Università
della
Svizzera
italiana

Faculty
of Informatics

Bachelor Thesis

June 20, 2017

A Voxel–based Approach to Earthquake Simulation

Leonardo Iandiorio

Abstract

Earthquakes are a major issue in Italy with an average of 2000 earthquakes per year. The "Istituto Nazionale di Geofisica e Vulcanologia" (INGV) provides a huge dataset containing all the information about earthquakes occurred in Italy since 1985. The problem with the provided data is that they are static and disaggregate, and the INGV service does not provide any dynamic visualization.

We propose as a solution Visual Earthquakes, a Web Application which provides an interactive and dynamic way of visualizing earthquakes in a 3D environment modeled with voxels. It offers the visualization of hypocenters and also shows the arrival order of the seismic waves to the seismographic stations and their propagation.

We illustrate with some use cases how the visualization of hypocenters works and how the user can visualize the propagation of earthquakes through the terrain.

Advisor
Prof. Dr. Michele Lanza
Assistant
Dr. Andrea Mocci

Advisor's approval (Prof. Dr. Michele Lanza): Date:

Contents

List of Figures

1 Introduction

Italy has an important seismic history due to the fact that two fault lines run through it. Almost 300'000 earthquakes occurred since 1985. These seismic events are registered by the INGV, but such service doesn't provide an easy way to visualize their data. In general there is a lack of web applications showing understandable and interactive earthquake's data. For instance INGV monthly produces static images showing the major earthquakes occurred.
Therefore, Visual Earthquakes aims to fill this lack, providing to the everyday user the possibility to visualize and interact with earthquakes in a 3D environment modeled with voxels and to see how they propagate.
In this section we will introduce the concept of earthquake, describing its characteristics and the measurement techniques, and after that we will explain what is a voxel and what they are used for.

1.1 Earthquakes

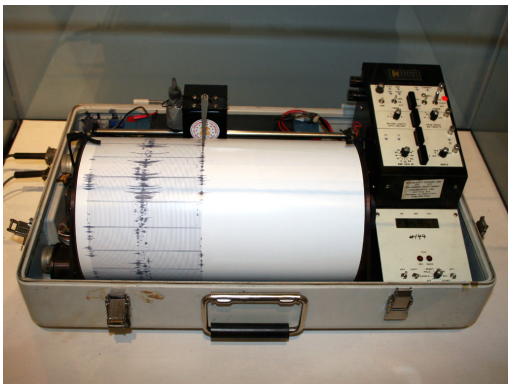
An earthquake is a geological event that occurs when two tectonic plates slide and create a fault in the Earth's crust. Such faults are classified by their movement:

- **Dip–Slip Faults**: the movement of the blocks is vertical and they can be divided into:
  - **Normal–Faults**: the block above the fault moves down with respect to the block below the fault.
  - **Reverse–Faults**: the block above the fault moves up with respect to the block below the fault.
- **Strike–Slip Faults**: the movement of blocks it's horizontal and it can be or right–lateral or left-lateral, depending on where the block on the far side of the fault moves.
- **Oblique–Slip Faults**: is a combination of both Dip–Slip and Strike–Slip faulting resulting in an oblique movement.

The location where the earthquake starts is usually below the earth's surface and is called the **hypocenter**, while its projection on the surface is called the **epicenter**.

1.1.1 Earthquake Instrumentation

Earthquakes are recorded using **seismometers** which consists of a pendulum hanging on a support base. The pendulum is composed by a weight and a spring plus a recording system (in the early seismometers a pen was used). The simplest ones measure only the vertical axis while professional ones measure in all three axes. The product of a seismometer is called the **seismogram** and is used to determine the size of an earthquake. By measuring the wiggles produced by the seismometer, is possible to obtain the magnitude of the earthquake.



(a) Old version of a seismometer.

(b) Newest version of a seismometer.

Figure 1.1. Examples of seismometers.

1.1.2 Earthquake Magnitude

The magnitude of an earthquake can be measured using the Richter magnitude (also called local magnitude "M_L") which is determined from the logarithm of the amplitude of waves recorded by seismographs according to the following formula:

$$M_L = \log_{10} A - \log_{10} A_0(\delta)$$

where A is the maximum excursion of the seismograph measured in micrometers and $A_0(\delta)$ is the empirical function which depends on the epicentral distance δ.
Other than Richter magnitude we have others types of magnitude depending on the distance of the station or depending on the type of the earthquake or its size:

- Body Wave Magnitude M_b for earthquakes registered at a distance greater than 600km.
- Surface Wave Magnitude M_s for shallow earthquakes (most superficial ones).
- Moment Magnitude M_w based on the seismic moment generated by the rupture of the earth's crust.
- Duration Magnitude M_d which compare the length of the seismic wave on the seismogram to the earthquake's size.

1.1.3 Earthquake Waves

We can distinguish earthuakes waves between **body waves** and **surface waves**:

- Body waves travel through the interior of the Earth and the two most important ones are:
  - Primary waves (P–waves) which are compressional and longitudinal waves that travel faster than the others and are the first to arrive to the seismograph stations.
  - Secondary waves (S–waves) which are shear and transversal waves. This kind of waves arrives at the seismograph stations after the P–waves.
- Surface waves travel along the Earth's surface and they travel more slowly than body waves.

1.1.4 Locating an Earthquake

To determine where an earthquake occurred, scientists use a method called *triangulation*. This method works as follows:

- Take three arbitrary stations that have measured the earthquake.
- Determine their distance from the epicentre using the S–P waves arrival time.
- Once the distance is known, draw a circle for each station, having the station as center and the distance from the epicentre as radius.
- The earthquake occurred at the point where all the three circles intersect.

1.2 Voxels

Voxels are defined as pixels in three dimensions. The third component of a voxel is the volume whence it takes its name: the word voxel, originated from pixel, combines the prefix "vo" representing "volume" and "el" which stays for "element". Voxels are widely used to visualize scientific data and to discretize volumes. Other common uses of voxels include the representation of terrains in simulations or volumetric imaging in medicine.
One the most famous application of voxel is Minecraft as we will further explain in section 2

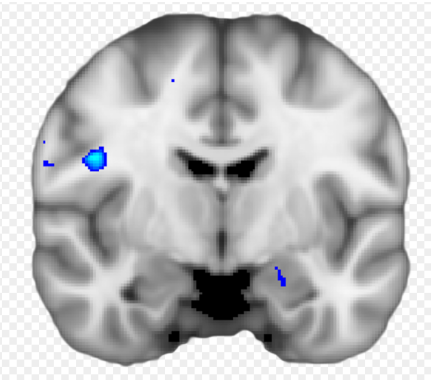


Figure 1.2. Example of voxel usage: Voxel-based Morphometry used in diffusion-weighted magnetic resonance imaging.

1.3 Structure of the report

- **Section 2** – is about the current state of the art. It introduces the existing solutions that provide earthquake visualization and simulation and also illustrate the current use of voxels on the web.
- **Section 3** – is the main section and explains in details how Visual Earthquakes application works. It describes how both client and server sides work, the technologies we used and our visualization choices.
- **Section 4** – deals with the interaction of the user with the application. With the help of some use cases it shows the functionalities of our application.
- **Section 5** – concludes the report describing the potential of the application and the possible future works.

2 State of the Art

In this section we will discuss the existing systems that let the user visualize earthquakes and their propagation. On the web there are not many applications that let the user interact with a map of a country and see the earthquakes happened in there. The majority of these applications are desktop applications so there is the lack of a web version of them.

2.1 Earthquakes Applications

One web application that half does these tasks is **Shake Finder**[1]. It is listed in the available demos of the Cesium[2] documentation. It uses the data collected by the United States Geological Survey (USGS) to display earthquakes querying the USGS using date and magnitude constraints. But it is very limited because it has very low performances and doesn't handle a big amount of data.

Figure 2.1. Interface of Shake Finder web application.

Another application is called **Paradigm Seismic Interpretation**[3] which is a modern interpretation and visualization system designed to fulfill a lot of seismic interpretation tasks like structural, stratigraphic, and volume interpretation. This application though is not easily accessible by the everyday user, because it requires a deep understanding of seismic events to interact with it.

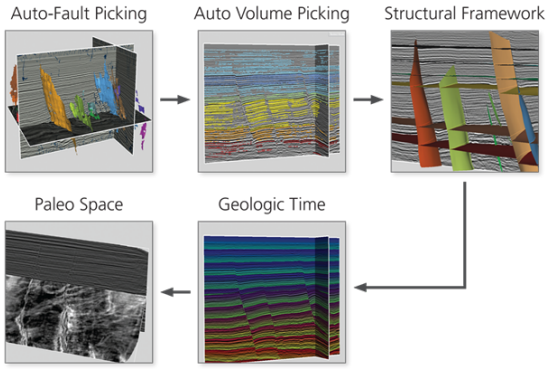


Figure 2.2. Paradigm workflow showing how the user can interpret faults, by auto–picking the seismic reflectors, and he can create a 3D model.

2.2 Voxels Applications

Regarding the use of voxels, the main applications we found are related to gaming. One of the most famous games that is based on voxels is **Minecraft**[4]. Minecraft is a sandbox video game which lets players go around a world modeled with voxels and lets them build and craft whatever they want always using voxels.

[1] See `https://danj707.github.io/shake-finder/`
[2] See section 3.1.2
[3] See `http://www.pdgm.com/solutions/interpretation-and-modeling/seismic-interpretation/`
[4] See `https://minecraft.net/en-us/`

Figure 2.3. Common landscape of the world in Minecraft.

Another example of use of voxels is the **Voxel Builder**[5] application. It is a web application that offers a 3D canvas where the user can draw objects with the only use of voxels. It is then possible to export your drawings in various formats and then print them in 2D or 3D.

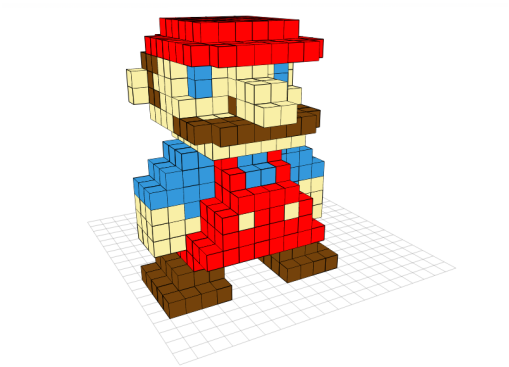


Figure 2.4. Example of a critter created with Voxel Builder.

[5] http://voxelbuilder.com/

3 Visual Earthquakes

In this section we illustrate the core and the structure of the Visual Earthquakes application. In the first part we will describe how our application looks like on the client side and the libraries used. Thus, we will explain how we got the data from our providers and how we modeled it.

3.1 Client Side

In the client side of our Web Application, we used standard web technologies such as Html, Css, and Javascript together with the jQuery library and the Cesium library.
This was not the first choice: At the beginning, we used the Leaflet library to manage the client side.

3.1.1 A first try: Leaflet Js

Leaflet is an open–source lightweight Javascript library for interactive maps. It renders the map you want using the GeoJSON format. GeoJSON is a data format to encode different geographic data structure. So for the first part we tried to build our own GeoJSON file using the QGIS[6] application. The result was a map like the one you see in Figure 3.1.

Figure 3.1. First try with leaflet js.

The solution using Leaflet was not really performant as it does not use the GPU to render the objects and so we had a limit of 2000 visualizable earthquakes. Moreover, it did not provide a 3d environment so we decided to abandon this idea, adopting the Cesium library.

3.1.2 Cesium

Figure 3.2. Cesium's logo.

Cesium is an open–source Javascript library for 3D maps and models. It provides a 3D globe where you can draw a huge number of geometries maintaining high performance using its GPU optimizations, due to the fact that Cesium relies on WebGL at its core. It also provides an high–resolution visualization of the terrain using different imagery

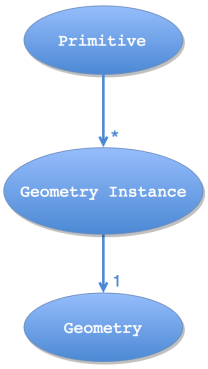
[6] Geographic Information System, see `http://www.qgis.org/en/site/`

layers from multiple sources like OpenStreetMaps[7], Bing Maps[8], or MapBox[9]. With its camera functions you can easily navigate, rotate and zoom through the globe but without letting you go inside the earth, thanks to its default terrain collision detection. Cesium also offers an high-precision coordinate system using the standard World Geodetic System (WGS84)[10]

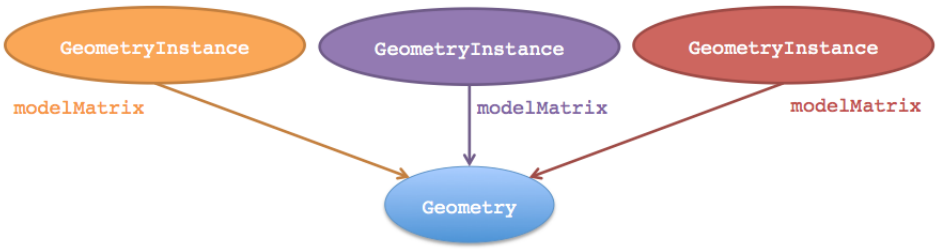
Figure 3.3. Cesium globe with MapBox imagery layer.

3.1.3 Integrating Cesium in our system

The Cesium environment where we draw our objects is called the **Scene**. There we draw geometries using the **Primitive** type. A Primitive represents a geometry in the scene and can be formed by a single or an array of **GeometryInstances**. A GeometryInstance is a container for a **Geometry** and it is also used to position, scale and rotate the same geometry in different parts of the scene. We can have multiple GeometryInstances that refer to the same Geometry, each instance with different attributes and different modelMatrix (which defines its position). The Geometry we use to represent our voxels is the BoxGeometry which defines a cube centered at the origin of the scene.
In this way, we have a huge performance benefit: Combining Geometries into GeometryInstances reduces the CPU overhead and lets us better utilize the GPU. Combining GeometryInstances with Primitives is done on a web worker to keep the UI responsive. A schema on how this hierarchy works is shown in Figure 3.4.



(a) Primitive relationship with geometries.



(b) How GeometryInstances refer to the same Geometry.

Figure 3.4. Hierarchy in the Cesium Scene.

With this optimization we achieved the goal of displaying more than 600'000 geometries on the web, maintaining a decent frame–per–second ratio.

[7] https://www.openstreetmap.org
[8] https://www.bing.com/maps
[9] https://www.mapbox.com/
[10] http://wiki.gis.com/wiki/index.php/WGS84

To further optimize the rendering of the scene, we made the following design choice:

> During the rendering of the 3D model of Italy, which is made up of $\sim 667'114$ voxels, it was not possible to render all the voxels at the same time. Therefore we decided to render Italy line by line, so that to start rendering we do not have to wait all the voxels but as the first line of Italy is ready, we start the render of the model from top to bottom, one line at time. This leads us to a minor waiting time and to a smoother effect as Italy is created.

3.1.4 Visualization Choices

The first choice we have to take is how to represent Italy. Having voxels with side length of 1km, it is not so easy to see the variations in the elevation. Therefore, we decided to exaggerate the heights to make a better visual effect of the different elevations of the ground. To better visualize this effect we also interpolate the color of each voxel by its height to recreate the classic shade of the physical maps. We also decided to elevate the model of Italy as much as the depth of the deepest earthquake we have, so that we can visualize the hypocenters of the earthquakes under the Italy's surface at the right depth.

Figure 3.5. 3D representation of Italy.

The second choice is how to visualize the hypocenters. We started thinking on visualizing each hypocenter as a single voxel, at the right depth, but it was not real user–friendly because, due to the small unit of the voxel and to the layer of voxels that creates the surface, the hypocenter was imperceptible. Therefore, we decided to modulate the hypocenters using a different number of voxels and a different shade depending on their magnitude:

- A single voxel for earthquakes of magnitude less than 4;
- A cube with side of 3 voxels for earthquakes of magnitude less than 5;
- A cube with side of 5 voxels for earthquakes of magnitude less than 6;
- A cube with side of 7 voxels for earthquakes of magnitude greater or equal to 6;

The color of the hypocenters is interpolated between light green and dark red according to the magnitude of the earthquake (from the smallest ones to the bigger ones).
Figure 3.6 shows the hypocenters of different magnitude located under the Apennine mountains.

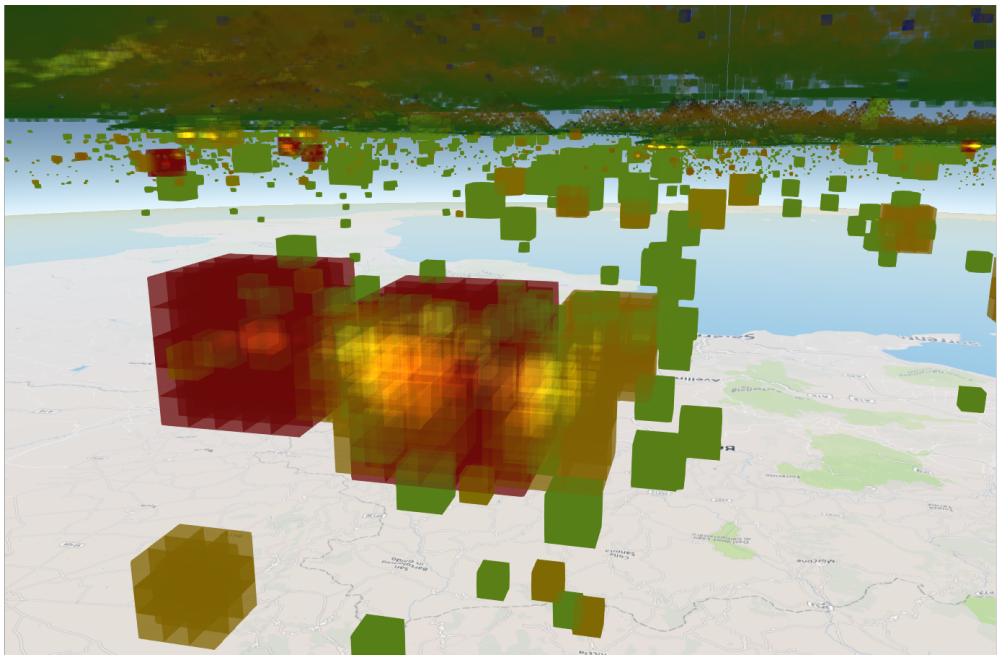


Figure 3.6. Detailed view of different magnitude hypocenters.

In Figure 3.7 instead you can see the seismographic stations which are used to measure earthquakes in Southern Italy. Our choice was to let the 1'138 stations always visible and blue shaded on top of the Italy model.

Figure 3.7. Part of the stations in Southern Italy.

3.2 Architecture and Data Modeling

Regarding the server side of our application we used the Java programming language combined with the Spring framework. We have two main services that retrieve the data from the INGV, one for the basic earthquakes and one for the propagation information. We also have a service to retrieve the elevation data from the Google Maps Elevations API to build the elevation matrix.
We collect our data into a MySQL database using a Data Access Object. Our database size is constantly increasing as we update it with the newest earthquakes. Table 1 shows the size of our database on June 20, 2017. The number of entries is approximate since we are constantly updating it with the newest earthquakes.

Table Name	Number of Entries
earthquake	~ 263'000
magnitude	~ 263'000
origin	~ 263'000
station_magnitude	~ 2'000'000
amplitude	~ 2'000'000
station	~ 1'000
elevation	1'384'449

Table 1. Entries in our database, on June 20, 2017

Before dealing with the implementation of the server side, Figure 3.8 shows the big picture of the architecture of our system.

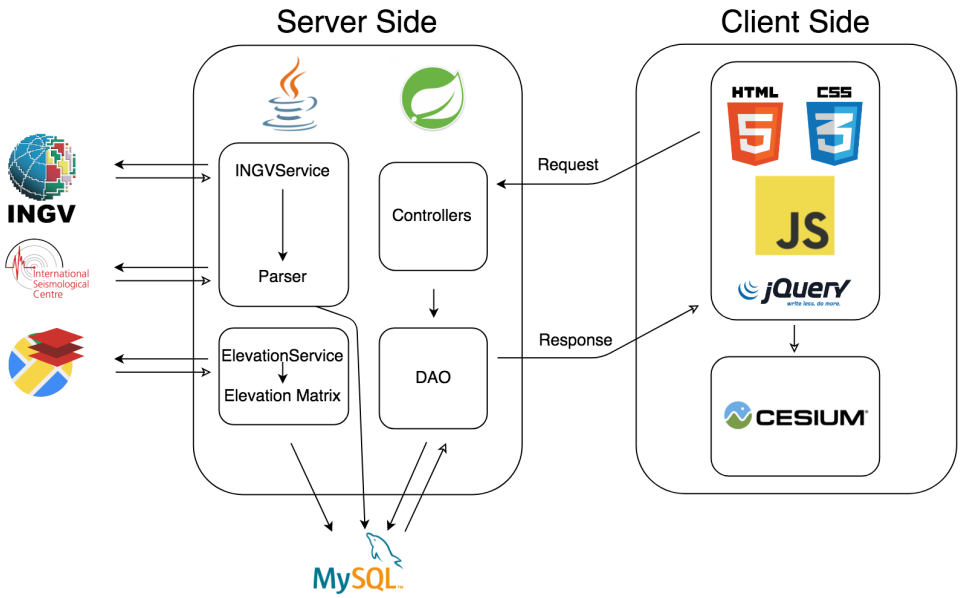


Figure 3.8. Architecture of Visual Earthquakes.

3.2.1 INGV

INGV is the acronym for "Istituto Nazionale di Geofisica e Vulcanologia" and it is our data provider. It offers a webservice that lets you access their earthquake's data in their archives, stored in 'QuakeML'[11] format. This webservice lets the user perform different queries, for example it lets you ask for the events happened in a time window or, knowing the id of the event, it lets you ask for additional information about that event.
The basic query format that we initially used is the following:

```
/query? [geographic-constraints] [depth-constraints] [temporal-constraints]
        [magnitude-constraints] [format-option]
```

where geographic–constraints represents the up–left and the bottom–right vertices of the rectangle containing the preferred area (in our case Italy), depth–constraints represents the minimum and maximum depth of the event, temporal–constraints represents the starting time and end time of the events, magnitude–constraints represents the minimum magnitude and maximum magnitude and format–option such as xml, text or kml (in our case xml).
The resulting xml contains all the events with the requested constraints.
Figure 3.9 shows an example of an event returned by the service.

[11] Xml format for seismic events, see `https://quake.ethz.ch/quakeml/`

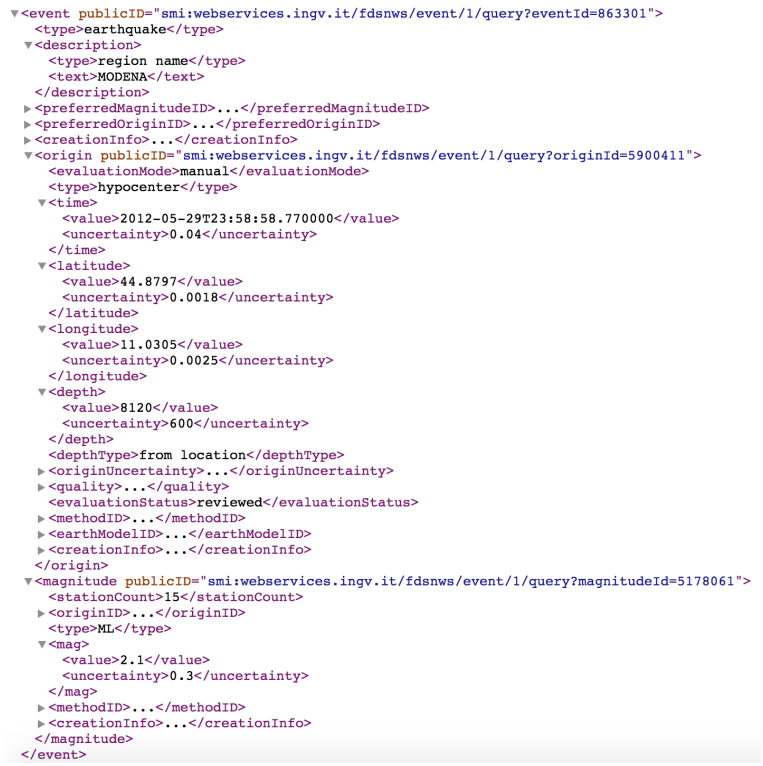


Figure 3.9. Example of event response from INGV webservice.

After having populated our database as we will explain in sub–section 3.2.3, knowing the id of every earthquake, we started to query for additional information using this query format:

```
/query? (id-options) [includearrivals] [includeallstationsmagnitudes] [format-option]
```

The result of this query is an xml containing all the information about a single event:

- **Station Magnitude** – Contains a station code and an amplitude id of the measurement.

- **Amplitude** – Contains the amplitude and the time of the measurement.

```
<stationMagnitude publicID="smi:webservices.ingv.it/fdsnws/event/1/query?stationMagnitudeId=amp-2918639">
 <mag>
  <value>0.0</value>
 </mag>
 <originID>
  smi:webservices.ingv.it/fdsnws/event/1/query?originId=755599
 </originID>
 <methodID>
  smi:webservices.ingv.it/fdsnws/event/1/query?methodId=1
 </methodID>
 <amplitudeID>
  smi:webservices.ingv.it/fdsnws/event/1/query?amplitudeId=amp-6447619
 </amplitudeID>
 <type>ML</type>
 <creationInfo>
  <agencyID>INGV</agencyID>
  <creationTime>2013-01-05T14:17:13</creationTime>
 </creationInfo>
 <comment>
  <text>original mag_type: ML</text>
 </comment>
 <waveformID networkCode="IV" stationCode="NRCA" channelCode="HHN" locationCode=""/>
</stationMagnitude>
```


(a) Station Magnitude Object


```
<amplitude publicID="smi:webservices.ingv.it/fdsnws/event/1/query?amplitudeId=amp-6447619">
 <genericAmplitude>
  <value>0.49822500000000003</value>
 </genericAmplitude>
 <period>
  <value>1.44</value>
 </period>
 <unit>m</unit>
 <category>other</category>
 <type>AML</type>
 <comment>
  <text>Wood-Anderson amplitude</text>
 </comment>
 <timeWindow>
  <begin>0</begin>
  <end>0</end>
  <reference>2009-04-06T01:32:50.81</reference>
 </timeWindow>
 <creationInfo>
  <agencyID>INGV</agencyID>
  <author>BULLETIN-SISPICK</author>
  <creationTime>2014-08-18T09:42:37</creationTime>
 </creationInfo>
 <waveformID networkCode="IV" stationCode="NRCA" channelCode="HHN" locationCode=""/>
</amplitude>
```


(b) Amplitude Object

Figure 3.10. Correlation between Station Magnitude and Amplitude.

Unfortunately INGV provides only a station code, so to retrieve the information about every station (i.e., location, elevation, and name) we have to use an external service[12]. Such service does not provide any 'easy–to–use' format but it simply returns an HTML page where we have to parse the inside plain text.
We collected these additional information only for earthquakes with magnitude greater or equal to three, because they are the most interesting ones regarding the propagation. Furthermore, the INGV service provides this data only for the most recent earthquakes, after 2006.

3.2.2 Google Maps Elevations API

In order to retrieve the data for the Italy model, we used the Google Maps Elevation API. Given a latitude–longitude coordinate, it returns the elevation for that coordinate. Here the problem is the huge number of request we have to do because to have a realistic representation of Italy, we decided to take a sample of the elevation at every kilometer. The standard Google service provides a free number of requests up to 2500 per day, but considering a minimum bounding rectangle of Italy, the number of samples exceeds the million. So we decide to use a secondary Google service that, given a starting point, an end point and a number of samples (with a maximum of 512), returns all the elevations for that path, taken at every sample. Doing this way, the number of requests drastically decrease because we can do two request for every kilometer going down along the height of the bounding rectangle and so the number of requests becomes much smaller, i.e., around 5000.

3.2.3 Modeling the Data

Earthquake's Data

We have a service that every 3 minutes asks to the INGV for new earthquakes. We parse the Xml returned by the INGV and we take the following information:

- Earthquake
- Magnitude

[12]International Seismic Center, see `http://www.isc.ac.uk/registries/search/`

- Origin

Earthquake is the main object and contains an earthquake id, a region name and two references, one to a Magnitude object and one to an Origin object. The Magnitude object represents the magnitude of the earthquake and contains a magnitude id, a magnitude value, the type of the magnitude (which can vary depending on how it is measured, as explained in section 1.1.2) and the uncertainty of the measurement. The Origin object represents the location of the hypocenter and contains an origin id, the depth of the hypocenter, its latitude and longitude, and the date of the earthquake.
Simultaneously, another service is running. For every earthquake in the database, it asks to the INGV for additional information such as StationMagnitude and Amplitude. The xml returned consists of a list of all the StationMagnitudes and Amplitudes for a single earthquake. Every StationMagnitude has a station code of the station that has taken the measurement, so for every station code, another service is triggered and search for the existence of such code in the database and if it doesn't find the code, it requests the station information to the ISC service. So we add 3 new models to our database:

- StationMagnitude
- Amplitude
- Station

The StationMagnitude is the main object and contains a stationMagnitude id, the magnitude value, a reference to an Amplitude object, a reference to an Earthquake object and a reference to a Station object. The Amplitude object contains an amplitude id, a value of the amplitude and the arrival time. The Station object contains a station id, the elevation of the station, its coordinates, and the station name.
Figure 3.11 shows the structure of our database with the relative relationships.

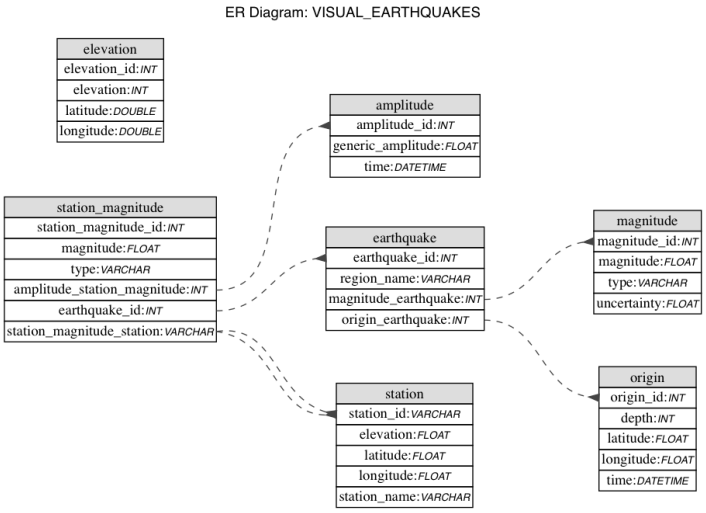


Figure 3.11. Structure of the database.

Elevation Matrix

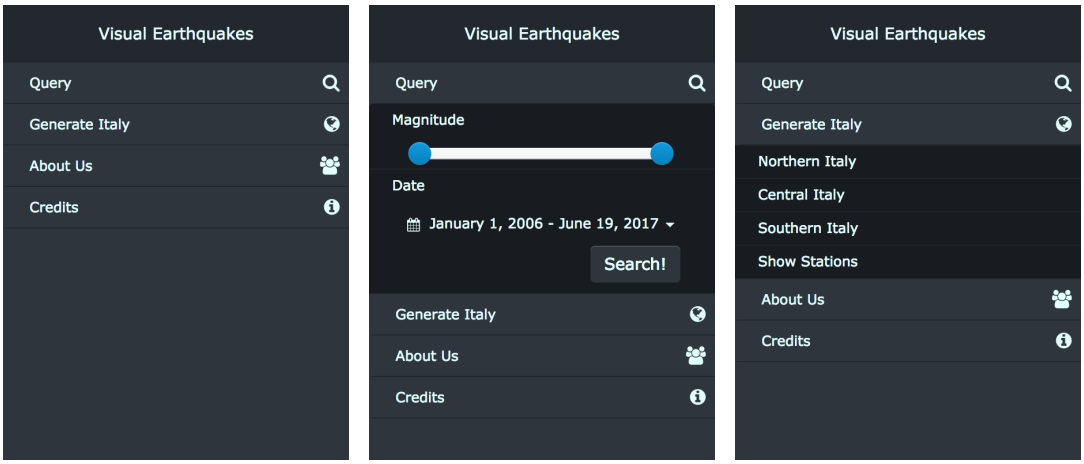
With the data from Google Maps, we built an elevation matrix of Italy.
We started by bounding Italy in a rectangle to construct a matrix. Since Earth is not flat, we cannot apply Euclidean geometry. Specifically, the distance between two points variates as you go from the earth's poles to the equator.
So, constructing our matrix, we had to take care of this and be sure that we were always decreasing the same distance.
The final result is a matrix where every row corresponds to one line of voxels.
In Figure 3.12 you can see the bounding rectangle we used to inscribe Italy.

Figure 3.12. Bounding rectangle of Italy.

4 Use Cases

Our application offers different use cases. In this section we will explain how the user can interact with our web application proposing different examples of usage. We will show how he can perform queries, create the 3D model of Italy and see how the earthquakes propagate.

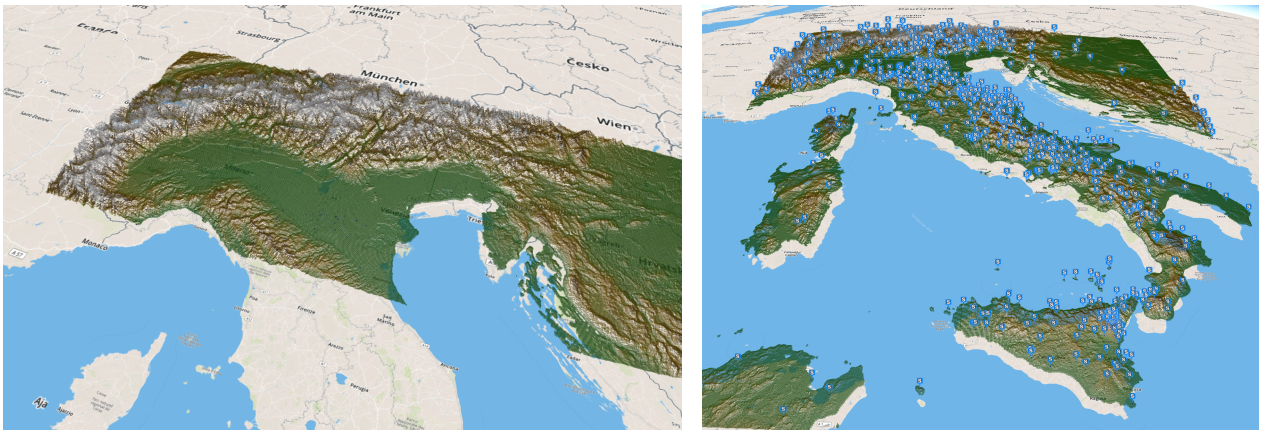
4.1 User Interface



(a) Detail of the menu. (b) Detail of the query interface. (c) Detail of the generate section.

Figure 4.1. The interactive menu of Visual Earthquakes.

Figure 4.1(a) shows the menu that the user can use to interact with the application. The user can first of all generate the 3D model of Italy, and then he can do queries to search for earthquakes.
The **Query** section works as shown in Figure 4.1(b). The user can search for earthquakes filtering them by magnitude or by date.
The **Generate Italy** section in Figure 4.1(c) lets the user choice what to generate. He can generate the whole model of Italy or just a part of it. In addition he can also choose to visualize a pin over every available station.



(a) Model of Northern Italy (b) All the stations highlighted

Figure 4.2. Results of the generate section.

4.2 Visualizing Hypocenters

For instance the query of Figure 4.1(b) produces as output the result (representing all earthquakes with magnitude between 3 and 10 and from the 1st of January until now) shown in Figure 4.3.

Figure 4.3. Visualization of Hypocenters under the Earth's surface.

Figure 4.4 shows an important seismic swarm under the Etna Vulcano in Sicily.

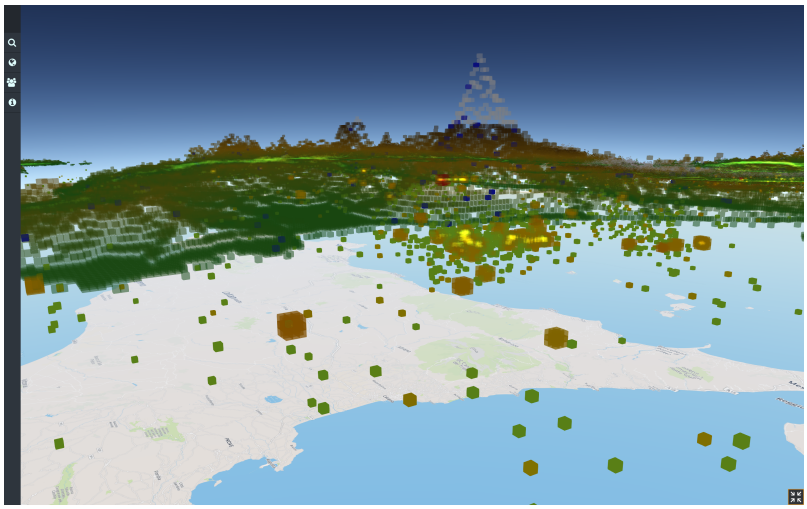


Figure 4.4. Seismic swarm under Etna volcano.

Figure 4.5 illustrates a portion of the hypocenters under Calabria's surface.

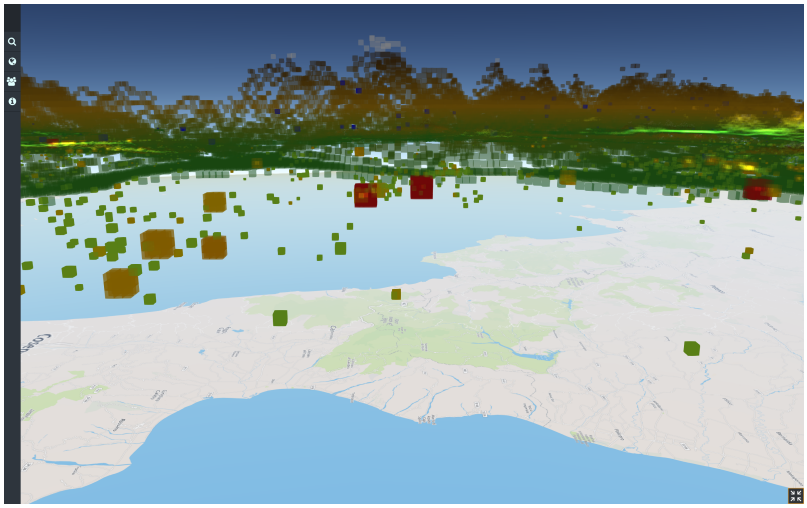


Figure 4.5. Hypocenters under Calabria's surface.

4.3 Interacting with Hypocenters

Once the user have selected the earthquakes to visualize, he can interact with the hypocenters by clicking them. An info box will appear, showing the relevant information about the earthquake.
The information we that we display is:

- The Magnitude and the Type of the selected earthquake measured using one of the different methods.
- The Date and Time at which it occurred.
- The Depth expressed in kilometers of the hypocenter.

Figure 4.6 shows the info box regarding the Central Italy earthquake happened on October 30, 2016.

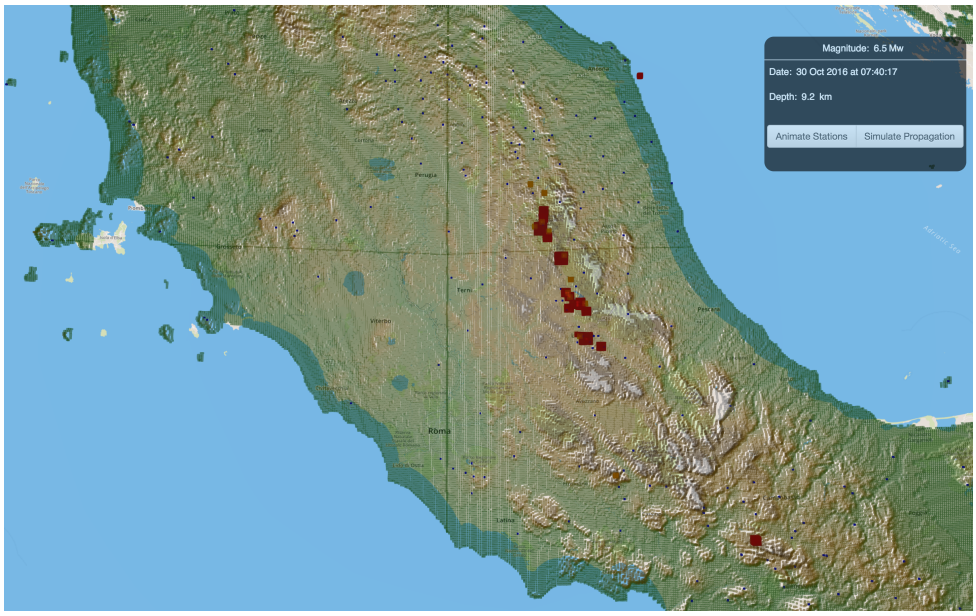
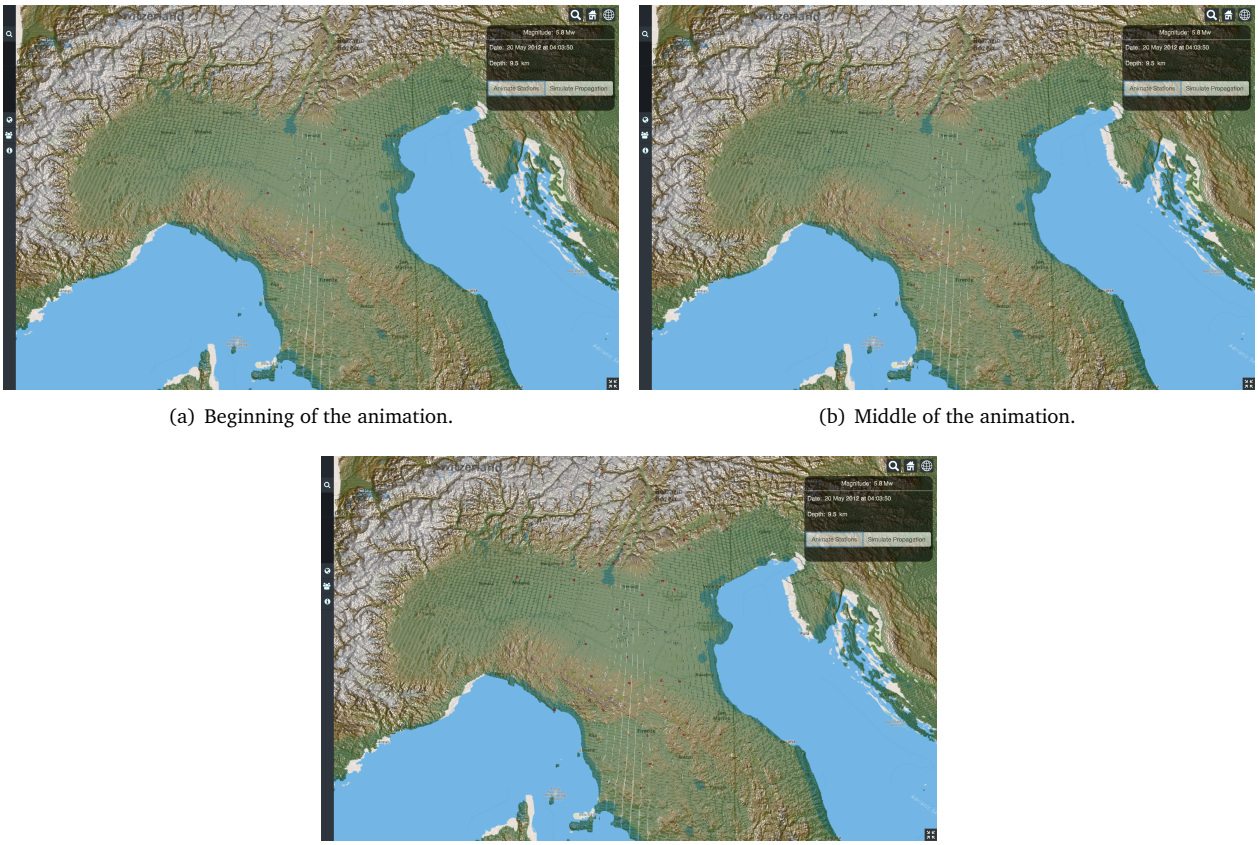


Figure 4.6. Info box for the Central Italy earthquake of October 2016.

In addition the info box contains the two buttons ‘Animate Stations’ and ‘Simulate Propagation’.

4.4 Animate Stations

The button ’Animate Stations’ starts an animation that lets the user visualize the stations that measured a single earthquake in order of arrival time of the wave. Every station is highlighted based on the magnitude at which the wave arrived to it. In Figure 4.7 you can see the stations that measured the Emilia–Romagna earthquake of May 2012.



(a) Beginning of the animation.

(b) Middle of the animation.

(c) End of the animation.

Figure 4.7. Three different states of the animation.

4.5 Simulate Propagation

Clicking on the 'Simulate Propagation' button, the user can visualize how the earthquake propagated to a station as you can see in Figure 4.8

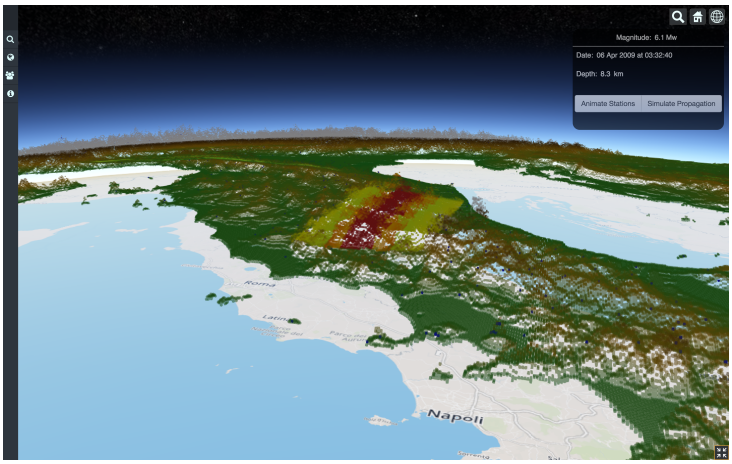


Figure 4.8. Propagation of the earthquake to a station.

The propagation is computed starting from the epicenter, that we obtain projecting the hypocenter on top of the Earth's surface. Then we retrieve the position of the station and we define the area of propagation of the earthquake. We then interpolate the color of every voxel between red and green based on its distance from the epicenter.

5 Conclusion and Future Work

Visual Earthquakes has great power, because it is the first web application that models the INGV data in a completely dynamic way. It is a useful application for showing earthquakes in a three dimensional context and is accessible by everyone.
Using our application, the everyday user can easily interact with a 3D model of Italy and he can visualize:

- The hypocenters of the earthquakes. Visualized at their real depth, shaded and modeled based on their magnitude.
- How the stations received the seismic waves. Stations appear following the arrival time of the seismic wave.
- How the earthquakes propagate though the terrain. The propagation begins from the epicenter and incrementally propagates to the measuring station.

Moreover, Visual Earthquakes is alive. It is always running up–to–date since it constantly fetches the newest earthquakes.
One limit of our application could be the performance. We are displaying a huge number of geometries (hundreds of thousands), and since Visual Earthquakes is a Web Application, it is a very big number of objects to manage.

Visual Earthquakes is still work in progress so it will grow more and more.
One possible future work could be the expansion of our system outside Italy, starting from other countries with high seismic risk like United States or Japan, until covering the whole globe.