Global Journal of Computer Science and Technology: F
Graphics & Vision
Volume 21 Issue 1 Version 1.0 Year 2021
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

An Ambigramic Image File Format

By Wenyi Cui, Kohei Inoue & Kenji Hara
Kyushu University

Abstract- We propose an image file format that can be read in two ways, where two images are recorded in a single file as a bit sequence, and the forward reading the bit sequence makes one of the two images visible, or the backward reading makes another image visible. Such a way of looking at a binary data in two ways resembles that of an ambigram, which is a piece of calligraphy that can be read in two ways by rotating it or introducing other perspectives. The proposed ambigramic image file format is compared with the graphics interchange format (GIF) experimentally, and the results show the better quality of the ambigramic images than that of GIF images.

Keywords: *ambigram, image file format, error diffusion, integral image-based signal-to-noise ratio (ISNR).*

GJCST-F Classification: *I.3.3*

An Ambigramic Image File Format

Wenyi Cui [α], Kohei Inoue [σ] & Kenji Hara [ρ]

Abstract- We propose an image file format that can be read in two ways, where two images are recorded in a single file as a bit sequence, and the forward reading the bit sequence makes one of the two images visible, or the backward reading makes another image visible. Such a way of looking at a binary data in two ways resembles that of an ambigram, which is a piece of calligraphy that can be read in two ways by rotating it or introducing other perspectives. The proposed ambigramic image file format is compared with the graphics interchange format (GIF) experimentally, and the results show the better quality of the ambigramic images than that of GIF images.

Keywords: ambigram, image file format, error diffusion, integral image-based signal-to-noise ratio (ISNR).

I. Introduction

Ambigram is a typographical design that can be read in multiple orientations as shown in Fig. 1, where a word 'ambigram' is written in a rotationally symmetric manner. There are other types of ambigrams such as mirror, perceptual shift and 3D ambigrams [1]. Langdon [2], [3] has made a lot of interesting ambigrams for a long time. Those ambigramic artworks give pleasure to the viewers, and provide great inspiration for artists and engineers.

Figure 1: Example of ambigram

Inspired by such successful ambigrams, in this paper we suggest an application of the idea of ambigram to an ambigramic interpretation of a bit sequence. In other words, we propose a method for describing two images with a bit sequence, which is a digital ambigram that can be seen as one of the two images if the bit sequence is read forward direction, or as another image if the bit sequence is read backward direction. The proposed ambigramic images are compared with the graphics interchange format (GIF) images [4], and the effectiveness of the proposed method is demonstrated in the experiments using natural images, where an image quality measure is used for evaluating the quality of halftone images including palette-based images such as GIF images.

The rest of this paper is organized as follows: Section 2 describes the proposed algorithm for generating ambigramic bit sequences. Section 3 shows experimental results. Finally, Section 4 concludes this paper.

II. Proposed Ambigramic Image Data Structure

In this section, we propose a method for formatting image data into an ambigramic data structure, which is recorded in a binary file. Assume that two images are given as an input data as $A = [a_{ij}]$ and $B = [b_{ij}]$, where a_{ij} and b_{ij} denote color vectors at the pixel position $(i; j)$ in the images A and B, respectively, for $i = 1; 2; : : : ;m$ and $j = 1; 2; : : : ; n$, i.e., A and B are the same size. Then we attempt to store A and B in a storage as a single file. Figure 2 illustrates the situation, where A and B are represented by a bit sequence, and stored in some place of a storage. If we read the bit sequence from left to right, then we see the image A. On the other hand, if we read it from right to left, then we see the image B. The procedure for constructing such a bit sequence from A and B is described as follows.

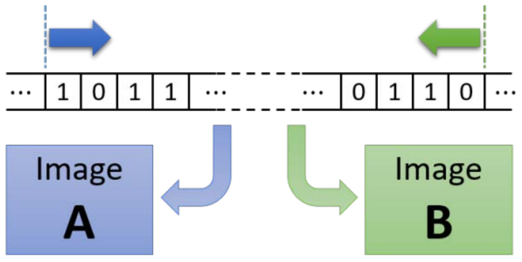


Figure 2: Illustration of an ambigramic bit sequence

Let us consider a typical case that aij and bij are 24-bit RGB color vectors as $a_{ij} = [r_{ij}^A, g_{ij}^A, b_{ij}^A]$ and $b_{ij} = [r_{ij}^B, g_{ij}^B, b_{ij}^B]$, each element of which is represented by 8 bits. Here, we introduce a binary representation of r_{ij}^A as $(r_{ij1}^A\, r_{ij2}^A\, r_{ij3}^A\, r_{ij4}^A \,|\, r_{ij5}^A\, r_{ij6}^A\, r_{ij7}^A\, r_{ij8}^A)_2$, where the third subscript in each element ranging from 1 to 8 indicates the significance of each digit, i.e., r_{ij1}^A denotes the most significant bit (MSB), and r_{ij8}^A denotes the least significant bit (LSB) in this binary representation. Other elements in a_{ij} and b_{ij} are also represented in binary in the same manner as r_{ij}^A. Moreover, we abbreviate a series of four bits $r_{ij1}^A\, r_{ij2}^A\, r_{ij3}^A\, r_{ij4}^A$ to $r_{ij1:4}^A$ for the sake of simplicity.

Author α σ ρ: Department of Communication Design Science, Kyushu University, 4-9-1, Shiobaru, Minami-ku, Fukuoka, 815-8540 Japan.
e-mails: cuiwenyi1996@outlook.com,
{k-inoue, hara}@design.kyushu-u.ac.jp

Our basic principle is to preserve more significant bits of the original image data in the resultant ambigramic image file. Following this principle, we first extract four MSBs from both *A* and *B* as follows:

$$r^A_{1,1,1:4}, g^A_{1,1,1:4}, b^A_{1,1,1:4};\; r^A_{1,2,1:4}, g^A_{1,2,1:4}, b^A_{1,2,1:4};\; \cdots$$
$$\cdots; r^A_{m,n,1:4}, g^A_{m,n,1:4}, b^A_{m,n,1:4} \quad (1)$$
$$r^B_{1,1,1:4}, g^B_{1,1,1:4}, b^B_{1,1,1:4};\; r^B_{1,2,1:4}, g^B_{1,2,1:4}, b^B_{1,2,1:4};\; \cdots$$
$$\cdots;\; r^B_{m,n,1:4}, g^B_{m,n,1:4}, b^B_{m,n,1:4} \quad (2)$$

where the horizontally adjacent pixels are separated by semicolons. Next, we reverse the order of the bit sequence in (2) as follows:

$$b^B_{m,n,4:1}, g^B_{m,n,4:1}, r^B_{m,n,4:1};\; \cdots$$
$$\cdots;\; b^B_{1,2,4:1}, g^B_{1,2,4:1}, r^B_{1,2,4:1};\; b^B_{1,1,4:1}, g^B_{1,1,4:1}, r^B_{1,1,4:1} \quad (3)$$

where note that the third subscripts (1:4) is also reversed as (4:1), e.g., $r^B_{1,1,4:1}$ denotes $r^B_{1,1,4}\, r^B_{1,1,3}\, r^B_{1,1,2}\, r^B_{1,1,1}$. Then we combine (1) and (3) as follows:

$$(r^A_{1,1,1:4}|b^B_{m,n,4:1})_2, (g^A_{1,1,1:4}|g^B_{m,n,4:1})_2, (b^A_{1,1,1:4}|r^B_{m,n,4:1})_2;$$
$$\cdots;$$
$$(r^A_{m,n,1:4}|b^B_{1,1,4:1})_2, (g^A_{m,n,1:4}|g^B_{1,1,4:1})_2, (b^A_{m,n,1:4}|r^B_{1,1,4:1})_2 \quad (4)$$

which can be viewed as an ambigramic bit sequence, because if we read the bit sequence in (4) from left to right then the four MSBs in every pixel show the image *A*, on the other hand, if we read it from right to left then the four LSBs in every pixel are inversely read to show the image *B*. However, the resultant images can be corrupted by the replacement of the original four LSBs with others. To alleviate such a quality deterioration in the images, we next propose an error diffusion algorithm for generating better bit sequence, which improves the visual quality of the images.

The proposed error diffusion algorithm is described as follows. For one image *A*, all pixels are processed in a standard raster scan order, and at the same time, for another image *B*, all pixels are processed in the inverse raster scan order. For the first pixel in image *A*, the original color vector $a_{1,1}$ changes to

$$\tilde{\boldsymbol{a}}_{1,1} = \begin{bmatrix} (r^A_{1,1,1:4}|b^B_{m,n,4:1})_2 \\ (g^A_{1,1,1:4}|g^B_{m,n,4:1})_2 \\ (b^A_{1,1,1:4}|r^B_{m,n,4:1})_2 \end{bmatrix}. \quad (5)$$

Subtracting $\tilde{a}_{1,1}$ from $a_{1,1}$, we define an error vector by

$$\boldsymbol{e}^A_{1,1} = \boldsymbol{a}_{1,1} - \tilde{\boldsymbol{a}}_{1,1}, \quad (6)$$

which is diffused to unprocessed neighboring pixels as

$$\boldsymbol{a}_{1+k,1+l} \leftarrow \boldsymbol{a}_{1+k,1+l} + w_{k,l}\boldsymbol{e}^A_{1,1}, \quad (7)$$

Where *k* and *l* denote relative indices to access the neighboring pixels, and $w_{k,l}$ denotes error diffusion coefficients or error filter [5]. On the other hand, for the last pixel in image *B*, the original color vector $b_{m,n}$ changes to

$$\tilde{\boldsymbol{b}}_{m,n} = \begin{bmatrix} (r^B_{m,n,1:4}|b^A_{1,1,4:1})_2 \\ (g^B_{m,n,1:4}|g^A_{1,1,4:1})_2 \\ (b^B_{m,n,1:4}|r^A_{1,1,4:1})_2 \end{bmatrix}. \quad (8)$$

Algorithm 1 Constructing ambigramic bit sequence

```
1:  procedure AMBIGRAMIC($A, B$)
2:      $\tilde{A} \leftarrow$ zeros_like($A$) where $A = [\boldsymbol{a}_{ij}]$
3:      $\tilde{B} \leftarrow$ zeros_like($B$) where $B = [\boldsymbol{b}_{ij}]$
4:      $m, n, \_ \leftarrow A$.shape
5:      for $i \leftarrow 1$ to $m$ do
6:          $i_R \leftarrow m + 1 - i$
7:          for $j \leftarrow 1$ to $n$ do
8:              $j_R \leftarrow n + 1 - j$
9:              Round the elements of $\boldsymbol{a}_{ij}$ and $\boldsymbol{b}_{i_R,j_R}$ to 8-bit integers
10:             $\tilde{\boldsymbol{a}}_{ij} \leftarrow \begin{bmatrix} (r^A_{i,j,1:4}|b^B_{i_R,j_R,4:1})_2 \\ (g^A_{i,j,1:4}|g^B_{i_R,j_R,4:1})_2 \\ (b^A_{i,j,1:4}|r^B_{i_R,j_R,4:1})_2 \end{bmatrix}$
11:             $\tilde{\boldsymbol{b}}_{i_R,j_R} \leftarrow \begin{bmatrix} (r^B_{i_R,j_R,1:4}|b^A_{i,j,4:1})_2 \\ (g^B_{i_R,j_R,1:4}|g^A_{i,j,4:1})_2 \\ (b^B_{i_R,j_R,1:4}|r^A_{i,j,4:1})_2 \end{bmatrix}$
12:             $\boldsymbol{e}^A_{ij} \leftarrow \boldsymbol{a}_{ij} - \tilde{\boldsymbol{a}}_{ij}$
13:             $\boldsymbol{e}^B_{i_R,j_R} \leftarrow \boldsymbol{b}_{i_R,j_R} - \tilde{\boldsymbol{b}}_{i_R,j_R}$
14:             for $(k, l) \in KL_{ij}$ do
15:                 $\boldsymbol{a}_{i+k,j+l} \leftarrow \boldsymbol{a}_{i+k,j+l} + w_{k,l}\boldsymbol{e}^A_{i,j}$
16:                 $\boldsymbol{b}_{i_R-k,j_R-l} \leftarrow \boldsymbol{b}_{i_R-k,j_R-l} + w_{k,l}\boldsymbol{e}^B_{i_R,j_R}$
17:             end for
18:         end for
19:     end for
20:     return $\tilde{A} = [\tilde{\boldsymbol{a}}_{ij}]$ and/or $\tilde{B} = [\tilde{\boldsymbol{b}}_{i,j}]$
21: end procedure
```

Subtracting $\tilde{b}_{m,n}$ from $b_{m,n}$, we define an error vector by

$$\boldsymbol{e}^B_{m,n} = \boldsymbol{b}_{m,n} - \tilde{\boldsymbol{b}}_{m,n}, \quad (9)$$

which is diffused to unprocessed neighboring pixels as

$$\boldsymbol{b}_{m-k,n-l} \leftarrow \boldsymbol{b}_{m-k,n-l} + w_{k,l}\boldsymbol{e}^B_{m,n}, \quad (10)$$

where note that the sign of the relative indices k and l is reversed except in $w_{k,l}$ because of the inverse raster scan.

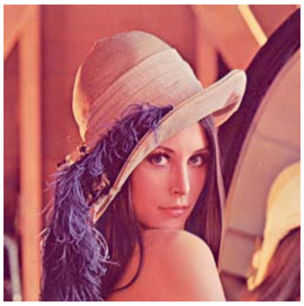
After the above error diffusion procedures in (7) and (10), we proceed to the next pixels $a_{1,2}$ and $b_{m,n-1}$, and the error diffusion procedures are repeated until the end of the scan.

The proposed error diffusion procedure is summarized in Algorithm 1, where the function 'zeros_like' returns an array of zeros being the same size as the argument array, A.shape is a property to get the array dimensions of A, and KL_{ij} denotes a set of the pair of the relative indices k and l satisfying $1 \le i + k \le m$ and $1 \le j + l \le n$.

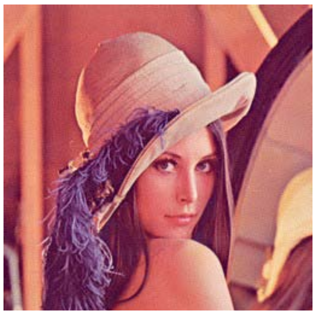
III. Experimental Results

In this section, we demonstrate the performance of the proposed method for formatting image data into an ambigramic data structure.

First, we show an example of the proposed ambigramic image data with two images selected from the standard image database SIDBA [6]. Figures 3(a) and (b) show the two standard images from SIDBA. Assume that Figs. 3(a) and (b) are images A and B, respectively. Then Algorithm 1 returns the corresponding ambigramic images $\tilde{A} = [\tilde{a}_{ij}]$ and $\tilde{B} = [\tilde{b}_{ij}]$ as shown in Figs. 3(c) and (d), which can be saved as an ambigramic bit sequence, the forward reading of which shows Fig. 3(c), and the backford reading shows Fig. 3(d). For example, the RGB values at the top left pixel of Fig. 3(c) and at the bottom right pixel of Fig. 3(d) are shown in Fig. 4, where the top left and the bottom right pixels $\tilde{a}_{11}$ and $\tilde{b}_{mn}$ have the RGB values (232; 130; 116) and (46; 65; 23), which are also expressed as binary numbers. If we reverse the order of the bit sequences of RGB values in $\tilde{a}_{ij}$, then we have the bit sequences of BGR values in $\tilde{b}_{mn}$, and vice versa.


(a) Image A

(b) Image B


(c) $\tilde{A}$ by Alg. 1

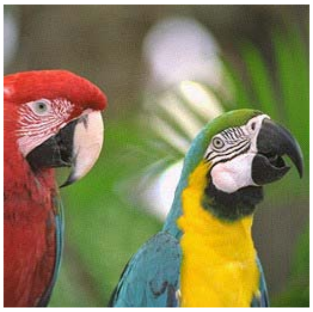

(d) $\tilde{B}$ by Alg. 1

Figure 3: Original images (a) and (b) converted into ambigramic data (c) and (d)

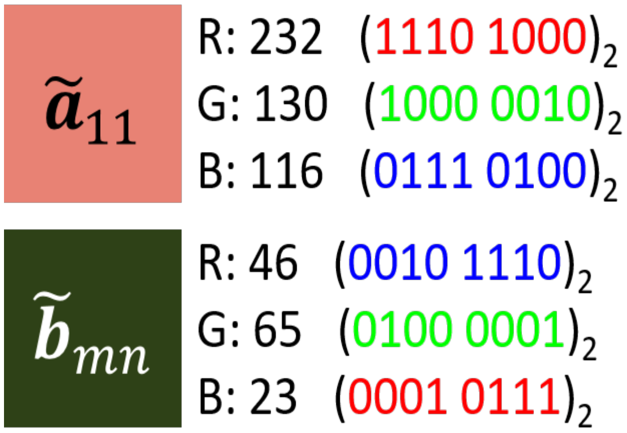


Figure 4: Comparison of pixel values

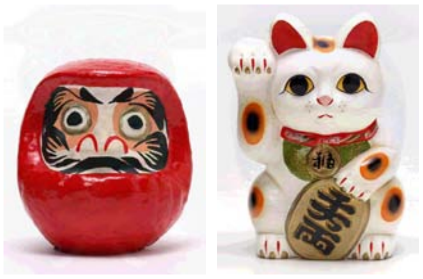
Figures 5(a) and (b) show two original images each of which has 640 $\times$ 480 pixels, and the following figures show the format-converted images from the original ones (a) and (b). Figures 5(c) and (d) show the graphics interchange format (GIF) images, and the GIF format is widely used on the Web due to its wide support and portability [4]. Figures 5 (e) and (f) show the results of the proposed ambigramic method without error diffusion (ED), where a daruma (dharma) doll and a beckoning cat can be seen as well as the above GIF images. Figures 5(g) and (h) show the results with ED (Algorithm 1), where we used Floyd and Steinberg's error filter [7] for $w_{k,l}$ in (7) and (10) as shown in Fig. 6 where "#" denotes the current pixel being processed, and "-" denotes the past pixel, and the visual quality is improved compared with the former results (e) and (f).

(a) Image A

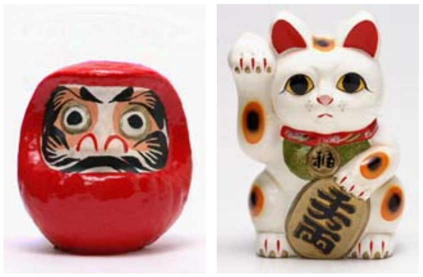
(b) Image B

(c) GIF image A

(d) GIF image B



(e) $\tilde{A}$ without ED (f) $\tilde{B}$ without ED



(g) $\tilde{A}$ by Alg. 1 (h) $\tilde{B}$ by Alg. 1

Figure 5: Examples of file format conversion

The GIF images are palette-based ones each of which has a palette table of 256 colors. That is, the GIF images are a sort of halftone images [5] as well as the proposed ambigramic images. To evaluate the quality of those images objectively and quantitatively, we present a parameter-free measure based on integral image [8].

-	#	$w_{0,1} = \frac{7}{16}$
$w_{1,-1} = \frac{3}{16}$	$w_{1,0} = \frac{5}{16}$	$w_{1,1} = \frac{1}{16}$

Figure 6: Error diffusion coefficients by Floyd-Steinberg

First, we compute the integral images of both reference (original) R and test (format-converted) T images, which are denoted as $\bar{R} = [\bar{r}_{ij}]$ and $\bar{T} = [\bar{t}_{ij}]$, respectively. Then we compute the signal-to-noise ratio (SNR) [9] of the integral images $\bar{R}$ and $\bar{T}$ as follows:

$$\mathrm{ISNR}(R,T) = \mathrm{SNR}\left(\bar{R},\bar{T}\right) \qquad (11)$$

$$= 10\log_{10}\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\bar{r}_{ij}^2}{\sum_{i=1}^{m}\sum_{j=1}^{n}(\bar{r}_{ij}-\bar{t}_{ij})^2}. \qquad (12)$$

We call this measure the integral image-based SNR (ISNR).

For the format-converted images in Fig. 5, their ISNRs are plotted in Fig. 7, where the vertical and horizontal axes denote the ISNR value and image name, respectively, and green, yellow and orange bars denote the file formats, GIF, ambigramic format without ED and the final ambigramic format by Algorithm 1, respectively. Algorithm 1 achieves higher ISNR values than GIF format for both images, which demonstrates the effectiveness of the proposed algorithm.

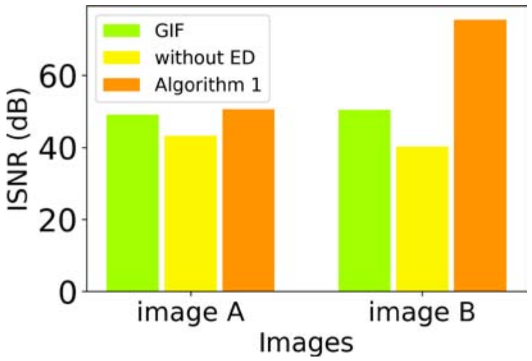


Figure 7: Integral image-based SNR (ISNR)

Additionally, we examine the applicability of the proposed algorithm to natural images shown in Fig. 8, where Fig. 8(a) shows five pairs of natural images where the top row shows the first images A, and the second row shows the corresponding second ones B. Figures 8(b), (c) and (d) show the corresponding format-converted images by GIF, the proposed ambigramic method without ED and Algorithm 1. Although, in Fig. 8(c), we can see false contours, they are removed in Fig. 8(d).

The ISNR values for those images in Fig. 8 are summarized in Table 1, where the proposed Algorithm 1 achieves higher values than the GIF images and ambigramic images without ED.

Table 1: ISNR values (dB)

	Pair 1	Pair 2	Pair 3	Pair 4	Pair 5
GIF image *A*	49.97	49.39	46.80	48.41	50.13
GIF image *B*	46.51	50.96	46.26	48.82	47.08
A˜ without ED	38.05	46.97	33.59	47.34	45.28
B˜ without ED	34.87	45.56	35.46	50.90	40.00
A˜ by Alg. 1	**73.93**	**60.81**	**82.48**	**95.21**	**75.76**
B˜ by Alg. 1	**63.61**	**88.92**	**53.09**	**94.37**	**85.13**

IV. Conclusions

In this paper, we proposed an algorithm for formatting an ambigramic image file into which two images of the same size are recorded. If we read it forward direction, then we see the first image, on the other hand, if we read it backward direction, then we see the second one. We compared the proposed ambigramic image file format with the GIF format which has acquired a widespread use on the Web, and demonstrated that the proposed ambigramic images achieved higher quality than the GIF images based on an image quality measure. Experimental results showed that the proposed algorithm is also applicable to natural images.

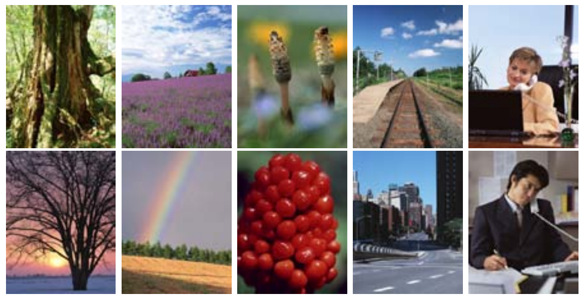
Acknowledgments

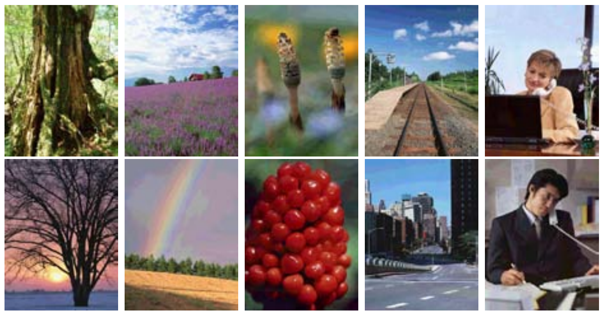
This work was supported by JSPS KAKENHI Grant Number JP16H03019.

References Références Referencias

1. Wikipedia, "Ambigram — Wikipedia, the free encyclopedia," https://en.wikipedia.org/wiki/ Ambigram, 2020, [Online; accessed 06-March-2020].
2. John Langdon, "Ambigrams, logos, & word art.," http://www.johnlangdon.net/, 1996, [Online; accessed 07-February-2019].
3. J. Langdon, Wordplay: The Philosophy, Art, and Science of Ambigrams, Bantam, 2005.
4. Wikipedia, "GIF — Wikipedia, the free encyclopedia," https://en.wikipedia.org/wiki/GIF, 2020, [Online; accessed 06-March-2020].
5. Daniel L. Lau and Gonzalo R. Arce, Modern Digital Halftoning, Second Edition, CRC Press, Inc., Boca Ra- ton, FL, USA, 2007.
6. M. Sakauchi, Y. Ohsawa, M. Sone, and M. Onoe, "Man- agement of the standard image database for image pro- cessing researches (sidba)," ITEJ Technical Report, vol. 8, no. 38, pp. 7–12, 1984.
7. Robert W. Floyd and Louis Steinberg, "An Adaptive Al- gorithm for Spatial Greyscale," Proceedings of the So- ciety for Information Display, vol. 17, no. 2, pp. 75–77, 1976.
8. Paul Viola and Michael Jones, "Robust real-time object detection," International Journal of Computer Vision, vol. 57, no. 2, pp. 137–154, 2002.
9. Wikipedia, "Signal-to-noise ratio — Wikipedia, the free encyclopedia," https://en.wikipedia.org/wiki/ Signal-to-noise_ratio, 2020, [Online; accessed 06-March-2020].

(a) Original images


(b) GIF images


(c) Ambigramic images without ED

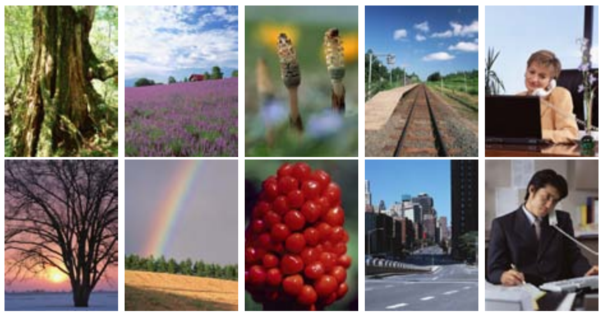

(d) Ambigramic images by Algorithm 1

Figure 8: Examples with natural images