# GEEdownloader

Ver 1.1

## Introduction

Following is a JavaScript program used to download satellite images from Google Earth Engine, and the program was written by Xiaodong Zhang ([zxd@ouc.edu.cn](mailto:zxd@ouc.edu.cn)), and all the rights are preserved. Please refer the paper “Using free satellite imagery to study the long-term evolution of intertidal bar systems” if you use the program.

The code should be copied to the browser (<https://code.earthengine.google.com/>), detailed usage of the browser interface please refer to the help in the page. The shooting time of the Landsat satellite images can be copied from the run results located in the top right of the browser (Console panel).

The images you want to download will be listed in the Tasks panel, and click the run button to download it one by one.

Following script can be used to download the images in a batch:

“function runTaskList() {

var runButtons = document.querySelector('#task-pane').shadowRoot.querySelectorAll(".run-button")

runButtons.forEach(function(e) {e.click()})

}

runTaskList()

setTimeout(

function(){

var taskDialog = document.querySelectorAll("ee-image-config-dialog")

taskDialog.forEach(function(e) {e.shadowRoot.querySelector("ee-dialog").shadowRoot.querySelector("paper-dialog").querySelector(".ok-button").click()})

},5 \* 1000 );

“

Press F12 to show the control panel of IE or other browser (not the GEE), and copy the above texts in the control panel, and wait a few minutes, which depend on the number of images, then the downloads begin.

The satellite images you want to download may be too much to be prepared by the Google Earth Engine at one time, if so, you can process each dataset separately. For example: you can add the red “/\*” “\*/” pairs in the program to make the sentences used to download Landsat images disabled. Therefore, you can only download the Sentinel images. If the satellite images are still too much to be prepared by the Google Earth Engine at one time, you can revise the start and finish date to reduce the image number.

“var start = ee.Date('1984-1-1');

var finish = ee.Date('2022-1-1');”

use a small CLOUD\_COVER, such as 20, can also reduce the image numer.

“.filter('CLOUD\_COVER <= 20')” for Landsat

“.filter('CLOUDY\_PIXEL\_PERCENTAGE < 80')” for Sentinel

The image showed in the browser may be totally black or white, it is due to the improper range used in the following sentences:

“var rgbVis = {

min: 0,

max: 100,

bands: CurBands,

};“

Change the min and max to get a better show, such as change the max to 1000 or 5000. The proper min and max of different dataset are different. The show in browser don’t influence the downloaded images.

## JavaScript Program

//The program start from here

var district="Feng River mouth";//change the name to your own

var geometry = ee.Geometry.Rectangle([120.02, 35.80, 120.09, 35.896]);//the longitude and latitude range of study area

var start = ee.Date('1984-1-1');

var finish = ee.Date('2022-1-1');

var scale=12.34;//the scale of the satellite image. For example, the resolution of Sentinel image is 10 m, however, the Pseudo Mercator Projection will stretch a pixel to 12.34 m in the latitude of 35.85.

/\*

var CurBands=['B4', 'B3', 'B2'];

var ds='LANDSAT/LT04/C01/T1';

var dataset = ee.ImageCollection(ds)

.filterBounds(geometry)

.filterDate(start, finish)

.filter('CLOUD\_COVER <= 80')

;

print (dataset);

dispimage(dataset);

downloadlist(dataset);

var ds='LANDSAT/LT05/C01/T1';

var dataset = ee.ImageCollection(ds)

.filterBounds(geometry)

.filterDate(start, finish)

.filter('CLOUD\_COVER <= 80')

;

print (dataset);

dispimage(dataset);

downloadlist(dataset);

var ds='LANDSAT/LE07/C01/T1';

var dataset = ee.ImageCollection(ds)

.filterBounds(geometry)

.filterDate(start, finish)

.filter('CLOUD\_COVER <= 80')

;

print (dataset);

dispimage(dataset);

downloadlist(dataset);

var ds='LANDSAT/LC08/C01/T1';

var CurBands=['B5', 'B4', 'B3'];

var dataset = ee.ImageCollection(ds)

.filterBounds(geometry)

.filterDate(start, finish)

.filter('CLOUD\_COVER <= 80')

;

print (dataset);

dispimage(dataset);

downloadlist(dataset);

\*/

var ds='COPERNICUS/S2';

var CurBands=['B8', 'B4', 'B3'];

var dataset = ee.ImageCollection(ds)

.filterBounds(geometry)

.filterDate(start, finish)

.filter('CLOUDY\_PIXEL\_PERCENTAGE < 80')

//.filter(ee.Filter.eq("MGRS\_TILE", '51STV'))//use the sentence to filter the Sentinel images at the same shooting time, and the ‘51STV’ need to be changed in other zones.

;

print (dataset);

dispimage(dataset);

downloadlist(dataset);

function dispimage(dataset){

var disp = dataset.median().select(CurBands);

var rgbVis = {

min: 0,

max: 100,

bands: CurBands,

};

Map.centerObject(geometry, 13);

Map.addLayer(disp.clip(geometry), rgbVis, ds);

}

function downloadlist(dataset){

var indexListcloud = dataset.reduceColumns(ee.Reducer.toList(), ["CLOUD\_COVER"]).get("list");

var indexListtime = dataset.reduceColumns(ee.Reducer.toList(), ["SCENE\_CENTER\_TIME"]).get("list");

var indexList = dataset.reduceColumns(ee.Reducer.toList(), ["system:index"]).get("list");

print(indexList,indexListtime,indexListcloud);

indexList.evaluate(function(indexs) {

print("Total images:",indexs.length);

for (var i=0; i<indexs.length; i++) {

var Curimage = dataset.filter(ee.Filter.eq("system:index", indexs[i])).first().select(CurBands);

var resampled = Curimage.resample('bicubic');

exportImage(resampled, geometry, indexs[i]);

}

});

}

function exportImage(Curimage, region, fileName) {

Export.image.toDrive({

image: Curimage,

description: fileName,

folder: district,

scale: scale,

region: geometry,

fileFormat: 'GeoTIFF',

formatOptions: {

cloudOptimized: true

},

crs: "EPSG:3857",// Pseudo Mercator Projection

maxPixels: 1e13

});

}