Retrogressive "Classic Game" Image Processing

Daniel Page
Department of Computer Science
University of Manitoba
umpage7@cs.umanitoba.ca

Abstract

Classic games have captured the imagination of the previous generation, and has come to appear as an artistic style found especially in game development or art. In this paper, we present software which allows one to load pictures into it and process and enhance the image before mapping to a colour palette restriction of an older game system. This software focuses on the Nintendo Entertainment System palette, but we also included palettes for Nintendo Gameboy, Sega Master System, Super Nintendo Entertainment System, and TurboGrafx–16. We will discuss the features and enhancements of the software, and discuss the methods we treat certain conditions of images using the features we provided for image processing. Upon so, we will discuss the benefits or flaws of each feature to specific palettes, or types of images.

1. Introduction

The problem of emulating colour with more primitive colour palettes called Colour Quantization [6] has been studied for a very long time, and we wish to give an application which makes use of it for artistic purposes. Classic games have captured the hearts of niche groups since their arrival in early game consoles. Today, we still see some projects replicate these styles yet more advanced colour hardware support is available. For example video games such as *Mega Man 9* and *Mega Man 10* emulate their earliest counterparts on the *Nintendo Entertainment System* graphically, yet they were released in 2008, and 2010 respectively. The idea of reproducing colour sets which are able to be translated to from complete RGB images is of interest to hobby game developers, or niche groups who fashion the visual aspects for aesthetic purposes.

We will present software prepared which can perform the task of taking RGB images and transforming these images based on colour palettes of a variety of older game consoles. The consoles we have given attention to are (primarily) *Nintendo Entertainment System* (NES), *Nintendo Gameboy* (GB), *Sega Master System* (SMS), *Super Nintendo Entertainment System* (SNES), and *TurboGrafx–16* (TG16). We will also address the issues with directly mapping to any one of these palettes, and how our software provides enhancements to resolve these issues for many cases.

In this paper we will begin by describing the *Retrogressive "Classic Game" Image Processing* (RCGIP) software, then we will give our observations for each colour palette and how to treat some image cases. Next, let us describe the RCGIP software.

2. RCGIP Software

The *Retrogressive "Classic Game" Image Processing* (RCGIP) software lets the user load images, provide enhancements for better mapping, and maps images to a particular colour map with the option to save outputted images. Figure 1 provides a screenshot of the overall graphical user interface with a sample image loaded and processed with default settings. In this section we will begin by giving the general features for images, then we will discuss the palettes and the capabilities of each. Lastly, we will give each transformation enchancement and how each works whether a choice or optional. As we describe components, we will use the labels found in Figure 1.

2.1. General Features

The software includes a variety of features. To begin, the user loads an image by selecting an option and choosing such in a file open dialog. This can be found through the pull–down menu at Figure 1.A, or the button labelled at Figure 1.L. When an image is loaded, it will appear in the axes at Figure 1.B, and any retrogressed image output will appear beside it on the right in the axes at Figure 1.C. If the user would like to zoom in or out either the input, or output they can click the "Expand" button for each respectively at Figure 1.D, or Figure 1.E. The user can select a palette in the Colour Palette panel located at FIgure 1.F, where only one palette can be selected at a time. In the panel Retrogressive Transform at Figure 1.P, the user can select one transform (Figure 1.Q – Figure 1.S), and up to four different optional

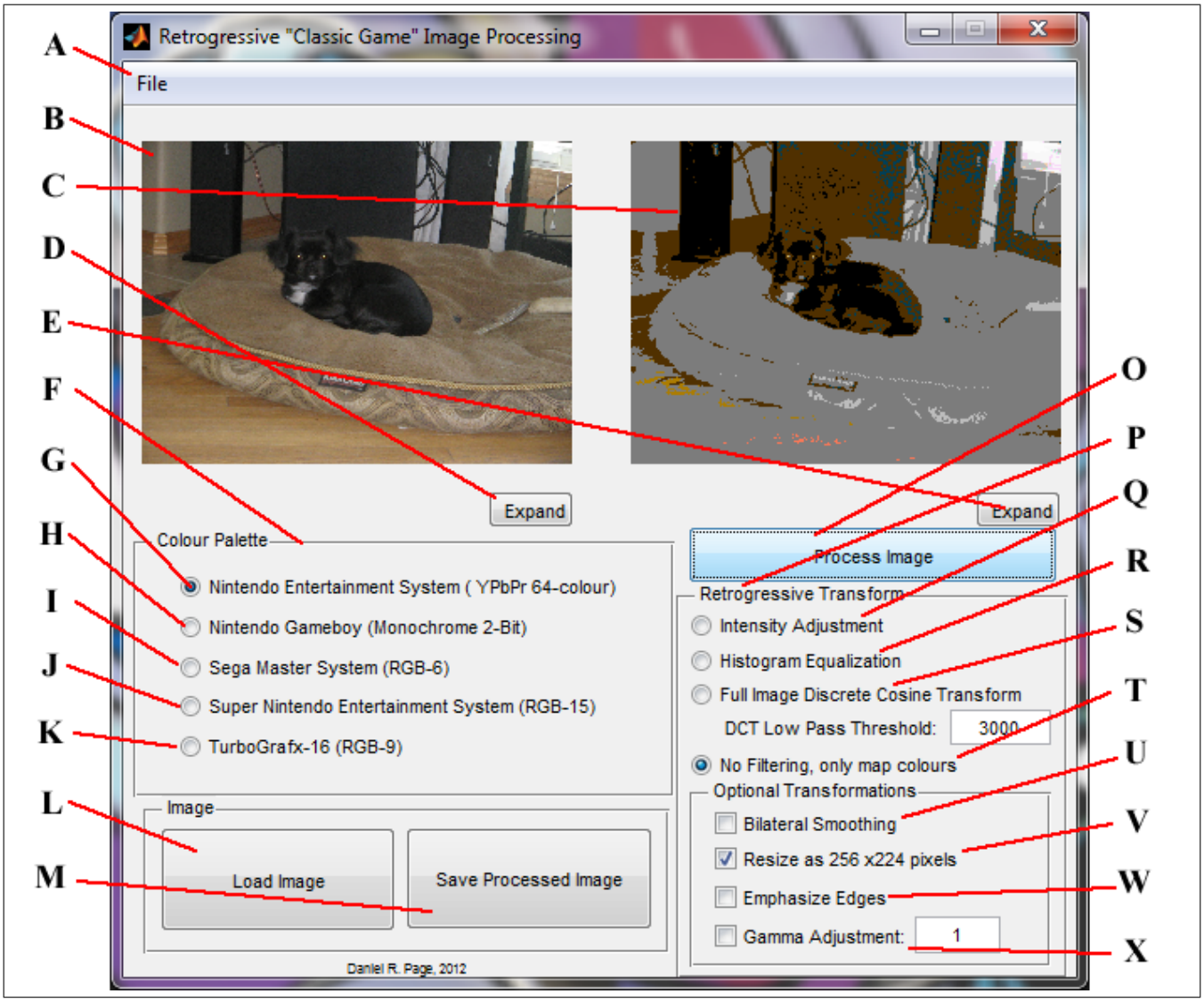

Figure 1. Labelled screenshot of Retrogressive Classic Game Image Processing graphical user interface window

transformations (Figure 1.U – Figure 1.X). Once the user is satisfied with their options, the user can click "Process Image" which is located at Figure 1.O. The output will appear in the right axes at Figure 1.C. The user can modify settings and press process image as many times they would like until the sought result is found. Once satisfied, the user can save their image by clicking the button at Figure 1.M, or finding the same option in the pull–down menu at Figure 1.A.

2.2. Palettes

The RCGIP software supports five different classic game console colour palettes. Our goal in our software is not to mimic the limitations of screen drawing for each system, but to effectively map to their colour systems. Since our main focus is the NES colour palette, we have supplied transformations best suited to that particular aesthetic look which will be described in the next section. In each palette described, we give their specifications, and what we took into consideration for each palette.

2.2.1 Nintendo Entertainment System (NES)

The NES has a YPbPr 64–colour palette where only fifty four [8] to fifty six [3] of the colours are useful. The remainder of the colours are all black. For the effectiveness of our program, we have removed all the redundant colours from our colour map and kept only fifty five from the original palette [3]. In our software we are not considering local bit limitations. A default option offered for all palettes which follows from the capabilities of the NES system is that the

screen size is 256 by 224 pixels. This option can be found under Optional Retrogressive Transformations.

2.2.2 Nintendo Gameboy (GB)

Unlike the NES palette, the Nintendo Gameboy has only two–bit monochrome grayscale support [8]. As a consequence, there are only four intensities supported. In our software, we emulated the *green–like* appearance of the screen of the Nintendo Gameboy.

2.2.3 Sega Master System (SMS)

Consisting of sixty–four colours, the Sega Master System colour palette is a 6–bit RGB system where each colour component is encoded by two bits [4, 7]. Each bit change spreads intensity values between 0 and 255 in a roughly uniform manner for each colour component. Our implementation does not consider local bit limitations for drawing on the screen.

2.2.4 Super Nintendo System (SNES)

The SNES colour palette is a fifteen bit RGB colour palette [7, 8]. Each colour component consists of five bits of encoding. RGB values are distributed uniformally for each band as each bit change. In our palette we are not considering the special modes of the SNES graphics chip. This palette has exactly 32,768 colours.

2.2.5 TurboGrafx–16 (TG16)

Finally, the TurboGrafx–16 colour palette consists of 512 colours. This palette is a 9–bit RGB system where each colour channel is encoded using three bits [7, 8]. As with the SNES, RGB values are uniformly spread, but we are not considering the drawing limitations of the system on screen itself.

2.3. Retrogressive Transforms

In this subsection, we will describe each of our methods of enhancement to assist the user acheive a good quality retrogressive image. The software offers the option to not include any of the general retrogressive transforms at Figure 1.T (and the user may also use the optional transformations with this choice). All transformations occur before mapping to any particular colour palette.

2.3.1 General Retrogressive Transforms

Next, we will describe each general retrogressive transform. At most, only one of these retrogressive transforms can be used at a time. The general retrogressive transforms are meant to assist the mapping process to enhance the entire image depending on the palette. Retrogressive transforms can help aid most cases while others may not require any transforms at all.

2.3.1.1 Intensity Adjustment The following transform is MATLAB's *imadjust* function. The following function adjusts the intensity levels of a grayscale image. In our implementation we apply this function to each colour band. This is meant to aid images with poorly lit conditions.

2.3.1.2 Histogram Equalization This transformation is MATLAB's *histeq* function. The transform applies histogram equalization in the RGB–space to each colour band. In experiments we found that this filtering aids the mapping of images with overall poor contrast.

2.3.1.3 Full Image Discrete Cosine Transform This transform employs MATLAB's *dct2* and *idct2* functions for two dimensional Discrete Cosine Transform. Once DCT is applied to each colour band, the transform employs a low–pass filter. The threshold is given by the user in the text field to the right of its item in the panel at Figure 1.S. This transform is for stylistic impression. Some retro artists make use of multiple kinds of colours to emphasize shades where normally only a shorter range may only provide little shading. This transform may result for higher range colour palettes in rainbow effects. For palettes like the GB palette, this can serve useful for lighting effects. We wish to simulate this effect. If the filter parameter is low, there will be less structure, and a high parameter value will give a high contrast image in our software's use of DCT.

2.3.2 Optional Retrogressive Transformations

Each of the following transformations are optional. They can be applied with some restrictions to each of the main retrogressive transforms, or no transforms at all.

2.3.2.1 Bilateral Smoothing Based on Tomasi and Manduchi's paper on Bilateral Filtering [5], we used Lanman's MATLAB implementation [2] of Bilateral filtering. For our smoothing operation, we used $W = 1$, and $\sigma = 1$. This provides a smoothing operator without losing edge content for images. This transformation provides a stylistic effect which can emulate the effects classic game images have for shading in some cases. This effect will have no effects on the Full Image Discrete Cosine Transform. This can take time to process if the optional resizing to 256 by 224 pixels transformation is disabled.

2.3.2.2 Resize as 256×224 pixels By default, this transform is set due to performance, and attaining stylistic downsampling. This effect is geared towards the NES palette

where the screen resolution is of size 256 by 224 pixels. We highly recommended keeping this effect enabled for good quality retrogressed images. When images are smaller than both dimensions, this operation is redundant.

2.3.2.3 Emphasize Edges When enabled this transformation emphasizes edges. Classic games will often have highly emphasized edges. Using Sobel edge filtering, this edge detection transformation will subtract the edge map from the processed image. For the Full Image Discrete Cosine Transform, this transform will construct the edge map from the original image instead.

2.3.2.4 Gamma Adjustment For hard to treat images, gamma adjustment can help map images with poor segmentation, or poor lighting. The user can enter their own gamma value beside the selection for this feature at Figure 1.X. By default, the positive floating point value $\gamma = 1$. If gamma is equal to one, then there will be no change in effect. If $\gamma > 1$, then darker portions of the image such as shadows will become darker. If $\gamma < 1$ darker portions of the image will become brighter. Gamma Adjustment will have no effects on images where Full Image Discrete Cosine Transform is also applied.

2.4. RCGIP Algorithm

This algorithm has eight steps. These steps are applied to input image I:

1. (IF ENABLED) APPLY: *Resize as* 256×224 *pixels* ,
2. APPLY SELECTED: Retrogressive Transform,
3. (IF ENABLED) APPLY: *Gamma Adjustment*,
4. (IF ENABLED) APPLY: *Bilateral Smoothing*,
5. (IF ENABLED) APPLY: *Emphasize Edges*.
6. APPLY SELECTED: Convert to indexed image using colour map of palette using MATLAB's *rgb2ind* function. This algorithm is a heuristic algorithm that is based on the Median Cut Algorithm [1].
7. Convert to RGB image,
8. Output image.

Upon completion, the processed image is displayed in the output axes (Figure 1.C).

3. Observations

In our tests, we used seven images. With the seven images (Figure 2.A – Figure 2.G), we will explain how to treat certain types of images and the drawbacks of each palette. All of the test photographs are owned by Daniel Page. Our test images (see Figure 2) each represent specific cases:

A. Image with lots of similar colours,
B. Digital video game image,
C. Dark, low contrast photograph,
D. Bright, high contrast photograph,
E. Bright, low contrast photograph,
F. Dark, high contrast photograph,
G. Digital art with shades.

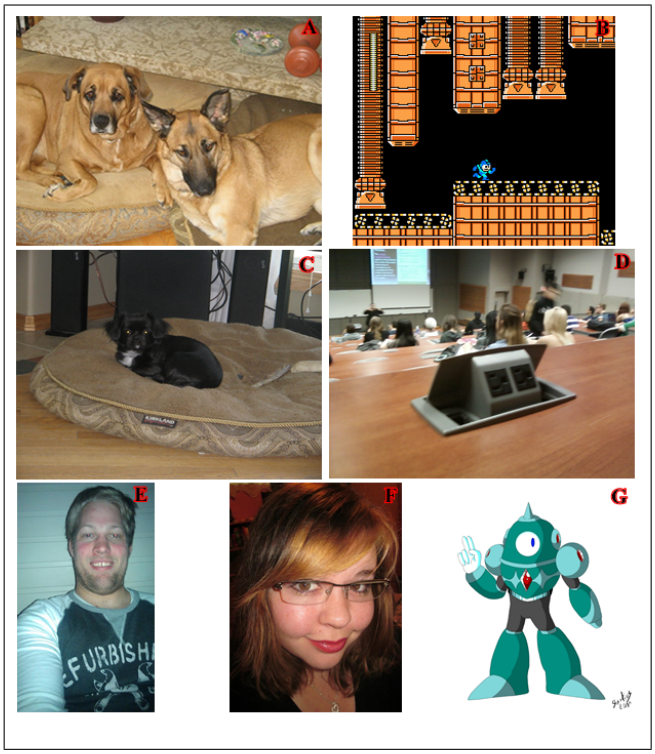

Figure 2. Labelled images used for testing RCGIP software.

We will only be considering three of the colour palettes in our observations. We are not including TurboGrafx–16, and Super Nintendo Entertainment System in our results as these two colour palettes are high enough quality. We analyzed the NES, GB, and SMS colour palette results. Let us give our observations.

3.1. Nintendo Entertainment System (NES)

When applied to different types of images, we obtained quite different results. As seen in Figure 3, we were successful at creating images which appear to capture the

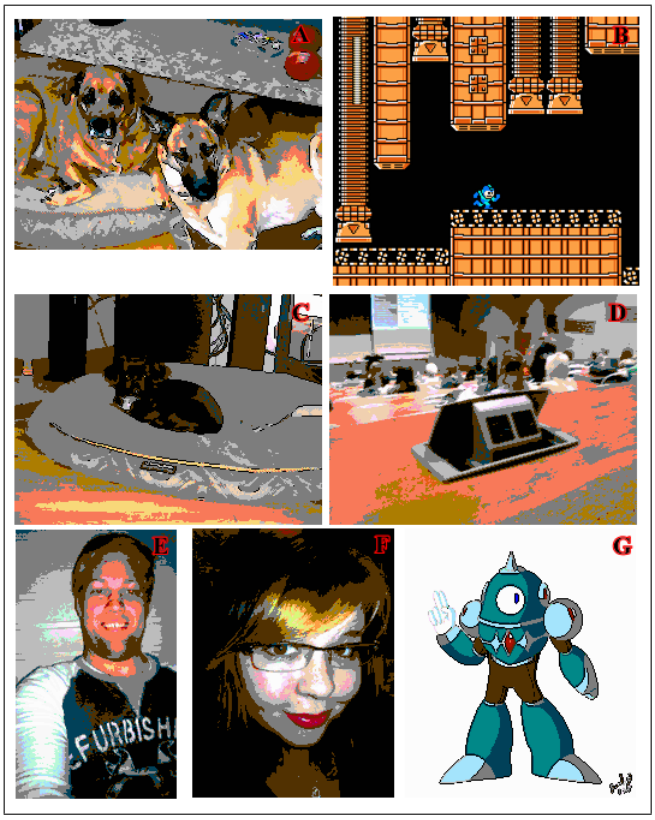

Figure 3. Images produced from test images with NES palette.

colours of the NES. With darker images, it is very common to obtain very dark yellow tones. This is due to the next darkest tone in the palette from dark gray being this tone of yellow. This can be very unappealling unless properly treated. The NES palette is less distributed than other palettes due to the large number of duplicate colours. Since colours in terms of the RGB space are not uniform, experimentation is quite common to gather satisfactory results.

Figure 4. Figure 2.A processed with only mapping to NES palette.

For example, Figure 4 shows Figure 2.A without any transformations mapped to the NES palette. We see some dark yellow, but a lot of the common colours do not reflect boundaries very well. We found experimentally that intensity adjustment to increase the contrast, and using $\gamma = 1.2$ to segment the image by making the image darker improved the quality along with emphasizing the edges to result in Figure 3.A. Our experiments show that brighter images with high contrast (Figure 3.A, Figure 3.D), or images with darker colours (Figure 3.F) but low contrast (Figure 3.C) are improved through image intensity adjustment. If an image is applied to intensity adjustment, in these cases, normally increasing gamma will help improve the results. On the other hand, images with lots of brightness and low contrast we found are resolved through histogram equalization. We predict this is due to the poor segmentation, which can be improved by spreading levels across all colour bands as it may include lots of gray originally. Digital images with primitive colours typically do not require any filtering unless they are darker. For example in Figure 3.G contains the darker tone of yellow but looks very appealing aside from this, especially with edge emphasis. Figure 3.B did not require any filtering at all due to its appearance but only two colours that are very hard to tell have changed between two. This shows its usefulness as a palette verification for images. If no transformations are used, an image with actual NES colours will remain unchanged after processing.

3.2. Nintendo Gameboy (GB)

We found results with the GB palette to be exceptional. Since the GB palette is only four intensities, it is a matter of obtaining the detail wanted through either segmenting brighter, or darker portions of the image. The Retrogressive Transform Full Image DCT serves especially useful with this palette. For darker images (Figure 5.C, Figure 5.F) treating such with Full Image DCT allows for further textures to be shown by representing many different shades with different RGB values. When these various RGB values are mapped to the GB palette these strange colours become better texture, and lighting for the image which can capture the style of GB graphics very well.

An alternative to treating high contrast or bright images (Figure 5.D) is to use intensity adjustment, with varying gamma adjustment. By varying the contrast levels across all colour bands, the function will transform the image to level out the intensities. Once these are levelled, darkening the image is often the solution for more detail. Also, we predict images with little contrast (Figure 5.A, Figure 5.E), or lots of similar colours are resolved with histogram equalization. We believe this is due to a lackthereof in variation of intensities across the bands along with value in each band. Histogram equalization will equalize the levels across each band to for better segmentation in the GB palette mapping phase.

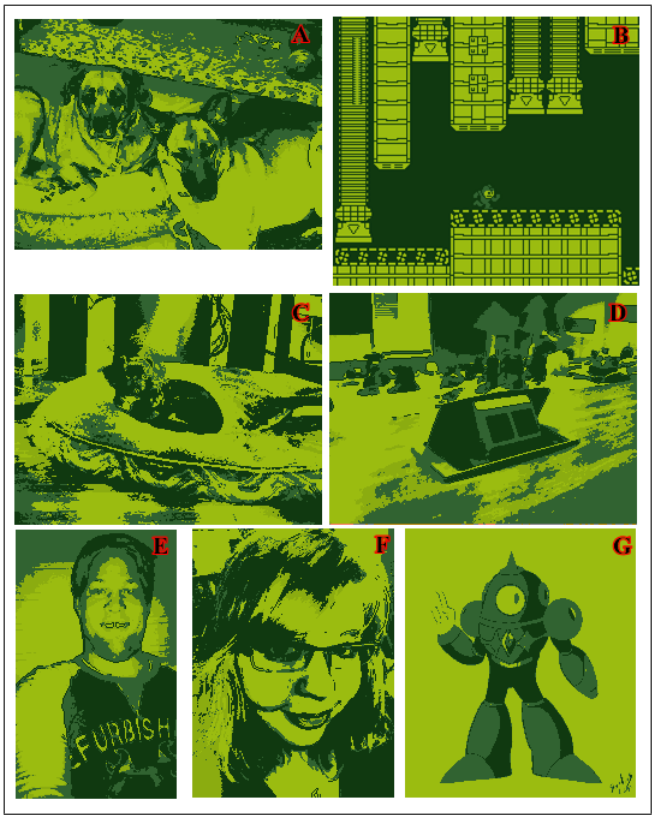

Figure 5. Images produced from test images with GB palette.

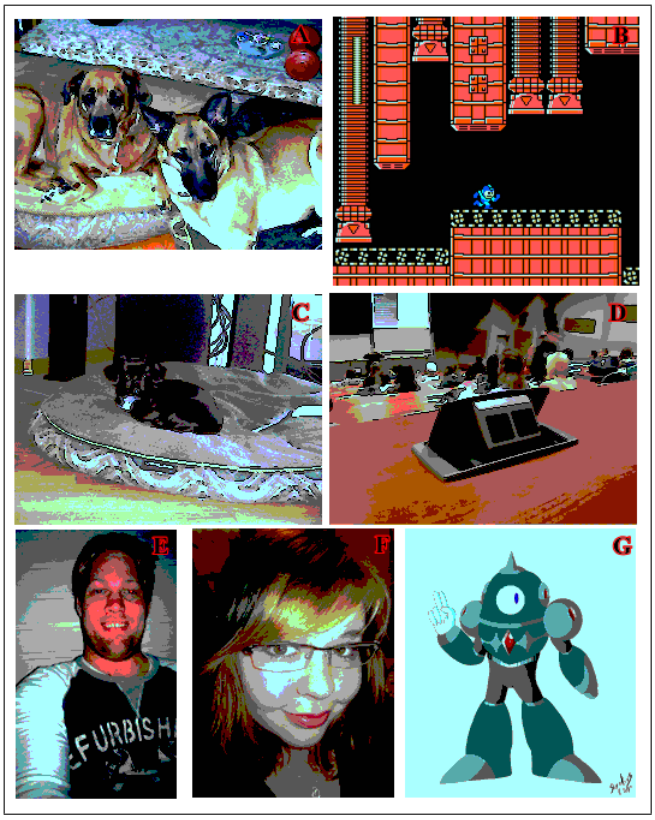

Figure 6. Images produced from test images with SMS palette.

3.3. Sega Master System (SMS)

From our results (see Figure 6), it is evident one flaw comes apparent due to the palette. There is a dilution of very bright blue due to the lackthereof of near–white values in this palette. Close to white values are mapped immediately to the light blue, but in ways this is similar in style to some SMS style graphics, but may not be desirable for all cases. An interesting effect observed due to its close relationship with the NES palette (both being very close in number of colours) is its effect on digital images.

Figure 6.C is effectively mapped, but actually causes less detail to appear but looked just like if it were on the Sega Master System. This is due to the closely related colours being mapped to the same colours while the varying outlines remain the same. This is a desirable effect, and only requires mapping to acheive. To capture SMS like graphics, it has a reverse relationship to the NES processing. We acheive an SMS appearance on photographs when applied to histogram equalization. Since the SMS palette is a very distributed variation of our RGB colour system with less colours, spreading out the levels on each band will map much more effectively than if we only considered the contrast. The only exceptions we found were with images with lots of contrast such as Figure 6.D, and Figure 6.F. These two examples were treated with intensity adjustment to balance the darker portions of the image with the brighter portions on each colour band for effective mapping. For digital art (such as Figure 6.G) mapping did not seem to be very effective but using gamma adjustment assisted in proper mapping.

4. Conclusions

Using a variety of techniques such as histogram equalization, intensity adjustment, discrete cosine transform, bilateral filtering, edge detection, downsampling, indexed images, and gamma adjustment we presented an effective method for users to create retrogressed classic images for both photographs and digital images. We believe such can be useful to those who wish to retrogress their images to suit a classic game appearance. In conclusion, the Retrogressive Classic Game Image Processing software effectively produces images which give the appearance of classic game consoles such as NES, GB, SMS, SNES, and TG16.

References

[1] P. Heckbert. Color image quantization for frame buffer display. *ACM SIGGRAPH '82 Proceedings*, 1982.

[2] D. Lanman. Matlab central – bilateral filtering, 2006. http://www.mathworks.com/matlabcentral/fileexchange/12191-bilateral-filtering/content/Bilateral%20Filtering/bfilter2.m.

[3] NESHacker. Nes palette – nes hacker wiki, 2012. http://www.thealmightyguru.com/Games/Hacking/Wiki/index.php?title=NES_Palette.

[4] SegaRetro. Palette, 2012. http://segaretro.org/Palette#Master_System_Palette.

[5] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. *In Proceedings of IEEE International Conference on Computer Vision*, 1998.

[6] Wikipedia. Color quantization, 2006. http://en.wikipedia.org/wiki/Color_quantization.

[7] Wikipedia. List of monochrome and rgb palettes, 2012. http://en.wikipedia.org/wiki/List_of_monochrome_and_RGB_palettes.

[8] Wikipedia. List of video game console palettes, 2012. http://en.wikipedia.org/wiki/List_of_video_game_console_palettes.