User Directed Parameter Variation Applied To Images

Eileen Bai *Philip Ly*

University of California, Berkeley

Figure 1 An image progression of a landscape using our method.

Abstract:

This paper describes a new approach for user interaction with image manipulation that incorporates a variation of genetic programming. The approach involves an implementation of a new type of interface for generating pictures with variable parameter changes as well as introducing a new take on the creative process for image manipulation. The main features of our interface include various filters to apply to images, side-by-side comparison of multiple images, and a way to choose the *best* image that will *survive* to the next round of manipulation. The idea of "best" images and "survivors" is based on the idea of genetic programming work done by previous researchers where images that the user favors will continue in the progression thereby spawning new generations of what would appear to be even *better* images. Using this interface, it becomes easier to experiment with different outcomes which we then analyze along with user experience.

Keywords: Genetic programming, evolutionary algorithms, comparisons, user interface, photo manipulation.

1 Introduction:

The definition of a "good" image depends on many factors. These factors can range from color, subject matter, mathematical rules of composition, as well as being subjective to the particular user. But perhaps one of the most principal ideas a user uses when deciding whether or not an image is good, is by using the process of comparison. Most people will decide whether or not they like an image by consciously or subconsciously comparing it to other images they have seen before. In this paper, we hone in on this comparison process to try and improve the efficiency as well as have the user generate good images that constantly evolve.

> **Problem:** There is a user who wants to use a photo manipulation tool to alter an image but does not necessarily have an exact idea of what he wants to change in terms of the filter and parameter values he want to apply. In addition, this user is not a photo manipulation expert which means that their options can be pretty limited as the learning curve for good parameter input can be difficult. This process of changing parameter values also takes a lengthy amount of time since it involves multiple iterations of trying new values and choosing the values that produce the most appealing results. The user's goal is to generate the best looking image using his limited knowledge of photo manipulation tools.

This is a problem that we came across based on our own experiences with using image manipulation tools and software. We focused on the idea that users would appreciate a tool to generate an assortment of images all with slight parameter variations in order to choose which one they liked the most and progress from there.

Our idea is based upon the evolutionary algorithm in computer science called "genetic programming."

> *Genetic programming is an automated method for creating a working computer program from a high-level problem statement of a problem. Genetic programming starts from a high-level statement of "what needs to be done" and automatically creates a computer program to solve the problem.*
>
> --Genetic-programming.org

In our case, the "high-level problem statement" is described as how to have a user generate good images efficiently while incorporating a computer algorithm to produce randomness. Random variation is necessary to our approach because in order for images—as well as subjects in nature in general—to evolve, there must be a constant production of feature variation that forces each of the images to compete against each other and therefore enforce the Darwinian concept of "survival of the fittest."

Currently, the leading photo manipulation tools only allow one image to be seen at a time. In this way, it restricts progress and reduces efficiency because a user cannot view many different options in one single window. That being said, it is still possible for a user to compare images, but the process would involve tedious manual manipulations to generate different variations of images and then reposition all the windows in order to see everything in one screen. This is one of the main problems our approach aims to improve upon.

Our approach consists of two parts. The first is the interface that the user uses to load images, apply filters and variations, choose their favorite output, and repeat the process until they reach a final image they are satisfied with. We incorporate the idea of interactive evolution since while the computer generates the randomness, the user is the one that drives the creativity and final direction. The image variations that are produced per round can be considered the current generation while the one picture the user chooses out of all of them can be considered the "fittest" that spawns new children by moving onto the next round. The interface was created using the Matlab user interface creation program as shown in Figure 2.

The second part of our research involves analyzing the different results generated and testing the experience of users with various photo manipulation backgrounds to see if this is a technique worth more investigation.

Further into this paper we will provide a more detailed discussion of our particular algorithm along with an explanation of the filters we chose and how we calculated random parameter variations.

2 Related Work and Background

The general idea of genetic programming is an idea that has its roots much further back in other computer science fields such as artificial intelligence. The overall idea involves an algorithm that takes as input a very large amount of random variations of which only the strongest variants survive to produce more and more children.

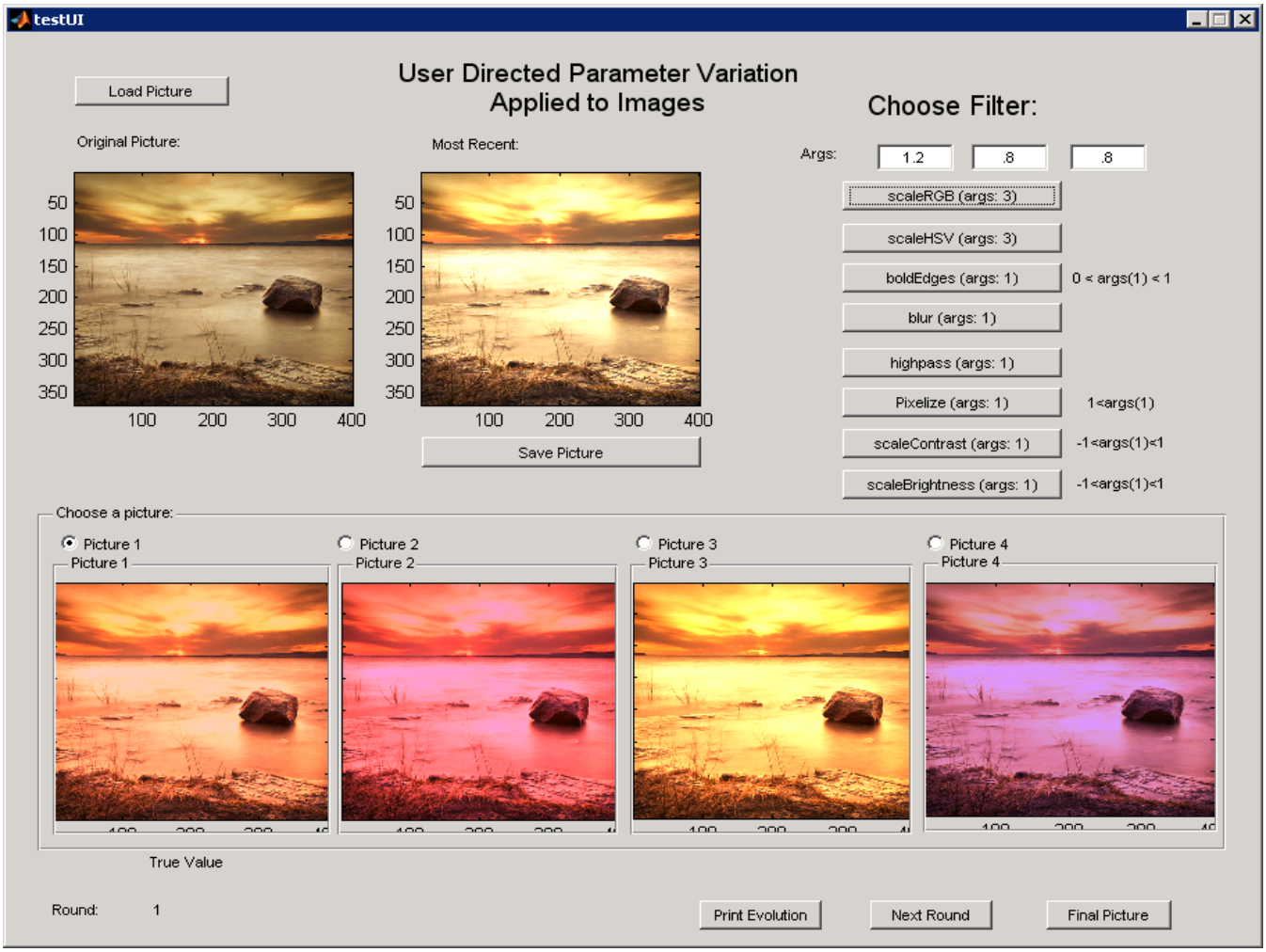


Figure 2 Snapshot of our user interface in action.

Specifically, genetic programming and its applications to image processing has already been explored by previous researchers such as Karl Sims and Ken Musgrave. A general method of finding the best parameter values to apply to images is also an area that has been researched by the Siggraph submission "Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation."

Although our work was heavily inspired by previous genetic programming and parameter variation research, the end result has many different qualities and accomplishes quite a different goal.

Genetic images work by Karl Sims is similar to our work in that it deals with user interactive evolution of images, however the images that he started with are quite different from ours and the algorithm he uses for evolution alters the actual structure of the image instead of just the filters being applied to it as in ours. Sims work was originally unveiled as an art exhibit where images were placed generated on multiple screens and users stood in front of the images they liked the most. The computer would then take the information from which images were the favorites and spawn offspring off of those.

Sims eloquently describes the relationship between human and machine in interactive evolution which strongly motivated our work:

> *This interactive installation is an unusual collaboration between humans and machine: the humans supply decisions of visual aesthetics, and the computer supplies the mathematical ability for generating, mating, and mutating complex textures and patterns. The viewers are not required to understand the technical equations involved. The computer can only experiment at random with no sense of aesthetics -- but the combination of human and machine abilities permits the creation of results that neither of the two could produce alone.*
>
> -Karl Sims

In addition, our work is also not as heavily based on finding good parameter values for image manipulation as the "Design Galleries" paper submitted to Siggraph in '97. We do not focus on what the optimal values for parameter modification are and we do not try and describe any mathematical models or algorithms to learn these values.

4 Application

4.1 Procedure

1. Load up the original image to be manipulated.
2. Pick the filter to apply.
3. The GUI then displays the "truth" image along with a few variations. The user then picks which variation they like the best to save. This is the one they want to move forward with.
4. The user then repeats the process of applying filters until they reach an image they are finally satisfied with.

Throughout the process of filter application, the user can also keep applying the same filters to see more specific variation. The important point to note is that even if the user does not necessarily like the final picture, it is the process they went through and the visual cues they absorbed along the way that gives them a better idea of what direction they want to go in if they choose to try and re-manipulate the image.

4.2 Interface

Our interface was created using Matlab's graphical user interface creator (GUIDE). An example can be seen in Figure 2 of which the main features include:

1. **Load Picture:** loads the original picture that the user wants to modify.
2. **Filter List:** list of filters to apply to the image.
3. **Filter Arguments:** parameters each filter takes as input.
4. **Image Outputs:** four different variations of the filter and parameter combination. The first output on the left is the "truth" which has no variation.
5. **Next Round:** save the selected picture to use it in the next round.
6. **Most Recent Picture:** the most recently selected picture; consequently the one being manipulated.
7. **Final Picture:** displays the final image whenever the user decides he wants to end the process.
8. **Save Picture:** save the "Most Recent Picture."

4.3 Filters

We chose the filters based on a combination of what we thought were the most common changes made to a picture, the difficulty of writing the filter, as well as what we thought would create interesting variations.

A complete list and explanation of our filters is as follows:

Scale RGB: Given 3 arguments for red, green, and blue scaling factors, we can scale each color channel accordingly in the picture. We varied each channel by a fraction after the scaling occurs.

Scale HSV: Given 3 arguments for hue, saturation, and brightness, we can alter the image from RGB space to HSV coloring. We applied variance to the image after it is

converted to HSV space.

Bold Edges: Given a single argument, it is passed to the threshold variable for the edges() command in MATLAB. We then darken the pixels where edges are found in the image and we vary the threshold argument only fractionally.

Blur: Given an argument in the first field, we can use that in the sigma variable in MATLAB's Gaussian filter function. We can then run that low-pass filter on the image to blue the image. We vary that sigma value to obtain different magnitudes of blurring.

Highpass Filter: Similarly to the blurring filter, we take in an argument for the sigma variable in MATLAB's filter function and then subtract the blurred image from the original image to obtain the high-pass filtered image.

Pixelize: Taking in a single argument, we define a sampling block size and gather color information from the pixels that the sampling block covers. We then average those values and apply them to the corresponding pixels in a new image. We vary the input argument to get varying sample block sizes.

Scale Contrast: Passing in one argument, ranging from -1 to 1, we are able to increase or decrease the contrast of the image. We apply a fractional, random factor to the input to obtain varying contrast levels close to the desired input.

Scale Brightness: Input one argument to adjust the brightness of the most recent picture. We simply vary the argument by a range of .4.

4.3 Parameter Variation

To decide on the amount of variance we would apply to each filter, we ran several tests on different images. We found that it was more relevant to have only minimal alterations between variants to have more useful comparisons. Thus we have specified ranges for each filter's arguments to adjust images accordingly.

5 Results

[see pictures]

For research, we had several people test our interface, both familiar with image manipulation and not. We then had them take a survey of what they liked and didn't like about it. Reviewing the answers, we found that people who had more experience preferred it and wouldn't mind seeing it implemented better. They found it helpful and more efficient as more pictures could be seen at once and they appreciated the fact that they had several options to work with.

On the other hand, users with less photo editing experience were relatively confused by the interface. They didn't feel comfortable with the many filters at first and had a harder time learning the functionality.

6 Limitations

A big factor in this project was utilizing better computing power from the original project, Genetic Images by Karl Sims, from 1993 to generate several variations of images per generation. Not requiring what 1993 called supercomputers, we are able to apply filters to several images, but processing power and RAM still prove to be an issue. Running on smaller-sized images works rather instantaneously. However, once applying filters on images of 14 megapixels, the runtime becomes unreasonable. Especially for a machine without enough RAM. 4 image variations at 14 megapixels each quickly took up 5-6GB of memory.

Another limitation we felt we had was screen real estate. To compare several photos next to each other, displaying them at a lower resolution doesn't allow the user to check for all the minute details on every variation. This also does not allow the user to completely determine whether a specific variant is better than the other. With more screen space, larger previews of images would be possible as well as more than just four variant selections.

Developing filters also proved to be quite a limitation because with a limited selection of tools to apply to images, we could only get so many variations. Implementing Photoshop API could've helped here.

7 Future Work

Our prototype interface opens up possibilities in several areas of image editing. A lot of improvements can be implemented into program as well. Taking plenty of feedback from our tested users, we realize how much the design can play a role in how appealing the interface can be. The users we tested seemed to be interested in the many options they could see at the same time and compare to get the best result.

On the current interface itself, there are several improvements we could do to provide a better overall experience for the user. The most obvious would be to generate more variations with one click to allow the user to see more images and compare them. Ideally, we can optimize the processing and memory usage by distributing each variant job to a machine on the cloud. That way we aren't limited by CPU and RAM. Not only would applying filters be parallelized, but the user wouldn't have almost infinite memory to generate as many variants as they desired. With more images to compare with, the selection process becomes even more efficient. We know we won't be showing the full resolution of each image. To increase the performance of this, we can down-sample the images of the variant previews, which will decrease runtime and save memory space.

If we develop more filters to work with the interface, we should be able to gather more variations. More of these functions will allow the images to grow and evolve into possibly, more interesting images. It would open up the computer and the user to more creativity. Within these filter functions, we could also implement a feature to pass in default arguments without user input. This would enforce our purpose of reducing precise argument knowledge even more and make image manipulation seem more user friendly. Of course we should also add the actual arguments

that are used for each variant to encourage the user to learn what arguments cause what effect.

Another feature we could have this implemented on is giving several variations on local manipulations. Applying to content-fill aware, which uses the Patch Matching algorithm and doesn't give the same result every time, we can generate several different results of patch matching all at once to compare among each other. This could even work for selection tools where the area of selection slightly varies. Much like what we have already, the user could just select a generated result and continue from there.

Another interesting feature to add would be real-time variation. To have variance applied to real-time tools such a the brush or eraser. In this case, we would vary the path of the stroke a little. With several different strokes generated, the user can continue on from a selected option.

By applying a machine learning algorithm, we can train the program to apply arguments other users have found aesthetically pleasing depending on the type of photo used as the input.

More feedback on the advantages and disadvantages of the process would be helpful as well which can be easily gathered through more research and surveying.

8 Conclusion

Our method can be effective if executed correctly. As seen in the amount of future work there is, this project can be very expansive and applied to various techniques. However, the central idea remains. Users Enjoy the fact that they can quickly compare photos next to each other and generate more images closer to the desired result. It seems to apply mostly to people who already have photo manipulation experience. Although, the many, generated results can inspire many new users to start using image manipulation software. Allowing the computer to be "creative" as well saves time and can make editing images more efficient.

Figure 3 Two different variations by two different users of a manipulated image.

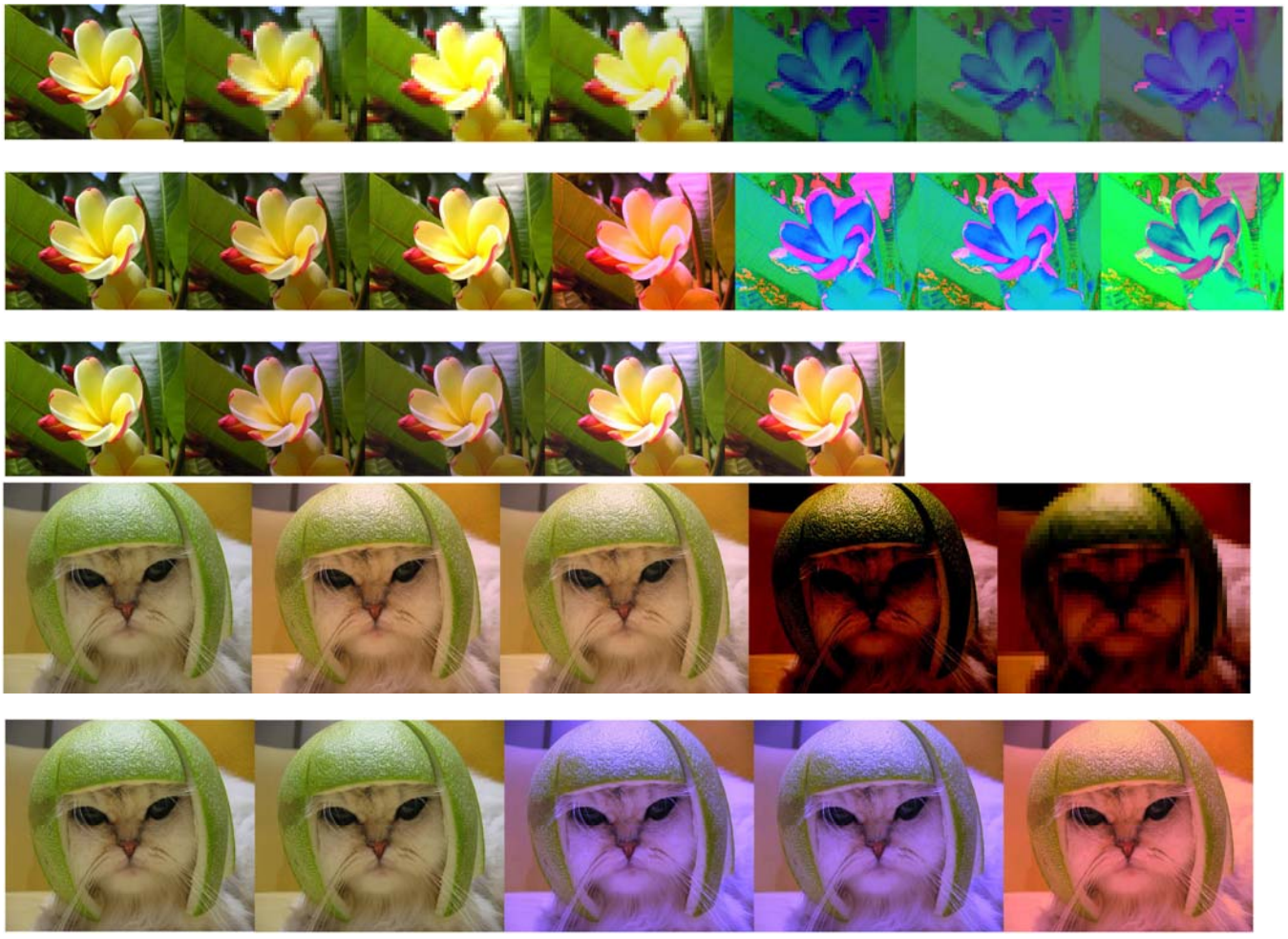
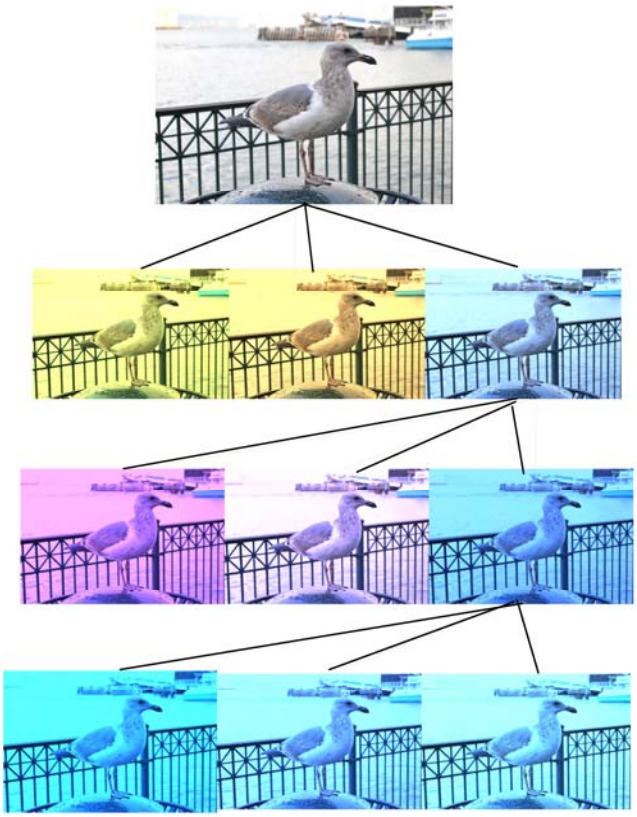


Figure 4 Three different variations by three different users of a manipulated image of a flower.

Figure 5: Two different variations by two different users of a manipulated image of a landscape.

Figure 6: One variation of a manipulated image of a building.

Figure 7: Image evolution



References

[1] MARKS, J., ANDALMAN, B., BEARDSLEY, P.A., FREEMAN, J., GIBSON, S., HODGINS, J., KANG, T. Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation. SIGGRAPH '97 proceedings, 1997, ACM Press/Addison-Wesley Publishing Co. New York, NY, USA.

[2] MUSGRAVE, KEN. Genetic Programming, Genetic Art.

http://www.kenmusgrave.com/mutatis.html

[3] SIMS, KARL, Computer Graphics (SIGGRAPH '91 proceedings), July 1991, pp.319-328.