The Journal of the Society for Art and Science, Vol. 17, No. 2, pp. 52 – 61 (2018)

Generation of CAPTCHA Using Sprinkled Destructors

Youngha Chang Akinori Urayama Miho Watanabe Nobuhiko Mukai

Tokyo City University

{chang, urayama, miho_watanabe, mukai} @ vgl.cs.tcu.ac.jp

Abstract
CAPTCHA is a security system that is intended to distinguish computers and humans. Although many types of CAPTCHA programs exist, many CAPTCHA solver services are now becoming available. Humans can recognize partially-hidden objects that computers cannot. In this paper, we use this feature to propose a new text-based CAPTCHA, in which sprinkled destructors are placed on and under the characters. Based on the evaluation of our method, we have found that a general character recognition software called Google Tesseract and a CAPTCHA solver system are unable to recognize the CAPTCHA generated by our method although humans can recognize them.

1 Introduction

Web-based services are used all over the world. Unfortunately, programs called spam bots often try to abuse these services, such as adding ads in the comment fields of blogs. In this section, we explain three methods to prevent spam bots.

The first method is an automatic comment filter such as Mollom [1] and Akismet [2], that screen all comments and block spam messages before they are posted to websites. However, an automatic filter sometimes causes over-filtering, which eliminates real user comments, or keeps spam comments. The second method is called form honeypot, that is based on the creation of an input field that is invisible to humans. Although real users leave it out, spam bots are likely to fill in the field. The disadvantage of this technique is that spammers can access the HTML of the page, find out the use of honeypot on the form, and easily bypass it. The third method is *Completely Automated Public Turing test to tell Computers and Humans Apart* (CAPTCHA), which is a test to discriminate between humans and computers [3]. According to the website for DRUPAL, one of the most famous open source content management system (CMS) projects, the order of the utilization frequency is text-based CAPTCHA, honey-pot, image-based CAPTCHA, and automatic comment filtering [4]. Actually, web sites such as Google®, Yahoo!®, and Microsoft® adopt CAPTCHA services to prevent spam bots. CAPTCHA is still regarded as the most useful tool to detect spam bots. In this paper, we focus on the text-based CAPTCHA system.

Gimpy [3], which was originally built for Yahoo!, is a typical example of traditional text-based CAPTCHA. It chooses words from a dictionary and distorts them. However, Mori and Malik have already recognized the Gimpy series of CAPTCHA [5]. For EZ-Gimpy, which is a refined version of the original Gimpy system, the recognition rate of CAPTCHA image was 92%. For the most difficult CAPTCHA of the Gimpy system, which has 3 distorted and overlaid words in an image, Mori and Malik's method succeeded in recognizing one image at a rate of 33% [5].

Microsoft also adopts a CAPTCHA system on its account creation web page. The CAPTCHA has distorted and multi-layered characters. Hong et al. collected 600 CAPTCHAs from the Microsoft website and tried to recognize them. The method could automatically separate the characters, and it recognized one character with 91.07% accuracy by using the convolutional neural network(CNN). Because Microsoft CAPTCHA uses six characters in one image, the accuracy rate of one CAPTCHA image is 57.05% [6]. Gao et al. also tried to recognize the Microsoft CAPTCHA system. Their target was the newest Microsoft CAPTCHA system, which had two layers. Their method estimated the boundary lines between layers and characters, and the method fed the characters to CNN. Finally, the average accuracy was 44.6% for one CAPTCHA image [7].

Google provides another CAPTCHA system called *reCAPTCHA*, which was broken by Starostenko et al. [8]. They used support vector machine (SVM) classifier, and the recognition rate of one character was 94% [8].

Komiyama et al. proposed a text-based CAPTCHA [9] with subjective contour, which is one of the human visual features. They evaluated the robustness of their CAPTCHA system with SVM. On the CAPTCHA, characters were recognized with the rate of 60%, which means the recognition rate of one CAPTCHA image is 7.8% because there are five characters in one image.

George et al. proposed another very powerful method to break text-based CAPTCHAs [10]. They proposed a new network model called Recursive Cortical Network (RCN), which was inspired by human brain's visual cortex. The remarkable point is its training efficiency. For example, for reCAPTCHA, their method can recognize one character with 94.3% accuracy only if they feed five clean training examples per character.

Up to the present, we described text-based CAPTCHA. There are other styles of CAPTCHA to prevent attacks.

For example, Tamura et al. proposed an image-based CAPTCHA system [11]. The algorithm picks a noun from a predefined dictionary, and selects a related image by using a search engine.

Then, the algorithm partially hides the image by using destructors. This partially hidden image is used as a CAPTCHA, and the user answers the name of the object shown in the image. The authors investigated the recognition rate of their system, and the accuracy rate was 97.85% for humans. However, they also mentioned that this CAPTCHA was broken by image search with the rate of 11.3% .

Mori et al. proposed a video CAPTCHA [12] with amodal completion, which is one of the abilities of human visual interpolation. Their method utilizes black letters and gray circles that occlude letters. The occluded letter areas are then made transparent. The letters do not move while the gray circles move randomly in every frame of the video. When the circles return to the initial position, users could recognize the CAPTCHA. Users can recognize the CAPTCHA with the success rate of 68%, and the average response time was 13.8 seconds. In their consecutive research, however, their methods including the updated one were broken by CNN [13]. The recognition rate for one letter was 99.9%. They also tried to seek some improved points, such as changing of character color from black to red, but it was also broken by CNN with the same rate of 99.9% [14].

Google provided a checkbox CAPTCHA system called *no captcha reCAPTCHA system*. It analyzes user's clicking behavior with a risk analysis system and calculates the confidence score. If the click is suspicious, then it shows an image-based CAPTCHA test to verify whether or not the click is done by humans. Sivakorn et al. reported that they could break 52,000 to 59,000 checkbox CAPTCHA tests on one day in single IP address, and the image-based CAPTCHA was broken with 70.78% accuracy by CNN [15].

CAPTCHA categories now include text, image, checkbox, and so on. However, the improvement of image recognition techniques, especially CNN, makes attackers decode many existing CAPTCHAs. In this paper, we focus on the text-based CAPTCHA. As shown above, in the recent CAPTCHA attack system, character regions are extracted and machine learning such as CNN is applied to each character. Therefore, for prevention of CAPTCHA from being learned by CNN, it is important not to use features that are easily identified. In this paper, we assume that "randomness" or "irregularity" is useful to prevent attackers from creating CAPTCHA breaking programs.

We propose a new text-based CAPTCHA test that uses amodal perception, which is one of the human vision features [16]. Figure 1 shows the letters *A* that is partially hidden by blue-colored destructors. Because of the blue destructors, computers cannot recognize the letter, while humans can because we can interpolate the hidden parts.

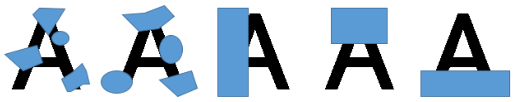


Figure 1: Alphabetic characters showing the amodal effect

We use this feature in our work. For existing text-based CAPTCHA, such as reCAPTCHA v1, Microsoft, and Yahoo! CAPTCHA systems deform the characters so that it makes difficult for spam bots to analyze the text. However, machine learning methods can understand the deformed characters. Instead of deforming characters, our algorithm hides parts of characters by randomly placing random-shaped destructors on and under the characters. The randomness makes it difficult for machine learning or attacker systems to find and recognize the characters. Figure 2 shows our proposed CAPTCHA. It contains four letters with many destructors. Although we can recognize letters as *FPOF*, the destructors make it difficult to recognize the position of letters and their contents.

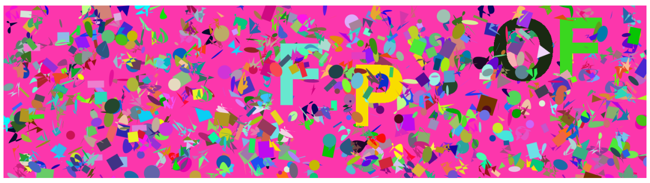


Figure 2: An example of proposed CAPTCHA (*FPOF*)

2 Amodal Effect and Character Recognition

When we place many obstacles on and under the characters, we need to be aware of several points. For example, we recognize the character in the left side of Figure 3 as *B*. However, we cannot identify the letter shown on the right side of Figure 3 because we cannot tell if it is *C* or *O*. Therefore, some parts of a character should not be covered with destructors so that humans can identify it. We call these parts as *non-shielded part*.

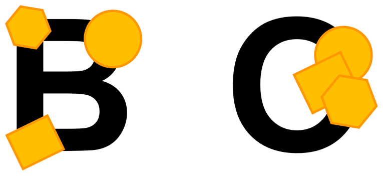
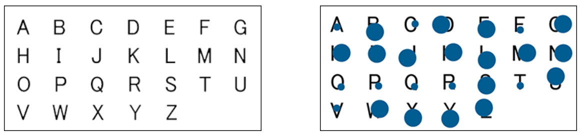


Figure 3: Recognizable (left) and unrecognizable (right) letters with randomly placed objects

We have performed a subject experiment to define non-shielded parts for each letter. 29 examinees, who were 20's, participated in this experiment. They put circular seals with 8, 16, or 20 [mm] in diameter on some spots that should not be hidden to identify the character on a question sheet shown in Figure 4(a). Figure 4(b) shows an example of their answers.



(a) Original question sheet (b) User response

Figure 4: User experiment to search for non-shielded parts

Figure 5 shows the result of the experiment, where red-colored portions are non-shielded parts. This figure is generated by synthesizing all answer sheets with 96% transparency. Because these parts are important for humans to recognize characters, they should not be completely covered. The following section shows how we have used this information to place the obstacles.

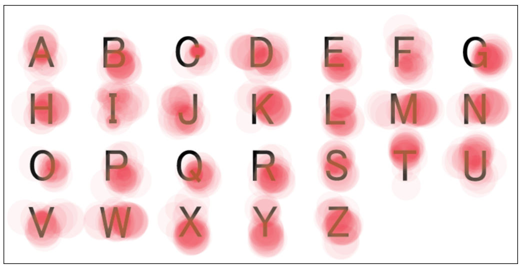


Figure 5: Result of the experiment

3 CAPTCHA Generation

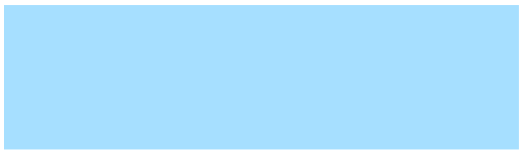
Figure 6 shows an overview of the proposed algorithm. The CAPTCHA has four layers. The first layer contains only the background color (Figure 6(a)). The color is defined randomly. Next, some destructors are placed (Figure 6(b)). The shapes, positions, and colors are determined randomly. Third, alphabets are selected and placed randomly (Figure 6(c)). Finally, destructors are put randomly again (Figure 6(d)). In this section, we explain the detailed algorithm.

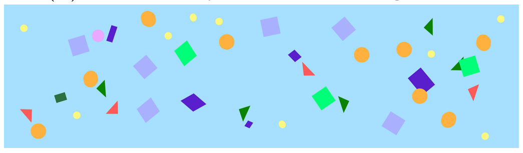
3.1 Calculation of the Shielded Rate

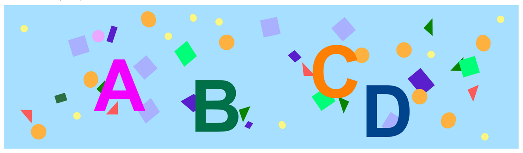
For a simple implementation, the non-shielded parts shown in Figure 5 are approximated as rectangles shown in Figure 7 and Table 1. The origin of the coordinate system is set at the top-left corner of each character. The horizontal and vertical lengths of a character are l_w and l_h, respectively. The non-shielded part should be covered with destructors less than T_1% for human readability.

3.2 Character settings

The RGB values of each character are selected randomly. However, if the character and the background colors are similar, it is difficult to distinguish the character from the background. To avoid this, the contrast should be kept between the character and background. In this paper, the minimum color difference value $\Delta_{76}E$ (CIELAB color difference) between each character and the background color is set to T_2. If


(a) Randomly colored background


(b) Randomly placed destructors


(c) Randomly placed characters

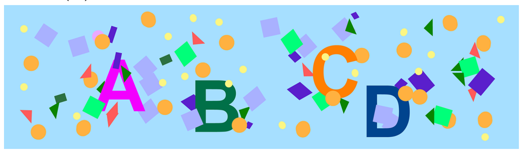

(d) Randomly placed destructors and characters

Figure 6: Algorithm flow

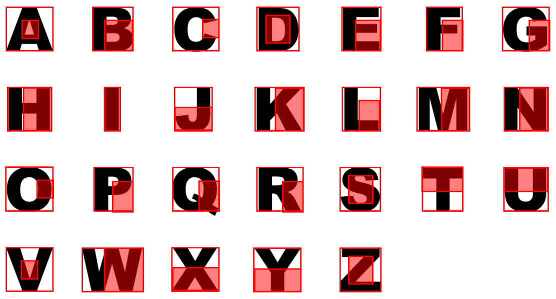


Figure 7: Simplification of non-shielded parts

Table 1: Coordinates of the non-shielded parts depending on letters

Letters	Coordinates	
	Start	End
A/V	$\left(\frac{1}{3}l_w, \frac{1}{3}l_h\right)$	$\left(\frac{2}{3}l_w, \frac{2}{3}l_h\right)$
C/O	$\left(\frac{2}{3}l_w, \frac{1}{3}l_h\right)$	$\left(l_w, \frac{2}{3}l_h\right)$
B/E/F/G/L/P/Q/R	$\left(\frac{1}{3}l_w, \frac{1}{3}l_h\right)$	$\left(l_w, l_h\right)$
H/K/M/N/W	$\left(\frac{1}{3}l_w, 0\right)$	$\left(l_w, l_h\right)$
J/X/Y	$\left(0, \frac{1}{2}l_h\right)$	$\left(l_w, l_h\right)$
T/U	$\left(0, 0\right)$	$\left(l_w, \frac{1}{2}l_h\right)$
D/S/Z	$\left(\frac{1}{4}l_w, \frac{1}{4}l_h\right)$	$\left(\frac{3}{4}l_w, \frac{3}{4}l_h\right)$
I	$(0, 0)$	(l_w, l_h)

the color difference is less than T_2, the character color is randomly re-assigned. The center position (x, y) of each character in the image is also selected randomly so that $x = [l_w/2, w - l_w/2]$ and $y = [l_h/2, h - l_h/2]$, where w and h are the CAPTCHA image width and height, respectively. In order to ensure readability, the algorithm re-allocates the (x, y) value randomly if one of the following cases occurs :

- The character covers more than T_1% area of the non-shielded parts in the other characters
- The x-coordinate of a character surpasses the half of the previous character width

The first condition is for readability to humans, and the second one is for character ordering.

3.3 Destructor Settings

The shapes of destructors are ellipse, rectangle, polygon, and closed region by Bezier curve. We show examples of destructors in Figure 8.

The occurrence probability is 25% for each shape. The length of the rectangle and the radius of the ellipse are determined randomly in the range of [1, T_3]. For polygon or Bezier region, we select 3 to 13 points. The number and the position of the points are selected randomly, and the maximum size of the bounding box is $T_3 \times T_3$. Each component of RGB values for each shape is selected randomly in the range of [0, 255].

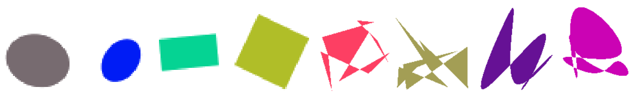


Figure 8: Shapes of the destructors

Destructors are placed before and after drawing the characters. Before we place the characters, we do not need to consider the non-shielded parts because destructors are placed under the texts. After we place the characters, we cannot ignore the non-shielded parts because it is difficult for humans to read the text if destructors are placed on the text. Therefore, we set threshold T_1. If one of the characters is covered more than T_1% of the non-shielded parts, then we stop to place destructors.

4 Evaluation

We have performed readability test on CAPTCHA images for both humans and computers. Table 2 describes the parameter settings to generate CAPTCHA images. The maximum number of destructors, T_1, T_2, and T_3 affects the readability of the CAPTCHA image. Therefore, we set these parameters based on a pre-experimentation, which tests the readability for human.

Table 2: Values of each CAPTCHA parameters

Parameter	**Value**
Image Size	1500 × 400 [px]
Font Size	200 [pt]
Font Family	Arial Bold
Maximum destructors	800
T_1 (explained at 3.1 and 3.2)	50 [%]
T_2 (explained at 3.2)	90
T_3 (explained at 3.3)	40 [px]

Table 3 summarizes the evaluation results.

4.1 Readability by Humans

21 participants reviewed 60 CAPTCHA images with four characters for each image. Figure 9 shows the result of the examination. The blue bars show the recognition rate [%] for one image, and the red line shows the average response time [seconds]. The average recognition rate and the average response time were 94.2% and 5.41 seconds per image, respectively.

Table 3: Rates of CAPTCHA image readability

Reader	**Recognition Rate [%]**
Humans	94.2
OCR software	0
CAPTCHA solver	0
LeNet	54.95 (one letter)
	9.11 (estimated for four letters)
	2.75 (estimated for six letters)
RCN	13.07 (one letter)
	0.029 (estimated for four letters)

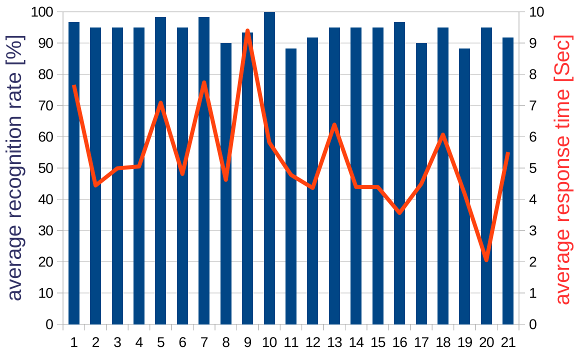


Figure 9: Result of human Recognition

On the other hand, Bursztein et al.[17], asked Amazon's Mechanical Turk to solve CAPTCHAs created by the 13 most popular text-based CAPTCHA schemes, including Google, Yahoo, and Microsoft CAPTCHA systems. The average recognition rate and the time were 87.0% and 9.8 seconds, respectively. Therefore, we have concluded that our CAPTCHA is easy for users to recognize compared with other text-based CAPTCHA systems.

Figure 10 shows CAPTCHA images that have lower recognition rates for humans. Recognition rates of Figure 10 were 4.8% for the letters of *WOUD*, 19.1% for the letters of *IUOC*, and 61.9% for the letters of *MEHQ*, respectively. All participants who answered incorrectly recognized O as Q and vice versa because the characters or destructors covered the lower right part, which is the non-shield part of Q. Then, the readabilities of O and Q were reduced.

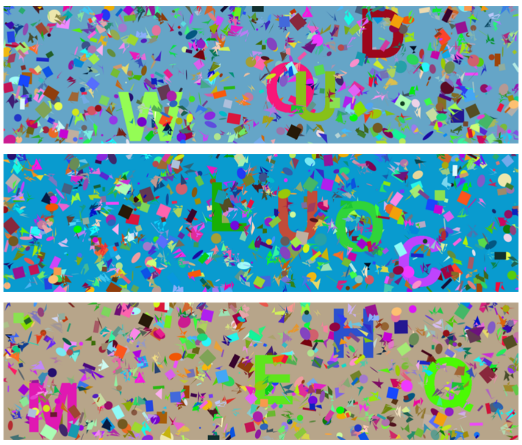
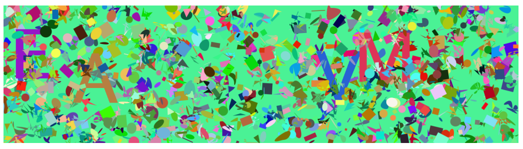
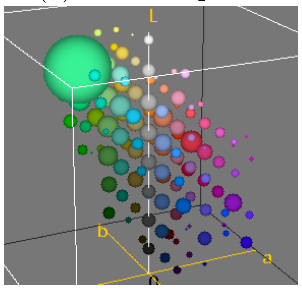
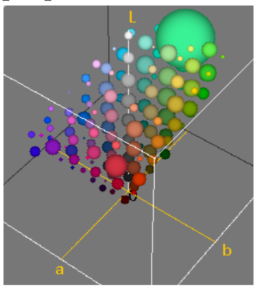


Figure 10: CAPTCHA images with lower recognition rates:WOUD (top), IUOC (middle), MEHQ (bottom)


(a) An example of the proposed CAPTCHA



(b) color histogram



(c) color histogram from another viewing angle

Figure 11: Color histogram of CAPTCHA

4.2 Readability by Conventional Software

We have used a general OCR software called *Tesseract* [18] for the computer readability test, which had 1,000 CAPTCHA images with four characters for each image. The result shows that Tesseract did not recognize any of the CAPTCHA images because destructors were placed in various positions. We have also tested CAPTCHA solver software [19]. Based on 100 CAPTCHA images, we have found that none of them were recognized.

4.3 Possibility of Specified Attacker

CAPTCHA solver systems usually estimate the CAPTCHA region, divide it into each character, and recognize each character [6, 7, 20, 21]. In this section, we discuss possibilities for each process.

4.3.1 Estimation of character position

If the destructors are placed only under the characters, then the characters are placed on the top, and the text region could be estimated with T junction [22]. However, if the destructors are placed both *on* and *under* the characters, it is difficult to clarify the character position because the destructors are placed even *under* the character in some area, while others are placed *on* the character.

Another possibility is to use a color histogram. Some color bins of the characters might have a large values that lead to the estimation of the character region. Figure 11(a) shows an example of the proposed CAPTCHA, and (b) and (c) show the visual histograms (in CIELAB) in which the color of sphere shows the representative color of each histogram bin, and the radius of the sphere corresponds to the value of the bin. Although the color of the background shows the largest value, we cannot recognize which bins indicate the characters. While other CAPTCHAs have regularities to segment image colors into characters, our algorithm uses *randomness* to prevent attackers. Therefore, it is difficult to identify the character region with image processing techniques.

4.3.2 Recognition of Character with Machine Learning

We have used the experimental setting as George et al. [10]. Their training data were five characters for each alphabet and achieved the character recognition rate of 94.3% for reCAPTCHA system, 91.6% for Botdetect system, 92.5% for Yahoo! CAPTCHA system, and 89.3% for PayPal CAPTCHA system, respectively. Under the same condition as their work, the accuracy rate for recognition of one character was 11.54% for the proposed CAPTCHA system. Because the

CAPTCHA has four characters in an image, the recognition rate for one CAPTCHA image is 0.018%.

We have also used the same experimental setting as Hong et al. [6], who used LeNet to break the Microsoft CAPTCHA service. They used 600 CAPTCHA images. Their method automatically estimated the character region, from which they obtained 3,600 character regions. These were used as the input data, 90% of which were used for the training, and the remaining were used for the test. Hong et al. achieved 91.07% character recognition rate. Because the Microsoft CAPTCHA system contains six characters per image, their method can break a CAPTCHA image with 57.07% success rate. For the proposed CAPTCHA image, we cannot employ their automatic character region estimation algorithm, because it is specialized in the Microsoft CAPTCHA generation algorithm. Therefore, we manually separated 3,600 character regions and then used LeNet to investigate the recognition rate. The recognition rate for one character was 54.95%, which shows that the proposed CAPTCHA is more robust against the CNN attack. The recognition rates become 9.11% and 2.75% for CAPTCHA images with four and six characters, respectively, where Microsoft CAPTCHA image has six characters.

Although the recognition rate is not 0%, it is low compared with the conventional methods. Therefore, it is possible to distinguish between humans and bots by preparing a mechanism that excludes access from IP addresses with low recognition rate.

5 Conclusion

In this paper, we have proposed a new generation method of text-based CAPTCHA by using human perception. It lowers the possibility of recognition by computer software, while it is easily recognized by humans. As the result of the evaluation, humans could read the CAPTCHAs correctly at the rate of 94.2%. The average response time to read the CAPTCHA was 5.41 seconds, meaning that it is easy for humans to recognize the CAPTCHA generated by the proposed method. A conventional OCR software and the CAPTCHA solver system were unable to recognize the generated CAPTCHA at all. CNN could recognize the CAPTCHA with low accuracy. Therefore, we can distinguish bots and humans.

In the future, we would like to examine reallocation of the non-shielded parts of the letter Q for higher read-ability. We also plan to use various fonts to prevent breaking CAPTCHA by pattern matching. Moreover, we should optimize the CAPTCHA generation parameter such as the number of maximum destructor, T_1, T_2, and T_3. These affects the readability for both human and CNN. We should perform more detailed parameter tuning. Furthermore, we should carefully determine the colors of letters and destructors to improve the readability by color-blind persons.

References

[1] Mollom, `https://www.mollom.com/`, (referred on 2018.2).

[2] Akismet, `https://ja.wordpress.org/plugins/akismet/`, (referred on 2018.2).

[3] The Official CAPTCHA Site, `http://www.captcha.net/` (referred on 2017).

[4] Drupal Project usage overview, `https://www.drupal.org/project/usage`, (referred on 2018.2).

[5] G.Mori and J.Malik, “Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA”, proceedings of CVPR, pp. 134-141 (2003).

[6] C. Hong, B. Lopez-Pineda, K. Rajendran, and A. Recasens, “Breaking Microsoft’s CAPTCHA”, Technical Reports (2015).

[7] H. Gao, M. Tang, Y. Liu, P. Zhang, and X. Liu, “Research on the Security of Microsoft’s Two-Layer Captcha”, IEEE Transactions on Information Forensics and Security, Vol.12, No.7, pp. 1671–1685 (2017).

[8] O.Starostenko, C. Cruz-Perez, F. Uceda-Ponga, and V. Alarcon-Aquino, "Breaking text-based CAPTCHAs with Variable Word and Character Orientation", Pattern Recognition, Vol. 48, Issue 4, pp. 1101–1112 (2015).

[9] T.Komiyama, T.Umezawa and N.Osawa, "Robustness of CAPTCHA based on Subjective Contour", Forum on Information Technology, Vol. 14, No.4, pp. 191–192 (2015).

[10] D.George, W.Lehrach, K.Kansky, M.Lazaro-Gredilla, C.Laan, B.Marthi, X.Lou, Z.Meng, Y.Liu, H.Wang, A.Lavin and D.S.Phoenix, "A Generative Vision Model that Trains with High Data Efficiency and Breaks Text-based CAPTCHA", Science, Vol.358, Issue 6368, eaag2612 (2017).

[11] T.Tamura, S.Kubota, K.Aburada, T.Katayama, M.Park and N.Okazaki, "A Proposal of an Image-based CAPTCHA using Random Obstruction Figures to Absorb OCR-based Bot-attacks", IPSJ Journal, Vol.56, No.3, pp. 808–818 (2015).

[12] T.Mori, R.Uda and M.Kikuchi, "Proposal of Movie CAPTCHA Method Using Amodal Completion", IEEE/IPSJ International Symposium on Applications and Internet, pp. 11–18 (2012).

[13] T.Azakami, C.Shibata and R.Uda, "Challenge of Deep Learning against CAPTCHA with Amodal Completion and Aftereffects by Colors", International Conference on Network-Based Information Systems, pp. 127–132 (2016).

[14] T.Azakami and R.Uda, "Deep Learning Analysis of Amodal Completion CAPTCHA with Colors and Hidden Positions", International Conference on Advanced Information Networking and Applications Workshops, pp. 341–346 (2017).

[15] S. Sivakorn, J. Polakis, and A. D. Keromytis, "I'm not a human: Breaking the Google reCAPTCHA", Black Hat ASIA 2016 (2016).

[16] S.Lehar, "Gestalt Isomorphism and the Quantification of Spatial Perception", Gestalt Theory, Vol.21, No.2, pp. 122–139 (1999).

[17] E. Bursztein, S. Bethard, C. Fabry, J. C. Mitchell, and D. Jurafsky, "How Good Are Humans at Solving CAPTCHAs? A Large Scale Evaluation", IEEE Symposium on Security and Privacy, pp. 399–413 (2010).

[18] Tesseract-OCR, `https://code.google.com/p/tesseract-ocr/` (referred on 2017).

[19] GSA Captcha Breaker, `http://gsa-captcha-breaker.soft112.com` (referred on 2017)

[20] H. Gao, X. Wang, F. Cao, Z. Zhang, L. Lei, J. Qi, and X. Liu, "Robustness of Text-based Completely Automated Public Turing Test to Tell Computers and Humans Apart", IET Journals on Information Security, Vol.10, Issue 1, pp. 45–52 (2016).

[21] R. Hussain, H. Gao, R.A. Shaikh, and S.P. Soomro, "Recognition based Segmentation of Connected Characters in Text based CAPTCHAs", IEEE Conference on Communication software and networks, pp. 673–676 (2016).

[22] T. Lindeberg, "Feature Detection with Automatic Scale Selection", International Journal of Computer Vision, Vol.30, Issue 2, pp. 79–116 (1998).

Youngha Chang

Youngha Chang is a Associate Professor of Tokyo City University. She received her B.E. from Ewha Woman's University in 1998. She also earned Ph.D from Tokyo Institute of Technology in 2004.

She became a researcher, a research associate, and an assistant professor at Tokyo Institute of Technology in 2004, 2006, and 2007, respectively. She joined Tokyo City University from 2012. Her research interests are image processing and color science. She is a member of SAS and IPSJ.

Akinori Urayama

Akinori URAYAMA was a graduate student at Tokyo City University and graduated in 2017. His research interest was image engineering.

Miho Watanabe

Miho WATANABE was a undergraduate student at Tokyo City University and graduated in 2014. Her research interest was image engineering.

Nobuhiko Mukai

Nobuhiko Mukai is a professor of Tokyo City University. He received his B.E., M.E. and Ph.D degrees from Osaka University in 1983, 1985, and 2001, respectively. He started to work at Mitsubishi Electric Corporation and changed to work as an associate professor for Musashi Institute of Technology in 2002. He is currently a professor of Tokyo City University from 2007. His research interests are computer graphics and image processing. He is a member of ACM, SAS, VRSJ, IEICE, ITE, IPSJ, IIEEJ, and JSUM.