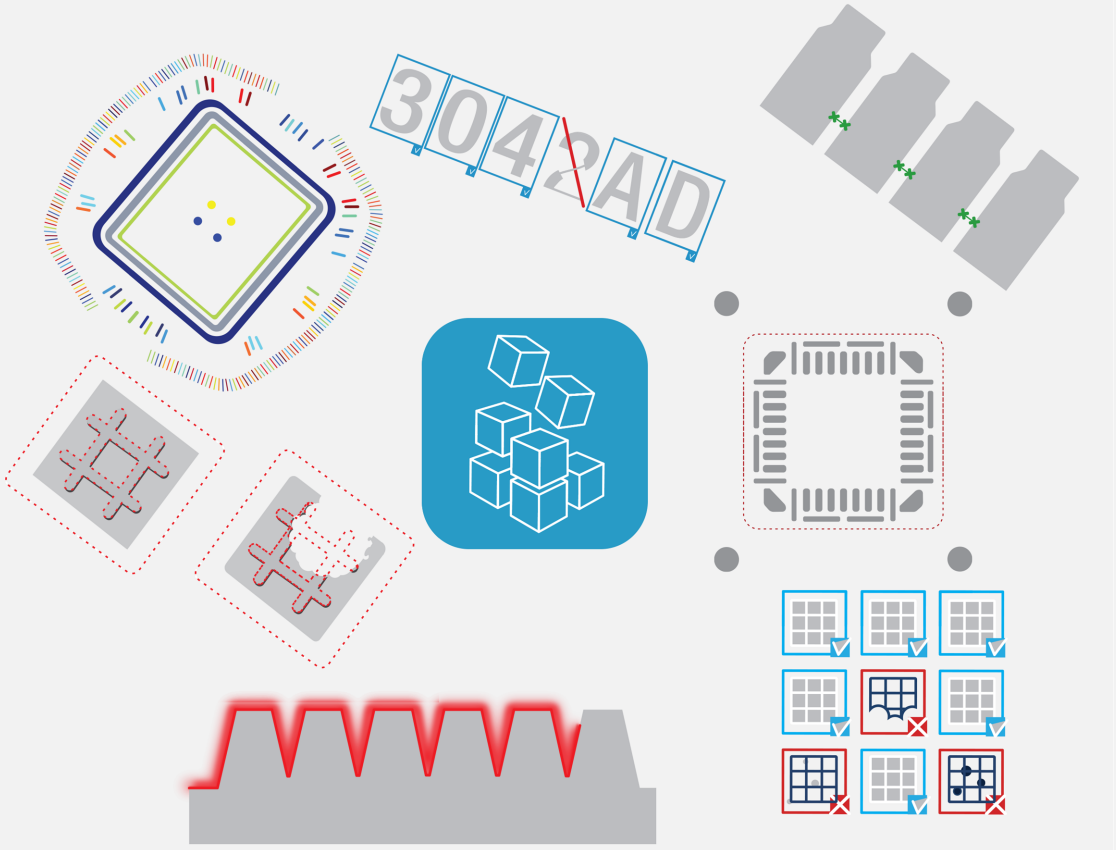
euresys
Empowering Computer Vision

USER GUIDE

Open eVision

Easy3D Compatibility with Intel RealSense 3D Sensors



This documentation is provided with Open eVision 2.14.0 (doc build 1143).
www.euresys.com

Easy3D Compatibility with Intel RealSense 3D Sensors

Introduction

The **Intel RealSense D415 and D435** are active stereo cameras, targeting medium range 3D acquisition. There are available as USB devices or depth module sub-systems.

The specifications are available on the manufacturer website:
https://www.intelrealsense.com/stereo-depth/

The **Intel RealSense L515** is a LIDAR depth camera.

The specifications are available on the manufacturer website:
https://www.intelrealsense.com/lidar-camera-l515/

- This document explains how to use the 3D data coming from these sensors with **Open eVision** 3D libraries and tools.
- A sample application distributed with source code demonstrates that integration. This application is freely available in the Easy3D Sensors Compatibility additional resources package on **Euresys** web site.

Resources

This document and the sample applications are based on the following resources:

- **Intel RealSense D415 and D435**, firmware 5.12
- **Intel RealSense L515**, firmware 01.05.00
- **Intel RealSense** SDK 2.38
- **Open eVision** 2.13
- Microsoft Visual Studio 2017

Features

- The **Intel RealSense** cameras and SDK expose the depth data in the following formats:

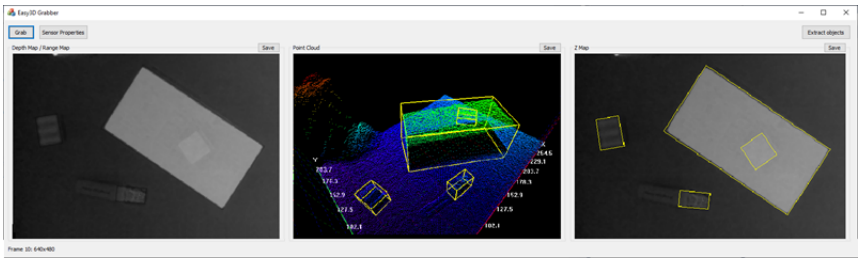
Format	Description	Bits per pixel
RS2_FORMAT_Z16	16-bit linear depth values	16
RS2_FORMAT_XYZ32F	32-bit floating point 3D coordinates	96
RS2_FORMAT_DISTANCE	32-bit floating point depth distance value	32

- The XYZ positions are expressed in a coordinate system centered on the camera with a Z axis towards the scene.

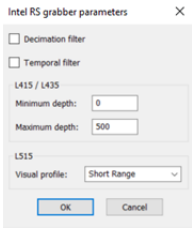
Easy3DGrab sample application

Easy3DGrab is distributed with C++ source code as an **Open eVision** additional resource.

- It features the acquisition of an Intel RealSense depth frame, the conversion to depth maps, point clouds and ZMaps.
- You can save these representations.
- Click on the Grab button to acquire a new image.
- Open the Sensor Properties dialog to access some of the device parameters.
- The Object extraction function is exposed but you can use it only with the Easy3DObject license.



The Easy3DGrab application: EDepthMap (left), EPointCloud (center), EZMap (right) with automatic extraction of 3D objects with Easy3DObject library



The Sensor Properties button opens a dialog exposing some camera controls

C++ code sample to convert Intel RealSense range data to Easy3D objects

Converting Intel RealSense depth data

The **Intel RealSense** SDK can generate `EDepthMap16` and `EPointCloud` objects.

Here are the code snippets to fill an `Easy3D::EDepthMap16` and an `Easy3D::EPointCloud` objects from an `rs2::depth_frame`.

- Initialize a data stream with a 16 bits linear depth format.

```
rs2::config config;
config.enable_stream(RS2_STREAM_DEPTH, 640, 480, RS2_FORMAT_Z16);
rs2::pipeline_profile pipeline_profile = pipeline.start(config);
```

- Acquire frames (without postprocessing):

```
rs2::frameset frames;
frames = pipeline.wait_for_frames(timeout_ms);
```

- Convert the acquired frame to an `Easy3D::EDepthMap`.

```
rs2::depth_frame df = frames.get_depth_frame();

int w = df.get_width();
int h = df.get_height();
int stride = df.get_stride_in_bytes();
const uint8_t* data = (const uint8_t*)df.get_data();

// Copy buffer to EDepthMap16
map.SetSize(w, h);

for (int y = 0; y < h; ++y)
{
  const uint8_t* src = data + (y*stride);
  void* dst = map.GetBufferPtr(0, y);
  memcpy(dst, src, 2 * w);
}
```

- Convert the acquired frame to an `Easy3D::EPointCloud`.

```
rs2::depth_frame df = frames.get_depth_frame();
rs2::pointcloud src_pc;
rs2::points points = src_pc.calculate(df);

size_t size = points.size();
const rs2::vertex* vertices = points.get_vertices();

// Push valid 3D points to an EPointCloud
std::vector<Easy3D::E3DPoint> pts;
pts.reserve(size);
Easy3D::E3DPoint p;
for (size_t i = 0; i < size; ++i, vertices++)
{
  // (0,0,0) means no data (the origin is the center of the camera)
  if (vertices->x != 0.f || vertices->y != 0.f || vertices->z != 0.f)
  {
    // Convert from meters to millimeters
    p.X = vertices->x * 1000.f;
    p.Y = vertices->y * 1000.f;
    p.Z = vertices->z * 1000.f;
    pts.push_back(p);
  }
}
```

```
Easy3D::EPointCloud& pc;
pc.AddPoints(pts);
```

ZMap

- You cannot generate a ZMap (a gray scale image encoding distance from a reference plane) directly from **Intel RealSense** data.
- Generate a ZMap from the point cloud with the `Easy3D::EPointCloudToZMapConverter` class.

> **TIP**
> The sample application **Easy3DGrab** implement the `EDepthMap16`, `EPointCloud` and `EZMap` conversions.