11th International Conference on Research in
ENGINEERING, SCIENCE & TECHNOLOGY
Budapest, Hungary
19-21 March, 2021

Analysis of Computational Vision Algorithms for Generation of Textures for Realistic Rendering

Carlos Eduardo Capanema[1]; Gustavo Aquino Torres Teixeira[1], Luiz Melk de Carvalho[1], Diva de Souza e Silva Rodrigues[1], Flávio Henrique Batista de Souza[1]

[1]Centro Universitário de Belo Horizonte, Brazil

Abstract.

This research presents an application based on the combination of image editing algorithms for the generation of auxiliary textures. It focuses on the generation of Specular Map and Bump Map from a Diffuse texture, to enable the automatic generation of high-quality images for games and movies. The image database for experiments was separated into two groups: Regular and Irregular, with 21 images of each and with different resolutions. The tool developed to make the comparison between the algorithms was developed in C #. In the proposed software, it is possible to visualize the result of the texture processing in real time on a window where the 3D rendering occurs, using the Unity graphic engine, with the texture applied on a plane. For the manipulation of textures, instead of the standard implementation (pixels are obtained and changed by calling the Bitmap GetPixel and SetPixel functions), the manipulation takes place directly in an Array. The experiment had multiple analysis of the following algorithms combined in pairs: Histogram Equalization, Adaptive Histogram Equalization, Contrast Limited Histogram Equalization, Grayscale Equal Weights, Desaturation, Grayscale with Luma Correction. The evaluated parameters were the time to generate the textures Bump Map and Specular Map, as well as the minimum and maximum values of parameters outputted by the algorithms. Those parameters represent a visual qualitative analysis of the final render. In preliminary tests the proposed strategy proved to be at least 4 and up to 70 times faster than the standard version present in the literature for adjusting brightness, grayscale and contrast in a 1080 x 768 image resolution.

Keywords: Image processing, Rendering process, Specular Map, Bump Map, Diffuse

1. Introduction

Rendering is a digital process applied on image manipulation and adjustment and requires intensive calculations and ostensibly consumes the hardware resources employed in the task (Akenine-Möller et al.,2019). Depending on the size and complexity of the image being optimized, its processing becomes quite costly in time issues, increasing production costs. According to Aristidou et al. (2018), the pursuit of realism through the improvement of algorithms would be one of the main motivations of research focused on computer graphics. As an example, in the production of the Monsters University animation, Pixar had used 2,000 computers, with approximately 100 million CPU hours (if added to the hours worked by each

core of the processors of each machine used in the process) to perform the rendering of the film in your final state (Takahashi, 2013).

Few software provide this image processing with little human intervention, which means that, many manual adjustments are necessary to obtain good results, and they have a high commercial value, leading the end user to have a high expense for a return on investment in long term. Manually it is possible to perform this task in several image editing software or specialized for this purpose, at the cost of inconsistency in the quality of the images between collaborators and delay in the validation of results, since the free solutions mostly have no sample in real time. In these cases, it is necessary to import the resulting image into the software in which the film or game is being developed and apply it to an object, and, if it is unsatisfactory, to return to the editing process.

According to Seeram (2019), a digital image is a representation of a real image as a set of numbers, which can be stored and manipulated by a computer. To translate the image into numbers, it is divided into small areas called pixels. The texture of an image, according to Nixon & Aguado (2019), is a set of metrics calculated during its processing, which represent information about the spatial arrangement of colors or intensities in each analyzed pixel.

As stated by Blinn (1978), rendering is the process of generating a digital product from a specific input. Bump Map is a rendering technique that consists of assigning a wavy (or rough) aspect, without the need to model such effects, through an angular variation obtained in a texture map. Specular Map is a texture variation to simulate brightness on a surface by estimating the angle of the shadows.

As specified by Woo et al. (2017), Diffuse is the reflection of light on a surface in which a light source comes from several angles, unlike the Specular case where it is only a reflected angle. It is not known which of the available algorithms are the most efficient for generating realistic rendering auxiliary images from Diffuse texture. It is also not known if there are groups of textures with common characteristics in which the same processing can be done, as well as the intensity adjustments of each algorithm for each case in order to normalize this process and decrease the manual and subjective editing of the professional.

Based on that, the objective of this research is to analyse the existence of algorithms (or a combination of them) as well as the variation in the parameters received by them, that is more efficient in the generation of Specular Map and Bump Map from a Diffuse Map, in order to enable, in the future, more automated and standardized means for the generation of these images with quality and reducing costs of the process of films and digital games production. Thus, as specific objectives, it is aimed to: determine, if any, how textures can be classified and grouped; to realize the generation of the auxiliary images with different combinations and adjustments in these algorithms; analyse the efficiency of the algorithms by the groups of images found. Some papers are presented in the literature that have a similar slope to what is developed in the present paper: Tan and Ikeuchi (2008) made a research about the presence of highlights, which in non-homogeneous dielectric objects are linear combinations of Specular and diffuse reflection components, which is inevitable; and Huang et al. (2013) proposes an efficient method to modify histograms and to improve the contrast in digital images.

2. Methods

2.1 Image Treatment Process Analysis

The image processing relies on the analysis of four basic concepts: digital image, texture, rendering and histogram. Transparency Market Research (2015) points that the digital imaging market is increasingly being embraced by government and business agencies to increase productivity and provide greater access to certain types of information. The great companies group in the digital imaging market include companies such as Canon Inc., Microsoft, Sony Corporation, among others. This market owes its growth to various possibilities arising from the treatment applied to the final product, such as: recovering image without errors or losses, reducing the need for physical storage space and eliminating environmental problems (caused by film-based images).

As reported by Lukac & Plataniotis (2018), system requirements are essential for image processing. The amount of memory, processing speed and storage capacity are crucial for a good procedure. Normally, to facilitate the processing of the image, a transformation of the image into a reduced range of information is done through segmentation. Thresholding consists of segmenting the image into regions that correspond to the structural units of the scene, or that distinguish the objects of interest, separating the objects from the image, which corresponds to the foreground without the background image information.

For Nixon & Aguado (2019), associated with an image, there is its texture, which is a color map applied over an existing surface. It is created by a regular repetition of an element or pattern, called texel. There are regular textures, which are formed from a fixed geometric shape, and irregular textures, which are typically represented in terms of spatial frequency properties, some of these being textures of wood or cement.

For the treatment of images, rendering consists on the activity of generating images or videos through a computer. Render research and development have been largely motivated by the discovery of rendering simulation in efficient ways. Some discoveries relate directly to particular algorithms and techniques, while others are produced together.

Some concepts and techniques used are described: *Bump Mapping* (relief simulation method of small scale for surfaces); *Diffraction* (reflection, scattering and interference of light passing through an object or aperture that disturbs the ray); *Indirect lighting* (surfaces illuminated by light reflected from other surfaces, not directly from a light source); *Texture Mapping* (method of applying detail to surfaces); *Noise* (random variation of brightness or color in an image); *Refraction* (flexibility of light associated with transparency); *Saturation* (relative color of an area proportional to its brightness); *Texel* (a fundamental unit of a texture map).

Szeliski (2010) says that a histogram is a simple pixel count at each intensity level for both color and brightness channels of an image. For an 8-bit image an accumulation table of 256 columns is required ($2^8 = 256$). The closer to pure black, the more left to the table, and the closer to pure white, the more to the right.

2.2 Experimentation Materials and Methods

The base of images for experiments was selected from the internet, randomly and from several sites and image repositories. They were separated into two groups, called Regular and

Irregular, as previously described. The comparison solution between the algorithms was developed in the C# language using Visual Studio IDE. The type of application was the Windows Form Presentation with Unity. The library used as interface was the Noesis GUI, which enables XAML code to be used in Unity with some limitations. In this software the user inserts the diffuse type image and, from it, the implemented algorithms are applied without using more than one histogram equalization algorithm and one grayscale type for each processed image.

The tests were performed on a desktop computer, with i5-3550 processor, 8GB of RAM, NVIDIA GTX 1060 6GB video board and 240GB Kingston SSD storage system. The software has a window where it is possible to see the result of the real-time texture processing, and another window where the 3D rendering takes place using the Unity graphics engine to apply the texture on a plane. For manipulation of textures, instead of the default implementation where pixels are obtained and changed by calling the GetPixel and SetPixel Bitmap functions, manipulation occurs directly in an array. This technique was chosen because in the standard methods for each function call the image is copied completely to a new array, then only one pixel is manipulated, and the data is copied back.

In preliminary tests this strategy was shown to be 4 to 70 times faster than the standard version for just a brightness, grayscale and contrast adjustment in a 1080 x 768px image. The default runtime ranged from 400 milliseconds (ms) to 13 seconds while the other ranged from 58ms to 230ms, depending on the type of algorithm.

To optimize the processing time in the use of multiple algorithms, a list was implemented containing the result of the image in an array, the name of the algorithm executed, and the variables entered by the user. Every time the user modifies a parameter on the interface that is attached to an image-editing algorithm it is added to the end of the list if it does not exist or replaced in the position where it existed before. In the case of the substitution, the pixels from the previous position of the list are copied to the current one before the processing, and then the current one is copied to the next position and the algorithm registered for that position is executed until all the results are updated.

In this way, not all algorithms are executed every time the user changes a variable, but only those of the position in which the algorithm was registered on and forward. The Unity render scene and application settings of the images have been configured as: Omnidirecional; Color: White; Intensity: 3; Shadow Type: Soft Shadows; Resolution: Very High; Specular smoothness: 0.2; Bump Map smoothness: 0.5. The parameters evaluated during the tests consist of the time to generate the Bump Map and Specular Map textures, as well as the minimum and maximum values of the parameters of the algorithms with which a satisfactory result can be obtained, this being a qualitative analysis of the final rendering.

3. Results

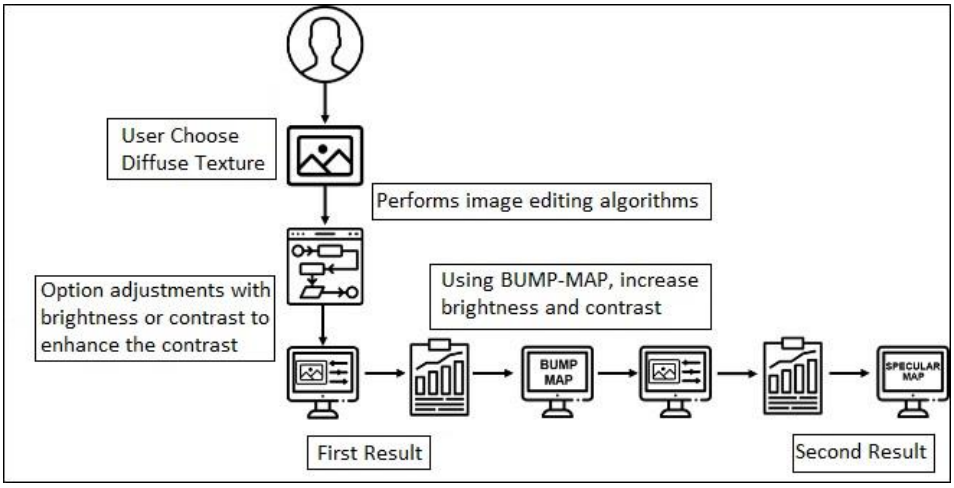
3.1 Structure of Tests

It was structured a sequence of steps to be performed in order to standardize the tests as described in Figure 1. Firstly a “selecting image process” is performed. At this stage, a dialog box opens in which the user selects an image of type JPG or PNG, of any size. A manual

edition is executed, for each selected auxiliary image, using an equalization algorithm type: one of grayscale type, brightness and contrast according to the need.

Secondly, for each combination of algorithms it is recorded the value of all variables used, trying to identify the minimum and maximum for an acceptable result; the variables are which algorithms were used, the size of the window for the histogram algorithms, brightness intensity and contrast. Then, the flows with the best visual quality, if any, are recorded. The variables used in the processing and elapsed time are copied from the program to a worksheet. The comparison of values found; for each possible flow in relation to the chosen algorithms is analysed to check if there is a range of values that would obtain a satisfactory result for all of them, using the data of the regular and irregular groups separately, and then together. Flows in which the final renderings present significant unwanted artefacts are registered as invalid.

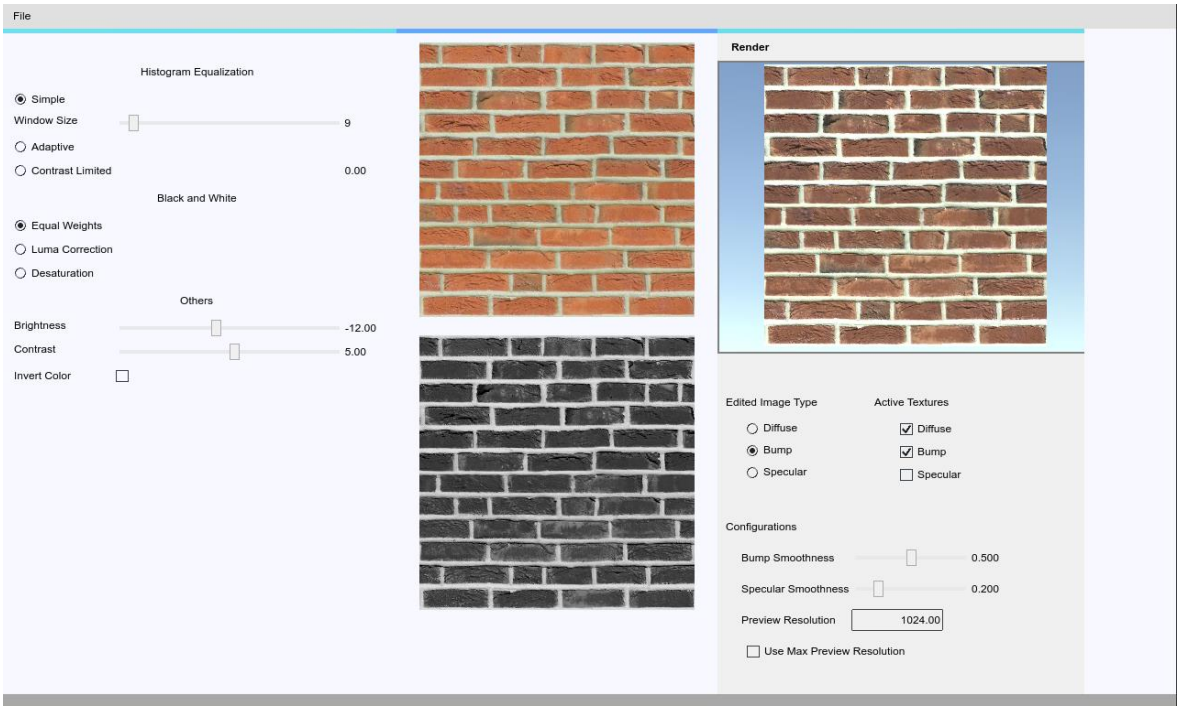
Figure 1: Usage sequence diagram



Source: (Authors)

Figure 2 presents the program interface with a loaded image. On the left side are the selectors of the algorithms, in the middle are the original and below that the resulting image, and at the right side all of them combined and applied in a plane in a 3D real-time render.

Figure 2: Program interface



Source: (Authors)

3.2 Experiments

The experiments were done with 42 images randomly selected from the internet, where 21 are regular and 21 irregular, which was determined considering that, although the execution of the algorithms was practically in real time, a visual analysis of the quality of the results, which is time-consuming. About these 42 images, 33 have the resolution of 1024 x 1024 pixels, 7 have 4256 x 2832 pixels, 1 has 4928 x 3264 pixels and the last one has 2832 x 2832 pixels.

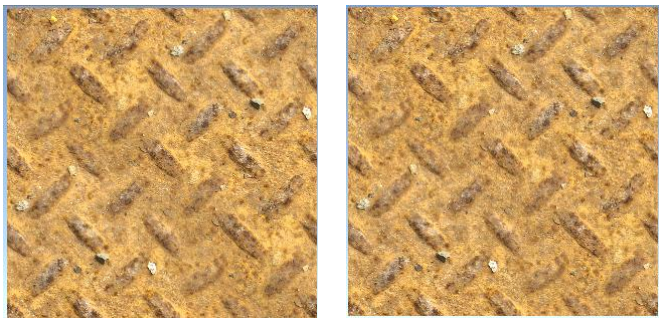
It was decided to use images with resolution of 1024 x 1024 pixels because it is a good balance between file size and amount of information manipulated. Images that exceed this resolution have been resized to maintain the pattern of testing in processing time and quality. As observed during the tests, the processing time of the algorithms is directly proportional to the size of the images. It was found that most of the texture images in repositories are of lower resolution and those with higher resolution are used for high level professional renderings, which was not the purpose of this work and would make testing too difficult. It was performed 841 tests using all the algorithms mentioned in this work.

When using Histogram Equalization (HE), Adaptive Histogram Equalization (AHE) with window values at an approximate 100 or Contrast Limited Histogram Equalization (CLAHE) with windows of size between 15, 17 and 19, good results were obtained in general, preserving the original appearance of the image and showing well the high-level reliefs. HE tends to intensify stains on the image such as shadows or color variations, while AHE softens them. However, depending on the window value for AHE it is possible to create small blocks of color variation that distance themselves from the original image.

In relation to the CLAHE, there is a tendency to darken considerably the images and to preserve fine details, besides being the algorithm with less tolerance in the contrast value to obtain good results. This was the equalization algorithm that the results needed more often to increase the brightness so that the result was close to the original texture. In the flows using the Grayscale Equal Weights (EW) algorithm, good results were obtained in general, maintaining the highest-level variations and fine details in moderation, while the Grayscale Luma preserved fine details better and increased the intensity with to which the color variations became a variation of relief.

The Desaturation algorithm captures great differences in level even in low contrast images, but almost completely eliminates the fine details of the image. In Figure 3 it is visible on the left side, which uses HE, that the diagonal has a dark shade much stronger than in the right image, which uses AHE. The Desaturation algorithm did not achieve a satisfactory result in some cases, because it completely eliminated the intermediate variations of the image leaving a large gray area. As a result the plane remains flat and the rendering does not have the reflection effects.

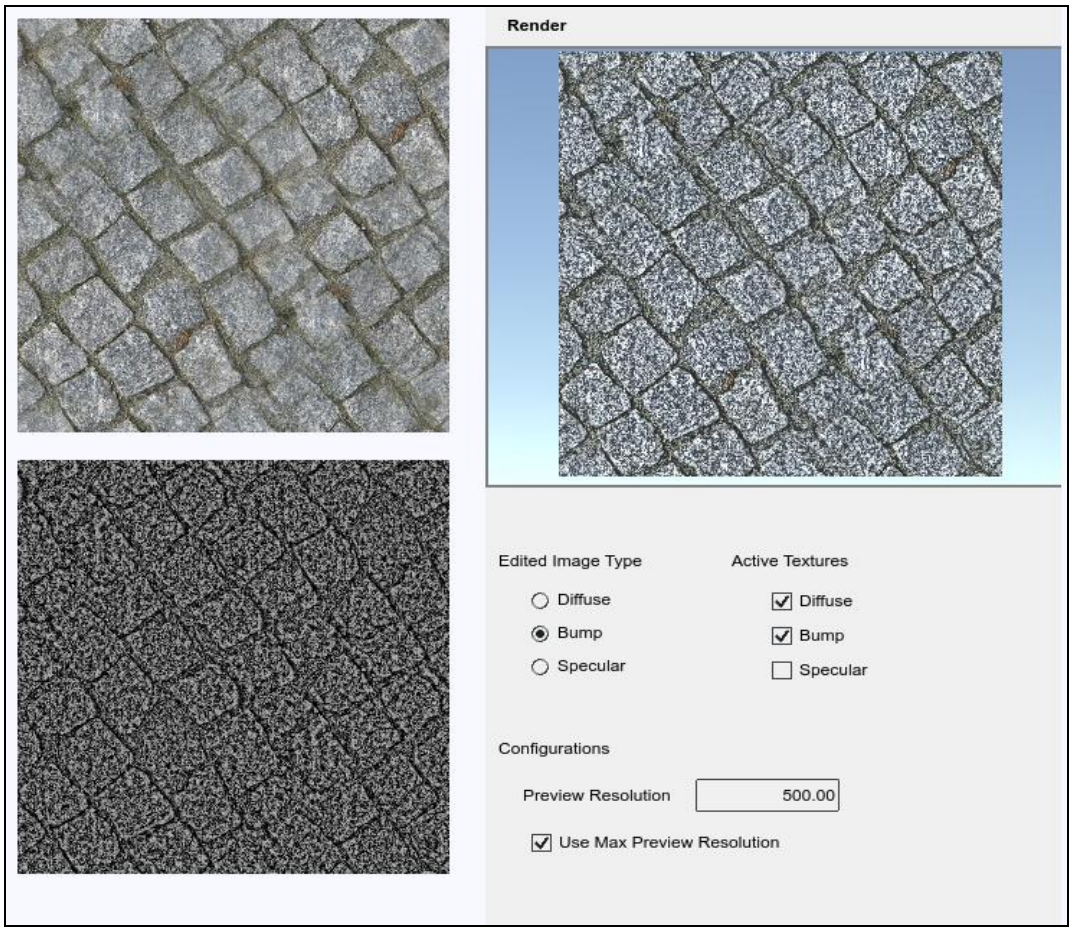
Figure 3: HE Image x AHE Image



Source: (Authors)

The CLAHE was not able to produce any satisfactory results for some images with strong colors (Figure 4), because when correcting the variation of contrast with a small window, the image becomes very pixelated, and when using a very large window the image turns completely black, invalidating the use of the image.

Figure 4: Image defect using CLAHE



Source: (Authors)

The AHE algorithm can bring more refined results compared to HE, but this depends on the size of the window to fit the size of the cracks in the image. Considering the distance between one high relief part and the other in an image, it is expected that AHE can focus more assertively by grouping in the same window a larger number of high or low relief pixels. When the opposite occurs, where the window takes part of the image with a homogeneous amount of relief, an artefact is created in which the application of the window is remarkable. Desaturation obtains very good results when the extension of the lower parts is small, such as on a wooden floor, for example, and, together with CLAHE, in cases where the color variations of the high parts do not well represent the relief variations.

When one piece of wood has on half of it a dark color, and the other half a light color, not necessarily both have different heights. The numbered flows in the following tables represent the combination of algorithms, followed by an optional contrast and brightness adjustment, as shown in table 1. It is slightly easier to get Bump Map and Specular Map quality from an irregular image than from a regular image. Table 2 shows the evaluation by applying the Bump Map. It can be seen that the percentage of images classified as "Very Good" are larger when the images are irregular. In irregular images, the percentage of images classified as "Very Good" are, in most streams, the same as images classified as "Good". This is because small distortions in editing are more difficult to perceive when one has a recognizable pattern

and because in irregular images there is a tendency for unevenness to occupy a smaller area of the image, as shown in Tables 3 and 4.

Table 1: Flows by Algorithms

Flow	Algorithm	Flow	Algorithm	Flow	Algorithm
1	HE + EW	4	AHE + EW	7	CLAHE + EW
2	HE + Luma	5	AHE + LUMA	8	CLAHE + Luma
3	HE + Dessaturation	6	AHE + Dessaturation	9	CLAHE + Dessaturation

Source: (Authors)

Table 2: Bump Map Evaluation (21 images evaluated by flow)

Bump Map (Regular)					Bump Map (Irregular)				
Flow	Very Good		Good		Flow	Very Good		Good	
	Nº of images	(%)	Nº of images	(%)		Nº of images	(%)	Nº of images	(%)
1	15	71%	10	48%	1	13	62%	13	62%
2	16	76%	7	33%	2	14	67%	14	67%
3	14	67%	9	43%	3	15	71%	15	71%
4	13	62%	11	52%	4	17	81%	17	81%
5	12	57%	9	43%	5	17	81%	17	81%
6	11	52%	11	52%	6	16	76%	16	76%
7	10	48%	5	24%	7	14	67%	4	19%
8	11	52%	3	14%	8	11	52%	6	29%
9	7	33%	7	33%	9	10	48%	6	29%

Source: (Authors)

Table 3: Bump Map - Regular Images - Windows Standard Deviation (21 images evaluated by flow)

Bump Map (Regular)								
Flow	Very Good				Good			
	Results	Min. Window	Max. Window	Std. Deviation	Results	Min. Window	Max. Window	Std. Deviation
4	9	15	101	5,42	4	23	101	4,72
5	10	33	101	4,86	3	59	85	3,00
6	8	35	101	4,24	5	15	101	4,07
7	3	19	41	2,94	3	25	31	1,63
8	4	21	41	2,45	3	19	31	1,91
9	3	15	41	2,52	5	15	41	2,61
Total	37	15	101	-	23	15	101	-

Source: (Authors)

Table 4: Bump Map - Irregular Images - Windows Standard Deviation (21 images evaluated by flow)

Bump Map (Irregular)								
Flow	Very Good				Good			
	Results	Min. Window	Max. Window	Std. Deviation	Results	Min. Window	Max. Window	Std. Deviation
4	17	13	101	4,61	6	15	101	5,35
5	17	15	101	5,78	3	61	101	4,40
6	16	25	101	5,07	2	15	23	2,00
7	14	7	25	1,91	4	19	21	1,73
8	11	13	25	1,76	6	9	21	1,91
9	10	13	25	1,76	6	9	21	1,91
Total	85	7	101	-	27	9	101	-

Source: (Authors)

As shown in table 2, for Bump Map Regular the best result was stream 2, with 76% of the images with "Very Good" result, and the worst flow being 9, with only 33% "Very Good" results. The HE flows generated the highest number of "Very Good" results.

As for Bump Map irregular flows 4 and 5 had the best results for both "Very Good" and "Good" quality, reaching 81% accuracy in all these cases. Flows from 1 to 3 use HE and it does not have a window, so they are not presented in tables 3, 4, 6 and 7. The table for minimum and maximum window values is considered among the 21 images tested for that flow and quality of result, which were the highest minimum and lower maximum, respectively, for the parameter window in the application of the algorithms AHE and CLAHE.

In table 3, for example, 9 tests were performed with flow 4, and that, if the algorithms of that flow are selected in the program, the user will get results classified as "Very Good" with the window varying between 15 and 101, in 71% of the 21 images tested.

The Bump Map Irregular window size data are presented in Table 4, it shows that flows from 4 to 6, which use AHE, have a greater tolerance for window size variation to obtain good results, while the flows from 7 to 9 with CLAHE exhibit much less variation.

Table 5 shows the values obtained from the Specular analysis. Once again, when the images are regular, the percentage of the images classified as "Very Good" is superior to those classified as "Good". When the images are irregular, the percentage of images classified as "Very Good" is much higher than those classified as "Good", different from the same situation when the analysis is done using Bump Map. Flow 3 obtained 86% of the results classified as "Very Good" to speculate with regular images.

Table 5: Specular evaluation (21 images evaluated by flow)

Specular – Regular					Specular Irregular				
Flow	Very Good		Good		Flow	Very Good		Good	
	Nº of images	(%)	Nº of images	(%)		Nº of images	(%)	Nº of images	(%)
1	16	76	6	29	1	18	86	3	14
2	14	67	7	33	2	16	76	5	24
3	18	86	2	10	3	15	71	4	19
4	11	52	9	43	4	18	86	3	14
5	13	62	7	33	5	16	76	4	19
6	11	52	7	33	6	14	67	3	14
7	13	62	5	24	7	14	67	4	19
8	14	67	4	19	8	14	67	3	14
9	12	57	6	29	9	12	57	2	10

Source: (Authors)

The fluxes with HE presented the best results for this type of texture. In the case of the Irregular, flows 1 and 4 reached 86% accuracy for "Very Good" quality, this indicates that the Grayscale Equal Weights algorithm presented in both streams may have a greater affinity with this type of image.

The window data for Specular with regular images are in table 6, it is notable that the flows with CLAHE are less tolerant to variation, working only with values 17 and 19, since the size of the window is always odd. Although the minimum and maximum window values are the same for "Very Good" results, and close to "Good", there are variations in the standard

deviation as the result of the intermediate window values used in the tests are different. Figure 7 shows the behavior of the time consumed in relation to the method used (Specular/Bump). It can be observed that when the texture considered is Specular, the lowest and highest recorded time for HE, Grayscale, Contrast and Brightness algorithms are 71 and 392 ms, 26 and 316 ms, 64 and 192 ms and 0 to 113 ms, respectively. With Bump, the minimum and maximum time were 19 and 784 ms, 6 and 395 ms, 0 and 206 ms, 0 ms and 111 ms, in the same order.

Table 6: Specular - Regular Images - Windows Standard Deviation

Specular (Regular)								
Flow	Very Good				Good			
	Results	Min. Window	Max. Window	Std. Deviation	Results	Min. Window	Max. Window	Std. Deviation.
4	6	35	101	4,42	5	29	101	4,60
5	7	35	101	4,63	4	53	85	3,16
6	5	35	101	4,12	4	59	101	3,61
7	6	17	19	1,47	4	17	19	1,73
8	6	17	19	1,47	4	17	19	1,73
9	5	17	19	1,48	5	17	19	1,55
Total	35	17	101	-	26	17	101	-

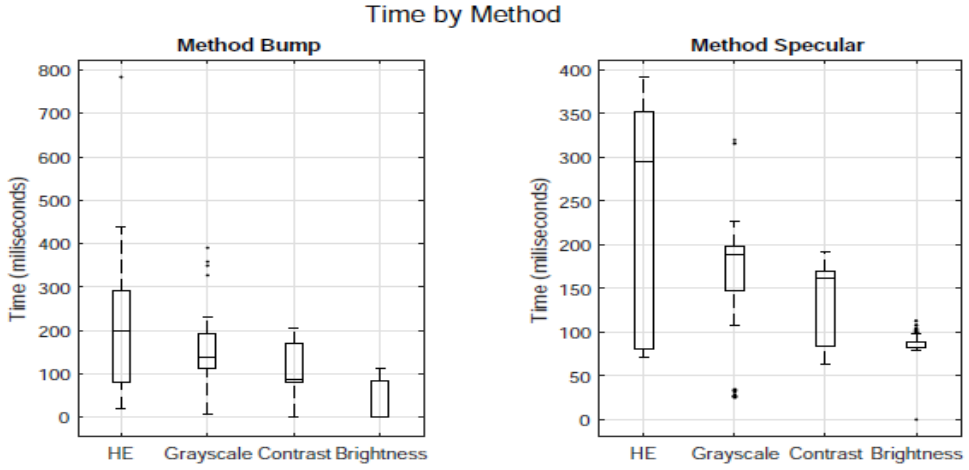
Source: (Authors)

Table 7: Specular - Irregular Images - Windows Standard Deviation

Specular (Irregular)								
Flow	Very Good				Good			
	Results	Min. Window	Max. Window	Std. Deviation	Results	Min. Window	Max. Window	Std. Deviation
4	6	35	101	5,42	3	21	89	4,83
5	7	35	101	5,39	4	21	89	4,90
6	5	35	101	5,51	3	21	89	4,83
7	6	17	19	1,54	4	17	19	1,50
8	6	17	19	1,54	3	17	19	1,53
9	5	17	19	1,73	2	17	19	1,41
Total	35	17	101	-	19	17	89	-

Source: (Authors)

Figure 7: Box Plot - Time by Methods

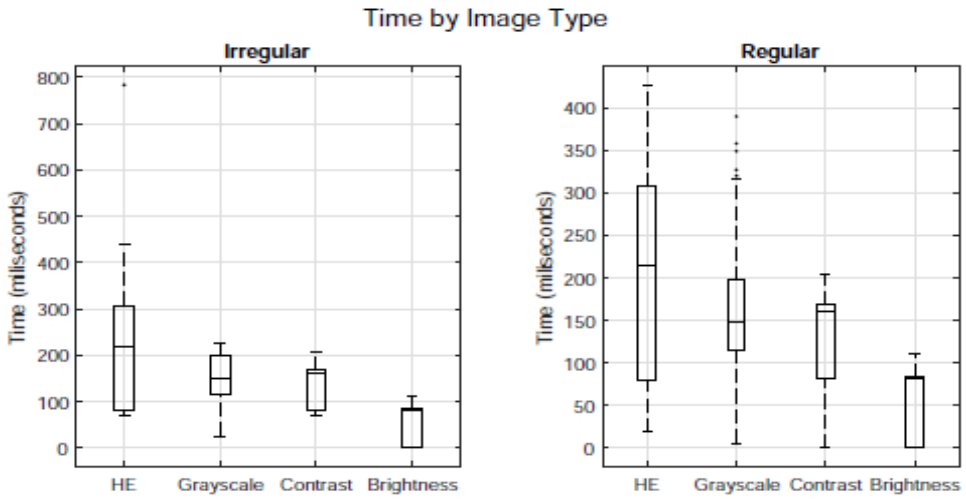


Source: (Authors)

Figure 8 shows the data of the types of images (regular/irregular) versus the times of each algorithm. The time of grayscale algorithms have an approximately equal concentration in both types of images. When the image is regular, the concentration is in 116 and 198 ms and when the image is irregular, the concentration is between 117 and 198 ms. The other algorithms also have fairly similar time concentrations. When they are regular images, the HE time is between 80 and 214 ms.

The Contrast time is between 81 and 169 ms, Brightness between 0 and 82 ms. With irregular images, the HE and Contrast times had a concentration of 80 to 308 ms and 82 and 160 ms, respectively, while the Brightness time has a concentration between 0 and 83 ms. In summary, considering the total image evaluations, there are 620 reviews "Very Good" and 221 reviews "Good". As the tests show, some streams obtained expressive results according to the image type (regular/irregular, Bump/Specular). Thus, it is possible to notice that the use of these algorithms for previously classified images can accelerate the generation of these images in a development environment.

Figure 8: Box Plot - Images Types x Time



Source: (Authors)

For Bump Map, the HE algorithm with Luma obtained 76% accuracy with regular images and in the case of irregular images the best algorithms are AHE with EW or Luma, with 81% of very good results. As for Specular Map, for regular HE images, 86% obtained very good results with Desaturation, and for the irregular images the HE algorithms with EW and AHE with EW reached 86% of hits. Some algorithms achieve exceptional results in specific cases, and in a generic evaluation of the image they would probably be discarded, since the recommended algorithms are those that have assertiveness in a larger range of images and other classifications of the images that could fit in these exceptional cases were not identified in this study.

4. Conclusion

The objective of this paper was to perform an analysis of the image processing algorithms in groups of pre-classified images, in order to find out if these groups are relevant in the

generation of auxiliary textures like Bump Map and Specular Map. It was also analysed if there are more efficient algorithms among the ones mentioned in this paper for realistic rendering. This objective was concluded by means of the classifications of the images that are important for the generation of the auxiliary images, according the performed the tests.

Some algorithms have a considerably higher success rates when compared to others. It is still not possible to fully automate this process due to the high complexity of the images and transformations that occur during processing, but when indicating the most efficient algorithms for that image it is already possible to reduce the time invested in this task notably, especially if the images with larger resolution in which the processing time is exponentially greater.

This indication does not replace the visual analysis of the results, as well as the final adjustments to obtain the maximum quality and reduce the number of artefacts that these algorithms can generate and that are not identifiable without human evaluation. As a hint of improvement, it is advisable to do a multithread and/or parallel processing implementation to speed up the processing of images at higher resolutions.

As well as the implementation of new algorithms and the combination of these with those already used. Another analysis that can be done is the application of HE and Grayscale algorithms in lower resolution images to verify if it has the same parameters when applied in higher resolution images, so the processing for visual analysis could always be done in images with small resolutions in real time, and only the slower final processing in the original image using a higher resolution.

References

[1] Akenine-Möller, T., Haines, E., and Hoffman, N. (2019). *Real-time rendering.* Crc Press.

[2] Aristidou, A., Lasenby, J., Chrysanthou, Y., and Shamir, A. (2018) “Inverse kinematics techniques in computer graphics: A survey” *In Computer Graphics Forum*, V. 37, N. 6, pp. 35-58.

[3] Takahashi, D. (2013) “How pixar made monsters university, its latest technological marvel”, Available in: www.venturebeat.com/2013/04/24/the-making-of-pixars-latest-technological-marvel-monsters-university/

[4] Seeram, E. (2019). Digital image processing concepts. In Digital Radiography (pp. 21-39). Springer, Singapore.

[5] Nixon, M. and Aguado, A. (2019). Feature extraction and image processing for computer vision. Academic press.

[6] Blinn, J. F. (1978), “Simulation of wrinkled surfaces,” in ACM SIGGRAPH computer graphics, vol. 12, pp. 286–292, ACM.

[7] Woo, S. M., Lee, S. H., Yoo, J. S., and Kim, J. O. (2017) “Improving color constancy in an ambient light environment using the phong reflection model”, *IEEE Transactions on Image Processing*, 27(4), 1862-1877.

[8] Transparency Market Research (2015), "Digital imaging market - global industry analysis, size, share,growth, trends and forecast 2016 - 2024", Available in: www.transparencymarketresearch.com

[9] Lukac, R., and Plataniotis, K. N. (Eds.). (2018) *Color image processing: methods and applications*, CRC press.

[10] Szeliski, R. (2010), "Computer vision: algorithms and applications". Springer Science & Business Media.

[11] Tan, R. T. and Ikeuchi, K. (2008) "Separating reflection components of textured surfaces using a single image," *in Digitally Archiving Cultural Objects*, pp. 353–384, Springer.

[12] Huang, S.C., Cheng, F.C. and Chiu, Y.S. (2013) "Efficient contrast enhancement using adaptive gamma correction with weighting distribution," *IEEE Transactions on Image Processing*, vol. 22, no. 3, pp. 1032–1041, 2013.