

SECE



Geoloc

Riddhi Mehta
MS Computer Science
Columbia University
UNI: rnm2119

Under the Guidance of:
Prof. Henning Schulzrinne
Jan Janak

Table Of Content

1. Introduction
2. Overview of Geoloc
 - 2.1 Challenges
3. Geoloc Implementation
 - 3.1 Google Maps API
 - 3.2 Outdoor SmartObject Mapping
 - 3.2.1 Adding New smartObjects
Fig 4: SECE Geoloc Outdoor maps add smartObjects page view(Including Street View)
 - 3.2.2 View smartObjects
 - 3.2.3 Deleting New smartObjects
 - 3.3 Indoor SmartObject Mapping
 - 3.3.1 Adding new smartObjects to Indoor maps
4. Conclusion
6. References

1. Introduction

Geoloc is a functionality within SECE which has been implemented in order to provide the users the ability to locate the user's smart objects within the sense everything control everything system by means of geographical co-ordinates. It is one of the many user interfaces that have been created to make the use of the SECE system much easier especially for non-technical users. The basic functionality that this feature tries to provide to the user of SECE is to be able to locate and view any smartObject - what we call any electrical device that the user might own and would like to link to the SECE system - using the geoloc UI. This UI shows the smartObjects mapped in the SECE system on google maps^[1], street-view^[2] and indoor-maps. This shows the user the location of their devices on maps and also on street view giving them a look-and-feel of actually being able to view and place the object in the exact location it is in physically. One of the objectives kept in mind for the UI with the google maps (outdoor mapping) is provided so for assistance with distance based queries submitted into SECE. Besides google maps for mapping smartObjects externally we also provide users with the functionality of indoor mapping using floor plan images uploaded by users. Using the indoor maps the user will be able to put in compound queries of the form "turn off all smartObjects in the Living room".

One of the biggest challenges in the development of the outdoor version of this feature was the ability let the user visualize the actual location as seen in the google street view and be able to distinguish perfectly between two smartObjects and their location. The ability of locating of the smartObject as seen from the street level instead of approximating its position by placing it from the overhead view provided by normal google map makes this system much more intuitive and also allow the user to provide more accurate location information to SECE.

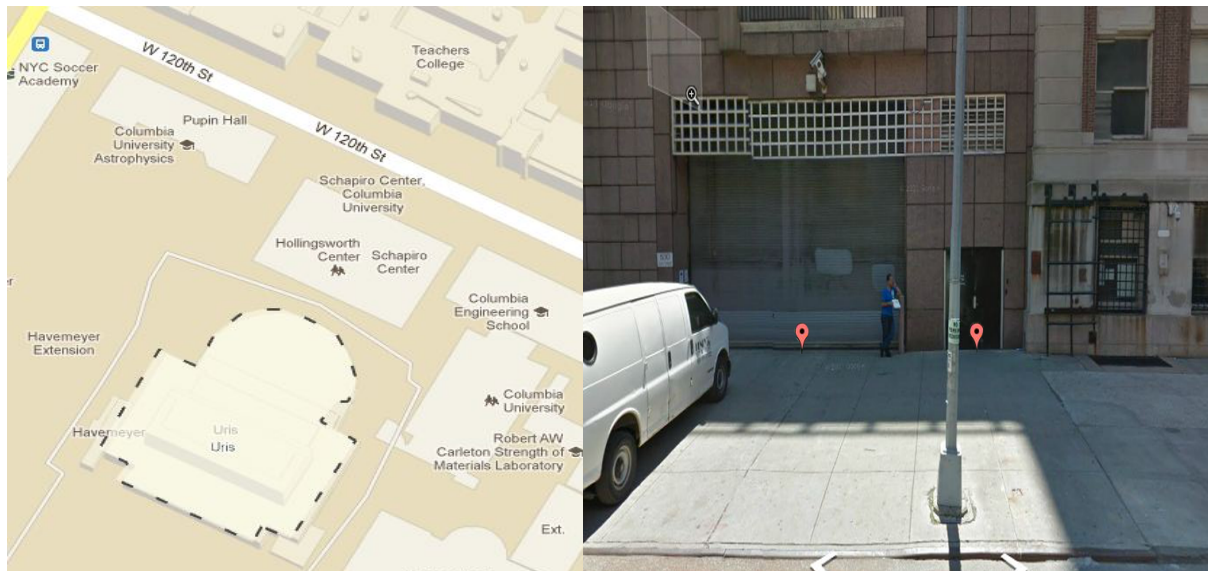


Fig 1: Being able to distinguish between the two doors in the street view as opposed to the overhead google map

Another challenge faced in the implementation of this system, was in the indoor version wherein, how to translate pixels values on the floor-plan uploaded by the user into the system to latitude-longitude. This in order to make the indoor-maps fully compatible with the outdoor google maps feature as well as make it compatible to the distance related queries the user might put into the system. I will go into greater detail regarding these challenges and how they were solved in the following sections.

2. Overview of Geoloc

As per the purpose of geoloc the web interface basically provides a simple user interface which allows the user to view & add delete objects that are owned by them and are linked to the SECE system.

Once the user logs into the system the smartObjects owned by this user are initialized and shown on the google map in the home for the user. The location of the individual smartObjects of the user is indicated on the map using a marker^[3]. All the objects owned by the user are also displayed in the form of a list for the user in a side panel giving information of its name as provided by the user and it's geo-location(i.e. latitude, longitude). The geolocation is obtained when the user clicks on the google map to drop a marker on the map.

A new object can be added to the SECE system through geoloc by simply providing the details for this object and placing the marker thus rendered at the user specified location in the google map or street view to provide the user with the ability to place the objects in their exact physical location.

2.1 Challenges

Some of the challenges that were faced while implementing the above features were the ability to obtain the exact latitude longitude from the street view once the user provides the address at which the smartObject is present. Using the google maps API once the user drops the marker on the main google maps the street view of the location are the coordinates of the center of the street and not the exact spot which the user actually intends.

The first possible solution we could apply in this case was to use the street view attributes that are maintained by the google maps API which are the street view Points of View or POVs^[5]. These POVs include the attributes pitch, heading and zoom. Although using these attributes we would be able to locate and distinguish between objects within google maps, this approach is not universal and would not have any meaning without the google maps API.

Another solution was to calculate the distance between the center of the street and the building in question and thereby calculate the actual location of the smartObject based on the heading angle^{[8][9]} and the latitude longitude of the point perpendicular to the marker in the middle of the street on the building which can be calculated based on the latitude, longitude of the edge of the buildings. Heading^{[8][9]} is the rotation angle around the camera locus in degrees relative from true north.

This solution was not explored as we found a way by means of which the markers placed on the map can be moved so as to mark the exact location of the smartObject in the street view and provide the system with the exact location of the smartObject by simply taking the location of the marker.

3. Geoloc Implementation

3.1 Google Maps API

Google maps Javascript API V 3.8 has been used to implement geoloc. The various services that are provided by the google maps API such as the geocoding service, street view service & geometry service has been used in this system. The complete reference to the google maps JavaScript API can be found here^[6]. The following functionalities use the google maps API and the aforementioned services.

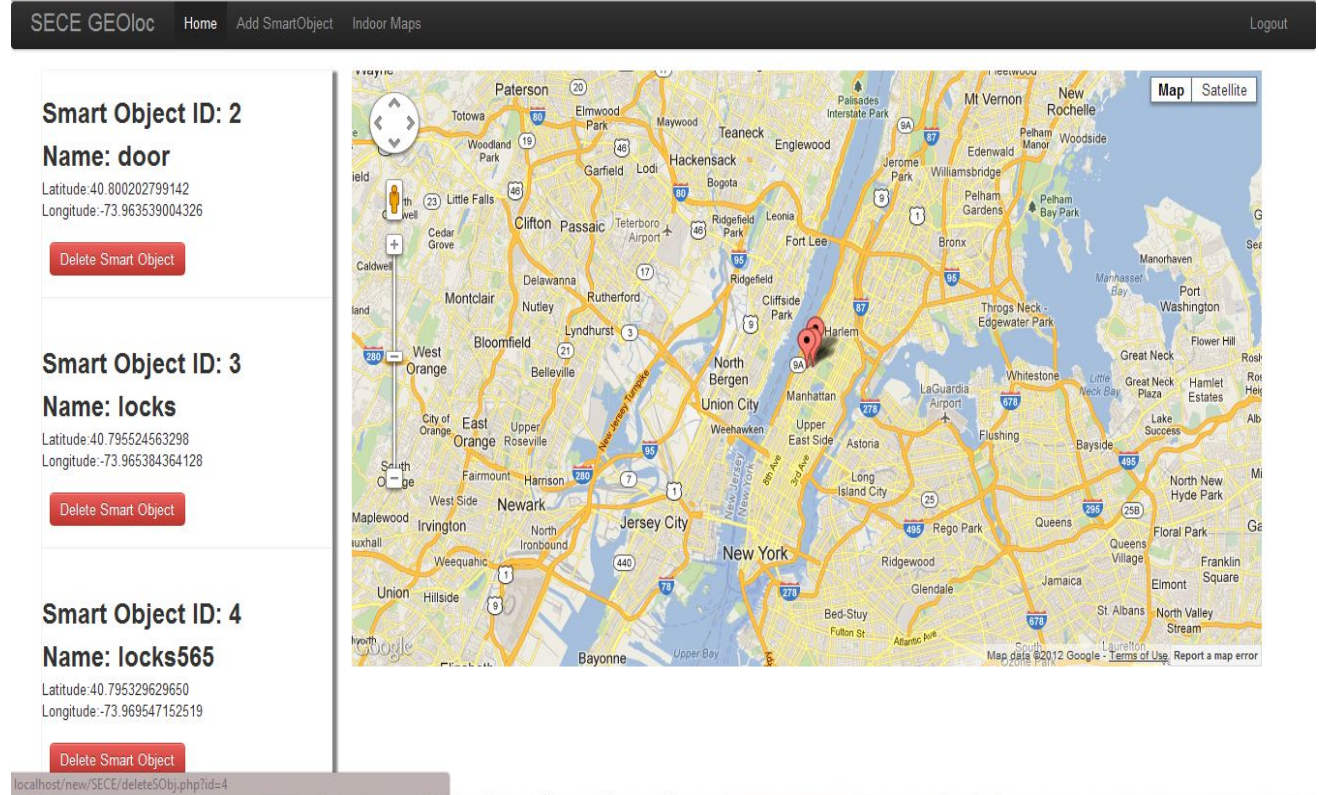


Fig 2: SECE Geoloc Outdoor maps home page view

3.2 Outdoor SmartObject Mapping

3.2.1 Adding New smartObjects

The first and most important functionality that is provided by the geoloc is the ability of the user to add new objects and locate them on google maps and street view. Once the user enters the name of the smart object to be placed the name appears in the side pane after which the user can click on the map at the location where the actual smartObject is present and drop a marker. If google streetview is available in this location then the web page also displays the streetview so that the user can adjust the streetview marker by dragging and dropping it, to point exactly to the position where smart object is located.

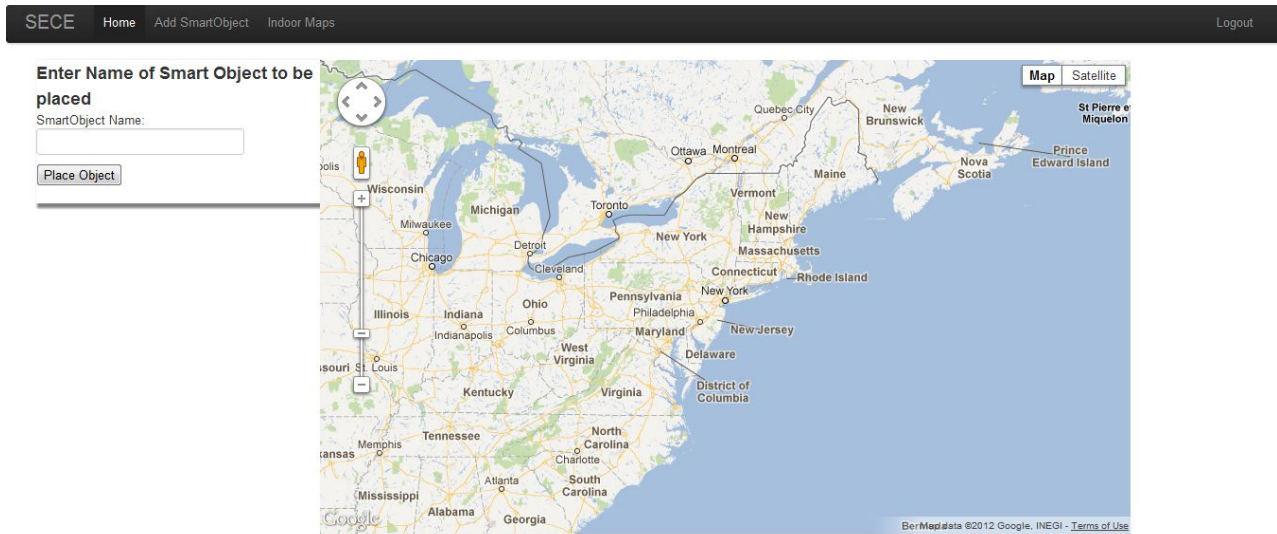


Fig 3: SECE Geoloc Outdoor maps add smartObjects page view

The user can now drag and drop the marker displayed both in the overhead map as well as street view to point to the exact physical location of the smart object. Once the object is placed the user can confirm this position as the location of the smartObject and the geoloc interface stores the latitude, longitude pointed to by the maker into the database containing the smartObject's information as its current location.

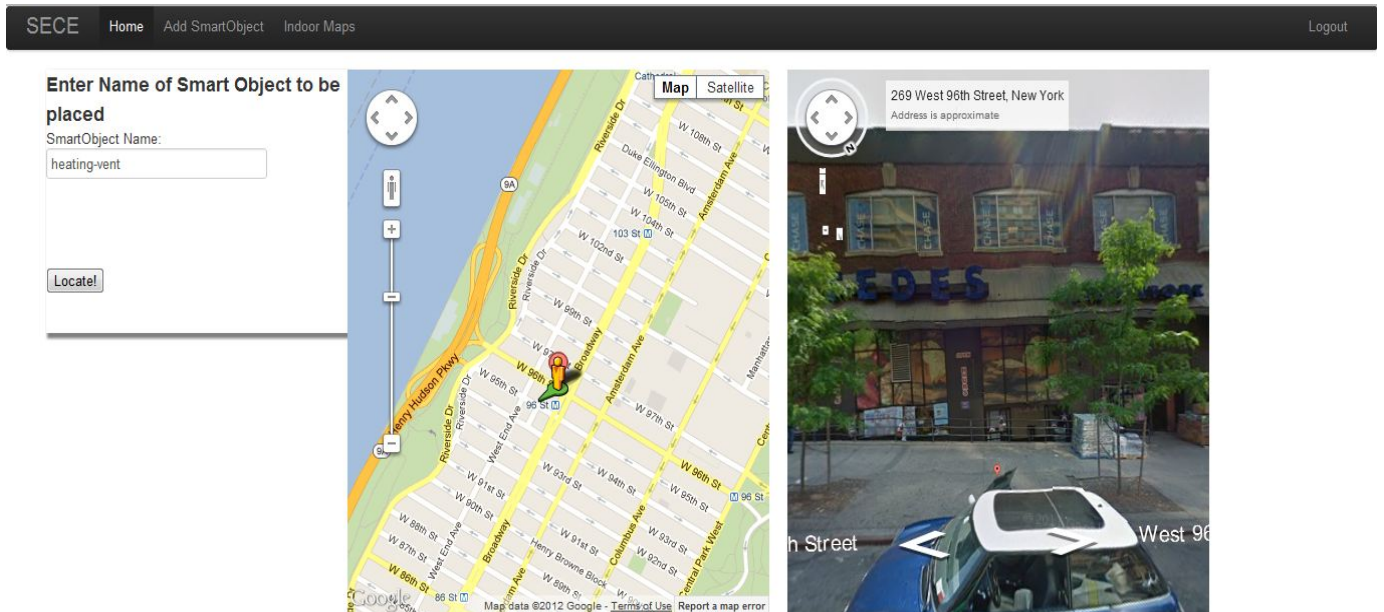


Fig 4: SECE Geoloc Outdoor maps add smartObjects page view(Including Street View)

3.2.2 View smartObjects

Once the user has logged into the SECE system they can navigate to the geoloc page wherein they can view the current position of all their objects. The database contains an attribute called “info” which is currently unused but can contain information related by the smartObject to the system which can be shown in the geoloc UI by means of a infowindow. The interface also has a list of all the smartObjects that are owned by the user.

3.2.3 Deleting New smartObjects

As mentioned before in the home of the geoloc system all the smartObjects that are owned by the user are displayed. Along with the listing is also the link to delete a particular object from the user’s profile.

3.3 Indoor SmartObject Mapping

Using outdoor maps this system allows smartObjects that lie outdoors precisely by viewing the exact location using street view. However, it does not allow the user to precisely place indoor smartObject. Besides this simply by storing the geographical location of the smartObject only allows the SECE system to let users specify distance related rules and not other general rules such as “turn on all smartObjects in the kitchen”. As we will discuss further the indoor maps provide a solution to a number of these issues.

In order to implement this indoor maps feature I explored the possibility of using google indoor maps however, google indoor maps is available to use only on android devices using google navigate and also indoor maps are available only for public areas which will not be of importance to most users would rather map smartObjects which are located within their own personal premises. So we decided that the best approach was to come up with a working model of a indoor maps application without the use of any external API except google maps API to get latitude and longitude values during initial setup.

The indoor maps UI allows the user to upload the floor-plan for the location into which they want to add and control smart objects. The system then asks the user to select 2 points on the uploaded floor plan and their corresponding locations on the google map which is displayed along with the floor-plan image. Once the user marks these two points based on their longitude latitude and their corresponding pixels on the floor-plan image, the system can calculate the longitude latitude of any other point which are selected by the user to add a smart object in this location. The indoor maps UI not only stores the longitude, latitude of the smart object added as well as the their location within the floor-plan for example store that a particular smartobject “lamp” lies in the area called “living-room”. These areas in which the smartobjects can belong can be specified by the user.

Thus, indoor maps provides the ability to support a wider range of queries as well as being able to provide the users with the ability to place indoor smart objects precisely.

3.3.1 Adding new smartObjects to Indoor maps

One of the biggest challenges to the indoor maps extension of this system was the calculation of the longitude, latitude values for the new objects which are placed in the indoor maps since such information is not tied to the image itself.

One of the possible solutions that was explored was using the open source google overlay tiler^{[10][11]}, through which the user has to modify and place the image on the map every time they login and then place their indoor objects. Since the image would be directly overlaid on the google-map, the longitude, latitude for this position would be available through the google maps API. However, the problem with this solution was that the user would have to upload, resize and overlay the floor-plan image every time they needed to place/ delete a smartObject to/ from the SECE system.

Therefore, in order to make the task for the user to interact with the system simpler I came up with an alternative. Through this alternative, the user only the first time they upload the indoor image to the system mark 2 points on the floor-plan image and mark the corresponding points on the google map displayed next to the floor-plan. Since, the image is placed at a constant location on the SECE geoloc webpage the rest of the pixels on the image can be mapped to their corresponding to their respective latitude-longitude values with reference to the 2 points marked initially by the user.

Based on the 2 latitude longitude and the corresponding pixels values available the tool gets the heading (i.e. the angle between the line joining the two points selected by the user and the north pole) using the google maps API, geometry library^[9]. Once the tool has the heading any the angle between this line and the line made by any other point on the map is added/subtracted from the heading angle and based on its distance from the first point it's latitude longitude is calculated.

Once the actual geographical distance is calculated every time a new point is selected indicating the position of the new smartObject, the pixel distance is calculated and converted into the corresponding geographical distance by equating the sample distance of the 2 points in the setup. Using this distance and the heading of the new location the latitude longitude of the new smart Object is calculated.

Besides this the user also provide in the form for indoor maps the room in which the smartobject belongs as well as the altitude of the floor-plan provided.

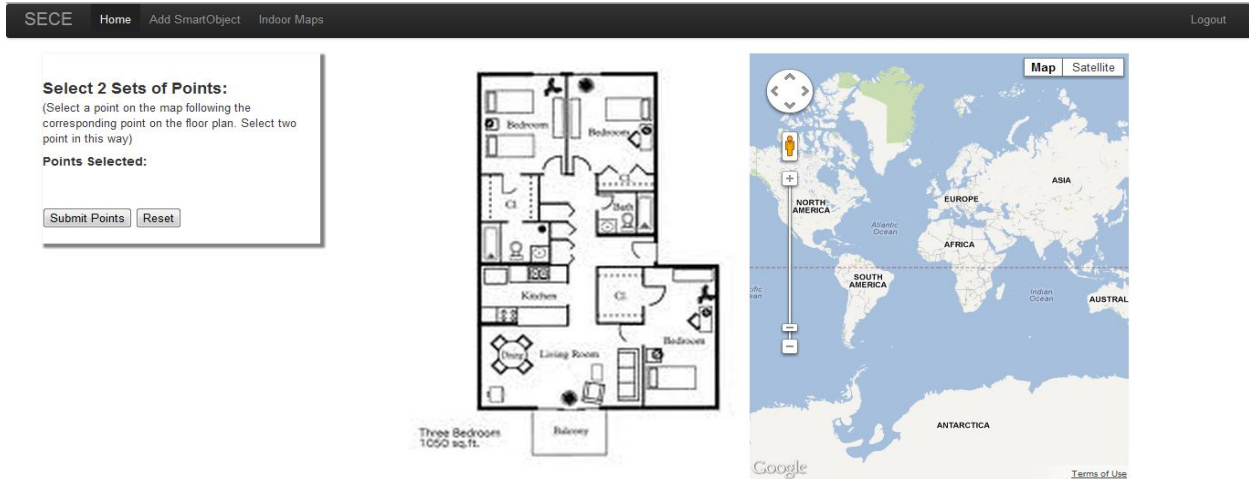


Fig 5. Mapping pixels on fixed position floor-plan with latitude longitudes on global map

The new smartObjects are displayed in the indoor maps by means of a side-pane as well as a indoor marker indicating the position of the smartObject on the floor-plan

The smartObjects added to indoor maps can be deleted using the icon on the side pane.

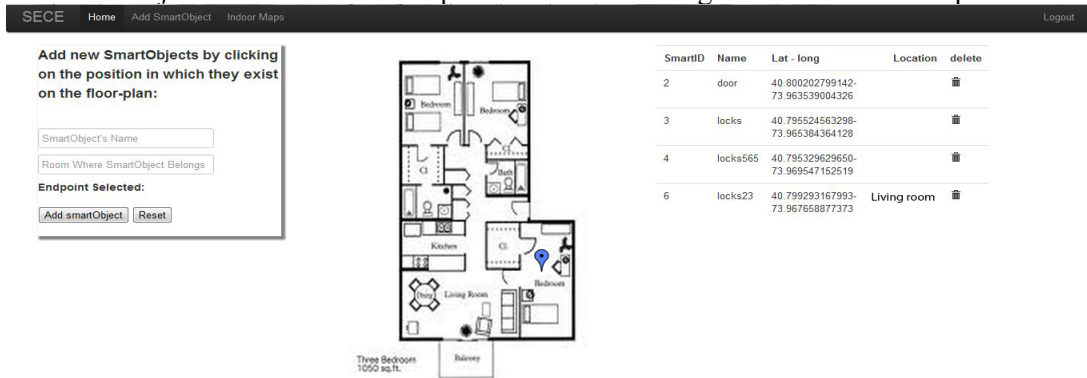


Fig 6. Indoor maps with marker indicating smartObjects placed by the user

4. Conclusion

Based on the current implementation I hope to provide a user with a new UI for the SECE system but also means for the system to accommodate new kinds of queries to be put in using the location information available to the system through the geoloc feature.

There are of course a number of things we would like to further add to this system. The most important one being the ability to add multiple floor plans for a single user thereby making it possible for this feature to be accessible for the user in a number of locations.

6. References

- [1] <http://maps.google.com/>
- [2] http://maps.google.com/intl/en/help/maps/streetview/#utm_campaign=en&utm_medium=van&utm_source=en-van-na-us-gns-svn
- [3] <https://developers.google.com/maps/documentation/javascript/overlays#Markers>
- [4] <https://developers.google.com/maps/documentation/javascript/geocoding>
- [5] <https://developers.google.com/maps/documentation/javascript/streetview#StreetViewPov>
- [6] <https://developers.google.com/maps/documentation/javascript/reference>
- [7] Twitter Bootstrap: <http://twitter.github.com/bootstrap/>
- [8] Google maps API, geometry library: <https://developers.google.com/maps/documentation/javascript/geometry>
- [9] Heading, Longitude & Latitude Calculations and definitions: <http://www.movable-type.co.uk/scripts/latlong.html>
- [10] Google maps Overlay Tiler: <http://code.google.com/p/overlay-tiler/>
- [11] Google maps tiler: <http://code.google.com/p/maptiler/>
- [12] <http://www.stackoverflow.com>