

# Jump Stitch Metadata & Depth Maps

## Version 1.0

jump-help@google.com

### Contents

<b>1. Introduction</b>	<b>1</b>
<b>2. Stitch Metadata File Format</b>	<b>1</b>
<b>3. Coordinate Systems</b>	<b>3</b>
<b>4. Camera Model</b>	<b>4</b>
<b>5. From Pixels To Rays</b>	<b>5</b>
<b>6. Depth Maps</b>	<b>7</b>
<b>A. Example Stitch Metadata</b>	<b>8</b>
<b>B. Version History</b>	<b>8</b>

## 1. Introduction

This document defines the stitch metadata and depth map file formats produced by the Jump Assembler. It also describes the projection equations by which depth map values map to 3D points in the scene and how 3D scene points project into the source cameras.

## 2. Stitch Metadata File Format

For every shot, the Jump Assembler exports a [JSON](#) formatted metadata file. This file contains information about the rig including calibration parameters for each of the cameras,

the vertical field of view of the shot, and the interpupillary distance. See Appendix A for an example.

In this section we describe each field found in the metadata file. Some field names are followed by the mathematical symbol(s) that represent their value in equations in this document.

**major\_version\_number:** Major version of the metadata file. This value will be incremented when changes are made to the file format which are not backward compatible.

**minor\_version\_number:** Minor version of the metadata file. Incrementing this value indicates that one or more new fields have been added to the format. Any other type of change will result in an update to the **major\_version\_number** instead.

**interpupillary\_distance\_in\_meters:** The interpupillary distance (IPD) used for rendering the stitch, measured in meters.

**rig\_type:** A string defining the type of rig used for the capture. Currently the only valid value is *Odyssey*.

**rig\_id:** A string that gives a globally unique rig id. Two shots with the same **rig\_id** must have been captured using the same physical rig, although calibration may still change between the shots as individual components move or are replaced.

**missing\_bottom\_coverage\_in\_degrees:** The elevation, in degrees, at the bottom of the stitch, where content is missing.

**missing\_top\_coverage\_in\_degrees:** The elevation, in degrees, at the top of the stitch, where content is missing.

**minimum\_encodable\_depth\_in\_meters:** If depth stitches are saved then they are encoded as inverse depth. This value is the minimum depth which can be encoded. See section 6 for more details.

**distortion\_delta\_in\_meters:**  $\delta$  - Jump stitches are formed using an omnidirectional stereo (ODS) projection that has been distorted to better fit the physical camera geometry. This value is the distance by which ODS viewing ray origins are shifted to produce the distorted ODS projection. See Section 5 for more details.

**cameras:** An array of *cameras* which provide the intrinsics and extrinsics of each camera in the rig as described below.

Each camera in the **cameras** array contains the following fields:

**name:** The name of the camera. By convention this is the name of the video file corresponding to that camera without the file extension.

Table 1: Metadata field names and the mathematical symbols corresponding to them.

JSON Field	Symbol
<code>distortion_delta_in_meters</code>	$\delta$
<code>camera::position</code>	$[c_x, c_y, c_z]$
<code>camera::orientation</code>	$[a_x, a_y, a_z]$
<code>camera::image_size</code>	$[w, h]$
<code>camera::principal_point</code>	$[u_0, v_0]$
<code>camera::radial_distortion</code>	$[r_0, r_1]$
<code>camera::focal_length</code>	$f$

**position:**  $[c_x, c_y, c_z]$  - The position of the camera in world coordinates. Given in meters.

**orientation:**  $[a_x, a_y, a_z]$  - The orientation of the camera in angle-axis format, where angle in radians is equal to the magnitude of the vector.

**projection\_type:** The projection model used for the camera. Either **perspective** or **fisheye**. For the GoPro Odyssey, this value will always be **fisheye**.

**image\_size:**  $[w, h]$  - The width and height of the camera's image in pixels.

**principal\_point:**  $[u_0, v_0]$  - The horizontal and vertical position of the camera's principal point in pixels.

**radial\_distortion:**  $[r_0, r_1]$  - First and second order radial distortion parameters for the camera.

**focal\_length:**  $f$  - Focal length of the camera in pixels.

### 3. Coordinate Systems

In image space the top left corner of the top left pixel has coordinates  $[0, 0]$ . The center of the top left pixel lies at  $[0.5, 0.5]$  and for an image size of  $w \times h$  the center of the bottom right pixel is at  $[w - 0.5, h - 0.5]$ .

We use a right handed coordinate system. A camera at the origin which has not been rotated looks down the positive  $z$  axis, with the positive  $x$  axis pointing to the right and the positive  $y$  axis pointing down.

The output stitches are over-under equirectangular in which the left eye is on top. The positive  $y$  axis points down the image. The positive  $z$  axis corresponds to the left edge of the stitch.

## 4. Camera Model

In this section, we describe how a point in world space with homogeneous coordinates  $[p_x, p_y, p_z, p_w]$  is projected into the image space of one of the cameras in the rig with coordinates  $[u, v]$ . The process can be broken down into two steps as follows:

### 1. Transform Into Camera Space

For a camera with **position**  $[c_x, c_y, c_z]$  and **orientation**  $[a_x, a_y, a_z]$  we transform a point in homogenous coordinate in world space  $[p_x, p_y, p_z, p_w]$  to a point in camera space  $[x, y, z]$  using the following equation,

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} p_x - p_w \times c_x \\ p_y - p_w \times c_y \\ p_z - p_w \times c_z \end{bmatrix}. \quad (1)$$

Here, the  $3 \times 3$  rotation matrix  $R$  can be computed from the angle axis representation of **orientation**  $[a_x, a_y, a_z]$  as follows:

$$\theta = \sqrt{a_x^2 + a_y^2 + a_z^2}, \quad (2)$$

$$\begin{bmatrix} \hat{a}_x \\ \hat{a}_y \\ \hat{a}_z \end{bmatrix} = \frac{1}{\theta} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \quad (3)$$

$$R = \begin{bmatrix} \hat{a}_x \hat{a}_x (1 - \cos \theta) + \cos \theta & \hat{a}_x \hat{a}_y (1 - \cos \theta) - \hat{a}_z \sin \theta & \hat{a}_x \hat{a}_z (1 - \cos \theta) + \hat{a}_y \sin \theta \\ \hat{a}_x \hat{a}_y (1 - \cos \theta) + \hat{a}_z \sin \theta & \hat{a}_y \hat{a}_y (1 - \cos \theta) + \cos \theta & \hat{a}_y \hat{a}_z (1 - \cos \theta) - \hat{a}_x \sin \theta \\ \hat{a}_x \hat{a}_z (1 - \cos \theta) - \hat{a}_y \sin \theta & \hat{a}_y \hat{a}_z (1 - \cos \theta) + \hat{a}_x \sin \theta & \hat{a}_z \hat{a}_z (1 - \cos \theta) + \cos \theta \end{bmatrix} \quad (4)$$

Here,  $\theta$  is the angle of rotation around the unit vector  $[\hat{a}_x, \hat{a}_y, \hat{a}_z]$ .

### 2. Project Into Image Space

Given **focal\_length**  $f$ , **principal\_point**  $[u_0, v_0]$  and **radial\_distortion**  $[r_0, r_1]$ , we can now, depending on the **projection\_type**, project a point  $[x, y, z]$  in camera space into the image as follows.

- fisheye

$$\alpha = \tan^{-1} \frac{\sqrt{x^2 + y^2}}{z} \quad (5)$$

$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \frac{\alpha}{\sqrt{x^2 + y^2}} \begin{bmatrix} x \\ y \end{bmatrix}, \quad (6)$$

$$d = 1 + r_0 \alpha^2 + r_1 \alpha^4 \quad (7)$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f d \hat{x} + u_0 \\ f d \hat{y} + v_0 \end{bmatrix}. \quad (8)$$

Here,  $\alpha$  is the the angle between the optical axis and the ray to the point  $[x, y, z]$ , and  $d$  is the radial distortion factor.

- perspective

$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \frac{1}{z} \begin{bmatrix} x \\ y \end{bmatrix} \quad (9)$$

$$d = 1 + r_0 (\hat{x}^2 + \hat{y}^2) + r_1 (\hat{x}^2 + \hat{y}^2)^2 \quad (10)$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f d \hat{x} + u_0 \\ f d \hat{y} + v_0 \end{bmatrix} \quad (11)$$

Here,  $d$  is the radial distortion factor.

## 5. From Pixels To Rays

In this section we describe the mapping from a point in the stitched video to a 3D ray.

Jump videos use the omnidirectional stereo (ODS) projection. A summary of the ODS model and how it can be used to render imagery can be found in [Rendering Omni-directional Stereo Content](#)<sup>1</sup>. That document describes the *ideal* ODS projection, where viewing rays originate from a circle of diameter equal to the interpupillary distance (IPD). In practice though, the diameter of a Jump camera is significantly larger than IPD. To avoid holes in the stitch, we use a *distorted ODS projection* as shown in Figure 1. In this model the rays originate from the circle on which the cameras lie. Visually this results in subjects that are close to the camera being vertically stretched.

---

<sup>1</sup>Currently [Rendering Omni-directional Stereo Content](#) and this document differ in their coordinate system conventions. We are working on fixing this discrepancy.

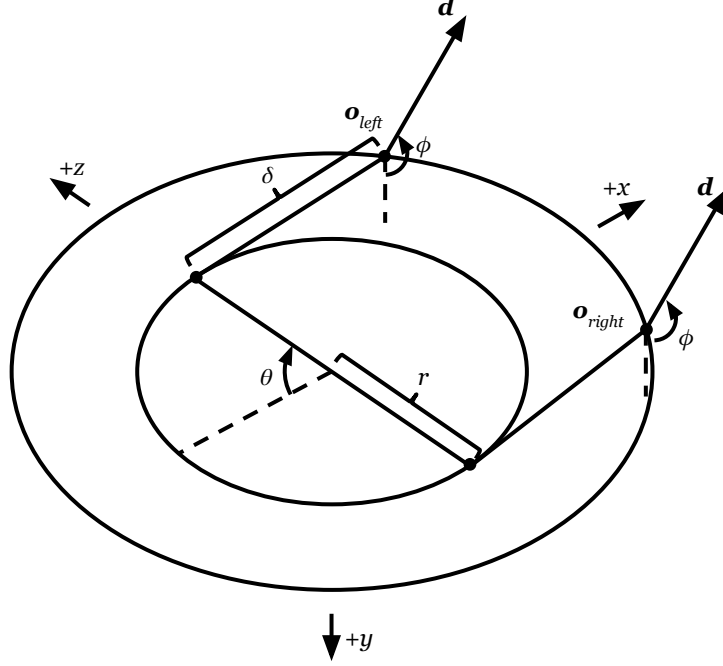


Figure 1: This figure shows the distorted ODS model where ray origins are shifted horizontally by the distance  $\delta$  to  $\mathbf{o}_{\text{left}}$  and  $\mathbf{o}_{\text{right}}$ .

Consider a pixel with position  $[\theta, \phi]$  in an equirectangular panorama, where  $\theta \in [0, 2\pi]$  is the latitude and  $\phi \in [0, \pi]$  is the longitude and  $[0, 0]$  corresponds to the bottom left of the panorama. Then for the two eyes, the respective rays have the same direction

$$\mathbf{d}(\theta, \phi) = \begin{bmatrix} \sin \theta \sin \phi \\ \cos \phi \\ \cos \theta \sin \phi \end{bmatrix}, \quad (12)$$

but different origins:

$$\mathbf{o}_{\text{left}}(\theta, \phi) = \begin{bmatrix} \delta \sin \theta - r \cos \theta \\ 0 \\ \delta \cos \theta + r \sin \theta \end{bmatrix}, \quad \mathbf{o}_{\text{right}}(\theta, \phi) = \begin{bmatrix} \delta \sin \theta + r \cos \theta \\ 0 \\ \delta \cos \theta - r \sin \theta \end{bmatrix}. \quad (13)$$

where  $r = \text{interpupillary\_distance\_in\_meters}/2$ .

## 6. Depth Maps



Figure 2: Example color and depth frames output from Jump Assembler.

The depth maps produced by the Jump Assembler are in the same format as the stitches, over-under equirectangular in which the left eye is on top. The depth stitches use an inverse depth encoding to minimize quantization artifacts and the metric depth along a ray can be found from the greyscale values (in the range 0 to 1) as follows:

$$\text{depth\_in\_meters} = \frac{\text{minimum\_encodable\_depth\_in\_meters}}{\text{value\_in\_depth\_stitch}} \quad (14)$$

The value of `minimum_encodable_depth_in_meters` can be found in the stitch metadata.

Using the projection described in Section 5 `depth_in_meters` can then be used to generate a 3D point for each pixel in the stitch.

## A. Example Stitch Metadata

An abbreviated example of a sample stitch metadata file is shown below

```
{
  "major_version_number": 1,
  "minor_version_number": 0,
  "rig_type": "Odyssey",
  "rig_id": "ATSaZ63HTME",
  "interpupillary_distance_in_meters": 0.055,
  "missing_bottom_degrees": 32.5,
  "missing_top_degrees": 32.5,
  "minimum_encodable_depth_in_meters": 0.3,
  "distortion_delta_in_meters": 0.13076696830622023,
  "cameras": [ {
    "name": "camera01",
    "position": [ 0, 0, 0.14000000000000001 ],
    "orientation": [ 0.01723743655045681, 0.008937656832199924, 1.5859976030659266 ],
    "projection_type": "fisheye",
    "image_size": [ 2704, 2028 ],
    "principal_point": [ 1329.8280987941459, 1024.8454566245296 ],
    "radial_distortion": [ 0.056819340353988931, 0.00088211808542281063 ],
    "focal_length": 1189.902651384682
  }, {
    "name": "camera02",
    "position": [ 0.053575680531112571, 0, 0.12934313455158017 ],
    "orientation": [ 0.32101163488975037, -0.29629325657275551, 1.5568519044086573 ],
    "projection_type": "fisheye",
    "image_size": [ 2704, 2028 ],
    "principal_point": [ 1346.7324346317939, 1019.0243684750756 ],
    "radial_distortion": [ 0.058365680758322422, 0.00047230884540058597 ],
    "focal_length": 1185.2216166244161
  }, {
    ...
  } ]
}
```

## B. Version History

- Version 1.0 September 14, 2016.