ESRI
EXTERNAL

Oriented Imagery Catalog (OIC) Schema

User Documentation | May 31, 2019

380 New York Street
Redlands, California 92373-8100 USA
909 793 2853
info@esri.com
esri.com

This document applies to Oriented Imagery Version 1.1.2 Oriented Imagery and associated tools are currently provided as a prototype and for testing only. The functionality has not been exhaustively tested and is not currently covered under ArcGIS Support. Please address questions or suggestions related to this workflow to GeoNet or to ImageManagementWorkflows@esri.com.

Contents

Overview

We refer to “oriented” imagery as nontraditional mapping imagery. Such imagery may be captured for applications such as streetview from a mobile platform, close-range inspection (e.g. transmission towers, bridge repairs) using drones, under water rovers, or simple “man in the field” smart phone pictures. It also includes many forms of oblique imagery for viewing the different sides of buildings, for example in property assessment or for the inspection of power lines. Such imagery is not typically suitable or desired for orthorectification due to a number of factors, including high oblique orientation (camera field of view partially or completely above the horizon), insufficient metadata, confusing content in the background, and/or subject matter with significant vertical structure. Although the ArcGIS mosaic dataset has robust support for nadir and low oblique imagery, the data model is not suitable for many forms of oriented imagery.

This document defines a schema for describing an oriented imagery catalog (OIC) for use in the ArcGIS Platform. The OIC is defined as a JSON that references point based feature service that defines the camera location and orientation as well as metadata. The design should allow for scaling to many millions of images in a single catalog.

The OIC is used to find appropriate imagery for any location and enable an indication of the image coverage. The imagery should then be displayed and if the accuracy of the orientation is sufficient provide a mapping between ground features and image features as well as the ability to take measurements, superimpose the display of features into the images, or annotate and record features in the image with the location in map space also being recorded.

The range of applications include 2D web maps, 3D scenes, mobile apps, and desktop applications such as ArcGIS Pro.

Oriented Imagery is primarily intended to support applications with the camera aimed near the horizon or oblique views, but can also be used for nadir imagery.

Imagery Requirements

Minimum requirements

This document presumes there exist individual images with metadata describing the approximate camera location as well as orientation, hence the term ‘oriented’ imagery. Additional metadata and more accurate orientation can be included for more advanced functionality. Geotagged imagery is defined as a subset of oriented imagery that has location but no orientation and is supported, but cannot provide any mensuration and can only provide limited navigation capabilities.

Camera support

Oriented Imagery is designed to support the following types of cameras:

- **Frame cameras**. Includes most drone and aerial imagery, typically with a field of view of about 70 degrees or smaller. Most digital cameras and video fall into this category.
- **Wide-angle or fisheye cameras**. Typically with a >70deg field of view.

- **Omnidirectional Cameras**. Panoramic camera, spherical camera, catadioptric camera, fisheye camera.
- **Synthetic Omnidirectional Cameras**. These have similar properties to an Omnidirectional camera, but area typically created by processing the input from multiple image into a virtual image that typically has a predefined direction and distortion.

These images are defined by methods, which include single frame, equirectangular images and cube images.

Image structure

The images may be a single image as in the case of a frame came or they may be made up of multiple image that need to be stitched together or images that have been processed to represent a single stitched image (e.g. equirectangular images and cube images). Various stitching methods are supported.

Image format

A variety of formats are supported. These include simple browser readable JPG/PNG files, as well as formats such as MRF that are more optimized for cloud storage. Additionally, the imagery can be accessed from Tile services or image services. Tile services provide a predefined tiling scheme were each tile can be accessed as a URL. Image Services enable dynamic request for specified AIOs/Resolution and return only the pixels required.

Metadata

Metadata about the imagery is required. This not only includes the location of the imagery but various attributes that define the orientation, view angles and a wide range of attributes about each camera. A generic view of the data structure indicates that each image has a wide range of standardized parameters that define all properties such as location, orientation and image specific metadata. Such metadata is provided as a point-based feature services that enables the image for an AOI to be efficiently queried and key attributes returned. In a typical scenario the feature service is created from a collection of imagery and its attributes. Oriented imagery also provides support for image services sourced from a mosaic dataset of oblique imagery. This enables an image service to be directly referenced in an Oriented imagery application removing the requirement for a separate point-based feature service. Alternatively, an oriented imagery point based feature services can define the image orientation with the imagery being read from an image service by making use of the ICS export capability.

Oriented Imagery Catalogs (OICs)

Data structure

An OIC is defined by a JSON data structure that defines key properties as well as a reference to the image specific metadata. The key properties include not only generic parameters, but also define defaults and variables. The use of the default and variables can significantly reduce the number of attributes required to define a collection of imagery with similar properties. If all the images in a collection of imagery have the same field of view, this can be defined in the OIC JSON file and need not

be defined for each image. Similarly, if part of the URL to the imagery is always the same, this can be defined as a variable to reduce redundancy in the database.

Approximate vs accurate georeferencing

A key principle in the OI design is to enable both approximate and accurate georeferencing. The accuracy of the georeferencing is highly dependent on the calibration of the camera, knowing the camera parameters as well as auxiliary data such as a DTM or depth image. To enable support for a wide range of accuracies while still enabling simple implementation the georeferencing is split into two parts. One set of georeferencing is always required and provides georeferencing accuracy suitable for the determination of the appropriate image location and navigation. This provides approximations to the image2ground and ground2image transforms and is typically not sufficient for any mensuration. An optional second set of georeferencing can define accurate transformation that can be used to define accurate mensuration and associated accuracies.

Security

Security is handled at the OIC and feature service level. This means that the OIC JSON contains all the information required to access the imagery and a link to the service that provides image specific metadata. In a typical implementation the OIC is defined as an item on ArcGIS Online (or portal) and users that have access to this can access the key properties including a reference to the feature (or image service that defines the camera specific data. This feature service may be a direct URL to a service or an item on ArcGIS Online (or portal) such that only users that have access to the service can access the imagery. This enable the inclusion of views or similar into the feature service so that users can only access specific subsets of the whole. It is typically assumed that if the OIC item can be accessed then the imagery can also be accessed. IE the feature service will typically point directly to a URL that enables access to the imagery. It is though possible to include additional authentication into the access of the imagery.

The client application therefore needs access to the OIC Item, the feature service providing metadata as well as the imagery. Access to the pixels can be directly from cloud storage or using a tile service API. One recommended way of providing access to the imagery is to put the data in an obfuscated cloud storage location such that access is not specifically authenticated, but access to the URL is assumed to equivalent to access to the data source. Such a system enables massive volumes to imagery to be accessed while simplify access control.

Licensing

Oriented Imagery is controlled through the use of an OI API that is used both in web and desktop application. This API should limit capability depending on the user's subscription level in ArcGIS Online. Access to imagery and navigation is open. Access to simple mensuration requires viewer access. Access to collection of feature requires Editor. Access to any authoring capability requires Creator.

Third-party integration

It is realized that some third parties have very advanced navigation and mensuration capabilities. In many cases this may be beyond the capabilities of the existing OIC viewer. Oriented imagery was

designed to enable the integration of such applications so long as they conform to a specified API. It is required that the key metadata for each image is defined in an OIC.

Modes

Oriented Imagery has many modes that are controlled by the OIC. This provides great flexibility, while providing simplicity for the end users, but can create complexity for the authoring of OICs and the software development. The following attempts to define the different modes.

The OIC can refer to a feature services or image services as the source of metadata that defines the location and orientation for all images. If define by a feature service, the location of the images for navigation is defined by attributes of the service. The mensuration capabilities are determined based on availability of approx. orientation, more accurate sensor specific orientation or by a 3rd party viewer API.

If the source is an image service:

> Then the navigation, image access and mensuration is defined by the image service.

If the source is a feature service:

> Access to the imagery can be from multiple sources (as defined below) including direct access to imagery stored in web accessible (cloud) storage or tile servers or image service. Note in this mode the geometry is defined by a feature service but the access to the pixels can be through ICS (Image coordinate system) requests to a simple image services, optionally with no georeferencing, that provides access to subsets of the imagery.
>
> As mentioned above the georeferencing of the imagery for navigation purposes is defined in the feature by attributes. These provide only a limited level of accuracy. The attributes can also define a collection of sensor specific parameters that can be used to provide more accurate mensuration.

Third dimension

Determining a relation between image space and ground space requires a definition of three dimensions in each space as well as a transformation between them. A typical image does not have a 3rd dimension so one needs to be defined. This third dimension can be defined for each image in one of two ways:

DEM – Digital elevation model that is defined in map space. For any defined X,Y location the DEM should provide a height. Based on X,Y & Z + the ground to image transform the appropriate pixel in image space can be identified. Having a DEM (and ground 2 image transform) for an image means that for any point on the map the associated image location can be found. Going from image2grounds is more complex and can only be achieved iteratively, while needing to make assumptions on the form of the DEM. For many applications no DEM is known and ground needs to be approximated by the height of the camera above a flat terrain.

Distance Map – For image that are collected simultaneously with a Lidar system or other depth measures the suitable geometry between frames a distance image can be defined. If such an associated image exists then the transform from image space to ground space is simplified to a single equation, but transformation of ground features to image locations is not unique defined and must be solved iteratively. Existence of a distance map is important for more accurate mensuration in complex terrains.

Oriented imagery attempts to support both methods of defining the 3rd dimension.

Integration with artificial intelligence

In addition to providing navigation and mensuration capabilities for oriented imagery, OIC are also designed to be used as the source for applications requiring image-based feature identification. They can be used a catalog to define what imagery should be submitted to AI algorithms but more importantly be used to transform labels and features from image space to ground space for additional spatial processing.

Camera position and orientation

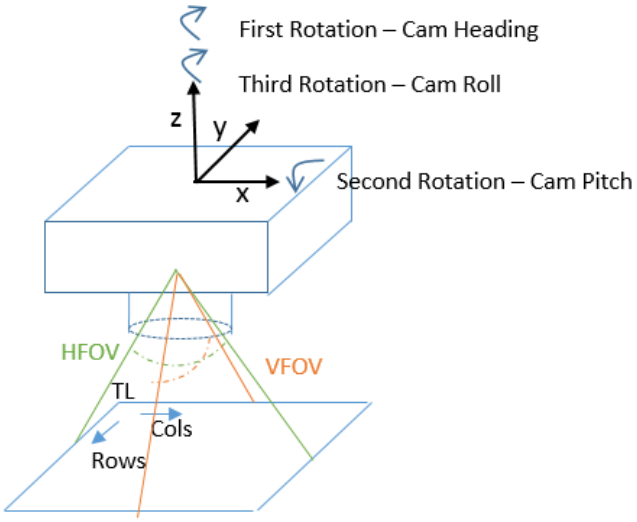
By default the x,y position of the point feature defines the location of the camera in a suitable coordinate system. This may be a projected coordinate system such as UTM & Web Mercator, or geographic coordinates. It is assumed that the Y axis is towards North.
This location is primarily used to aid in the search for the appropriate images to display. As clarified later the location of the camera may also be offset from this location.

If defined, the Z value of the point is assumed to be that of the camera location and may be orthometric or ellipsoid height defined as part of the coordinate system. As will become apparent, the Z value is only used to show the location of the camera in 3D environments and is not necessarily used to define the location of objects in the image or the visible extent of the image.

The camera orientation is described in terms of **CamHeading**, **CamPitch**, and **CamRoll** ("cam" to clearly differentiate from orientation of the aircraft/vehicle body). These angles describe the camera orientation relative to a local projected coordinate system and refer to the point between the camera position and a point running through the center of the image.

Camera orientation is described as follows:

- The initial camera orientation is with the lens aimed at nadir (negative Z axis), with top of the camera (columns of pixels) pointed north, and rows of pixels in the sensor aligned with the X axis of the coordinate system.
- The first rotation (**CamHeading**) is around the Z axis (optical axis of the lens), positive rotations **clockwise** (left hand rule) from north.
- The second rotation (**CamPitch**) is around the X axis of the camera (rows of pixels), positive **counterclockwise** (right hand rule) starting at nadir.
- The final rotation (**CamRoll**) is a second rotation around the Z axis of the camera, positive **clockwise** (left hand rule).



Example orientations:
Camera pointing down with the rows of pixels going from West to East would have orientation: 0,0,0
Rotating the camera by 90degree so that the pixel go from North to south would be: 90,0,0
Rotating the camera to the horizon would have orientation: 90,90,0
Rotating the camera counter clockwise by 20deg would result in an orientation of 90,90,20.

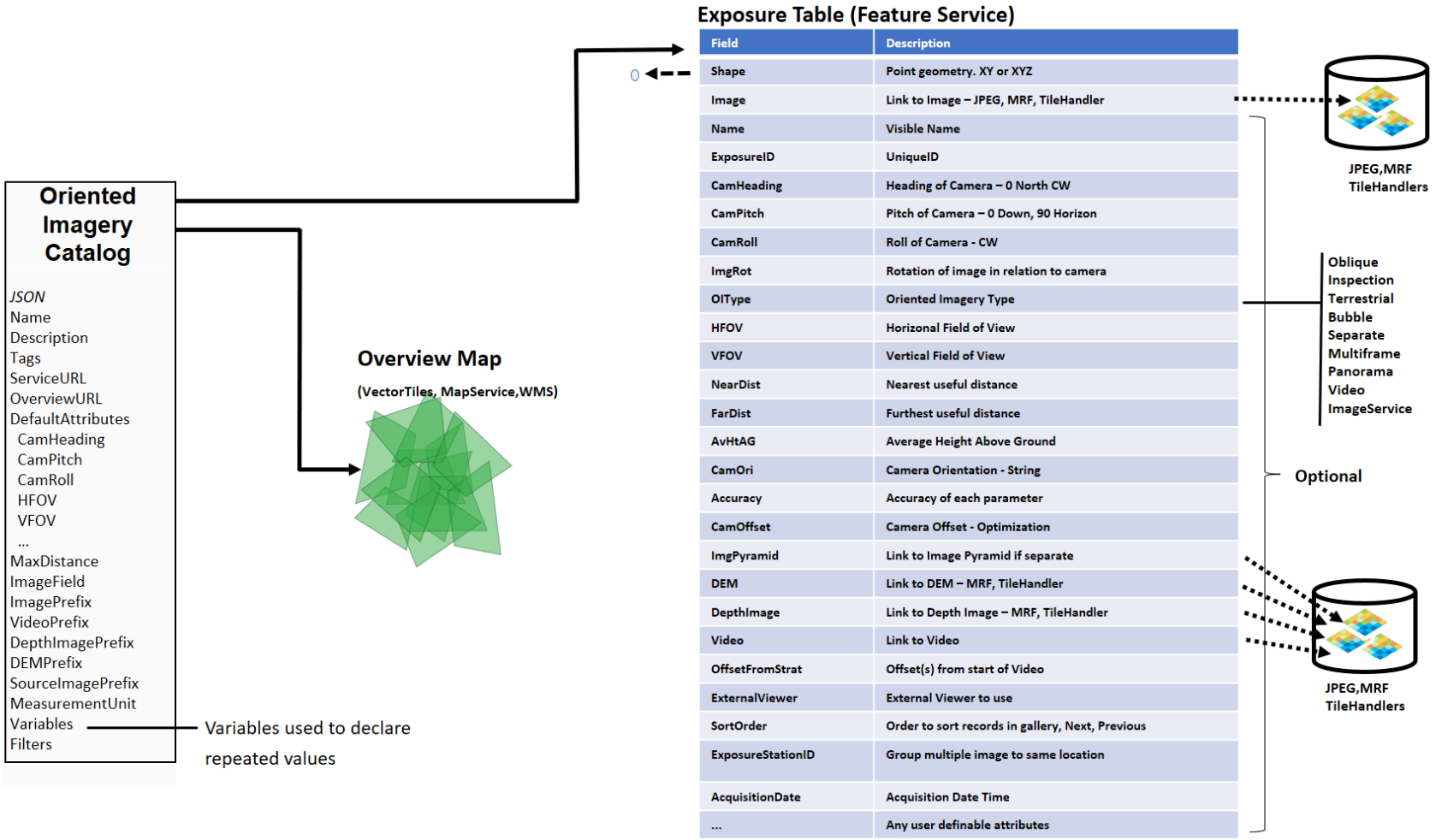
In most applications the Roll angle is 0. The roll is used to indicate that the camera body is rotated around the lens axis and is required to determine the correct pixel to image relation.

In some cases the image is rotated with respect to the camera. This is handled by the ImgRot field. HFOV and VFOV should be that of the camera and should not change based on Roll.

Note the classic photogrammetric definitions for angles as Omega, Phi and Kappa are not used as they are not suitable for imagery directed at the horizon.

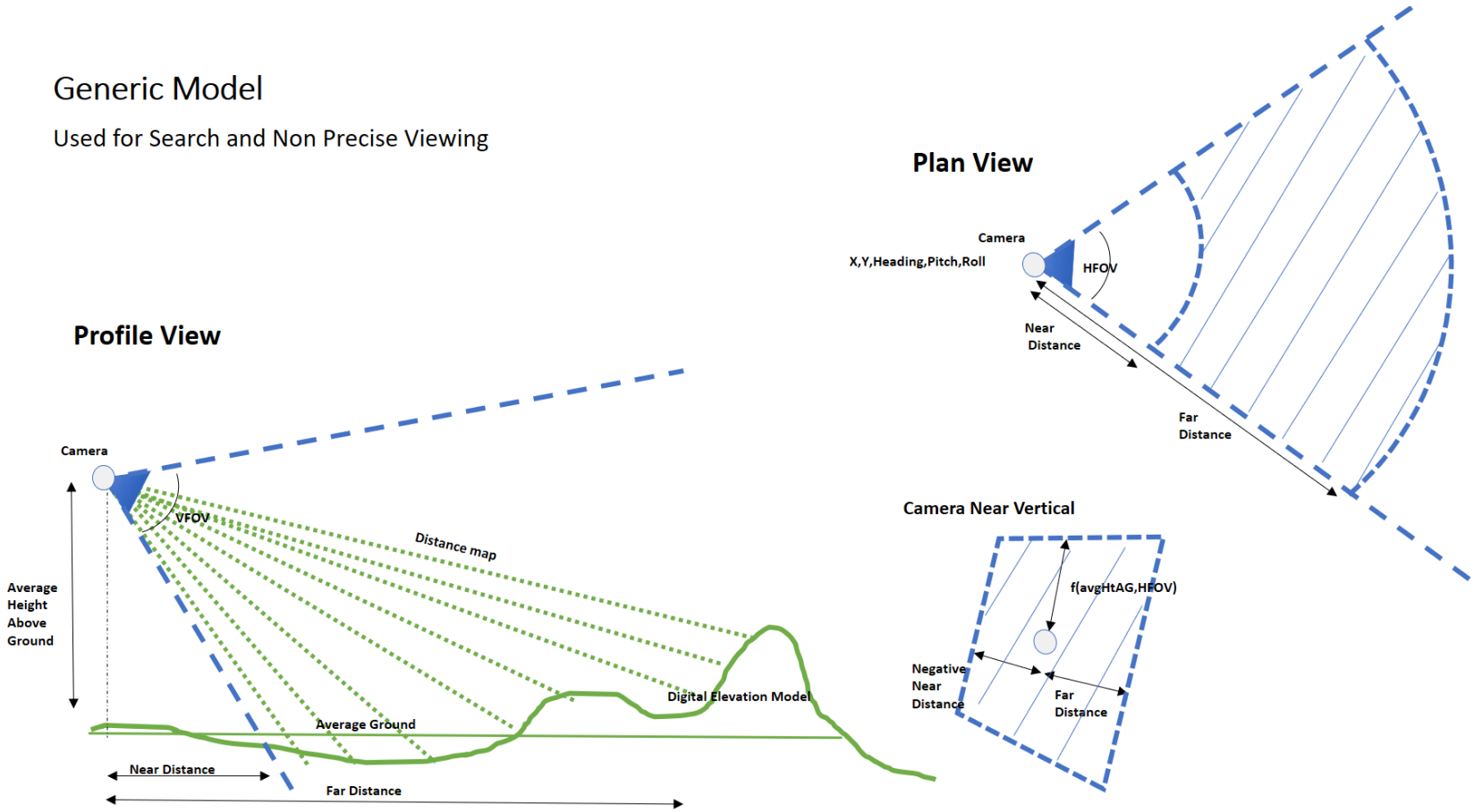
Oriented imagery catalog schema

The structure of the oriented imagery catalog is summarized by the following:


Exposure Table (Feature Service)
Field
Description
Shape
Point geometry. XY or XYZ
Image
Link to Image – JPEG, MRF, TileHandler
Name
Visible Name
ExposureID
UniqueID
CamHeading
Heading of Camera – 0 North CW
CamPitch
Pitch of Camera – 0 Down, 90 Horizon
CamRoll
Roll of Camera - CW
ImgRot
Rotation of image in relation to camera
OIType
Oriented Imagery Type
HFOV
Horizonal Field of View
VFOV
Vertical Field of View
NearDist
Nearest useful distance
FarDist
Furthest useful distance
AvHtAG
Average Height Above Ground
CamOri
Camera Orientation - String
Accuracy
Accuracy of each parameter
CamOffset
Camera Offset - Optimization
ImgPyramid
Link to Image Pyramid if separate
DEM
Link to DEM – MRF, TileHandler
DepthImage
Link to Depth Image – MRF, TileHandler
Video
Link to Video
OffsetFromStrat
Offset(s) from start of Video
ExternalViewer
External Viewer to use
SortOrder
Order to sort records in gallery, Next, Previous
ExposureStationID
Group multiple image to same location
AcquisitionDate
Acquisition Date Time
...
Any user definable attributes
JPEG,MRF
TileHandlers
Oblique
Inspection
Terrestrial
Bubble
Separate
Multiframe
Panorama
Video
ImageService
Optional
JPEG,MRF
TileHandlers
Oriented Imagery Catalog
JSON
Name
Description
Tags
ServiceURL
OverviewURL
DefaultAttributes
CamHeading
CamPitch
CamRoll
HFOV
VFOV
...
MaxDistance
ImageField
ImagePrefix
VideoPrefix
DepthImagePrefix
DEMPrefix
SourceImagePrefix
MeasurementUnit
Variables
Filters
Variables used to declare repeated values
Overview Map
(VectorTiles, MapService,WMS)

Generic Model

Used for Search and Non Precise Viewing



Oriented Imagery Exposure Table

This section defines field names, types, and default values in the Feature Class/Service. Note the primary use for this metadata is an efficient search capability to enable a user to quickly find and display images covering a site of interest and so includes a number of approximations. An optional CamOri field is provided to support precise measurements and can include accurate metadata (e.g. re: focal length, lens distortion, etc.) to enable more accurate measurements. This separation of approximate and precise metadata enables the support of an extensive range of imagery types.

Fieldname <default values in brackets>

- **Shape X,Y** – As above defined location of the camera. See also Dx,Dy below. Optionally may also be X,Y,Z
- **Name** – Optional – Name given to the raster and may be displayed for identification purposes. It may not be unique. It should not include the file extension.
- **ExposureID** - Optional – String Max 20char – String that can be used as a unique identifier for the exposure. This is used primarily capturing details in the image so that the geometry of the pixels can be associated back to a specific image. In the geometry feature the reference to the image is stored as the ImagUrn attribute. This is determined as follows: If the Exposure table (that forms the OIC) has a GlobalID then this should be used. Else if the OIC contains an ExposureID field then this should be used. Else it should be set to:
  right(ServiceName_Name_ObjectID, 20)
  where ServiceName is the service name (excluding \feature\..)
  Note: if Name is undefined then this would be right(ServiceName__ObjectID, 20)
  On the creation of an OIC some sources may already have unique IDs that are used to identify the image and so can put this in the ExposureID.

- **Image** - Reference to the Image. The can be one of following:
  - URL to a web accessible image file (including extension) in cloud storage. Initial supported formats are: JPG or MRF. MRF provides a cloud optimized multi-resolution tiled format for faster access. JPEG may also be used, but may not be optimum especially for larger files. See notes below on providing multi-resolution access.
  - In some managed (database only) implementations Image may refer to a blob in which the image is stored. This may be applicable in offline mobile type applications.
  - May also refer to different services. Currently three service are supported. The URL needs to be prefixed with an identifier that indicates the type of file service and the schema used. The following notation should be used:

    ArcGIS ImageServer
    A|ImageServiceURL|ObjectID|Col|Row - ArcGIS Image Server.
    A – Identifier
    ImageServiceURL – URL to the image services (excluding Export)
    ObjectID – Object ID of the image. The API will construct the URL to enable specified rows/cols to be returned.
    Col – Number of columns (Image width)
    Row – Number of rows (Image height)
    TileService
    T|ZSTS|Col|Row|URL or T|ZSTS|URL
    T – Identifier

ZSTS – Defines scale factor, number of levels and tilesize
Col – Number of columns (Image width) (In case of cube images please specify the width of one face in cube)
Row – Number of rows (Image height)
Eg 42512 defines that there are 4 levels of zoom each with a factor of 2 and the tilesize is 512. Only the following are supported: Number of Levels 1-9; zoom factors 2,3; tile sizes 256,512.
URL is URL to tile service using following notation to define the location of tile
|z| - zoomLevel
|x| - col
|y| - row
|f| or |F| - For bubble images. F denotes the face of the cube. Valid values are f,r,b,l,u,d or F,R,B,L,U,D

Example: Tile url - https://esri.com/Tiles/ImageID/|z|/|x|/|y|.jpg
Api will replace |z|with valid zoom level value, |x| with appropriate column value and |y| with row value.

- **SourceImage** - Optional – Similar to Image but used to point to a local source. This should only exist in the authoring components when imagery is local. Should not be included in the published services. Enables the authoring tools to keep track of local and web accessible versions of the images.
- **AcquisitionDate** – Optional - DATE field. An important attribute for many applications. Also used to support temporal imagery search or filtering. Allows optional recording of date only, or may also include time of day, as required by the application. Note all dates and times should be relative to UTC.
- **CamHeading** <default = -1> Defined above, in degrees. CamHeading = 0 means the top of the camera is oriented toward the North, or CamHeading = 90 means the top of the camera is oriented toward the East. Value should be between 0 and 360. A negative value or NULL indicates that orientation is not known, such as for georeferenced imagery.
- **CamPitch** <default = 90> Defined above, in degrees. CamPitch = 90 means camera is viewing toward the horizon. 0 indicates straight down.
    - The default value is based on the context of "inspection/streetview" (non-nadir-mapping) imagery.
    - Drone imagery may be at a range of oblique angles and if nadir would require a value of 0.
    - For inspection imagery CamPitch > 90 may occur
    - A negative value is allowed, but is equivalent to using CamHeading+180 & CamPitch+180.
- **CamRoll** <0>. Defined above, measured in degrees. CamRoll will typically be near 0. CamRoll is used to account for +/- 90 degree rotations in the camera housing.

- **ImgRot** – Orientation of the image. The orientation of the camera relative to the scene, when the image was captured. Default is <0>. This rotation is added in addition to Roll. Value can be from 0 to 360. Negative values are allowed.

- **HFOV** <60>. Horizontal Field of View in degrees. This typically refers to the FOV for the rows direction of the camera, but may be column direction if the CamRoll is +- 90. The value can be estimated based on the focal length of the camera and CCD width.
    - HFOV ≈ 2*atan(sensor_width/(2*focallength))
- **VFOV** <40>. Horizontal Field of View in degrees. Typically refers to the HOV for the column direction of the image, but may not be if CamRoll is +/- 90. The value can be estimated based on the focal length of the camera and CCD height.
    - VFOV ≈ 2*atan(sensor_height/(2*focallength))
- **AvgHtAG** <1.8>. Average Height (m) above ground of the camera. For aerial or drone imagery, this is equivalent to flying height above ground. For indoor imagery, this is height about the appropriate floor. For inspection imagery, it is the height above the base of the appropriate feature. The value is used to determine the visible extent of the image so large values will result in greater view extent. It is also used to help estimate the location of an object of known relative coordinates in the image. It can be used to enable identification of appropriate image when using multiple images of tall objects. See diagrams for computations later.
- **FarDist** <20>. Furthest usable distance from the camera position of the imagery in meters.
    - This value is used for searching on a 2D map, e.g. click on the map, "Do I have an image that shows this location?" Beyond this distance, the answer may be technically yes, but it's beyond the distance for the image to be useable for detailed inspection or interpretation (a judgment based on the application).
    - For close range applications the value is typically set manually vs. calculated – e.g. for close range inspection, this may be ~5 m; for streetview imagery, perhaps ~20 m; for structural inspection/damage assessment from a drone, ~50 m.
    - If nadir then set FarDist = AvgHtAG*sin(VFOV/2)
    - FarDist should always be >0
- **NearDist** <0>. Nearest usable distance of the imagery in meters. See diagram for how to calculate. For imagery that is near nadir the value can be negative indicating that the x,y location of the camera is covered by the image.
    - If nadir then set NearDist = -AvgHtAG*sin(VFOV/2)
- **OIType** – Optional. Oriented Imagery type. String to define type of imagery.
    - **O** – Oblique – Frame image from aerial or drone sensor
    - **I** – Inspection – Frame image with feature of interest are near camera
    - **T** – Terrestrial (Default) – Typical frame imagery taken at street level
    - **B** – Bubble– A single stitched panorama image with 360view (Equirectangular Image – Photosphere)

- **S** - 6 separate images stitched together to form a cube. Order of images should be [Front, Right, Back, Left, Down, Up] and front face of the cube should have CamHeading = 0; Cubemap image – a cube with 6 faces
  Example
  
- **D** - Multiple frame images taken from the same location pointing in different directions, but considered a group. Order of images should be [Front, Right, Back, Left, Down, Up] and front face of the cube should have CamHeading = 0; Cubemap image – a cube with 6 faces
  Example – Image URL of front face(provided in Image Field) – http://s3.amazonaws.com/sample_F.png
  It should have _F in the end. API will append character at the end of the string to get all the faces of the cube.
  For Front, it will create URL as http://s3.amazonaws.com/sample_F.png,
  For Right, it will create URL as http://s3.amazonaws.com/sample_R.png
  For Back, it will create URL as http://s3.amazonaws.com/sample_B.png
  For Left, it will create URL as http://s3.amazonaws.com/sample_L.png
  For Down, it will create URL as http://s3.amazonaws.com/sample_D.png
  For Up, it will create URL as http://s3.amazonaws.com/sample_U.png
  These URLs should exist in the same location.
  (Note these names may change)
- **P** – Panorama – HFOV >4x VFOV. Equirectangular – photosphere
  
- **V** – Video. Image may or may not exist. Video field should define location of video.
- **A** – ArcGIS Image Service – Defines that mensuration should be achieved use the image server.
  This can be used to change the functionality of the client app.

- **ExternalViewer** – A string value indicating that the image should be open in an external viewer. Value typically exist as a variable in OIC file. If the value is null or does not match any variable define in OIC file, then the image is opened in oriented imagery viewer.
- **Order** – Optional LongInteger. A number used to define the order of the imagery. Is used to enable gallery type applications where users make a selection and go from one image to the next, typically along a street or down a powerline or pipeline. If undefined then AcquisitionDate is used so long as it exists and includes time, else ObjectID.
- **CamOffset** – Optional string of 2 or 3 comma delimited values dx,dy,dz values that define actual camera location relative to the feature point. The use of this is primarily for oblique imagery with longer focal lengths. It enables smaller search distances to be used when identifying suitable image for a location. When determining the extent and positioning of the camera the

applications must add these values to the feature location to determine the camera location. Note that this units are defined in that of the feature service. IE ie the service is feet then this unit is in feet. If the service is in webmercator then this would be in meters appropriate to the webmercator coordinate system (which are not real meters)

- **Accuracy** – Optional comma delimited string of values with the following order. StdDevXY(m), StdDevZ(m), StdDevHeading(deg), StdDevPitch(deg), StdDevRoll(deg),StdDevDistNear(m), StdDevDistFar(m), StdDevElevation(m). Default or 0 defines unknown. These are used to indicate the accuracy of the orientation measurements and used to control how the location cursors are displayed. If undefined the position cursors should indicate very approximate positions. StdDevXY,Z are standard deviation of camera location. StdDevHead,Pitch,Roll are standard deviation of angles. StdDevDistNear,Far are standard deviation of the distance measurements for measurements at the near distance and far distance when using a depth image. It is assumed that measurement accuracy changes linearly with distance. Note if near distance is negative then StdDevDistNear may also be negative.
- **ExposureStationID** – Optional. Long that can be used to associated multiple image take at the same location. This can be used to aid in editing such that if one point is moved all the points with the same ExposureStationID are also moved.
- **ImgPyramids** – Optional. Optimization field providing information on the approx size of the image and the possible availability of reduced-resolution JPG versions that can be used be to enable faster access to thumbprints of the full image. The field is a comma delimited string of integers defining the approx. number of cols (or rows) of the image width and availability of reduced resolution image. The image size is defined as the round (cols|rows / 100). IE an image with 5550 columns would be defined as 56. Subsequent numbers define the availability, approx size and name of reduced resolution versions. The ULR to the reduced resolution versions are assumed to be the same as the full resolution version but with extension replaced by _XXXX.jpg. Eg if the ImageRef was http://a.com/img/D023.jpg and had a size of 5550 columns then a string of the form 55,20,5 would indicate the following image also exist
  http://a.com/img/D023_20.jpg has about 2000 columns or rows
  http://a.com/img/D023_5.jpg has about 500 columns or rows
  The application can now access the appropriate image for display depending on the display window size and zoom ratio.
  Note although MRF files will typically have internal pyramids, such small JPEG files may still be used to increase performance as small JPEG files can be accessed as a single get request.
- **DEM –** Optional.

  ArcGIS ImageServer for Height
  I|PS|ImageServiceURL - ArcGIS Image Server|RenderingRule.
  I – Identifier
  PS- Pixel size in meters -
  ImageServiceURL – URL to the image services (excluding Export)
  Based on the coverage the system should extract a terrain model of defined pixel size covering the possible image extent and use that to define elevation.
  RenderingRule (optional) – Rendering Rule defined on the service.

  ArcGIS Tile ImageServer for Height
  E|ImageServiceUrl – ArcGIS Cached Image Services|LOD
  E – Identifier
  ImageServiceUrl – URL to the cached image service

LOD(optional) – Level of Detail (max LOD Level). Tile will be retrieved from this level or below

MRF|LERC for Height
URL – URL to MRF|LERC file.
Source can be image service, tile cache image service, MRF, or LERC

- **DepthImg** – Optional. If Undefined or 0 then distance should be computed purely from AvgHtAG and camera angles assuming the ground is horizontal and flat.
  The value is used to define the depth dimension. Depending on the application different methods are more appropriate. The following notations are used:
  The default notation is to define a URL to a web accessible image that provides depth information for the pixels. The extent of the image is assumed to be same extent as the primary image although it may have different number of rows and columns. The format of the image is assumed to be MRF.
  It is often possible to define a depth image as a single value (e.g. some inspection imagery) or a plane (e.g oblique imagery of a flat terrain).

  The following notations are used to define different ways of representing plane depth.

  MRF|LERC|PNG for Depth
  URL – URL to MRF|LERC|PNG file.

  ArcGIS ImageServer for Depth
  A|ImageServiceURL|ObjectID - ArcGIS Image Server|Col|Row.
  A – Identifier
  ImageServiceURL – URL to the image services (excluding Export)
  ObjectID – Object ID of the image. The API will construct the URL to enable specified rows/cols to be returned.
  Col – Number of columns (Image width)
  Row – Number of rows (Image height)

  TileService
  T|ZSTS|URL
  T – Identifier
  ZSTS – Defines scale factor, number of levels and tilesize
  Eg 42512 defines that there are 4 levels of zoom each with a factor of 2 and the tilesize is 512. Only the following are supported: Number of Levels 1-9; zoom factors 2,3; tile sizes 256,512.
  URL is URL to tile service using following notation to define the location of tile
  |z| - zoomLevel
  |x| - col
  |y| - row
  |f| or |F| - For bubble images. F denotes the face of the cube. Valid values are f,r,b,l,u,d or F,R,B,L,U,D

  Example: Tile url - https://esri.com/PanoramaTiles/ImageID/|z|/|x|/|y|.jpg

Api will replace |z|with valid zoom level value, |x| with appropriate column value and |y| with row value.

Note such tiles services are assumed to return PNG files that have elevation encoded (See Cyclomedia)

- **CamOri** – Optional. Detailed Camera Orientation, stored as comma delimited string. This field provides support for more accurate measurements, depending on the application.
The first number indicates the type. Over time many different types may exist. Some applications may only support some of the types.

  Type 1 - Heading, Pitch, Roll
  Value ordering:
  1|WKID_H|WKID_V|X|Y|Z|H|P|R|A0|A1|A2|B0|B1|B2|FL|PPX|PPY|K1|K2|K3
  - WKID_H - WKID for Horizontal coordinate system
  - WKID_V - WKID for Vertical coordinate system. Can be undefined, but must be same unit as the WKID_H.
  - X,Y,Z - Camera center coordinate (Perspective point)
  - H,P,R – Heading pitch roll – Define as in key attributes
  - A0 A1 A2 B0 B1 B2 Affine transformation parameters to camera center in the ground to image direction. (IE A0 and B0 are offsets in cols and rows and A1,B2|A2,B1 are 1/pixelsize in microns
  - FL - FocalLength in microns
  - PPX,PPY - PrincipleX,PrincipleX - Principal Point offset in X and Y from camera center in microns
  - K1,K2,K3 - Konrady distortion coefficients in microns

  Type 2 – Omega, Phi, Kappa
  Value ordering:
  2|WKID_H|WKID_V|X|Y|Z|O|P|K|A0|A1|A2|B0|B1|B2|FL|PPX|PPY|K1|K2|K3
  - WKID_H - WKID for Horizontal coordinate system of the point
  - WKID_V - WKID for Vertical coordinate system of the point
  - X,Y,Z - Camera center coordinate (Perspective point)
  - O,P,K – Omega, Phi, Kappa
  - A0 A1 A2 B0 B1 B2 Affine transformation parameters to camera center (IE A0 and B0 are offsets in cols and rows and A1,B2|A2,B1 are 1/pixelsize in microns
  - FL - FocalLength in microns
  - PPAX,PPAY - Principal Point offset in X and Y from camera center in microns
  - K1,K2,K3 - Konrady distortion coefficients in microns

OIC properties

Each OIC is defined by a properties JSON that is referenced as a file or an Item on ArcGIS Online. In the OIC management tools in ArcGIS Pro this file also defines the key properties.

The file should have the extension .oic

Following is the data structure:

```
{
    "type": "OIC",
    "version": "1.0",
    "properties": {
            "Name": "",                 Name to be displayed
            "Description": "",          Short description
            "Tags": "",                 Tags used to aid identification
            "ServiceURL": "",           URL to the Exposure station service or item.
            "OverviewURL": "",          URL to Overview coverage
            "DefaultAttributes": {      Following are default for each field if undefined or NULL
                "CamHeading": "",
                "CamPitch": "",
                "CamRoll": "",
                "HFOV": "",
                "VFOV": "",
                "AvgHtAG": "",
                "FarDist": "",
                "NearDist": "",
                "OIType": "",
                "Order": "",
                "CamOffset": "",
                "Accuracy": "",
                "ImgPyramids": "",
                "DepthImg": "",
                "ExternalViewer":""
            },
               "About": "",
            "ImageField":"",                If defined, then the name of the field to be used
                                            as the Image Field
            "ImagePrefix": "",              Prefix added to Image attribute
            "VideoPrefix": "",              Prefix added to Video attribute
            "DepthImagePrefix": "",         Prefix added to DepthImage attribute
            "DEMPrefix": "",                Prefix added to DEM attribute
            "SourceImagePrefix": "",        Prefix added to Source image
            "MaxDistance": "",              Max search distance to be used
        "MeasurementUnit": "feet",          feet or meter
            "Editor": "",                   Defines if Editor mode should be enabled  True/False

"Credentials":{                             Credentials required to access imagery
    "Username":"",
    "Password":""
              },
 "Variables": {                             Define Variables

            },
  "Filters": {                              Define filters to be enabled

          },
```

```
   "Copyright": {                                    Define copyright/attribute text to display on image
        "text": "",
        "url": ""
                }
                    }
}
```

Notes:
ServiceURL – URL to the feature service that defines the points and attributes.
OverviewURL – URL to the vector tile cache, map service, WMS or WMTS or service item use to provide overview of the image coverage. This is only used as a display of coverage and not queried.
ImagePrefix – Prefix to be added to start of image name. Typically used to define the root URL for the image.
VideoPrefix – Prefix to be added to start of video name. Typically used to define the root URL for the video.
ExternalViewer – Json object containing the wrapperUrl and config. This is a typical example were a variable would be defined. Eg $_cyclo_$
DepthImgPrefix – Prefix to be added to start of depth image name. Typically used to define the root URL for the image.
DEMPrefix – Prefix to be added to start of DEM name. Typically used to define the root URL for the image.
SourceImagePrefix - – Prefix to be added to start of Source image name. Typically used to define the local network or path where the images are stored. Does not exist as part of the service.

Filters – There is often a necessity to subset a OIC into subsets. A typical example is the streetview imagery taken over multiple year and user want so see imagery only for a specific year. Another example may be imagery for a building at different floors and users want to see only imagery for a specific floor. This can be handled by define a set of filters as follows in the OIC JSON

```
{
“filters”:{
“All” : { “”}
“2016” : { “Where Year = 2016”}          //* Replace these with appropriate statements *//
“2014” : { “Where Year = 2014”}
“Old” : { “Where Year<2014”}
}
```

The OIC API will list the defined filters in their respective order and labels.

Variables – The number of records in the feature services can become very large and there are many cases were parameters may need to be changed depending on a user. An example is an access key that may be embedded as part of a URL. This key may need to change over time or be specific to a user. To enable this the concept of variables is used.
Any string attribute value may have one or more variables defined using the notation $_Variable_$. Eg
URL = $_Path_$/MyDirectory/A.JPG?/$_tag_$
As each attribute it read any text between $_ and _$ will be replaced by a variable in the OIC. If no corresponding variable is defined, it is excluded.

Note: Variable should not be used in attributes that are searched, e.g. AcquistionDate.

Variables are defined as JSON as follows:

```
{
“variables”:{
“cyclo” : {
                “wrapperUrl”: “ ”,
                “config”: {
                                “username” :”aad”,
                “password”: “asdasd”,
                “apiKey” : “asdasdasdasdasd” }
        }
“path” :”C:/temp”
          }
}
```

Defaults are value to be used in place of attributes fields. Each image is initiated with these default values and then the defaults are replaced by any value in the attribute table.

This provides a much more compact method of defining OIC as only the values that vary from camera to camera need to be defined in the attribute table.

Note that values such as AcquisitonDate do not have defaults as they are used as query fields.

These properties need to be refined. Names will be refined based on ArcGIS Online naming conventions.