Image and Scene Tagging: An Alternative Approach to Current CAPTCHA Techniques

Andrew Mantel Peter Matthews
Computer Science Department
University of Central Florida

ABSTRACT

In this paper we investigate image tagging-based CAPTCHA systems, presenting analysis of the range of answers that should be accepted by an image tagging system to maximize both security and user friendliness via the use of WordNet, and extend this in providing an optimal set of multiple choice selections. We also consider the likely attacks such a system would face and propose possible countermeasures via the use of image distortion and obfuscation. We then propose scene tagging, a novel form of image-based CAPTCHA that extends the concept of image tagging into a more attack resistant, yet still user-friendly, format. Scene tagging utilizes a question format based on relationships between objects in an automatically generated composite image. A system capable of creating both image tagging and scene tagging CAPTCHAs was implemented for the purpose of conducting a user study of feasibility, and results show that it holds great promise for real-world deployment.

1. INTRODUCTION

Computer Automated Turing Test for telling Computers and Humans Apart, or CAPTCHA, are challenge-response tests used by many web sites to establish that a user is a human rather than an automated script or bot. This method was created to help prevent the automated abuse of web services such as posting of spam to comment sections, mass user account registration, dictionary or brute force password attacks, and abuse of online polls. Text-based CAPTCHAs, in which users are required to transcribe text which has been distorted in some fashion, rely upon the fact that deciphering distorted text is typically a simple task for humans but a difficult task for computers. Such systems are in very wide use amongst many of the most popular websites on the Internet.

However, their vulnerability to attack has been repeatedly demonstrated by computer vision researchers [12,13]. In 2004 [14], for example, several commercial CAPTCHA implementations were attacked by Microsoft researchers with 80% - 95% success rates achieved. They noticed that their attacks had the most difficulty with the character segmentation task, and later implemented a CAPTCHA system based on making this task as difficult as possible. Unfortunately, this very system was later attacked by other researchers [15] who were able to defeat the system more than 60% of the time.

While this cat-and-mouse game has led to advancements in optical character recognition (OCR) technology, they have also required the distortions and obfuscations performed upon the text to become increasingly complicated and extreme. This makes the CAPTCHA images more difficult for computers to understand, but also makes the images more difficult for humans to read. Unfortunately, modern day text-based CAPTCHA images are becoming quite difficult for even humans to read; for example, one system proposed by researchers at Lehigh University [16] was found by a study to have a 53% human legibility rate. This problem will likely be exacerbated as OCR technology continues to improve. Further, there is incentive for non-academic attackers to break CAPTCHA systems due to the monetary benefits of malicious activities that it makes possible; one example of this is the automated registration of web-based e-mail addresses for the purpose of sending spam.

To deal with the shortcomings of current CAPTCHA methodologies, we have developed a CAPTCHA technique based on image tagging. Image tagging tasks a user to identify the object portrayed in an image that has been obfuscated in some way. Furthermore, we expand upon image tagging by also exploring scene tagging. Scene tagging is similar to image tagging in that the user is still asked to identify an object portrayed within a picture. Unlike image tagging, however, a scene tagging problem consists of multiple objects within a single picture and tasks the user with identifying a certain one of those objects or understanding the relationship between a number of those objects.

The remainder of the paper is organized as follows. Section 2 explores related work in developing alternative CAPTCHA techniques. Sections 3 and 4 discuss image tagging and scene tagging, respectively. Section 5 describes the experiments held to empirically test the viability of image and scene tagging, and Section 6 relates our analysis from the experimental results. In Section 7 we summarize the paper and discuss possible future work.

2. RELATED WORK

A number of CAPTCHA systems based upon the understanding of semantic image content have been proposed. Chew and Tygar [5] propose a CAPTCHA system based on a number of image recognition tasks. In testing the validity of these tasks, they utilize a corpus of 627 English words, for each of which there is an associated set of images automatically retrieved via keyword from Google Images. A set of such images are presented to the user, who must then perform a task such as naming a term associated with the set of images or identifying the image with a different subject from the others presented. They perform usability testing with a focus group to show that the system is solvable by humans and to evaluate the problems which affect human performance on the system.

Rui and Liu propose a CAPTCHA system [7] based on the innate capacity humans have for recognizing human faces. Distorted images are generated from facial models, and the user must locate the corners of the eyes and mouth to pass the test. One unfortunate side effect of the system is that a number of users found the images generated to be quite unsettling. Misra and Gaj present a somewhat similar system [6] that requires users to select the matching pair from a set of distorted images of human faces mined from a public database.

Researchers from Sharif University of Technology present a system [17] in which a set of images are rotated before being presented to the user who is prompted to select the image of a specified subject. However, given that there are a number of rotation-invariant image properties in computer vision, it would likely not be difficult to break such a system if an attacker was able to build up the appropriate image corpus.

Hoque, Russomanno, and Yeasin present a system [8] based upon the use of 3d models to generate 2d images. These 3d models are subjected to randomized lighting effects, rotation, scaling, and other distortions to create a 2d image that is then presented to the user who must select the object from a short list. It is possible that our approach of scene tagging could utilize similar methods in creating a 2d scene image from a 3d scene composed of numerous 3d models. However, there are disadvantages to the approach in that large numbers of 3d models are difficult to acquire.

Baird and Bentley propose a concept they call Implicit CAPTCHAs [9] that attempt to provide a less-demanding user experience. They propose disguising site links necessary for accessing bot-restricted sections in imagemaps. An example of this may be a photo of a climber in front of a mountain, with the instructions to “click on the mountaintop to continue”. A problem with the system is that it requires significant manual labor in the manual annotation of images in order to create these challenges, and thus, inevitably, frequent reuse of these challenges. This means that each challenge would only have to be solved once in order to break the system, a task that would be made quite feasible by the use of low-cost labor resources. Thus the proposed system is likely to not be of significant use in anything else but restricting automated web-crawling bots. While Implicit CAPTCHAs also utilize a form of scene tagging, it is important to note the fundamental difference between their approach and ours; namely, we do not require manual annotation of images in our system, rather our system generates scene instances programmatically in a random fashion.

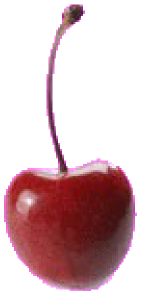
Microsoft’s Asirra [4] asks users to identify the images of cats out of a set of twelve photographs of cats and dogs. This paper presents a compelling solution to the problem of building a database of manually categorized images via a partnership with pet adoption site Petfinder.com, in which Asirra is able to use the site’s constantly-updated database of photos of cats and dogs in exchange for providing an “Adopt me” link under the images of those animals yet to be adopted. The paper also proposes a “token bucket” scheme which punishes users who fail a number of ASIRRA challenges by requiring them to solve a number of challenges in a row before being passed by the system. This makes brute force attacks probabilistically less likely to succeed. We believe the contributions of this paper could be incorporated into future generations of our system. However, a successful attack [10] has been demonstrated against the system with a (support vector machine-based) binary classifier that utilizes texture and color information.

Researchers at The Pennsylvania State University have proposed an image-based CAPTCHA system named IMAGINATION [11] that has a number of similarities with our approach, in that they also utilize WordNet to avoid ambiguity in the selection of multiple choice options and utilize a combination of image composition and distortion in generating the images presented to the user. However, they utilize a different methodology, involving two tasks performed by the user. In the first, the image space presented to the user has been divided into eight non-overlapping sub-rectangles, each of which is filled with a different (scaled) source image. The image space is then divided into eight different non-overlapping sub-rectangles, each of which receives a different form of dithering to obscure these image boundaries. The user must click in the center of any of the eight images to proceed to the second task. In the second task, a single image is distorted and then the user must choose the appropriate image tag based on a number of choices presented. Our system differs in that we consider the combination of numerous forms of distortion and obfuscation, utilize a different form of image creation via object composition, and use different forms of questioning that focus on the relationships between objects in the image rather than image center selection or simple identification.

3. IMAGE TAGGING

Image tagging tasks a user to identify the thing portrayed in a picture; for instance, if the user is presented with the image tagging problem in Figure 1, then a proper response would be "cherry". However, a response such as "fruit" would not be accepted (see Section 3.1 for further details).

Figure 1: Sample Image Tagging Problem



In order to prove that image tagging is a viable replacement to current CAPTCHA techniques, we will prove the following features:

1. Image tagging is easy for a human to solve reliably.
2. Image tagging has a sufficiently large solution surface to probabilistically avoid random computer attacks.
3. Image tagging is sufficiently difficult for current computers to solve.
4. Image tagging is scalable to adjust to advancements in computer technologies.

Feature 1 is investigated in Sections 5 and 6 of this paper. Feature 2 is discussed in Section 3.1. Features 3 and 4 are investigated in Sections 3.2, 3.3, and 4.

3.1 Solution Surface

To prove that image tagging has a sufficiently large solution surface to probabilistically avoid random computer attacks, we mathematically analyzed this solution surface. This analysis required some clever thinking because one cannot just assume that every noun in the dictionary can be represented as an image tagging problem; for instance, we don't expect that the majority of people would be able to correctly image tag a picture of a "lory" (a lory is a particular type of parrot) because a lot of people may be unfamiliar with this particular animal. Instead, a noun should only be considered a viable image tagging problem if it has the following two characteristics:

1. The noun is not too specific that the majority of people would be unfamiliar with it and thus be unable to correctly identify it in an image.
2. The noun is not too generic that a computer could randomly guess the high-level concept.

To aid us in determining the solution space of image tagging, we utilized an often used NLP (Natural Language Processing) tool called WordNet. WordNet is an ontology of language in which nodes are connected to each other in a hierarchy of is-a relationships. Nodes are denoted as synsets representing a grouping of terms of synonymous meaning. If Synset A is-a parent concept of Synset B, then WordNet refers to Synset A as the hypernym of Synset B and Synset B as the hyponym of Synset A. Similarly, any term within a synset has the same properties of the synset, so in the previous example any term from Synset A would be a hypernym of any term from Synset B. As a real example, "parrot" would be a hypernym of "lory", and vice versa "lory would be a hyponym of "parrot", because lory is a particular subconcept of parrot (i.e. lory is a type of parrot).

Using the WordNet ontology, we began estimating the solution space of image tagging by beginning at the high-level concept "physical object" and analyzing its complete hyponym tree. A complete hyponym tree includes the direct hyponyms of a concept, all of those hyponyms' hyponyms, and so forth to the leaves of the ontology. "Physical object" in WordNet refers to any tangible or visible entity, which includes such subconcepts as animals, foods, instruments, etc., and thus represents a good high level concept of potential image tagging problems. What we are interested in is how many of these synsets could be used to create an image tagging problem. Note that we do not care how many terms are within each synset because a correct solution to an image tagging problem would be any term within the represented synset; therefore, even if a synset has multiple terms within it, the size of the solution space is unaffected.

The complete hyponym tree of "physical object" has about 29,600 synsets in it. Ideally, we would go through each synset and determine whether that synset meets the criteria of a valid image tagging problem. Since we are limited by time restraints though, we decided to estimate the number of valid image tagging synsets under "physical object". We did this by observing that the leaves of the hyponym tree (i.e. those synsets which have no hyponym) are often very specific instances of a concept and may be too specific to make good image tagging problems. The first-level hypernyms of the leaves can be a little less specific, but depending on the location within the ontology these synsets may still be too specialized to be image tagging problems. We therefore decided to analyze only three levels of the "physical object" hyponym tree: the leaves, the first-level hypernyms of the leaves, and the second-level hypernyms of the leaves. We took a random sample of fifty synsets of each of these levels. Then, we went through each randomly chosen synset of each of the selected ontology levels and manually determined whether that synset could be used in an image tagging problem. Table 1 contains are observations:

Table 1: "Physical Object" Complete Hyponym Tree Analysis

Ontology Level	**Total # of Synsets**	**% of Viable Image Tagging Synsets Based on Random Sampling**
leaves	22852	10%
first-level leaf hypernyms	6386	10%
second-level leaf hypernyms	2262	20%

From the data in Table 1, we estimate that there are at least 3375 synsets that could be used for image tagging problems. So even if a computer-based attack was aware of these synsets, it would only have about a 0.03% chance of guessing the correct synset for a particular image tagging problem. We believe this probability is significantly low enough to protect image tagging from random computer attacks.

3.2 Countermeasures Against Likely Attacks

There are two primary types of attacks to consider when considering implementing an image tagging CAPTCHA system. The first countermeasure is based on data mining, while the second countermeasure relies on a type of image recognition system called content-based image retrieval (CBIR).

In the data mining-based attack, an attacker collects a large number of image tagging problems from a web site along with the corresponding answers. Although collecting the image tagging problems could be automated, the attacker would have to use manual labor to solve the problems. If the attacker can accumulate a large enough number of image tagging problems, and if the web site implementing image tagging disregards the feasibility of a data mining attack, then a system could be setup to easily compare an image tagging problem to the attacker's backend database of solved problems. To prevent the possibility of the data mining attack, an image tagging implementation should either regularly update the image tagging problems, use a distortion engine (see Section 3.3), or use scene tagging instead (see Section 4).

Next, we took a quick survey of existing state-of-the-art CBIR systems in order to determine if a computer system currently exists that could reliably solve image tagging problems. From what we have read, it seems that the majority of CBIR systems today take a two phase approach. First, the system will separate the image into several components, each component representing a potentially different concept. As an example, imagine an image of an airplane flying through the sky. In this case, the first phase of image recognition would attempt to separate the airplane, patches of sky, and clouds into separate components. Depending on the system, these components may either be automatically separated [3] or manually specified by the user who draws a region of interest around the desired component [1,2]. Once separated, the second phase of the system compares each component to a backend database of manually annotated images using one or more types of image similarity algorithms. Each component is then given a label based on the most closely similar image or set of images from the backend database. Continuing from the earlier example, the system would compare the airplane component against all images in the backend database, possibly find that this component is similar to one or more images of an airplane, and thus give the component the appropriate label.

To take a closer look, let's examine the MEMORI system [1]. MEMORI is able to isolate rotated and scaled objects from an image with a complex background by asking the user to draw a polygon around part or all of the object. Once the object has been isolated, similar objects can be pulled from the backend object database using a similarity algorithm. Assuming that an attacker has a large enough object database with manually annotated image tags, MEMORI could potentially be used to help narrow the solution space of an image tagging problem by providing a list of similar objects to the given image tagging object. In the reported research though, the system was only being used to find similar types of furniture from a furniture manufacturer's web site; there is no mention in the research as to how well MEMORI is able to discern similar types of objects when the object database is expansive and diverse. Furthermore, although MEMORI can handle rotation and scaling of objects, there is no data to support that it can perform reliably when other obfuscation techniques are implemented.

3.3 Image Distortion and Obfuscation

The distortion and obfuscation of images is essential in guarding the system against both simple data-mining based attacks and more complex computer vision-based object recognition attacks. A distortion filter used in image tagging must have the primary quality that it impedes computer vision algorithms significantly without taking an undue toll on human understanding of images. One additionally desirable quality is that the filter has a number of parameters which control the way in which the distortion affects the image. This results in a far larger space of possible image distortions and thus makes searching such a distortion space much more difficult. Another desirable quality is that the distortion is not easily invertible. An example of a filter that meets these criteria is one that performs the addition of varying levels of randomized colored pixel noise to the image. Moderate levels of noise make machine recognition of an object more difficult, especially if the levels of noise change randomly across an image, but have little effect on human recognition of an object. Further, it is not invertible without resorting to a mean-based blur, which results in a significant loss of image fidelity and textural information.

There has been little work in quantifying the robustness of machine object recognition algorithms in the face of strong distortions and clutter used in an adversarial manner. As a result, it appears that the best way to consider the effectiveness of a distortion against machine vision techniques is to consider its effects on the lower-level machine vision image feature primitives on which these techniques rely. These primitives include edge detection, segmentation, interest point location, local texture information, and local color information.

The first filter utilized by our system, Grid Warping, performs a randomized warping of the image thereby obscuring shape information. This is done by first creating a 5x5 grid of Cartesian points that represent appropriately spaced points of the image. The locations of points in this 5x5 grid are then independently adjusted in a randomized fashion. The filter then deforms the image such that an image point in the original grid moves to its counterpart in the destination grid and all points in between are interpolated appropriately. This results in a distortion of object shapes in a manner that poses no significant barrier to human recognition but should make things more difficult for object recognition algorithms by disturbing the relative locations and relationships of edges, segments, and interest points. The Swim filter warps the image in a way which make it look like it’s underwater. The end result is fairly similar to grid warping, but the warping of the image instead occurs in a regular, repeating pattern. It should likewise have a significant effect on edge detection, segmentation, and interest point detection.

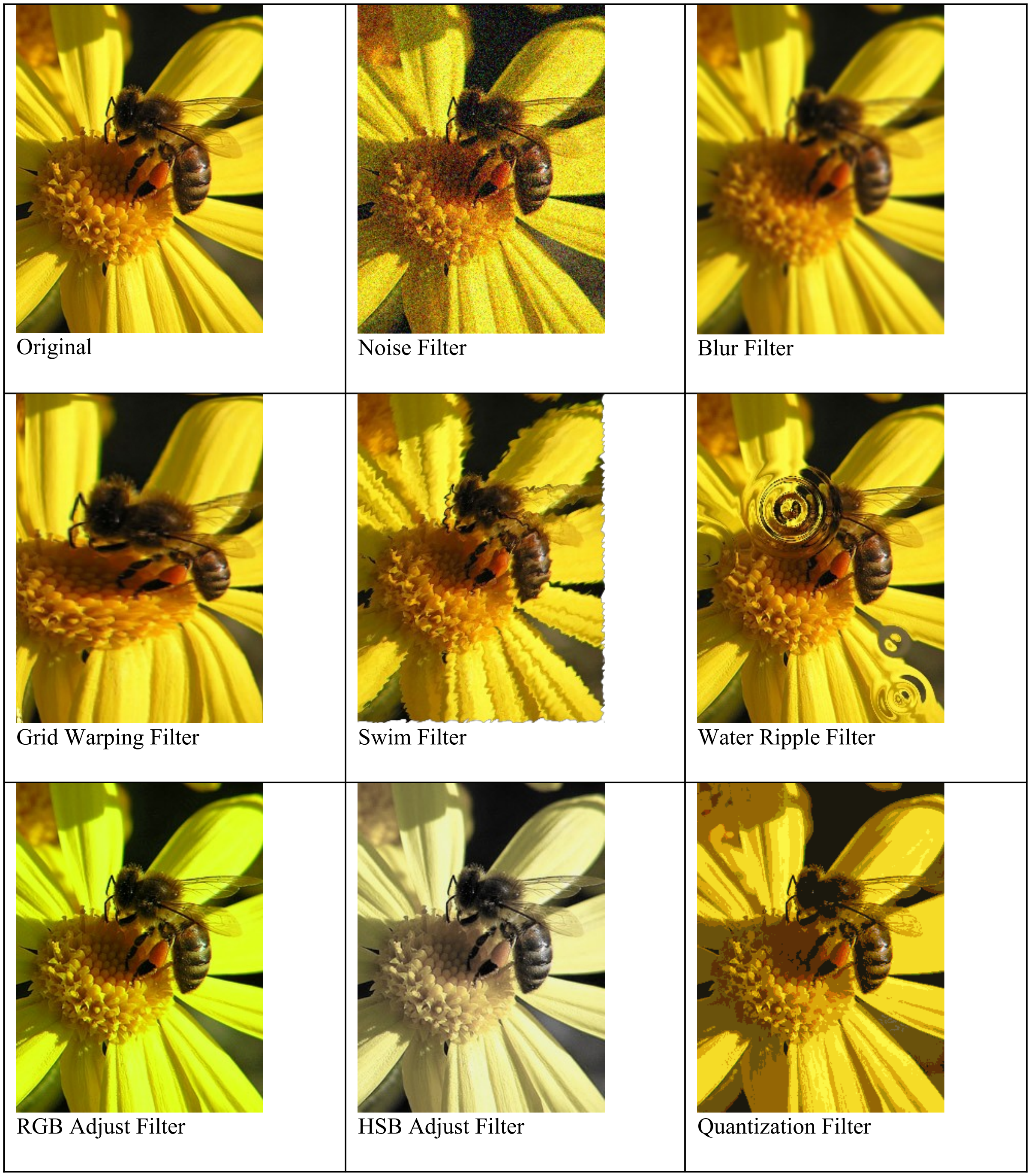
The Water Ripple filter produces an effect like that of a water ripple centered at a point on the image. The position, wavelength, amplitude, and phase of the ripple are set in a random fashion. This filter severely impacts edge, segment, and textural information in a localized fashion. It can, however, cause problems with human object recognition if it distorts too much of an object -- thus it must be used in a careful, controlled fashion.

Image Color Quantization reduces the number of colors in the image to a random number between 64 and 127. A large amount of textural and color information may be lost in the process, but the preservation of object shape and basic color information usually allows object identification by humans with a minimum of effort. The HSB Adjust filter shifts the hue, saturation, or brightness of the image by a random number in a range associated with that component. These individual ranges are used to ensure that the distortions that result are not too drastic. This is necessary because while a reasonably sized brightness shift would likely leave the image’s objects in a human recognizable state, major hue shifts can result in very odd image artifacts becoming visible or make objects far less recognizable. The RGB Adjust filter adds or subtracts a random amount from the red, green, or blue channels of an image. Once again the degree of change is constrained, due to humans reliance on color information in identifying objects. (A user may have trouble identifying a purple pizza slice, for example.) These two filters are used to make the use of local color information more difficult.

The Noise filter adds random noise to an image, and was discussed previously. The Blur filter performs a simple convolution based blurring of the image. This obscures textural information and makes edge detection more difficult.

A random combination of the aforementioned filters is used on each image before presented to the user. For a visual demonstration of the effects of these filters, please see figure 2. Finally, the system also places a number of random shapes over the image to obfuscate the object. The use of random shape placement in this stage attempts to exploit the Gestalt perception abilities of humans, namely that human have a strong ability to recognize and understand images in the face of incomplete or fragmented visual information whilst machines have a very difficult time doing the same thing. These shapes are placed in a random image location and are of random type (line, ellipse, or curve). They are also of random color, being either a flat random color, a gradient between two random colors, or "image-textural" -- colored with pixel colors corresponding to those found in sections of the image a set direction and distance away from the shape. Finally, they are of random alpha-level, as it is believed that allowing such transparency effects will make their detection and removal more difficult for computer vision algorithms while preserving information useful for human recognition.

Figure 2: Example Distortion Filter Result



Original

Noise Filter

Blur Filter

Grid Warping Filter

Swim Filter

Water Ripple Filter

RGB Adjust Filter

HSB Adjust Filter

Quantization Filter

3.4 Scalability

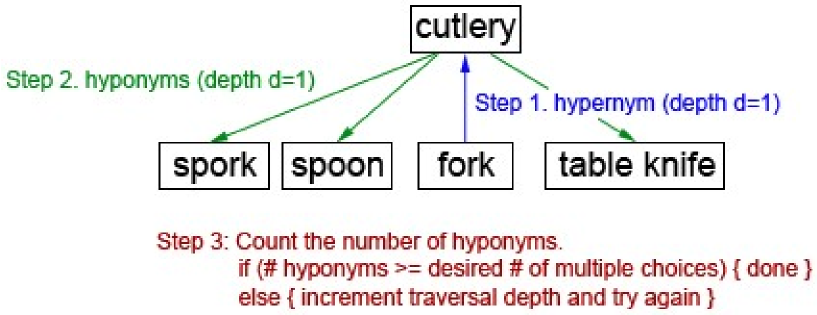
On a machine with a 2.4 Ghz Intel Core 2 Duo E6600 processor, generating a object tagging instance (a object image, question, and answer tuple) upon which 7 random shapes have been overlaid and two effects applied takes approximately 36.09 milliseconds. However, the system was implemented in Java and no code optimization has yet been performed. It is believed that porting the system to a non-interpreted language and optimizing the image distortion functions would yield a very significant reduction in this generation time, such that the overhead incurred in object tagging instance generation would not be prohibitive for widespread use.

3.5 Multiple Choice Engine

Our multiple choice engine connects into WordNet in order to generate a list of words similar to the given solution term sense. For instance, if we wanted a list of multiple choices to go along with the cutlery sense of the term "fork", then the multiple choice engine would return such choices as plate, dish, table knife, and spoon. This design was based on the possibility that even if a computer could determine the high level concept of an image tag (in the previous example, if the computer could determine that this is an image of "tableware"), it would still be unable to guess the correct image tag among the multiple choices because all of the choices are closely related.

The mutliple choice engine functions by starting from the solution term sense. From there, the system traverses upwards the minimum number of hypernyms *d* and then back down the same *d* number of hyponyms in order to find terms that are both related to the solution term sense and of similar descriptive level. The depth of traversal must be large enough to generate the desired number of multiple choices. Only moving up and then down a single hypernym level, for example, may not provide the desired number of terms, depending on the local ontology near the solution term sense. If there are more choices generated than desired, then the mutliple choice engine randomly chooses from the available choices. Figure 3 portrays a simple example trace of generating multiple choices for the cutlery sense of the term "fork". Note that if we wanted more than four multiple choices for this example, then the engine would have to increase the traversal depth *d* above one.

Figure 3: Example Trace of the Multiple Choice Engine


cutlery
Step 2. hyponyms (depth d=1)
Step 1. hypernym (depth d=1)
spork
spoon
fork
table knife
Step 3: Count the number of hyponyms.
if (# hyponyms >= desired # of multiple choices) { done }
else { increment traversal depth and try again }

Of course, by providing multiple choices we are greatly reducing the solution space of the image tagging problem. So if it is decided that multiple choices should be used for image tagging problems, then it would probably also be necessary to require the user to solve more than one problem. The probability *p(n,k)* of a computer randomly selecting the correct multiple choice for *n* image tagging problems, where each image tagging problem has *k* multiple choices, is:

Equation 1: Multiple Choice Random Probability

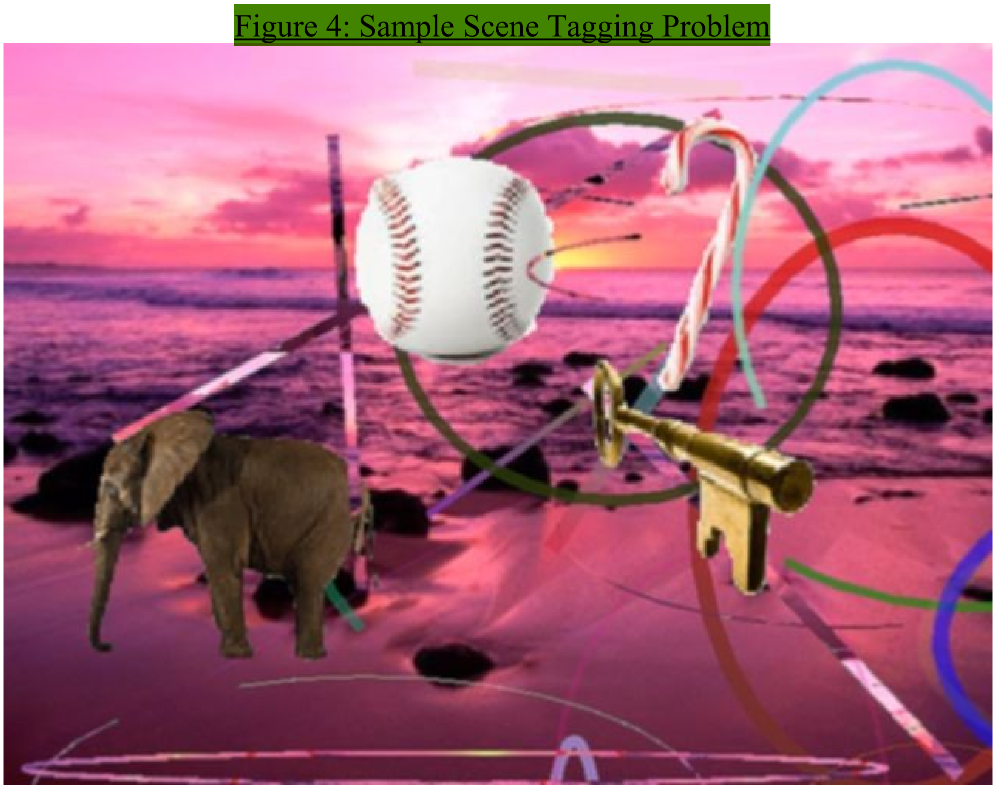
$$p(n,k) = \left(\frac{1}{k}\right)^n$$

The choice of *n* and *k* can be decided appropriately based on the desired level of security versus convenience. In other words, although increasing either *n* or *k* will decrease the chance of a computer randomly solving the problems, it will also require additional time for a human user to solve, especially for larger values of *n*. For our experiments we provided every multiple choice problem with sixteen choices. In order to make a system using multiple choice have an equivalent solution space to an image tagging problem without multiple choice, the implementation would have to require the user to solve three problems (resulting in a random guess probablity of 0.024%).

4. SCENE TAGGING

Scene tagging represents a natural progression from image tagging. It first creates an image via composition of a background image and several object images, and then asks the user to answer a question based on the relationships between these objects in the presented image. Figure 3 below displays a sample scene tagging problem. A question for such a problem might be "Name the object the lower-left of baseball", and the correct answer would be "elephant".

Figure 4: Sample Scene Tagging Problem



4.1 Scene Tagging Engine

A system to automatically generate scene tagging problem instances was created in the Java programming language. This system first randomly chooses a background image from the set of available backgrounds. It then performs the first round of image distortion and obfuscation, described in section 4.4. At this point, a number of objects are selected randomly from the object corpus and the associated images are placed over the image. The second round of image distortion and obfuscation is then performed. System parameters, such as the number of objects to be placed, are configurable and may be changed at any point. Based on the number and type of objects placed in the instance, a question and answer pair is generated based on the relationship between two or more of these objects. The first type of question the system currently creates are based upon relative spatial information, e.g. “name the object to the left of the strawberry” or “name the closest object to the upper-right of the basketball”. The second type of question is based on the number of a particular object in an image, e.g. "name the object of which there are two visible”.

4.2 Countermeasures against attacks

A scene tagging CAPTCHA system must concern itself with two types of attacks. The first involves an attacker building a large set of possible scene tagging instances via repeatedly querying the CAPTCHA system and manually answering the associated questions. If the stored set represents a large enough portion of the possible scenes that may be created by the system, an attacker may have some success in providing correct answers by measuring the similarity between a scene presented and those in the stored set and returning an answer corresponding to the most similar image. One important defense against this attack is to enlarge the set of backgrounds and objects to a sufficiently large size, thus resulting in a number of possible background and object type and placement combinations that is simply too large to store and search efficiently. Another is the use of distortion and obfuscation to even further increase the number of possible scene instances that the system may generate. The second attack is that of machine vision based recognition of the objects in the scene. To make this object recognition as difficult as possible, the distortions and obfuscations applied to scene tagging images should have a significant effect on the lower level primitives on which machine vision based recognition generally lies. Further, they should make the process of isolation and separation of objects in a scene from the background as difficult for machines as possible.

One difference between the distortion and obfuscation scheme discussed for use with image tagging and that appropriate for scene tagging is that when applied to scene tagging, we need two different sets of distortion filters. The need for two different sets of possible distortions is due to the differing purposes served by distortions applied only to the background and distortions applied to the composite image. Distortion of the background image is primarily performed to make attacks based on machine learning of the corpus of possible background images far more difficult. If an attacker were able to construct a corpus of possible background images and perform simple matching between the value of the majority of image pixels and the corresponding background, then isolation of the object portions of the image would be a much easier task. The filter-based distortions and randomized shape generation significantly change the properties of the image background, and thus make object isolation a far more daunting task. As an added benefit, they add elements of interest to portions of the background image that may otherwise be uninteresting, e.g. a flat blue sky in a landscape image. This ensures that a flat portion of the background does not make the task of isolation of an object in that portion of the image significantly easier than it would if placed elsewhere. It should be noted at this point that taking care to preserve the semantic information contained in a background image is not necessary. While it may be demonstrated that having a coherent setting as the background image makes it easier for users to spot the object which "does not fit", our results later demonstrate that it is by no means a requirement for users to be successful in object recognition.

Distortion of the composite image, conversely, primarily attempts to make machine recognizability of objects more difficult. However, preserving the recognizability of the objects placed in the scene is of the utmost importance for the system to be able to create scene instances which humans may answer. Thus, a careful choice of distortion effects and parameter ranges must be chosen as to maximize the difficulty it causes for machine object recognition while not distort the objects in the image beyond the point of human recognition.

The choice of question format also enters into considering the strength of the system against attacks. Our system utilizes questions based on the relationship between one or more objects in a scene tagging instance. Thus, it requires the machine to be able to recognize all of the objects in this relationship in order to guarantee a correct answer.

A point to note at this point is that the solutions to preventing machine vision and data mining-based attacks do not need to be perfect; rather, we need only make machine attacks on the system require a higher cost per success than the alternative of employing human labor to perform the same task. We believe the system we present meets this goal.

4.3 Scalability

On a machine with a 2.4 Ghz Intel Core 2 Duo E6600 processor, generating a scene instance (a 640x480 scene image, question, answer choice list, and answer tuple) upon which 2 objects and 20 random shapes have been overlaid along with three effects applied takes approximately 501.1 milliseconds. However, the system was implemented in Java and no code optimization has yet been performed. It is believed that porting the system to a non-interpreted language and optimizing the image distortion functions would yield a very significant reduction in this generation time.

Ideally, a dedicated server would handle the creation and grading of scene tagging CAPTCHAs, ensuring that there is a sufficiently large pool of scene instances in reserve for instant use when necessary. This pool would allow the sites to handle times of heavy usage without CAPTCHA generation becoming a bottleneck or an attractive avenue for a denial of service attack. Alternately, given that many websites are likely to have “off-peak” periods of time in which little demand exists, it should be possible to utilize a webserver’s unused processor cycles in these times in order to maintain a sufficient reserve of scene tagging instances. It is also worth considering that with a significantly large pool size there is little risk in reusing a scene instance a small number of times if traffic demands require it, as long as these re-servings were sufficiently separated in time, not served to the same IP, and different questions were asked of the user each time.

4.4 Scene Distortion and Obfuscation Engine

In the first round of distortion and obfuscation, a number of randomly chosen distortion filters from the set of background distortion filters and parameter ranges are applied to the background image to create the base for the scene tagging image. A number of shapes of random size, type, and color/texture source are then placed over this image. The second round occurs similarly, in that a number of randomly chosen distortion filters from the set of composite distortion filters and parameter ranges are then applied to the composite image, and finally a number of shapes of random size, type, and color/texture source are then placed over this composite image. The number of effects and shapes to be applied in each round, amongst other parameters, are easily configurable.

5. EXPERIMENTAL DESIGN

We conducted experiments to empirically prove whether image and scene tagging can be used as a viable CAPTCHA alternative. For all of these experiments, we used an image corpus based on graphics collected from fotosearch.com. We compiled a total of fifty images, where each image contained a single physical object. The backgrounds of these images were made transparent to allow for future processing during scene generation.

All experiments were set up to be taken from a web site. For each experiment, we measured the average precision of the solutions submitted by the participants, as well as the standard deviation of precision. For the first experiment (see Section 5.1), we evaluated the accuracy of a response based on whether we believed the user could correctly identify the thing portrayed in the image. We did not discount cases of misspellings or of responses being more specific than we had hoped for (for example, we would accept a reply of "half of a green apple" when the answer we were simply looking for was "apple") because we believe that a real image tagging system would be able to handle such responses appropriately. However, underspecification was counted as a wrong answer; for instance, if the user was presented with a picture of an apple and answered with a tag of "fruit", then this answer would not be accepted. We restrict underspecification because allowing high-level concepts as answers would greatly reduce the solution space of image tagging since there are much fewer high-level concepts (such as "fruit") than low-level concepts (such as "apple").

For experiments with multiple choice (see Sections 5.2, 5.3, 5.4), the precision of a response is simply whether or not the user selected the correct choice.

5.1 Basic Image Tagging

In the first experiment, we were interested in determining if people are able to reliably solve image tagging problems when the image is presented unmodified (i.e. no obfuscation techniques have been applied to it) and the person must tag the image without being given any multiple choices. By not offering multiple choices, a computer would have a much harder time randomly guessing a correct answer; at the same time, we were not sure that a group of people could look at a single image and all realize what the correct image tag is.

For this test, the experiment web site randomly chose ten images from our image corpus and tasked the user with image tagging these images. The instructions for the experiment were intentionally simple: "Below each image, please identify the thing portrayed in the image." We also included a single example of image tagging a picture of a fork.

The results of the first experiment are displayed in Table 2.

Table 2: Results of Basic Image Tagging Experiment

Total Number of Participants	Average Precision	Std. Dev. of Precision
45	96.4%	0.57031

5.2 Basic Image Tagging with Multiple Choice

The second experiment was similar to the basic image tagging experiment, except that rather than having the user type in an image tag, we provided the user with multiple choices and tasked him or her with selecting the best choice. We defined the best choice as the image tag which best identifies the thing portrayed in the image. Again, we presented the user with ten randomly selected images from our image corpus, and for each of these images we generated a list of sixteen multiple choices using our multiple choice engine. We believed that the accuracy of human responses would increase if the user was presented with multiple choices.

The results of the second experiment are displayed in Table 3.

Table 3: Results of Basic Image Tagging with Multiple Choice Experiment

Total Number of Participants	Average Precision	Std. Dev. of Precision
46	99.8%	0.14744

5.3 Distorted Image Tagging with Multiple Choice

The third experiment is almost identical to the second experiment. The user was presented with ten random image tagging problems and given sixteen multiple choices for each problem. The images for these problems, however, were distorted using our distortion engine. The purpose of this experiment was to test whether humans could still solve image tagging problems if the images were more difficult to interpret.

The results of the third experiment are displayed in Table 4.

Table 4: Results of Distorted Image Tagging Experiment

Total Number of Participants	Average Precision	Std. Dev. of Precision
38	99.2%	0.358

5.4 Scene Tagging with Multiple Choice

In the final experiment, we used our scene tagging engine to generate seventy-one scene tagging problems. Each of these scene tagging problems contained either two, three, or four objects randomly chosen from the image corpus. Two types of questions were asked for this experiment. The first type of question asked the user to identify the closet object in some direction from a specified object; as an example of this type of question, "Name the closet object to the left of plunger". The other type of question asked the user to identify the object of which there are a specified number present in the image.

As described in Section 4.1, the background images and object images used to generate the scene tagging problem were distorted using various filters. Sixteen multiple choices were presented for each problem, but unlike the previous experiments these multiple choices were not generated using the multiple choice engine. Instead, the multiple choices for scene tagging problems included all objects within the scene image, as well as randomly selected terms from the image corpus.

The results of the fourth experiment are displayed in Table 5.

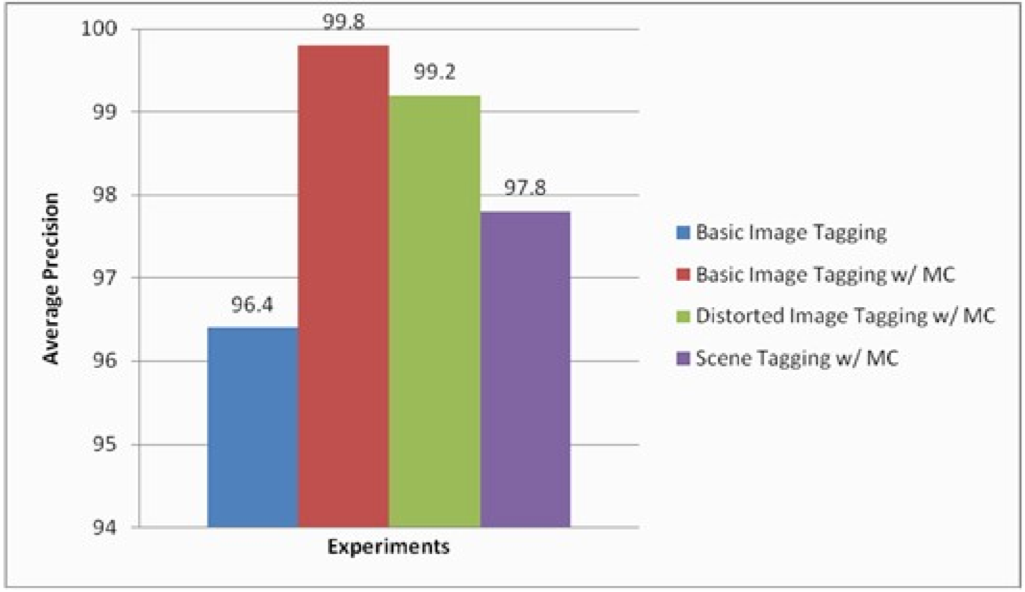
Table 5: Results of Scene Tagging Experiment

Total Number of Participants	Average Precision	Std. Dev. of Precision
37	97.8%	0.629

6. ANALYSIS

Overall, the average precision of responses for all of the experiments was high. This result in itself is enough to substantiate the first feature of image tagging, that image tagging is easy enough for a human to solve reliably. Figure 5 presents a comparison of the average precision between the four experiments.

Figure 5: Comparison of Average Precision of the Four Experiments



Of the four experiments, the second experiment (Basic Image Tagging with Multiple Choice) had both the highest precision and the smallest standard deviation. This outcome was expected since the second experiment provided multiple choices, making it easier to correctly label than the first experiment, and it also didn't use the distortion engine, making the images clearer to interpret than those in the third or fourth experiments. For the second experiment, every participant correctly answered all ten questions presented, except for one participant who labeled an image of a "soccer ball" as a "football". We believe this answer was a result of cultural bias as in countries outside the United States soccer is sometimes referred to as football.

The first experiment (Basic Image Tagging) had the lowest precision of the conducted experiments. A closer look at the questions answered incorrectly reveals that there were two major types of errors made during this experiment. The most common kind of missed answer occurred when a participant underspecified the label of an image, such as labeling an image of a "football helmet" as just a "helmet". We anticipated underspecification in the first experiment since this was the first time that participants had tried image labeling, and the instructions for the experiment didn't warn the user of underspecification. The other type of incorrect answer represented instances where the participant was unfamiliar with the image to be labeled, such as an image of "brass knuckles". This problem illustrates the importance of selecting image tagging problems that can be solvable by the general population. It also shows the value in providing multiple choices because although people may not know the exact label for an image, they can often select the correct label from a list of available choices, as shown by the high precision of the second experiment.

We were not sure how much the precision would drop for the third experiment (Distorted Image Tagging with Multiple Choice) compared to the second experiment. As noted earlier, the third experiment is the same as the second experiment except that the image tagging problems were processed by the distortion engine, making them more difficult to interpret. We were glad to see that the precision of the third experiment was almost identical to that of the second experiment. This positive result demonstrates that a web site implementing image tagging could also use a distortion engine, creating images that are more difficult for computers to recognize but are still reliably interpretable to a human.

The fourth experiment, scene tagging, consisted of much more difficult problems compared to the other experiments. Not only were the objects within the scene distorted, but there was also a background image to the scene (which, in turn, was distorted). Furthermore, unlike the other experiments, each scene tagging problem was accompanied by a unique question that required the participant to identify more than one object within the scene in order to correctly solve the problem. The precision for this experiment was still very high, and since this is the most secure of the discussed techniques due to its increased complexity, we believe that scene tagging is the best of our explored systems. Of the incorrect responses collected from the participants, the most common mistake was to confuse a part of the background as being part of the foreground. Circumventing this misconception in the future may require adjusting the amount of distortion applied to the background image. Perhaps the selection of background images may also need further consideration to avoid any background that could be confused with the foreground.

Although we have already discussed the trade-off between providing or not providing the user with multiple choices, we believe that the results of the experiments support the usage of multiple choice solutions. Not only will this alleviate the overhead to computationally interpret a manually crafted response, but it also helps in situations where a user may somewhat recognize an image but not fully remember the correct label for it -- when presented with multiple choices, humans seem to often be able to correctly label an object with which they only have a passing familiarity.

7. CONCLUSIONS

In our paper, we have shown that an image tagging-based CAPTCHA system can benefit, both in terms of attack resistance and user-friendliness, from the use of WordNet in considering the range of answers that should be accepted and the optimal set of multiple choice selections. We have also considered a range of likely attacks that such a system would face and make the case that proper use of distortion and obfuscation would yield greater attack resistance, while user study results show that user success and the user experience are largely unaffected by careful use of such measures. For greater attack resistance without sacrificing user-friendliness, we propose a novel form of image-based CAPTCHA we term scene tagging. This system utilizes a question format based on relationships between objects in an automatically generated composite image. We argue that it is best to utilize multiple rounds of image distortion and obfuscation during the composition process, and that when this is done the system is strongly attack-resistant. User study results show a very high success rate of humans in solving problems generated by our scene tagging CAPTCHA system, suggesting that attack-resistance measures such as image distortion and the question format utilized have a minimal impact on the user experience. We thus propose that it is both suitable for consideration in real-world deployment and worthy of future study.

7.1 Future Work

One draw-back of the system presented is that it may present difficulties to users by whom English is not spoken or for whom English is a second language. In a multiple-choice context, it should be fairly simple to utilize information from WordNet (such as part of speech and word meaning sense) in combination with an automated language translation engine to provide choices in a number of common languages as necessary. A more difficult problem is that of cultural bias; users' cultural knowledge and experiences may mean that a number of the objects in the object database are foreign concepts to them. Our results demonstrated this in users' having trouble identifying "brass knuckles". This problem applies to the vast majority of image-based CAPTCHAs that have been proposed, and thus is surely a matter that deserves future consideration.

It is important to note that most, if not all, images on the Internet are subject to copyright laws. Accordingly, a real implementation of an image tagging system cannot legally collect images from various sites without expressed permission. The creation of an appropriate corpus of image-object pairs and background images is orthogonal to our study, but it would be beneficial for future works to investigate feasible ways to lawfully collect images to be used in future image-based CAPTCHA systems.

Our system could be improved upon by the investigation of alternative question types and formats, such as the drawing of a line connecting the center of three specified objects. Likewise, a comprehensive study of the leading machine image recognition/tagging techniques in the face of strong, adversarial distortion and obfuscation would be of great worth in designing future systems. In conjunction with an extensive quantification of the ability of human users to identify images/objects in the face of distortions and obfuscation of diverse form and strength, it would allow system designers to utilize a set of such distortions that maximize the impairment of machine recognition while minimizing the impairment of human recognition.

8. REFERENCES

[1] Lecca, M., Messelodi, S., and Andreatta, C. 2007. "An object recognition system for automatic image annotation and browsing of object catalogs". In *Proceedings of the 15th international Conference on Multimedia* (Augsburg, Germany, September 25 - 29, 2007). MULTIMEDIA '07. ACM, New York, NY, 154-155.

[2] Wu, W. and Yang, J. 2006. "SmartLabel: an object labeling tool using iterated harmonic energy minimization". In *Proceedings of the 14th Annual ACM international Conference on Multimedia* (Santa Barbara, CA, USA, October 23 - 27, 2006). MULTIMEDIA '06. ACM, New York, NY, 891-900.

[3] Carbonetto, P. and de Freitas, N. 2003. "Why can't José read?: the problem of learning semantic associations in a robot environment". In *Proceedings of the HLT-NAACL 2003 Workshop on Learning Word Meaning From Non-Linguistic Data - Volume 6* Human Language Technology Conference. Association for Computational Linguistics, Morristown, NJ, 54-61.

[4] 2007. "Asirra: a CAPTCHA that exploits interest-aligned manual image categorization". In Proceedings of the 14th ACM Conference on Computer and Communications Security (Alexandria, Virginia, USA, October 28 - 31, 2007). CCS '07. ACM, New York, NY, 366-374.

[5] M. Chew and J. D. Tygar, “Image Recognition CAPTCHAs”, In Proc. of the 7th Annual Information Security Conference (ISC’04), pp. 268–279, Palo Alto, CA, September 2004.

[6] Deapesh Misra and Kris Gaj. "Face Recognition Captchas". In Telecommunications, 2006. AICT-ICIW '06. International Conference on Internet and Web Applications and Services/Advanced International Conference on, page 122, Washington, DC, USA, February 2006.

[7] Yong Rui and Zicheng Liu. "Artifacial: Automated Reverse Turing Test Using Facial Features". Multimedia Systems, 9(6):493-502, June 2004.

[8] Mohammed E. Hoque, David J. Russomanno, and Mohammed Yeasin. "2d Captchas from 3d Models". Proceedings of the IEEE SoutheastCon, pages 165-170, April 2006.

[9] Henry S. Baird and Jon Louis Bentley. "Implicit Captchas". Proceedings of the IST SPIE Document Recognition and Retrieval XII Conference, volume 5676, San Jose, CA, USA, January 2005.

[10] Philippe Golle. "Machine Learning Attacks Against the Asirra Captcha". Proceedings of the 15th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, October 2008.

[11] Ritendra Datta, Jia Li and James Z. Wang, ``IMAGINATION: A Robust Image-based CAPTCHA Generation System," Proceedings of the ACM Multimedia Conference, pp. 331-334, Singapore, ACM, November 2005.

[12] G. Mori and J. Malik, "Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA", Proc. IEEE CVPR, 2003.

[13] G. Moy, N. Jones, C. Harkless, and R. Potter, "Distortion Estimation Techniques in Solving Visual CAPTCHAs", Proc. IEEE CVPR, 2004.

[14] Kumar Chellapilla and Patrice Y. Simard. "Using Machine Learning to Break Visual Human Interaction Proofs (HIPs)". In Lawrence K. Saul, Yair Weiss, and Leon Bottou, editors, Advances in Neural Information Processing Systems 17, pages 265-272, Cambridge, MA, December 2004.

[15] Jeff Yan and Ahmad Salah El Ahmad. "A Low-Cost Attack on a Microsoft CAPTCHA". In Proceedings of the 15th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, October 2008.

[16] Henry S. Baird and Terry Riopka. "Scattertype: A Reading Captcha Resistant to Segmentation Attack". Proceedings of the IST SPIE Document Recognition and Retrieval XII Conference, volume 5676, pages 197-207, San Jose, CA, USA, January 2005.

[17] M. Shirali-Shahreza and S. Shirali-Shahreza. “Collage CAPTCHA,” Proceedings of the 20th IEEE International Symposium Signal Processing and Application (ISSPA 2007), 2007.

Table 6: Tools and Other Data Used to Develop and Run the System

Tool/Data	**Purpose**	**Source**
Eclipse	Java IDE	http://www.eclipse.org/
fotosearch.com	Collection of images used for our image corpus	http://www.fotosearch.com/
Java (1.5 & 1.6)	Programming language	http://java.sun.com/javase/downloads/index.jsp
Java WordNet Interface (JWI)	Access WordNet from Java	http://projects.csail.mit.edu/jwi/
JHLabs Java Image Filters	Java image filters used for image distortion	http://www.jhlabs.com/ip/filters/index.html
NetBeans	Java IDE	http://www.netbeans.org/
WordNet 3.0	Estimate size of image tagging solution surface Used by multiple choice engine	http://wordnet.princeton.edu/