JSON 文件

SteamVR™ Tracking

简介

SteamVR™ 定位系统中的每个定位对象均包含说明其传感器几何结构以及与设备相关的其他重要数据的文件。此文件以 JSON 文件格式编写。尽管许多文件使用 JSON 格式，但定位对象中存储的文件与对象的开发和性能密切相关，因此得名“*JSON 文件*”。JSON 文件最初包含极少的传感器位置和方位数据，但在设计和集成过程中，文件会被扩大以包括 IMU 数据、透镜变形数据和 SteamVR™ 使用的其他元数据。最后，校准例程会根据传感器在特定对象上的具体位置对初始传感器几何值进行优化并重写 JSON 文件，进而为每个定位对象生成一个唯一 JSON 文件。

本文档将介绍 JSON 文件中可能存储的所有变量、变量含义，以及如何指定这些变量。用于介绍设计流程各个步骤的其他文档都引用过本文档。要理解每个变量会在何时、因何以及以何种方式添加至 JSON 文件之间，请遵循**对象设计和集成概述**中所述的过程。

JSON 格式

JSON 是指 JavaScript Object Notation，这是一种轻量级数据交换格式。数据交换格式是数据格式化方式，从而能够让不同的计算平台之间以及平台与（偶尔）人类阅读者之间共享数据。JSON 格式的完整说明位于 www.json.org。由于 JSON 文件表示一个 JSON 对象，因此以大括号 { 开始，并以大括号 } 结束。JSON 文件中存储的不同数值以值对方式存储，并使用逗号分隔。每个值对均通过字符串识别，冒号将该字符串与其值分隔开来，如“name”: value。定位对象 JSON 文件的有效对象成员如下所述，文档最后是一个完整的 JSON 文件示例。

> **提示：**Notepad++ 等免费源代码编辑器均能够理解 JSON 格式，并提供有语法高亮显示和代码折叠等便利功能。

JSON 的对象成员

“manufacturer”

字符串值，用于表示制造商公司名称。

示例：

```
"manufacturer" : "Valve"
```

“model_number”

字符串值，用于指示制造商所指定的对象型号。

示例：

```
"model_number" : "REF-HMD"
```

“device_class”

字符串值，可设置为“hmd”或“controller”。

将 device_class 设置为“controller”是告诉 SteamVR™ 在 VR 中渲染对象。将值设置为“hmd”是告诉 SteamVR™ 将定位对象与双目显示屏关联，并使用对象姿势作为虚拟现实中的视点。

示例：

```
"device_class" : "hmd"
```

“device_vid”

USB 供应商标识号。每个制造商都应从 USB.org 申请 VID。对于原型设计，请使用如下所示的 Valve VID。

示例：

```
"device_vid" : 10462
```

“device_pid”

USB 产品标识号。每个供应商应为每个型号创建一个 PID。对于原型设计，请使用如下所示的 PID。

示例：

```
"device_pid" : 8960
```

“device_serial_number”

字符串，用于表示设备唯一序列号。

此值仅可用作将 JSON 文件与实际对象匹配的基准值。每个对象会根据对象处理器中的序列号创建其独有序列号。通过将该序列号写入 JSON 文件，开发人员可将 JSON 文件与实际对象关联。在光学和 IMU 校准后维持此关联至关重要。对象生成的序列号使用 lighthouse_console 显示。

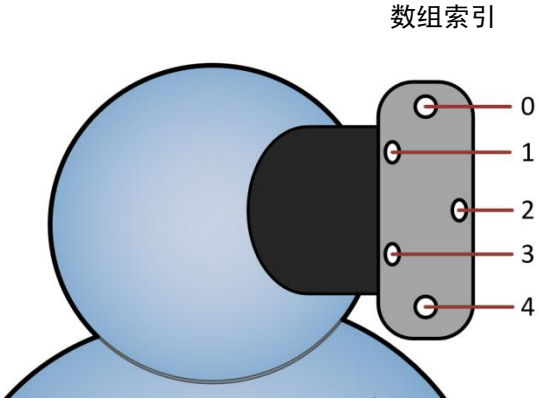
示例：

```
"device_serial_number" : "LHR-F8DE9EBE"
```

“lighthouse_config”

对象，包含三个成员数组。每个数组索引与定位对象上的传感器对应。通过在 modelPoints、modelNormals 和 channelMap 中指定值，将会定义传感器的实际位置、方位及其电气连接（端口号）。当 SteamVR™ 在特定端口收到传感器的定位数据时，SteamVR™ 可以使用 lighthouse_config 中的数据将该信号与定位对象上的具体物理传感器关联。SteamVR™ 还会读取 lighthouse_config 中描述的传感器位置，以针对设备建立精确的传感器几何结构。SteamVR™ 根据已知传感器几何结构处理输入定位数据，以解析对象的当前姿势。

对于对象上的每个传感器，channelMap、modelNormals 和 modelPoints 数组内都会有相应的数据元素。如果我们仅在下图所示的 HMD 对象上放置五个传感器，则每个数组中都会有五个数据项。在该例中，数组索引按如下所示赋值。



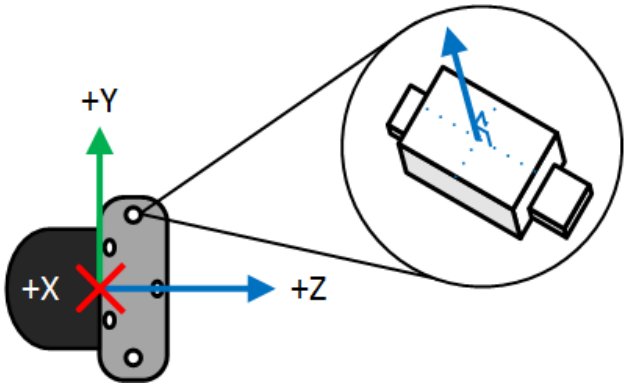
示例：（将省略号替换为下方每个成员中描述的数组。）

```
"lighthouse_config" : {
    "channelMap" : [...],
    "modelNormals" : [...],
    "modelPoints" : [...]
 }
```

“modelNormals”

[x, y, z] 单位向量的数组，用于在对象坐标系内指定光学传感器方向。

每个光电二极管均位于对象上朝外的表面上。假设下图中的传感器在 -X 与 +Y 轴之间的 45°角处朝外。



则与光电二极管所在面相互垂直的单位向量所在方位为与 +X 轴呈 135° 角，与 +Y 轴呈 45° 角。因此，指明该方位的单位向量应为 [-0.7071, 0.7071, 0.0]。如果我们在 45°的面上放置了其他传感器，可能会出现以下法线。

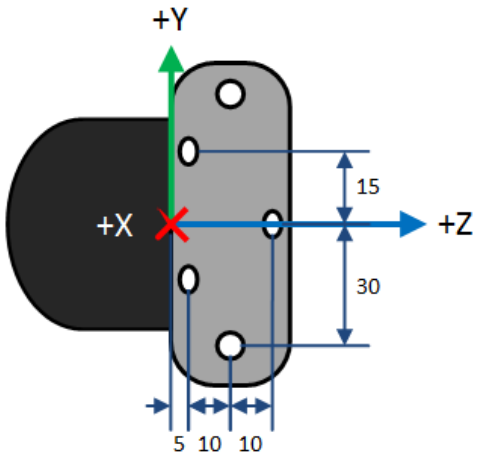
示例：

```
"modelNormals" : [
          [ -0.7071, 0.7071, 0.0 ],
          [ -0.7071, 0.0, -0.7071 ],
          [ -0.7071, 0.0, 0.7071 ],
          [ -0.7071, 0.0, -0.7071 ],
          [ -0.7071, -0.7071, 0.0 ]
      ]
```

“modelPoints”

[x, y, z] 坐标的数组，用于在对象坐标系内指定光学传感器光敏区域的中心位置。坐标值以米为单位。

如果我们向位于 HMD 侧面的传感器位置赋予以下尺寸，我们将需要以 modelPoints 数组形式表示这些尺寸，如下所示。



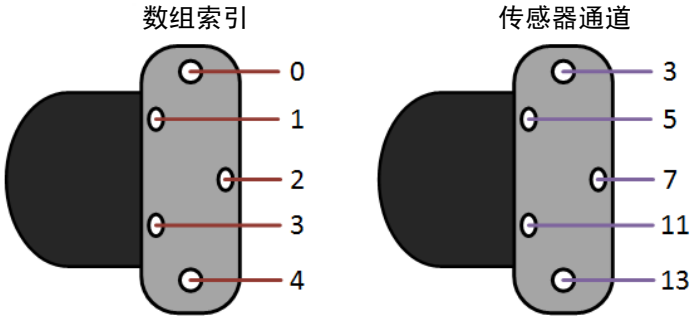
示例：

```
"modelPoints" : [
        [ -0.055, 0.030, 0.015 ],
         [ -0.050, 0.015, 0.005 ],
         [ -0.050, 0.0, 0.025 ],
         [ -0.050, -0.015, 0.005 ],
         [ -0.055, -0.030, 0.015 ]
      ]
```

“channelMap”

端口号数组。对于对象上的每个传感器，该数组中内都有一个相应的元素。数组元素中的值与连接到传感器的电气通道对应。当 SteamVR™ 收到某个通道的定位数据时，SteamVR™ 可以使用该数组将此类数据映射到 modelPoints 中指定的传感器位置。

如果此示例中 HMD 侧面显示的五个传感器均已连接到如下所示的电气通道，我们将需要 channelMap 数组 [3, 5, 7, 11, 13]。传感器通道编号由传感器与对象 PGA 之间的电气连接决定。



注意：通道编号从零开始，并且范围为 0 - 31。Altium 的多通道示意图强制从 1 开始编号。当在 Altium 中通过网络名称复制通道编号时，要特别小心。Net SENSOR_X1 很有可能连接到 FPGA 通道编号 0。从参考编号中减去 1，以获取正确的通道编号。

示例：

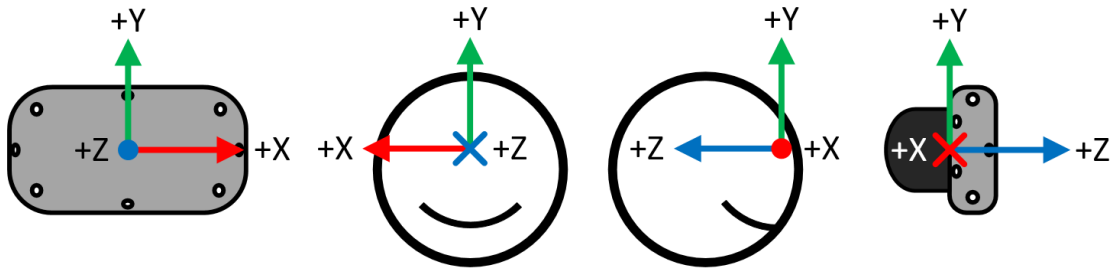
```
"channelMap" : [3, 5, 7, 11, 13]
```

“head”

该对象中所包含的成员用于让定位对象的坐标系与现实世界相适应。Head 成员具有两种不同的含义，具体取决于 device_class 的值。

HMD

当对象作为 HMD 时，head 变量会让 SteamVR™ 的坐标系适应跟踪对象。SteamVR™ 的原点是用户瞳孔之间的某个点，+Y 朝上，+X 朝向用户右侧，-Z 沿视线朝外。如果该模型的坐标系与此不匹配，则需要更改 head 成员。例如，考虑以下 HMD 对象及其坐标系。



Head 的 +X 轴指向 HMD 的 -X 轴方向。要纠正此问题，需要 “plus_x” 的值为 [-1, 0, 0]。

同样地，head 坐标系的 +Z 轴指向 HMD 坐标系的 -Z 轴方向。这就需要 “plus_z” 的值为 [0, 0, -1]，从而与方位匹配。

头部 (head) 的原点位于瞳孔中间，但 HMD 的原点位于 HMD 透镜之间。佩带 HMD 后，头部原点将位于 HMD 原点之后 20 mm 处（沿 -Z 轴方向）。为了让这两个原点匹配，“位置”坐标值需为 [0.0, 0.0, -0.020]。

示例：

```
"head" : {
   "plus_x" : [ -1, 0, 0 ],
   "plus_z" : [ 0, 0, -1 ],
   "position" : [ 0.0, 0.0, -0.020 ]
}
```

控制器

当对象作为控制器时，head 变量会让 SteamVR™ 中显示的渲染模型适应定位对象。渲染模型是根据**渲染模型**中所述方式，在 SteamVR™ 坐标系中创建。确定 head 变量的一种方法是以 STL 文件导出渲染模型，将该 STL 文件导入对象的 3D CAD 空间，并让渲染模型与对象对齐。然后，从对象原点测量渲染模型的 plus_x 和 plus_z 法线和原点。

示例：

```
"head" : {
   "plus_x": [1, 0, 0],
   "plus_z": [0, 0.05233595624, 0.99862953475],
   "position": [0, 0.015, -0.040]
}
```

“plus_x”

HMD：单位向量 [x, y, z]，用于在对象坐标系中表示 head 的 +X 轴。

控制器：单位向量 [x, y, z]，用于让 SteamVR™ 中的渲染模型与对象的坐标系对齐。

“plus_z”

HMD：单位向量 [x, y, z]，用于在对象坐标系中表示 head 的 +Z 轴方向。

控制器：单位向量 [x, y, z]，用于让 SteamVR™ 中的渲染模型与对象的坐标系对齐。

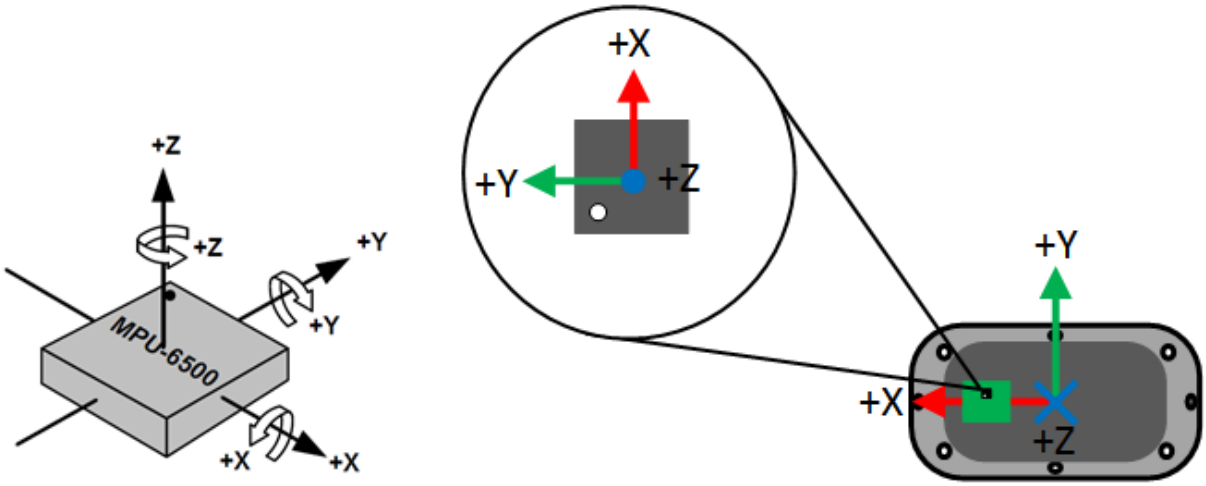
“position”

HMD：坐标 [x, y, z]，用于在对象坐标系内指定用户双眼之间的中心点。

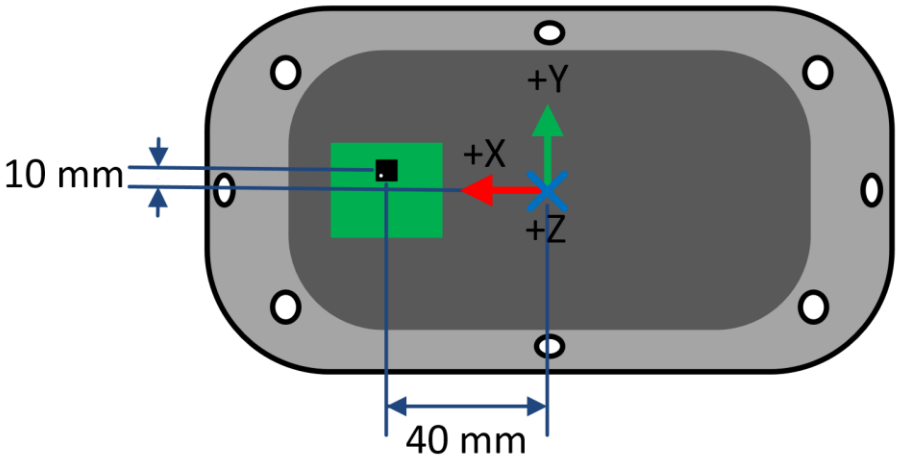
控制器：坐标 [x, y, z]，用于在对象的坐标系内定位 SteamVR™ 中的渲染模型。

“imu”

对象，包含了 IMU 的校准和方向数据。plus_x、plus_z 和位置成员将 IMU 坐标系映射到对象的坐标系中。加速度和陀螺仪成员包含了加速计和陀螺仪校准数据。举例来说，我们看一下这个位于 HMD 对象印刷电路板上的 IMU。



数据表指明了该 IMU 的坐标系。但是，该坐标系未与 HMD 的坐标系对齐。实际上，所有这些轴都需要调整。IMU 的 +X 轴指向 HMD 的 +Y 方向，要求 plus_x 值为 [0, 1, 0]。IMU 的 +Z 轴指向 HMD 的 -Z 方向，要求 plus_z 值为 [0, 0, -1]。此外，IMU 的 Y 轴与 HMD 也不相同，但指定 X 和 Z 轴后，Y 轴即已受到约束。



不仅 IMU 的轴不同，而且 IMU 不位于 HMD 坐标系的原点。IMU 位于 X/Y 平面中，这表示 Z 偏移为 0 mm，但 X 偏量为 -40 mm，Y 偏量为 +10 mm。因此，所需位置值为 [-0.040, 0.010, 0.0]。

加速计和陀螺仪在设备中具有固有偏移。例如，加速计的偏移量最初可能低至 0.025g。然而，在组装过程中和零件寿命周期内，该值可能增加至 0.150g 或更高。陀螺仪的最初偏移量通常为 3°/s - 5°/s。加速计和陀螺仪偏移会随温度变化而变化。为此，SteamVR™ 具有内部校准机制，可在使用期间不断纠正误差。为确保 SteamVR™ 可以尽快趋同于实际偏移，JSON 文件的 IMU 数值中带有初始校准数据，用于修正生产时所存在的偏移量。使用 **IMU 校准**中介绍的软件实用程序，可以轻松确定这些字段的正确数值。IMU 校准工具会输出 JSON 片段，以便复制并粘贴到 JSON 文件的 IMU 数值中。

IMU 校准器输出：

```
Calibrating to gravity sphere, radius 9.8066
0.05265 accelerometer fit error (6 sample vectors x 8 subsamples per vector)

"acc_scale" : [ 0.998, 0.9982, 0.9912 ],
"acc_bias" : [ 0.04646, -0.04264, -0.2414 ],
"gyro_scale" : [ 1.0, 1.0, 1.0 ],
"gyro_bias" : [ 0.06343, 0.01029, -0.02168 ],
```

示例：

```
"imu" : {
   "acc_scale" : [ 0.998, 0.9982, 0.9912 ],
   "acc_bias" : [ 0.04646, -0.04264, -0.2414 ],
   "gyro_scale" : [ 1.0, 1.0, 1.0 ],
   "gyro_bias" : [ 0.06343, 0.01029, -0.02168 ],
   "plus_x" : [ 0, 1, 0 ],
   "plus_z" : [ 0, 0, -1 ],
   "position" : [ -0.040, 0.010, 0.0 ]
   }
```

“plus_x”

单位向量 [x, y, z]，用于在对象坐标系中表示 IMU 的 +X 轴方向。

“plus_z”

单位向量 [x, y, z]，用于在对象坐标系中表示 IMU 的 +Z 轴方向。

“position”

坐标 [x, y, z]，用于在对象坐标系内指定 IMU 封装的中心。

“render_model”

里面包含一个字符串值，指明了在 SteamVR™“rendermodels”文件夹内，用于存放对象默认渲染模型的子文件夹名称。

示例：

```
"render_model" : "ref_controller"
```

“display_edid”

描述

示例：

```
"display_edid" : [ "", "" ]
```

“direct_mode_edid_vid”

用于显示 EDID 供应商 ID 的整型数。要让您的显示屏能够在直接模式下运行，此数值必须位于 NVIDIA 的白名单内。之后，此值将告知 SteamVR™ 在将 VR 显示到 HMD 时使用哪个显示屏。请注意：此时会忽略所有文字前缀。

示例：

```
"direct_mode_edid_vid" : xxxxx
```

“direct_mode_edid_pid”

用于显示 EDID 产品 ID 的整型数。要让您的显示屏能够在直接模式下运行，此数值必须位于 NVIDIA 的白名单内。之后，此值将告知 SteamVR™ 在将 VR 显示到 HMD 时使用哪个显示屏。请注意：此时会忽略所有文字前缀。

示例：

```
"direct_mode_edid_vid" : xxxxx
```

“device”

描述

示例：

```
  "device" : {
      "eye_target_height_in_pixels" : 1080,
      "eye_target_width_in_pixels" : 960,
      "first_eye" : "eEYE_LEFT",
      "last_eye" : "eEYE_RIGHT",
      "num_windows" : 1,
      "persistence" : 0.01666999980807304,
      "physical_aspect_x_over_y" : 0.8000000119209290
   }
```

“eye_target_height_in_pixels”

描述

“eye_target_width_in_pixels”

描述

“first_eye”

描述

“last_eye”

描述

“num_windows”

描述

“persistence”

描述

“physical_aspect_x_over_y”

描述

“lens_separation”

描述

示例：

```
"lens_separation" : 0.06230000033974648
```

“tracking_to_eye_transform”

描述

示例：（将省略号替换为下方每个数值中描述的值。）

```
  "tracking_to_eye_transform" : [
      {
         "distortion" : {...},
         "distortion_blue" : {...},
         "distortion_red" : {...},
         "extrinsics" : [...],
         "grow_for_undistort" : 0.0,
         "intrinsics" : [...],
         "undistort_r2_cutoff" : 1.50
      },
      {
         "distortion" : {...},
         "distortion_blue" : {...},
         "distortion_red" : {...},
         "extrinsics" : [...],
         "grow for undistort" : 0.0,
         "intrinsics" : [...],
         "undistort_r2_cutoff" : 1.50
      }
   ],
```

“distortion”、“distortion_blue”、“distortion_red”

描述

示例：

```
"distortion" : {
            "center_x" : 0.0,
            "center_y" : 0.0,
            "coeffs" : [ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ],
```

```
            "type" : "DISTORT_DPOLY3"
         }
```

“extrinsics”

描述

示例：

```
"extrinsics" : [
            [ 1.0, 0.0, 0.0, -0.03115000016987324 ],
            [ 0.0, 1.0, 0.0, 0.0 ],
            [ 0.0, 0.0, 1.0, 0.0 ]
         ]
```

“grow_for_undistort”

描述

示例：

```
"grow_for_undistort" : 0.0
```

“intrinsics”

描述

示例：

```
"intrinsics" : [
            [ 1.250, 0.0, 0.0 ],
            [ 0.0, 1.0, 0.0 ],
            [ 0.0, 0.0, -1.0 ]
         ]
```

“undistort_r2_cutoff”

描述

示例：

```
"undistort_r2_cutoff" : 1.50
```

“type”

字符串值 “Lighthouse_HMD”

此值始终为 “Lighthouse_HMD”，即使将 device_class 设置为控制器也是一样。

示例：

```
"type" : "Lighthouse_HMD"
```

JSON 文件示例


```
{
   "device" : {
      "eye_target_height_in_pixels" : 1080,
      "eye_target_width_in_pixels" : 960,
      "first_eye" : "eEYE_LEFT",
      "last_eye" : "eEYE_RIGHT",
      "num_windows" : 1,
      "persistence" : 0.01666999980807304,
      "physical_aspect_x_over_y" : 0.8000000119209290
   },
   "device_class" : "controller",
   "device_pid" : 8192,
   "device_serial_number" : "LHR-F8DE9EBE",
   "device_vid" : 10462,
   "display_edid" : [ "", "" ],
   "lens_separation" : 0.06230000033974648,
   "lighthouse_config" : {
      "channelMap" : [ 17, 15, 13, 21, 19 ],
      "modelNormals" : [
         [ 0, 0, -1 ],
         [ -0.13309992849826813, 0.11159992963075638, -0.98479938507080078 ],
         [ 0.11159992963075638, 0.13309992849826813, -0.98479938507080078 ],
         [ 0.13309992849826813, -0.11159992963075638, -0.98479938507080078 ],
         [ -0.11159992963075638, -0.13309992849826813, -0.98479938507080078 ]
      ],
      "modelPoints" : [
         [ -0.0015368221793323755, 0.017447538673877716, -0.0040629836730659008 ],
         [ -0.046612702310085297, 0.039085414260625839, 0.011825915426015854 ],
         [ 0.039518974721431732, 0.046799946576356888, 0.011834526434540749 ],
         [ 0.046315468847751617, -0.038777932524681091, 0.011671472340822222 ],
         [ -0.03922756016254425, -0.046778313815593719, 0.011606470681726933 ]
      ]
   },
   "manufacturer" : "",
   "model_number" : "",
   "render_model" : "lighthouse_ufo",
   "revision" : 3,
   "tracking_to_eye_transform" : [
      {
         "distortion" : {
            "center_x" : 0.0,
            "center_y" : 0.0,
            "coeffs" : [ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ],
            "type" : "DISTORT_DPOLY3"
         },
         "distortion_blue" : {
            "center_x" : 0.0,
            "center_y" : 0.0,
            "coeffs" : [ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ],
            "type" : "DISTORT_DPOLY3"
         },
```

```
      "distortion_red" : {
         "center_x" : 0.0,
         "center_y" : 0.0,
         "coeffs" : [ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ],
         "type" : "DISTORT_DPOLY3"
      },
      "extrinsics" : [
         [ 1.0, 0.0, 0.0, 0.03115000016987324 ],
         [ 0.0, 1.0, 0.0, 0.0 ],
         [ 0.0, 0.0, 1.0, 0.0 ]
      ],
      "grow_for_undistort" : 0.0,
      "intrinsics" : [
         [ 1.250, 0.0, 0.0 ],
         [ 0.0, 1.0, 0.0 ],
         [ 0.0, 0.0, -1.0 ]
      ],
      "undistort_r2_cutoff" : 1.50
   },
   {
      "distortion" : {
         "center_x" : 0.0,
         "center_y" : 0.0,
         "coeffs" : [ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ],
         "type" : "DISTORT_DPOLY3"
      },
      "distortion_blue" : {
         "center_x" : 0.0,
         "center_y" : 0.0,
         "coeffs" : [ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ],
         "type" : "DISTORT_DPOLY3"
      },
      "distortion_red" : {
         "center_x" : 0.0,
         "center_y" : 0.0,
         "coeffs" : [ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ],
         "type" : "DISTORT_DPOLY3"
      },
      "extrinsics" : [
         [ 1.0, 0.0, 0.0, -0.03115000016987324 ],
         [ 0.0, 1.0, 0.0, 0.0 ],
         [ 0.0, 0.0, 1.0, 0.0 ]
      ],
      "grow_for_undistort" : 0.0,
      "intrinsics" : [
         [ 1.250, 0.0, 0.0 ],
         [ 0.0, 1.0, 0.0 ],
         [ 0.0, 0.0, -1.0 ]
      ],
      "undistort_r2_cutoff" : 1.50
   }
],
```

```
   "head" : {
      "plus_x" : [ 1, 0, 0 ],
      "plus_z" : [ 0, 0, 1 ],
      "position" : [ 0, 0, 0 ]
   },
   "imu" : {
      "acc_bias" : [ 0, 0, 0 ],
      "acc_scale" : [ 1, 1, 1 ],
      "gyro_bias" : [ 0, 0, 0 ],
      "gyro_scale" : [ 1, 1, 1 ],
      "plus_x" : [ 0, 1, 0 ],
      "plus_z" : [ 1, 0, 0 ],
      "position" : [ 0.0034600000362843275, 0.0013079999480396509,
0.077326998114585876 ]
   },
    "type" : "Lighthouse_HMD"
}
```