Auswärtiges Amt

DAAD Deutscher Akademischer Austauschdienst
German Academic Exchange Service

Geo-IT Online Seminar
Freie Universität Berlin
Institute of Geographical Sciences

Freie Universität Berlin

TRAINING ON GOOGLE EARTH ENGINE

MODULE 5 : IMAGE COLLECTION in Google Earth Engine

Prof. Dr. Eng. Ayman Abdulrahman

Coordinated from

Dr. Wahib Sahwan (Freie Universität Berlin)

Geo-IT

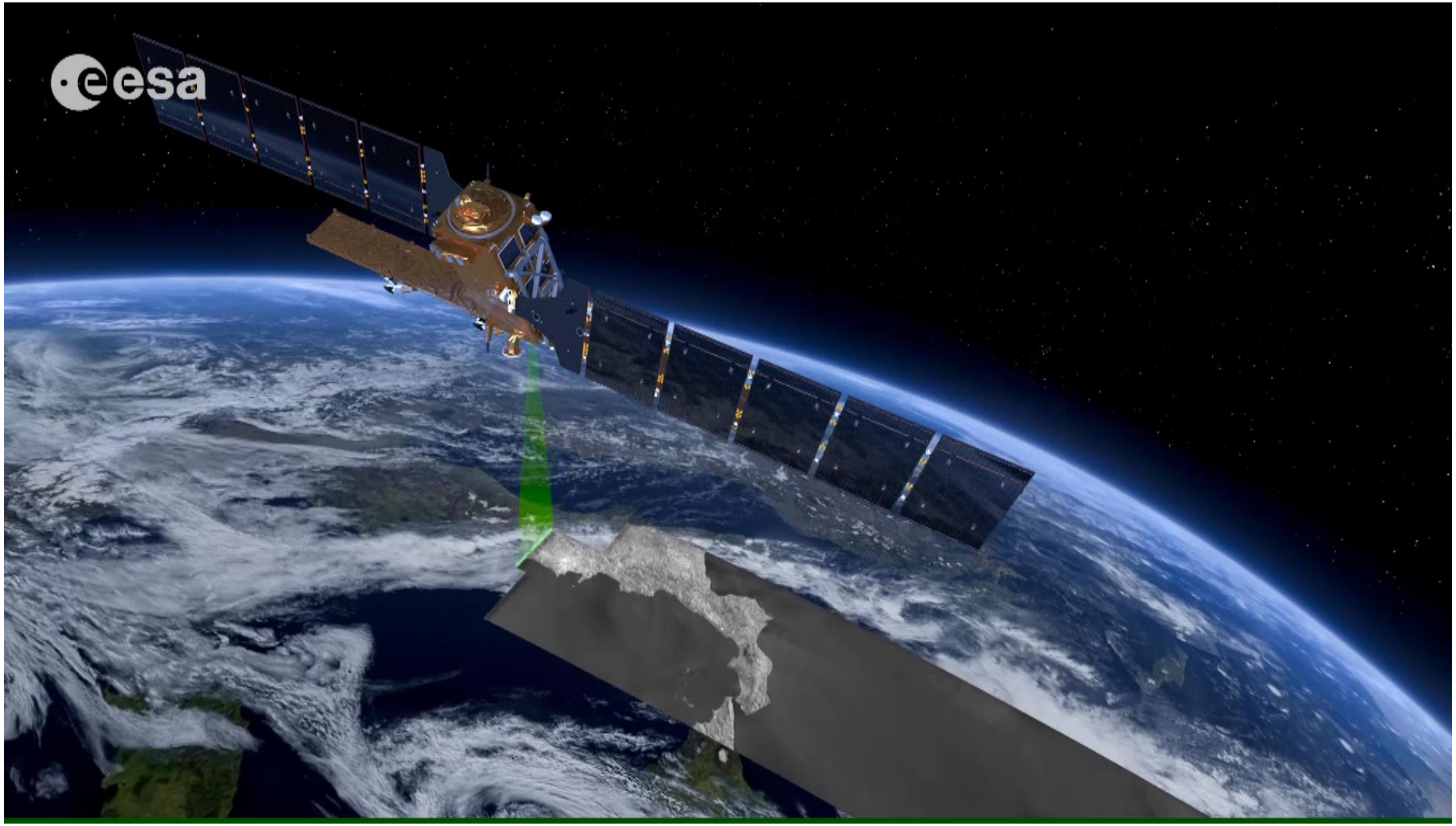
The Technology of Data Acquisition for Sustainable Development and Crisis Management

(Germany, Jordan, Lebanon and Syria)

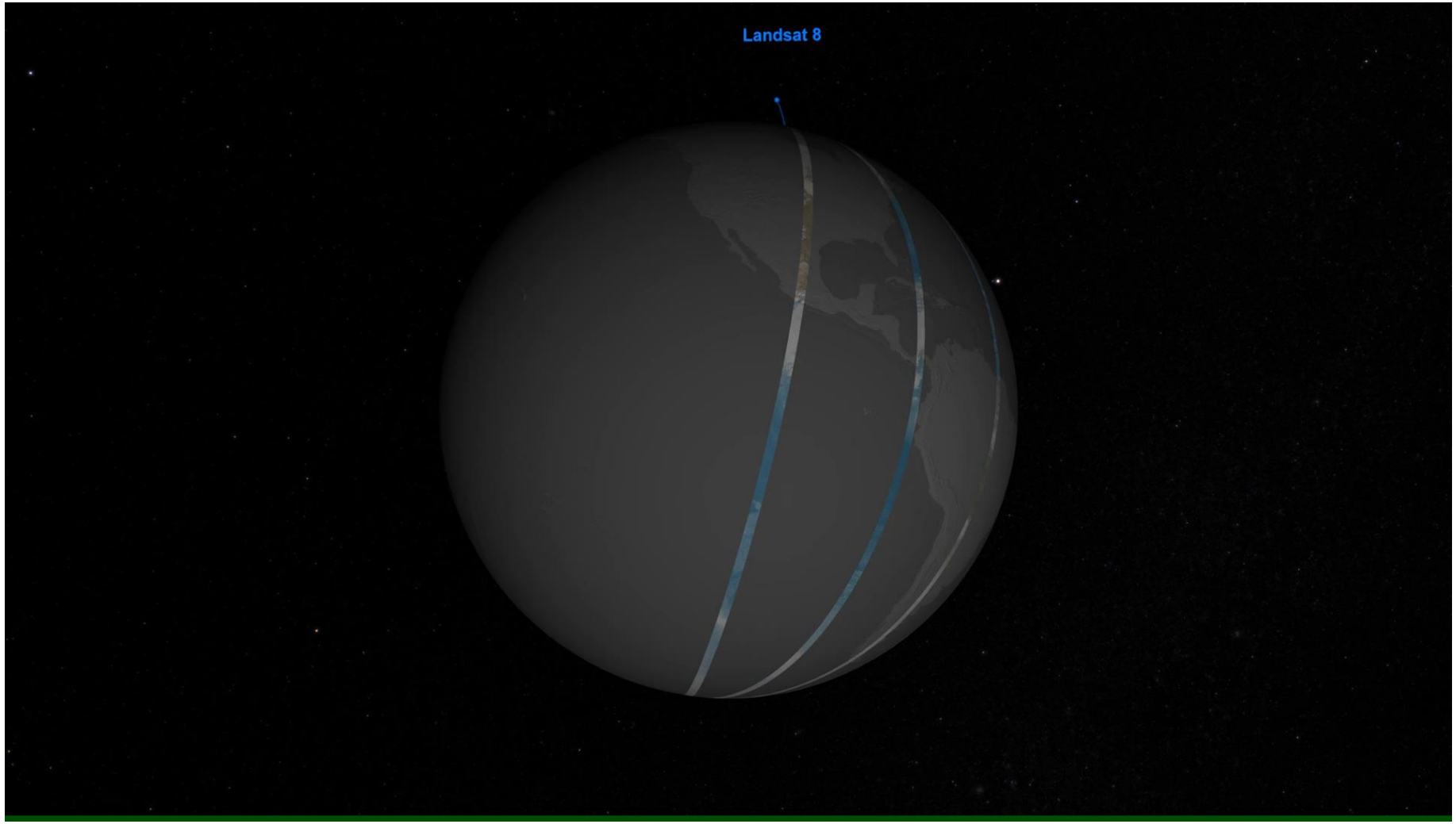
MODULE 5 : IMAGE COLLECTION in Google Earth Engine

- Import Image Collection
- Processing
- Filtering
- Mosaic
- Masking
- Reducing
- Exporting
- Examples and exercises

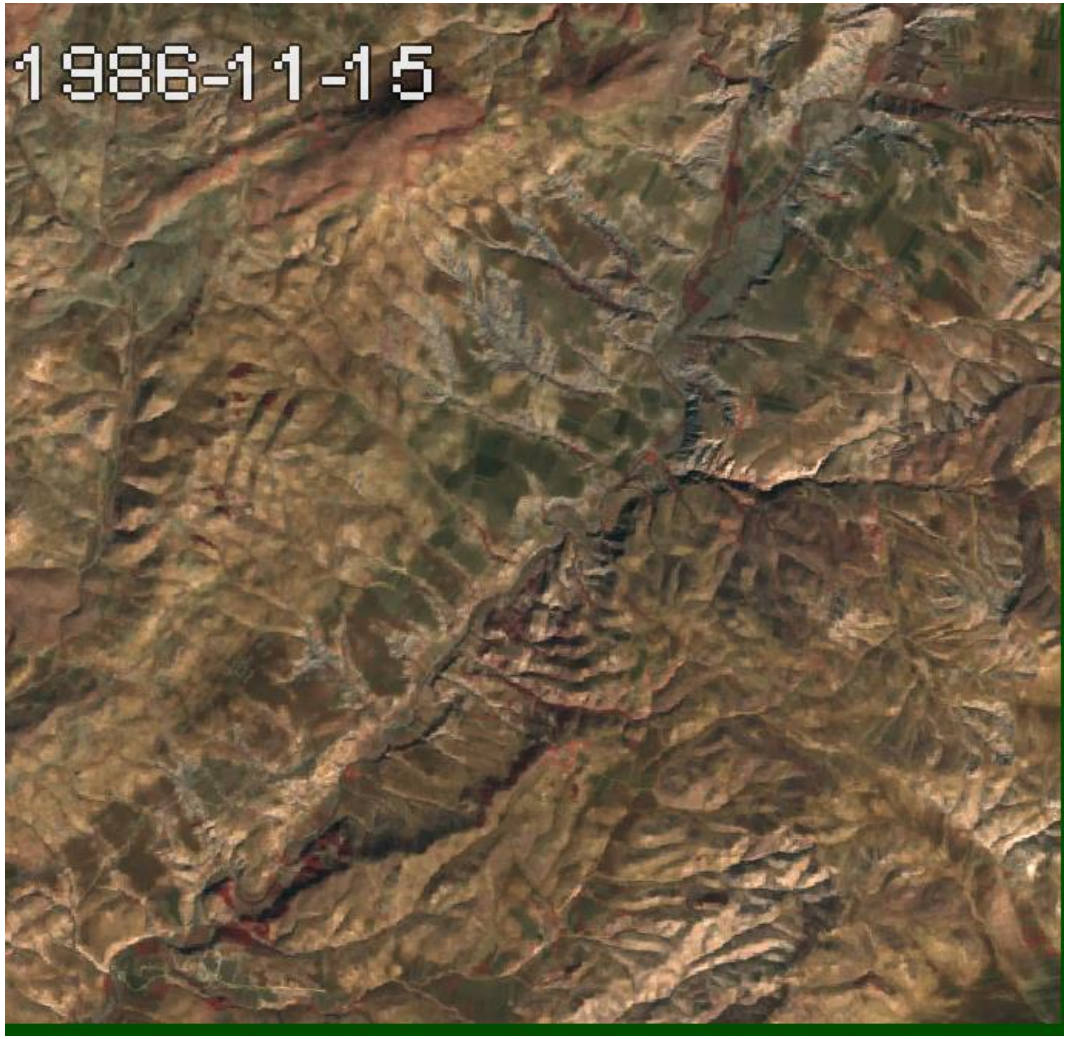
ImageCollection



- **Landsat Sentinel 2**

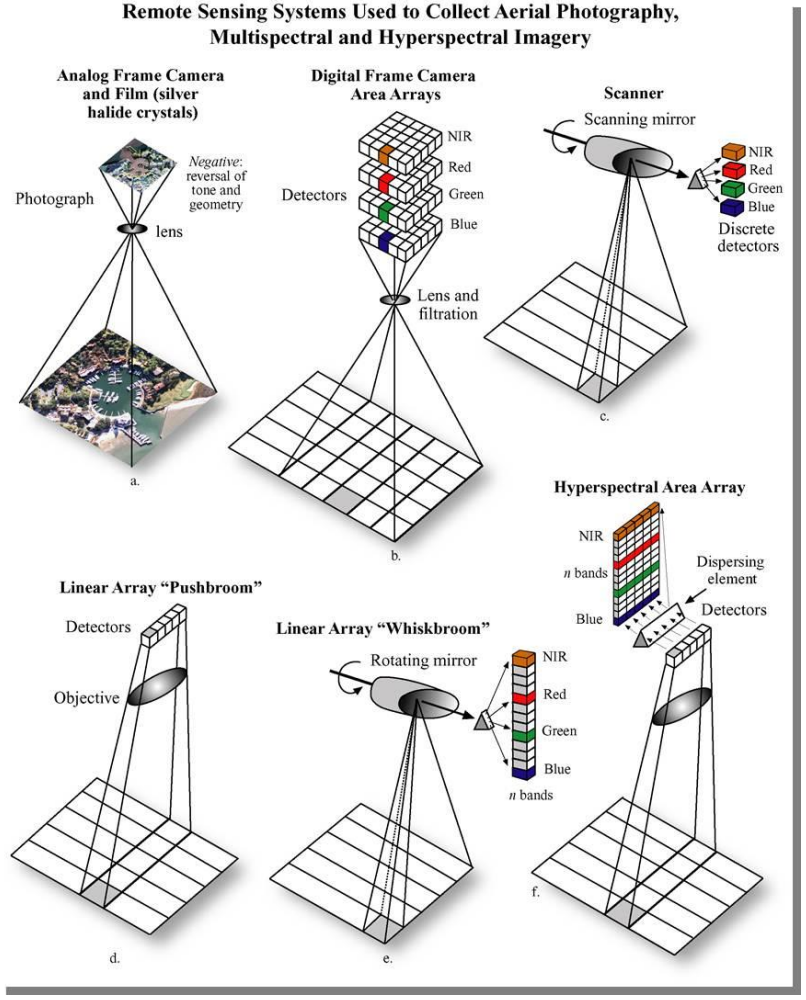


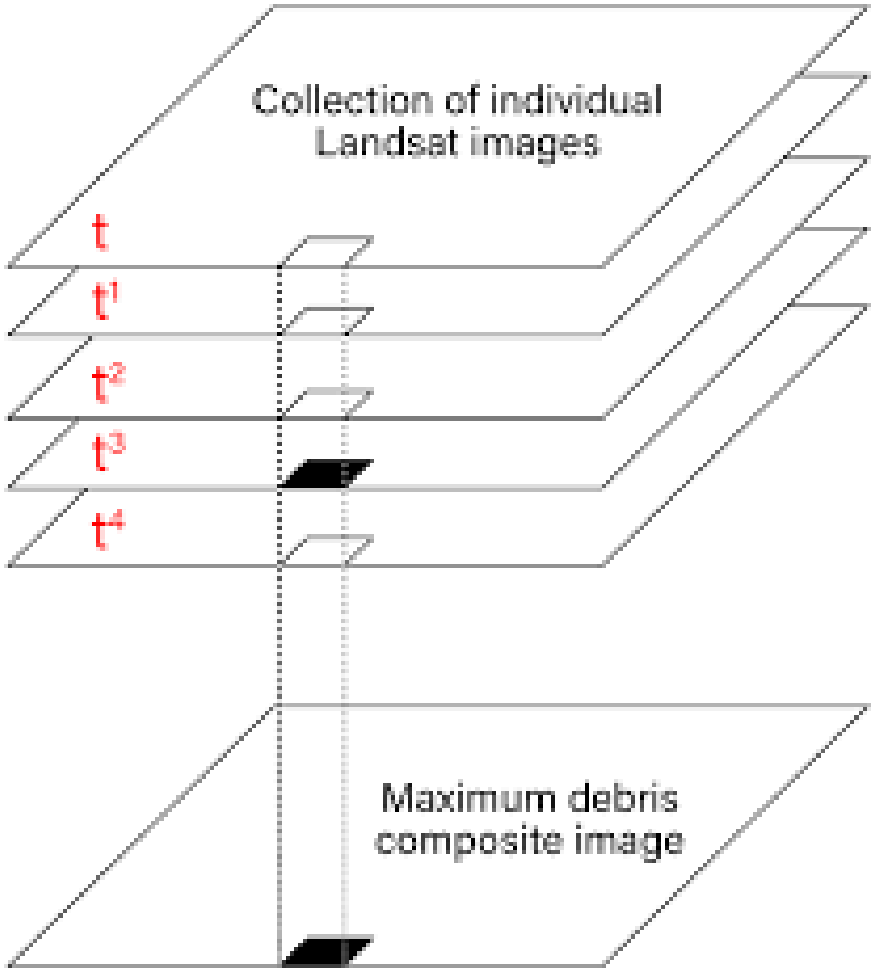
- **Landsat 5 (Midanky Lake)**

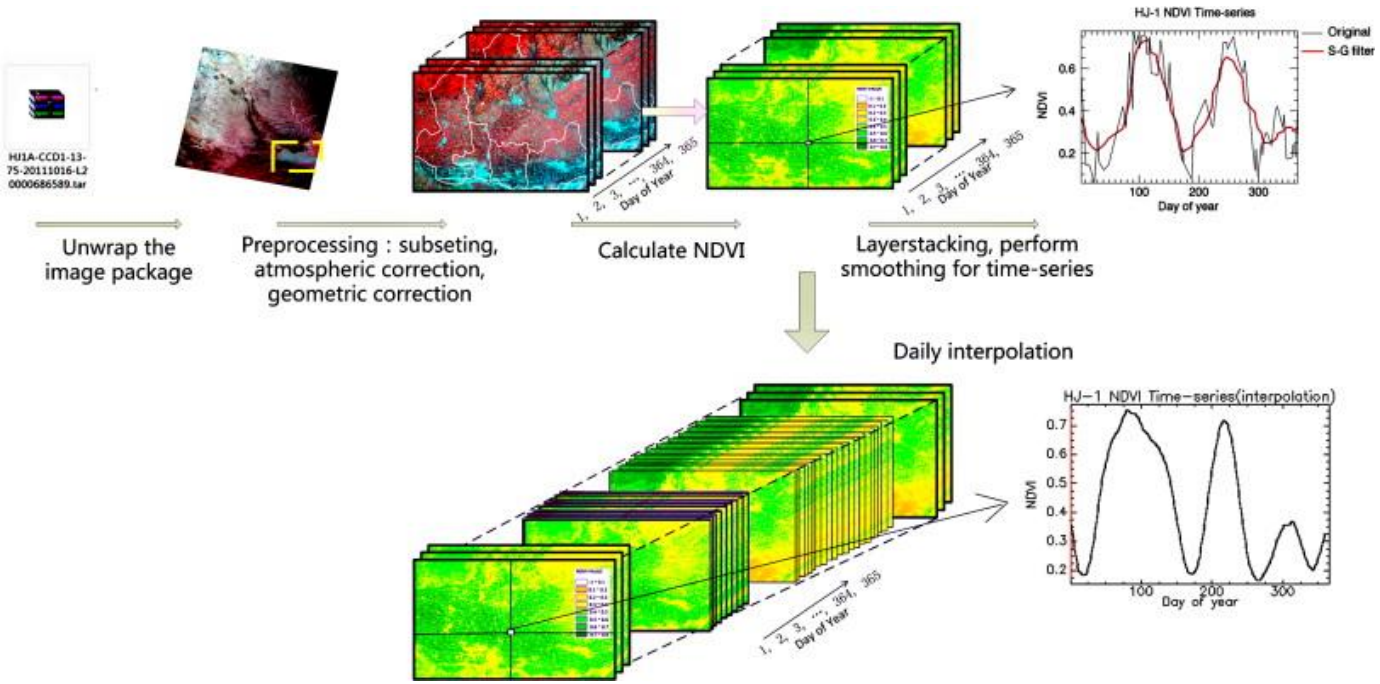


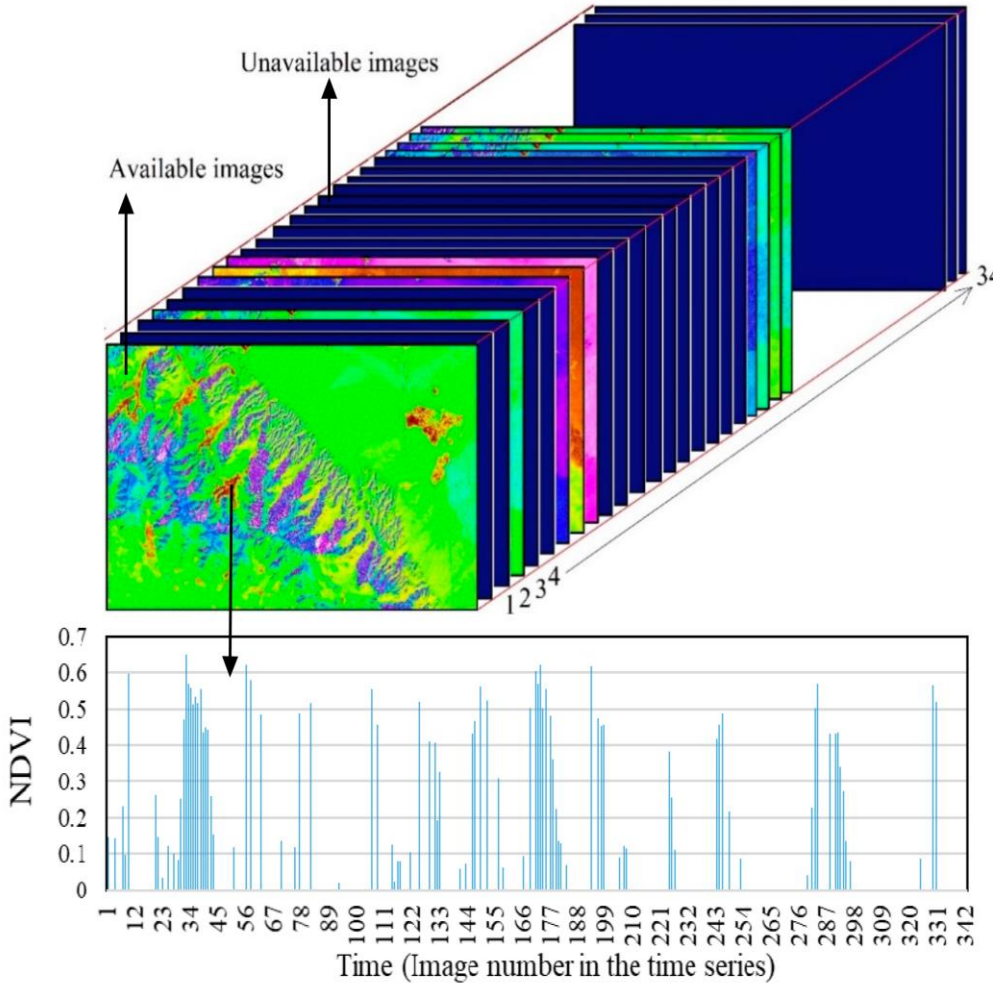
- **Landsat 8 (Midanky Lake)**

ImageCollection

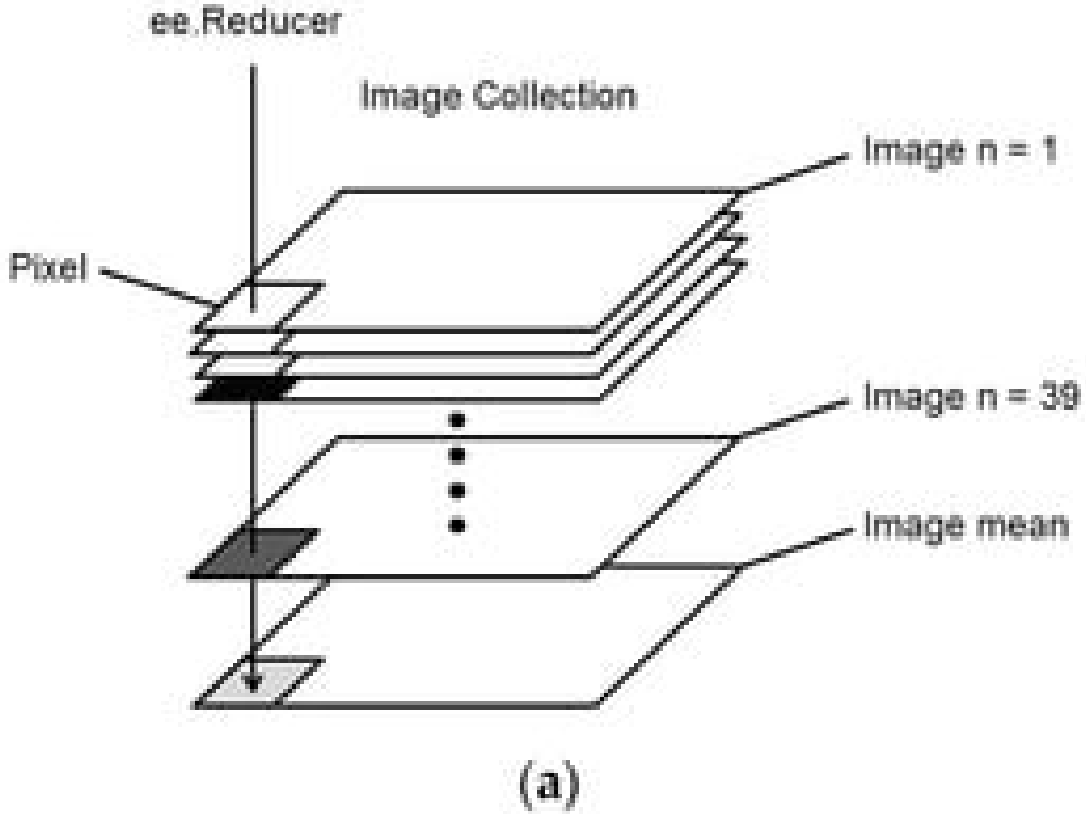




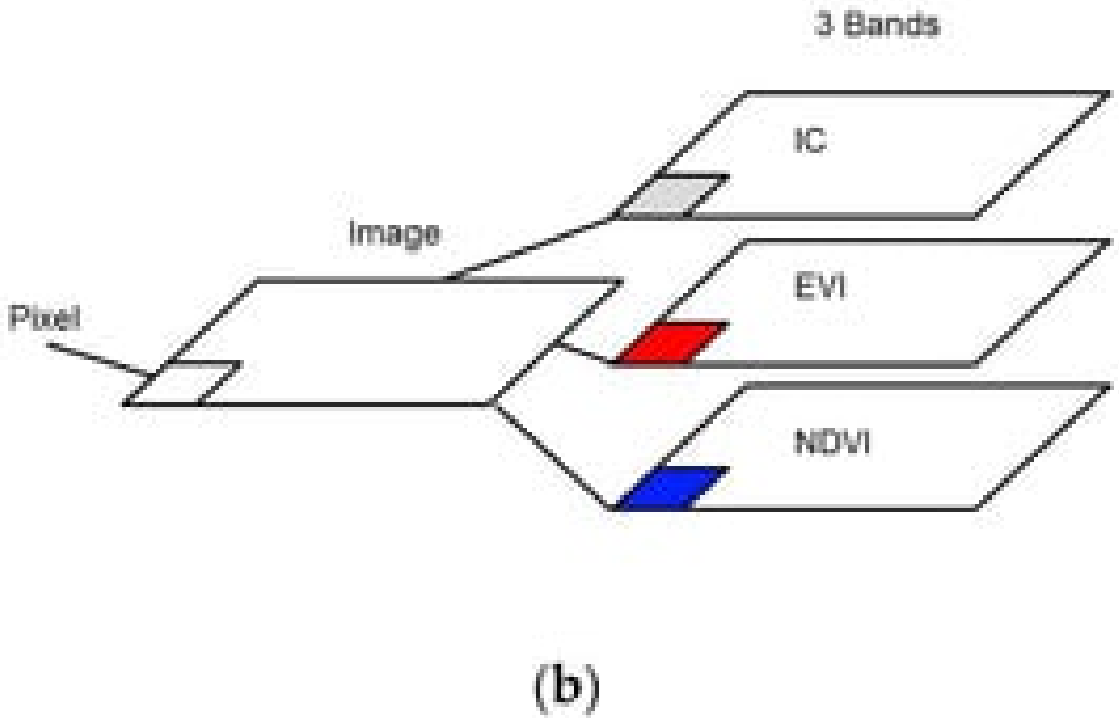

HJ1A-CCD1-13-75-20111016-L2-0000686589.tar
Unwrap the image package
Preprocessing : subseting, atmospheric correction, geometric correction
Calculate NDVI
Layerstacking, perform smoothing for time-series
HJ-1 NDVI Time-series
Original
S-G filter
NDVI
Day of year
Daily interpolation
HJ-1 NDVI Time-series(interpolation)


Unavailable images
Available images
345
1 2 3 4
NDVI
Time (Image number in the time series)

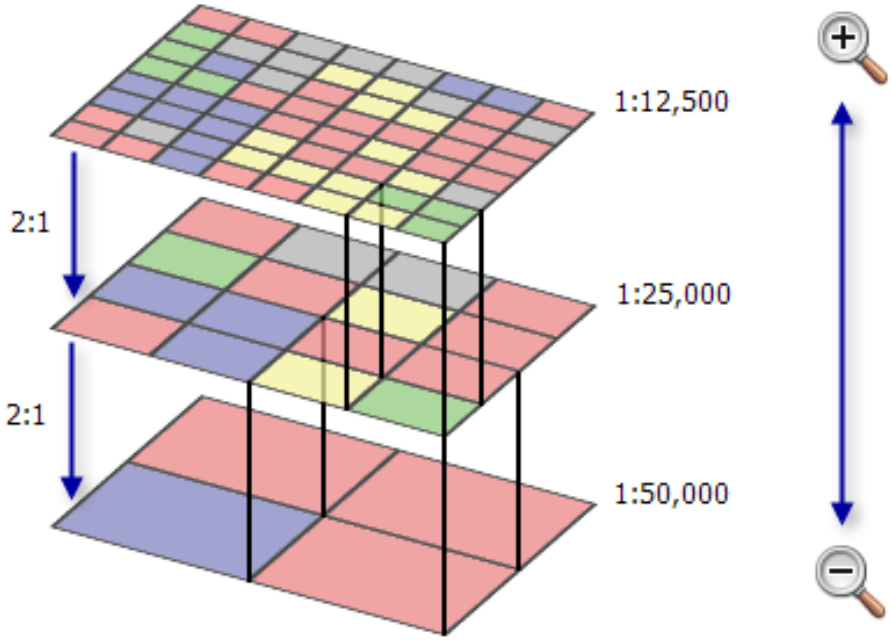
ImageCollection

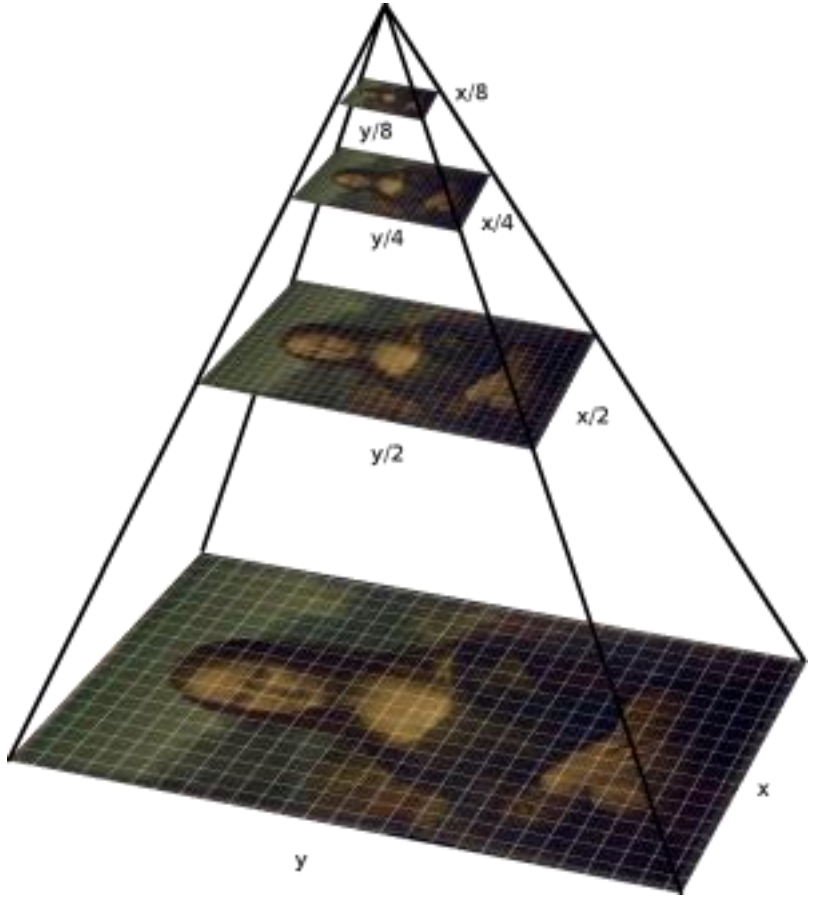


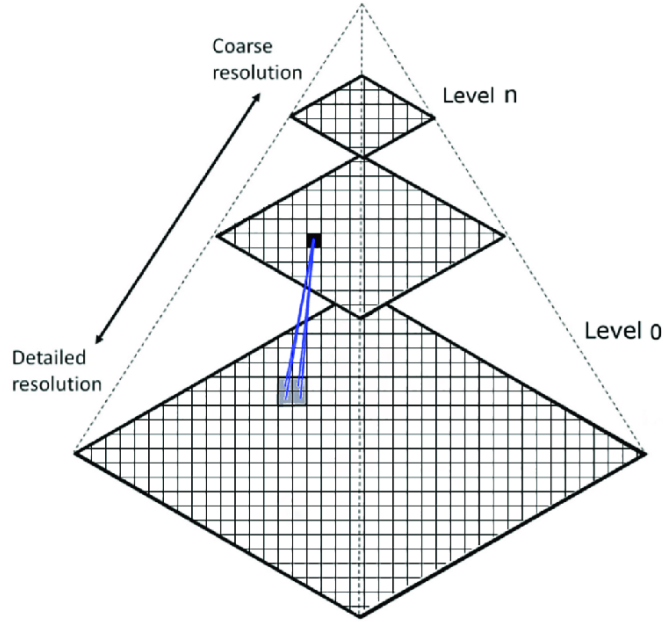
(a)

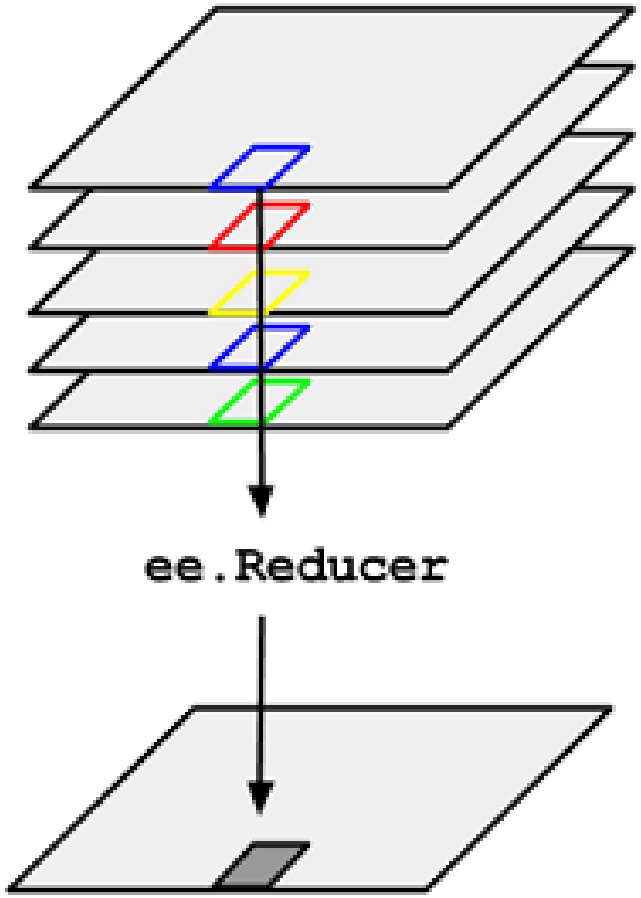


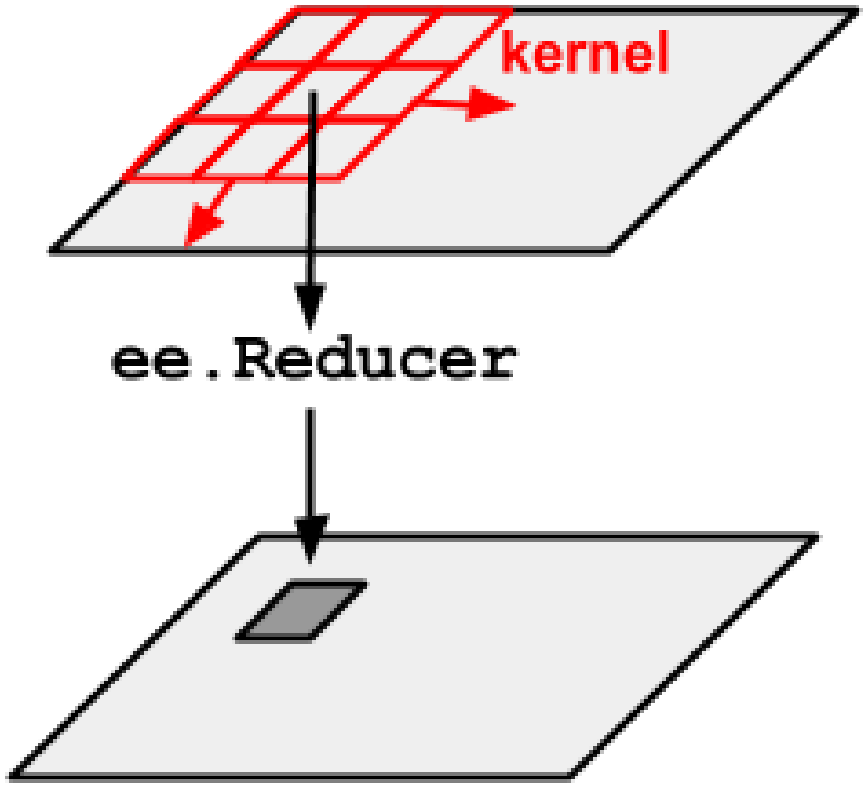
(b)


1:12,500
2:1
1:25,000
2:1
1:50,000


x/8
y/8
x/4
y/4
x/2
y/2
x
y


Coarse
resolution
Level n
Level 0
Detailed
resolution


ee.Reducer


kernel
ee.Reducer

ImageCollection Overview

An ImageCollection is a stack or sequence of images.
An ImageCollection can be loaded by pasting an Earth Engine asset ID into the ImageCollection constructor.
You can find ImageCollection IDs in the data catalog.
For example, to load the Sentinel-2 surface reflectance collection:
var sentinelCollection = ee.ImageCollection('COPERNICUS/S2_SR');
This collection contains every Sentinel-2 image in the public catalog. There are a lot. Usually you want to filter the collection.
In addition to loading an ImageCollection using an Earth Engine collection ID, Earth Engine has methods to create image collections.
The constructor ee.ImageCollection() or the convenience method ee.ImageCollection.fromImages() create image collections from lists of images. You can also create new image collections by merging existing collections.

ee.ImageCollection

ImageCollections can be constructed from the following arguments:

- **A string: assumed to be the name of a collection,**
- **A list of images, or anything that can be used to construct an image.**
- **A single image.**
- **A computed object - reinterpreted as a collection.**

ee.ImageCollection(args)

ee.ImageCollection (Function)

- **ee.ImageCollection(args)**
- **ee.ImageCollection.load(id, version)**
- **ImageCollection.combine(secondary, overwrite)**
- **ee.ImageCollection.fromImages(images)**
- **ImageCollection.merge(collection2)**

ImageCollection Exploring

The Functions for Exploring the ImageCollection :

ImageCollection.size()
ImageCollection.limit(max, property, ascending)
ImageCollection.first()
ImageCollection.getInfo(callback)
ImageCollection.propertyNames()
ImageCollection.toDictionary(properties)
ImageCollection.toBands()
ImageCollection.select(selectors, names)
ImageCollection.sort(property, ascending)
ImageCollection.reduce(reducer, parallelScale)
ImageCollection.aggregate_array(property) ....

Filtering image collections

The way to limit the collection by time or space is by filtering it.
For example, to filter the collection to images that cover a particular location, first define your area of interest with a point (or line or polygon) using the geometry drawing tools. Pan to your area of interest, hover on the Geometry Imports (if you already have one or more geometries defined) and click +new layer (if you don't have any imports, go to the next step). Get the point drawing tool () and make a point in your area of interest. Name the import point. Now, filter the l8 collection to get only the images that intersect the point, then add a second filter to limit the collection to only the images that were acquired in 2015.
Here, **filterBounds()** and **filterDate()** are shortcut methods for the more general filter() method on image collections, which takes an ee.Filter() as its argument.
This is one way to discover the ID of an individual image. Another, more programmatic way to get individual images for analysis is to sort the collection in order to get the most recent, oldest, or optimal image relative to some metadata property.

Filtering image collections

Apply a filter to this collection. **>>>>>>** Returns the filtered collection

ImageCollection.filter(filter) **Collection**

Argument	**Type**	**Details**
this: collection	Collection	The Collection instance.
filter	Filter	A filter to apply to this collection.

ImageCollection.filterBounds(geometry)

ImageCollection.filterDate(start, end)

Processing

Mathematial

– ImageCollection.max()
– ImageCollection.mean()
– ImageCollection.median()
– ImageCollection.min()
– ImageCollection.mode()

– ImageCollection.and()
– ImageCollection.or()

Processing

Earth Engine uses a parallel processing system to carry out computation across a large number of machines. To enable such processing, Earth Engine takes advantage of standard techniques commonly used by functional languages

For Loops

The use of for-loops is discouraged in Earth Engine. The same results can be achieved using a **map()** operation where you specify a function that can be independently applied to each element. This allows the system to distribute the processing to different machines**.**

Maps an algorithm over a collection.

If/Else Conditions

the API does provide a **ee.Algorithms.If()** algorithm

Cumulative Iteration

You may need to do sequential operation, where the result of each iteration is used by the subsequent iteration. Earth Engine provides a **iterate()** method for such tasks.

Maps an algorithm over a collection.

ImageCollection.map(algorithm, dropNulls) >>>>> Returns the mapped collection.

Argument	Type	Details
this: collection	Collection	The Collection instance.
algorithm	Function	

The operation to map over the images or features of the collection. A JavaScript function that receives an image or features and returns one. The function is called only once and the result is captured as a description, so it cannot perform imperative operations or rely on external state.

dropNulls Boolean, optional

If true, the mapped algorithm is allowed to return nulls, and the elements for which it returns nulls will be dropped.

Although map() applies a function to every image in a collection, the function visits every image in the collection independently.

Iterating over an ImageCollection

ImageCollection.iterate(algorithm, first) **>>>** Returns the result of the Collection.iterate() call.

ImageCollection.iterate(algorithm, first)

Argument	Type	Details
this: collection	Collection	The Collection instance.
algorithm	Function	The function to apply to each element. Must take two arguments: an element of the collection and the value from the previous iteration.
first	Object, optional	The initial state.

Masking

Each pixel in each band of an image has a mask. Those with a mask value of 0 or below will be transparent.
Those with a mask of any value above 0 will be rendered.
The mask of an image is set using a call like image1.mask(image2). This call takes the values of image2 and makes them the mask of image1. Any pixels in image2 that have the value 0 will be made transparent in image1.

Composite : Reducing an ImageCollection

To composite images in an ImageCollection, use imageCollection.reduce(). This will composite all the images in the collection to a single image representing, for example, the min, max, mean or standard deviation of the images

More complex reductions are also possible using reduce(). For example, to compute the long term linear trend over a collection, use one of the linear regression reducers.

Composite images created by reducing an image collection are able to produce pixels in any requested projection and therefore have no fixed output projection. Instead, composites have the default projection of WGS-84 with 1-degree resolution pixels. Composites with the default projection will be computed in whatever output projection is requested

Mosaicking

In general, compositing refers to the process of combining spatially overlapping images into a single image based on an aggregation function.

Mosaicking refers to the process of spatially assembling image datasets to produce a spatially continuous image.
A mosaic is a combination or merge of two or more images. you can create a single raster dataset from multiple raster datasets by mosaicking them together. Alternatively, you can create a mosaic dataset and create a virtual mosaic from a collection of raster datasets.
These overlapping areas can be handled in several ways; for example, you can choose to only keep raster data from the first or last dataset, you can blend the overlapping cell values using a weight-based algorithm, you can take the mean of the overlapping cell values, or you can take the minimum or maximum value. When mosaicking discrete data, the First, Minimum, or Maximum options give the most meaningful results. The Blend and Mean options are best suited for continuous data. If any of the input rasters are floating point, the output is floating point. If all the inputs are integer and First, Minimum, or Maximum is used, the output is integer.

ee.ImageCollection.qualityMosaic

Composites all the images in a collection, using a quality band as a per-pixel ordering function.

ImageCollection.qualityMosaic(qualityBand) >>>>> Image

Argument	Type	Details
this: collection	ImageCollection	The collection to mosaic.
qualityBand	String	The name of the quality band in the collection.

ImageCollection Visualization

Images composing an ImageCollection can be visualized as either an animation or a series of thumbnails referred to as a “filmstrip”. These methods provide a quick assessment of the contents of an ImageCollection and an effective medium for witnessing spatiotemporal change

- getVideoThumbURL() produces an animated image series
- getFilmstripThumbURL() produces a thumbnail image series