Fangnan Du

Google earth engine- Final Project

Imaginary for pipeline delivery network

Background

In today’ s society, the technology is going up fast. We should have large imaginary. Currently, we still rely on the conventional logistics network, which is airplanes, ships and vehicles. But imagine, what if we can create the pipeline delivery network, which is much more time-saving and has low cost. However, it is very difficult to achieve that network, because of lacking the technology and the high capital expenditure.

So in this project, I am going to build a pipeline network from the eastern coast, Connecticut to the great lake area, Rochester and Buffalo, near Canada. So, this pipeline is very meaningful. It can convey the goods from the European to Inner Canada, which will greatly save the time and cost.

Logic

At first, we set 6 states as the study area. In order to identify the two most growing city for pipeline.

Use the light change and do the regression for the light to identify the most energetic city.

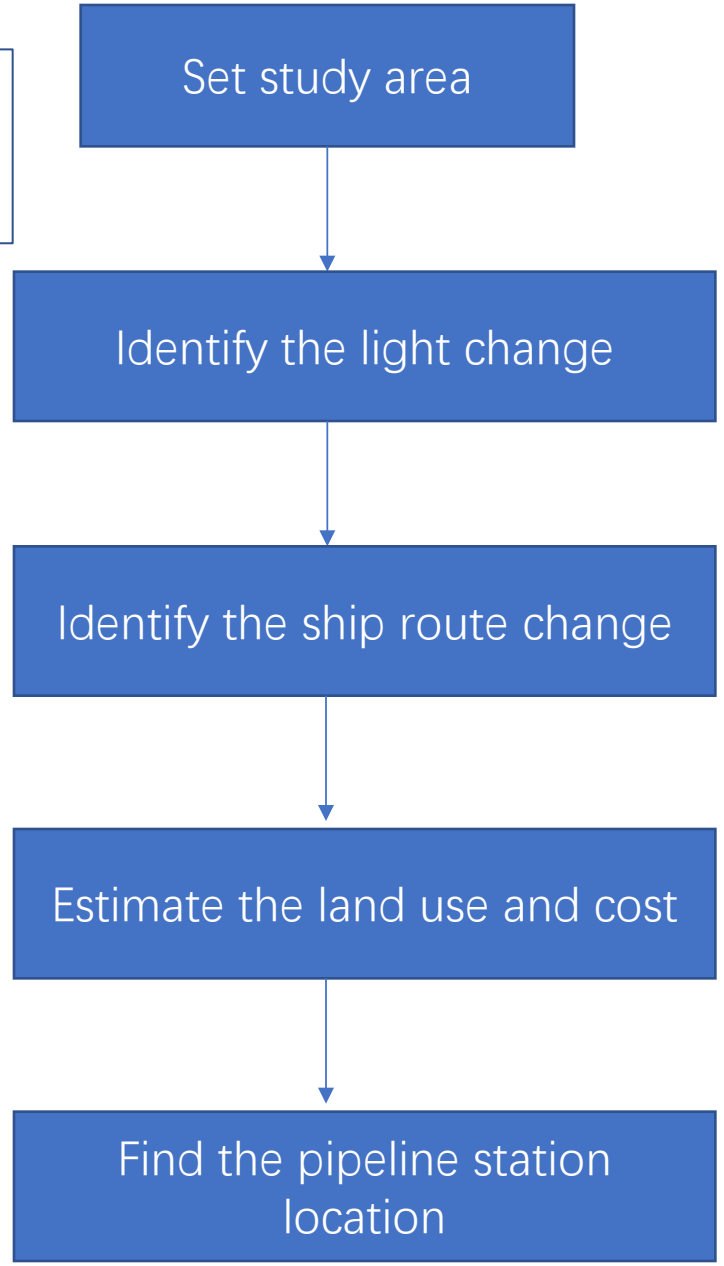
Use the ship route change, to identify the city has growing ship routes which means the potential supply for goods for delivery.

Connecticut

Buffalo

Estimate the cost for build pipeline between these two place

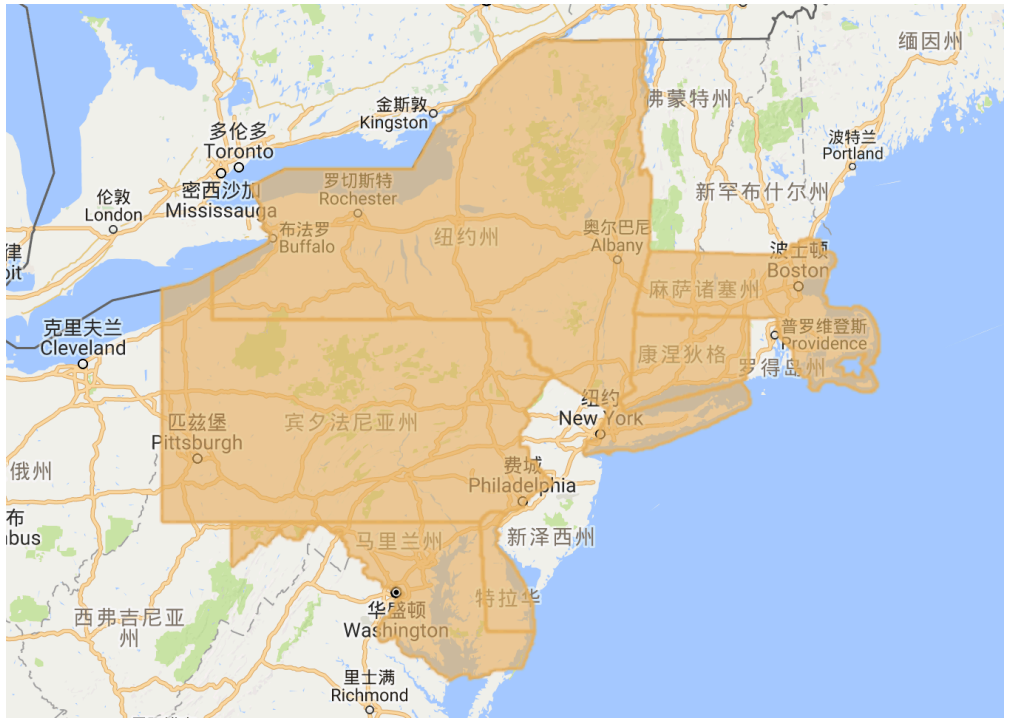
Find the pipeline station location



Scripts

Set study area

```
//import States data
var States =
ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNa
R7Xk8');
print (States);
//Choose our study area–Massachusetts, Maryland, Delaware,
New York & Pennsylvania, Connecticut
var FilterML = ee.Filter.equals('Name', 'Maryland');
var FilterDL = ee.Filter.equals('Name', 'Delaware');
var FilterNY = ee.Filter.equals('Name', 'New York');
var FilterPA = ee.Filter.equals('Name', 'Pennsylvania');
var FilterMA = ee.Filter.equals('Name', 'Massachusetts');
var FilterCT = ee.Filter.equals('Name', 'Connecticut');
var Filter = ee.Filter.or(FilterMA, FilterDL, FilterPA,
FilterML, FilterNY ,FilterCT);
var studyarea = States.filter(Filter);
Map.setCenter( -74, 42, 6 );
Map.addLayer(studyarea, {color:'e8b261'} ,'Study Area');
```



Scripts

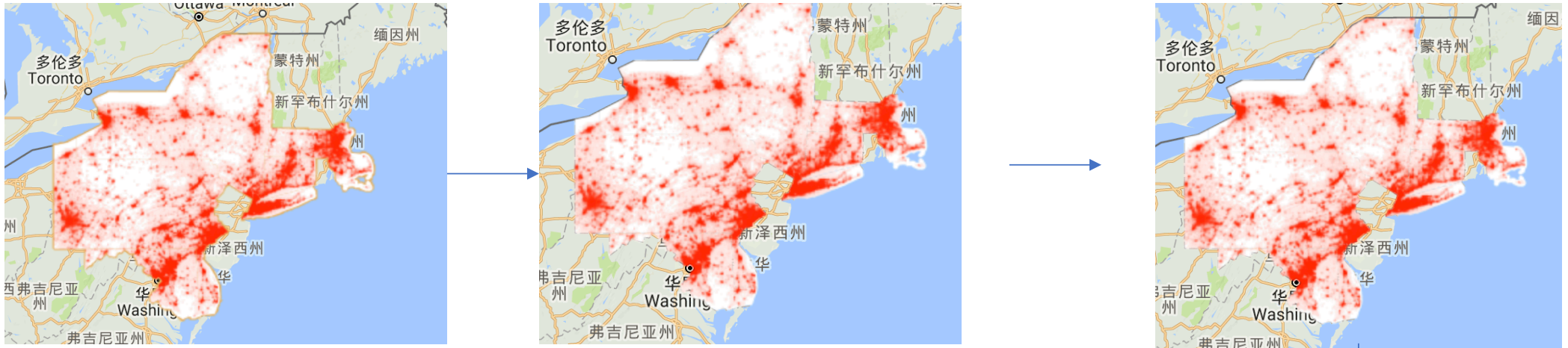
Identify the light change

```
// Get the light change from 2001 to 2012 in study area
// the trend of nighttime lights
var imageCollection = ee.ImageCollection("NOAA/DMSP-OLS/NIGHTTIME_LIGHTS");
print(imageCollection);

// the trend of 2013 nighttime lights
var light2013 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2013_clipped = light2013.clip(studyarea);
Map.addLayer(light2013_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2013_clipped');

var light2012 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2012_clipped = light2012.clip(studyarea);
Map.addLayer(light2012_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2012_clipped');


var light2011 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2011_clipped = light2011.clip(studyarea);
Map.addLayer(light2011_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2011_clipped');


var light2010 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2010_clipped = light2010.clip(studyarea);
Map.addLayer(light2010_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2010_clipped');


var light2009 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2009_clipped = light2009.clip(studyarea);
Map.addLayer(light2009_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2009_clipped');
```

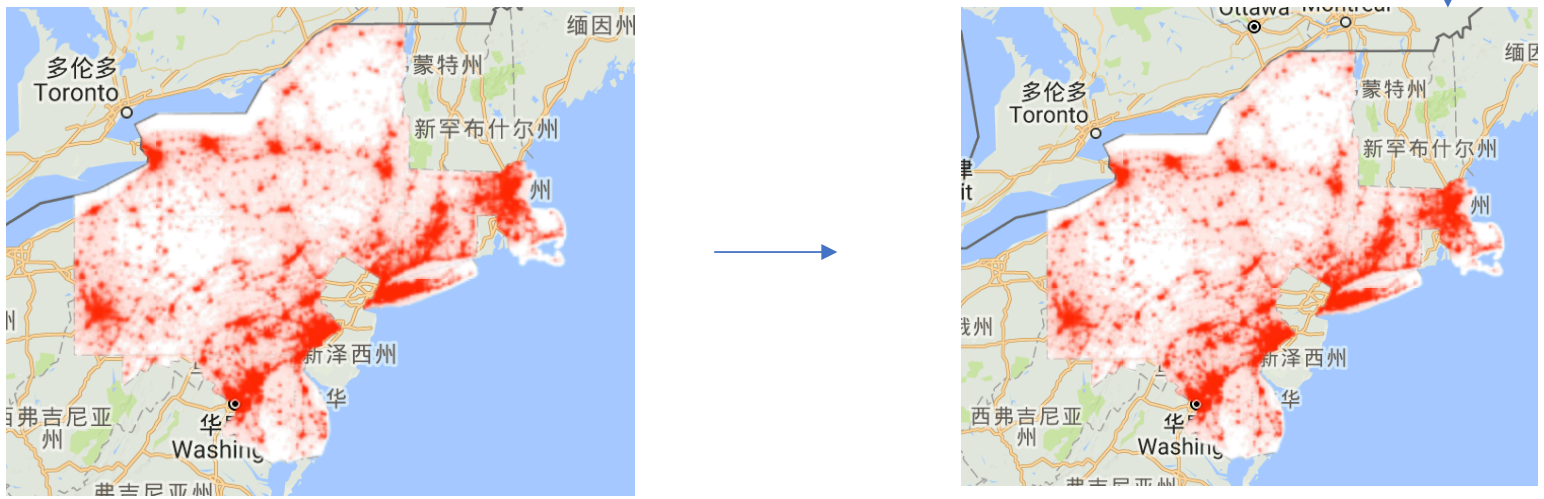
Scripts

Identify the light change

The light change from 2009 to 2013



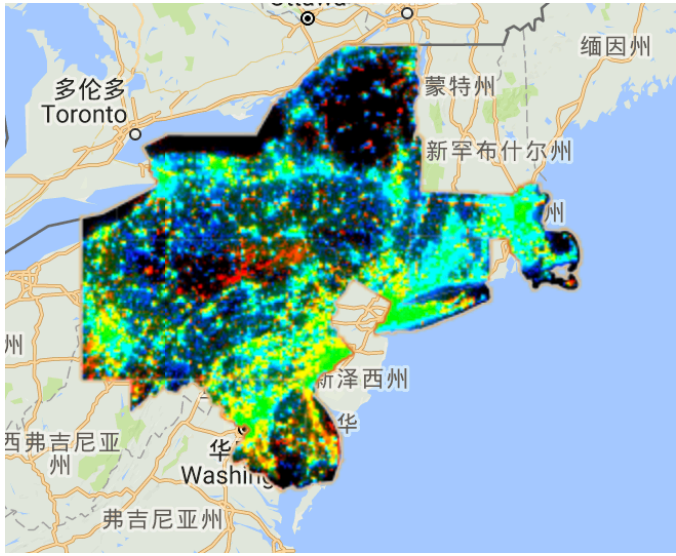
Based on the light change, we will choose the Connecticut and Buffalo as pipeline location.

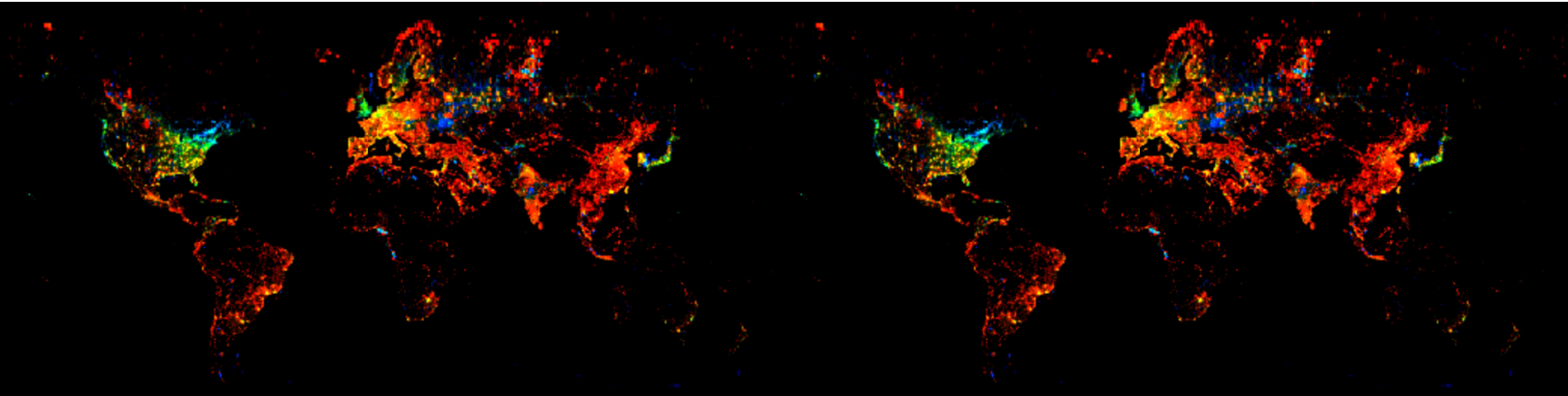


Scripts

Identify the light change

```
// do a linear regression for the lightchange and predict the lightchange for study area and
the entire world
var collection = ee.ImageCollection('NOAA/DMSP-
OLS/NIGHTTIME_LIGHTS') .select('stable_lights').map(createTimeBand);
// Prediction for study area
var fit = collection.reduce(ee.Reducer.linearFit());
var fit_clipped = fit.clipToCollection(studyarea);
Map.addLayer(fit_clipped,
{min: 0, max: [0.18, 20, -0.18], bands: ['scale', 'offset', 'scale']}, 'stable lights trend');
// Prediction for entire world
Map.addLayer(fit,
{min: 0, max: [0.18, 20, -0.18], bands: ['scale', 'offset', 'scale']}, 'stable lights trend');
function createTimeBand(img) {
var year = ee.Date(img.get('system:time_start')).get('year').subtract(2001);
return ee.Image(year).byte().addBands(img); }
```



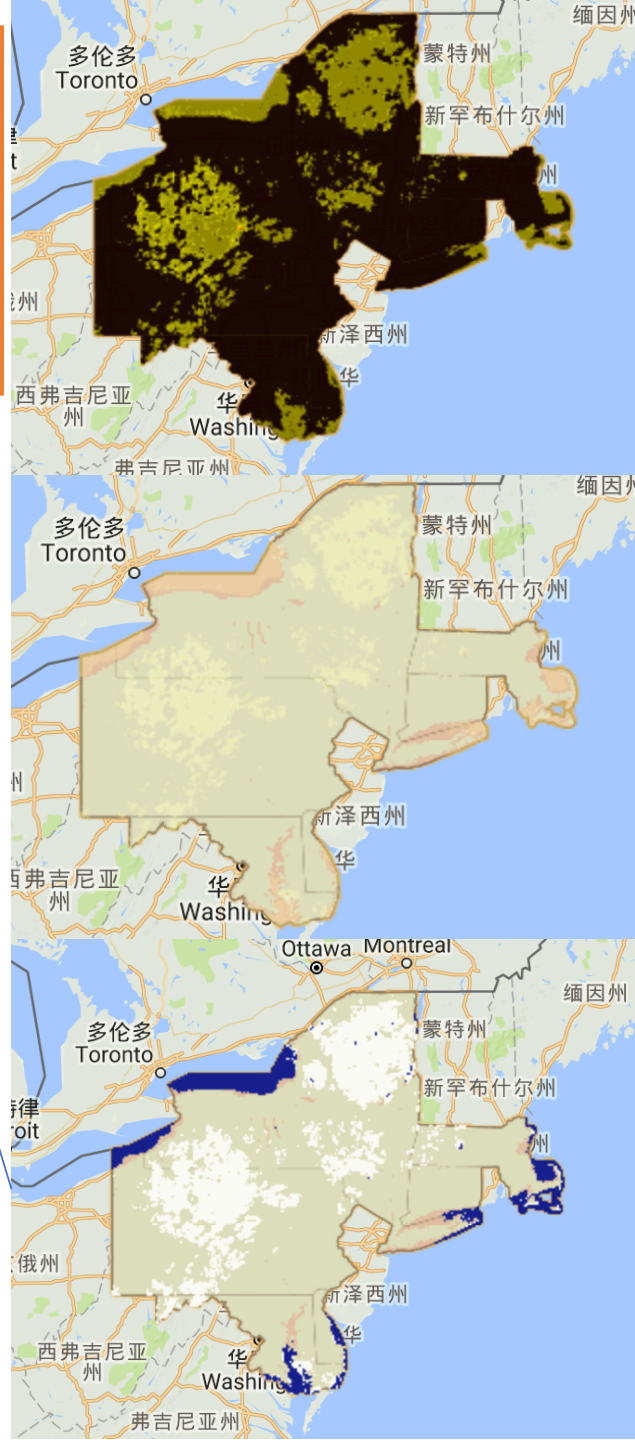


Scripts

Identify the ship route change

```
// load NightImage data which is the nightimage in 2013
var NightImage = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013');
//clip the night for the study area
var ClippedNight = NightImage.clip(studyarea);
//clip the unstable night, which is the new route of the ships
var Night0 = ClippedNight.expression('b(0)');
// Band 0: All Lighting
var Night1 = ClippedNight.expression('b("stable_lights")');
 // Band 1: Persistent Lighting
var Night = ClippedNight.expression('b(0)-b(1)');
 // Changes in Lighting in 2013
print(ClippedNight, Night0, Night1, Night);
Map.addLayer( Night, {min: 0, max: 15, palette: ['190707','ffff00','FF0000'],
opacity:0.9}, 'Changes in Lighting');


//load landcover data in 2013
var landcover = ee.Image('MODIS/051/MCD12Q1/2013_01_01');
print(landcover.getInfo());
//select the feature which is water
var water = landcover.select('Land_Cover_Type_1').eq(0);
//clip the water feature in the study area
var water2 = water.clip(studyarea)
Map.addLayer( water2, {min: 0, max: 15, palette: ['F5F6CE','FF0000','FF0000'],
opacity:0.9}, 'water');
//mask the other lights which are not on the water surface
var Newrount = water2.mask(Night);
Map.addLayer(Newrount, {palette:['FFFFFF', '08088A'],opacity:0.9} , 'New ship
route');
```

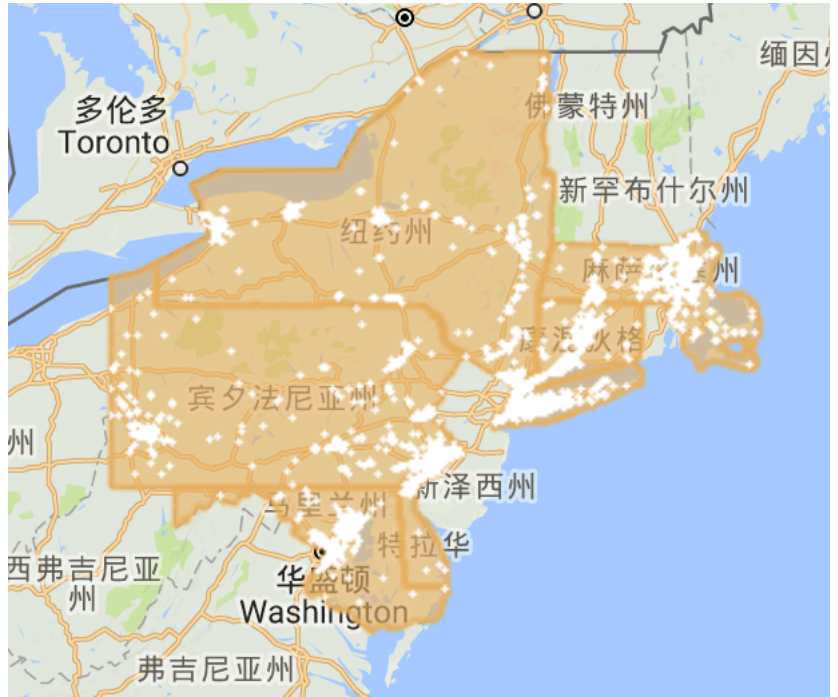
Based on the ship route change , we will choose the Connecticut and Buffalo as pipeline location.



Scripts

Estimate the land use and cost

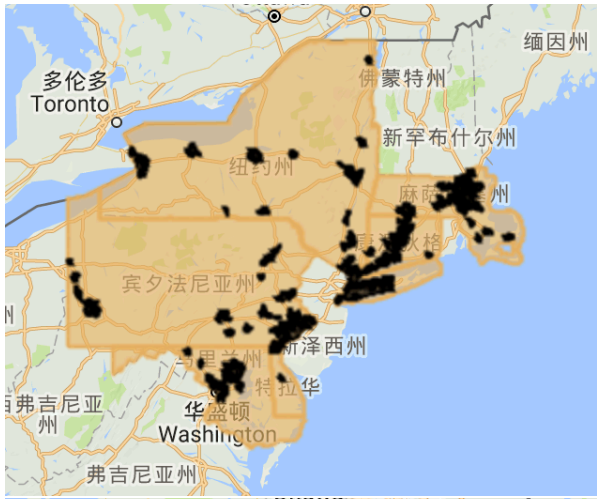
```
// Import land use data
var landuse = ee.Image('MODIS/051/MCD12Q1/2013_01_01');
// Clip the land use data by the three states
var landuse_study = landuse.clip( studyarea );
// Urban area
var Urban  = landuse_study.select(['Land_Cover_Type_1']).eq(13) ;
Map.addLayer( Urban, {opacity:0.5,palette:['ffffff,dd0000']} ,
'Urban' );
var landuse_study = landuse.clip( studyarea );


// *** Calculate the percentage of urban area
var reduce_mean = ee.Reducer.mean();
var percent_urban = Urban.reduceRegion( reduce_mean, studyarea, 500,
null, null, true );
print(percent_urban);
 //5.1% of land is urban in this region



// *** Find the large urban area
// Smooth the urban area in order to connected as region
var urban_max = Urban.reduceNeighborhood( ee.Reducer.max( ),
ee.Kernel.circle( 1 ) );
var urban_smooth = urban_max.mask( urban_max );
Map.addLayer(urban_smooth, {}, 'Smoothed Urban area');
```

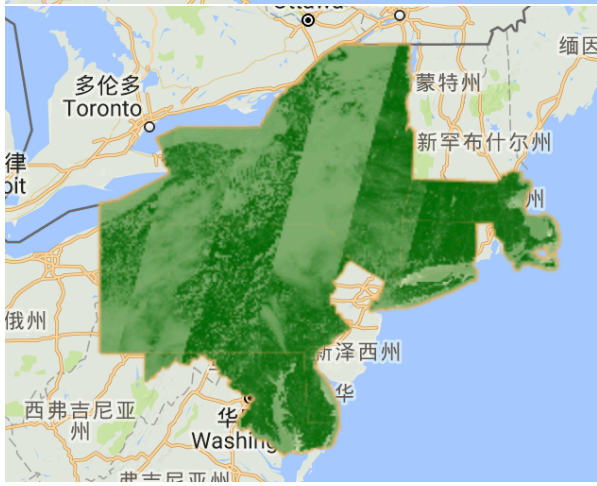


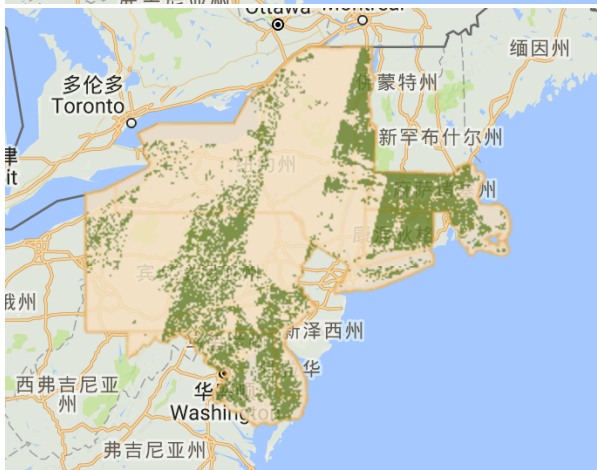
Scripts

Estimate the land use and cost

```
// Calculate size of each urban area region
var urban_size = urban_smooth.connectedPixelCount( 1024, true );
// find the larg urban area
var largesize = urban_size.gt(1023);
var serve_urban = largesize.mask(largesize);Map.addLayer(serve_urban, {},
'serve urban area');
// convert to polygon
var TheREDUCER = ee.Reducer.countEvery();
var urban_polygon = serve_urban.reduceToVectors( TheREDUCER, studyarea,  200,
'polygon',  true);
print(urban_polygon);Map.addLayer(urban_polygon, {}, 'urban polygon');
//** Find out the area with dense vegetation according to NDVI
// Import landsat image collection. Filter by the study area and by date
var Landsat =
ee.ImageCollection('LANDSAT/LC8_SR').filterBounds(studyarea).filterDate('2016-
07-01','2016-07-20');
print(Landsat);
var image = Landsat.mosaic ();
var image_study = image.clip(studyarea);
print (image_study);
Map.addLayer(image_study,{bands: ['B4','B3','B2'],min:0, max:1000},'landsat');

// calculate NDVI = (NIR - RED) / (NIR + RED)
var NDVI = image_study.expression('(b(4)-b(3))/(b(4)+b(3))');
print (NDVI);
Map.addLayer(NDVI,{min:-1,max:1,palette: ['FFF8DC','006400']},'NDVI');


//NDVI higher than 0.7 means the vegetation there is dense.
var vegetation = NDVI.gt(0.7);
Map.addLayer(vegetation,{palette:['FFFFFF','006400'],opacity:0.5},'vegetation'
);
```



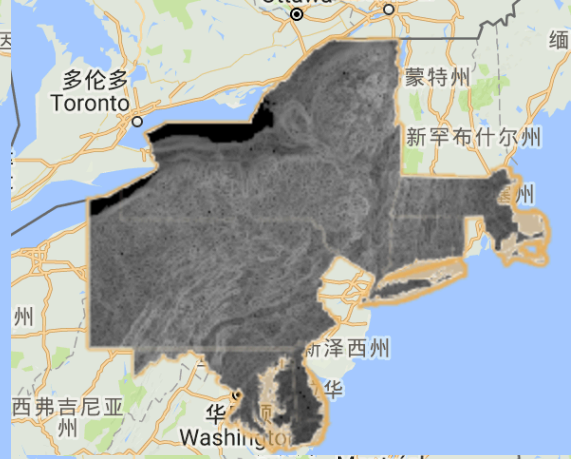


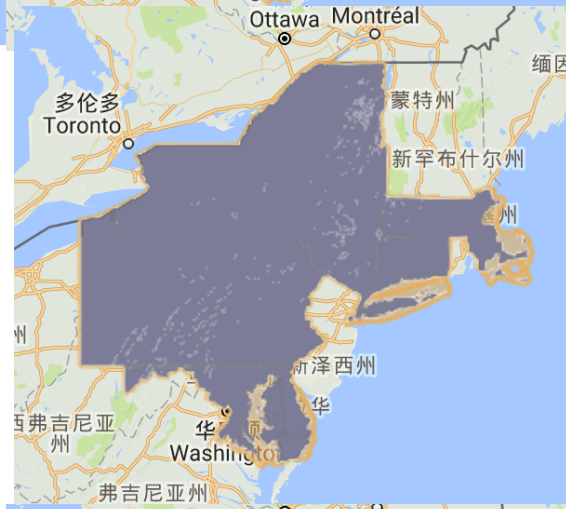


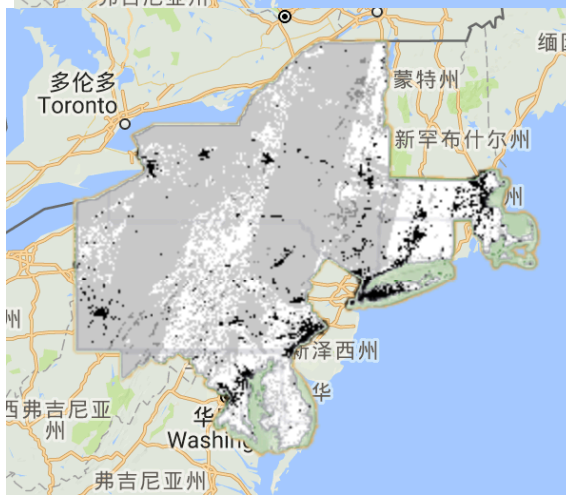
Scripts

Estimate the land use and cost

```
// ** Slope
// Import elevation
var elevation = ee.Image( 'CGIAR/SRTM90_V4' );
// Clip the elevation by the six states
var elevation_study = elevation.clip( studyarea );
Map.addLayer (elevation_study, {min:0, max: 2500, opacity:0.8,
palette:'ffffff,8dbeb1'} , 'elevation of study area');
// calculate slop of this area
var slope = ee.Terrain.slope( elevation_study );
Map.addLayer( slope, {gamma:6}, 'Slope' );


//reclassify the slope
var score_slope
=slope.lt(3).multiply(3).add( slope.lt(10).and(slope.gt(3)).multiply(2).add(slope.gt(
10).multiply(1)));
Map.addLayer( score_slope, {palette:'ffffff,000033'},'score_slope' );


# the final cost
var cost=
  Urban.eq(0).multiply(vegetation.eq(1).multiply(3)
  .add(score_slope.multiply(4)));
var reduce_max = ee.Reducer.max();


var max_value = cost.reduceRegion(reduce_max,
studyarea, 500, null, null, true );
print(max_value);//15

Map.addLayer(cost, {min: 0, max: 15},"cost");
```





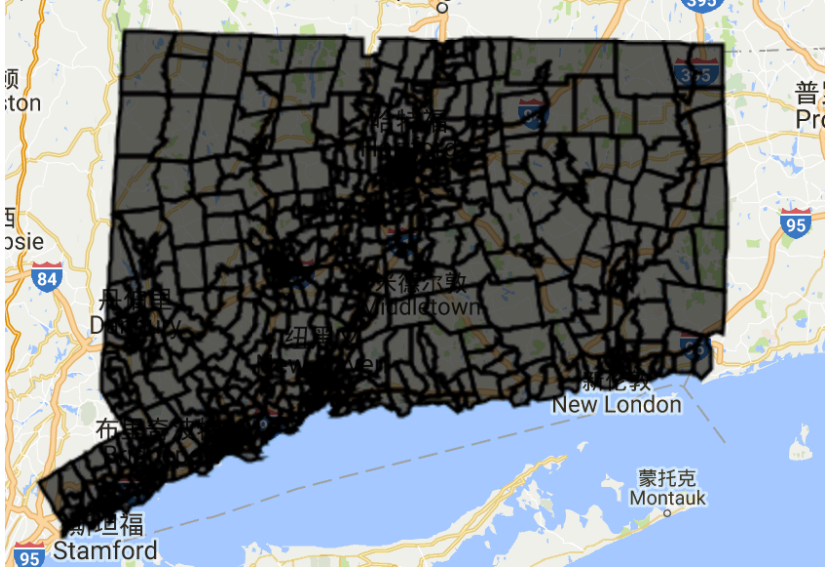


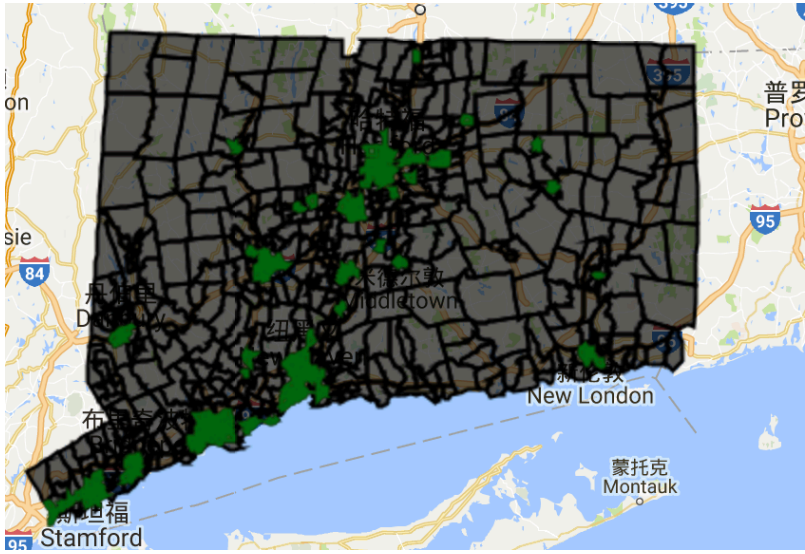
The company can based on this cost to build the pipeline

Scripts

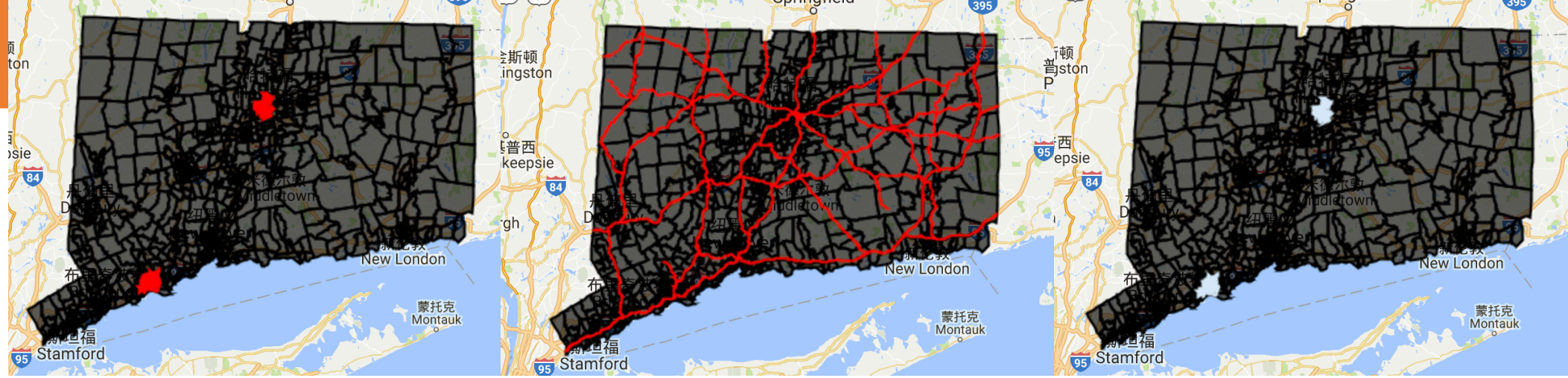
Find the pipeline station location

```
//. In order to locate the starter of pipeline, I would like to make several
criteria:
//1.located in the place which has a higher population density.
//2. Select tracts that have major access to the main roads, which we can create a
//1000m buffer around the road.
//3. The distance should be with in 3000m from the major cities.
var CTTracts =
ee.Collection.loadTable("ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6");
Map.centerObject(CTTracts,8);
Map.addLayer(CTTracts, {color:"000000"}, "CT Census Tracts");
print(CTTracts);
//1st Criteria: the population density should larger than the median population
density //clean the data
var CTTracts_filt =
CTTracts.filterMetadata("AREA","not_equals",null) .filterMetadata("AREA","greater_
than",0)
.filterMetadata("POP","not_equals",null)
.filterMetadata("POP","greater_than",0);
//using the function to define the poplation density
var AddPop_Density=function(feature){
var PopDensity =ee.Number(feature.get("POP")).divide(feature.get("AREA")); return
feature.set("Pop_Density",PopDensity);
};
var CTTracts1 = CTTracts_filt.map(AddPop_Density); print(CTTracts1);
//calculating the mean of the population mean and select
var PopDen_mean = CTTracts1.aggregate_mean('Pop_Density');
var PopDen_high = CTTracts1.filterMetadata('Pop_Density', 'greater_than',
PopDen_mean);
Map.addLayer(PopDen_high,{color:'06660e'}); Map.centerObject(PopDen_high, 8);
```

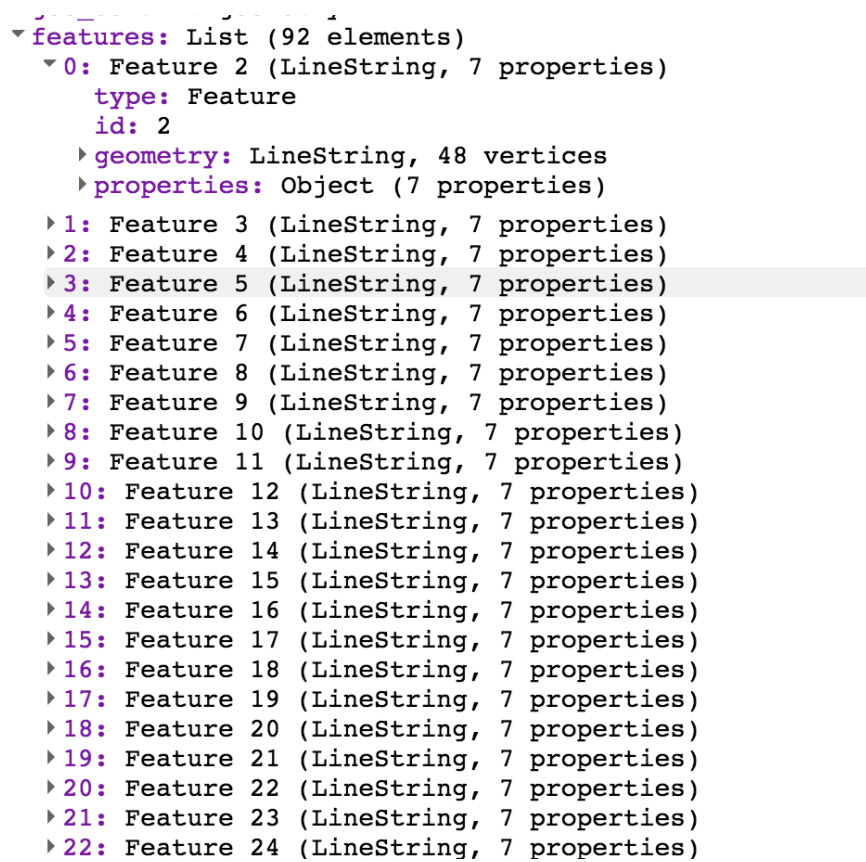




Scripts




```
//select the distance from the city with in 3000
var CityFEATURES =
ee.FeatureCollection("ft:1G3RZbWoTiCiYv_LEwc7xKZq8aYoPZlL5_KuVhyDM"); var
cityFilter=ee.Filter.metadata("city_name","contains","Conn");
var mainCities=CityFEATURES.filter(cityFilter);
var distanceFilter=ee.Filter.withinDistance(3000,'.geo',null,'.geo');
var distJoin=ee.Join.simple();
var CityFiltered=distJoin.apply(PopDen_high,mainCities,distanceFilter);
Map.addLayer(CityFiltered, {color:"ff0000"}, 'CityFiltered');
Map.centerObject(CityFiltered, 8);
// Select tracts that have major roads going through.
var ct_road =ee.FeatureCollection('ft:1GqjJDesYsPkLVqJ1dyiPplfMvcAY2LCwfTPye9O0');
print(ct_road);
Map.addLayer(ct_road, {color:'ff0000'},'ct_road' ); Map.centerObject(ct_road,8);
var road_geometry = ct_road.geometry();
var road_buffer = road_geometry.buffer(1000);
//colored in yellow Map.addLayer(road_buffer, {color:"f3ecbc"}, 'Road Buffer');
var final=CityFiltered.filterBounds(road_buffer);
Map.addLayer(final, {color:"D6EAF8"}, 'final');
Map.centerObject(final, 8);
```



Appendix: full script

Code link:

https://code.earthengine.google.com/66611a674e493b6daf982528aa9ac00d

```
//import States data
var States = ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNaR7Xk8');
print (States);
//Choose our study area–Massachusetts, Maryland, Delaware, New York & Pennsylvania, Connecticut
var FilterML = ee.Filter.equals('Name', 'Maryland');
var FilterDL = ee.Filter.equals('Name', 'Delaware');
var FilterNY = ee.Filter.equals('Name', 'New York');
var FilterPA = ee.Filter.equals('Name', 'Pennsylvania');
var FilterMA = ee.Filter.equals('Name', 'Massachusetts');
var FilterCT = ee.Filter.equals('Name', 'Connecticut');
var Filter = ee.Filter.or(FilterMA, FilterDL, FilterPA, FilterML, FilterNY ,FilterCT);
var studyarea = States.filter(Filter);
Map.setCenter( -74, 42, 6 );
Map.addLayer(studyarea, {color:'e8b261'} ,'Study Area');

 // Get the light change from 2001 to 2012 in study area
 // the trend of nighttime lights
 var imageCollection = ee.ImageCollection("NOAA/DMSP-OLS/NIGHTTIME_LIGHTS");
 print(imageCollection);

 // the trend of 2013 nighttime lights
 var light2013 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
 var light2013_clipped = light2013.clip(studyarea);
 Map.addLayer(light2013_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2013_clipped');
```

Appendix: full script

```
var light2012 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2012_clipped = light2012.clip(studyarea);
Map.addLayer(light2012_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2012_clipped');


var light2011 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2011_clipped = light2011.clip(studyarea);
Map.addLayer(light2011_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2011_clipped');


var light2010 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2010_clipped = light2010.clip(studyarea);
Map.addLayer(light2010_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2010_clipped');


var light2009 = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013').select("stable_lights");
var light2009_clipped = light2009.clip(studyarea);
Map.addLayer(light2009_clipped,{min: 0, max: 63, palette: ['FFFFFF','FF2805']},'light2009_clipped');
// do a linear regression for the lightchange and predict the lightchange for study area and
the entire world
var collection = ee.ImageCollection('NOAA/DMSP-
OLS/NIGHTTIME_LIGHTS') .select('stable_lights').map(createTimeBand);
// Prediction for study area
var fit = collection.reduce(ee.Reducer.linearFit());
var fit_clipped = fit.clipToCollection(studyarea);
Map.addLayer(fit_clipped,
{min: 0, max: [0.18, 20, -0.18], bands: ['scale', 'offset', 'scale']}, 'stable lights trend');
// Prediction for entire world
Map.addLayer(fit,
{min: 0, max: [0.18, 20, -0.18], bands: ['scale', 'offset', 'scale']}, 'stable lights trend');
function createTimeBand(img) {
var year = ee.Date(img.get('system:time_start')).get('year').subtract(2001);
return ee.Image(year).byte().addBands(img); }
```

Appendix: full script

```
// load NightImage data which is the nightimage in 2013
var NightImage = ee.Image('NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182013');
//clip the night for the study area
var ClippedNight = NightImage.clip(studyarea);
//clip the unstable night, which is the new route of the ships
var Night0 = ClippedNight.expression('b(0)');
// Band 0: All Lighting
var Night1 = ClippedNight.expression('b("stable_lights")');
 // Band 1: Persistent Lighting
var Night = ClippedNight.expression('b(0)-b(1)');
 // Changes in Lighting in 2013
print(ClippedNight, Night0, Night1, Night);
Map.addLayer( Night, {min: 0, max: 15, palette: ['190707','ffff00','FF0000'],
opacity:0.9}, 'Changes in Lighting');


//load landcover data in 2013
var landcover = ee.Image('MODIS/051/MCD12Q1/2013_01_01');
print(landcover.getInfo());
//select the feature which is water
var water = landcover.select('Land_Cover_Type_1').eq(0);
//clip the water feature in the study area
var water2 = water.clip(studyarea)
Map.addLayer( water2, {min: 0, max: 15, palette: ['F5F6CE','FF0000','FF0000'],
opacity:0.9}, 'water');
//mask the other lights which are not on the water surface
var Newrount = water2.mask(Night);
Map.addLayer(Newrount, {palette:['FFFFFF', '08088A'],opacity:0.9} , 'New ship
route');
```

Appendix: full script

```
// Import land use data
var landuse = ee.Image('MODIS/051/MCD12Q1/2013_01_01');
// Clip the land use data by the three states
var landuse_study = landuse.clip( studyarea );
// Urban area
var Urban  = landuse_study.select(['Land_Cover_Type_1']).eq(13) ;
Map.addLayer( Urban, {opacity:0.5,palette:['ffffff,dd0000']} , 'Urban' );
var landuse_study = landuse.clip( studyarea );


// *** Calculate the percentage of urban area
var reduce_mean = ee.Reducer.mean();
var percent_urban = Urban.reduceRegion( reduce_mean, studyarea, 500, null, null, true );
print(percent_urban);
 //5.1% of land is urban in this region



// *** Find the large urban area
// Smooth the urban area in order to connected as region
var urban_max = Urban.reduceNeighborhood( ee.Reducer.max( ), ee.Kernel.circle( 1 ) );
var urban_smooth = urban_max.mask( urban_max );
Map.addLayer(urban_smooth, {}, 'Smoothed Urban area');
```

Appendix: full script

```
// Calculate size of each urban area region
var urban_size = urban_smooth.connectedPixelCount( 1024, true );
// find the larg urban area
var largesize = urban_size.gt(1023);
var serve_urban = largesize.mask(largesize);Map.addLayer(serve_urban, {}, 'serve urban area');
// convert to polygon
var TheREDUCER = ee.Reducer.countEvery();
var urban_polygon = serve_urban.reduceToVectors( TheREDUCER, studyarea,  200, 'polygon',  true);
print(urban_polygon);Map.addLayer(urban_polygon, {}, 'urban polygon');
//** Find out the area with dense vegetation according to NDVI
// Import landsat image collection. Filter by the study area and by date
var Landsat = ee.ImageCollection('LANDSAT/LC8_SR').filterBounds(studyarea).filterDate('2016-07-01','2016-07-20');
print(Landsat);
var image = Landsat.mosaic ();
var image_study = image.clip(studyarea);
print (image_study);
Map.addLayer(image_study,{bands: ['B4','B3','B2'],min:0, max:1000},'landsat');

// calculate NDVI = (NIR - RED) / (NIR + RED)
var NDVI = image_study.expression('(b(4)-b(3))/(b(4)+b(3))');
print (NDVI);
Map.addLayer(NDVI,{min:-1,max:1,palette: ['FFF8DC','006400']},'NDVI');


//NDVI higher than 0.7 means the vegetation there is dense.
var vegetation = NDVI.gt(0.7);
Map.addLayer(vegetation,{palette:['FFFFFF','006400'],opacity:0.5},'vegetation');
```

Appendix: full script

```
// ** Slope
// Import elevation
var elevation = ee.Image( 'CGIAR/SRTM90_V4' );
// Clip the elevation by the six states
var elevation_study = elevation.clip( studyarea );
Map.addLayer (elevation_study, {min:0, max: 2500, opacity:0.8, palette:'ffffff,8dbeb1'} , 'elevation of study area');
// calculate slop of this area
var slope = ee.Terrain.slope( elevation_study );
Map.addLayer( slope, {gamma:6}, 'Slope' );


//reclassify the slope
var score_slope =slope.lt(3).multiply(3).add( slope.lt(10).and(slope.gt(3)).multiply(2).add(slope.gt(10).multiply(1)));
Map.addLayer( score_slope, {palette:'ffffff,000033'},'score_slope' );




# the final cost
var cost=
  Urban.eq(0).multiply(vegetation.eq(1).multiply(3)
  .add(score_slope.multiply(4)));
var reduce_max = ee.Reducer.max();


var max_value = cost.reduceRegion(reduce_max, studyarea, 500, null, null, true );
print(max_value);//15

Map.addLayer(cost, {min: 0, max: 15},"cost");
```

Appendix: full script

```
//. In order to locate the starter of pipeline, I would like to make several criteria:
//1.located in the place which has a higher population density.
//2. Select tracts that have major access to the main roads, which we can create a
//1000m buffer around the road.
//3. The distance should be with in 3000m from the major cities.
var CTTracts = ee.Collection.loadTable("ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6");
Map.centerObject(CTTracts,8);
Map.addLayer(CTTracts, {color:"000000"}, "CT Census Tracts");
print(CTTracts);
//1st Criteria: the population density should larger than the median population density //clean the data
var CTTracts_filt =
CTTracts.filterMetadata("AREA","not_equals",null) .filterMetadata("AREA","greater_than",0)
.filterMetadata("POP","not_equals",null)
.filterMetadata("POP","greater_than",0);
//using the function to define the poplation density
var AddPop_Density=function(feature){
var PopDensity =ee.Number(feature.get("POP")).divide(feature.get("AREA")); return
feature.set("Pop_Density",PopDensity);
};
var CTTracts1 = CTTracts_filt.map(AddPop_Density); print(CTTracts1);
//calculating the mean of the population mean and select
var PopDen_mean = CTTracts1.aggregate_mean('Pop_Density');
var PopDen_high = CTTracts1.filterMetadata('Pop_Density', 'greater_than', PopDen_mean);
Map.addLayer(PopDen_high,{color:'06660e'}); Map.centerObject(PopDen_high, 8);
```

Appendix: full script

```
//select the distance from the city with in 3000
var CityFEATURES =
ee.FeatureCollection("ft:1G3RZbWoTiCiYv_LEwc7xKZq8aYoPZlL5_KuVhyDM"); var
cityFilter=ee.Filter.metadata("city_name","contains","Conn");
var mainCities=CityFEATURES.filter(cityFilter);
var distanceFilter=ee.Filter.withinDistance(3000,'.geo',null,'.geo');
var distJoin=ee.Join.simple();
var CityFiltered=distJoin.apply(PopDen_high,mainCities,distanceFilter);
Map.addLayer(CityFiltered, {color:"ff0000"}, 'CityFiltered');
Map.centerObject(CityFiltered, 8);
// Select tracts that have major roads going through.
var ct_road =ee.FeatureCollection('ft:1GqjJDesYsPkLVqJ1dyiPplfMvcAY2LCwfTPye9O0');
print(ct_road);
Map.addLayer(ct_road, {color:'ff0000'},'ct_road' ); Map.centerObject(ct_road,8);
var road_geometry = ct_road.geometry();
var road_buffer = road_geometry.buffer(1000);
//colored in yellow Map.addLayer(road_buffer, {color:"f3ecbc"}, 'Road Buffer');
var final=CityFiltered.filterBounds(road_buffer);
Map.addLayer(final, {color:"D6EAF8"}, 'final');
Map.centerObject(final, 8);
```

Reference:

The final EE project-from Zhang Mingfang
The final EE project-from Zhang Qian
The final EE project-from Zhao Luyun