SYNETIC AI

Dataset Documentation

V1.2

Nov 19, 2025

Introduction

Synetic believes that better data leads to better vision systems. By rendering photorealistic, physics-accurate, and perfectly annotated datasets, we give engineers and researchers the ability to train models that are more robust, adaptable, and production-ready than those built on limited real-world data.

Each Synetic dataset follows a consistent structure to ensure clarity, reproducibility, and easy integration into training pipelines.

This document explains the organization of Synetic delivered datasets, including directory layouts, metadata files, and job configurations, so teams can make full use of the value our rendered data provides.

Structure

Synetic image deliveries are in the form of one or more zip files. Each of these zip files are referred to as a dataset.

Inside of each dataset are several folders. At the top level is:

/datas

This directory contains custom data formats and folder structures. These may be internal Synetic structures or customer-specific (e.g., customer taxonomy masks).

/metas

This directory contains image metadata in knowledge-based formats, as well as instance segmentation masks.

At the root of this folder are the meta files that document the dataset. More information about these can be found in the meta file documentation section below.

/metas/vals

/meta/trains

These directories contain instance segmentation masks, depth buffers, and other such metadata, depending on customer requirements.

/metas/jsons/

This directory contains image metadata in text format related to various intrinsics and extrinsics for each image (by root name).The idea is that materials and lighting randomization variations share the same metadata, so only one metadata file exists per image group.

/metas/geos/

Contains all GeoTIFFs equivalent to every RGB image in the dataset. These GeoTIFFs can be imported directly into GIS software.

We have verified and validated all GeoTIFFs using Google Earth Pro (Windows Desktop App).

/metas/kmls/

Contains KML files for all RGB and meta images in the dataset. These provide a geospatial reference overlay for visualization in Google Earth or similar GIS tools.

Also verified and validated with Google Earth Pro (Windows Desktop App).

/geoJsons

Contains GeoJSON files mirroring the contents of /kmls. These follow the same naming and structure conventions but use the GeoJSON format for compatibility with web-based mapping tools.

/yolos

This directory contains the standard YOLO-format folder structure.

Synetic Datasets Meta File Documentation

This document provides detailed information about the meta file formats used in Synetic datasets.

Meta Files

There are several meta files included in every Synetic Dataset:

- **Frame Metadata**: `metas/_${zipName}_FramesMetas.csv`
- **Frame Assets**: `metas/_${zipName}_FramesAssets.csv`
- **Job Configuration**: metas/`_${zipName}_JobsConfigs.json`
- **Frame Metadata JSON Files:** `/metas/[trains, vals]/{uuid}_{frameIndex}.json`

Frame Metadata CSV (_${zipName}_FramesMetas.csv)

Purpose

Each unique position of the scene, weather, lighting, actors, etc is one “frame”.

This file contains one row per frame with all of the camera positioning, lighting, and rendering parameters.

Column	Type	Description	Example
waypointIdx	int	Waypoint index for camera path	0
frameIdx	int	Frame index within the dataset	0
idx	int	Additional index	0
lensNames	string	Lens configuration names	“Clear”
weatherName	string	Weather condition name	“Cloudy”
seed	int	Random seed used for generation	12345
run	int	Run number	0
isTrain	int	1 if training frame, 0 if validation	1
actorSyneticIdx	int	Actor index in Synetic	0
stereoIdx	int	Stereo pair index	0
yaw	float	Camera yaw rotation (degrees)	0.0
pitch	float	Camera pitch rotation (degrees)	0.0
roll	float	Camera roll rotation (degrees)	0.0
range	float	Distance to target (units)	1000.0
sunYaw	float	Sun yaw angle (degrees)	180.0
sunPitch	float	Sun pitch angle (degrees)	45.0
materialMax	float	Material randomization max	1.0
material	string	Material name	“Material 1”
cameraLocationXYZ	string	Camera position as “x,y,z”	“100,0,100”
cameraRotationYawPitchRoll	string	Camera rotation as “yaw,pitch,roll”	“0,0,0”

Column	Type	Description	Example
cameraLookAtLocationXYZ	string	Look-at target as “x,y,z”	“0,0,0”
outImageWidth	int	Output image width	3840
outImageHeight	int	Output image height	2160
cameraIntrinsicsFlSwShFxFyCxCy	string	Camera intrinsics as “focal_length,sensor_width,sensor_height,fx,fy,cx,cy”	“35.0,36.0,20.25,976,980,510,470”
uuid	string	Unique identifier for the frame	“uuid-123”

Asset Metadata CSV (_${zipName}_FramesAssets.csv)

Purpose

Each frame is used to export multiple physical assets. This file tracks the association of the frame to the assets.

Contains one row per generated asset file with file paths and metadata.

Column	Type	Description	Example
frameIdx	int	Frame index within the dataset	10
datasetName	string	Dataset name/identifier	"7cc6b714-4fc2-c893-eb0e-63846e4ca2af"
waypointIdx	int	Waypoint index for camera path	0
uuid	string	Unique identifier for the frame	"0d20a908-4a28-2d67-a1c6-3ca8d317cdda"
dataTypeName	string	Type of asset	"ImgPositive"
pathFile	string	Relative path to the asset file	"7cc6b714-4fc2-c893-eb0e-63846e4ca2af/yolos/images/vals/0_0_7cc6b714-4fc2-c893-eb0e-63846e4ca2af_0d20a908-4a28-2d67-a1c6-3ca8d317cdda_0D0_91+0.jpg"

Common Asset Types

Asset Type	Description	Ext	Usage
ImgPositive	Main rendered image	.jpg	Training/validation images
ImgBackground	Background image	.jpg	Background reference
ImgDepthsPng	Depth maps	.png	Depth estimation training
InstancesBufferPng	Instance segmentation buffer	.png	Instance segmentation training
LabelsDet	Detection labels (bounding boxes)	.txt	Object detection training
LabelsSeg	Segmentation labels (pixel masks)	.txt	Semantic segmentation training
LabelsPose	Pose/Keypoint annotations	.txt	Pose and Keypoint training
MetaJson	Frame metadata	.json	Frame-specific configuration

LiDAR Asset Types

(Included when LiDAR sensor simulation is included)

Asset Type	Description	Ext	Usage
ImgLidarsInstances	Instance Segmentation Mask for LiDAR array - RGB encoded, each object has unique RGB value	.png	Instance segmentation for LiDAR point clouds
ImgLidarsDepthsNormalized	Human-viewable grayscale depth map	.png	Depth visualization
ImgLidarDepthsRgbPng	Absolute depths in centimeters, RGB encoded (same encoding as ImgDepthsPng)	.png	Depth maps with absolute distance
LidarsDepthsPkl	Absolute depths in millimeters as floating point values (gzipped pickle)	.pkl	Programmatic depth access
LidarsDepthsOusterPkl	Absolute depths using Ouster LiDAR-specific arithmetic (gzipped pickle)	.pkl	Ouster-compatible depth format

LiDAR Data Formats

Overview

When LiDAR sensor simulation is enabled, Synetic datasets include multiple LiDAR-specific data formats for depth, instance segmentation, and point cloud generation.

RGB-Encoded Depth Maps

Both `ImgDepthsPng` and `ImgLidarDepthsRgbPng` use RGB encoding to store absolute depth values:

- Depth values are encoded across RGB channels

- Allows storing high-precision depth in standard image formats
- Same encoding scheme used across all Synetic depth products

Loading LiDAR Depth Data (Python)

Loading Pickle Files

```
import gzip

import pickle

# Load standard LiDAR depths (millimeters)

with gzip.open('path/to/LidarsDepthsPkl.pkl', 'rb') as f:

    depths_mm = pickle.load(f)

# depths_mm is a numpy array with shape (height, width)

# Values are in millimeters as floating point
```

Loading Ouster Format

```
import gzip

import pickle

# Load Ouster-specific format

with gzip.open('path/to/LidarsDepthsOusterPkl.pkl', 'rb') as f:

    depths_ouster = pickle.load(f)

# Depths use Ouster-specific range encoding
```

Ouster LiDAR Arithmetic

The `LidarsDepthsOusterPkl` files use Ouster-specific range encoding for compatibility with existing Ouster processing pipelines:

```
range_clipped = (range_actual + 0b10) >> 2

range_shown = r > pixel_value_max ? 0 : range_clipped

where pixel_value_max is std::numeric_limits<uint16_t>::max()
```

This matches the native Ouster LiDAR data format.

Instance Segmentation for LiDAR

`ImgLidarsInstances` provides per-object segmentation masks:

- Each object instance has a unique RGB color value
- Corresponds to instance IDs in the frame metadata JSON
- Use to isolate individual objects in point cloud data

Job Configuration (_${zipName}_JobsConfigs.json)

Purpose

The Configuration file contains the complete configuration for the dataset generation, including camera settings, rendering parameters, asset types, and pipeline configuration.

File Structure

Basic Dataset Information

```
{
  "datasetName": "",                    // Dataset display name
  "datasetDescription": "",             // Dataset description
  "systemId": "uuid",                   // Unique system identifier
  "runPctCompleteUrl": "https://...",   // Progress tracking URL
  "pathLevel": "/LevelsStates/...",     // Level state file path
  "moviePipelineQueueName": "Queue",    // Pipeline queue name
  "playsMax": 1                         // Maximum play iterations
}
```

Synetic Jobs Configuration

For internal use

Camera Configuration

```
"cameras": {
  "outImageWidth": 3840,              // Output image width
  "outImageHeight": 2160,              // Output image height
  "sensorWidth": 36.0,                // Sensor width (mm)
  "sensorHeight": 20.25,              // Sensor height (mm)
  "useIntrinsics": false,             // Use camera intrinsics
  "sensorHorizontalOffset": 0.0,      // Horizontal sensor offset
  "sensorVerticalOffset": 0.0,        // Vertical sensor offset
  "aperture": 99999.898438,          // Camera aperture
  "focalLength": 35.0,                // Focal length (mm)
```

```
  "manualFocusDistance": 99999.898438, // Manual focus distance
  "alphaSkew": 0.71,                   // Alpha skew parameter
  "fx": 976,                           // Focal length X
  "fy": 980,                           // Focal length Y
  "cx": 510,                           // Principal point X
  "cy": 470                            // Principal point Y
}
```

Class Names and IDs

```
"classNamesIds": {
  "Bush": {
    "classId": 0,                      // Class identifier
    "isTarget": true                   // Is target class
  },
  "PipeHelper": {
    "classId": 1,
    "isTarget": true
  },
  "default:": {
    "classId": -1,
    "isTarget": false,
    "syneticRgbIdx": [0, 0, 0]         // RGB color indices
  }
}
```

Lens Configurations

```
"lensesConfigs": [
  {
    "lensConfigs": [
      {
        "name": "Clear",               // Lens effect name
        "intensityMin": 0.0,           // Minimum intensity
        "intensityMax": 1.0,           // Maximum intensity
        "sizeMin": 0.0,                // Minimum size
        "sizeMax": 1.0                 // Maximum size
      }
    ]
  }
]
```

Weather Settings

```
"weathersNames": [
  "Cloudy"                             // Available weather conditions
]
```

Animation Configuration

```
"animations": {
  "randomSampleAnimations": 0,         // Random animation samples
```

```
  "classAnimations": []              // Class-specific animations
}
```

Interface and Debug Settings

```
{
  "isAnimationInterface": false,     // Enable animation interface
  "isMapHidden": false,              // Hide map interface
  "randomizeLensPostProcessesDuringInstancesBuffers": false,
  "randomizeOnce": false,            // Randomize once only
  "isOnlyTargetActorVisible": false, // Show only target actors
  "isDebug": false                   // Enable debug mode
}
```

Sun/Lighting Configuration

```
"sun": {
  "useSplines": true,                // Use spline interpolation
  "useOnlySplines": false,           // Use only splines
  "splinesStep": 0.01,              // Spline step size
  "max": 10,                        // Maximum sun count
  "yawMin": 0.0,                    // Minimum yaw angle
  "yawMax": 359.0,                  // Maximum yaw angle
  "pitchMin": 5.0,                  // Minimum pitch angle
  "pitchMax": -80.0                 // Maximum pitch angle
}
```

Material and Occlusion Settings

```
{
  "materialMax": 1,                  // Maximum material count
  "opComputeOcclusions": false,     // Compute occlusions
  "opComputeOcclusionsDebugPngs": false, // Debug occlusion PNGs
  "occlusionBufferWidth": 32,        // Occlusion buffer width
  "occlusionBufferHeight": 32,       // Occlusion buffer height
  "antiAliasingSpatialSampleCount": 3,  // Spatial anti-aliasing
samples
  "antiAliasingTemporalSampleCount": 1   // Temporal anti-aliasing
samples
}
```

Camera Technical Settings

```
{
  "aperture": 99999.9,              // Camera aperture
  "focalLength": 35.0,              // Focal length
  "manualFocusDistance": 99999.9,   // Manual focus distance
  "randomSeed": 0                   // Random seed for
reproducibility
}
```

Target Actors Data

```
"targetsActors0Data": {
  "doOnly0Data": false,              // Process only first data
  "applyRotations": false,           // Apply rotations
  "yawMin": 0.0,                    // Minimum yaw
  "yawMax": 0.0,                    // Maximum yaw
  "yawStep": 1.0,                   // Yaw step size
  "pitchMin": -10.0,                // Minimum pitch
  "pitchMax": 10.0,                 // Maximum pitch
  "pitchStep": 1.0,                 // Pitch step size
  "rollMin": -14.0,                 // Minimum roll
  "rollMax": 14.0,                  // Maximum roll
  "rollStep": 1.0,                  // Roll step size
  "applyMorphs": false,              // Apply morphing
  "morphMin": 0.5,                  // Minimum morph value
  "morphMax": 1.0,                  // Maximum morph value
  "morphStep": 0.1                  // Morph step size
}
```

Waypoint Configuration

```
"waypoints": {
  "valPct": 0.1,                     // Validation percentage
  "targetsActorsMax": 0,             // Maximum target actors
  "waypointsMax": 0,                 // Maximum waypoints
  "shuffle": false,                  // Shuffle waypoints
  "useSplines": true,                // Use spline interpolation
  "useOnlySplines": true,            // Use only splines
  "splinesStep": 0.03,               // Spline step size
  "targetSplines": true,              // Target splines
  "lookAtIsLevel": false,            // Look at is level
  "lookAtPitch": 0.0,                // Look at pitch
  "useLookAtPitch": false,           // Use look at pitch
  "yawMin": 0.0,                     // Minimum yaw
  "yawMax": 360.0,                   // Maximum yaw
  "yawStep": 120.0,                  // Yaw step size
  "pitchMin": 20.0,                  // Minimum pitch
  "pitchMax": 90.0,                  // Maximum pitch
  "pitchStep": 1.0,                  // Pitch step size
  "rollMin": 0.0,                    // Minimum roll
  "rollMax": 0.0,                    // Maximum roll
  "rollStep": 1.0,                   // Roll step size
  "rangeMin": 1000.0,                // Minimum range
  "rangeMax": 1000.0,                // Maximum range
  "rangeStep": 1000.0,               // Range step size
  "randomizeLightsEveryWaypoint": true,
  "randomizeTargetsActorsMaterialsEveryWaypoint": true,
  "randomizeTargetsActorsMorphsEveryWaypoint": true,
  "randomizeTargetsActorsAnimationsEveryWaypoint": false,
```

```
  "randomizeLensPostProcessesEveryWaypoint": false,
  "stereoBaselineDistance": 17.78   // Stereo baseline distance
}
```

Frame Configuration

```
"frames": {
  "previewMax": 0,                   // Maximum preview frames
  "warmupsMax": 10,                  // Maximum warmup frames
  "framesMaxTrains": 1640,           // Maximum training frames
  "framesMaxVals": 182,              // Maximum validation frames
  "framesMax": 0,                    // Maximum total frames
  "randomizeLightsEveryFrame": true, // Randomize lights per frame
  "randomizeTargetsActorsMaterialsEveryFrame": true,
  "useNightVision": false,           // Use night vision
  "nightVisionValue": 1.0,          // Night vision value
  "solidColorProbability": 0.8      // Solid color probability
}
```

Frame Metadata JSON Files

`/metas/[trains,vals]/{uuid}_{frameIndex}.json`

Purpose

Contains additional detailed frame-specific metadata including camera positioning, object instances, and their properties

File Structure

Basic Frame Information

```
{
  "isTrain": true,                              // Whether this frame is
for training (true) or validation (false)
  "targetInstanceSyneticIdx": 15855,            // Target instance index
  "targetDistanceFromCamera": 2247.30615234375 // Distance from
camera to target
}
```

Camera Configuration

```
"cameraLocation": {
  "x": -3101.6630249023438,                     // Camera X position
  "y": 10.000003814697266,                      // Camera Y position
  "z": 457.19998168945312                       // Camera Z position
},
"cameraRotation": {
  "yaw": 2.4314227076160932e-08,                // Camera yaw rotation
  "pitch": -30,                                 // Camera pitch rotation
  "roll": 0                                     // Camera roll rotation
},
"cameraLookAtLocation": {
  "x": -1155.4388068915941,                     // Look-at target X
position
  "y": 10.000004640603432,                      // Look-at target Y
position
  "z": -666.45309448242165                      // Look-at target Z
position
},
"cameraIntrinsics": {
  "focalLength": 177.48225402832031,            // Camera focal length
  "sensorWidth": 36,                            // Sensor width (mm)
  "sensorHeight": 36,                            // Sensor height (mm)
  "aperture": 99999.8984375,                    // Camera aperture
  "fx": 4444,                                   // Focal length X
  "fy": 4444,                                   // Focal length Y
  "cx": 320,                                    // Principal point X
```

```
  "cy": 320                                    // Principal point Y
}
```

Object Instances

```
"instances": {
  "20424": {                                   // Instance ID (string
key)
    "idx": 20424,                              // Instance index
    "volume": 1,                               // Volume/scale factor
    "classId": 8,                              // Class identifier
    "occluded": -1,                            // Occlusion status (-1
= not occluded)
    "distanceFromCamera": 369.618896484375,    // Distance from camera
    "tags": ["CornFlowerHelper"],               // Object tags
    "className": "CornFlowerHelper",           // Class name
    "rgbIdx": {                                // RGB color indices
      "r": 201,
      "g": 79,
      "b": 0
    }
  }
}
```

Data Validation Rules

Frame Metadata

- `frameIdx` must be unique within a `runKey`
- `isTrain` must be 0 or 1
- `yaw`, `pitch`, `roll` are in degrees (-360 to 360)
- `range` must be positive
- `outImageWidth` and `outImageHeight` must be positive integers

Frame Assets

- `runKey` + `frameIdx` + `dataTypeName` + `pathImg` must be unique
- `dataTypeName` must be a valid asset type
- `pathImg` must be a valid relative path
- `frameIdx` must correspond to an existing frame in the metadata

Job Configuration

Required Fields

- `systemId` must be a valid UUID
- `runPctCompleteUrl` must be a valid URL
- `pathLevel` must be a valid file path
- `moviePipelineQueueName` must be a non-empty string

Numeric Constraints

- `outImageWidth` and `outImageHeight` must be positive integers
- `aperture`, `focalLength`, `manualFocusDistance` must be positive numbers
- `classId` values must be unique within `classNamesIds`
- `framesMaxTrains` and `framesMaxVals` must be non-negative integers

Boolean Fields

- All boolean fields must be `true` or `false`
- No string representations like `"true"` or `"false"`

Array Constraints

- `syneticJobs` array must contain at least one job
- `weathersNames` array must contain at least one weather condition

- `lensesConfigs` array must contain at least one lens configuration

Frame Meta JSON files

Required Fields

- `systemId` must be a valid UUID
- `runPctCompleteUrl` must be a valid URL
- `pathLevel` must be a valid file path
- `moviePipelineQueueName` must be a non-empty string

Numeric Constraints

- `outImageWidth` and `outImageHeight` must be positive integers
- `aperture`, `focalLength`, `manualFocusDistance` must be positive numbers
- `classId` values must be unique within `classNamesIds`
- `framesMaxTrains` and `framesMaxVals` must be non-negative integers

Boolean Fields

- All boolean fields must be `true` or `false`
- No string representations like `"true"` or `"false"`

Array Constraints

- `syneticJobs` array must contain at least one job
- `weathersNames` array must contain at least one weather condition
- `lensesConfigs` array must contain at least one lens configuration