Integrating tsunami simulations in web applications using NAMI, an open source client side GPU powered tsunami simulation library

José Galaz, Rodrigo Cienfuegos, Alejandro Echeverria, Sebastian Pereira, Celeste Bertin, Grazia Prato, Juan-Cristobal Karich

Integrating tsunami simulations in web applications using Nami, an open source client side GPU powered tsunami simulation library

J. Galaz [1,2], R. Cienfuegos[1,3,4], A. Echeverria[5], S. Pereira[2], C. Bertin[2], G. Prato[2], and J. C. Karich[3]

[1]Departamento de Ingeniería Hidráulica y Ambiental, Escuela de Ingeniería, Pontificia Universidad Católica de Chile. Vicuña Mackenna 4860, Santiago, Chile.
[2]Inria Chile, Avenida Apoquindo 2827 piso 12, Las Condes, Santiago, Chile.
[3]Centro de Investigación para la Gestión Integrada del Riesgo de Desastres (CIGIDEN), CONICYT/FONDAP/1511007, Santiago, Chile.
[4]Marine Energy Research and Innovation Center (MERIC). Av. Apoquindo 2827 piso 12, Las Condes, Santiago, Chile.
[5]Universidad de los Andes, Monseñor Álvaro del Portillo 12455, Las Condes, Santiago, Chile.

Abstract

Tsunami simulation software is essential in modern warning systems to characterize tsunami hazard, but its complexity discourages uses in risk management such as communication and public education. However, the ubiquity of web browsers and the appearance of web standards like WebGL that enable access to the Graphics Processing Unit (GPU), open the opportunity to reach other disciplines and communities. In this work, we developed an open source Javascript-WebGL library that enables users to run tsunami simulations that represent the propagation of long waves in the ocean efficiently on the web browser using the GPU. Through examples, it is demonstrated how Nami can produce results commonly seen in tsunami hazard assessment, and also other applications where the simulator is tightly integrated with other web elements, data sources and sensors. The latter is demonstrated with TsunamiLab, a web platform for public education and tsunami awareness developed with Nami at its core.

Keywords: Tsunami; Simulation; Web; Integration; GPU; WebGL

* *e-mail adresses:* ***jdgalaz@uc.cl*** *(J. Galaz),* ***racienfu@ing.puc.cl*** *(R. Cienfuegos),* ***alejandro.echev@gmail.com*** *(A. Echeverria),* ***sebastian.pereira@inria.cl*** *(S. Pereira),* ***celeste.bertin@inria.cl*** *(C. Bertin),* ***grazia.pratop@gmail.com*** *(G. Prato),* ***cristobalkarich@gmail.com*** *(J. C. Karich).*

Highlights

- Nami is a WebGL - Javascript tsunami simulation library
- Simulations run on web browsers "out of the box"
- Comparison with measurements and other simulators verify its accuracy
- Examples show how Nami integrates seamlessly with other web elements and technologies
- New use cases are highlighted by introducing TsunamiLab, an educational web platform

1 Introduction

Tsunamis have become one of the most disastrous natural hazards in the world, leaving in the last 20 years more than 200,000 casualties and billions of dollars in economic losses (Kânoğlu et al., 2015). The latter largely justifies the important efforts in increasing tsunami preparedness and awareness that different governments have undertaken over the last decades. Tsunami early warning systems, such as those from Japan, Chile, and the USA (Catalán et al., 2013; Titov et al., 2016; Kamigaichi, 2009), are important examples of these efforts which have enabled communities to react more quickly and prevent casualties. These systems are based on state of the art scientific knowledge and include numerical modeling as an essential tool for performing wave propagation forecast (Kânoğlu et al., 2015) but, usually, simulation results are only used to generate time series plots and 2D static maps representing wave propagation and inundation, leaving out other possibilities such as interactive applications and integration with Geographical Information Systems (GIS).

However, there are some interesting examples where efforts have been made to communicate tsunami risks to emergency managers and communities using simulation results. For example, Keon et al. (2014) integrate inundation estimation with agent based modeling to develop a framework that allows for interactive visualization and exploration of human response. Hsieh et al. (2013) show an application that allows for efficient computation and visualization of tsunami simulations on tiled displays taking advantage of the synchronized computing power of a GPU cluster, reaching high performance while efficiently rendering its results. Most remarkably, the Center for Tsunami Research (NCTR) of the National Oceanic and Atmospheric Administration (NOAA) of the United States Titov et al. (2016) fosters international collaborations providing emergency managers with different tools that can help assessing the hazard impact visualizing demographics, evacuation routes, infrastructure, and other variables that are relevant for the decision.

These applications can be related to a generalized need for new geovisualizations that can help to communicate the hazard and risk to non expert audiences, such as scientists from other fields, emergency managers, coastal

communities, policy makers (Teeuw et al., 2013), and also children, teachers, and adults (Kinzel, 2009). For example, regarding flood hazards, (Jacquinod et al., 2016) create 3D visualizations using GIS to increase the awareness and (Curebal et al., 2016) showcased how GIS can be useful to analyze inundation areas by interacting with engineering software tools such as HEC RAS (Brunner, 2010).

Important advances have been made also in computational modeling, increasing the resolution and complexity of the representation of tsunami hazards, but with higher demands of computational resources (e.g, GeoClaw (Clawpack Development Team, 2018), (Berger et al., 2011), COMCOT (Wang, 2009), CoulWave (Lynett et al., 2002), NeoWave (Yamazaki et al., 2012), EasyWave (Christgau et al., 2014), ANUGA (Nielsen et al., 2005), MOST (Titov et al., 2016)). These advances, however, have been made without consideration of the needs of visualization and risk communication since they have been done from the perspective of trained field experts. This approach has brought complexity to tsunami simulation software and posed barriers to other possible uses such as integration with GIS and web systems as reported by (Merati et al., 2009).

One possible solution to this problem comes from the advances in GPU computing that allow users to accelerate computations at a lower cost than traditional CPU clusters, and also produce complex visualizations efficiently. Recent examples of these are the open source softwares TsuPy (Schäfer and Wenzel, 2017) and Celeris (Tavakkol and Lynett, 2017). TsuPy is a script based python library that uses the NVIDIA Compute Unified Device Architecture CUDA (Nickolls et al., 2008) to calculate wave propagation and inundation at the regional scale with a shallow water solver, obtaining high performance computations on NVIDIA GPUs. Celeria on the other side, is a Boussinesq wave model that uses the Microsoft Direct3D API (Blythe, 2006) shaders to calculate and render the simulations in a 3D interactive view with a graphical user interface that allows users to change parameters on the fly. These tools demonstrate that it is possible to implement a tsunami simulator that can be both efficient and more accessible. However, the former software requires specific vendor hardware for its execution and the use of server-side computations to be implemented on a web system, and the latter is strictly bound to Microsoft Windows operating system and its integration with other software is not straightforward.

We propose to tackle this issue by taking advantage of web browsers whose ubiquity facilitates the development of cross-platform applications, being available to most people with a computer. In particular, we propose to take advantage of WebGL (Marrin, 2011), a Javascript API designed for rendering 2D and 3D interactive graphics that is supported by most modern browsers. Through WebGL the browser can run GPU-accelerated algorithms and easily mix their output with other elements in a website. Compared to other graphics technologies such as CUDA and DirectX, WebGL is better suited for developing general access tools since it has no bindings to particular hardware nor operating systems.

In this paper we are proposing Nami as a first-of-its-kind open source WebGL

tsunami simulation library that can be integrated with other web technologies to produce novel interactive applications and visualizations. In section 2 we give a brief overview of the mathematical equations and numerical algorithm to then explain Namis design and architecture in section 3. In section 4 we give examples of use cases that demonstrate, on one side, the capabilities of Nami to obtain accurate results (compared to other software and measurements) while also making efficient use of the available graphics card (dedicated or integrated), and on the other, that it is possible to build more complex applications, highlighting how it has helped us build TsunamiLab: a web platform for educating and increasing tsunami awareness.

2 Mathematical model

2.1 Model equations

The mathematical modeling of tsunami waves is a multi-scale problem that usually requires coupling several algorithms to properly cover all the physical phenomena observed from wave generation and propagation in the ocean, to wave shoaling and run-up (LeVeque et al., 2011; Wang, 2009; UNESCO, 1997). Here we focus on tsunami generation and propagation for far field wave forecast since it is the first step in the representation of tsunamis; the mathematical equations and algorithms are simpler and better known; and it provides relevant information for the understanding and communication of tsunami hazard.

In deep ocean, tsunamis have a characteristic wave length $L \approx 100$ km, a characteristic amplitude $a \approx 1$ m and a characteristic water depth of $h_0 \approx 4$ km, i.e. $h_0/L \ll 1$ and $a/h_0 \ll 1$ (Dean and Dalrymple, 1991). Under these assumptions a suitable approximation is given by the linear shallow water equations in spherical coordinates (Liu et al., 1995):

$$\begin{aligned} \frac{\partial \eta}{\partial t} + \frac{1}{Rcos(\theta)} \left(\frac{\partial M}{\partial \lambda} + \frac{\partial}{\partial \theta}(N \cos\theta) \right) &= 0 \\ \frac{\partial M}{\partial t} + \frac{gh}{R\cos\theta} \frac{\partial \eta}{\partial \lambda} &= fN \\ \frac{\partial N}{\partial t} + \frac{gh}{R} \frac{\partial \eta}{\partial \theta} &= -fM \end{aligned} \tag{1}$$

where t is the time coordinate and λ, θ are the longitude and latitude geographical coordinates respectively; η, M, N the wave height and longitudinal and latitudinal momentum components; $R = 6378$ km is the earth's radius; $g = 9.81$ m/s the earth's gravitational acceleration; $h(\lambda, \theta)$ is the bathymetry at location (λ, θ), where $h > 0$ indicates underwater floor; and $f = 2\omega \sin(\theta)$ is the Coriolis factor with $\omega = 7.29 \times 10^{-5} [rad/s]$, the earth's rotation frequency.

Similarly to Christgau et al. (2014), Wang (2009) and UNESCO (1997), equations (1) are discretized using finite differences with a second order in space and time leapfrog scheme as:

$$
\begin{aligned}
&\frac{\eta_{i,j}^{n+1/2} - \eta_{i,j}^{n-1/2}}{\Delta t} + \frac{1}{R\cos\theta_j}\left(\frac{M_{i+1/2,j}^{n} - M_{i-1/2,j}^{n}}{\Delta\lambda} + \right. \\
&\left. \frac{N_{i,j+1/2}^{n}\cos\theta_{j+1/2} - N_{i,j-1/2}^{n}\cos\theta_{j-1/2}}{\Delta t}\right) = 0 \\
&\frac{M_{i+1/2,j}^{n+1} - M_{i+1/2,j}^{n}}{\Delta t} + \frac{gh_{i+1/2,j}}{R\cos\theta_j}\frac{\eta_{i+1,j}^{n+1/2} - \eta_{i,j}^{n+1/2}}{\Delta\lambda} = fN' \\
&\frac{N_{i,j+1/2}^{n+1} - N_{i,j+1/2}^{n}}{\Delta t} + \frac{gh_{i,j+1/2}}{R}\left(\frac{\eta_{i,j+1}^{n+1/2} - \eta_{i,j}^{n+1/2}}{\Delta\theta}\right) = -fM' \\
&N' = \frac{1}{4}\left(N_{i+1,j+1/2}^{n} + N_{i+1,j-1/2}^{n} + N_{i,j+1/2}^{n} + N_{i,j-1/2}^{n}\right) \\
&M' = \frac{1}{4}\left(M_{i+1/2,j+1}^{n} + M_{i+1/2,j}^{n} + M_{i-1/2,j+1}^{n} + M_{i-1/2,j}^{n}\right)
\end{aligned}
\tag{2}
$$

where $\Delta\lambda, \Delta\theta$ is the grid size; Δ the time step; and i, j, n indicate values of the respective variable at time $n\Delta t$ and location $(i\Delta\lambda, j\Delta\theta)$. Given a domain and its bathymetry h, the input required to integrate equations (1) is contained in the initial and boundary conditions. As mentioned by Wang (2009), numerical stability is ensured by choosing $\Delta t = CFL \times \Delta s_{min}/c_{max}$, with $CFL < 1$ the Courant-Friedrichs-Lewy number, $c_{max} = \max_{ij}\sqrt{gh_{ij}}$ and Δs_{min} the minimum geodesic distance between two vertically or horizontally adjacent grid nodes.

2.1.1 Initial conditions

Even though arbitrary initial conditions can be provided through h, M and N, there are formulas that characterize the generations of the tsunami depending on a set of parameters. Two of these are implemented in Nami: earthquakes and asteroids. For earthquake generation we use a Finite Fault model, assuming that the water is at rest and a perturbation is added around the epicenter according to the formulas of Okada (1985). These formulas are an explicit solution to a linear elasticity problem that assumes that the area affected by the earthquake is rectangular and depends on parameters such as its length, width, hypocenter, fault slip, and the 3D orientation of the fault plane. Complex earthquakes can then be represented superposing several of these rectangles with different parameters depending on the heterogeneity of the earthquake being modeled.

For asteroid generated tsunamis, Ward and Asphaug (2000) describe several formulas that try to approximate the deformation of the water due to the impact of the asteroid depending on how its energy is converted into water waves. In Nami one of such formulas is implemented, and assumes that the initial surface has radial symmetry and can be described by a paraboloid of positive curvature whose size depends on the size, mass, density and speed of the impactor, as well as the amount of energy that is transmitted to the water from the asteroid.

2.1.2 Boundary conditions

Three types of boundaries are considered, usually named periodic, open and closed. Periodic boundaries repeat the information of the opposite boundary and are used for the East - West borders whenever the domain covers a full circle of latitude. Open boundaries approximately allow waves to leave the domain without reflections and are used whenever the domain does not cover a latitudinal circle; this boundary condition considers the trajectories of the characteristics of the Riemann invariants of the wave equations to extrapolate values in time as explained by UNESCO (1997). Closed boundaries are used to simulate a reflecting wall by imposing $\eta = M = N = 0$; this is used to model continents shorelines as a closed internal boundary, which makes sense at the large scale, where run-up displacements are negligible (UNESCO, 1997).

3 Architecture

3.1 Overview

Nami is designed to be integrated in web applications as shown in figure 1. First, a user such as an emergency manager or any person interested on visualizing and configuring tsunami simulations, interacts with a Web Application through a Graphical User Interface, on which relevant information is displayed, possibly along with interaction controls such as buttons, menus, etc. Then this application uses Nami to simulate tsunamis given some desired configuration and data, which then turn into GPU instructions in the WebGL engine that runs the simulation efficiently. This basic structure reveals the necessity of configuring the simulation with custom data provided by the Application and the User; also that it is important to be able to change the simulation on the fly as the user interacts with the Application; and finally, that Nami should provide output that can conform to some standard, so results can be integrated into other systems.

3.2 Nami components details

To explain how these three features are achieved in Nami, a zoomed architecture diagram is shown in figure 2, detailing the components of Nami that facilitate the configuration, interactivity and integration of the simulations. First the Application should provide input data such as domain configuration, bathymetry and initial conditions. Some of this data may be in different formats such as CSV, JSON text files, or PNG and JPG images. This data is internally received by a Reader component, which contains methods to parse these different formats. The mission of the Reader is to convert this data into useful input for the Shallow Water Model component which can only receive input data in one format. Once the data is received, the Shallow Water Model component is the one in charge of executing the simulation one step at a time and call the WebGL instructions that use GPU acceleration to speed up the calculations.

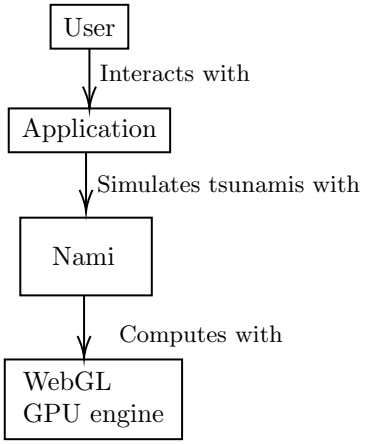

Figure 1: Nami architecture overview for a standard use case

For running several iterations until a specific timestamp is reached, a Simulation Controller component is implemented, which not only runs the simulation program, but also calls Model functions every certain number of numerical iterations to fill a Canvas HTML5 element with a pseudo color map or colored texture, representing the values of a desired variable such as current wave heights, maximum heights or arrival times. Then, using the Controller, the Application can interrupt the simulation with common playback commands (play, pause, restart, etc) on user demand, and with the Model, the Application can collect the results of the simulation and populate its views. for example, by overlaying the canvas texture on a map or displaying a time series plot of a specific point of interest.

3.3 Simulation life cycle

Though the Shallow Water Model and Simulation Controller components help with interactivity and integration, it is important to notice that they are not enough. For example, one may be calling Model functions before the data is available, or even before the components exist; or because it may be necessary to interrupt the simulation cycle under some specific condition, say, once the wave height is greater than a threshold or after a certain number of iterations. For this reason, we introduce the concept of life cycle, shown in figure 3, describing the sequence of relevant stages since the data is received until the Controller has reached its final simulation time t_f. The application can then subscribe to events by providing a set of callback functions (listed on the left side of figure 3), i.e., Application custom functions that are internally called at specific steps in the life cycle.

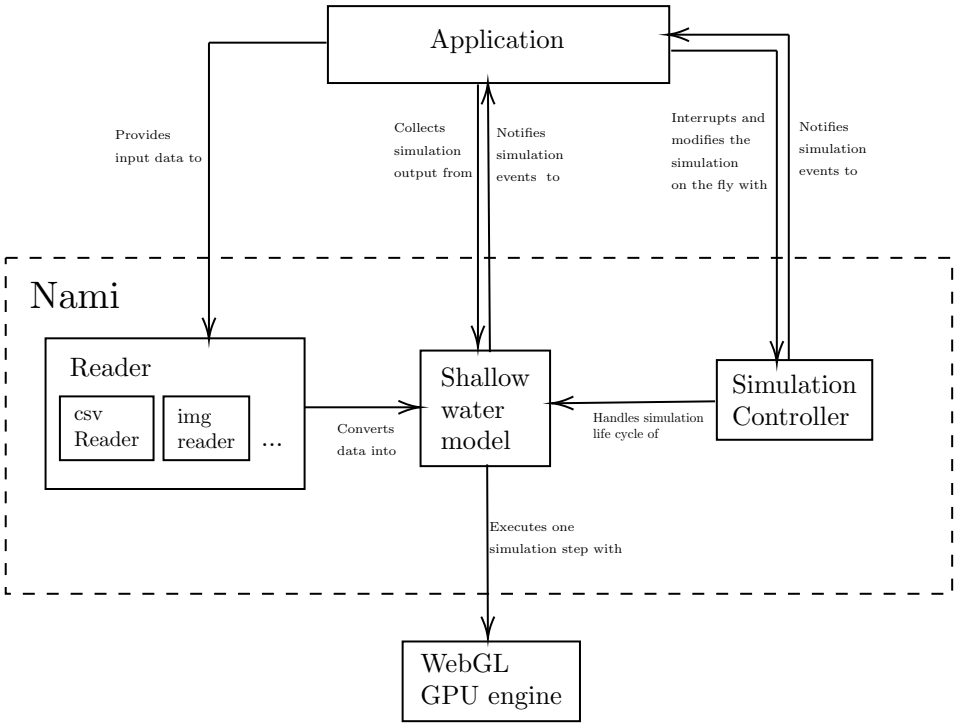

Figure 2: Architecture of Nami in terms of its internal components

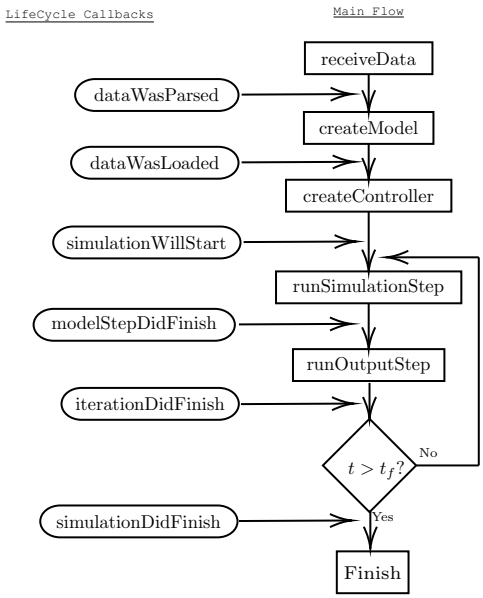

Figure 3: Life cycle diagram of a simulation in Nami.

3.4 Model implementation in WebGL

The Shallow Water Model component is the software representation of the algorithm and mathematical equations presented in section 2. It uses WebGL, an open web standard for GPU accelerated computer graphics on the web browser (Marrin, 2011), and as such, it requires one to adapt the implementation to concepts and design constraints of that field. The reason behind why it is useful for numerical computing is that every node in the numerical mesh depends only on a small number of neighbor nodes, making it analogous to image rasterization algorithms.

The rendering process is performed in a special type of program called Fragment Shader, written in a strong typed language called OpenGL Shading Language (GLSL), which assigns to each pixel in the target image its color through four values in the RGBA color space, i.e., three intensity levels for Red, Green and Blue and one for the transparency level (Alpha). Also, by design, a Shader program has no memory of previously rendered scenes so any prior information must be provided explicitly. For this reason, and for better performance, instead of rendering to the screen, Nami renders to a Frame Buffer Object (FBO), which is a WebGL object that allows storing the pixel data of a Shader into a texture that can be used as input later in other Shader instance. In total it uses two FBOs: FBO_1 to store results of the previous time step and FBO_2 for storing new results that depend on FBO_1. On each pixel of a FBO, the RGBA values are stored corresponding to (η, M, N, h) at each point in the simulation.

Figure 4 shows the flow of the simulation, internal to the Shallow Water

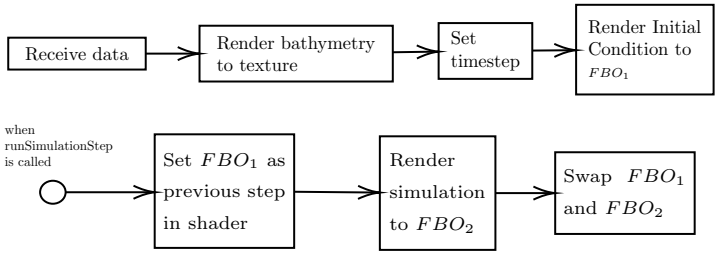

Figure 4: Flow diagrams showing two processes of the simulation internal to the Simulation Model component of figure 2. Top diagram shows the initial configuration since the data is received from the Reader. Bottom diagram is executed every time the *runSimulationStep* Simulation Model function is called.

Model component. After the data is received from the Reader component (top diagram), the bathymetry is rendered into a texture, the simulation timestep is calculated and the initial condition is rendered into FBO_1. Then, when the `runSimulationStep` function of the Shallow Water Model component is called FBO_1 is assigned as the previous step texture and the simulation shader program is run, with its output stored into FBO_2. The flow ends with FBO_1 and FBO_2 being swapped, so the data is ready for when the `runSimulationStep` function is called again.

4 Use case examples

4.1 Example 1: Model verification and visualization of maximum amplitudes

Here we consider the basic use case of Nami where the user, such as a scientist, needs to configure and run a simulation to post-process the hydrodynamic variables to generate figures such as maps of maximum amplitudes and line plots with time series. Examples of these are in figures 6, 7, 8, and 9, which correspond to the 8.8 Mw 2010 Maule earthquake and 9.1 Mw 2011 Tohoku earthquake.

To configure a simulation the user has to characterize the scenario, provide output options and optional life cycle callbacks to catch key events of the simulation. An example standalone HTML/Javascript code that that can be run from the web browser is shown in figure 5.

For each scenario, bathymetry and earthquake information are provided through external files. The bathymetry source is ETOPO-1 (Amante, 2009) and the earthquake is represented by Finite Fault models proposed by Delouis et al. (2010) and U.S. Geological Survey (2018) for the scenarios of Chile and Japan respectively, using the formulas mentioned in section 2. As shown in figure 5 the domain covers the spherical rectangle $[90, 325.83] \times [-60, 70]$ and is discretized in a spherical uniform grid of $4717 \times 2600 \approx 12.26$ million nodes

```
<body>
   <script src="nami.js"></script>
   <script>
       const scenario = {
           xmin: 90,
           xmax: 290,
           ymin: -70,
           ymax: 70,
           discretizationWidth: 4000,
           discretizationHeight: 2800,
           bathymetry: 'bathymetry',
           earthquake: 'earthquake.csv'
       };
       const outputOptions = {
           displayWidth: 1000,
           displayHeight: 700,
           stopTime: 60 * 60 * 25
       };
       const lifeCycle = {
             simulationDidFinish: (model, controller) => {
               controller.downloadMaximumHeights();
               controller.downloadArrivalTimes()
           }
       };
       const nami = new NAMI(scenario, outputOptions, lifeCycle
           );
   </script>
</body>
```

Figure 5: Example standalone HTML file with Javascript code for configuring Nami to produce heat maps of figures 6 and 8

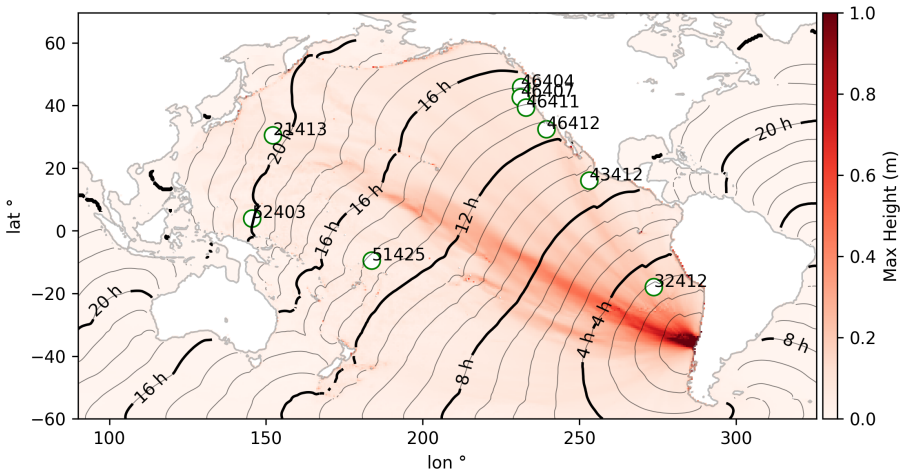

Figure 6: "Heat maps" of maximum wave amplitude and arrival time isochrones calculated by Nami for the 9.1 Mw scenario of Tohoku, Japan, 2011 and DART buoys locations.

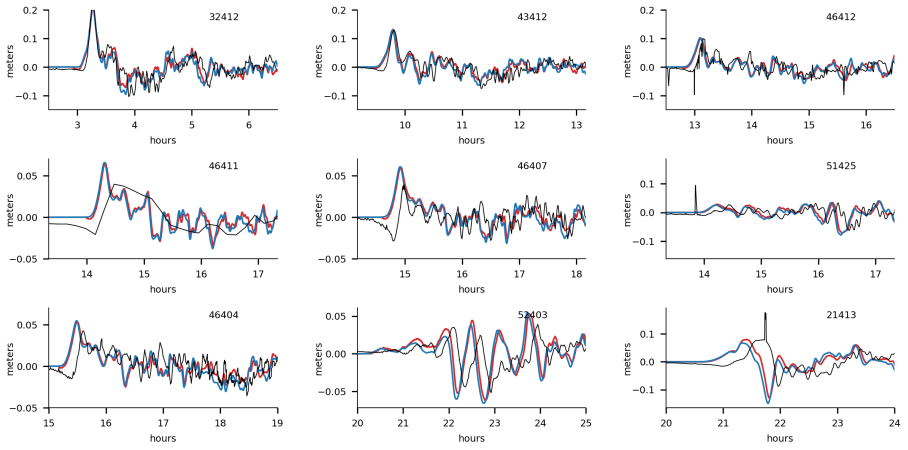

Figure 7: Time series of wave amplitude calculated by Nami (blue line), Easy-Wave (red line) and measurements (black line) at DART buoys shown in figure 6

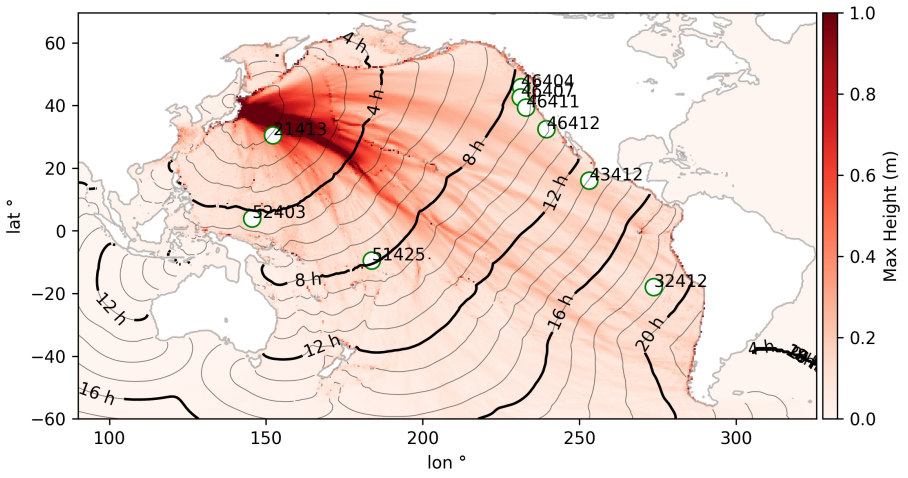

Figure 8: "Heat maps" of maximum wave amplitude and arrival time isochrones calculated by Nami for the 9.1 Mw scenario of Tohoku, Japan, 2011 and DART buoys locations.

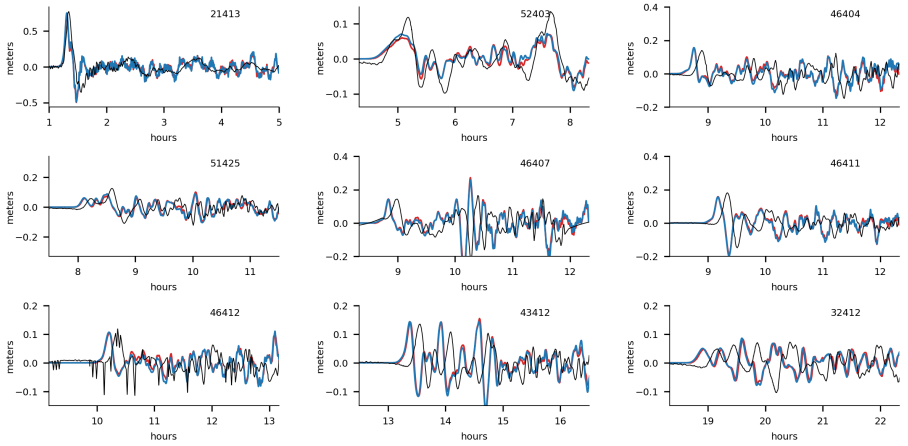

Figure 9: Time series of wave heights calculated by Nami (blue line), EasyWave (red line) and measurements (black line) at DART buoys shown in figure 8

	Integrated GPU	**Dedicated GPU**
15 minutes grid	1.6 min	1.44 min
3 minutes grid	43.6 min	8.44 min

Table 1: Execution times for the simulation of the 8.8 Mw, 2010 Maule earthquake for two grids of different resolution, using an integrated and dedicated GPU. Integrated GPU: INTEL(R) HD Graphics 630; Dedicated GPU: NVIDIA Geforce GTX 1060

with a spacing of 3 minutes. Each simulation is run until 25 hours of propagation (the `stopTime` parameter) with a timestep of 2.9365 seconds, configured by default for a CFL number of 0.5; all boundary conditions are open.

Since the goal in this use case is to extract the computed hydrodynamic variables and plot them from another environment such as a Jupyter Notebook (Kluyver et al., 2016), the only provided lifecycle callback is the `simulationDidFinish` function (see figure 3), where it commands the Nami `Controller` to export gridded values of maximum amplitudes and arrival times to files after the simulation has reached the `stopTime`.

Simulations were calculated on a computer with an Intel Core i7-7700HQ 2.8GHz CPU and 8GB of RAM. Table 1 shows the execution times of Nami for the 8.8 Mw Maule scenario, obtained with an integrated and a dedicated graphics card (Intel HD Graphics 630 and NVIDIA Geforce GTX 1060 resp.) on the same computer. This is shown for a 3 minutes and 15 minutes resolution grid. Though the execution times are very similar for the coarse grid, the simulation ran 5.16 times faster with the dedicated GPU. This reveals on one side the increased efficiency when using a dedicated GPU, and on the other, that it is still possible to run the simulations with an integrated graphics card which can be more accessible.

"Heat maps" of maximum amplitudes and travel times isochrones are produced in python using the results of Nami. Figures 6 and 8 show these results, where it is possible to see how the implemented numerical model represents tsunami propagation, including reflection and refraction patterns due to its interaction with bathymetry and shore lines.

Line plots showing time series of wave amplitude for each scenario are located in figures 7 and 9. Each line corresponds respectively to: Deep-ocean Assessment and Reporting of Tsunamis (DART) buoys measurementes after filtering out the astronomical tide using a low pass filter (black line); results of Nami (blue); and results of EasyWave (red), a software developed in the German Research Center for Geosciences, Potsdam (Christgau et al., 2014) that implements the same numerical model explained in section 2 with C++ and CUDA. DART buoys location and identification numbers are also shown with circles in figures 6 and 8.

In both scenarios Nami and EasyWave show little differences between each other and an overall good agreement with measurements when examining the maximum amplitude, shape and arrival. A consistent delay is also observed

with both models in the arrival of the first wave, which can grow from 0 to 15 minutes for the farthest points. As reported by Abdolali and Kirby (2017) and Watada et al. (2014) this difference comes from neglecting effects such as water compressibility and earth's elasticity; the inclussion of these effects in the estimation of arrival times is considered as a future improvement of Nami. Overall these results show that Nami can produce results with the same level of accuracy as other similar tsunami simulation software, in good agreement with measurements.

4.2 Example 2: TsunamiLab, a web platform of tsunami simulation

The second use case of Nami illustrates how it can be integrated with other technologies and, at the same time, how this integration facilitates access to scientific knowledge to a new group of users, especially those who are not experts in the field. Specifically, we describe TsunamiLab (TsunamiLab Development Team, 2018), a web platform for education and scientific outreach of tsunami hazard built using Nami, that aims at increasing the awareness and interest of people in science by providing interactive experiences. TsunamiLab possesses several features that enable users to observe the propagation of tsunami waves around the world, and select and modify scenarios based on historical and synthetic data, i.e., user-defined earthquake location and magnitude.

Figure 10 shows the main view of TsunamiLab, available at `www.tsunamilab.cl`. TsunamiLab is built with React (Fedosejev, 2015), a Javascript library maintained by Facebook that facilitates the development of complex interfaces under a reactive programming paradigm, structuring the application in terms of data flows and the state of individual components. The software architecture can then be described in a component tree as shown in figure 11. On this tree, components at the same level of depth can have access to the same state variables (such as the `magnitude` and `location` of the current scenario for the Nami, Canvas, Scene, Synth. Scenarios and Historical Scenarios components) and can pass them to the next component in the same branch. Components down the tree on a same branch can also receive callback functions to trigger changes in state variables of a parent component, triggering updates and data requests, for example.

TsunamiLab uses Nami for running the simulation and the Three.js library (Cabello et al., 2010) for displaying a 3D scene (in the Scene component of figure 11) where the Globe, Pins and other 3D graphics components are rendered. The integration of Nami with Three.js is performed by sharing an HTML5 Canvas (the Canvas component of fig. 11) between them. This Canvas serves for the instatiation of the WebGL context used by Nami to run the simulations, and also for storing the "heatmap" with the colors of the simulation at every timestep, which is then displayed in the Globe of the Scene as a texture on top of the base map.

The simulation domain covers the spherical rectangle $(lon, lat) \in [-70, 70] \times [-180, 180]$ with a numerical grid of 10 minutes. The bathymetry is sampled

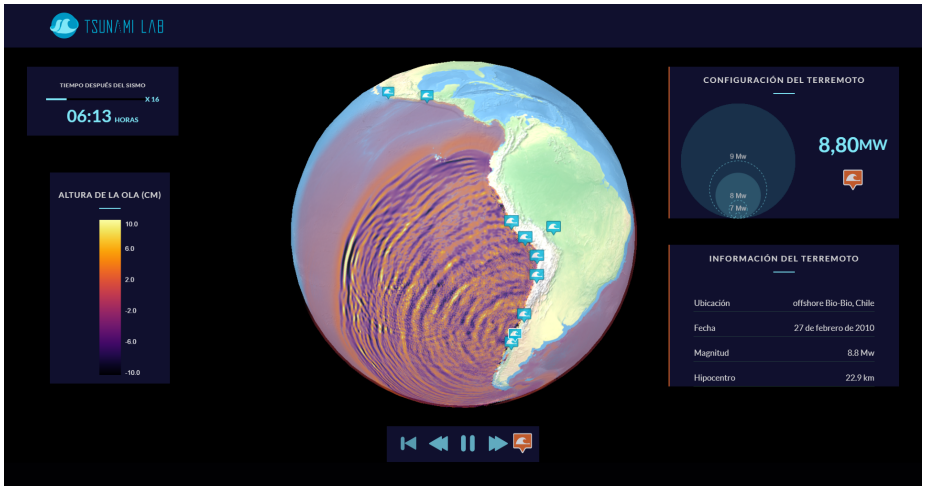

Figure 10: Screenshot of the main view of www.tsunamilab.cl

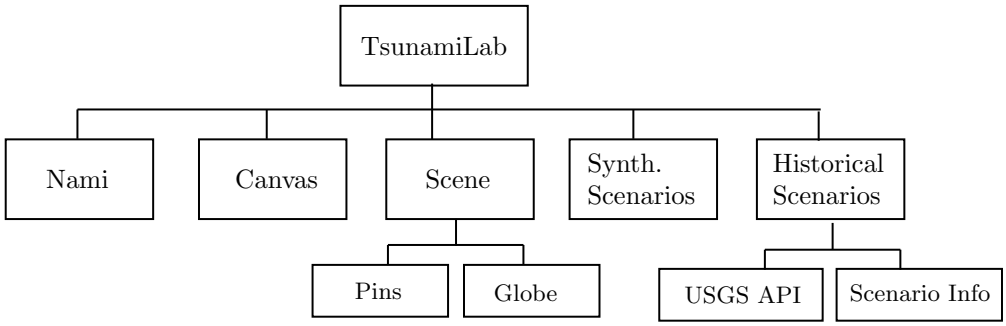

Figure 11: Component tree describing the architecture of TsunamiLab

from the ETOPO-1 dataset and compressed into a JPG image with the same resolution of the simulation to facilitate navigation. Quantization error from the JPG compression was found to be insignificant, since the only region where it becomes important is at the shores, which are excluded from the numerical domain of the simulator. The E-W boundary conditions are periodic whereas the N-S open.

Historical scenarios are represented in the Globe as a collection of wave Pins scattered around the Globe at earthquake locations. Earthquake location, magnitude, and focal mechanism information are obtained through the publicly available USGS Earthquake Catalog web API; whenever the user clicks on a visible Pin, a request is sent to the USGS Earthquake Catalog and the shared `magnitude` and `location` states are updated, triggering the update of Nami with the new scenario. At the same time, text information is updated in the Scenario Info component (bottom right table in figure 10) to give the user context about the current scenario.

Synthetic scenarios can be configured in a separate box in the top right corner of figure 10. The magnitude can be chosen by increasing or decreasing the radius r of the inner circle, on which case the magnitude of the new scneario is selected proportional to r^3. Earthquake location can also be selected by dragging and dropping the orange Pin on top of the globe. Nami is updated automatically whenever the `location` or the `magnitude` is changed through any of these two actions.

4.2.1 Experience with non-expert users

TsunamiLab has been presented on several ocassions where we have seen how these tools facilitate the communication of tsunami hazard to people of different ages. Examples of these participations are: the international competition "Mathematics of Planet Earth 2017" organized by Imaginary (IMAGINARY, 2017) and the Futur.E.S. festival organized by Cap Digital (Devillard) (see picture (a) in figure 12). Through these activities, we have noticed that it is possible to reach people and empower them to discover and formulate questions that can be valuable for a better understanding of tsunami risk.

More specifically, with Nami we have extended the capabilities of TsunamiLab and improved the user interaction with interfaces that make it better suited for exhibitions. Examples of this are shown in pictures of figure 12, where instead of a keyboard or a mouse we use a Sony PlayStation 3 Move Controller and a Leap Motion Controller as interfaces for user interactions. The former is a wireless joystick developed by Sony that, in addition to traditional buttons, includes a marker whose position is tracked by a camera and mapped accordingly into the scene to help with pointing actions such as rotating the Globe and changing the location of new scenarios; the latter consists of a set of infrared lights and camera that are used to infer the position of the users fingers, hands and arms, enabling actions based on gestures. These interfaces have helped us explain complex phenomena such as tsunami wave propagation, refraction, reflection and diffraction to people of different ages in a more attractive and usable

format, facilitating the communication of scientific results that are important for understanding tsunami hazard.

5 Conclusions

On this paper we have presented Nami, a Javascript - WebGL library that uses the GPU to run tsunami simulations in the web browser. The numerical model implemented in Nami represents the propagation of long waves in the ocean, of which tsunami waves are a good example. Through examples, it has been demonstrated how Nami can cover use cases such as generating heat maps of maximum wave amplitudes and time series plots, and also less common ones, where the simulator is tightly integrated with other web elements to generate interactive geovisualization tools. Specifically, the latter was covered by introducing TsunamiLab, an educational platform of tsunami simulation that provides interactive experiences to non-expert users in different contexts such as classrooms and science exhibits.

It was possible to verify that web technologies are mature enough to execute numerical simulations from a web browser, without imposing complex hardware or software setups such as server configurations or vendor specific graphic cards. Moreover, it was possible to verify that the results extracted from the simulations (gridded values and time series at points of interest) can be used in other contexts for further analysis, but also integrated with other web elements, providing interactivity and flexibility. The most important factor for achieving all this is the availability of open web standards such as WebGL that can provide hardware acceleration from the web browser.

The results of Nami were compared with measurements and other simulation software, which demonstrated that no accuracy is lost by using web technologies, despite WebGL being a low level technology designed for computer graphics. Instead, the differences with measurements are explained by the physical phenomena that are neglected in the mathematical equations such as water compressibility and earth's elasticity. These factors are considered as a future improvement of Nami.

The straightforward integration of the simulator with other web elements has been the most important factor to simplify development of the different versions of TsunamiLab. However, even though we have been able to reach different groups of people, the question of how effective this kind of tools are for improving tsunami awareness, risk perception and other topics has not been covered here and remains as a line of future work.

Future improvements of Nami include changes to the Model component to improve the accuracy of arrival times, as proposed by Watada et al. (2014). Also, improving the numerical dispersion to match the physical dispersion as proposed by Ha and Cho (2015) and ultimately the inclusion of higher resolution algorithms to include non linear effects and bottom friction is important to represent smaller scale phenomena. Other questions that still remain open are in regards to the possibility of using WebGL for high performance com-

(a)

(b)

Figure 12: Pictures of activities performed with non-experts users. (a) Interactive display that uses the SONY ™PS Move Controller to configure earthquake magnitude and location, at the 2018 Paris FUTUR.E.S. technology festival organized by Cap Digital; (b) Interactive display that uses the Leap Motion Controller to configure earthquake magnitude and location using hand gestures at a local science fair in Chile.

puting in tsunami modelling, which would require to study the limitations of server side support on the scalability of the simulator and also the ability of reading/exporting from/to other file formats efficiently, among others.

6 Acknowledgements

This project has been developed under projects CONICYT/FONDAP/15110017 "Centro de Investigación para la Gestión Integrada de Desastres Naturales" and CORFO 10CEII-9157 "Inria Chile". Support was also received by the Marine Energy Research & Innovation Center (MERIC - project CORFO 14CEI2-28228).

The authors are also grateful for the contributions of professor Felipe Cortez and the students of the course "Usabilidad y Nuevos Medios" of the School of Design of the second semester of 2018 of the Pontifical Catholic University of Chile, for their feedback for TsunamiLab.

References

Abdolali, A. and Kirby, J. T. (2017). Role of compressibility on tsunami propagation. *Journal of Geophysical Research: Oceans*, 122(12):9780–9794.

Amante, C. (2009). Etopo1 1 arc-minute global relief model: procedures, data sources and analysis. *http://www. ngdc. noaa. gov/mgg/global/global. html*.

Berger, M. J., George, D. L., LeVeque, R. J., and Mandli, K. T. (2011). The GeoClaw software for depth-averaged flows with adaptive refinement. *Advances in Water Resources*, 34(9):1195–1206.

Blythe, D. (2006). The direct3d 10 system. *ACM Transactions on Graphics (TOG)*, 25(3):724–734.

Brunner, G. W. (2010). *HEC-RAS river analysis system: hydraulic reference manual*. US Army Corps of Engineers, Institute for Water Resources, Hydrologic .

Cabello, R. et al. (2010). Three. js. *URL: https://github. com/mrdoob/three. js*.

Catalán, P., Caas, J., Zñiga, C., Zelaya, C., Gubler, A., Pizarro, L., Valdés, C., and Miranda, S. (2013). Sistema integrado de prediccin y alerta de tsunamis (sipat). In *XXII Congreso Chileno de Ingeniera Hidráulica*.

Christgau, S., Spazier, J., Schnor, B., Hammitzsch, M., Babeyko, A., and Waechter, J. (2014). A comparison of cuda and openacc: accelerating the tsunami simulation easywave. *PARS-Mitteilungen: Vol. 31, Nr. 1*.

Clawpack Development Team (2018). Clawpack software. Version 5.5.0.

Curebal, I., Efe, R., Ozdemir, H., Soykan, A., and Sönmez, S. (2016). Gis-based approach for flood analysis: case study of keçidere flash flood event (turkey). *Geocarto International*, 31(4):355–366.

Dean, R. G. and Dalrymple, R. A. (1991). *Water wave mechanics for engineers and scientists*, volume 2. World Scientific Publishing Company.

Delouis, B., Nocquet, J.-M., and Vallée, M. (2010). Slip distribution of the february 27, 2010 mw= 8.8 maule earthquake, central chile, from static and high-rate gps, insar, and broadband teleseismic data. *Geophysical Research Letters*, 37(17).

Devillard, A. Les innovations à ne pas rater au festival futur.e.s en seine.

Fedosejev, A. (2015). *React. js Essentials*. Packt Publishing Ltd.

Ha, T. and Cho, Y.-S. (2015). Tsunami propagation over varying water depths. *Ocean Engineering*, 101:67–77.

Hsieh, T.-J., Liang, W.-Y., Chang, Y.-L., Satria, M. T., and Huang, B. (2013). Parallel tsunami simulation and visualization on tiled display wall using opengl shading language. *Journal of the Chinese Institute of Engineers*, 36(2):202–211.

IMAGINARY (2017). Winners of the second mathematics of planet earth competition.

Jacquinod, F., Pedrinis, F., Edert, J., and Gesquière, G. (2016). Automated production of interactive 3d temporal geovisualizations so as to enhance flood risk awareness. In *UDMV 2016*.

Kamigaichi, O. (2009). Tsunami forecasting and warning. In *Encyclopedia of complexity and systems science*, pages 9592–9618. Springer.

Kânoğlu, U., Titov, V., Bernard, E., and Synolakis, C. (2015). Tsunamis: bridging science, engineering and society. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 373(2053):20140369.

Keon, D., Steinberg, B., Yeh, H., Pancake, C. M., and Wright, D. (2014). Web-based spatiotemporal simulation modeling and visualization of tsunami inundation and potential human response. *International Journal of Geographical Information Science*, 28(5):987–1009.

Kinzel, M. R. (2009). Using educational tools and integrative experiences via geovisualizations that incorporate spatial thinking, real world science and ocean literacy standards in the classroom: a case study examined. `https://ir.library.oregonstate.edu/concern/graduate_projects/2v23vz927`. Last accessed on Feb. 17 2019.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J. B., Grout, J., Corlay, S., et al. (2016). Jupyter notebooks-a publishing format for reproducible computational workflows. In *ELPUB*, pages 87–90.

LeVeque, R. J., George, D. L., and Berger, M. J. (2011). Tsunami modelling with adaptively refined finite volume methods. *Acta Numerica*, 20:211–289.

Liu, P. L.-F., Cho, Y.-S., Yoon, S., and Seo, S. (1995). Numerical simulations of the 1960 chilean tsunami propagation and inundation at hilo, hawaii. In *Tsunami: Progress in prediction, disaster prevention and warning*, pages 99–115. Springer.

Lynett, P., Liu, P., Sitanggang, K., and Kim, D. (2002). Modeling wave generation, evolution, and interaction with depthintegrated, dispersive wave equations coulwave code manual. *Cornell University Long and Intermediate Wave Modeling Package*.

Marrin, C. (2011). Webgl specification. *Khronos WebGL Working Group*.

Merati, N., Chamberlin, C., Moore, C., Titov, V., and Vance, T. C. (2009). Integration of tsunami analysis tools into a gis workspace–research, modeling, and hazard mitigation efforts within noaas center for tsunami research. In *Geospatial Techniques in Urban Hazard and Disaster Analysis*, pages 273–294. Springer.

Nickolls, J., Buck, I., Garland, M., and Skadron, K. (2008). Scalable parallel programming with cuda. In *ACM SIGGRAPH 2008 classes*, page 16. ACM.

Nielsen, O., Roberts, S., Gray, D., McPherson, A., and Hitchman, A. (2005). Hydrodynamic modelling of coastal inundation.

Okada, Y. (1985). Surface deformation due to shear and tensile faults in a half-space. *Bulletin of the seismological society of America*, 75(4):1135–1154.

Schäfer, A. M. and Wenzel, F. (2017). Tsupy: computational robustness in tsunami hazard modelling. *Computers & Geosciences*, 102:148–157.

Tavakkol, S. and Lynett, P. (2017). Celeris: A gpu-accelerated open source software with a boussinesq-type wave solver for real-time interactive simulation and visualization. *Computer Physics Communications*, 217:117–127.

Teeuw, R. M., Leidig, M., Saunders, C., and Morris, N. (2013). Free or low-cost geoinformatics for disaster management: Uses and availability issues. *Environmental Hazards*, 12(2):112–131.

Titov, V., Kânoğlu, U., and Synolakis, C. (2016). *Development of MOST for real-time tsunami forecasting*. PhD thesis, American Society of Civil Engineers.

TsunamiLab Development Team (2018). Tsunamilab web platform.

UNESCO (1997). *IOC Manuals and Guides No. 35*. IUGG/IOC TIME Project.

U.S. Geological Survey (2018). M 9.1 - near the east coast of Honshu, Japan. Accessed July 22, 2018.

Wang, X. (2009). User manual for comcot version 1.7 (first draft). *Cornel University*, 65.

Ward, S. N. and Asphaug, E. (2000). Asteroid impact tsunami: a probabilistic hazard assessment. *Icarus*, 145(1):64–78.

Watada, S., Kusumoto, S., and Satake, K. (2014). Traveltime delay and initial phase reversal of distant tsunamis coupled with the self-gravitating elastic earth. *Journal of Geophysical Research: Solid Earth*, 119(5):4287–4310.

Yamazaki, Y., Cheung, K. F., Kowalik, Z., Lay, T., and Pawlak, G. (2012). Neowave. In *Proceedings and results of the 2011 NTHMP model benchmarking workshop, Boulder: US Department of Commerce/NOAA/NTHMP (NOAA Special Report)*, pages 239–302.