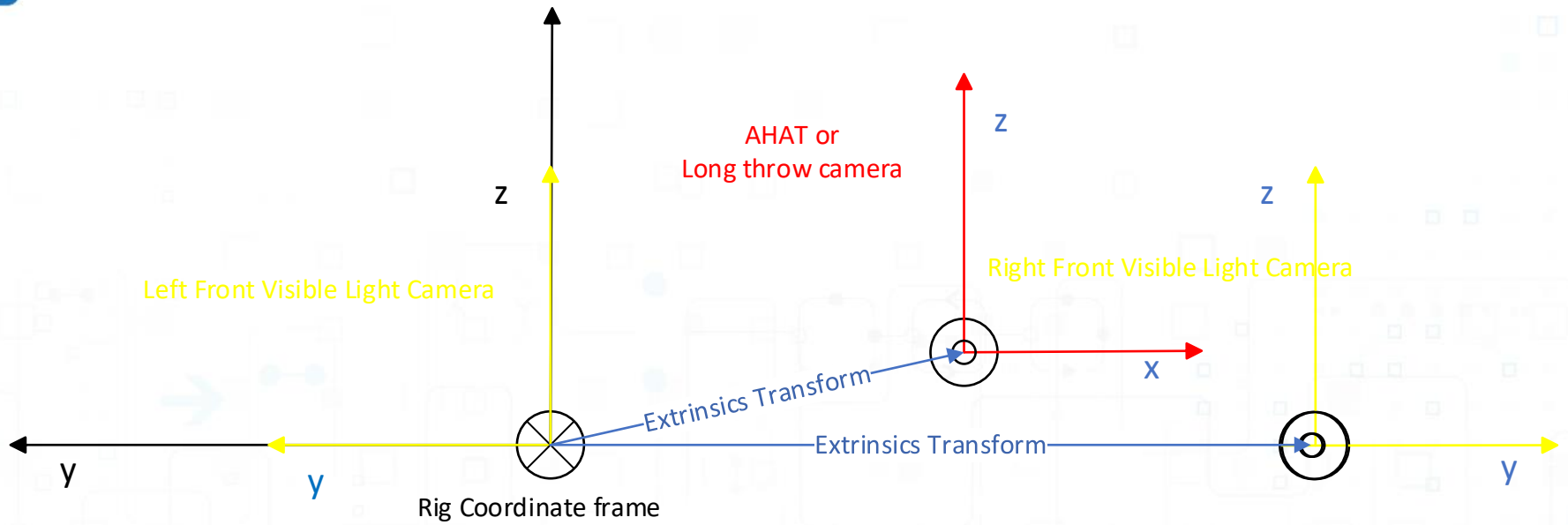
IMU Sensors

- Extrinsics returns R, T transform in rig space
- Frames are specialized for IMUs

```
Sensor Types:

enum ResearchModeSensorType
{
    LEFT_FRONT,
    LEFT_LEFT,
    RIGHT_FRONT,
    RIGHT_RIGHT,
    DEPTH_AHAT,
    DEPTH_LONG_THROW,
    IMU_ACCEL,
    IMU_GYRO,
    IMU_MAG
};
```

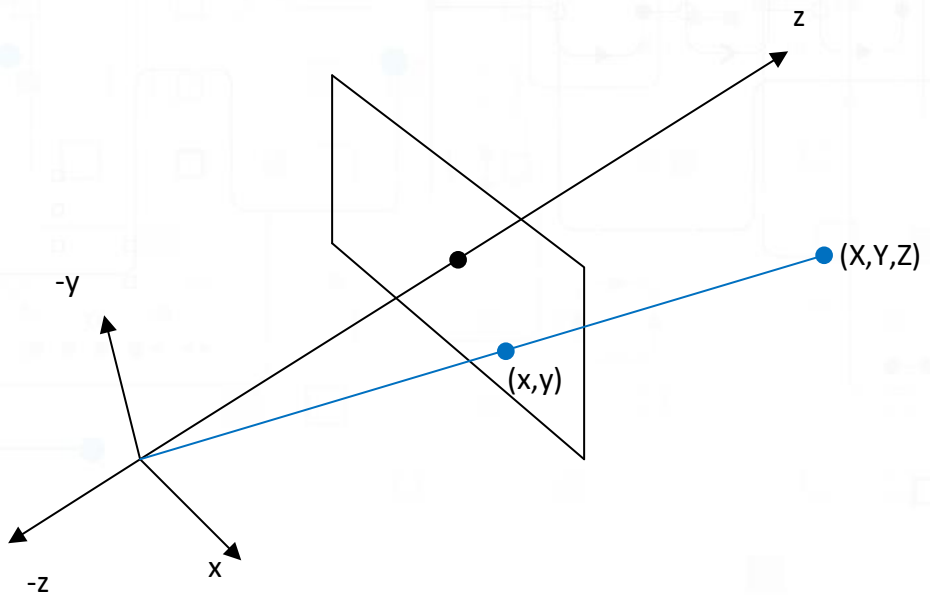
Sensor Coordinate Frames



- Sensor coordinate frames in rigSpace.
- API returns Perception rigSpace GUID
- The rigSpace GUID can be used to map to other Perception coordinate systems
- Four VLCs (yellow), Depth camera (red)

Sensor Coordinate Frames

- Camera sensors expose map unmap methods
  (note that the coordinate system conforms with usual computer graphics conventions with the negative z-axis pointing forward as opposed to the usual computer vision convention)
  - MapImagePointToCameraUnitPlane
  - MapCameraSpaceToImagePoint



Sensor Frames

- **All sensors have frame specialized frame interfaces:**
  - All frame types:
    - Times stamps
      (These are in HostTicks and SensorTicks. HostTicks are in filetime units and SensorTicks are in nanoseconds)
    - Sample size in bytes
  - Camera frames:
    - All camera frames provide getters for resolution, exposure, gain
    - VLC camera frames return grayscale buffer
    - Depth Long throw camera frames contain active brightness buffer, distance buffer and the sigma buffer
    - Depth AHAT camera frames contain an active brightness buffer and a distance buffer
  - IMU frames:
    - They contain batches of sensor samples
    - Accelerometer frames contain accelerometer values, SensorTicks , CpuTicks and temperature
    - Gyroscope frames contain gyro meter values, SensorTicks , CpuTicks and temperature
    - Magnetometer frames contain magnetometer values, SensorTicks , CpuTicks

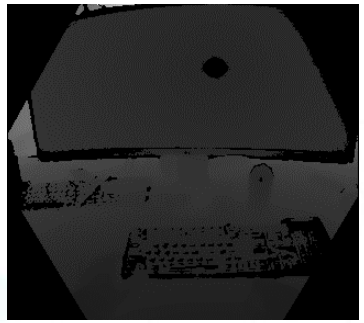
Sensor Frames

VLC images




Sensor Frames

- **Long throw Active Brightness and depth images**

Pixels in Long Throw images

LT has the invalidation codes and confidence embedded in the sigma buffer which is 8bit data for every depth pixel. If the MSB is set to 1, then the other 7 bits represents the invalidation reasons.

- **AHAT Active Brightness and depth images**


- High-frequency (45 fps) near-depth sensing used for hand tracking
- “Aliased depth” from the phase-based ToF camera: the signal contains only the fractional part of the distance from the device when expressed in meters
- “Wraps around” every 1m,
- Pixels greater than 4090 are invalid.

Sensor Frames

- **Accelerometer frames:**
  - Contain acceleration along the X, Y and Z axes.
  - Contain 93 samples in a single frame
- **Gyroscope frames:**
  - Contain angular rotations around the X,Y and Z axes.
  - Contain 315 samples in a single frame
- **Magnetometer frames:**
  - Contain absolute orientation estimation along the X, Y and Z axes.
  - Contain 10 samples in a single frame

Consent Prompt

- User Consent - required for UWP applications accessing camera and IMU streams via ResearchMode API
- Steps to be followed by an app for requesting consent:
  - Declare DeviceCapabilities in the app manifest for Webcam and/or backgroundSpatialPerception
  - Query for SensorDeviceConsent interface
  - Register camera and/or IMU consent callback from the main UI thread of the app
  - Define the callbacks where user consent is captured
  - Use a worker thread and wait on the callback for the consent provided by the user
  - Threads that open sensor streams need to wait for consent callback before opening streams
- Prompts appear only on the first use of the app. To change the access given to an app, the user can go to:
  - Settings->Privacy-> Camera→Application for cameras
  - Settings->Privacy→User Movements→Application for IMUs.