GPU-Accelerated Enhanced Marching Cubes 33 for Fast 3D Reconstruction of Large Bone Defect CT Images

Daniel Jie Yuan Chin[1], Ahmad Sufril Azlan Mohamed[1](✉), Khairul Anuar Shariff[2,3], and Kunio Ishikawa[4]

[1] School of Computer Sciences, Universiti Sains Malaysia, 11800 Gelugor, Penang, Malaysia
sufril@usm.my

[2] School of Materials and Mineral Resources Engineering, Universiti Sains Malaysia, 14300 Nibong Tebal, Penang, Malaysia
biokhairul@usm.my

[3] Dental Materials Science and Technology Division, Faculty of Dental Medicine, Airlangga University, JI. Prof. Dr. Moestopo No. 47, Surabaya 60132, East Java, Indonesia

[4] Department of Biomaterials, Faculty of Dental Science, Kyushu University, 3-1-1 Maidashi, Higashi-ku, Fukuoka 812-8582, Japan
ishikawa@dent.kyushu-u.ac.jp

Abstract. With the advancement in three-dimensional technologies, three-dimensional reconstruction of medical images serves as reliable assistance for doctors and surgeons in evaluating and diagnosing bone defects. Amongst the existing reconstruction methods, the Marching Cubes algorithm is highly popular in the surface rendering research study. There are many improvements made over the Marching Cubes algorithm, but due to the relatively small image datasets used during evaluation, it is difficult to judge the effectiveness of the improvements on large image datasets and the reconstructed models may not be viewable in lower-end specs digital devices like tablets and smartphones. Thus, an enhancement over the extended Marching Cubes 33 with graphics processing unit acceleration to improve the reconstruction accuracy, execution time, and model portability for large image datasets is proposed in this study. The obtained results show that the proposed enhancement successfully increased the accuracy by 5.29%, decreased the execution time by 11.16%, and decreased the number of vertices and faces by 73.72%. This shows that it is possible to view bone defect models with a high similarity percentage on lower-end spec digital devices and print them out with a three-dimensional printer.

Keywords: Fast 3D reconstruction · Enhanced marching cubes 33 · Large CT images

1 Introduction

Three-dimensional (3D) reconstruction is the process of visualizing a set of two-dimensional (2D) images in its equivalent 3D form. The 3D profile is captured and visualized as a 3D model in a 3D space. 3D reconstruction plays an important role in visualizing bone defects in the medical field. It has been proven that 3D models can improve the visualization and diagnosis of bone defects [1]. In this specific study, 3D reconstruction methods are applied on a set of 2D micro-computed tomography (CT) bone defect image slices so that the 3D model of the bone defect can be visualized.

There are two main subdomains in 3D reconstruction, namely surface rendering and volume rendering. Surface rendering allows the 3D data to be visualized as a stack of isosurfaces formed through triangulation of vertices and faces whereas volume rendering allows the volume visualization of the 3D data. Between the two main subdomains, surface rendering is selected to visualize the bone defects. This is because by extracting and rendering the mesh from the volumetric data into a 3D solid object, which can be 3D printed, the structure and shape of the bone defects can be visualized and studied effectively.

There are many existing algorithms under the surface rendering technique, one of them is the Marching Cubes algorithm, which is a popular surface rendering technique in the 3D reconstruction domain due to its simplicity in implementation and fast computation with parallel processing. However, the ambiguity issue affects the reconstruction accuracy, which leads to the generation of redundant triangular faces, affecting the overall 3D model size and portability. Even though this issue is actively addressed like the Marching Cubes 33 [2] and the extended Marching Cubes 33 [3], the image datasets used are relatively smaller in size, which may question the effectiveness of the improvements. Also, the 3D model portability is rarely addressed. Hence, this paper aims to improve the reconstruction accuracy, execution time, and portability of the 3D models between higher-end spec digital devices and lower-end spec digital devices like tablets and smartphones for large image datasets.

The large image dataset used in this study is a rabbit femur defect micro-CT dataset collected from Dr Khairul Anuar Shariff, School of Materials and Mineral Resources Engineering, Universiti Sains Malaysia, using SkyScan 1076. A sample image is as illustrated in Fig. 1. The dataset consists of 952 images with every image slice having 1400 dots per inch (DPI) horizontal resolution and 1400 DPI vertical resolution. It is preprocessed through thresholding, the Canny edge detector, and area opening before the data is reconstructed into a 3D model using a laptop with 12 gigabytes (GB) random-access memory (RAM). The 3D models are then loaded to a smartphone with 4 GB RAM for visualization purposes using a 3D visualization application.

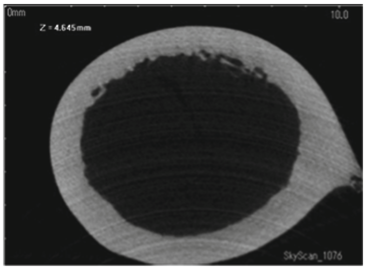

Fig. 1. Sample rabbit femur defect micro-CT image slice.

The rest of the paper is organized into several sections. In Sect. 2, the Marching Cubes, the Marching Cubes 33 and the extended Marching Cubes 33 are reviewed. In Sect. 3, the proposed enhancement over the extended Marching Cubes 33 (GPU-accelerated enhanced Marching Cubes 33) is elaborated. In Sect. 4, the obtained results on the reconstruction accuracy, execution time, and model portability are discussed. In Sect. 5, a conclusion is made about the work. This is followed by an acknowledgement and a list of references made in this paper.

2 Marching Cubes and Marching Cubes 33

The original Marching Cubes algorithm [4] first divides the input 3D volumetric data into *n* number of cubes with each cube representing a unit on isosurface. The defined *n* number of cubes are then placed between every two surfaces as elucidated in the 3D volumetric data.

When a cube is placed between two surfaces, the vertices of the cube are labelled as follows: a vertex is labelled as 1 when it falls within the boundary of the target object whereas the same vertex will be labelled as 0 when it is otherwise. Based on the labelling of all eight vertices, the index of the cube can be calculated and compared with the predefined lookup table consisting of various cube configurations. The index is then used to retrieve the configuration of the labelled cube.

In the original Marching Cubes algorithm, the cube configurations are categorized into 15 unique patterns identified by the authors [4]. The cube configuration consists of edges intersected by the vertices labelled as 1. In the triangulation step, the midpoint of the edges serves as vertices of the triangular faces. Finally, the algorithm marches on to the next cube in place.

The advantage of the Marching Cubes algorithm is the support for parallel processing. As each cube can be processed independently of one another, the overall reconstruction time can be reduced by processing the cubes in parallel. However, the number of triangular faces increases greatly when the size of the 3D volumetric data increases as well. This leads to bigger 3D models, affecting their portability between higher-end spec digital devices and lower-end spec digital devices. Also, ambiguity issue exists which leads to the formation of holes on the surfaces in certain cube configuration combinations. Despite that, the Marching Cubes algorithm is still a popular 3D reconstruction algorithm in recent years. For example, it is used in assisting the diagnosis of lumbar intervertebral disk herniation [5].

An improvement over the Marching Cubes algorithm is introduced by redefining all the 15 cube configurations into 33 cube configurations, which is named Marching Cubes 33 [2]. This improvement is introduced to cover the majority of the complex trilinear function of topology cases, effectively addressing the ambiguity issue on not just the surface but also within the cube itself [2]. The Marching Cubes 33 is further extended in a recent study by grouping all 33 cube configurations into 3 different categories during the triangulation step [3]. Through the three triangulation groupings, an extended Marching Cubes 33 triangulation is performed to support all complex cube topology cases in trilinear interpolation.

The groupings defined by the authors are based on the final shape of the isosurface in the cube after deformation. For deformed isosurface resembling a disc without requiring an additional point to support the representation, it is grouped into the simple leaves triangulation group [3]. For a deformed isosurface resembling a cylinder, it is grouped into the tunnel triangulation group [3]. For deformed isosurface resembling a disc requiring an additional vertex in the center of the cube to ensure its correct representation, it is grouped into the interior point leaves triangulation group [3]. After verifying all the connectivity between vertices and grouping all the cubes into any of the three categories, triangulation is performed following every category's triangulation rules and processes.

The advantage of the extended Marching Cubes 33 is the triangulation quality of the reconstructed 3D models are vastly improved due to the support for complex cube topology cases which solves the ambiguity issue [3]. However, it is noticed that the image dataset used in the study are relatively small, which questions the performance of the algorithm when large image datasets are used in testing. It is also noticed that there is no mention of whether the execution time is improved.

3 GPU-Accelerated Enhanced Marching Cubes 33

In this study, an enhancement over the extended Marching Cubes 33 is proposed to improve not just the reconstruction accuracy, but also the execution time and the portability of the reconstructed 3D models. The flowchart of the proposed enhancement is as illustrated in Fig. 2.

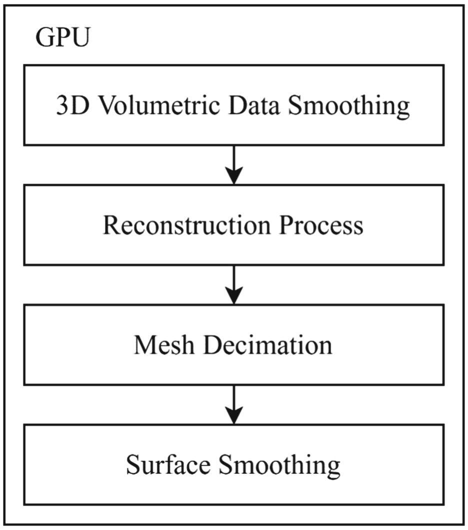

Fig. 2. Flowchart of GPU-accelerated enhanced Marching Cubes 33.

The proposed enhancement included three additional steps in addition to the extended Marching Cubes 33 algorithm. The first additional step is the smoothing of the 3D volumetric data before the reconstruction process. 3D data smoothing is a commonly applied technique in removing noises from the volumetric data itself. Noises, in this context, refer to the outliers in the 3D volumetric data which leads to wrong triangulation. Wrong triangulation often leads to surface mesh with redundant triangular faces and sharp edges, affecting the reconstruction accuracy. In this proposed enhancement, a 3-by-3-by-3 Gaussian filter is chosen as the smoothing method due to its simple implementation. Also, the result of the smoothing is independent of the starting point and the direction of the smoothing, which means better control over the smoothing result.

The reconstruction process is where the extended Marching Cubes 33 take place. The input is the smoothed 3D volumetric data from the 3D volumetric data smoothing step, and the output is a rendering of the reconstructed 3D model, a list of vertices and a list of faces. The rendering time is also recorded as part of the execution time evaluation. The list of vertices and faces are then used in the next step, which is the mesh decimation.

The second additional step is the mesh decimation, or in this case, faces and vertices decimation after the reconstruction step. Mesh decimation, or also known as mesh simplification, reduces the number of triangular patches, or in this case, the faces and vertices, resulting in a smaller 3D model size. This is achieved by manipulating the vertices' position which removes duplicated vertices. While mesh decimation is an effective step in improving the portability of the 3D models between higher-end spec digital devices and lower-end spec digital devices, it often leads to deformed surfaces, which affects the overall view of the 3D model when visualized. In this study, remeshing applied to the whole 3D model with a reduction factor specifying the percentage of the vertices and faces that should be maintained is implemented for the mesh decimation step. After computing the contraction cost between two vertices by combining the error matrices calculated for both vertices, the vertices are collapsed one by one, and vertices affected by this contraction will also be collapsed. The error metric is based on the quadric error metric [6]. This is repeated until the reduction factor is met. Two additional conditions are added to the remeshing rule, which preserves the mesh boundary and the vertex normal. This is to prevent extreme mesh deformation.

The third additional step is mesh smoothing after the mesh decimation step. Similar to 3D volumetric data smoothing, the purpose of mesh smoothing is to improve the visual representation of the surface meshes by manipulating the vertices' position. This results in reduced noises, or outliers in vertices, and smoother surface meshes. In this proposed enhancement, the Laplacian smoothing with preserved surface meshes is applied for the mesh smoothing step to prevent further deformation of the 3D model. The vertices are adjusted towards the average position of their neighboring vertices only if their new position still lies on the surface mesh. This is achieved by making sure that the displacement angle of the new position from its original position does not exceed a specified threshold. This is repeated for *i* number of iterations. It is recommended that the threshold is kept to a smaller degree. In this study, the threshold is kept at 0.5°, which is half of the maximum recommended threshold.

Finally, the whole reconstruction method, which includes the three additional steps namely 3D volumetric data smoothing, mesh decimation, and surface smoothing is executed with a graphics processing unit (GPU) to speed up the execution and rendering time of the 3D model. The GPU used in this work is the NVIDIA Geforce RTX 2060, which uses the Turing microarchitecture, supports core speed of up to 1680 megahertz (MHz) when boosted and memory speed of up to 14 gigabits per second (Gbps), memory bandwidth of up to 336 GB per second, and memory type of Graphics Double Data Rate 6 (GDDR6).

4 Results and Discussion

The reconstruction accuracy is evaluated by calculating the similarity percentage between the reconstructed model and the original model. Due to hardware limitations, 2D black and white images are captured for the 3D models at the same orientation and compared with the original 2D images in the image dataset which serves as the ground truth. Root mean squared error (RMSE) and structural similarity index measure (SSIM) are used to evaluate the similarity between both images. RMSE is used instead of mean squared error (MSE) as MSE can be easily biased towards higher values. The execution time is evaluated by recording the time taken, in seconds, for the whole reconstruction and rendering process five times and the time taken is averaged out. The execution time does not include the time taken for loading the image stack, image preprocessing, and exporting the 3D models. The model portability is evaluated by calculating the percentage decrease in the number of vertices and faces, and the smoothness of the 3D model manipulation and visualization on a smartphone.

For the results in Table 1, the extended Marching Cubes 33 will be represented as MC33, the extended Marching Cubes 33 accelerated with GPU as MC33GPU, and the proposed enhanced Marching Cubes 33 as EMC33GPU with different parameter values combination, namely the reduction factor for the remeshing and the number of iterations for the Laplacian smoothing.

There are a few things to be noted on the obtained results tabulated in Table 1. Firstly, reduction factors of 0.0 and 0.1 are excluded in this study as the reconstructed model is unable to be opened and viewed. Secondly, reduction factors 0.8 and 0.9 are also excluded in this study as the reconstructed models have the same number of vertices and faces as the reconstructed models with a 0.7 reduction factor. Thirdly, reduction factor 1.0 is excluded in this study as there is no reduction in the number of vertices and faces. Any reduction factors that do not fall in the range of 0.0 and 1.0 are rejected as the values are out of the parameter's acceptable value range. Lastly, for the number of Laplacian smoothing iterations, 100 and any number higher than that are excluded in this study as the results similarity percentages showed no further increase and the increase in execution time is very huge. As the execution time is also evaluated in this study, it is determined that any parameter value combination that leads to an increase in execution time but zero increase in similarity percentages is ignored. Negative-numbered and 0 iterations are rejected as it is impossible to run Laplacian smoothing with zero and negative iteration.

Using the obtained results in Table 1, the best parameter value combination is selected by first computing the percentage of increase in similarity percentage for RMSE and SSIM, percentage decrease in execution time, and percentage decrease in vertices and faces. The calculated percentages are divided by 100 to obtain the percentages in decimal form. This is followed by multiplying the percentage of increase/decrease in decimal form with a weighted score. The weightage score is assigned based on the focus of the enhancement. In this case, the weightage score assigned to reconstruction accuracy is 0.6,

Table 1. Results for MC33, MC33GPU, and EMC33GPU.

Methods	Reduction factor	Iterations	RMSE (%)	SSIM (%)	Reconstruct (s)	Rendering (s)	Faces	Vertices
MC33	–	–	69.48	85.47	72.24	14.23	5837093	2798075
MC33GPU	–	–	69.48	85.47	66.97	14.64	5837093	2798075
EMC33GPU	0.2	1	74.09	88.84	72.14	0.36	1503286	750127
EMC33GPU	0.2	5	74.08	88.83	72.84	0.40	1503286	750127
EMC33GPU	0.2	10	74.08	88.83	73.33	0.39	1503286	750127
EMC33GPU	0.2	25	74.08	88.83	77.62	0.42	1503286	750127
EMC33GPU	0.2	50	74.08	88.83	85.69	0.40	1503286	750127
EMC33GPU	0.3	1	74.14	88.83	73.84	0.45	2254929	1125945
EMC33GPU	0.3	5	74.16	88.81	75.58	0.41	2254929	1125945
EMC33GPU	0.3	10	74.16	88.81	77.64	0.47	2254929	1125945
EMC33GPU	0.3	25	74.16	88.81	84.59	0.43	2254929	1125945
EMC33GPU	0.3	50	74.16	88.81	94.63	0.40	2254929	1125945
EMC33GPU	0.4	1	74.14	88.84	74.06	0.42	3006572	1501400
EMC33GPU	0.4	5	74.18	88.82	76.28	0.44	3006572	1501400
EMC33GPU	0.4	10	74.19	88.82	79.14	0.41	3006572	1501400
EMC33GPU	0.4	25	74.19	88.82	87.29	0.39	3006572	1501400
EMC33GPU	0.4	50	74.19	88.82	91.16	0.44	3006572	1501400
EMC33GPU	0.5	1	74.12	88.83	70.95	1.27	3758215	1876913
EMC33GPU	0.5	5	74.14	88.80	73.42	1.26	3758215	1876913
EMC33GPU	0.5	10	74.15	88.80	73.70	1.28	3758215	1876913
EMC33GPU	0.5	25	74.14	88.78	87.19	1.29	3758215	1876913
EMC33GPU	0.5	50	74.13	88.76	104.06	1.26	3758215	1876913
EMC33GPU	0.6	1	74.12	88.84	71.48	1.98	4509859	2252288
EMC33GPU	0.6	5	74.13	88.83	74.64	2.01	4509859	2252288
EMC33GPU	0.6	10	74.16	88.84	78.58	2.06	4509859	2252288
EMC33GPU	0.6	25	74.18	88.84	90.9	2.04	4509859	2252288
EMC33GPU	0.6	50	74.19	88.84	110.15	2.01	4509859	2252288
EMC33GPU	0.7	1	74.14	88.86	73.37	2.67	4863552	2427460
EMC33GPU	0.7	5	74.17	88.87	76.74	2.61	4863552	2427460
EMC33GPU	0.7	10	74.22	88.88	81.17	2.61	4863552	2427460
EMC33GPU	0.7	25	74.25	88.89	93.78	2.63	4863552	2427460
EMC33GPU	0.7	50	74.28	88.89	115.67	2.62	4863552	2427460

0.3 for the reduction in vertices and faces, and 0.1 for execution time, which totals to 1.0. Execution time is given the lowest weightage score as it is hardware dependent and the execution time obtained per run is not consistent. After multiplying every percentage of increase/decrease in decimal form by the weightage score, the total enhancement score is obtained by adding them up. Parameter value combination with the enhancement score closest to 1.0 is selected for comparison with the original algorithm. The values are tabulated in Table 2.

Table 2. Enhancement score for every parameter value combination.

Reduction factor	1 iteration	5 iterations	10 iterations	25 iterations	50 iterations
0.2	0.2641	0.2631	0.2625	0.2572	0.2473
0.3	0.2226	0.2205	0.2179	0.2094	0.1972
0.4	0.1829	0.1803	0.1769	0.1669	0.1621
0.5	0.1461	0.1431	0.1428	0.1261	0.1054
0.6	0.1052	0.1013	0.0966	0.0816	0.0096
0.7	0.0837	0.0798	0.0747	0.0594	0.0326

Based on the calculated enhancement scores in Table 2, it seems that GPU-accelerated enhanced Marching Cubes 33 with remeshing reduction factor of 0.2 and 1 iteration of Laplacian smoothing is the best parameter combination with the highest enhancement score of 0.2641. From Table 1, it is noticed that the increase in similarity percentages when the reduction factor is increased is only by a very small margin while the number of vertices and faces increased by a very huge margin and the execution time increased as well. Similarly, when the number of Laplacian smoothing iterations increased, the similarity percentages showed very small changes while the execution time greatly increased and the number of vertices and faces remained the same. This means that to achieve a good balance between the increase in reconstruction accuracy, the decrease in execution time and the number of vertices and faces, it is better to use a lower value for Laplacian smoothing iterations and remeshing reduction factor. The reason behind the same number of vertices and faces despite the increase in Laplacian smoothing iteration is because the mesh surface is preserved.

Comparing with original extended Marching Cubes 33 for reconstruction accuracy and reduction in vertices and faces, and with the GPU-accelerated extended Marching Cubes 33 for execution time to prove that the reduction in execution time is not solely due to GPU, the proposed enhancement increased in reconstruction accuracy by an average of 5.29%, reduced the execution time by 11.16%, and reduced the number of vertices and faces by an average of 73.72%. The average percentage increase in the reconstruction accuracy is obtained by averaging out the percentage increase in RMSE and SSIM. The percentage decrease in the execution time is obtained by calculating the percentage decrease in the total execution time, which is the reconstruction time added with the rendering time. The average percentage decrease in the number of vertices and faces is obtained by averaging out the percentage decrease in number of vertices and faces. It is also noted that comparing both models, the proposed enhancement's model can load and manipulate with no major delay, whereas the original algorithm's model loads slowly with delays in manipulation. Besides that, the proposed enhancement's model can be 3D printed at a shorter time than the original algorithm's model with Ultimaker Cura. Screenshots of the 3D models reconstructed using the extended Marching Cubes 33 and the proposed enhancement are elucidated in Fig. 3.

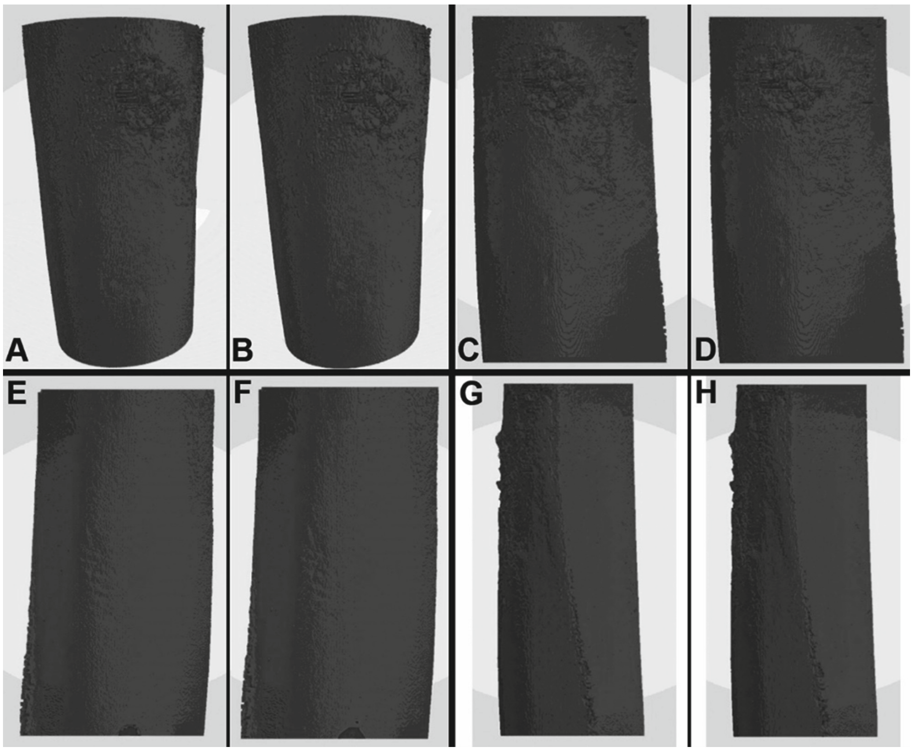

Fig. 3. Tilted view of MC33 3D model (A), tilted view of EMC33GPU 0.2 reduction factor 1 iteration 3D model (B), front view of MC33 3D model (C), front view of EMC33GPU 0.2 reduction factor 1 iteration 3D model (D), back view of MC33 3D model (E), back view of EMC33GPU 0.2 reduction factor 1 iteration 3D model (F), side view of MC33 3D model (G), side view of EMC33GPU 0.2 reduction factor 1 iteration 3D model (H).

The execution time for the extended Marching Cubes 33 and the proposed enhancement (with and without GPU) in GPU and central processing unit (CPU) environments is also tabulated in Table 3. This is to compare the performance of both methods in different environments using the same device.

Table 3. Execution time with GPU and CPU.

Reconstruction methods	Reconstruct (s)	Rendering (s)
Extended Marching Cubes 33 with CPU	72.24	14.23
Extended Marching Cubes 33 with GPU	66.97	14.64
Proposed enhancement with CPU	85.62	0.44
Proposed enhancement with GPU	72.14	0.36

Based on the obtained results, it is noted that the total execution time, which is the reconstruct time added with the rendering time, for the proposed enhancement and the original method ran in CPU environment is roughly the same, with the proposed enhancement only slightly faster than the original method by 0.41 s. This is because although additional processes are involved in the proposed enhancement, the increase in reconstruction time is offset by the decrease in the time needed to load the 3D model. When both methods are performed in a GPU environment, the difference in the total execution time increased to 9.11 s. The reason behind such drastic improvement in the execution time is because the 3D volumetric smoothing, mesh decimation, and surface smoothing steps are all boosted by GPU besides the reconstruction process. Assuming that the extended Marching Cubes 33 is considered as one process and the three additional steps in the proposed enhancement are considered as three processes, while the reconstruction time for the original method is decreased from boosting one process, the reconstruction time for the proposed enhancement is decreased from boosting four processes. While it is proven that the improvement in the execution time is not solely due to GPU, with the proposed enhancement running faster than the original method by 0.41 s in the CPU environment, it is also proven that GPU played a huge role in drastically improving the execution time.

It is also noted that the rendering time is roughly the same when running both methods in GPU and CPU environments. This is because the 3D models are rendered using the native renderer instead of loading the models with 3D rendering software. If the 3D models are rendered with a 3D rendering software and the software is also configured to run in GPU, the rendering time will be drastically improved as well.

5 Conclusion

In conclusion, the proposed enhancement (GPU-accelerated extended Marching Cubes 33 with Gaussian smoothing, remeshing reduction factor of 0.2 and Laplacian smoothing 1 iteration) has successfully increased in reconstruction accuracy by 5.29%, reduced the execution time by 11.16%, and reduced the vertices and faces number by 73.72%. This proved that the proposed enhancement generates 3D models with greater accuracy, at a faster rate, and better portability than the original method. An extension on the proposed enhancement is possible with more different parameters and values combinations and tested with more large image datasets.

Acknowledgement. The authors are grateful to Universiti Sains Malaysia and ASEAN University Network/Southeast Asia Engineering Education Development Network (AUN/SEED-Net) Japan International Cooperation Agency (JICA) project for supporting this documented work through Special Program for Research Against COVID-19 (SPRAC) grant [304/PBA-HAN/6050449/A119].

References

1. Alasal, S.A., Alsmirat, M., Al-Mnayyis, A., Baker, Q.B., Al-Ayyoub, M.: Improving radiologists' and orthopedists' QoE in diagnosing lumbar disk herniation using 3D modeling. Int. J. Electric. Comput. Eng. **11**(5), 4336–4344 (2021)

2. Chernyaev, E.V.: Marching cubes 33: construction of topologically correct isosurfaces. In: GRAPHICON'95, pp. 1–8. Technical Report CERN CN95–17, Saint-Petersburg, Russia (1995)
3. Custodio, L., Pesco, S., Silva, C.: An extended triangulation to the marching cubes 33 algorithm. J. Braz. Comput. Soc. **25**(6), 1–18 (2019)
4. Lorensen, W., Cline, H.: Marching cubes: a high resolution 3D surface construction algorithm. In: SIGGRAPH'87 Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques 1987, vol. 21, no. 4, pp. 163–169. ACM, NY (1987)
5. Al-Mnayyis, A., Alasal, S.A., Alsmirat, M., Baker, Q.B., Alzu'bi, S.: Lumbar disk 3D modeling from limited number of MRI axial slices. Int. J. Electric. Comput. Eng. **10**(4), 4101–4108 (2020)
6. Garland, M., Heckbert, P.S.: Surface simplification using quadric error metrics. In: SIGGRAPH'97 Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques 1997, pp. 209–216. ACM, NY (1997)