Received March 25, 2014, accepted April 3, 2014, date of publication April 29, 2014, date of current version May 21, 2014.

Digital Object Identifier 10.1109/ACCESS.2014.2321001

*fg*CAPTCHA: Genetically Optimized Face Image CAPTCHA

BRIAN M. POWELL[1], (Member, IEEE), GAURAV GOSWAMI[2], (Student Member, IEEE), MAYANK VATSA[2], (Member, IEEE), RICHA SINGH[2], (Member, IEEE), AND AFZEL NOORE[1], (Senior Member, IEEE)

[1]Lane Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, WV 26506-6109 USA
[2]Indraprastha Institute of Information Technology, New Delhi 110078, India

Corresponding author: M. Vatsa (mayank@iiitd.ac.in)

ABSTRACT The increasing use of smartphones, tablets, and other mobile devices poses a significant challenge in providing effective online security. CAPTCHAs, tests for distinguishing human and computer users, have traditionally been popular; however, they face particular difficulties in a modern mobile environment because most of them rely on keyboard input and have language dependencies. This paper proposes a novel image-based CAPTCHA that combines the touch-based input methods favored by mobile devices with genetically optimized face detection tests to provide a solution that is simple for humans to solve, ready for worldwide use, and provides a high level of security by being resilient to automated computer attacks. In extensive testing involving over 2600 users and 40 000 CAPTCHA tests, *fg*CAPTCHA demonstrates a very high human success rate while ensuring a 0% attack rate using three well-known face detection algorithms.

INDEX TERMS Mobile security, web security, CAPTCHA, face detection.

I. INTRODUCTION

Due to recent developments in technology, users are rapidly adopting smartphones, tablets, and other non-traditional smart computing devices in lieu of desktop and laptop computers. Traditional input devices such as keyboards and mice are being replaced by more interactive touchscreen technology. With advanced mobile devices, users can easily access Internet services such as online shopping and e-banking. These large-scale applications require improved interfaces (including security systems) designed to easily serve the growing mobile market [1].

Presently, a number of techniques provide device-level security to protect users in case of loss or theft of their mobile device. Solutions based on typing such as passwords and PIN codes dominate, but newer mobile-friendly techniques such as picture puzzles [2], tracing patterns [3], and biometrics features including touch pattern analysis [4], fingerprints [5], and facial images [6] are gaining popularity and acceptance. While many online service providers have completely redesigned their website portals or maintain special mobile versions of their websites, relatively little progress has been made with similar redesigns of application-layer security tools [7] to protect the online resources which mobile users access.

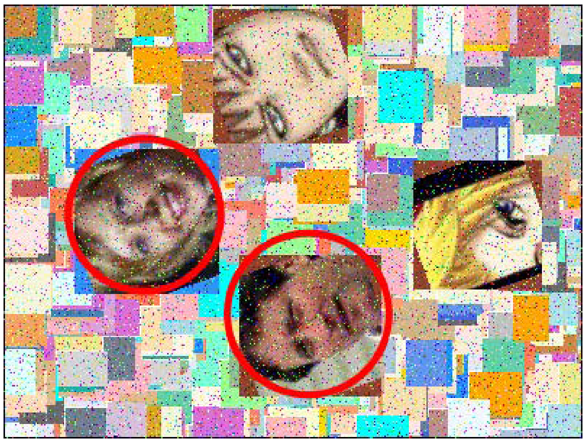


FIGURE 1. Example of a *fg*CAPTCHA image with correct selections, the human faces, circled.

CAPTCHA (Completely Automated Public Turing Test to Tell Computers and Humans Apart) is one major example of a security tool that is not yet mobile user-friendly. CAPTCHAs are designed to prevent automated attacks by requiring users to perform tasks that are relatively easy for humans but challenging for computers (automated algorithms) [8]. They have become ubiquitous in situations where websites want to prevent e-mail, instant messaging, and text message spam. CAPTCHAs provide an additional layer of security and are frequently paired with account login systems to prevent brute force password attacks [9]. Existing CAPTCHA implementations generally belong to one of three categories: (1) text-based, (2) image-based, or (3) video and audio-based. Some popular examples of each are shown in Table 1.

Most existing CAPTCHAs are text-based. The user is presented with visually distorted text and asked to type it in correctly to prove he or she is a human and not a computer algorithm masquerading as a person. Many mobile devices lack a physical keyboard, which makes text-based input cumbersome and error-prone [10]. Further, most text-based CAPTCHAs are (English) language-dependent and not suitable for multilingual worldwide usage. This paper mitigates the shortcomings of existing approaches and proposes a new CAPTCHA, termed as *fg*CAPTCHA, which leverages touchscreen technology in mobile devices to make CAPTCHAs user-friendly and intuitive. *fg*CAPTCHA presents users with a composite image containing several visually distorted human faces along with other objects and non-real faces embedded in a complex background pattern. To prove that a user is human, users must solve the CAPTCHA by correctly selecting only the real human faces without choosing any other objects or non-real face images. If this is successfully done, the user is considered to be human and granted access to the secured resource. Fig. 1 shows an example of how a *fg*CAPTCHA test can be correctly solved. In most cases, solving an instance only requires two or three taps from the user, making it extremely quick to complete and mobile device-friendly.

Key contributions of this research include:

1) Design of an interactive non-keyboard-based (touchscreen-compatible) image CAPTCHA to facilitate easy use on mobile devices.
2) Generation of computationally-challenging face detection CAPTCHA tests to provide enhanced security.
3) Utilization of genetic learning algorithms to optimize CAPTCHA parameters for better human performance and drastically lower the attack success rates of computer algorithms.
4) Development of large-scale human and automated testing processes to evaluate performance of the proposed image-based face detection CAPTCHA.

II. PROPOSED APPROACH

To address the usability shortcomings of existing implementations, this paper proposes *fg*CAPTCHA, a new image-based CAPTCHA that uses face detection as the test. This approach leverages the fact that humans are adept at recognizing faces but this task can be challenging for computers when distortions are applied. The proposed approach is primarily developed for the touch-based input methods of mobile devices but is also compatible with point-and-click techniques of traditional desktop and laptop computers. *fg*CAPTCHA is suitable for multilingual applications unlike many existing CAPTCHAs that are language-dependent. The proposed approach combines three distinct elements:

1) A set of embedded images, some of which are photographs of real human faces and others which are cartoons, sketches, or photos of animals representing face-like images to make correctly detecting human faces challenging for computers.
2) A complex background pattern designed to confuse the automatic face detection software, thereby increasing the false positive detection rate.
3) A set of visual distortion types (e.g., blurring, contrast adjustment) and the amount of distortion to apply, referred to as its intensity.

The generation process can be represented as,

$$C = f(n_{min}, n_{max}, width, height, \phi, I_{face}, I_{nonface}) \tag{1}$$

where function f creates a new CAPTCHA of dimensions *width*-by-*height* pixels, containing a total of between n_{min} and n_{max} embedded images taken from sets I_{face} and $I_{nonface}$. Distortion settings (distortion types and distortion intensities) selected from ϕ are applied to the rendered composite, yielding CAPTCHA C. The goal of the proposed CAPTCHA generation approach is to find distortion settings which maximize the chance that humans will be able to solve the CAPTCHA while minimizing the likelihood of successful automated attacks by computer algorithms. This can be shown as,

$$\text{argmax}_{\varphi}\, P(C_{\varphi}) = P_H(C_{\varphi}) - P_A(C_{\varphi}) \tag{2}$$

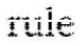
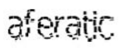
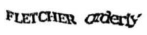
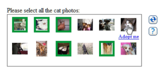
where C_{φ} is a CAPTCHA with distortion settings φ applied, P_H is the likelihood humans can solve the CAPTCHA, P_A is the likelihood automated attacks can solve the CAPTCHA, and P is the difference between the two likelihoods. Without including humans in the loop during the CAPTCHA generation process, it is impossible to know the actual values of P_H and P for a given CAPTCHA. Instead, a simulation process is used to model human performance. The results of the simulation are used to calculate a fitness value for a generated CAPTCHA such that,

$$F\left(C_{\varphi}\right) = S_H\left(C_{\varphi}\right) - S_A\left(C_{\varphi}\right) \tag{3}$$

where S_H is the simulated likelihood of human success, S_A is the likelihood of a successful automated attack, and fitness value F is the difference between the two likelihoods. Higher values of F indicate a CAPTCHA where it should be relatively easier for humans to detect the faces while being more difficult for computer-based automated attacks to successfully complete the face detection task. These attacks can be modeled by performing automated face detection and comparing detected face locations against known face locations. The proposed approach uses the Viola-Jones algorithm [34] to locate embedded faces. This algorithm works by calculating the integral image, the sum of all pixel values to the left and above a given point, as shown by,

$$ii(a, b) = \sum_{a' \leq a, b' \leq b} i(a', b') \tag{4}$$

TABLE 1. Summary of selected existing CAPTCHAs.

CAPTCHA	Modality	Mode of Operation	Human Accuracy	Attack Accuracy	Sample
AltaVista [11], [12]	Text: 8 random characters	Each character is rendered in a different font, ransom-note style. Different rotations and distortions are applied to each letter.	-	Reduced page accesses by "over 95%"	
EZ-GIMPY [11], [13]	Text: 1 English word	Word is randomly distorted by adding white lines and deformations.	-	92%	
ScatterType [14], [15], [16]	Text: English word-like non-dictionary string of 6-8 characters	Characters are segmented into many pieces then systematically scattered.	Up to 95%	-	
BaffleText [17]	Text: Pronounceable English non-word of 5-8 characters	Text overlaid with random geometric shapes, difference masking applied.	89%	25%	
MSN [18], [19]	Text: 8 random characters	Characters are distorted and rotated, then arcs added to visually connect characters.	-	over 90%	
Handwritten [20]	Text: Full name of a city	Uses images of handwritten city names taken from U.S. mail.	100%	4-9%	
reCAPTCHA [21], [22], [23]	Text: 2 English words	Scanned words that failed OCR presented side-by-side. Some additional distortions may be applied.	-	30%	
ESP-PIX [24]	Images: 4 images	User selects category to describe images from predefined list.	-	High random guess rate	
Asirra [25], [26]	Images: 12 images of cats and dogs	User identifies cats or dogs.	High	82.7%	
Scene Tagging [27]	Images: Small number of images on background image	User answers questions relating to number of images, placement, or relationship to each other.	96.6%	2.6%	
MosaHIP [28]	Images: Collage of many images	User drags descriptor labels on top of images they represent.	98%	4.1%	
IMAGINATION [29], [30]	Images: Collage of images	User clicks on center of one image then categorizes that image.	70%	4.95%	
Digg Audio [31]	Audio: Audio recording of random letters and numbers	User enters information from audio.	-	71%	
Video [32], [33]	Video: Flash video	User types three words describing video.	90%	13%	

Here, a, b are points, $i(a, b)$ is the original image, and $ii(a, b)$ is the corresponding integral image [34]. Using the integral image, a series of Haar-like rectangular features are computed across the image. The rectangular features are run through a cascade of classifiers to determine the probable locations of embedded faces [34], [35]. The face locations

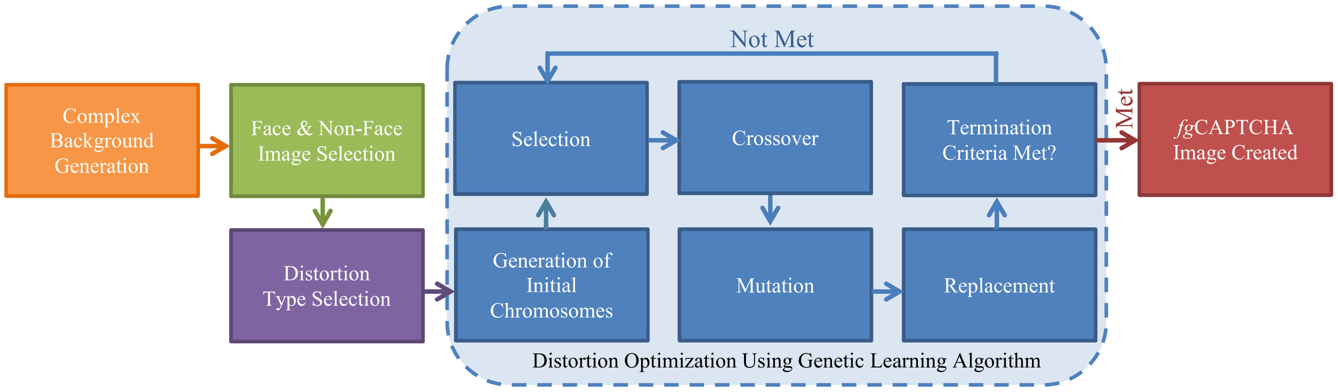

Complex Background Generation
Face & Non-Face Image Selection
Distortion Type Selection
Not Met
Selection
Crossover
Termination Criteria Met?
Met
*fg*CAPTCHA Image Created
Generation of Initial Chromosomes
Mutation
Replacement
Distortion Optimization Using Genetic Learning Algorithm

FIGURE 2. **Steps involved in generation of a *fg*CAPTCHA image.**

indicated by the Viola-Jones face detector are compared against the actual embedded human face locations, with the automated attack rate being the percentage of embedded human faces that are found. Lower values resulting from (5) are better,

$$S_A\left(C_\varphi\right) = \frac{d_{correct} - d_{false}}{n} \leq 1.0 \tag{5}$$

where n is the number of embedded human faces in CAPTCHA image C with distortion φ applied, $d_{correct}$ is the number of human faces correctly detected by the algorithm, and d_{false} is the number of false human face detections.

Since there is no feasible direct way of simulating human performance, the proposed CAPTCHA indirectly models human success rates using image quality metrics. Structural Similarity (SSIM), a metric designed to mimic the human visual system, compares distorted and undistorted versions of embedded images to look for differences in linear correlation, luminance, and contrast [36]. Values closer to 1.0 signify that the tested images are more similar, and hopefully, the distorted version will be relatively easier for humans to solve. SSIM is represented as,

$$SSIM\left(x,y\right) = \frac{\left(2\mu_x\mu_y + C_1\right)\left(2\sigma_{xy} + C_2\right)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_x^2 + \sigma_y^2 + C_2\right)} \leq 1.0 \tag{6}$$

where, μ_x and μ_y are the mean of images x and y; σ_x^2 and σ_y^2 are the variance of x and y; and σ_{xy} is the covariance of x, y. $C_1 = (k_1L)^2$, $C_2 = (k_2L)^2$ stabilize the denominator as it approaches zero, with k_1, k_2 being generic constants and L being the dynamic range of pixel values [36]. To model the human performance, SSIM is performed on each embedded image. The human success rate is an average of all SSIM values,

$$S_H\left(C_\varphi\right) = \frac{\sum_{j=0}^{n} SSIM\left(C_{j\varphi}\right)}{n} \leq 1.0 \tag{7}$$

where n is the number of embedded human faces in CAPTCHA image C and $SSIM\left(C_{j\varphi}\right)$ is the resulting SSIM value when distortion settings φ are applied to embedded image j.

As shown in Fig. 2, the generation of *fg*CAPTCHA images involves several distinct phases: complex background generation, face and non-face image selection, distortion type selection, and distortion optimization. Through the use of a genetic learning algorithm, the resulting CAPTCHA incorporates distortion types and distortion intensity levels such that humans can solve the CAPTCHA with ease but computers cannot.

A. BACKGROUND GENERATION

Creation of a new *fg*CAPTCHA image begins with the generation of a 400 × 300 pixel background composed of many overlapping rectangles in various colors and sizes. This size is chosen as it can be displayed at its full native resolution on common mobile devices, avoiding potential issues related to scrolling or downscaling. The individual colored rectangles have their colors chosen at random from a list of 56 common hues including skin tones. Height and width are based on a fraction of the overall image size, randomly scaled, such that,

$$s = \left\{\frac{r}{10}\min(height, width)\middle|0.75 \leq r \leq 1.25\right\} \tag{8}$$

where, s is the resulting size in pixels for one side of the rectangle, *height* and *width* are the overall height and width of the background, and r is a random real-valued scaling factor. Colored rectangles are scattered across the entire background until at least 95% is covered. This provides a complex pattern to interfere with the rectangular features used by the Viola-Jones detector and other similar face detection algorithms. The random sizes and colors make it difficult to isolate embedded images, and in some cases, lead algorithms to falsely detect faces in the background.

B. IMAGE SELECTION

Once the background is generated, a total of 4 to 5 face and non-face images are selected to be embedded such that,

$$n_{total} = \left\{n_{face} + n_{nonface}\middle| n_{face} \geq 2, n_{nonface} \geq 1, n_{total} = \{4,5\}\right\} \tag{9}$$

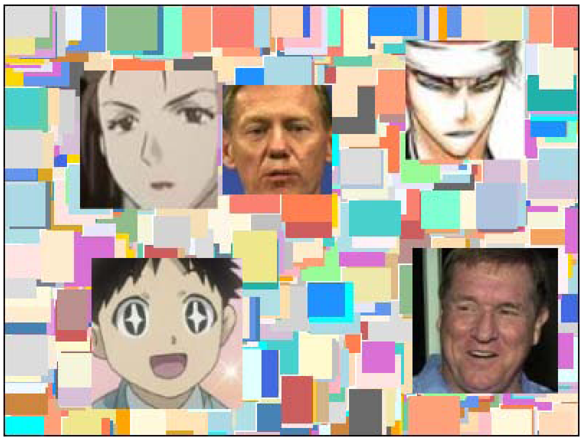


FIGURE 3. Example of a new undistorted CAPTCHA.

Here n_{face}, $n_{nonface}$, and n_{total} represent the number of embedded face, non-face, and total images, respectively. At least two face images are present to prevent a single guess from successfully solving the CAPTCHA. At least one image is a non-face image to provide a false target in case attackers can detect the location of embedded images. Each embedded image is scaled to approximately 100 × 100 pixels prior to placement. This size is chosen to correspond with the area covered by a fingertip for accurate use on touchscreen devices. The images are placed at randomly selected coordinates within the background, ensuring that the images do not overlap with each other or the outside boundary of the CAPTCHA. An example of an undistorted CAPTCHA showing the background and placed images appears in Fig. 3.

TABLE 2. Distortion types.

Distortion Type	Class	Applied	Parameters Adjusted	Intensity Range
Erosion	Degradation	Globally	Pixel radius	3.0-3.5
Increase Brightness	Degradation	Globally	Scale histogram range	0.30-0.45
Resolution Modification	Degradation	Locally	Scale factor	1:3.5-1:5
Width Scaling	Geometric	Locally	Scale factor	1.5-4.0
Height Scaling	Geometric	Locally	Scale factor	1.5-3.0
Rotation	Geometric	Locally	Degrees of rotation	60-180
Piecewise Scaling	Geometric	Locally	Scale factor	1.5-3.5
Periodic Noise	Noise	Globally	% of image removed	5-7.5
Salt-and-Pepper Noise	Noise	Globally	% of pixels to replace	10-20
Speckle Noise	Noise	Locally	Variance	2-5

C. DISTORTION SELECTION

In the proposed approach, the distortions applied to a CAPTCHA have a significant impact on human and automated attack success rates. During design, 10 distortion types have been identified that yield the best performance. Each distortion type has a range of possible intensities adjusted by various parameters as shown in Table 2.

In this step, the various distortion types are compared to find the types which provide the best performance when applied to the intended CAPTCHA. Each distortion type is applied at eight different intensities evenly spread over its range. Performance or fitness values are calculated for each of the resulting images using (3). The results are ranked by their fitness, with those distortion types yielding the top 50% most-fit CAPTCHAs selected for further use. A Cartesian product is created combining two each of the best-fit distortion types. Previous experience has shown that applying two distortion types to each CAPTCHA represents a good balance between making images too simple for automated attacks (one distortion type) or too hard for human users (three or more distortion types). Some distortion type pairs known to perform poorly are discarded, with the rest continued to the next distortion optimization step.

These distortions are classified into three categories: geometric, noise-based, and degradation distortions. Geometric distortions alter the shape, size, or position of embedded images. Width scaling makes an image narrower, whereas height scaling makes an image shorter. Piecewise scaling leaves the overall dimensions of the image untouched but changes the relative proportions of sections of the image. For example, the left half of an image might be compressed so it takes 50% less space than before while the right half is stretched to fill the available space. The rotation distortion rotates the image around a center axis. Any portion of the image falling outside the original dimensions of the image is removed. Rotation can be performed using the transformation matrix,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{10}$$

Here (x, y) are the original coordinates of a pixel, φ is the degree of rotation to be applied in radians, and (x', y') are the adjusted coordinates of the pixel.

Noise-based distortions add interference that is not present in the original image. Salt-and-pepper noise changes the values of the specified percentage of pixels to the maximum or minimum possible value, having the effect of adding randomly discolored pixels to the overall CAPTCHA. Speckle noise modifies the values of individual pixels in a pattern that is uniformly distributed with a mean of 0 and a variance specified by the distortion intensity. Periodic noise creates a repeating pattern of darkened bars across the entire image. It can be generated by,

$$v_d(x, y) = \max\left(0, \min\left(\frac{v(x, y) + (\sin(\frac{y+1}{\varphi}) * 255)}{2}, 255\right)\right) \tag{11}$$

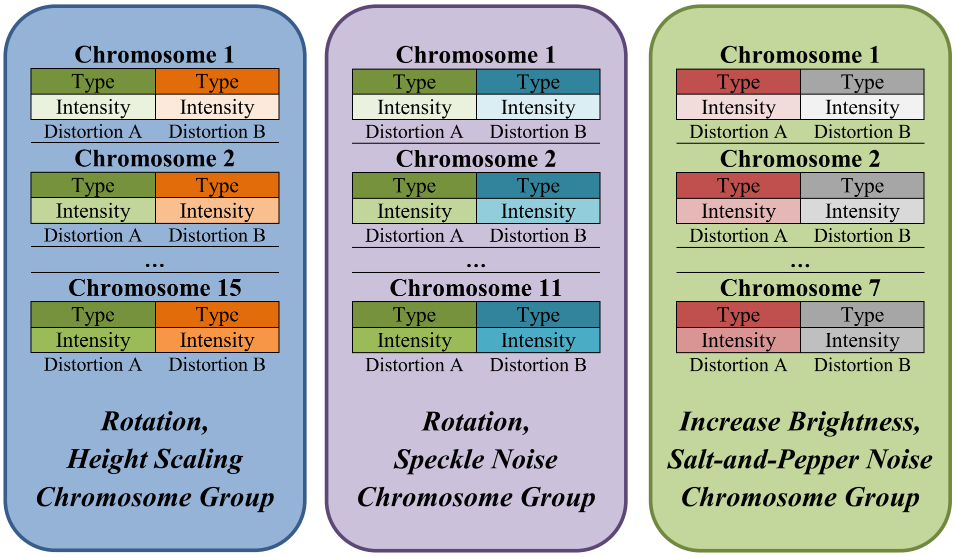

Chromosome 1
Type
Type
Intensity
Intensity
Distortion A
Distortion B
Chromosome 2
...
Chromosome 15
Rotation, Height Scaling Chromosome Group
Chromosome 11
Rotation, Speckle Noise Chromosome Group
Chromosome 7
Increase Brightness, Salt-and-Pepper Noise Chromosome Group

FIGURE 4. Example of chromosome groups.

where $v(x, y)$ is the original pixel value, a number between 0 and 255, at coordinates (x, y). $v_d(x, y)$ is the distorted pixel value and φ is the distortion intensity.

Degradation distortions are designed to reduce detail or contrast, making it difficult to distinguish embedded images. The increase-brightness distortion increases the luminance of each pixel by a specified percentage, effectively reducing the contrast of a CAPTCHA. Erosion works on the entire CAPTCHA in successive blocks. It compares the values of each pixel with those of its neighbors and eliminates unique values, reducing fine detail.

Resolution reduction is performed as a pair of bilinear resizing operations, the first reducing the size of the image and the second expanding it to its original size. As pixel data is lost, this yields a blocky-looking image. The bilinear resizing operation can be represented using,

$$v(x', y') = ax' + by' + cx'y' + d \tag{12}$$

where $v(x', y')$ is the pixel value of coordinates (x', y') and coefficients a, b, c, d can be solved using four equations in four unknowns for the four neighbors of (x', y') [37].

D. DISTORTION OPTIMIZATION

Once the distortion type pairs have been determined, optimal intensities for each distortion must be found. This is a complex problem with a huge search space; therefore, brute force exploration is not feasible. *fg*CAPTCHA instead uses a genetic learning algorithm (GA) to efficiently identify optimal distortion settings. Genetic algorithms are modeled on the biological process of evolution [38]. GAs work by producing successive generations of candidate solutions, referred to as chromosomes, to find the distortion settings which generate the optimized CAPTCHA. The algorithm includes several steps (input parameters are summarized in Table 3) as described below.

Step 1: Generate Initial Chromosomes - The algorithm begins by generating an initial set of 150 chromosomes, each representing one possible combination of distortion settings. The chromosomes contain two genes, each encoding a single distortion type and its associated real-valued intensity. Distortion types are selected from the list of approved distortion type pairs and their intensities are randomly set to a value within the distortion type's specified range. After the chromosomes are generated, a fitness value is calculated for each using (3). Since each distortion type has a distinct range of intensities, genetic algorithm operations such as crossover must be performed only between chromosomes with the same distortion types. Thus, the chromosomes are organized into groups based on their distortion types as shown in Fig. 4. To ensure genetic diversity within each group, a minimum of two chromosomes per group is maintained.

Step 2: Select Candidates for Next Generation - A roulette wheel-based process is used to select the chromosomes to create the next generation. The process selects chromosomes at a rate proportional to their fitness:

$$p_i = \frac{\alpha_i}{\sum_{i=0}^{n} \alpha_i} \tag{13}$$

where n is the total number of chromosomes, α_i is the fitness of chromosome i, and p_i is the probability chromosome i will be selected [39]. Roulette wheel selection works by first summing the fitness values of all chromosomes, yielding T. Then, for each chromosome, a random value λ, between 0 and T is selected. The list of chromosomes is iterated through, adding their fitness values, until the sum is greater than or equal to λ. The chromosome whose fitness value brings the sum over λ is selected to create the next generation [39].

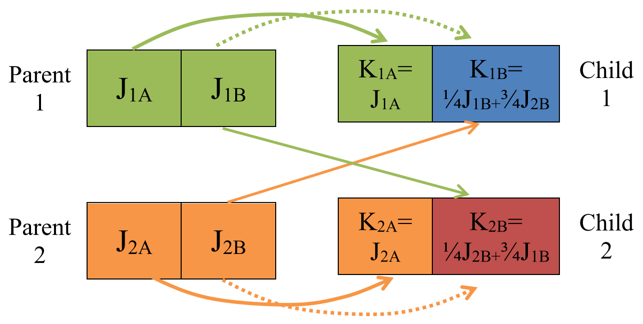


FIGURE 5. Demonstration of crossover process between parent chromosomes J_1 and J_2 to create child chromosomes K_1 and K_2.

Step 3: Perform Crossover - In the crossover step, the values from two parent chromosomes are used to produce two child chromosomes. Approximately 80% of parent chromosomes are randomly selected to participate in this process. A variation on single-point crossover, shown in Fig. 5, is used to accommodate the real-valued distortion intensities stored in the genes. As with single-point crossover, prior to the crossover point, child genes K_1, K_2 inherit directly from their parents J_1, J_2 such that for gene X, $K_{1X} = J_{1X}$ and $K_{2X} = J_{2X}$. After the crossover, a weighted combination of the two parents is used to simulate the value changes that would occur with a binary string representation in traditional single-point crossover. Here, $K_{1X} = \frac{1}{4}J_{1X} + \frac{3}{4}J_{2X}$, $K_{2X} = \frac{1}{4}J_{2X} + \frac{3}{4}J_{1X}$.

Step 4: Conduct Mutation - To prevent stagnation of results at local optima, mutation is applied to approximately 5% of gene values. This helps to ensure the entire solution space is searched rather than just values near those of the parent chromosomes. The traditional mutation approach of randomly flipping bits in a binary-encoded gene value does not work with real-valued genes. Instead, the existing gene value is averaged with a new random value when mutation is performed,

$$m = \left\{ \frac{c+n}{2} \middle| dist_{min} \leq n \leq dist_{max} \right\} \tag{14}$$

where c is the existing value of the gene, n is a random real-valued number between the $dist_{min}$ and $dist_{max}$ minimum and maximum intensity values allowed for the distortion, and m is the mutated gene value.

Step 5: Run Replacement - Once a new generation of chromosomes has been created, an $\lambda + \mu$-update replacement process is used to select which chromosomes will be retained. This method keeps the chromosomes with the best fitness values from both the parent and child generations, preserving good chromosomes from the parent generation that might otherwise be lost with a traditional generational replacement.

Step 6: Evaluate Termination Criteria - Once replacement has occurred, the fitness values for all chromosomes are compared. The best fitness value is recorded for each generation. The genetic learning algorithm can terminate if enough generations have been run or if the best fitness value stagnates.

TABLE 3. ***fg*CAPTCHA genetic algorithm details.**

Parameter	Type or Value
Initial Chromosomes	150
Selection Method	Roulette Wheel
Crossover Rate	80%
Crossover Method	Modified Single-Point
Mutation Rate	5%
Replacement Method	$\lambda + \mu$-update
Termination Criteria	Stagnation or Maximum Generations
Stagnation Condition	Minimum 50 generations, then less than 1% change in best fitness over 5 generations
Maximum Generations	100

Otherwise, operation of the genetic learning algorithm continues and the chromosomes resulting from the replacement process are provided as input to the selection step to create a new generation. Actions are determined by,

$$\text{Action} = \begin{cases} \text{Complete} & \text{if } g \geq 100 \\ \text{Complete} & \text{if } g \geq 50 \text{ and } best_g \leq 1.01 * best_{g-5} \\ \text{Continue} & \text{otherwise} \end{cases} \tag{15}$$

Here, g is the number of the current generation and $best_g$ is the best fitness value for generation g.

Step 7: Completion - The genetic learning process stops once the termination criteria have been satisfied. To ensure that any readily-attackable images do not see public use, all CAPTCHAs with computer-based attack success rates of $S_A = 1.0$ are discarded. The remaining CAPTCHAs with the best fitness values are recorded along with their embedded image coordinates so they can be used as tests. Examples of generated CAPTCHAs presented to users are shown in Fig. 6.

TABLE 4. Display sizes of common devices used to evaluate the performance of the proposed CAPTCHA.

Device	Type	Resolution (pixels)	Pixels per Inch	Screen Size (inches)
Apple iPhone 3GS	Smartphone	480x320	163	3.5
Apple iPhone 4S	Smartphone	960x640	326	3.5
Blackberry Storm 2	Smartphone	480x360	184	3.25
Motorola Electrify M	Smartphone	540x960	256	4.3
Apple iPad 2	Tablet	1024x768	132	9.7
Dell 170S	Monitor	1280x1024	96	17

This work is an extension of preliminary research [40], [41] where the CAPTCHA is generated using simple visual distortions. The proposed approach improves on the previous model in multiple ways:

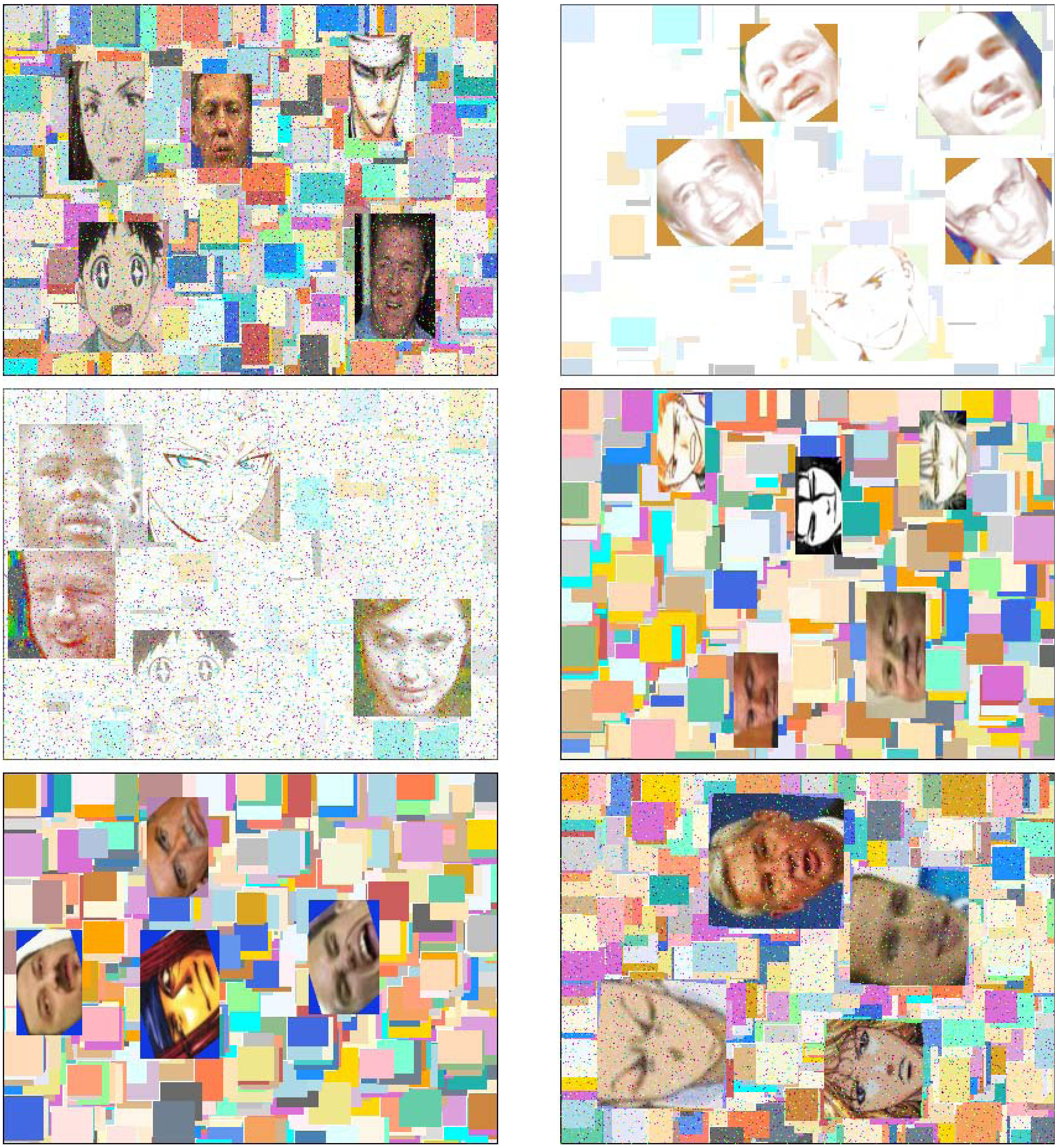


FIGURE 6. Examples of *fg*CAPTCHA.

1) Incorporates improved visual distortions which further strengthen the security of the CAPTCHA without sacrificing human ability to solve.
2) Uses color images and a genetic learning algorithm-based image generation process which increases human success rates while also reducing the automated attack rates in solving the face detection image CAPTCHA.
3) Removes the dependency on humans for parameter selection and optimization and therefore makes the CAPTCHA generation process highly scalable and able to meet the target success objectives.
4) Takes into account design requirements of the various devices used to view the CAPTCHA. As shown in Table 4, screen size and resolution can vary significantly even among devices of the same type.

A well-designed CAPTCHA must work effectively across the entire spectrum of computing devices, from smartphones where it may be the only item on-screen to tablets and computers where it is part of a larger webpage.

III. EXPERIMENTAL METHOD, RESULTS AND ANALYSIS

This section provides the details of image databases used, participants, and protocol followed for designing and evaluating the performance of the proposed CAPTCHA along with the results and analysis.

A. IMAGE DATABASE

For experimental evaluation, publicly-available photographs from the LFW face database are used for human face images [42]. Cartoons and high-quality sketches from photobucket.com comprise the non-face images used in the CAPTCHA.

B. PARTICIPANTS AND TESTING PROTOCOL

Evaluation of *fg*CAPTCHA is conducted with the help of 2,600 volunteers, all above 18 years of age. Prior to collecting responses, consent of the volunteers is obtained and they are informed that their responses would be used for research and analysis purposes. The users accessed the webpage protected by *fg*CAPTCHA in an uncontrolled environment using their preferred method of accessing the Internet. Users were free to use desktops, notebooks, and smartphones to access and solve *fg*CAPTCHA.

The size of each *fg*CAPTCHA image is 400 × 300 pixels. Only one *fg*CAPTCHA is present on the screen at one time along with other webpage content. If the user is unsuccessful in solving a particular *fg*CAPTCHA, a different *fg*CAPTCHA image is provided to solve and access protected content. In-depth mobile device testing has also been completed by 17 volunteers using a combination of tablets and smartphones. These users have compared *fg*CAPTCHA with two other popular CAPTCHAs, namely text-based reCAPTCHA and image-based IMAGINATION. Success rates are recorded and users also provide a ranking of the CAPTCHAs by their ease of use. Automated attack testing is performed using the Viola-Jones face detection algorithm and two commercial face detection packages. The faces detected by software are compared to the actual face locations. If any portion of the detected face overlaps an actual face, the face is considered to be correctly found. An automated attack is considered successful if all human faces in a CAPTCHA are found without any false detections.

C. ANALYSIS

1) EVALUATING HUMAN PERFORMANCE

To collect the data and evaluate the effectiveness of *fg*CAPTCHA, over 40,000 attempts by over 2,600 users have been recorded as a part of a university login page. The human success of solving *fg*CAPTCHA is dependent on

TABLE 5. ***fg*CAPTCHA success rates for distortion type pairs.**

Distortion Types	Human Success	Attack Success
Rotation, Height Scaling	97.0%	0.0%
Increase Brightness, Salt-and-Pepper Noise	92.9%	0.0%
Erosion, Salt-and-Pepper Noise	92.6%	0.0%
Rotation, Speckle Noise	91.7%	0.0%
Height Scaling, Width Scaling	91.4%	0.0%
Rotation, Width Scaling	89.9%	0.0%
Piecewise Scaling, Width Scaling	88.7%	0.0%
Rotation, Salt-and-Pepper Noise	88.5%	0.0%
Increase Brightness, Speckle Noise	88.2%	0.0%
Increase Brightness, Rotation	88.1%	0.0%
Piecewise Scaling, Salt-and-Pepper Noise	87.6%	0.0%
Salt-and-Pepper Noise, Height Scaling	87.3%	0.0%
Piecewise Scaling, Height Scaling	85.8%	0.0%
Salt-and-Pepper Noise, Width Scaling	85.8%	0.0%
Resolution Reduction, Height Scaling	85.1%	0.0%
Increase Brightness, Resolution Reduction	83.3%	0.0%
Salt-and-Pepper Noise, Speckle Noise	83.3%	0.0%
Periodic Noise, Width Scaling	81.3%	0.0%
Increase Brightness, Width Scaling	81.1%	0.0%
Erosion, Resolution Reduction	81.0%	0.0%
Increase Brightness, Height Scaling	81.0%	0.0%
Resolution Reduction, Salt-and-Pepper Noise	80.0%	0.0%
Overall	**87.9%**	**0.0%**

the complexity and level of distortions applied using genetic learning. With simple distortions such as rotation and height scaling, the human success rate is 97%; whereas, resolution reduction and adding noise affect the performance significantly. In our experiments, average human performance across all variations is 87.9%. Detailed results are summarized in Table 5. From these results, we can infer that, in general, geometric distortions such as height scaling and rotation yield higher success rates. These distortions do not fundamentally alter the appearance of images; they just resize or reposition facial features, which allow human users to easily detect the embedded faces. Noise-based distortions also yield similar performance. Degradation distortions yield lower accuracies as, in some cases, they tend to destroy the fine details needed to distinguish images. This effect is especially pronounced when sketches are used for non-face images. When degradation distortions are used, humans have significant difficulty in distinguishing between human face photographs and non-real face sketches.

Additionally, 17 volunteers participated in evaluating the proposed CAPTCHA on mobile devices where a combination of tablets and smartphones are used. In this evaluation, *fg*CAPTCHA achieves the best mobile device human success

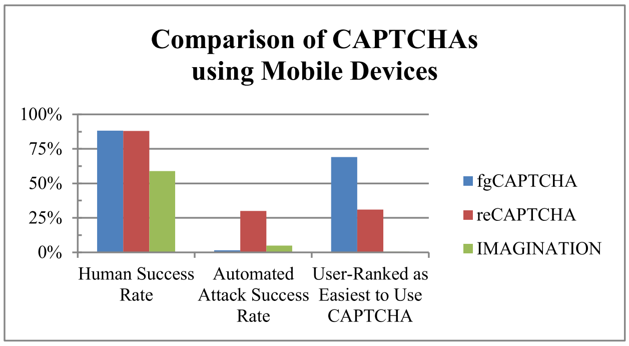


FIGURE 7. Comparison of CAPTCHAs when tested by humans on mobile devices, contrasted with automated attack success rates [23], [30].

rate with 88.2% accuracy. 70% of test volunteers indicate that *fg*CAPTCHA is easiest to use, with several individuals commenting that it can be completed quicker than other CAPTCHAs. Volunteers specifically appreciate the touch-friendly nature of the proposed *fg*CAPTCHA which can be solved with just a few taps to the screen. Moreover, the higher human success rate of *fg*CAPTCHA implies that there is a smaller chance of requiring multiple attempts at the CAPTCHA to access protected content compared to the alternatives. This is another highly desirable trait in determining ease of use. Fig. 7 illustrates these comparisons along with automated attack success rates. This comparison clearly shows that the proposed *fg*CAPTCHA is language-independent, easy to solve, and mobile user-friendly.

2) AUTOMATED ATTACK EVALUATION

In the automated attack evaluation, three off-the-shelf approaches are used to detect faces in *fg*CAPTCHA with varying rotation and scale parameters. In our experiments, none of the automated face detection algorithms are able to correctly solve any of the tested CAPTCHA images. In cases where the algorithms are able to detect some human face images, other faces are either missed or falsely detected. This is largely expected since the widely-used Viola-Jones face detector is incorporated into the CAPTCHA generation process and cases where the Viola-Jones detector locates all faces are automatically discarded from the test set.

It is unlikely that an automated brute force attack on *fg*CAPTCHA would be successful. Each CAPTCHA contains 2-4 human face images, each being approximately 100×100 pixels in size. Including the $\frac{1}{3}$ chance of guessing the number of embedded images, the likelihood of one random guess at solving the CAPTCHA being accurate is approximately,

$$\left(\frac{1}{3}\right)\prod_{i=0}^{3}\frac{(100)(100)i}{(400)(300)} = 0.157\% \qquad (16)$$

Since new CAPTCHA images are presented on each attempt, attackers are unable to use their previous guesses to improve the accuracy of future attempts. Attackers must make a new random guess each time. Thus, the effective attack success rate is less than 1.6-in-1000, thereby significantly enhancing security of the online environment using the proposed *fg*CAPTCHA.

IV. CONCLUSION

As demonstrated in this paper, the unique touchscreen technology of mobile devices can be leveraged to create an additional layer of security that is both effective and user-friendly. The proposed genetically optimized *fg*CAPTCHA works efficiently on both touchscreens used by tablets and smartphones and on traditional computers, achieving a high 88% human accuracy rate during evaluation. It does so without compromising performance, offering an effective 0% automated attack rate. This combination of low attack rates, high human accuracy rates, and convenient mobile device usage provides major improvements over existing desktop-centric security CAPTCHAs in widespread use today.

APPENDIX

A working demonstration of *fg*CAPTCHA is available at http://fgcaptcha.captcharesearch.com.

ACKNOWLEDGMENT

The authors would like to thank the volunteers involved in testing *fg*CAPTCHA's human performance.

REFERENCES

[1] R. A. Botha, S. M. Furnell, and N. L. Clarke, "From desktop to mobile: Examining the security experience," *Comput. Security*, vol. 28, nos. 3–4, pp. 130–137, 2009.
[2] J.-C. Birget, D. Hong, and N. Memon, "Graphical passwords based on robust discretization," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 3, pp. 395–399, Sep. 2006.
[3] N. Ben-Asher, N. Kirschnick, H. Sieger, J. Meyer, A. Ben-Oved, and S. Möller, "On the need for different security methods on mobile phones," in *Proc. 13th Int. Conf. Human Comput. Interaction with Mobile Devices and Services*, 2011, pp. 465–473.
[4] M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song, "Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 1, pp. 136–148, Jan. 2013.
[5] H. Lee, S.-H. Lee, T. Kim, and H. Bahn, "Secure user identification for consumer electronics devices," *IEEE Trans. Consum. Electron.*, vol. 54, no. 4, pp. 1798–1802, Nov. 2008.
[6] D.-J. Kim, K.-W. Chung, and K.-S. Hong, "Person authentication using face, teeth and voice modalities for mobile device security," *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2678–2685, Nov. 2010.
[7] M. Ongtang, S. McLaughlin, W. Enck, and P. McDaniel, "Semantically rich application-centric security in android," in *Proc. ACSAC*, Honolulu, Hawaii, Dec. 2009, pp. 340–349.
[8] S. Shirali-Shahreza, "Bibliography of works done on CAPTCHA," in *Proc. 3rd Int. Conf. Intell. Syst. Knowl. Eng.*, vol. 1. Xiamen, China, 2008, pp. 205–210.
[9] B. Pinkas and T. Sander, "Securing passwords against dictionary attacks," in *Proc. 9th ACM Conf. Comput. Commun. Security*, 2002, pp. 161–170.
[10] M. Kolsch and M. Turk, "Keyboards without keyboards: A survey of virtual keyboards," in *Proc. Workshop Sens. and Input Media-Centric Syst.*, Santa Barbara, CA, USA, Jun. 2002.
[11] H. S. Baird and K. Popat, "Human interactive proofs and document image analysis," in *Proc.5th Int. Workshop Document Anal. Syst. V*, 2002, pp. 531–537.
[12] M. D. Lillibridge, M. Abadi, K. Bharat, and A. Z. Broder, "Method for selectively restricting access to computer systems," U.S. Patent 6 195 698, Feb. 27, 2001.
[13] G. Mori and J. Malik, "Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1. Madison, WI, USA, Jun. 2003, pp. 134–141.

[14] H. S. Baird and T. Riopka, "ScatterType: A reading CAPTCHA resistant to segmentation attack," in *Proc. Document Recognit. Retr. XII*, vol. 5676. San Jose, CA, USA, Jan. 2005, pp. 197–201.
[15] H. S. Baird, M. A. Moll, and S.-Y. Wang, "ScatterType: A legible but hard-to-segment CAPTCHA," in *Proc. 8th Int. Conf. Document Anal. Recognit.*, vol. 2. Seoul, South Korea, 2005, pp. 935–939.
[16] H. S. Baird, M. A. Moll, and S.-Y. Wang, "A highly legible CAPTCHA that resists segmentation attacks," in *Proc. 2nd Int. Workshop Human Interact. Proofs*, 2005, pp. 27–41.
[17] M. Chew and H. S. Baird, "BaffleText: A human interactive proof," in *Proc. Document Recognit. Retr. X*, Santa Clara, CA, USA, Jan. 2003, pp. 305–316.
[18] J. Yan and A. Salah El Ahmad, "A low-cost attack on a microsoft captcha," in *Proc. 15th ACM Conf. Comput. and Commun. Security*, Oct. 2008, pp. 543–554.
[19] Microsoft, Albuquerque, NM, USA. (2006). *Microsoft Human Interaction Proof (HIP): Technical and Market Overview* [Online]. Available: http://download.microsoft.com/download/3/2/0/320e814a-d969-4c6c-a26e-2f3115032d4c/Human_Interaction_Proof_Technical_Overview.doc
[20] A. Rusu and V. Govindaraju, "Handwritten CAPTCHA: Using the difference in the abilities of humans and machines in reading handwritten words," in *Proc. 9th Int. Workshop Frontiers Handwriting Recognit.*, Tokyo, Japan, Oct. 2004, pp. 226–231.
[21] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "reCAPTCHA: Human-based character recognition via web security measures," *Science*, vol. 321, no. 5895, pp. 1465–1468, Sep. 2008.
[22] (2010). *What is reCAPTCHA?* [Online]. Available: http://recaptcha.net/learnmore.html
[23] P. Baecher, N. Buscher, M. Fischlin, and B. Milde, "Breaking reCAPTCHA: A holistic approach via shape recognition," in *Future Challenges in Security and Privacy for Academia and Industry*, vol. 354, J. Camenisch, S. Fischer-Hübner, Y. Murayama, A. Portmann, and C. Rieder, Eds. Berlin, Germany: Springer-Verlag, 2011, pp. 56–67.
[24] Carnegie Mellon Univ., Pittsburgh, PA, USA. (2004). *ESP-PIX* [Online]. Available: http://server251.theory.cs.cmu.edu/cgi-bin/esp-pix/esp-pix
[25] J. Elson, J. Douceur, J. Howell, and J. Saul, "Asirra: A CAPTCHA that exploits interest-aligned manual image categorization," in *Proc. 14th ACM Conf. Comput. Commun. Security*, Oct. 2007, pp. 366–374.
[26] P. Golle, "Machine learning attacks against the Asirra CAPTCHA," in *Proc. 15th ACM Conf. Comput. and Commun. Security*, Oct. 2008, pp. 535–542.
[27] P. Matthews and C. C. Zou, "Scene tagging: Image-based CAPTCHA using image composition and object relationships," in *Proc. 5th ACM Symp. Inf., Comput. and Commun. Security*, 2010, pp. 345–350.
[28] A. Basso and S. Sicco, "Preventing massive automated access to web resources," *Comput. Security*, vol. 28, nos. 3–4, pp. 174–188, May 2009.
[29] R. Datta, J. Li, and J. Z. Wang, "Exploiting the human-machine gap in image recognition for designing CAPTCHAs," *IEEE Trans. Inf. Forensics Security*, vol. 4, no. 3, pp. 504–518, Sep. 2009.
[30] B. B. Zhu *et al.*, "Attacks and design of image recognition CAPTCHAs," in *Proc. 14th ACM Conf. Comput. and Commun. Security*, Oct. 2010, pp. 187–200.
[31] J. Tam, S. Hyde, J. Simsa, and L. von Ahn, "Breaking audio CAPTCHAs," in *Advances in Neural Information Processing Systems*, vol. 22. Vancouver, BC, Canada: Univ. British Columbia, Dec. 2008.
[32] K. A. Kluever, "Evaluating the usability and security of a video CAPTCHA," M.S. thesis, Dept. Math., Rochester Inst. Technol., Rochester, Rochester, NY, USA, 2008.
[33] K. A. Kluever and R. Zanibbi, "Balancing usability and security in a video CAPTCHA," in *Proc. 5th Symp. Usable Privacy and Security*, Jul. 2009, pp. 1–11.
[34] P. Viola and M. Jones, "Robust real-time object detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, 2002.
[35] C. Zhang and Z. Zhang, "Boosting-based face detection and adaptation," in *Synthesis* (Lectures on Computer Vision), vol. 2. San Rafael, CA, USA: Morgan & Claypool, pp. 1–140, Sep. 2010.
[36] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[37] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.
[38] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1998.
[39] S. Sivanandam and S. Deepa, *Introduction to Genetic Algorithms*. Berlin, Germany: Springer-Verlag, 2008.
[40] B. M. Powell, A. C. Day, R. Singh, M. Vatsa, and A. Noore, "Image-based face detection CAPTCHA for improved security," *Int. J. Multimedia Intell. Security*, vol. 1, no. 3, pp. 269–284, 2010.
[41] G. Goswami, B. M. Powell, M. Vatsa, R. Singh, and A. Noore, "FaceD-CAPTCHA: Face detection based color image CAPTCHA," *Future Generat. Comput. Syst.*, vol. 31, pp. 59–68, Feb. 2014.
[42] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database forstudying face recognition in unconstrained environments," in *Proc. Workshop on Faces 'Real-Life' Images, Detection, Alignment, and Recognit.*, Marseille, France, 2008, pp. 1–11.

BRIAN M. POWELL (S'03–M'14) received the M.S. degree in computer science from West Virginia University, Morgantown, WV, USA, in 2006, where he is currently pursuing the Ph.D. degree with the Lane Department of Computer Science and Electrical Engineering. His areas of interest are face detection image-based CAPTCHAs for improved security, human computation, user interface design, and computer science education. He is a member of the Association for Computing Machinery, and the Phi Kappa Phi, Upsilon Pi Epsilon, and Sigma Zeta honor societies. He was a recipient of the West Virginia University Foundation Distinguished Doctoral Fellowship.

GAURAV GOSWAMI (S'12) received the B.Tech. degree in information technology from the the Indraprastha Institute of Information Technology, Delhi, India, in 2012, where he is currently pursuing the Ph.D. degree. His main areas of interest are image processing, computer vision, and their application in biometrics.

MAYANK VATSA (S'04–M'09) received the M.S. and Ph.D. degrees in computer science from West Virginia University, Morgantown, WV, USA, in 2005 and 2008, respectively. He is currently an Assistant Professor with the Indraprastha Institute of Information Technology, Delhi, India. He has more than 125 publications in refereed journals, book chapters, and conferences. He was a recipient of the FAST Award by the Department of Science and Technology, India. His areas of interest are biometrics, image processing, computer vision, and information fusion. He is a member of the Computer Society and the Association for Computing Machinery. He was a recipient of several best paper and best poster awards in international conferences. He is also an Area Editor of the IEEE Biometric Compendium, an Area Chair of *Information Fusion*, Elsevier, and a PC Co-Chair of the 2013 International Conference on Biometrics and the 2014 International Joint Conference on Biometrics.

RICHA SINGH (S'04–M'09) received the M.S. and Ph.D. degrees in computer science from West Virginia University, Morgantown, WA, USA, in 2005 and 2008, respectively. She is currently an Assistant Professor with the Indraprastha Institute of Information Technology, Delhi, India. She was a recipient of the FAST Award from the Department of Science and Technology, India. Her areas of interest are biometrics, pattern recognition, and machine learning. She has more than 125 publications in refereed journals, book chapters, and conferences. She is also an Editorial Board Member of *Information Fusion*, Elsevier and *EURASIP Journal of Image and Video Processing*, Springer. She is a member of the CDEFFS, the Computer Society, and the Association for Computing Machinery. She was a recipient of several best paper and best poster awards in international conferences.

AFZEL NOORE (S'84–M'87–SM'10) received the Ph.D. degree in electrical engineering from West Virginia University, Morgantown, WV, USA. He was a Digital Design Engineer with Philips, Chennai, India. From 1996 to 2003, he was the Associate Dean for Academic Affairs and a Special Assistant to the Dean of the College of Engineering and Mineral Resources at West Virginia University, where he is currently a Professor with the Lane Department of Computer Science and Electrical Engineering. His current research interests include computational intelligence, biometrics, software reliability modeling, machine learning, and computer vision. He serves on the Editorial Boards of *Recent Patents on Engineering*, the *Open Nanoscience Journal*, the *International Journal of Advanced Pervasive and Ubiquitous Computing*, and the *International Journal of Multimedia Intelligence and Security*. He has over 100 publications in refereed journals, book chapters, and conferences. He has been recognized as an outstanding teacher and outstanding researcher several times. He is a member of the Phi Kappa Phi, Sigma Xi, Eta Kappa Nu, and Tau Beta Pi honor societies. He was a recipient of seven best paper and best poster awards in international conferences.