Polygon File Format for Reflecting the Martian Terrain Data

Nicole Christoff[1, 2], Hadrien Greef[3], Sébastien Germond-Mazet[3], Jean-Luc Mari[2],
Laurent Jorda[4] and Agata Manolova[1]
[1]Technical University of Sofia, Visual System Information Lab
Bulgaria.
[2]Aix-Marseille Université, CNRS, LSIS UMR 7296, France.
[3]Aix-Marseille Université, France.
[4]Aix-Marseille Université, CNRS, LAM UMR 7326, France.

ABSTRACT: *In this work, we have introduced a new interface that can able to visualize the 3D mesh form of the Polygon File Format and it is used to reflect the Martian terrain data. It is diagonally opposite to the crater location that relates the center and diameter of the input file. The generated program can able to check the designs and adjust the results which are generated with detection algorithm. To ensure ergonomic interface and features we have used the validation process.*

Keywords: 3D Mesh, Interface, Visualization, Craters

Received: 30 October 2021, Revised 24 January 2022, Accepted 9 February 2022

DOI: 10.6025/jmpt/2022/13/2/42-49

1. Introduction

The collision of celestial bodies, such as asteroids or meteorites, gives as result craters on their surface. The study of different morphological features of craters, and counting their numbers, is the only method to estimate the age of the celestial body. Therefore, it is a problem of great scientific importance in astronomy. The complexity of the problem arises when the impact crater zones are very heterogeneous due to the distribution and size of the craters.

To be able to solve partially this issue, different methodologies of geometric image analysis are proposed. They are applied to 2D or 3D data for automatic crater detection. Indeed, the manual identification of craters on the planet surface is an effective and precise method. The volume of data that needs to be processed grows with the development of new imaging technologies. That pushes scientists to look for a fully automatic method.

The combination of the two techniques (manual and automated) is a good solution of the problem. It is based on a rapid automatic pre-detection of craters that serves as a base to the human validation. We provide an ergonomic interface for

visualization of 3D craters, which can correct the automatic detection. The rest of the paper is organized as follows: in Section 2, the existing approaches related to crater detection and interface are briefly described. In Section 3, we describe the environmental data, then we develop the core of our interface in Section 4. The Section 5 is dedicated to the theoretical solutions for the visualization of 3D meshes. We present our results in Section 6. Conclusions and direction of future research are given in Section 7.

2. Related Works

Many CDAs have been developed. They are based on 2D image data or 3D data, often in the form of Digital Elevation Model (DEM). The different approaches developed for 2D CDA are: the circle fitting [1], the exhibition of highlight and shadow region matching [2] or analysis textures [3]. Several approaches combine many of these methods [4].

The advantages of using detection from DEM data are the high resolution and the lack of hypothesis for the reflection of the surface, as well as the integration of additional 3D information. Previous works in this area are based on the detection of edges [5], identification of contours as a local circular depression [6] or the circle fitting Hough transform [7].

The creation of an ergonomic interface and 3D visualization of data, has been sparsely discussed in scientific works, often focusing on results and theoretical issues.

There are some works, which incorporate a graphical interface and they make the connection with CDA [8] and the existing functionality of Geographic Information System (GIS) software. For example, ArcGIS and QGis and various external searching modules [9], [10], based on the same software, each time in 2D. As far as we know, there is no reference regarding the 3D visualization of meshes, combined with the displaying of detecting craters by CDA.

3. Environment and Data

3.1. Environment

The C ++ programing language is used for the visualization of the interface and the data structures, used for meshes: Polygon File Format (PLY) and Comma Separated Values (CSV). Qt 4.8.6 in its version of 2013 is a cross-platform application framework that is widely used for developing application software [13].

The Visualization ToolKit (VTK) is used for the acquisition of a 3D visualization library for manipulating meshes and .PLY files. It is a tool, directly linked with Qt and supports advanced graphics features. VTK is optimized to manage heavy amounts of data and complex displays. Its graphic operating resources are developed to run our software on fairly modest configurations, even with large files.

3.2. Data

The 3D mesh, in .PLY format, represents an area of Mars. It is considered for the detection, already triangulated from the DEM. In our case, it is a large data: over 5 million points (vertex) and 10 million faces. For comparison, the smaller ones are about 3 million points and 6 million faces. The area is a sample of 0.46 km / mesh. The input file is detected with a .CRAT file (CRATer extension, especially created for this work). This is a CSV file, with the following structure: coordinates of X, Y, Z of the center and the diameter of craters, detected by a CDA.

4. Development of the Interface

4.1. Menus and windows

For the functional and ergonomic interface, we use the complex Qt classes - QMainWindow. The global window of the software contains a top menu, side toolbars, footer menu and the viewing window VTK. We made many inheriting of Qt classes to define our own widgets (Figure 2). The main file generates a MainWindow, containing a CentralWindow, derived from VTKWidget class. The latter inherits the characteristics of QVTKWidget class. These heritages allow us to override certain functions of the original classes, while maintaining the interactions between all these objects. In the main window, we define the areas of the program: (1) a horizontal toolbar (Figure 1. at the top of the window.), which contain the main interactions, opening files, saving, display options, etc. ; (2) a lateral toolbar (Figure 3) irremovable for configuring the mouse modes of action (add a crater trackball

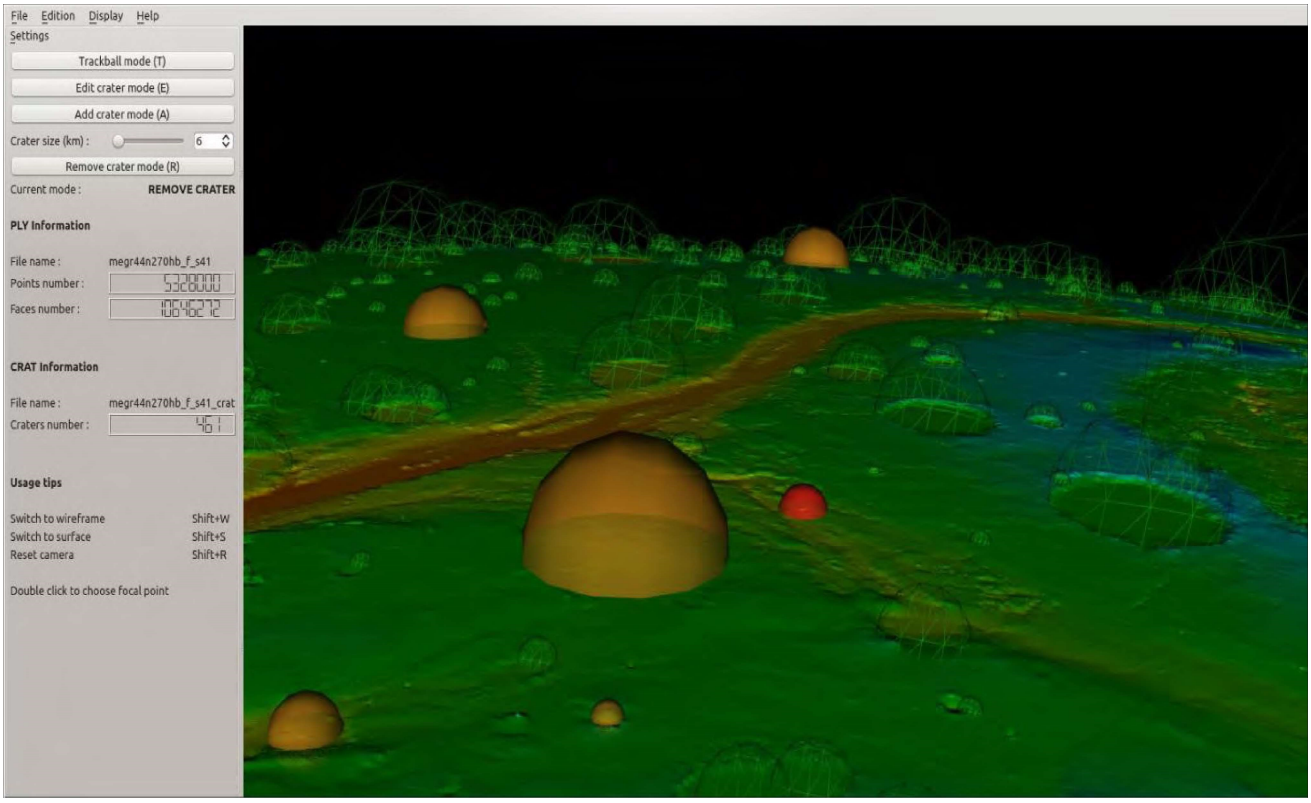

Figure 1. The interface

mode, etc.) and display a basic information about uploaded files; (3) the central window, in which the mesh is displayed and where the user can click to move or edit 3D craters (Figure 1 right).

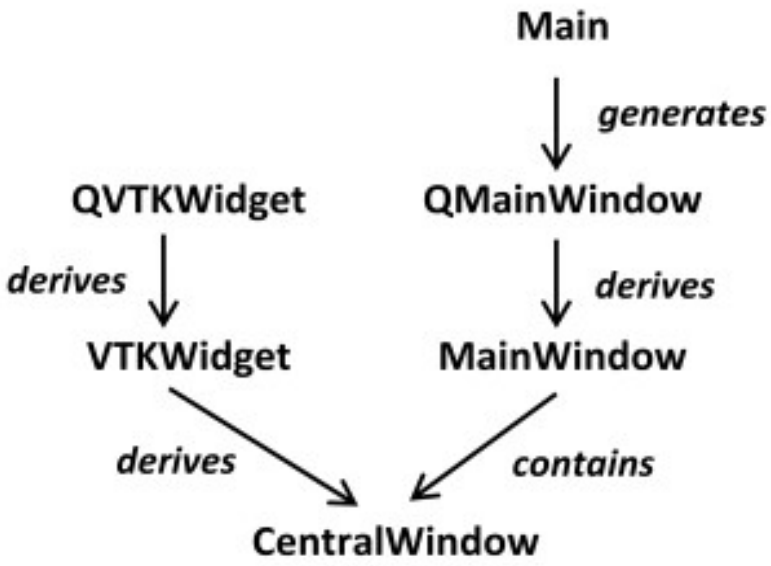

Figure 2. Bloc diagram

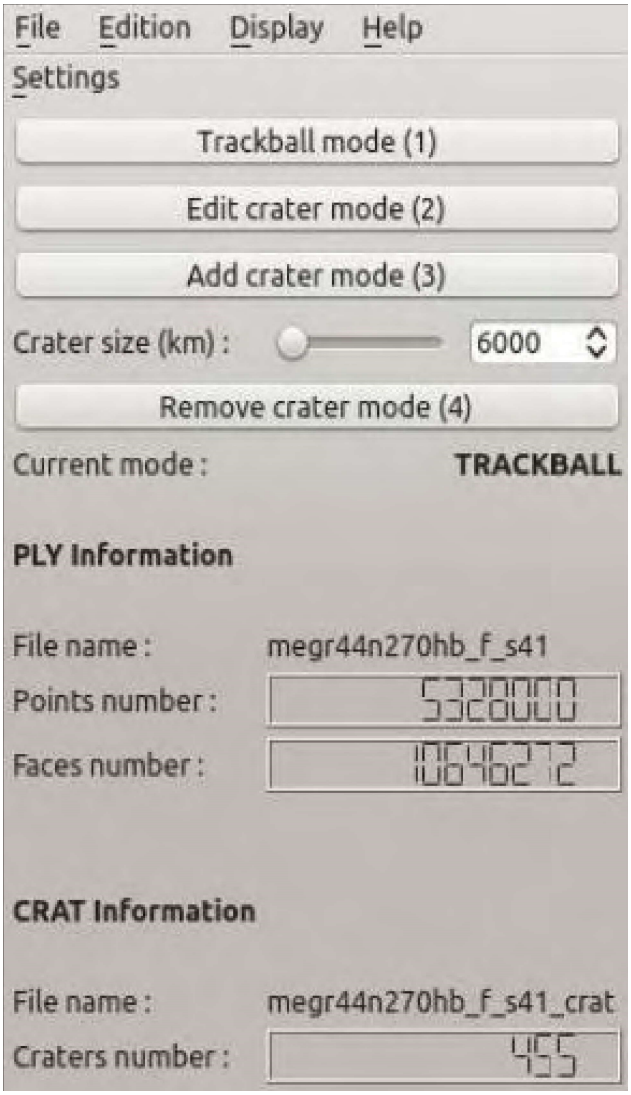

Figure 3. The lateral toolbar

All functionality of the menu and the toolbar are connected to signals and slots (a language construct introduced in Qt for communication between objects [14]), to pass them on the program's global data (the list of craters) or to change the Interactor mouse (see Section 4-c). We manage pop-ups: (1) for searching of files that will be used and (2) to link these windows to the main application. The theoretical solutions of all the features are explained in Section 5.

4.2. Displaying the Grid

The solution had to be considered for a fluid display of the mesh, which can be moved, enlarged and covered, in all directions, by the user. To load the file, a decimation algorithm was used to produce a lower level of detail. It is sufficient to continue to distinguish the biggest reliefs. In fact, it is destined to be displayed instead of the detailed model, every time the camera moves and zooms. In this way, to move the entire mesh remains a fluid process. Once the model is positioned, it is the detailed version that appears. This technique, used in GIS software [10] does not utilize many graphics resources and makes visualizing in real-time the mesh of millions of polygons possible. The decimation technique is presented in Section IV. The number of polygons, obtained in the decimation step, cannot be known in advance. In our case, it depends on the desired refreshing rate, produced during the movement and is thus likely to vary, depending on system configuration. The performances do not vary. Fig.4 illustrates both Level of Detail (LOD).

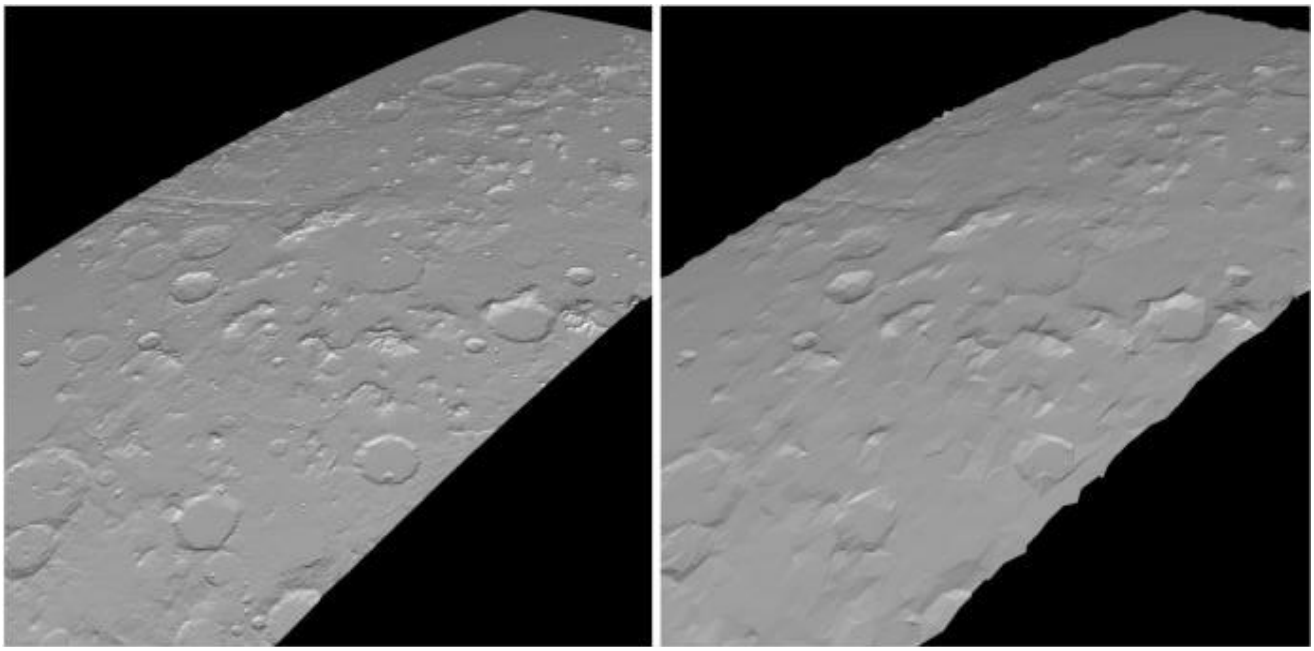

Figure 4. Comparison between original 10 million polygons meshes (left) and decimated one of ~ 1000 polygons (right)

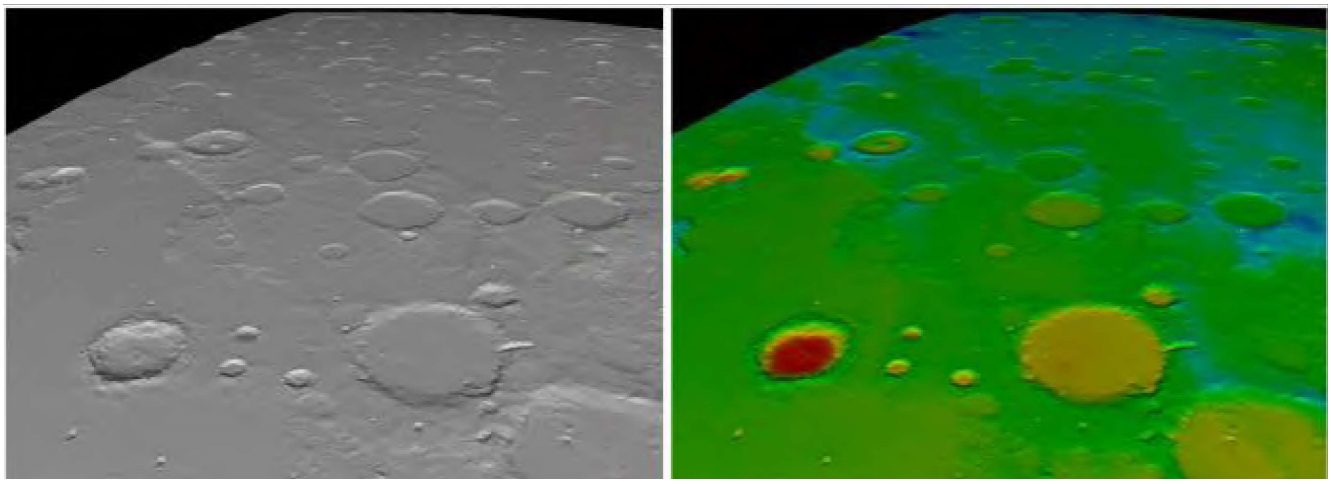

Figure 5. Comparison between the original (left) and the geographic staining (right)

This staining was obtained by the computing the radius of the planet center, located at X, Y, Z (0,0,0) in the landmark planet. It is a color scale between the minimal and the maximal height and the correct color range is assigned to each point. There is a significant improvement for the identification of small craters (a few kilometers) on the mesh. If we are very close to the surface, it was a difficult case to distinguish craters without coloring. That becomes an additional tool to facilitate the manual detection.

4.3. User Interactions and the .CRAT File

We present an ergonomic operating interface, which seems to be intuitive to users, who are accustomed to software as QGis [10]. We implemented four classes. Each of them corresponding to one of the four modes: (1) TRACKBALL uses to move the mesh to rotate and zoom in / out, and resumes the VTK usual trackball by adding features, such as changing the associated focal point with a double click. (2) ADD CRATER allows clicking on the mesh to include therein a crater. It is represented as a sphere.

Holding down the button, the size of the crater can be adjusted and it starts from the original size adjustment, with the slider lateral toolbar. (3) EDIT CRATER can "raise" a crater to its correct position and adjust its diameter. (4) REMOVE CRATER allows removing a crater. If the user makes a mistake or if he wants to invalidate a false positive, it is proposed a drag function. If we lose the click outside the bounding sphere of the crater, it is not deleted. During these actions, the overall craters list is updated in real time and is recordable, at any time, in .CRAT format. This file format is used as input or output information data for the CSV files. We can load the detection file, make all kinds of changes and save the modified file. This makes much faster the detection of craters, retaining the human critical look at the results. Finally, we illustrate the different modes and which colors are associated with the craters, as shown in Fig. 1 (central window). With green mesh are covered craters, detected by the CDA, orange mesh is added by hand and a red mesh indicates a crater ready to be removed.

5. Theoretical Solutions

5.1. Decimation of Surfaces

To streamline the interface, we discussed the idea of a hierarchy of levels of detail. It is called by the current performance of the center window. Thus, we use a quadratic mesh decimation that preserves the overall topology of the form by drastically reducing the number of polygons. It is based on the classification of points [12] according to their complexity: (1) by calculating the distance to the tangent to the surface plane and (2) by comparing the decimated surfaces and the not decimated, with a quadratic computing (improved decimation simple). It is in conjunction with the class, which manages the different LODs and allows setting the desired performance.

5.2. Picking

Another important feature is the ability to click on a 3D point and to select the mesh, corresponding to the clicked place, or picking. The mouse coordinates are tied with the 2D (on the screen) to their 3D projection on the existing surface and recovers the point, which is the nearest place, considered as a mesh. The VTK has a picking manager, which works by ray tracing. Between the camera and the clicked point, we are launching a radius straight until the launching of the mesh or the bounding sphere of a crater. Then, we made a linear interpolation and we found a contact with the point, which recovers a "real" point. This method of picking is very effective and accurate. It allows moving a crater, "sticky" to the surface below, rather than moving the camera in the plane or on an axis of the mark. It ensures that the action of moving of a crater remains faithful to other 3D craters.

5.3. Fitting / Orientation

Finally, two problems have been solved: (1) to correct an offset between the mesh and the real crater and (2) to fit the mesh covering the crater to the size of the real crater. The offset is a known error, due to the improper manual submission by geologists [11]. We propose a correction of inaccurate detections craters (see Section IV-C). To guide our disc in the space, we used the least squares optimization method. For surface fitting, we have the coordinates of the center of Mars, as information on the craters file. We calculate the direction of the Martian axe, passing through its center. We use it as a normal to orient our mesh in successive rotations. This method is more elegant in its simplicity.

6. Results

Once we have a .CRAT file, listing all the supposed detected craters by the CDA, the corresponding mesh is loaded into our visualization program. We can see the 3D detected craters. It is easy to identify the offset and to correct in some few clicks. In addition, we can add a forgotten crater.

It is difficult to represent exhaustively the results, because the detection rate is important. We can present it as a significant contribution to the detection algorithms. We illustrate the offset correction on the file, between the output of an algorithm CDA and its validation by the user on a small area of Mars. Corrections are applied in a few seconds (Figure 6 and Figure 7).

After the saving a .CRAT file, we obtain a theoretically perfect detection (if it is performed by an expert). The simplicity of the validation provides high-performance detection in a short time, by combining a starting rate and an effective manual correction.

7. Conclusion

We present an interface for validation, which allows users to browse intuitively 3D meshes. It induces a better perception of reliefs than 2D. It can move and adjust the craters. For this work, we were focused on the development phase on the fundamental

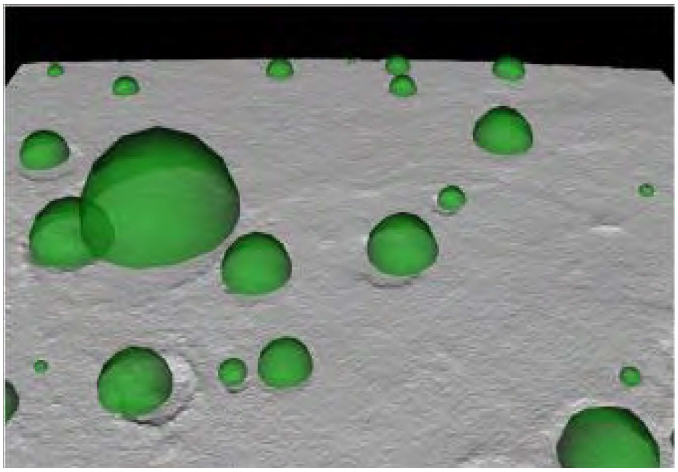

Figure 6. Visualization of craters, which are detected by a CDA

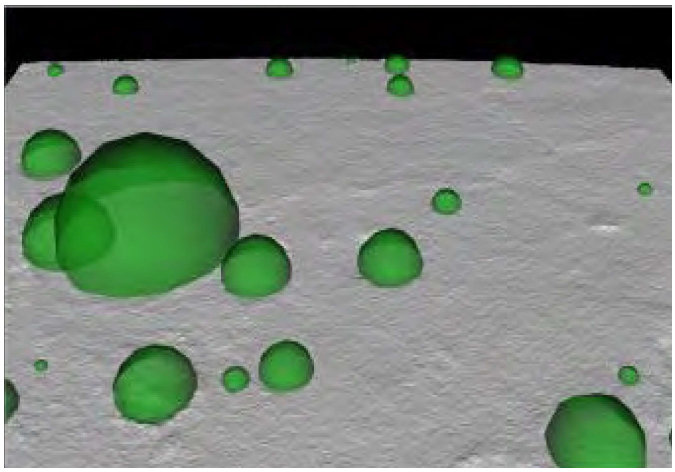

Figure 7. Visualization of craters, which are detected after manual corrections

points of the program, namely the picking and user interactions. For future work, we think to improve the performance, with the management of more than two levels of detail.

Acknowledgement

This paper was supported by Contract No 162??0033-07 of Technical University-Sofia, Research Sector. Research project: «Segmentation and modeling of geometrical characteristics of 3D objects» - 2016.

References

[1] Luo, L., Wang, X., Ji, W. & Li, C. (2011). Automated detection of lunar craters based on Chang'E-1 CCD data. Shanghaï, *The Fourth International Congress on Image and Signal Processing*, Vol. 2.

[2] Urbach, E.R. (2007). Classification of objects consisting of multiple segments with application to crater detection. *Proceedings of the Eighth International Symposium on Mathematical Morphology*. Rio de Janeiro, pp. 81–82.

[3] Barata, T., Alves, E.I., Saraiva, J. & Pina, P. (2004). Automatic recognition of impact craters on the surface of Mars. *Lecture Notes in Computer Science* (edited by A. Campiho, M. Kamel). Springer: Berlin, Heidelberg, 3212, 489–496 [DOI: 10.1007/978-3-540-30126-4_60].

[5] Salamuniccar, G. & Loncaric, S. (2010). Method for crater detection from Martian digital topography data using gradient value/orientation, morphometry, vote analysis, slip tuning, and calibration. *IEEE Transactions on Geoscience and Remote Sensing*, 48, 2317–2329 [DOI: 10.1109/TGRS.2009.2037750].

[6] Stepinski, T.F., Mendenhall, M.P. & Bue, B.D. (2009). *Machine cataloging of impact craters on Mars*, *Icarus* vol, 203, 77–87.

[7] Luo, L., Mu, L., Wang, X., Li, C., Ji, W., Zhao, J., Cai, H. (2013). Global detection of large lunar craters based on the CE-1 digital elevation model. Frontiers of Earth Science, 7, 456–464 [DOI: 10.1007/s11707-013-0361-3].

[8] Baum, F., Zanetti, M. (2015). Streamlined generalization tool for planetary surface mapping using ArcGIS ModelBuilder software on multispectral datasets, 46th Lunar and Planetary Science Conference.

[9] Kneissl, T., van Gasselt, S. & Neukum, G. (2010). New software tool for map-projection-independent crater size-frequency determination in ArcGIS, 41st Lunar and Planetary Science Conference.

[10] Ojala, T., Pietikäinen, M. & Harwood, D. (1996). A comparative study of texture measures with classification based on feature distributions. Pattern Recognition, 29, 51–59 [DOI: 10.1016/0031-3203(95)00067-4].

[11] Schroeder, W.J., Zarge, J.A. & Lorensen, W.E. (1992). Decimation of triangle meshes. ACM SIGGRAPH Computer Graphics. *Proceedings of the SIGGRAPH, Conference*, 26, 65–70 [DOI: 10.1145/142920.134010].

[12] Schroeder, W.J., Zarge, J.A. & Lorensen, W.E. (1992). Decimation of triangle meshes. ACM SIGGRAPH Computer Graphics. *Proceedings of the SIGGRAPH*, Conference, 26, 65–70 [DOI: 10.1145/142920.134010].

[13] https://wiki.qt.io/Qt_for_Beginners.

[14] https://en.wikipedia.org/wiki/Signals_and_slots. Wikipedia.