World Academy of Science, Engineering and Technology
International Journal of Civil and Architectural Engineering
Vol:10, No:1, 2016

Heterogeneous-Resolution and Multi-Source Terrain Builder for CesiumJS WebGL Virtual Globe

Umberto Di Staso, Marco Soave, Alessio Giori, Federico Prandi, Raffaele De Amicis

***Abstract*—**The increasing availability of information about earth surface elevation (Digital Elevation Models DEM) generated from different sources (remote sensing, Aerial Images, Lidar) poses the question about how to integrate and make available to the most than possible audience this huge amount of data.

In order to exploit the potential of 3D elevation representation the quality of data management plays a fundamental role. Due to the high acquisition costs and the huge amount of generated data, high-resolution terrain surveys tend to be small or medium sized and available on limited portion of earth. Here comes the need to merge large-scale height maps that typically are made available for free at worldwide level, with very specific high resolute datasets. One the other hand, the third dimension increases the user experience and the data representation quality, unlocking new possibilities in data analysis for civil protection, real estate, urban planning, environment monitoring, etc. The open-source 3D virtual globes, which are trending topics in Geovisual Analytics, aim at improving the visualization of geographical data provided by standard web services or with proprietary formats. Typically, 3D Virtual globes like do not offer an open-source tool that allows the generation of a terrain elevation data structure starting from heterogeneous-resolution terrain datasets. This paper describes a technological solution aimed to set up a so-called "Terrain Builder". This tool is able to merge heterogeneous-resolution datasets, and to provide a multi-resolution worldwide terrain services fully compatible with CesiumJS and therefore accessible via web using traditional browser without any additional plug-in.

***Keywords*—**Terrain builder, WebGL, virtual globe, CesiumJS, tiled map service, TMS, height-map, regular grid, Geovisual analytics, DTM.

I. Introduction

DIGITAL ELEVATION MODELS (DEM) are fundamental data within almost all typical earth science applications; simulations, e.g. flood modeling, and spatial analysis, requires terrain data. Recent advances in the fields of 3D data capturing, storage capacities, database technologies and web-based 3D-visualisation are gradually enabling the establishment and exploitation of large 3D terrains on a regional national or even global scale.

The interactive visualisation of large terrain surfaces requires multi-resolution tessellation strategies. The common strategy of a variety of approaches, based on regular or irregular, rectangular or triangular tessellations, is to represent the terrain surface with a minimum number of meshes per view and to progressively load and improve the terrain model in a dynamic environment. However, current classical Geo-Information-System (GIS) software cannot manage the demand of processing and analyzing these massive raw terrain data because for these purpose very computationally pre-processing operations are needed for handling the very massive amounts of raw data.

The main commercial 3D platforms for visualizing global terrain models (Google Earth, Nasa World Wind, Microsoft bing, CesiumJS) offers, by default, a large number of free terrain providers. The main difference between the available datasets is the file format in which information are stored and the average ground resolution. For performances reasons, mainly they are composed by a set of files, organized using a three structure. The use of the aforementioned solutions enables a set of pros and cons:

Pros:

1. The infrastructure (storage, bandwidth, load balancer, maintenance, etc.) is delegated to the data provider side.
2. Mainly, the provided datasets cover the whole world with a ground resolution up to 30 meters per pixel.

Cons:

1. The dataset is static; is offered, "as is" and it is not possible to add proprietary data.
2. Specific applications (e.g.: civil protections landslides monitoring, forest harvesting planning) require more accurate data for the portion of interest of the terrain, arising the needs of merging large-scale datasets with specific high-resolution data.

In order to overcome the above-mentioned limitation, our work was focused on designing, developing and testing a custom terrain builder. We focused our effort on a file-system based approach for performances reasons: conversion, re-projection, and merging of datasets are time-consuming operations, whose computational time is directly influenced by the weight of the input height maps.

In the illustrated approach, a substantial pre-processing phase allows to increase the real time performance of the client applications connected to the system.

This paper is organized in the following way:

- Section II is aimed to offer the theoretical concepts behind the pyramidal file system structure of elevation dataset together with a survey on existing solutions.
- Section III describes the proposed solutions and its evolution.
- In Section IV, the first experimental results are reported.
- Finally, in Section V, planned future works are described.

U. D. S. is with Fondazione Graphitech, Via alla Cascata 56/c, 38123 Trento, Italy (phone + 39 0461 3393, e-mail: umberto.di.staso@graphitech.it).

M. S, A. G, F. D. and R.D.A. are with the Fondazione Graphitech, Via alla Cascata 56/c, 38123 Trento, Italy (e-mail: marco.soave@graphitech.it, alessio.giori@graphitech.it, federico.prandi@graphitech.it, raffaele.de.amicis@graphitech.it).

II. CesiumJs Terrain Provider: Theory and Available Implementations

A common approach, adopted by a multitude of applications in this domain, is based on the Tile-Model [1], [2]. The Tile-Model is a recursive subdivision of a portion of territory based on the quad-tree structure [3]. Using this approach, it is possible to manage the level of detail (LOD), within the visualization system, in accordance with some parameters like the ground distance and the frustum of the camera. Each tile is structured as a regular grid [4] representing the digital elevation model [5] for a specific portion of the environment. For each tile belonging to an area where elevation data is available, an elevation file is stored in a specific location of the file system, according to specific a naming rule.

CesiumJS supports natively three different formats of elevation dataset: Band Interleaved by Line (BIL), Tile Map Service Height-map 1.0 (TMS) and Quantized Mesh.

A. CesiumJS Terrain Formats

1. The BIL File Format

The BIL file format can store 16-bit integer or 32bit floating point elevation data. It is composed by a sequence of binary coded values, each one representing the elevation of a point in the regular grid. 16-bit integer data can represent values in the range [−32768, 32767] with 1m vertical resolution. Given that the useful range on the earth surface is about [-11000, 8900] we can compress the represented range in order to reach higher vertical accuracy. CesiumJS represents elevation data in the range of [-1000, 12107.2] meters, compressing the value of unsigned INT16 dividing by 5 and applying an offset of -1000. In this way a vertical resolution of 20cm is reached.

It is very simple to estimate the size of a dataset and check the correctness, in terms of bytes, of the files. The following formula allows calculating the size of BIL file:

$$W \times H \times D = s \quad (1)$$

where, W and H are integer identifying the tile's grid resolution, D equals 2 for BIL-16 and 4 for BIL-32 and s is the size, in bytes, of the output file.

The main limitation of BIL files is related to the fact that it is not possible to specify, for each tile, when the maximum resolution of the source dataset is reached. As a consequence, a BIL-based file system structure requires a fully balanced pyramid organization, producing a huge number of interpolated data if the difference, under a ground-resolution point of view of the datasets involved in the pre-processing phase, is critical.

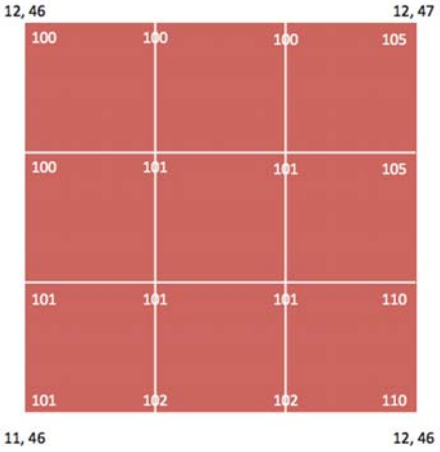

Fig. 1 4x4 BIL16 tile. The tile name allows calculating the outer values (the bounding box). The white values represent the elevation of the terrain in the given location

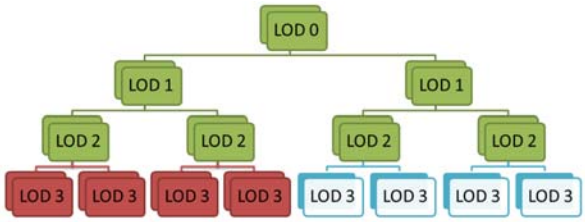

Fig. 2 BIL-based balanced file system pyramid. Green elements are generated from a 30m dataset, blue elements starting from a 1m dataset. Red elements are over-sampled version of parent tiles, not carrying any additional information. The generation of red tiles is required to produce a balanced pyramid and represent a waste of storage space and computational time

2. The TMS Height-Map 1.0 File Format

The TMS [10] file format is the evolution of the BIL data model including, besides the int16 height-map representation, 4 bits at the end of the file called childmask. Each childmask contains information about the availability of data at the next level of detail: "0000" represents the availability of the entire set of four tiles, the sequence "1111" means that the maximum ground resolution is reached at the current LOD. Additionally, the availability of each single child can be specified:

- Southwest - sequence 0001
- Southeast - sequence 0010
- Northwest - sequence 0100
- Northeast - sequence 1000

The adoption of the TMS data format enables to specify, for each tile, when the maximum resolution for each specific spatial area is reached or when it will be possible to increase the terrain resolution. These benefits allow:

- To create only the needed elevation maps, generating a unbalanced file system pyramid structure;
- To save storage in the infrastructure;
- To reduce data pre-processing time.

The file-system organization of the TMS terrain files is represented in Fig. 3: compared with Fig. 2 it is possible to better understand the number of elements not computed by the system.

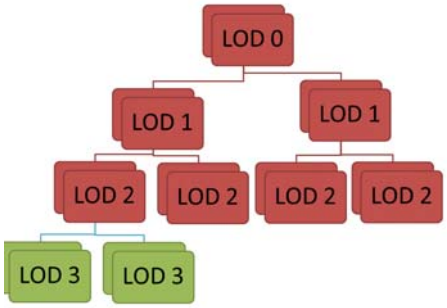

Fig. 3 TMS-based, unbalanced file system pyramid: the red elements are the ones generated using a 30m dataset while the green ones using a 1m dataset

B. Document Modification Available CesiumJS Terrain Datasets, Terrain Providers and Terrain Builders

The aim of this section is to analyze the existing CesiumJS-compatible elevation datasets and the set of free and commercial tools for produce them. For each element, an analysis has been carried out in order to exactly understand his strength and his weaknesses.

1. Datasets

a) Cesium Small Terrain

Cesium Small Terrain [11] is a general purpose, medium-resolution, worldwide coverage terrain elevation tile-set useful for Earth surface visualization and other uses. Cesium Small Terrain is offered as file system elevation pyramid, encoded in TMS – height-map 1.0 format. Additional information related to the water mask is provided in each tile elevation files. The maximum level of detail available is level (LV) 11, equals to ~90 meters ground resolution.

b) Cesium STK World Terrain

Cesium STK World Terrain [12] is a high-resolution, worldwide terrain elevation tile-set useful for Earth surface visualization and other uses. The maximum level of detail available is LV16 (~30 meters of ground resolution). Elevation files are provided as Triangular Irregular Network, TIN.

c) Esri ArcGis Image Server

Ersi ArcGis Image Server [13] produces terrain from height maps starting from an Esri Image Service, such as the World Digital Terrain Model (DTM), which contains datasets ranging from less than 1 meter up to 200 km.

d) VT MAK VR-TheWorld Server

VT MAK VR-TheWorld Server [14] produces terrain-using data from a VR-TheWorld Server. Over 4 terabytes of elevation, imagery, and digital map data pre-tiled for high-performance access in simulation and visualization applications:

- Global elevation - 90-meter (DTED Level 1) for the entire globe, including bathymetry;
- Global imagery - 15m resolution for the entire globe colour-corrected for seamless display;
- High-resolution inset areas, ranging from 1m down to 15cm per pixel.

2. Terrain Providers Tools and Servers

a) Cesium STK Terrain Server

STK Terrain Server [15] enables the access to a tile-set online and offline, such as on a private network disconnected from the Internet. It allows hosting locally the tile-set in order to improve performance, to overlay this tile-set with your own terrain data, to create an entirely independent tile-set from your own data and to maintain full up time and availability control. STK Terrain Server is a commercial solution offered by AGI.

b) Cesium Terrain Builder

Cesium Terrain Builder [16] is a C++ library and an associated command line tools designed to create terrain tiles. Cesium Terrain Builder can be used to create the tile-sets that sit behind a terrain server used by Cesium Terrain Provider. The limitation of this tool is that is not possible to merge automatically different datasets: the current released version allows only the concatenation of data, excluding spatial intersection between datasets.

c) GeoServer Terrain Provider

The GeoServer Terrain provider [17] enables the system to work with elevation maps originally developed for NASA World Wind in BIL format. Server side infrastructure is required: a running GeoServer instance equipped with the DDS/BIL plugin. Using this minimal Spatial Data Infrastructure, it is possible to export elevation maps as coverage grids. On the client side the GeoServer Terrain Provider allows reading the served files and to convert them in a Cesium-compatible height-map. The main limitation of this tool is that only one height-map can be served in a specific time interval and different datasets cannot be merged on server side.

III. The Proposed Multi Resolution and Multi Dataset Terrain Builder

In this section, technical specifications and methodologies about the processes behind the generation of a complex file-system based height-map pyramid are described.

The proposed implementation tries to solve all the problems identified in the currently available solutions starting from open source tools and libraries.

- In Section III *A*, the general idea at the base of the proposed approaches will be described;
- Section III *B* describes the first developed approach, where the BIL file format pyramids are generated using passing from the GeoServer web service with BIL/DDS plugin;
- In Section III *C*, the open problems related to the solution proposed in Section III *B* are described;
- Finally, in Section III *D* a novel approach, producing TMS tile pyramids and trying to solve the previously encountered problems, is described.

A. General Idea

Consider two different, but overlapping, elevation models: and high resolution one (A) and a low resolution one (B). The merging operation between A and B can be performed substituting data in B with sub-sampled data coming from A, ensuring that the position of each elevation point in the regular grid is maintained. The concept is described in the Fig. 4.

The proposed approach is simple but however many issues can affect the produced mixed dataset. Producing a complete tile pyramid merging a multitude of heterogeneous dataset without generating visual artifacts requires much shrewdness.

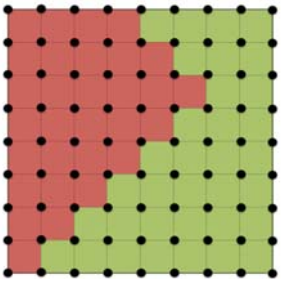

Fig. 4 Multi-dataset complex tile: red squares represent the base, low resolution, layer for a large-scale spatial area. Green squares represent the new data coming from the high-resolution dataset. The result is a single tile with data coming from two data sources

B. BIL Based Height-Map Construction with Middleware Layer

Taking the advantages provided by GeoServer Web Services and the DDS/BIL plug-in, the first tested approach was to develop software that can be executed on any operating system with a Java Runtime Environment available, aimed at merging regular grids for the same spatial area using the approach previously explained. The software has been engineered to be highly scalable and exploits as much as possible concurrency operations. The client is responsible to generate and perform all the needed tasks in order to generate the height-map tiles tree. The software performs the following steps:

1. It creates a new windows with a simple graphical user interface (GUI);
2. The user insert the IP and port of GeoServer;
3. The client reads and parses the GetCapabilities operation showing to the user the list of available datasets;
4. The user selects the desired layer and levels to be generated.

Height maps generation is divided in two phases; the first one is dedicated to the creations of tile jobs, while the second takes the jobs and generates the tiles. These two phases are required in order to parallelize the work and to minimize the amount of used memory.

The jobs once created are inserted in a First-In-First-Out (FIFO) queue, where multiple threads can write as publisher while at the same time other threads can read as subscribers. The queue data structure is able to handle concurrent accesses.

The tiles generation task is performed generating a request to GeoServer in order to get elevation data for that specific tile. The GeoServer response is composed by an array of Shorts (16bit signed Integer) that is converted in the BIL format and stored in the file-system under a specific naming rule.

The multi-layer problem is solved performing multiple iteration of the tile generation procedure, one for each dataset involved in the merging operation. For this reason, it is important to choose correctly the layer order because new data always overwrites old data. Obviously, the designed system performs some checks before overwriting: for example, it is required to manage correctly the GeoTIFF no-data values.

The concept expressed in this section is shown in Fig. 5.

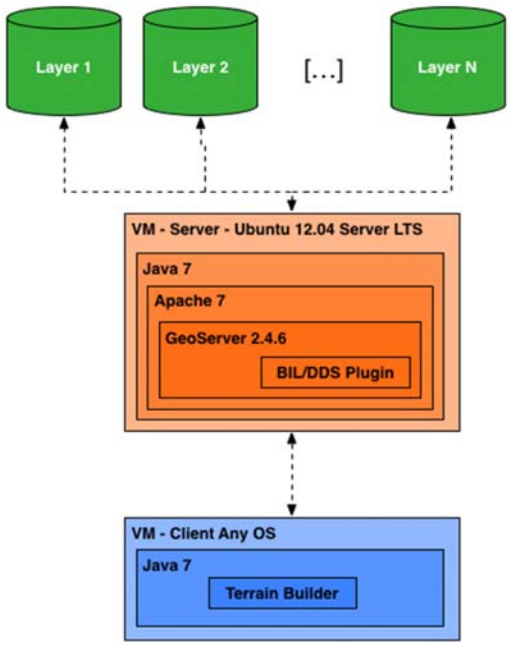

Fig. 5 Terrain Builder with middleware layer system architecture: the first of the Terrain Builder demands all the retail, re-projection and scaling operations, of recursively calculated tiles, to the middleware layer, composed by GeoServer plus BIL/DDS Plugin, that enables the system to expose elevation data as coverage grids, BIL, encoded as int16 values

C. BIL Based Approach Limitations

The approach described in the previous chapter is affected by some limitations:

1. The use of the pure BIL data format, whose limitations are described in Section II *B* 1;
2. Bugs in the BIL/DDS plugin related to the chosen interpolation algorithm. Data interpolation, when required, was performed only by using the nearest-neighbour approach, producing noisy area as shown in Fig. 6;
3. The use of a web-service based middleware layer constitutes a bottleneck for the entire system, reducing the overall performances.

D. TMS Based Height-Map Construction with Direct Access to the Data

In order to overcome the limitations of the previous

approach, a significant effort was spent in order to extend the Java-Based software for BIL merging, enabling the system to avoid any use of middleware layer. A custom terrain builder tool, based on the Java library GeoTools [9], has been deployed.

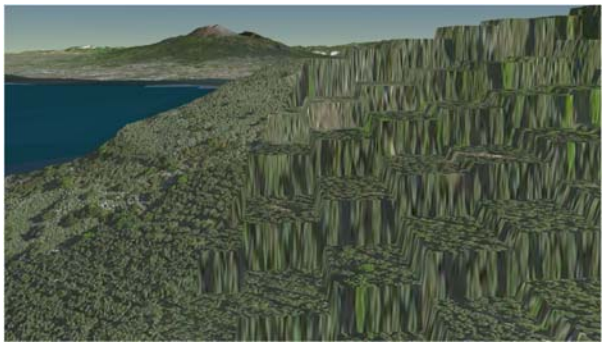

Fig. 6 A cross-border area between the 30m and 1m datasets. The stepped area is produced due to the use of the nearest neighbour interpolation algorithm on the less resolute dataset

It allows executing various operations on the input dataset, providing high flexibility, in-place execution and thus an improvement in the performances. This new development started from the work previously performed and re-utilized many of the components described in the Section III *C*. The tile pyramid structure component has been improved according with the new possibility offered adopting of the TMS-height map 1.0 file format, described in Section II *A* 2.

The datasets management is now a module of the ecosystem, avoiding the use of external components.

In general, the approach followed to merge different datasets was not changed from the previous one. In the first processing phase, the dataset is read and re-projected if necessary (the system works with the EPSG: 4326 coordinate reference system). The second step is the processing of the user preferences combined with the dataset metadata: using this information, it is possible to define the structure of the output pyramid. The formula (2) allows identifying the optimal LOD to be reached in order to avoid unnecessary data interpolation. The user defined bounding box (minor or equal to the input dataset) is fundamental determining the to-be-calculated branches of the pyramid where data exist; also, the child mask relies on this operation.

$$Lv_{max} = \left\lceil log_2 \left(\frac{180}{P_d * T_s} \right) \right\rceil \quad (2)$$

Lvmax is the pyramid level which resolution reaches the input dataset one. Pd is the spatial width, in degrees, of an input image pixel (after re-projection in EPSG 4326). Ts is the tile size: the number of pixel of each tile side (tiles must be square).

The calculated parameters allow defining the pyramid structure that will be generated: the directory tree is created and a queue containing tile objects (the file names and tile-bounding box) is generated. The threads manager component brings this queue as input, triggering the worker process on an arbitrary number of concurrent generation threads.

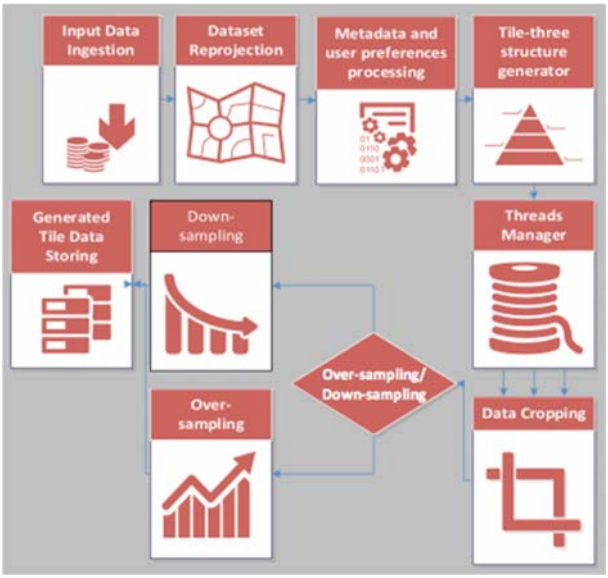

Fig. 7 Workflow of the TMS-based terrain builder with direct access to source datasets

TABLE I
TEST DATASETS DESCRIPTIONS

Name	Type	Size	Resolution	Area
Italia_40m	16bit single band GeoTIFF	1.6GB	40m	340000km2
PAT_1M	16bit single band GeoTIFF	36.5GB	1m	6500km2

Each generation threads, in the first phase, crop out from the original dataset the portion defined by the tile's bounding box. In the second phase an evaluation of the ratio between the tile and original dataset pixel size is performed, thus defining if it is needed to over-sample or to down sample the input dataset. Two different approaches are adopted as a function of the specific case: when down sampling the nearest neighbour interpolation is used, (faster and no visible artifacts introduced), while, when the tile implies an over-sampling of the input dataset, a custom bilinear interpolation algorithm is adopted. The needs of writing a custom algorithm come from the necessity to manage extreme cases: in the limit case the input data for the tile generation is composed of only one pixel. This situation can arise when two dataset, one really high resolute and a low-resolution one (e.g.: 40m and 0.1m) share the same boundaries. Furthermore when operating on the boundaries of a dataset tile where both no-data and elevation values are present: this situation must be cleverly handled from the interpolation algorithm (the no-data value must not be considered).

Finally, the output of the interpolation process is written to disk in the appropriate location of the tile pyramid, adding at the end the previously computed child mask. Same as the approach described in Section III *B* it is necessary to proceed with order, generating first the bigger and less resolute datasets, subsequently the smaller but more detailed ones. The tool will overwrite the portion of the already computed dataset

with data coming from the new one.

TABLE II
PERFORMANCES

Dataset	Start Level	End Level	#Tiles	Tile/sec
Italia_40m	0	10	66295	30
Italia_40m	0	12	92308	250
Italia_40m	0	13	293664	361
PAT_1M	0	10	110	14
PAT_1M	0	15	66295	30

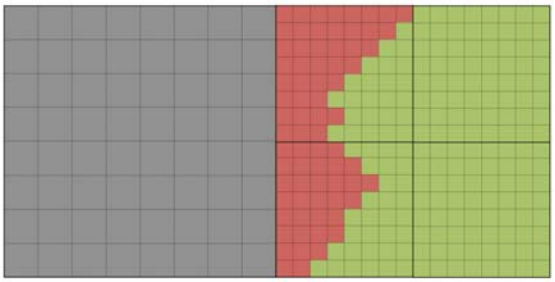

Fig. 8 Representation of the border zone between a high-resolution (HR) dataset (green, on the right) and a low-resolution (LR) one (grey + red, on the left). The tile on the left covers only LR data and do not need further splitting while the presence of HR data on the right one require an additional pyramid level. In the red zone the LR data is oversampled at the resolution of the HR data (not available in this portion)

IV. Testing and Results

In this section, performances and test results of the designed system are provided.

Performance tests results: During the testing phase, the system was executed on an Ubuntu Server 12.04 LTE machine, with 16 GB RAM, 8 processors, 1TB hard drive and Java7.

The results hereafter are related to the approach described in Subsection III.D. Due to the youth of this approach, not many results are available. As better explained in the next section the future work will focus on the optimization of the proposed approach through a test and improvement iteration.

The performances are strictly related to the dimension of the input dataset and the generation parameters: higher-level tiles including a small portion of the input dataset are much faster in the computation. This can be easily read from Table II: higher pyramid levels (e.g. 11,12 and 13) tile production is much faster (more than 200 tile/sec) than lower levels of the same dataset (average of 30 tile/sec in the level range 0-10). Furthermore, considering the same tile range (0-10), the production of tiles is much faster when processing smaller dataset (30 tile/sec for the Italia40 dataset Vs. 14 tile/sec for the Pat_1m dataset).

Fig. 9 Monte Bondone TMS – Trento - Italy

V. Future Works

In the phase to come the system will be accurately tested, trying to face all the possible situations. This testing phase will be conducted logging the computational time required for each phase of the process thus identifying where improvements are possible. We will work also on the implementation of functionalities for the maintenance of the file-system based TMS terrain files pyramid. In the current solution, it is possible only to add new data on top of existing one; a future improvement will be to design a method allowing users to rollback changes, enabling possibility to remove layers from the stack.

A further improvement will be the possibility to manage different input data sources: first tests were performed only using GeoTIFF files (int16 single band, 1m max vertical resolution), but it often occurs that height map files were provided in different file formats like ASCII Grid.

Major changes will be required in order to allow the system to generate Triangulated Irregular Network: The approach that will be used in order to design and develop this component, is to create an additional component that will be connected at the end of the pipeline in order to get as input TMS-based tiles and, using algorithm from the literature [6]-[8], create the TIN and save it as Quantized Mesh format [18] file.

VI. Conclusion

The work carried out during the development of the terrain generation tool allowed us to understand in deep the strength and the problems of various approaches. In the first tested approach (Section III.B) some problems were identified:

- The middleware layer constitutes a bottleneck for the entire system, influencing the overall performances for the elevation files generation;
- A problem with the DDS/BIL GeoServer plugin has been encountered: the “nearest neighbour” interpolation is not adapt for our purpose;
- Critical limitations related to the BIL data format was encountered as described in Section III.C.

The adoption of the TMS file format solves the last problems, optimizing the output pyramid provided by the system.

The implementation of the data management and processing

layer (Section III.D) allowed to overcome the interpolation problem and to improve the overall performances.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community's Competitiveness and Innovation framework Programme (CIP/2007-2013) under the Grant Agreement n. 325161. This publication reflects the authors view, and the European Commission is not responsible for any use, which may be made of the information contained therein.

REFERENCES

[1] Lindstrom-Koller, Real-Time, Continuous LOD Rendering of Height Fields, Siggraph 1996.
[2] Renato Pajarola, Large Scale Terrain Visualization Using the Restricted Quadree Triangulation, IEEE Visualization 1998.
[3] Patrick Cozzi, Kevin Ring. 3D Engine Design for Virtual Globes, CRC Press 2011.
[4] Burrough, Peter A., et al. Principles of geographical information systems. Vol. 333. Oxford: Oxford university press, 1998.
[5] Li, Zhilin, Christopher Zhu, and Chris Gold. Digital terrain modeling: principles and methodology. CRC press, 2010.
[6] Fowler, Robert J., and James J. Little. Automatic extraction of irregular network digital terrain models. ACM SIGGRAPH Computer Graphics 1979.
[7] Van Kreveld, Marc. "Algorithms for triangulated terrains." SOFSEM'97: Theory and Practice of Informatics. Springer Berlin Heidelberg, 1997.
[8] Garland, M. "qslim 2.1: The QSlim mesh simplification software." URL: http://graphics. cs. uiuc. edu/garland/software/qslim. html 6.
[9] Turton, Ian. Geo tools. Open source approaches in spatial data handling. Springer Berlin Heidelberg, 2008. 153-169, 2008.
[10] Agi, https://cesiumjs.org/data-and-assets/terrain/formats/heightmap-1.0.html, 17th November 2015.
[11] AGI, http://cesiumjs.org/data-and-assets/terrain/small-terrain.html, 17th November 2015.
[12] AGI, https://cesiumjs.org/data-and-assets/terrain/stk-world-terrain.html, 17th November 2015.
[13] AGI, https://cesiumjs.org/2013/01/04/Cesium-Imagery-Layers-Tutorial, 17th November 2015.
[14] VT-MAK, http://vr-theworld.com, 17th November 2015.
[15] AGI, http://www.agi.com/products/stk/terrain-server, 17th November 2015.
[16] Kaktus40 https://github.com/kaktus40/Cesium-GeoserverTerrain Provider, 17th November 2015.
[17] GeoSolutions, http://geoserver.org, 17th November 2015.
[18] AGI, http://cesiumjs.org/data-and-assets/terrain/formats/quantized-mesh-1.0.html, 17th November 2015.

Umberto Di Staso has been working in Fondazione Graphitech since 2012. He received a master degree in Computer Science from the University of Trento, Italy. Since 2012 he has been involved in a number of projects in the field of GeoVisual Analytics and Spatial Data Infrastructure, including among others, 3D Geobrowser for Smart Cities - 3D visualization of geo-referenced data.

Alessio Giori has been working in Fondazione GraphiTech since 2013. He received a master degree in Computer Science, with specialization in "Data, Media and Knowledge Management" from the University of Trento in Italy. He also received a professional master degree in "Technologies for e-Government" and a bachelor degree in Computer Science from the same university. He is currently working on two European research projects eENVplus and LIFE+IMAGINE.

Marco Soave: He is a developer and researcher in Fondazione Graphitech since 2012. He holds a Master of Science in telecommunication engineering. He has played an active part in the progress and development of various EU projects with tasks ranging from Service Oriented Architecture definition, User Interface Design, Natural User Interaction metaphors definition and implementation, management and processing of geospatial data and design, development and testing of computer vision techniques.

Federico Prandi has been working in Fondazione Graphitech since 2009. He received a master degree in environmental Engineering, a PhD degree in Politecnico of Milan. He has been involved in several EU and research project in the area of 3D geo-visualization and application and 3D reconstruction and image based modelling

Raffaele De Amicis is Director of Fondazione Graphitech, he holds a MEng in Mechanical Engineering, a Ph.D. on Surface Modelling in Virtual Environments. He has been research fellow at the Industrial Applications Department of Fraunhofer Institute, Darmstadt and senior researcher at the Interactive Graphics Systems Group, University of Darmstadt. He has been involved in several EU and Industrial projects. His research interests are in CAD, virtual reality, computer supported cooperative work in engineering.