Adaptive Reconstruction of Implicit Surfaces from Depth Images

Hallison Paz
Visgraf
Instituto Nacional de Matemática Pura e Aplicada
Rio de Janeiro, Brazil
Email: hallpaz@impa.br

Luiz Velho
Visgraf
Instituto Nacional de Matemática Pura e Aplicada
Rio de Janeiro, Brazil
Email: lvelho@impa.br

***Abstract*—In this work, we propose a surface reconstruction pipeline, which operates incrementally, from depth images. Depth cameras are increasingly present in the market, while there is a move to integrate them into mobile devices, combining their data with those of other sensors in a platform with an increasing computing power. Our proposal investigates how to benefit from this integration of sensors to derive geometric models of objects through the extension and adaptation of classical geometric modeling methods. In this work, the experiments and computations were performed on a conventional computer, but the choice of solutions and algorithms is intended to test the viability of a method whose complete pipeline could be executed in a mobile device. We show the advantages gained from using the features available in a current mobile device in designing solutions to classic problems in the area, while exploring computationally efficient methods to operate according to the hardware limitation these platforms have over a regular computer.**[1]

I. Introduction

In the past, the acquisition of geometric data was restricted to academia and certain industry sectors due to the high cost of equipment and the need to operate them in controlled environments. However, devices developed for consumer applications such as the Kinect made this process cheaper and more widespread. Later on, accessories like the Structure Sensor enabled the acquisition of geometric data in a mobile environment, by attaching a depth camera to an *iPad*.

In contrast, the accuracy and quality of the data acquired by a low-cost sensor as the Kinect are significantly lower than those of the expensive and robust sensors that were being used. This situation exposed new problems and made it necessary to adapt existing ones, such as lack of information, aggravation of acquisition noise, low resolution and, in the sense of practical applications, real-time problem solving. In this way, classic problems such as noise filtering, surface reconstruction or pattern recognition remain very active due to the challenges of finding functional, robust and efficient solutions to operate in several scenarios, including real-time interaction with system users.

At this point, it is necessary to develop methods and solutions that allow the exploitation of this kind of data in consumer applications, such as the modeling of non-controlled environments, which would enable the user to walk with his device through the scene to acquire data in an intuitive and natural way. We can take advantage of mobile devices features, such as attitude sensors, multiple cameras and wireless access, to minimize the disadvantages of operating in a resource constrained system, taking into account the peculiar way in it is used. Classical results generally do not consider these features and, because of this, some computational solutions become more complex when they approach the problem with less information.

In this paper, we propose a surface reconstruction pipeline in which we capture depth images, use externally obtained calibration data, construct an extrinsic representation of the acquired object from an implicit formulation of the surface, and extract an intrinsic representation of this surface with a resolution adapted to the geometry of the object. In our experiments we acquired data using a Kinect and we performed computations in one conventional computer, but each step presented was thought of as a self-contained feasibility test of our proposal of reconstruction method, so it could be implemented and executed on a mobile device.

In addition to this pipeline, our contributions are on how to update an extrinsic representation of the surface in a scenario where depth data is acquired incrementally; how to preserve the topology of the surface during the fusion of these data and; how to construct this representation semi-adapted to the geometry of the objects, saving resources and making it possible to extract an intrinsic model in the end of each cycle of the pipeline. To accomplish the previously stated goals, we extend and adapt classical methods for a solution that addresses this paradigm of mobile computing integrated with sensors.

II. Related Work

A. Geometric Reconstruction from Depth Images

In the recent literature about surface reconstruction the Kinect Fusion [1] is an important reference for this work. The Kinect Fusion uses depth images acquired with a first generation Kinect device to construct a geometric model of the captured scene. The algorithm fuses the data from many different frames captured with a moving kinect and then generates a virtual model that can be visualized in real time.

[1]This work relates to a M.Sc. thesis

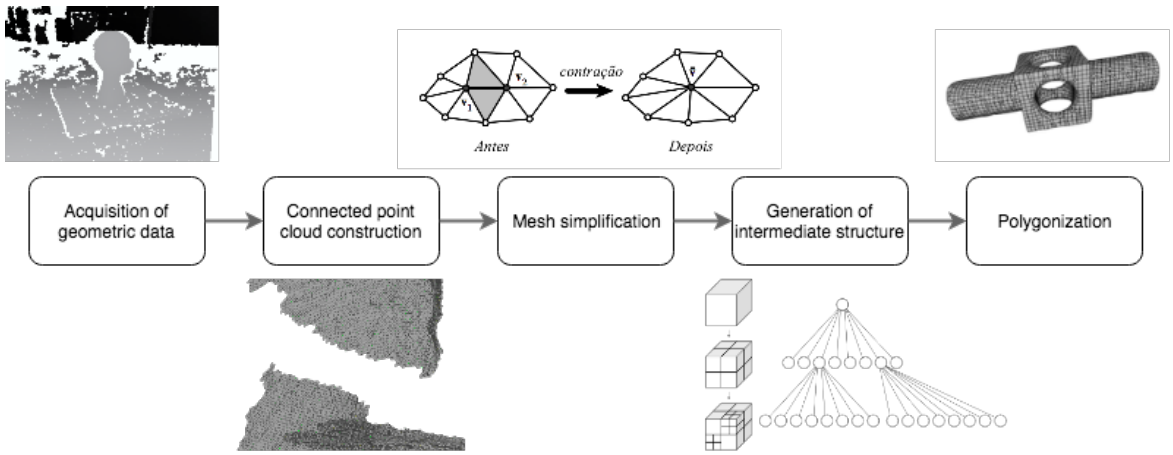


Fig. 1. Reconstruction Pipeline

Although some of the applications presented as examples in that work are similar to what we would like to get, we can highlight two conceptual differences between the Kinect Fusion and our work, both related to the fact that one case uses a static sensor without additional data, while the other adds the features available in a mobile device to the discussion.

First, the Kinect Fusion created a method to compute tracking data to align the captured frames. These calculations were made based on the already captured depth maps and the partial reconstruction of the model geometry. This is an important contribution of the project, cited both in early publications [1], [2] and in two other publications on camera relocation methods [3], [4]. In our work, we want to verify the feasibility of delegating the acquisition of the registration data to the sensors of a mobile device. We know that using only the attitude sensors of a mobile device, such as accelerometer, gyroscope and magnetometer, it is not possible to make this alignment accurately, since there is no information about the translation of the device in space. However, devices using the Structure Sensor or the Tango platform are able to perform accurate tracking with six degrees of freedom in small areas, using the integration between data from the attitude sensors, color cameras and depth sensor. For long distances, however, an approach based on these sensors may have to deal with the drift problem, where small measurement errors accumulate over time leading to a larger error situation.

The second difference we can highlight is that the Kinect Fusion works with very refined volumes, producing dense meshes. They developed very efficient solutions to use the computational power and the parallelism of graphic cards and made it possible to execute the program in real time. We want to propose a method that has a reduced memory consumption for a mobile device, seeking to make the geometric resolution adapted to the resolution of the object and not to that of the image.

B. Implicit Surfaces Polygonization

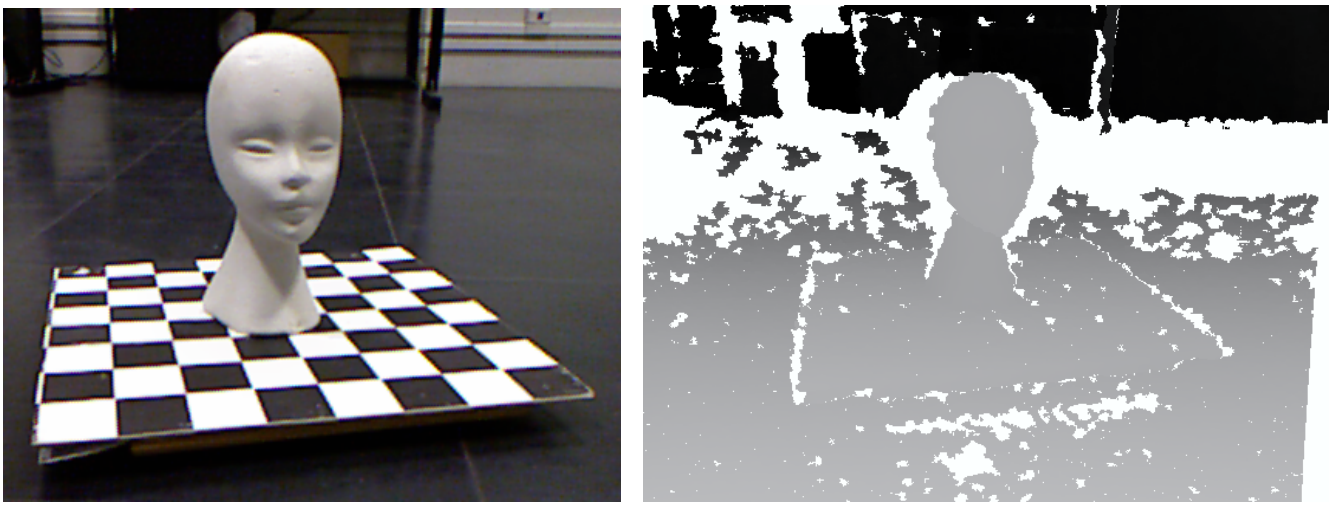
An inherent problem in geometric reconstruction applications from depth images is how to integrate data from different cameras into a single model. Trying to work with a parametric representation of the data would be quite difficult and inefficient. As a result, works like [1], [2], [5] adopt an implicit representation of the surface and perform their polygonization by implicit surface polygonization algorithms, which generally require a strategy of space segmentation.

Using an adaptive spatial structure such as an octree for this task allows the memory allocation of the algorithm to be much more efficient and the reconstructed mesh to be more adapted to the surface geometry. However, when an adaptive data structure is used, regions of the sampling grid where there is contact between cells of different detail levels can generate holes in the meshes. It is necessary to treat these cases with particular attention to generate the polygons and integrate them properly to the mesh under construction.

The *Dual Countoring algorithm* [6] presents a solution of polygonalization of implicit surfaces in non-uniform sampling grid whose vertices are classified as inside or outside the surface. The fact that we do not need the values of the implicit function at the vertices of the sampling grid, but only the classification of these vertices as inside or outside the surface, makes this method very suitable to our experiments, since we do not have a direct way to sample values of the function. We highlight two contributions of this method that are very relevant for the experiments of this work:

1. A way to calculate the position of the vertices inside the cell using Hermite data from the Quadratic Error Functions (QEF) optimization, quadratic functions similar to those presented in [7].

2. An efficient way to polygonize a surface from an adaptive spatial structure, an octree, avoiding holes in the generated mesh.



(a) Color frame (b) Depth frame

Fig. 2. Example of a captured frame

III. Reconstruction Pipeline

Our method of reconstruction is incremental and is arranged in an initialization phase and 5 main steps that can be repeated in cycle: i) Acquisition of geometric data; ii) Connected point cloud construction; iii) Mesh simplification; iv) Generation of intermediate structure; v) Polygonization. Figure 1 represents an overview of the pipeline.

Initially, we use a mobile device to capture a depth image and then we store the calibration data of the camera, which should be obtained by the sensors of the device. After that, each pixel of the image is converted to a vertex in space $\mathbb{R}^3$ and the calibration data is used to register them in the reconstruction frame. Next, we join close vertices to produce a dense triangular mesh with uniform appearance and then apply the simplification method described in [8] to the result to reduce its density and decrease the computational effort of the following steps. The simplification step is also responsible for the computation of the *quadrics*, which are useful data to compute mesh geometry. As in [8] the strategy consists in using the geometry originally acquired as reference for the quadrics and then send the quadrics and the simplified mesh to the next step.

During the initialization phase, we use the simplified mesh and the quadrics produced in the previous step to construct an extrinsic representation of the surface, stored in an Octree, which allows us to segment the space in an adapted manner. Our goal is to construct a structure similar to that presented in [6], because we can apply a modification of the algorithm *Dual Contouring*, described in that work, to obtain an adapted polygonal mesh. Every time this step is repeated in a cycle after initialization, we use the simplified mesh of the current frame to update the representation already built in the octree.

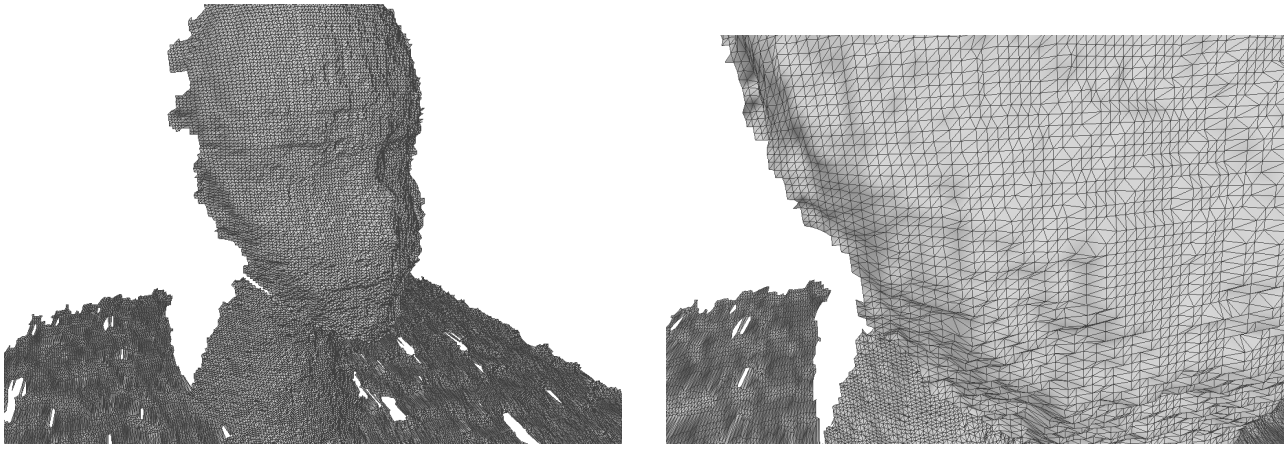
Note that each depth image presents a portion of the scene limited to its single point of view and, consequently, the surface we obtain using only the data from one frame is open, with edges. However, our chosen method for polygonization, the *Dual Contouring*, was designed to reconstruct closed surfaces. This fact led us to focus our investigative effort on solving problems inherent to the step 4 of our pipeline, in order to allow the application of this algorithm in this new usage scenario and integrate it into the pipeline. We needed to develop a new way of constructing a surface representation in a octree from the simplified point cloud and also figure out how to update this representation incrementally with data of new cameras in order to complement and close the final mesh as detailed in [9].

IV. Experiments and results

The steps of the method were performed separately so we could have more control over each one and study them individually, focusing on our own adaptations. We will present a proof of concept, addressing each of these stages since the acquisition until the polygonization. Preliminary results obtained during the research can be found in [10]. The description of the capture environment and the details of the experiment are available in [9].

A. Geometric Data Acquisition

We simulated the capture of frames with a Kinect around a mannequin head, our object of interest, obtaining 8 images that cover the surface of this object. A chessboard was introduced in the scene as a reference to perform the calibration of the camera with the *QtCalib* [11] software, based on the Tsai algorithm [12]. In our pipeline, by hypothesis, the calibration data is obtained from the sensors of the mobile device used in the capture and is an essential information to register all geometric data to the same frame of reference. Figure 2 shows one of the captured frames next to its corresponding color image.



(a) Noisy dense mesh (b) Detail

Fig. 3. Connected point cloud

B. Connected Point Cloud and Simplification

We chose a mannequin head, a smooth surface without abrupt variations of curvature, as the model for this experiments. In the Figure 3a we can see the result of taking into 3D space and connecting the points of one of the captured frames and it's possible to notice a particular high level of noise. The threshold used to connect nearby points is important, because if it's too low we might end up with a very disconnected mesh, full of holes even in areas away from the edges. In this moment it's better to build a fairly dense and connected mesh and leave geometric corrections to the mesh simplification and quadric optimization steps. The coarse resulting mesh was, then, simplified using the edge pairs collapse algorithm described in [7], reducing the number of faces by approximately 95%.

C. Extrinsic Representation and Polygonization

In this experiment, the operations of the previous steps were performed in all meshes as a pre-processing for this step, but the construction of the extrinsic representation was done incrementally, as in the use case we're interested. Due to the lack of integration in this moment, however, the quadrics that should come from the simplification step were constructed computing the intersection between the edges of the sampling grid and the simplified polygonal mesh taken as input. In our proposal it was already necessary to compute these intersections in order to classify the grid vertices as inside or outside the surface and use this information to reconstruct the surface topology; however, in this experiment we used this data - Hermite data - to compute the geometry as well. During the processing of a cell, we add to the quadric the intersection points and the normal vector for each intersection related to that cell. This way, the quadric takes into account the local tangent plane at the surface and an estimate of the trend of its movement. The data is similar to the one presented in [6], but the computation is different.

Using only the current frame, we can already perform the extraction of a polygonal mesh of this representation, as shown in Figure 4. After the initialization step, we can update the extrinsic representation by processing the following frame. Note that the sampling grid might be modified in relation to that constructed from the first frame, because it should include new informations to fill holes and possibly refine details. One of the advantages of an incremental method is that we could extract a new polygon mesh right after processing each new frame.

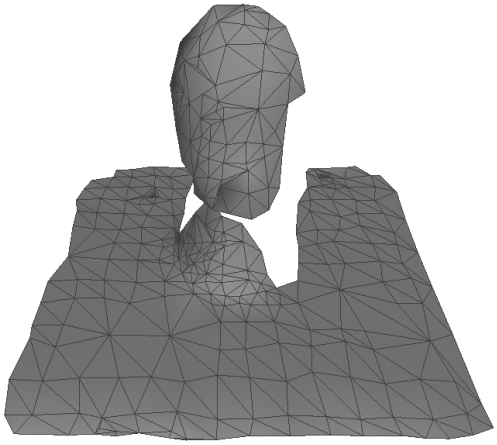


Fig. 4. Reconstruction from a single frame

In our implementation, we followed the flow proposed in our pipeline for almost all steps. The only exception is the quadric computation described in the subsection IV-C. After the construction or update of the octree, we can turn back in the cycle to acquire the next depth frame and register, connect and simplify its data to be added to the surface representation.

If the octree representation is never simplified in the middle of this process, the result of doing the polygonalization of the implicit surface in all or some of the steps of the cycle will be the same that we would get if we only processed each frame incrementally and left to extract the mesh in the end. However, if the octree is simplified in some iteration of

the cycle, there may be differences between the final result obtained in this way and the result that would be obtained continuously. After the capture of the last frame, we performed the polygonalization of the surface of interest (Figure 5).

Looking at Figure 5, we notice that the noise does not compromise much the topological structure of the reconstructed mesh, but its effect on the mesh geometry is perceptible. We see that the shape of the reconstructed mannequin is not as smooth as the shape observable in the captured frame. In the plane of the checkerboard it is possible to see small roughnesses due to the noise, besides a great unevenness due to inaccuracies in the register of the meshes in relation to the same frame of reference.

V. Conclusion

The popularization of RGB-D cameras increases the importance of effective, robust and efficient methods for mapping and interacting with the surrounding environment. The experiment carried out in this work guided us to the design of a reconstruction pipeline using existing technologies and adaptations of solutions proposed to correlated problems.

Following each step of the pipeline, it was possible to generate a polygonal mesh resulting from the merging of the acquired geometric data. The experiments showed that, although the mesh topology is adequate, the computation of the quadrics directly from the mesh intersections with the sampling grid did not improve the geometry. We believe that the integration of the quadric computation, done during the simplification step, into the next step of the pipeline would qualitatively improve the mesh. This way, we could leverage the strengths of each method in the pipeline to reconstruct surface geometry and topology.

We used a method to reconstruct closed surfaces, but in several situations, it is possible that the observed frames could not provide information to get a borderless surface. In these cases, we will observe imperfections in the reconstruction near the edges. One way to minimize this problem is to add a correction term to the quadrics of the cells near edge regions by inserting, for example, orthogonal planes to the edges as in [13]. A similar strategy could be adopted in cells near regions of high curvature, but the characteristics and impacts of the term to be added would have to be studied.

Within the scope of this work, we tested our hypotheses on each of the pipeline stages in a conventional computer, simulating situations such as registration of frames using data provided by attitude sensors. We used the results of the experiments to guide us in the formulation of the final pipeline proposal. As a future work, we propose an integrated implementation of these stages in a mobile device.

Due to the capabilities of mobile devices to connect to different forms of computer networks, we are also interested in studying how to build an efficient architecture that uses remote resources to store and process data for the reconstruction. It is possible, for example, to store geometric data along with geolocation metadata, allowing the complete scanning of establishments such as schools, museums or hospitals, by capturing images from each room and aggregating the data on the server.

Acknowledgment

This work was developed at Visgraf Lab - IMPA and was supported by IMPA and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). We also acknowledge Djalma Lucio for his aid during the setup of the capture environment.

References

[1] S. Izadi, A. Davison, A. Fitzgibbon, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, and D. Freeman, "Kinect Fusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera," *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, p. 559, 2011. [Online]. Available: http://dl.acm.org/citation.cfm?id=2047270{\%}5Cnhttp://dl.acm.org/citation.cfm?doid=2047196.2047270

[2] R. A. Newcombe, D. Molyneaux, D. Kim, A. J. Davison, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion : Real-Time Dense Surface Mapping and Tracking ."

[3] B. Glocker, S. Izadi, J. Shotton, and A. Criminisi, "Real-time rgb-d camera relocalization." IEEE, October 2013. [Online]. Available: https://www.microsoft.com/en-us/research/publication/real-time-rgb-d-camera-relocalization/

[4] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in rgb-d images." IEEE, June 2013. [Online]. Available: https://www.microsoft.com/en-us/research/publication/scene/coordinate-regression-forests-for-camera-relocalization-in-rgb-d-images

[5] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96*, pp. 303–312, 1996. [Online]. Available: http://dl.acm.org/citation.cfm?id=237170.237269

[6] T. Ju, F. Losasso, S. Schaefer, and J. Warren, "Dual contouring of hermite data," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 339–346, 2002. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-0036989769&partnerID=tZOtx3y1

[7] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," *Proceedings of the 24th annual conference on Computer graphics and interactive techniques - SIGGRAPH '97*, pp. 209–216, 1997. [Online]. Available: http://portal.acm.org/citation.cfm?doid=258734.258849

[8] M. Garland and E. Shaffer, "A multiphase approach to efficient surface simplification," *IEEE Visualization, 2002. VIS 2002.*, pp. 117–124, 2002. [Online]. Available: http://dl.acm.org/citation.cfm?id=602099.602116

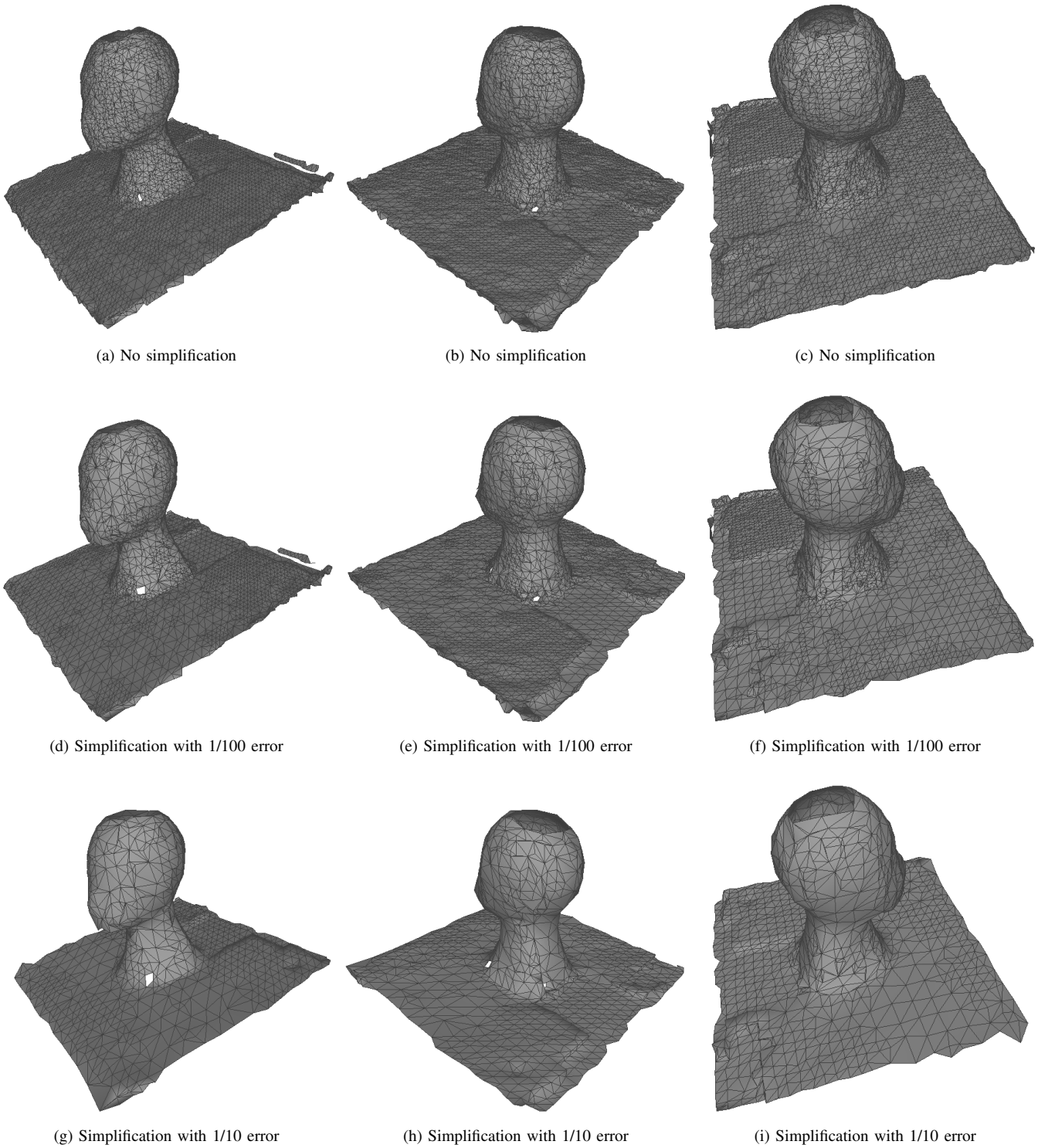
[9] H. O. Paz and L. Velho, "Reconstrução adaptativa de superfícies implícitas a partir de imagens de profundidade," Rio de Janeiro, 2017. [Online]. Available: http://preprint.impa.br/visualizar?id=7173

[10] ——, "Adaptive Polygonization Methods for RGB-D Images," Laboratório Visgraf, Rio de Janeiro, Tech. Rep., 2016. [Online]. Available: https://www.visgraf.impa.br/Data/RefBib/PS_PDF/tr-05-2016/tr-05-2016.pdf

[11] A. Zang, D. Lucio, and L. Velho, "QtCalib Software," Rio de Janeiro, 2012. [Online]. Available: http://w3.impa.br/~zang/qtcalib/

[12] R. Y. Tsai, "An efficient and accurate camera calibration technique for 3d machine vision," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 364–374, 1986.

[13] P. Lindstrom and C. T. Silva, "A Memory Insensitive Technique for Large Model Simplification," *IEEE Visualization 2001*, pp. 121–126,550, 2001.



(a) No simplification (b) No simplification (c) No simplification

(d) Simplification with 1/100 error (e) Simplification with 1/100 error (f) Simplification with 1/100 error

(g) Simplification with 1/10 error (h) Simplification with 1/10 error (i) Simplification with 1/10 error

Fig. 5. Reconstructed Object