

Development of WEB-GIS based on Google Maps Javascript API

Instructor - Prof.Dr.Yuji Murayama Teaching Assistant - GONG Hao

Division of Spatial Information Science Graduate School of Life and Environmental Sciences University of Tsukuba

## Getting Started

### What are the Google Maps APIs?

The Google Maps APIs give developers several ways of embedding Google Maps into web pages or retrieving data from Google Maps, and allow for either simple use or extensive customization. There are several API offerings:

#### Web APIs:

- Google Maps JavaScript API
- Google Static Maps API
- Google Street View Image API
- Google Maps Embed API

#### Web service APIs:

- Google Maps Directions API
- Google Maps Distance Matrix API
- Google Maps Elevation API
- Google Maps Geocoding API
- Google Maps Geolocation API
- Google Maps Roads API
- Google Maps Time Zone API
- · Google Places API Web Service

#### Mobile APIs:

- Google Maps Android API
- · Google Places API for Android
- · Google Maps SDK for iOS
- Google Places API for iOS

### Standard

- Free, external, and publicly available implementations
- ✓ Unlimited, free usage of Google Maps Android API and Google Maps SDK for iOS
- ✓ Pay for uplifts above usage limits
- ✓ Individual API pricing and usage limits

**ENABLE BILLING** 

### Premium

- Licenses for internal, OEM, and asset tracking implementations
- ✓ Annual contracts with enterprise terms
- 24 hour technical support
- ✓ Service level agreement (SLA)
- ✓ Guaranteed ad-free and enhanced API features

CONTACT SALES

(Source: https://developers.google.com/maps/pricing-and-plans)



Basically, standard Google Maps APIs are free for a wide variety of use cases. Based on Google Maps APIs, we could build our own WEB-GIS services.

This lesson will introduce how to use the Google Maps Javascript API to develop a simple WEB-GIS system.

Depending on your needs, you may find yourself using one or a combination of these APIs.

## Steps

### 1. Priori knowledge

Have programming experience in HTML, JavaScript and CSS. Have experience in processing GIS dataset.

### 2. Get an API key

All Google Maps JavaScript API applications require authentication. For standard API users, if you're using the API under the standard plan, you must use an API key set up in a project of your choice.

### 3. Start your project

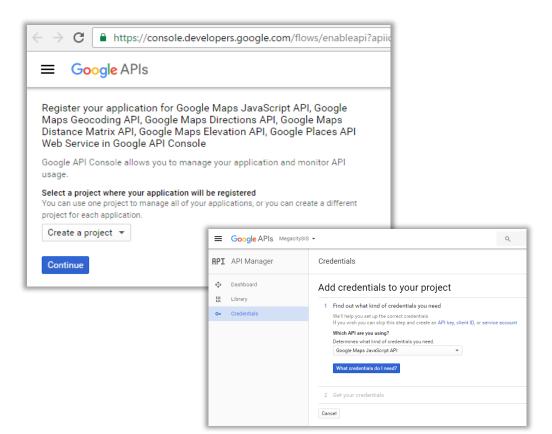
Use your GIS and programming skills to develop the site.

Final, deploy your site to a web serve.

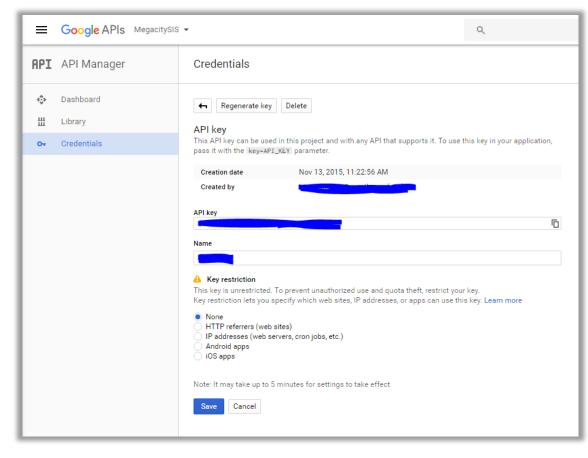


### Get an API key

- 1. Have a google account
- 2. Go to the Google API Console, enable your APIs, and check your APIs.



3. Go to the API manager, check your project name, API key and other setting.

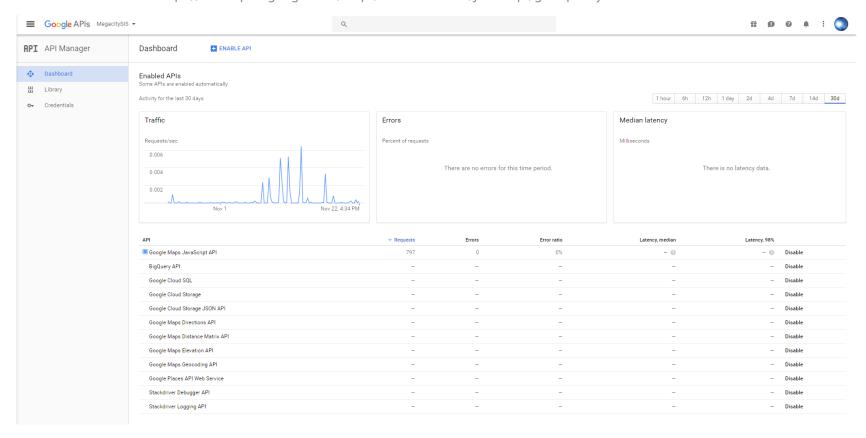


## Get an API key

From now on, you can use the Google Maps API services

<script src="https://maps.googleapis.com/maps/api/js?key= YOUR\_API\_KEY &callback=initMap"
async defer></script>

\* More information: https://developers.google.com/maps/documentation/javascript/get-api-key



## Start your project

4 simple WEB-GIS examples will be presented in this PPT.

Several things you should know before your starting,

- 1. understand your project your purpose, make it simple
- 2. understand your geospatial dataset
- 3. try to separate a big function into several small functions

# Examples

Simple Basic Map

Simple Marker

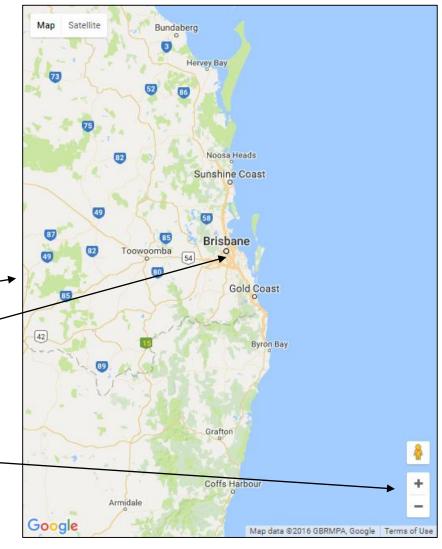
Simple Polylines

Ground Overlays



```
<!DOCTYPE html>
<html>
  <head>
    <title>Simple Map</title>
    <meta name="viewport" content="initial-scale=1.0">
    <meta charset="utf-8">
    <style>
      /* Always set the map height explicitly to define the size of the div
       * element that contains the map. */
      #map {
        height: 100%;
      /* Optional: Makes the sample page fill the window. */
      html, body {
        height: 100%;
        margin: 0;
        padding: 0;
                                                                               CSS style setting
   </style>
  </head>
  <body>
   <div id="map"></div>
                                                                              Define the map layout
    <script>
      var map;
                                                           get the map layout
      function initMap() {
        map = new_google.maps.Map(document.getElementById('map'), {
          center: {lat: -34.397, lng: 150.644}, \rightarrow Set the location
          zoom: 8
        });
                   \stackrel{\searrow}{\sim} Set the zoom level -
    </script>
    <script src="https://maps.googleapis.com/maps/api/js?key= YOUR_API_KEY &callback=initMap"</pre>
                                                                               Google JS parts
   async defer></script>
  </body>
</html>
```

## Simple Basic Map



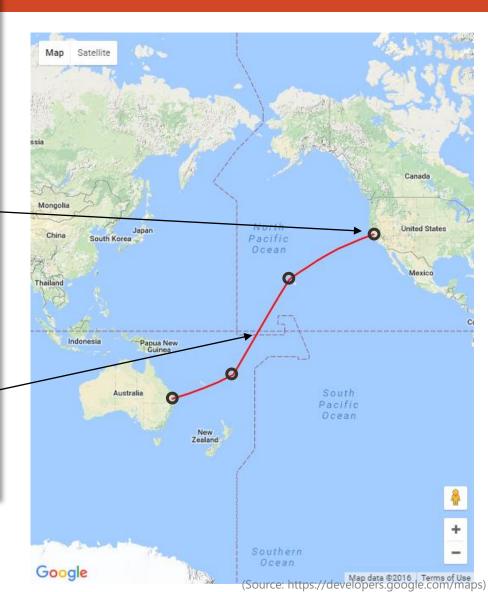
```
<!DOCTYPE html>
<html>
  <head>
    <title>Simple Map</title>
    <meta name="viewport" content="initial-scale=1.0">
    <meta charset="utf-8">
    <style>
      /* Always set the map height explicitly to define the size of the div
       * element that contains the map. */
      #map {
        height: 100%;
              <script>
      /* Opt:
      html,
               function initMap()
       heigh
                 var myLatLng = {lat: -25.363, lng: 131.044};
        marg1
        paddi
                 var map = new google.maps.Map(document.getElementById('map'), {
                   zoom: 4,
    </style>
                   center: myLatLng
  </head>
  <body>
                                              Define the marker
    <div id='
                 var marker = new google.maps.Marker({
   <script>
                    position: myLatLng,
      var map
                   map: map,
      functio
                    title: 'Hello World!'
        map
          cen
          Z00
              </script>
    </script>
    <script src="https://maps.googleapis.com/maps/api/js?key= YOUR_API_KEY&callback=initMap"</pre>
   async defer></script>
  </body>
</html>
```

## Simple Marker



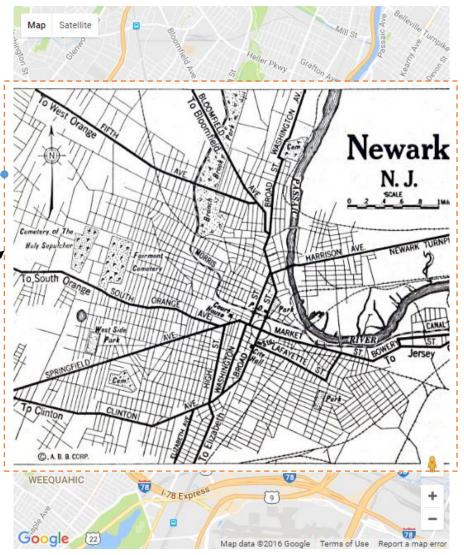
```
<!DOCTYPE html>
        <script>
<html>
  <head:
          // This example creates a 2-pixel-wide red polyline showing the path of William
    <ti
          // Kingsford Smith's first trans-Pacific flight between Oakland, CA, and
    <me
          // Brisbane, Australia.
    <me
    <st
          function initMap() {
            var map = new google.maps.Map(document.getElementById('map'), {
              zoom: 3,
              center: {lat: 0, lng: -180},
              mapTypeId: 'terrain'
            });
           var flightPlanCoordinates = [
              {lat: 37.772, lng: -122.214},____
              {lat: 21.291, lng: -157.821},
              {lat: -18.142, lng: 178.431},
              {lat: -27.467, lng: 153.027}
  </hea
                                                                   Draw polylines on the map (also
            var flightPath = new google.maps.Polyline({
  <body
                                                                   working for the points, polylines,
              path: flightPlanCoordinates,
    <di
                                                                   polygon circles, custom objects)
              geodesic: true,
              strokeColor: '#FF0000',
              strokeOpacity: 1.0, Link the 4 points into one polyline
              strokeWeight: 2
            });
            flightPath.setMap(map);
       </script>
    <script src="https://maps.googleapis.com/maps/api/js?key= YOUR_API_KEY &callback=initMap</pre>
   async defer></script>
  </body>
</html>
```

## Simple Polylines



```
<!DOCTYPE html>
<html>
  <head>
    <title>Simple Map</title>
    <meta name="viewport" content="initial-scale=1.0">
       <script>
         // This example uses a GroundOverlay to place an image on the map
         // showing an antique map of Newark, NJ.
         var historicalOverlay;
         function initMap() {
           var map = new google.maps.Map(document.getElementById('map'), {
             zoom: 13,
             center: {lat: 40.740, lng: -74.18}
                                  Georefencing (define the boundary
           var imageBounds =
                                  of overlay image)
             north: 40.773941,
             south: 40.712216,
  </hea
             east: -74.12544,
  <body
             west: -74.22655
           historicalOverlay = new google.maps.GroundOverlay(
               'https://www.lib.utexas.edu/maps/historical/newark_nj_1922.jpg',
               imageBounds);
           historicalOverlay.setMap(map); Overlay the custom map
       </script>
    </script>
    <script src="https://maps.googleapis.com/maps/api/js?key= YOUR_API_KEY&callback=initMap"</pre>
   async defer></script>
  </body>
</html>
```

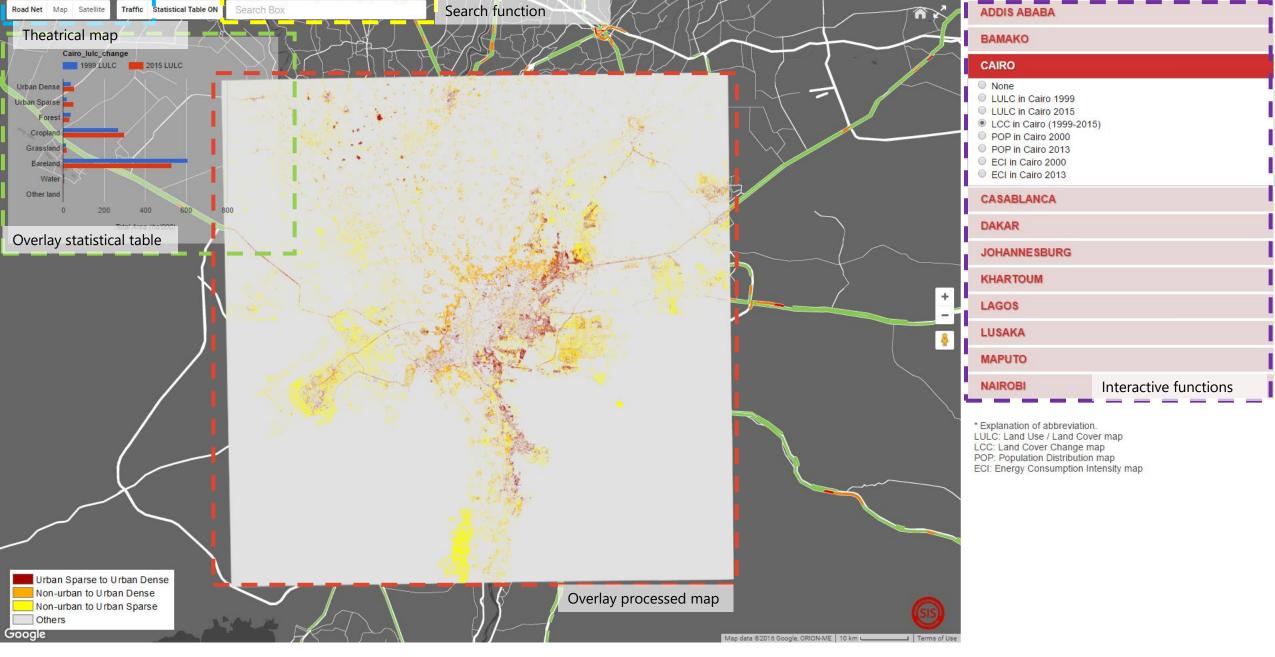
### Ground Overlays



# Application

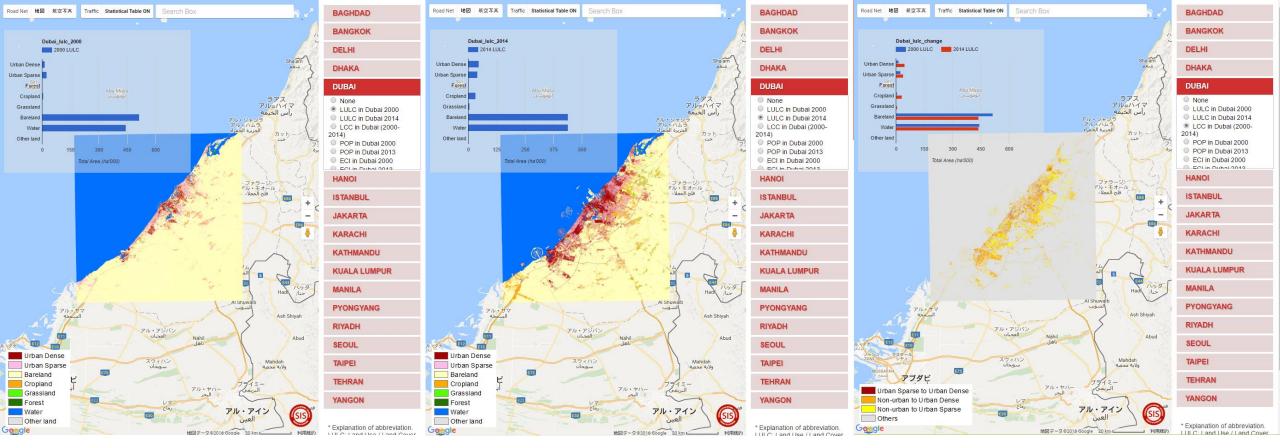
Megacity Project





Based on the integration of small functions, people could establish their own WEB-GIS services with the GIS dataset.

<sup>\*</sup> Screenshot from the homepage of Megacity Project WEB-GIS website. (http://giswin.geo.tsukuba.ac.jp/mega-cities/)



LUC map for Dubai 2000

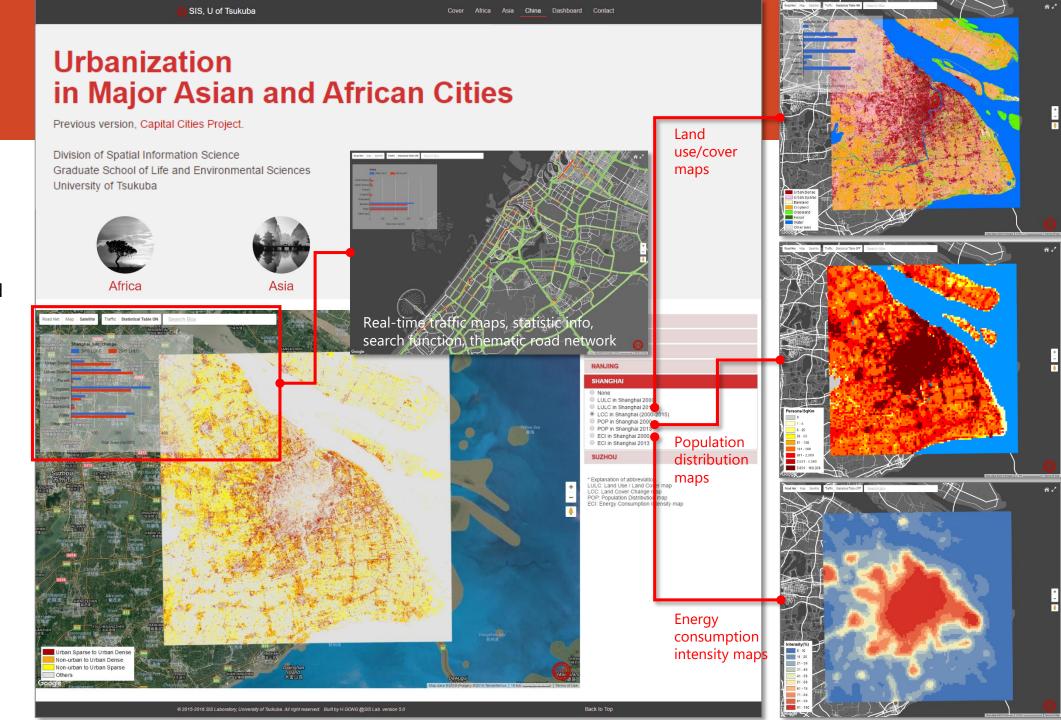
LUC map for Dubai 2014

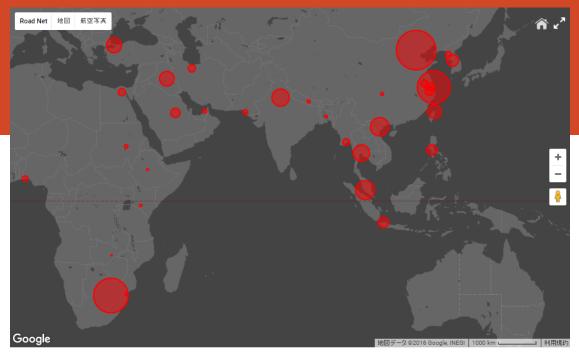
LUC change map for Dubai (2000-2014)

With the help of WEB-GIS services, an open access web-based GIS platform (Megacity project) was established. 35 major cities in Asian and African regions (Fig.3) were selected to examine the urban sprawl of mega-cities during the 15 years (2000-2014). In order to quantify and measure the urbanization process, all of these cities were separated into three groups (African cities, Asian cities and Chinese cities), and the size of the target area is a 100 × 100 km extent for each city.

Meanwhile, users could check the energy consumption intensity maps, population distribution maps, statistic information and thematic road network of study areas in our project.

Furthermore, benefit from the Google Javascript APIs, users can check the real-time traffic maps for study area in our Project.

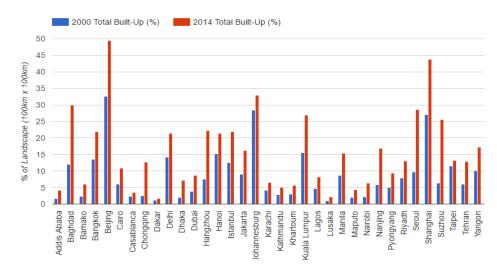


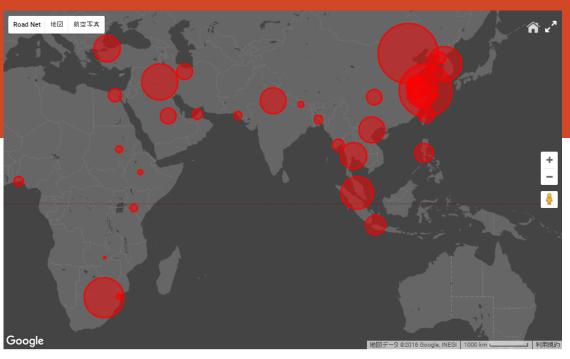


The rate of LULC types in each study area (100km \* 100km)

LULC (~2000) | Guilt-up | Urban Dense | Urban Sparse | Forest Cropland Grassland Bareland Water |
LULC (~2014) | Guilt-up | Urban Dense | Urban Sparse | Forest Cropland Grassland Bareland Water |

Clear All Markers



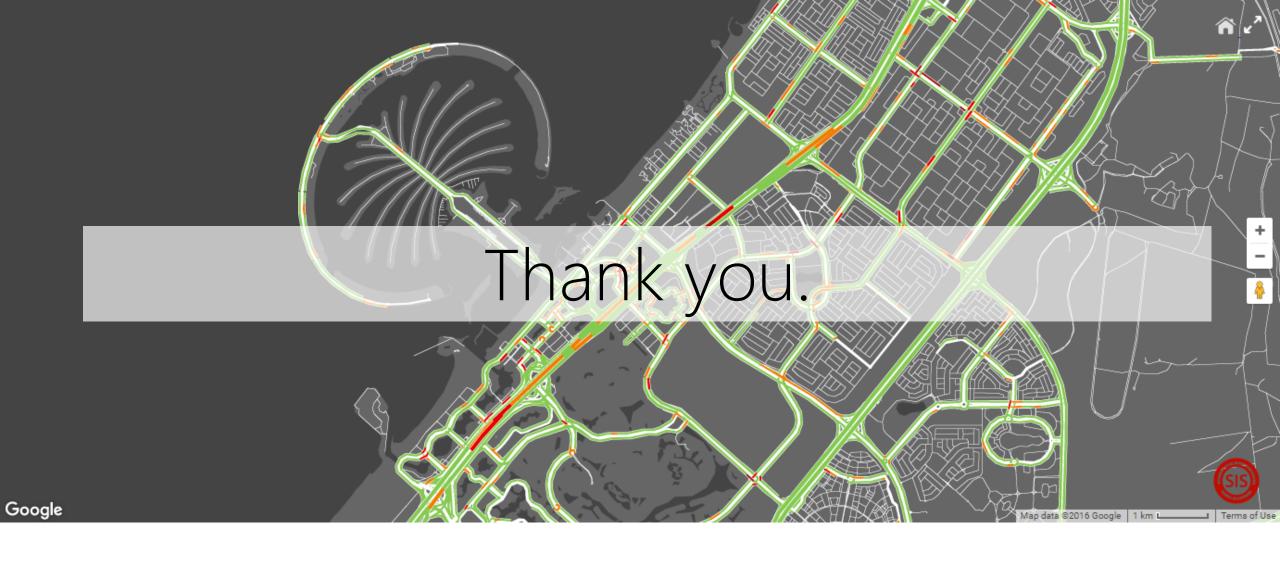


The rate of LULC types in each study area (100km \* 100km)

LULC (~2000) Built-up [ Urban Dense ] [ Urban Sparse ] Forest Cropland Grassland Bareland Water
LULC (~2014) Built-up [ Urban Dense ] [ Urban Sparse ] Forest Cropland Grassland Bareland Water

Clear All Markers

In the analysis page, people could compare the scale of urban growth process in a global view. On the other hand, users could download the dataset that they want, and use it for their own approaches.



Development of WEB-GIS based on Google Maps Javascript API

Instructor - Prof.Dr.Yuji Murayama Teaching Assistant - GONG Hao

Division of Spatial Information Science Graduate School of Life and Environmental Sciences University of Tsukuba