A Lossy Grayscale Image Compression Based on Delaunay Triangulation

T.M.V.D. Gunasekara and A. Ramanan
Department of Computer Science, Faculty of Science, University of Jaffna
Vikum.dheemantha@gmail.com

Introduction

As a result of technical advancements in digital imaging devices, the amount of images that any single person handles is increasing continuously. High quality images not only overuse the space but also becoming the main reason that causes the Internet traffic by transmitting the images through the Internet. Thus, there is a demand for a novel image compression techniques. Basically, image compression techniques can be categorised into two types: Lossy and lossless [1]. In lossless image compression, all information of an original image are retained in the compressed image. On the other hand, lossy image compression do not consider to retain all the information on the image, it mainly focuses on saving useful information while allowing the irrelevant information to be removed from the compressed image. Even for lossy image compression [2] there is a demand because there are plenty of applications of digital images that only consider about providing the visibly equivalent images to the users. Among several image compression techniques, block-based image compression [3] is enabled by the possibility of selecting large block from the image that can be represented as a single unit. Delaunay triangulation provides a better opportunity to create triangular mesh over an image [2, 4, 5]. Delaunay triangulation generates a mesh over a region to provide a set of non-overlapping triangular elements. The generation of non-overlapping elements can be utilised to provide a better image compression technique.

Methodology

Encode:

- Construct an initial lattice by performing Canny algorithm to detect the edges on an image and extract a set of points by performing run length algorithm on the edge map returned from Canny algorithm.
- Perform Delaunay triangulation on the initial lattice.
- Split triangles by adding new vertex to the barycentre of the triangles that are not homogeneous and perform Delaunay triangulation again on the new lattice. The splitting algorithm is continuously executed until convergence.
- Merge algorithm is performed and get rid of any unwanted vertices (vertex is considered as unwanted if all the triangular elements that share the vertex have near intensity value). Then perform Delaunay triangulation on the final lattice.
- Store the triangular elements with their mean intensity value as the result of the image compression.

Decode:

- Retrieve encoded data and re-construct triangulation mesh over an empty image.
- Fill each triangle with respective intensity value.

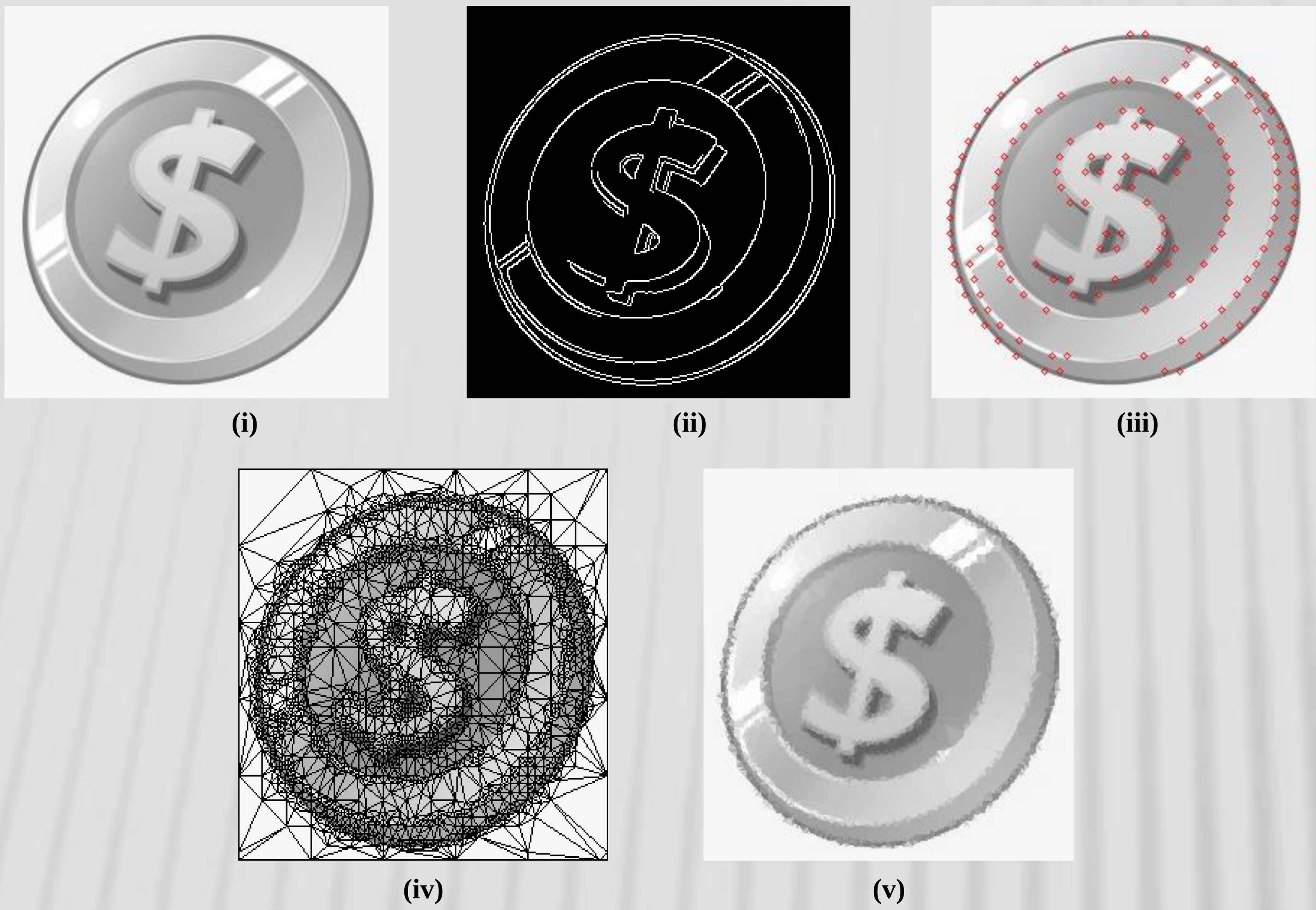

Figure 1: (i) Original image in grayscale, (ii) Canny edge map, (iii) Initial set of points extracted from the edge map in (ii) using run-length algorithm, (iv) Triangulation after performing split and merge algorithms (v) Decoded image after compression.

Results

Proposed method is tested with several grayscale images and some of the experimental results are given below.

Table 1. Experimental results of a compressed image using the proposed method with respect to the storage requirement and number of triangular elements in the generated mesh.

Image	Storage required (in Bytes)		#Triangular elements in mesh
	Original	Compressed	
Coin	65,536	25,809	3,687
French Flag	65,648	9,024	1,052
Artificial Image	326,312	225,408	25,340
Cartoon Girl	50,440	21,048	2623

- Triangles are stored using python data dump method. Hence, extra storage is consumed for store data structure details of python.

Discussion

Based on the experimental results, this study achieved better compression ratio with near quality image. But the quality and/or the compression ratio highly depends on the nature of the image. In our future work, we wish to reduce the storage requirement of 7 Bytes to store a single triangular element of a grayscale image so that a better compression ratio can be achieved.

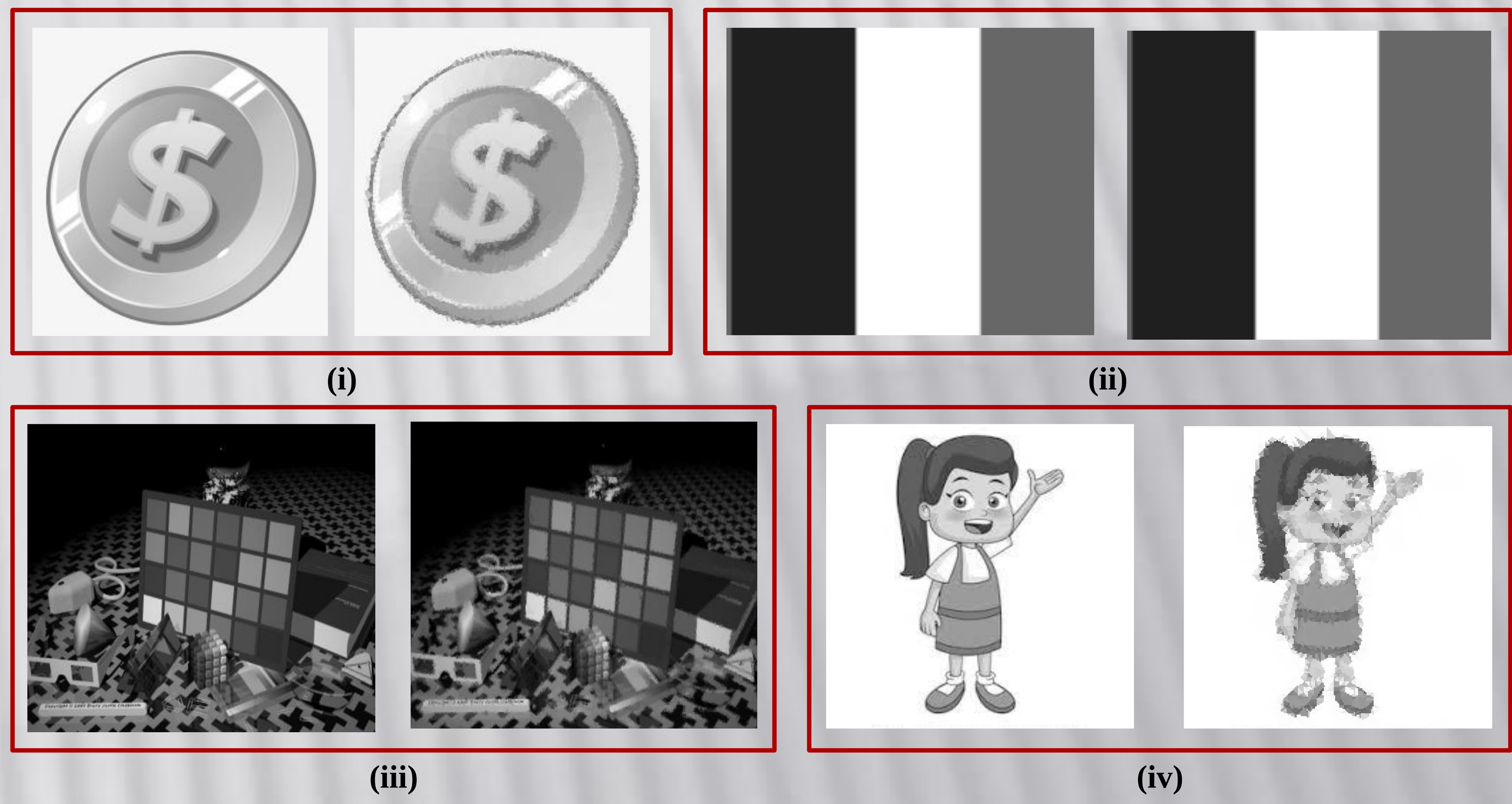

Figure 3: Visual comparisons of original images and decoded images, originals are on left and decoded images on right. (i) Image of a coin (digitally portrayed), (ii) Image of a flag, (iii) Image of some artificial component, (iv) Image of a cartoon girl.

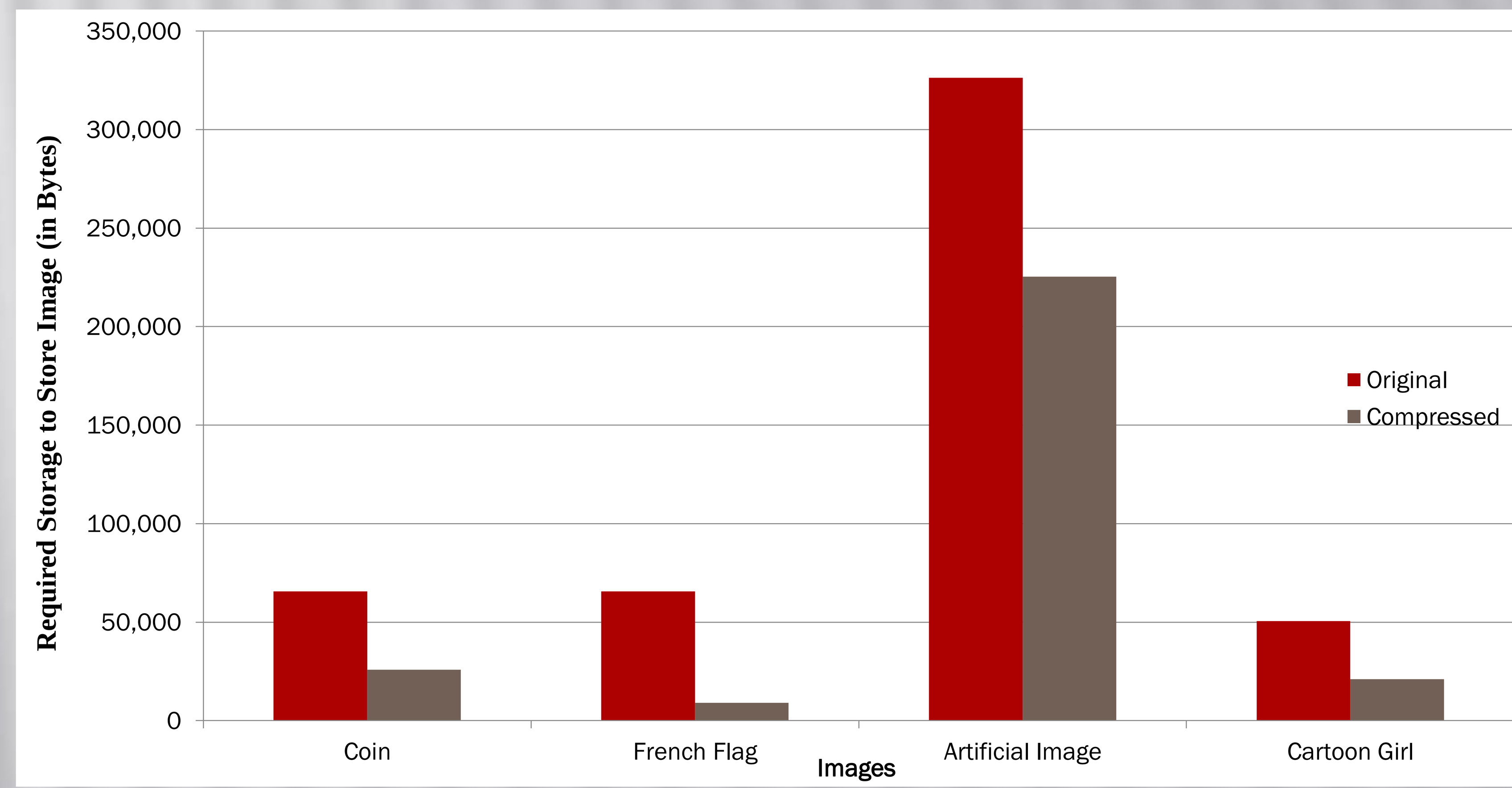

Figure 4: Comparison of the image sizes between original image and compressed image. (Size is calculated in bytes)

References

1. A.J. Hussain, A. Al-Fayadh and N. Radi, "Image compression techniques: A survey in lossless and lossy algorithms", pp.44-69, 2018.
2. D. Marwood, P. Massimino, M. Covell and S. Baluja, "Representing Images in 200 Bytes: Compression via Triangulation", 25th IEEE International Conference on Image Processing (ICIP), pp.405-409, 2018.
3. Qiang Du, M. Gunzburger, Lili Ju and X. Wang, "Centroidal Voronoi Tessellation Algorithms for Image Processing", Journal of Mathematical Imaging and Vision - JMIV, 2005.
4. D. Laurent, N. Dyn and A. Iske, "Image compression by linear splines over adaptive triangulations", Signal Processing, 86(7), pp.1604-1616, 2006.
5. F. Davoine, M. Antonini, J-M. Chassery, and M. Barlaud, "Fractal image compression based on Delaunay triangulation and vector quantization", IEEE Transactions on Image Processing, vol. 5, pp.338-346, 1996.