JOURNAL OF FORENSIC SCIENCES

J Forensic Sci, January 2015, Vol. 60, No. 1
doi: 10.1111/1556-4029.12619
Available online at: onlinelibrary.wiley.com

TECHNICAL NOTE

DIGITAL & MULTIMEDIA SCIENCES

Harris Mushtaq,[1] *B.Eng.; Shahryar Rahnamayan,*[1] *Ph.D., P.Eng.; and Areeb Siddiqi,*[1] *B.Eng.*

Color Separation in Forensic Image Processing Using Interactive Differential Evolution

ABSTRACT: Color separation is an image processing technique that has often been used in forensic applications to differentiate among variant colors and to remove unwanted image interference. This process can reveal important information such as covered text or fingerprints in forensic investigation procedures. However, several limitations prevent users from selecting the appropriate parameters pertaining to the desired and undesired colors. This study proposes the hybridization of an interactive differential evolution (IDE) and a color separation technique that no longer requires users to guess required control parameters. The IDE algorithm optimizes these parameters in an interactive manner by utilizing human visual judgment to uncover desired objects. A comprehensive experimental verification has been conducted on various sample test images, including heavily obscured texts, texts with subtle color variations, and fingerprint smudges. The advantage of IDE is apparent as it effectively optimizes the color separation parameters at a level indiscernible to the naked eyes.

KEYWORDS: forensic science, document examination, image processing, color separation, ink discrimination, fingerprint recognition, interactive differential evolution, interactive evolutionary computation, sequential differential evolution, optimization, uncovered text recognition

Many optimization methods to solve variant minimization and maximization problems have been proposed. They can be categorized into two main groups, namely, mathematical (classical) methods and metaheuristic methods. Regardless of the selected optimization method, an objective function is needed to evaluate the fitness value of candidate solutions during the searching process. However, it is impossible or at least extremely complicated to model many real-world problems with mathematical equations. In general, these problems are related to human perceptions and cognitions processes that cannot be defined by simple mathematical equations.

Interactive evolutionary computation (IEC) combines a human's perceptual and cognitive capabilities with the power of evolutionary computation (EC) to solve cognitive-based optimization problems efficiently (1). In image processing applications of IEC, the user helps to provide the fitness function of the image as the computer seeks to approach fitter and fitter images based on user selection. Over the past few years, IEC has been used to solve the problems of retrieving images based on their properties, altering existing images to improve them, and creating new images for various applications.

Interactive evolutionary computation is a suitable solution to image retrieval problems. In such problems, an image is required to be retrieved from a large image database or another image. IEC is suitable because it dynamically reflects the subjectivity of the user, while considering possible changes in the user's mind (2). Moreover, IEC considers the fact that the user's solution can be diverse, fragmentary, and ambiguous.

Content-based image retrieval (CBIR) involves the finding of images in large databases based on image properties, such as the general color or emotion of the image. In CBIR, the computer approaches a required picture when the user selects a group of pictures that have a required parameter in common. An evolutionary algorithm is used to find more pictures similar to the pictures selected, by finding a parameter common to the selected pictures, and looking for more pictures that share this parameter. With multiple iterations of this process, the parameter in question becomes increasingly clear, and a desired image will be approached (3). Lai and Chen's CBIR algorithm (4) uses parameters such as mean color and standard deviation to find similarities between selected images. Sung-Bae and Joo-Young (5) show, using statistics, that the wavelength function of an image represents the general emotion of the image and then use this concept to retrieve images from large databases based on their emotions.

Interactive evolutionary computation has also been used in problems where it is necessary to find the right combination of facial features to construct a facial image. Sugimoto and Honda (6) examined and implemented IEC in multiple ways to find a facial image envisioned by the user and claimed that the "picking-some" method, a method according to which only some of the faces provided by the computer are given a fitness value, was most effective. Later, in (7), the convergence rate of the algorithm was improved by introducing fuzzy logic. Such face recognition algorithms have very important applications in forensic science, especially as a criminal's face can easily be drawn by witnesses to a crime.

[1]Department of Electrical and Computer Engineering, University of Ontario Institute of Technology (UOIT), 2000 Simcoe Street North (ACE-2026), Oshawa, ON, L1H 7K4, Canada.

Received 2 Aug. 2012; and in revised form 25 April 2013; accepted 20 July 2013.

Interactive evolutionary computation has also been used to point out important features and gather qualitative and quantitative data for large images. For instance, IEC has been used to accurately outline leaves to gather information about their shapes, growth rate, and colors (8). An IEC-based method has also been used in a medical application, namely, improving the fitting of an ellipse in an ultrasound image (9).

Images have many parameters associated with them, such as brightness, contrast, and gamma. Minor changes in the parameters of an image often result in a drastic improvement in the quality of the image. However, the number of possible combinations of parameter changes is extremely large. IEC can be very helpful in finding a combination of parameter changes that optimizes the quality of the image.

Interactive evolutionary computation has been applied to the creation and improvement of filters that improve the quality of images by various methods such as removing noise from images and optimizing the image gamma. In earlier algorithms, IEC was used to take user input to find appropriate parameters for a denoising filter (10). Newer algorithms combine multiple nonlinear filters, which take interactive evolutionary parameters to denoise images (11). IEC has also been applied in image processing to the optimization of gamma correction in images (12). Here, the computer selects increasingly better parameters for gamma correction based on the user input. Hayashida and Takagi (13) have produced an algorithm that allows nonexperts to create good filters. In their later research, Jaksa and Takagi (14) tested different types of filters, and showed their effectiveness. Jung, Lee and Cho (15) applied this concept to create a simple program that allows users to improve the quality of pictures taken on mobile devices by changing their parameters using IEC. IEC has also proven effective in eye illusion enhancement.

Interactive evolutionary computation has also been used to improve photographs of faces. Arakawa and Nomoto (16) created an algorithm that removes undesirable skin components, such as wrinkles and spots, from images of human faces to beautify the face. The parameters for the algorithm are selected using IEC. The system is effective for some actual face images. Arakawa and Nomoto's algorithm is improved by introducing new ways of improving the quality of facial images, such as edge enhancement and contrast enhancement (17).

Interactive evolutionary computation can greatly simplify the procedure of creating images, and can provide images with a quality better than envisioned. Kowaliw et al. (18) used IEC to create aesthetically pleasing images based on artificial ecosystems. The art system was further augmented using special measures to promote individual diversity through agent-based pixel-level techniques. IEC has been used to simplify user interfaces and make certain tasks feasible for nonspecialists, in many application domains. The use of IEC in the making of terrains, which feature prominently in many video games, allows one to deal with up to 800 different parameters, and so helps a nonspecialist user to rapidly generate terrains (19). In another application, IEC has been used to generate logos, allowing business owners to create their own logos without knowledge of any image processing software (20). Furthermore, IEC has also simplified the coloring of 3D shapes (21), improved the creation of visual illusions (22), and enabled the creation of faces inside QR code (23). Without the application of IEC, these problems are extremely time-consuming and can be solved only by experts.

Image processing algorithms have been developed for the extraction of important forensic information, such as text or fingerprints, from crossed-over or smudged images (24). These algorithms require user-selected image parameters. Because the user or the image processing algorithm may not recognize the best parameters individually, it is important to introduce IEC to optimize the quality of the information obtained from the evidence. In this research, we devise a procedure for the optimization of the parameters of an image processing algorithm that exploits to beneficial effect subjective judgments made by the user.

Proposed Method

Color separation and interactive differential evolution (IDE) are the two techniques utilized in this paper.

Color Separation

Sometimes valuable information is accidently or deliberately smudged or hidden. A simple example is when someone crosses out a piece of text with a pen to hide it. Although it is very difficult or impossible to decipher text that has been hidden by destructive means, such as burning of a document, deciphering text that has been hidden by nondestructive means, such as smudging with ink of a different color, is more feasible. In the cases of crossing out and smudging, in principle, so long as there is a difference between the color of the original text and the color of the smudges or occluding marks, even if this difference is undetectable by the naked eye, a technique known as color separation should be able to distinguish and separate the hidden text and occluding smudges or marks.

Color separation (24) is an algorithm that accepts a digital image as input, and outputs a resultant image, ideally with only the desired color that corresponds to the original text. The algorithm requires the specification of three parameters: desired color, undesired color, and background (or paper) color. It then performs operations on single pixels, breaking down each pixel into three components: Red (R), Green (G), and Blue (B). The algebra underlying color separation is somewhat complicated. However, the basic concept is simple. If we use $\vec{c}$ to denote the true color of a pixel in RGB, and $\vec{u}$, $\vec{d}$, $\vec{n}$ and $\vec{p}$ to represent unit vectors of the undesired, desired, normal, and paper colors, respectively, then:

$$\vec{c} = u \cdot \vec{u} + d \cdot \vec{d} + n \cdot \vec{n} + \vec{p}$$

Subtracting the undesired color from $\vec{c}$ potentially leads to a new color $\vec{c}'$ that is stripped of the undesired color such that:

$$\vec{c}' = d \cdot \vec{d} + n \cdot \vec{n} + \vec{p}$$

Solving the above equation produces the following results (24):

$$\vec{c} = \frac{\mathbf{M}\vec{c} + (d_3 n_2 p_1 - d_2 n_3 p_1 - d_3 n_1 p_2 + d_1 n_3 p_2 + d_2 n_1 p_3 - d_1 n_2 p_3)\vec{u}}{(d_3 n_2 u_1 - d_2 n_3 u_1 - d_3 n_1 u_2 + d_1 n_3 u_2 + d_2 n_1 u_3 - d_1 n_2 u_3)}$$

Where

$$\vec{n} = \begin{pmatrix} d_3p_2 - d_2p_3 - d_3u_2 + p_3u_2 + d_2u_3 - p_2u_3 \\ d_1p_3 - d_3p_1 + d_3u_1 - p_3u_1 - d_1u_3 + p_1u_3 \\ d_2b_1 - d_1b_2 - d_2u_1 + b_2u_1 + d_1u_2 - b_1u_2 \end{pmatrix}$$

and

$$\mathbf{M} = \begin{bmatrix} d_2n_1u_3 - d_1n_2u_3 - d_3n_1u_2 + d_1n_3u_2 & (d_3n_1 - d_1n_3)u_1 & (d_1n_2 - d_2n_1)u_1 \\ (d_2n_3 - d_3n_2)u_2 & d_2n_1u_3 - d_1n_2u_3 + d_3n_2u_1 - d_2n_3u_1 & (d_1n_2 - d_2n_1)u_2 \\ (d_2n_3 - d_3n_2)u_3 & (d_3n_1 - d_1n_3)u_3 & d_3n_2u_1 - d_2n_3u_1 - d_3n_1u_2 + d_1n_3u_2 \end{bmatrix}$$

This process is applied to each pixel of the input image, resulting in an image that contains only the desired text on the background.

Interactive Differential Evolution

Some models may be optimized using objective criteria, others using subjective criteria. While a single objective criterion can often be represented by a single function, multiple, complex, or subjective criteria are difficult to represent in mathematical terms. Without an established function, a computer cannot by itself fully optimize a model. In such cases, human subjectivity is required.

The interactive evolutionary optimization method chosen for use in our technique is known as IDE (25). IDE is a user-interactive and user-selective technique that produces different variants of a given individual, and converges the population, generation after generation, to an individual ideal in the mind of the user. User selection drives the evolution of the population. Figure 1 highlights some of the key steps involved in a single iteration of IDE.

Initialization is the first step in IDE. It randomly creates a specific number of individuals in the parent population, each associated with a certain number of parameters. In general, the user specifies the number of individuals and the range of allowable values for the parameters. These random individuals comprise the first generation (parent population) for the first iteration of IDE. *Mutation* is the second major step in IDE. Three individuals from the parent population are chosen randomly and are combined using the following formula to create a *noisy* or *donor vector*:

$$\text{Donor vector} = pm_3 + F(pm_1 - pm_2)$$

Here, F is the mutation constant (set to 0.5), and pm_1, pm_2, and pm_3 are the parameters of the three randomly chosen individuals. In the third step of IDE, *crossover* occurs between the donor vector and the *target vector* to produce a *trial vector*. The target vector is identical to an individual from the parent population from which a corresponding offspring is being produced. The last step of IDE is user-based *selection*, in which the user examines the target vector and trial vector, and simply chooses the one that is closer to the ideal individual in his or her mind. This selected individual enters the offspring population.

The initialization step occurs only once in IDE, whereas the last three steps, which comprise a single iteration of IDE, are repeated while there are individuals present in the current parent population, with the target vector being set to the next individual during each iteration. Consequently, each offspring population contains the same number of individuals as the corresponding parent population. Once all the individuals of the parent population have been processed, the parent population is replaced with the newly produced offspring population. The iterative processing continues until the user stops it once a satisfactory individual has been produced. It is important to mention that, although the first step of IDE creates random individuals, the process as a whole is by no means random. The entire point of IDE is to have a population converge as closely as possible to an individual that is ideal in the mind of the user.

Sequential Differential Evolution: The process mentioned above is that of linear differential evolution (DE), otherwise known as two-array DE. In linear DE, two arrays are used simultaneously, one for the parent population and the other for the offspring population. There is another technique, however,

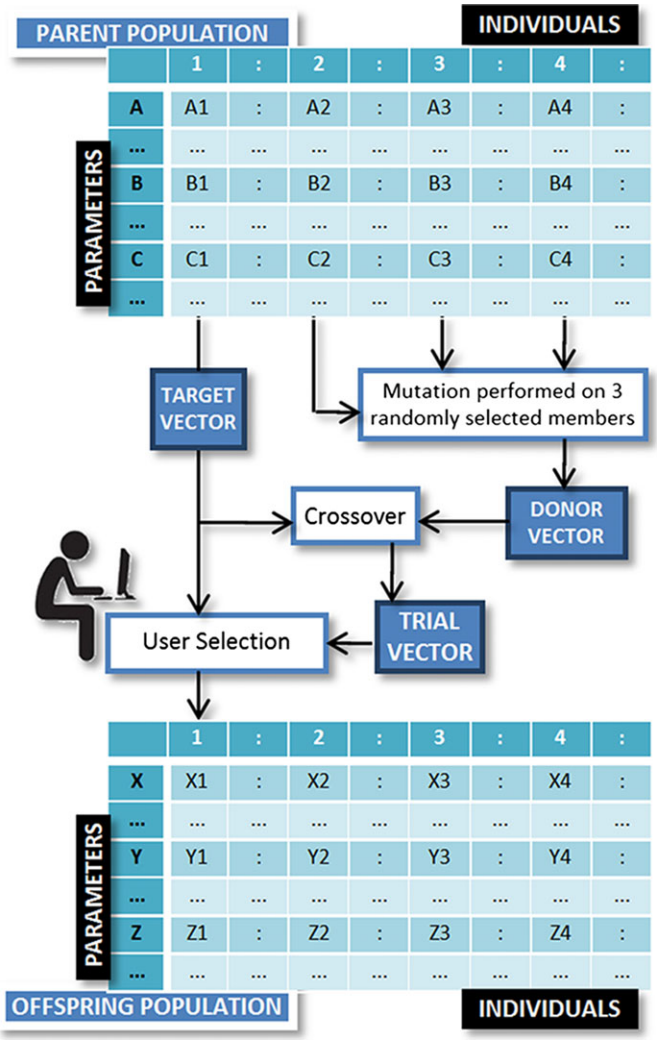

FIG. 1—*An iteration of interactive differential evolution.*

known as sequential DE or one-array DE (26), that uses only one array. Sequential DE is much more efficient when it comes to speed of convergence and the use of computer resources. The fundamental steps of sequential DE are the same as the linear DE, except for the important fact that in sequential DE only one array is used. The need to keep track of the offspring population is eliminated by replacing the parent member by the offspring member on the fly. In other words, as soon as an offspring is produced, it replaces the parent from which it was derived, as opposed to filling a separate array with offspring and then replacing the entire parent array with the offspring array in another step. It is clear that sequential DE saves computer memory by reducing the number of stored individuals by half. Furthermore, it also provides the important advantage of fast convergence. In the second step, when a parent member is mutated, three other members are randomly chosen from the same array. In linear DE, these three members are always from the parent population. However, in sequential DE, these members may have been selected by the user and re-entered into the array. These already "selected" members are better candidates for further mutations because they are better optimized than their corresponding parents. Thus, sequential DE speeds up the overall convergence of the population.

Application

Not only are color separation and IDE powerful algorithms individually, they also complement each other well. In our experiment, multiple test samples were created using various kinds of pens and inks. In addition, multiple methods were used to cover or smudge texts and fingerprints to simulate real scenarios. These samples were then scanned at high resolution to create a digital image. Matlab, which is a numerical computing software package, was deployed to combine color separation and IDE. Figure 2 shows an example of an interface created using Matlab, in which the user is prompted to make a selection between two images.

Color separation required a total of nine parameters to process an image: three components (RGB) of the three colors (desired, undesired, and paper color). However, to simplify the experiment, the background or paper color was always chosen to be white; that is, the default value of the background was set to [255, 255, 255], which corresponds to white in the RGB color scheme. This simplification is reasonable, because the paper color is usually constant and known in most cases. This simplification reduced the total number of required parameters for color separation to six. Thus, the purpose of IDE was to optimize these six unknown parameters.

Color separation and IDE worked concurrently. As mentioned above, the first step of IDE was to initialize a population. In the initialization stage, IDE assigned a random value to each of the six parameters. The members of the newly created parent population were then passed onto color separation, one by one. Figure 3 shows an example of a population initialized by IDE, processed by color separation, and displayed by an interface in Matlab. The next three steps—mutation, crossover, and selection—occurred for all members of the parent population. As seen in Fig. 2, a track of generation and member number was displayed in the interface, and a means of saving the best member observed was also implemented. Once all the members in the parent population had been processed, the offspring population became the parent population of the next generation. This time around, with the exception of the initialization step, all of the steps of IDE were performed again for all of the new members. This process continued until the user observed a satisfactory image, and saved this image as the final result.

FIG. 2—*User selection interface in Matlab.*

Discussion

Text or fingerprints can be covered or smudged in numerous ways. For that reason, several types of inks were used in this experiment to simulate real scenarios. In particular, ballpoint pens, felt-point pens, and water-based inks were used. The ultimate purpose was to optimize the color parameters of desired and undesired colors against a set background.

Figure 4 illustrates a sample processed using color separation and compares the results obtained using color separation individually versus using color separation in combination with IDE. Although both techniques produced readable images in this case,

FIG. 3—*Parent population initialized by interactive differential evolution.*

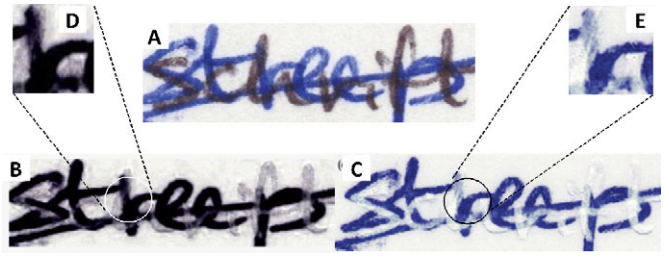

FIG. 4—*(A and C) Images produced by color separation alone. (B) Images produced using a combination of color separation and interactive differential evolution. (Insets D and E) Close-up views of areas where the differences are apparent.*

the example does highlight the advantage of using IDE. The image produced using color separation alone exhibited some loss of color of the original text. On the other hand, the combination of color separation and IDE resulted in a richer and a complete image. The insets in Fig. 4 further illustrate the improvements. It is therefore clear that choosing the original desired and undesired colors does not always yield the best results.

While the undesired color in most instances is easy to spot, the desired color may not always be so obvious. This can happen if the desired color is covered entirely with the undesired color, or if the difference between the desired and undesired colors is not detectable by the naked eye. Figures 5 and 6 illustrate some examples of such cases, and their corresponding desired and undesired colors. In Fig. 5*B*, green ink is completely obscured by the blue ink, and so it is virtually impossible to discern the two colors without any aid. Consequently, it becomes difficult to choose an appropriate color for the desired color parameter of color separation. In contrast, with the aid of IDE, a user can converge to optimized values of the desired color parameters.

On the other hand, Fig. 6*C* shows a sample that was created using two different pens of the same color—blue. Because the pens were different, their ink makeup played a role in producing

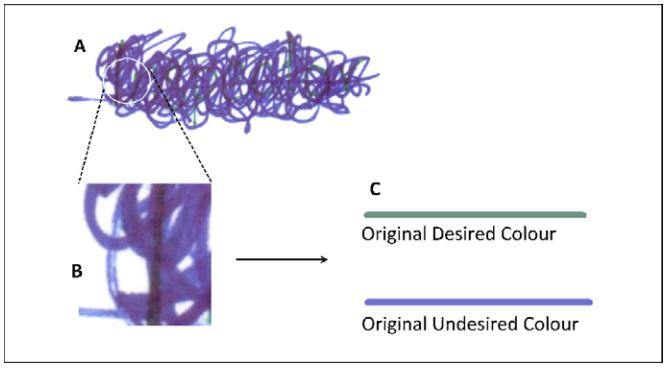

FIG. 5—*(A) Sample created with a green gel pen covered by a blue ball-point pen. (B) Inset of an area where it is difficult to spot the desired color.*

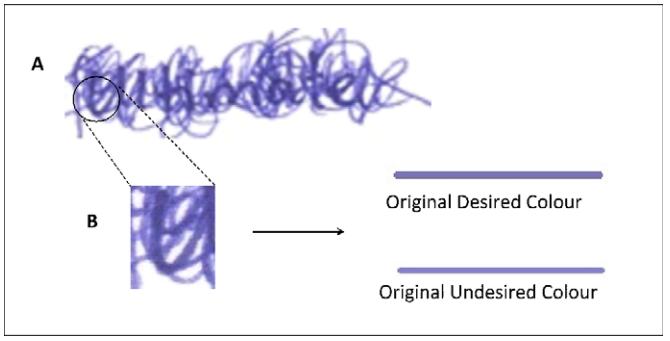

FIG. 6—*(A) Sample created with two blue pens. (B, C) The difference between the two colors is difficult to detect.*

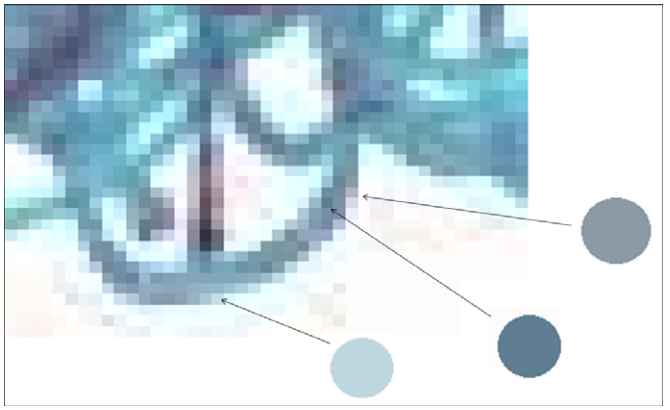

FIG. 7—*Blown-up image of a line. At least three different colors could be chosen to represent this line.*

FIG. 8—*(A–F) Samples (top-right of each box) and their corresponding resultant images using color separation alone (bottom-left) and color separation and interactive differential evolution in combination (bottom-right).*

FIG. 9—*A starting image, the randomly initialized parent population, and the offspring population ten generations later.*

different shades of blue. Such small variations cannot be detected by an unaided eye, even if zoomed, as shown in Fig. 6*B*. Again, IDE could be utilized to optimize both the desired and the undesired colors.

In addition to the difficulties mentioned above, there is one more factor that makes choosing a desired or an undesired color challenging. Figure 7 shows a blown-up image of a single line that may be of interest to a user. As illustrated, there is no way to simply choose the average color of the line. The pixelated view shows that there are several different shades along the length of a line. Choosing different colors will yield different results.

In this experiment, the algorithm combining color separation and IDE was applied to a set of samples. Figure 8 shows the results achieved by this combined algorithm, and the results achieved using color separation alone. With color separation alone, it is assumed that the desired and undesired colors were already known, although, as mentioned above, they are mostly unknown and often difficult to spot. It is clear from this illustration that IDE complements color separation well.

The power of IDE lies in the random initialization of the parent population and slow convergence to an optimized member. Figure 9 shows an example of a sample image that was initialized using IDE. It also shows the offspring population ten generations later, evolved and selectively chosen by a user. This example illustrates how the initial population is far from being optimized. In fact, none of the images in the initial population are readable. However, as IDE operates, the process of mutation generates further variations in the population, and genetic crossover converges the population toward an optimized member. Moreover, as the user observes more and more images, the better ones are selected and passed onto the offspring population.

Conclusion

The algorithm combining color separation and IDE is an effective method, making it specifically useful for forensic image processing. The results shown in Figs 8 and 9 were produced in less than ten generations, and were richer than their counterparts. Thus, a similar method could be used in a crime scene investigation where someone might have tried to cover-up a phone number, address, or other important information that could serve as evidence or lead to useful links in the investigation.

Despite the many advantages mentioned above, the algorithm combining color separation and IDE is limited by a number of factors. For instance, if the color of the background is not white, then three additional parameters must be added to IDE—the RGB components of the paper color. This would require more computation time, as a larger population size and more generations will be needed to reach an optimized image. User fatigue is an intrinsic limitation of IEC and it increases with time. Other factors that may render an approach based on the combination of color separation and IDE useless or impractical are the background color being multicolored, or the undesired color being not sufficiently translucent to let some shades of the desired color pass through. External factors, such as the scanned quality and resolution also influence the final results.

In summary, the key advantages of the algorithm combining color separation and IDE are:

- *Accuracy*: Optimized combinations of desired and undesired colors are found.
- *Extendibility*: Indiscernible colors to the naked eye become discernible to the computer.
- *Simplicity*: There is no need to guess or choose desired and undesired colors.
- *Speed*: There is quick convergence in a large data space of RGB.

References

1. Takagi H. Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. Proceedings of the IEEE. Piscataway, NJ: IEEE, 2001;89(9):1275–96.
2. Kato S, Iisaku S. An image retrieval method based on a genetic algorithm. Proceedings of the 12th International Conference on Information Networking; 1998 Jan 21–23; Tokyo, Japan. Piscataway, NJ: IEEE, 1998;333–6.
3. Arevalillo-Herraez M, Ferri FJ, Moreno-Picot S. An interactive evolutionary approach for content based image retrieval. IEEE International Conference on Systems, Man and Cybernetics (SMC 2009); 2009 Oct 11–14; San Antonio, TX. Piscataway, NJ: IEEE, 2009;120–5.
4. Chih-Chin L, Ying-Chuan C. An adaptive approach for color image retrieval. 2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN); 2011 May 27–29; Xi'an, China. Piscataway, NJ: IEEE, 2011;137–40.

5. Cho S-B, Lee J-Y. A human-oriented image retrieval system using interactive genetic algorithm. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans. Piscataway, NJ: IEEE, 2002;32 (3):452–8.
6. Sugimoto F, Honda N. A human interface to search and draw facial images in mind by using psychometrical space model of faces. 1999 IEEE International in Fuzzy Systems Conference Proceedings (FUXX-IEEE '99); 1999 Aug 22–25; Seoul, South Korea. Piscataway, NJ: IEEE, 1999;3:1585–90.
7. Sugimoto F, Yoneyama M. Robustness against instability of sensory judgment in a human interface to draw a facial image using a psychometrical space model. 2000 IEEE International Conference on Multimedia and Expo; 2000 July 30–Aug 2; New York, NY. Piscataway, NJ: IEEE, 2000;2:635–8.
8. Otoba K, Tanaka K, Hitafuji M. Image processing and interactive selection with Java based on genetic algorithms. In: Kozai T, Murase H, Hoshi T, editors. Artificial Intelligence in Agriculture 1998 (IFAC Proceedings Volumes); 1998 Apr 24–26; Makuhai, Japan. Oxford: Pergamon Press, 1998;83–8.
9. Lamard M, Hamitouche-Djabou C, Morice JDM, Bressolette L, Roux C. Interactive ellipse fitting in ultrasound images. Proceedings of the 25th Annual International Conference in Engineering in Medicine and Biology Society, 2003; 2003 Sept 17–21; Cancun, Mexico. Piscataway, NJ: IEEE, 2003;754–7.
10. Arakawa K, Nomoto K. Nonlinear denoising filter for images with interactive evolutionary computing considering the subjective assessment. IEEE Conference on Soft Computing in Industrial Applications (SMCia '08); 2008 June 25–27; Muroran, Japan. Piscataway, NJ: IEEE, 2008;264–8.
11. Katsuyama Y, Arakawa K. Color image interpolation for impulsive noise removal using interactive evolutionary computing. International Symposium on Communications and Information Technologies (ISCIT 2010); 2010 October 29–26; Tokyo, Japan. Piscataway, NJ: IEEE, 2010;264–8.
12. Tokuda Y, Hashino H, Ohashi G, Tsukada M, Kobayashi R, Shimodaira Y. Image quality enhancement support system by gamma correction using interactive evolutionary computation. IEEE International Conference on Systems, Man and Cybernetics, 2007 (ISIC); 2007 Oct 7–10; Montreal, Canada. Piscataway, NJ: IEEE, 2007;2906–10.
13. Hayashida N, Takagi H. Interactive EC-based signal processing. Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning; 2002 Nov 18–22; Jeju Island, South Korea. Singapore: Nanyang Technological University, School of Electrical & Electronic Engineering, 2002;375–9.
14. Jaksa R, Takagi H. Analysis and evaluation for interactive evolutionary computation based image processing. Proceeding of MPS Symposium; 2003 Jan; Kyoto, Japan, 2003;243–50. http://neuron.tuke.sk/jaksa/publications/Jaksa-Takagi-MPS2003.pdf.
15. Jung TM, Lee YS, Cho SB. Mobile interface for adaptive image refinement using interactive evolutionary computing. IEEE Congress on Evolutionary Computation; 2010 July 18–23; Barcelona, Spain. Piscataway, NJ: IEEE, 2010;1–7.
16. Ohchi S, Sumi S, Arakawa K. A nonlinear filter system for beautifying facial images with contrast enhancement. Proceedings of the 10th International Symposium on Communications and Information Technologies (ISCIT); 2010 Oct 26–29; Tokyo, Japan. Piscataway, NJ: IEEE, 2010;13–7.
17. Arakawa K, Nomoto K. A system for beautifying face images using interactive evolutionary computing. Proceedings of 2005 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2005); 2005 Dec 13–16; Hong Kong, China. Piscataway, NJ: IEEE, 2005;13–6.
18. Kowaliw T, McCormack J, Dorin A. An interactive electronic art system based on artificial ecosystemics. 2011 IEEE Symposium on Artificial Life (ALIFE 2011); 2011 April 11–15; Paris, France. Piscataway, NJ: IEEE, 2011;162–9.
19. Walsh P, Gade P. Terrain generation using an interactive genetic algorithm. 2010 IEEE Congress on Evolutionary Computation; 2010 July 18–23; Barcelona, Spain. Piscataway, NJ: IEEE, 2010;1–7.
20. Yamada M, Onisawa T. Logo drawing system applying interactive genetic algorithms. Proceedings of the 2006 IEEE International Conference on Information Reuse and Integration (2006 IRI); 2006 Sept 16–18; Waikoloa, HI. Piscataway, NJ: IEEE, 2006;238–43.
21. Kagawa T, Nishino H, Utsumiya K. A sensitive coloring and texture mapping on 3D shapes. Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics; 2004 Oct 10–13; The Hague, Netherlands. Piscataway, NJ: IEEE, 2004;6:5748–53.
22. Mohamad Z, Darvish A, Rahnamayan S. Eye illusion enhancement using interactive differential evolution. Proceedings of the 2011 IEEE Symposium on Differential Evolution; 2011 April 11–15; Paris, France. Piscataway, NJ: IEEE, 2011;1–7.
23. Ono S, Nakayama S. A system for decorating QR code with facial image based on interactive evolutionary computation and case-based reasoning. 2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC); 2010 Dec 15–17; Fukuoka, Japan. Piscataway, NJ: IEEE, 2010;401–6.
24. Berger CEH, de Koeijer JA, Glas W, Madhuizen HT. Color separation in forensic image processing. J Forensic Sci 2006;51(1):100–2.
25. Storn R, Price K. Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces. J Global Optim 1997;11(4):341–59.
26. Price KV. Differential evolution: a fast and simple numerical optimizer. 1996 Biennial Conference of the North American Fuzzy Information Processing Society (1996 NAFIPS); 1996 June 19–22; Berkeley, CA. Piscataway, NJ: IEEE, 1996;524–27.

Additional information and reprint requests:
Shahryar Rahnamayan, Ph.D., P.Eng.
Department of Electrical, Computer, and Software Engineering
University of Ontario Institute of Technology (UOIT)
2000 Simcoe Street North (ACE-2026)
Oshawa
ON L1H 7K4
Canada
E-mail: shahryar.rahnamayan@uoit.ca