Automated Collection of High Quality 3D Avatar Images

James Kim, Darryl D'Souza, Roman V. Yampolskiy

Computer Engineering and Computer Science
University of Louisville, Louisville, KY
jhkim012@louisville.edu, darryl.dsouza@louisville.edu, roman.yampolskiy@louisville.edu

Abstract CAPTCHAs are security tests designed to allow users to easily identify themselves as humans; however, as research shows (Bursztein et al. 2010) these test aren't necessarily easy for humans to pass. As a result a new test, which requests users to identify images of real humans among those of 3D virtual avatars, is proposed that would create a potentially unsolvable obstacle for computers and a quick, easy verification for humans. In order to provide test cases for this new test, an automated bot is used to collect images of 3D avatars from Evolver.com.

INTRODUCTION

Today's commonly used security test distinguishing real humans from bots on the Internet is CAPTCHA. Two noticeable issues result from using such tests. One is that these tests are gradually becoming less reliable as exposed design flaws and improved algorithms successfully crack CAPTCHAs. For example, a CAPTCHA cracking program called deCAPTCHA, which was developed by a group of Stanford researchers, successfully decrypted image and audio-based text CAPTCHAs from a number of famous sites. The success rate of passing CAPTCHAs varies from site to site, but some examples include a 66% success rate on Visa's Authorize.net, 43% on Ebay, and 73% on captcha.com (Bursztein, Martin, and Mitchell 2011). Each website uses a combination of different text distortion factors in their CAPTCHAs; however, it was evident that design flaws were present in most tests. For example, the deCAPTCHA program was able to achieve a 43% success rate because the researchers were able to exploit the fact that eBay's CAPTCHAs were using a fixed number of digits and regular font sizes (Bursztein, Martin, and Mitchell 2011).

Second, according to research, humans encounter difficulties in solving CAPTCHAs, especially audio-CAPTCHAs. Perhaps the tests are becoming more challenging for the human users than they are for computers, considering deCAPTCHA's performance (Bursztein et al. 2010). This test is designed to be an unsolvable obstacle for computers while remaining easy for humans. One may even hypothesize that there may be a correlation between the combination of factors that determine a CAPTCHA's insolvability to computers and the level of ease for a human to solve the test.

Other simpler more novel and user friendly types of CAPTCHAs do exist, such as the Animal Species Image Recognition for Restricting Access (ASIRR) or the 'drag and drop' CAPTCHA (Geetha and Ragavi 2011). However text-based CAPTCHAs are more widely used than sound or image based tests because of the numerous variations of distortion methods to increase the level of insolvability to computers, despite the added difficulty for humans to pass them (Geetha and Ragavi 2011). Each test has its own design flaws, thus security tests that present the highest level of insolvability to computers and are easiest to solve for humans are sought.

Computers have the potential to detect text from a variety of distorted images, but their ability to differentiate between two objects in different colors or textures is a different matter entirely. Considering this as a foundation, a test was proposed that would create a small table using images of virtual avatars and real humans and ask the user to select those that are of real humans. Generally, the test would include two or more images of real humans in order to increase the difficulty for computers.

In addition, the colors, textures, and other factors of the images would be variable in order to create as few viable variables as possible. For example, the images of virtual avatars or humans may be black and white, pastel, or even sketched. Furthermore, various clothing and accessories would potentially add more to the challenge. Inevitably, the test would require an algorithm that demands critical thinking, one that answers the question "How does a real human appear?"

In order to begin testing, a dataset of high quality 3D virtual avatars was required. Virtual communities such as Second Life or Active Worlds provided an extensive selection of body characteristics and articles of clothing for avatar creation. The tradeoff for extensive customization was the detail or quality of the avatars. Considering that both these virtual communities in particular have massive online communities of users, it's understandable that the system requirements to run the client application are very low since hardware costs are often a barrier to accessing online content.

Evolver, which is a 3D avatar generator that allows users to create and customize their own avatar, was found to be the best choice. The avatars were highly detailed and elaborate with options to orient them into certain body positions, facial expressions, or animations. Although Evolver did not possess the extensive selection of avatar characteristics as the other virtual words, its selections were large enough for our purposes.

In addition, one noticeable difference between Evolver and the other virtual worlds was the option to morph different physical attributes together. Evolver uses a large database of pre-generated body-parts called a virtual gene pool to morph together two physical attributes in morph bins. Users choose the degree of which they are morphed together by using a slider, which measures the domination of one attribute over the other (Boulay 2006).

The collection of avatar images was collected using Sikuli, which is an image recognition based scripting environment that automates graphic user interfaces (Lawton 2010). Sikuli's integrated development environment features an image capturing tool that allows the users to collect the images required to fulfill the parameters of unique functions.

A Sikuli script could be run continuously, but there is always some factor, such as server latency, that could cause an error, which would cause a major setback to data collection as the script would terminate. As a result, Evolver's simple and easily navigable user interface was another reason why Evolver was selected over *Second Life*, *ActiveWorlds*, and other virtual communities. Evolver's interface divides physical attributes and articles of clothing into organized tabs, which would decrease the possibility of failure when moving from one section to another.

ActiveWorlds in particular relied on scrolling, which at times significantly impeded data collection because the possibility of failure would increase when the script failed to scroll down to continue with the script. Generally, few images were able to be generated without the Sikuli script misrecognizing certain images. However, in any case of data collection of avatar images from the web, server latency is a huge factor in successful collection as it could delay the timing for actions dictated in the script.

METHODOLOGY

The Evolver avatar customization interface is slightly more unique in that it prioritizes displaying explicit details of a particular attribute than emphasizing the overall variety in a section. This visual design explains why when a mouse hovers over an attribute; the image automatically enlarges so that the user would have a better visual. Some physical attributes with variations, such as varying colors or patterns, show gray arrows on each side of the image when enlarged, which allows the user to rotate through the variations instead of viewing all of them in a grid on a reloaded webpage.

At its execution, the script would randomly click either a male or female gender option to begin the avatar creation process. Once the webpage loaded the Face tab, the script would make sure that both checkboxes for male and female physical attributes were selected so as to open more choices for a truly random avatar. Seven icons were present on the left side, each of which focused on a specific area of the face (Figure 1). In each area, a slider exists to enable the user to control the degree of attribute morphing. Initially, the random die button is clicked first to obtain two random face presets, and the slider would be dragged to a random location on the line to correlate a random degree of morphing. The script would click 65 pixels down from the center of the first selected icon in order to randomize the degree of morphing for each area.

The script would proceed to the Skin tab on which there is a grid that showed all the physical attributes for the skin. Under each tab there is a specific number of pages of attributes. As a result, the script would randomly click through the pages given the range of the pages, and would randomly select an image "cell" within the grid. Under most of the tabs that share a similar grid design, there would be a region that the script would act within to select an image. The region is the area defined by the x-y coordinate of the top left corner, and the dimensions

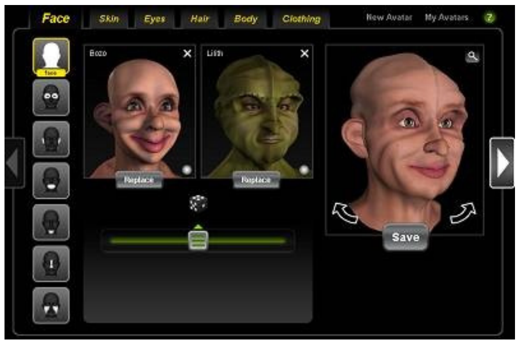

Figure 1. Face Tab(Evolver.com)

of the rectangle. Each tab had a differently sized region because of the varying dimensions of the physical attribute images.

Essentially, one attribute image within the grid was considered as a cell, and after randomly selecting a grid page, a random cell was chosen by moving the mouse cursor to the center of the cell. In order to confirm that the image has enlarged and that there was an image located at the cursor coordinate, a yellow box with the text "Selected," which appears in the bottom right corner of the image, was scanned on the screen. Other cases were also resolved, such as the issue when the script failed to recognize that the mouse was hovering over the image given the default waiting period. Once the selected box is confirmed, the script checks to see if there are variations for the attribute by scanning for an unselected gray arrow on the right side of the image. If there were variations for that attribute, the script would quickly rotate through the variations given a random number of clicks. It is worth noting that viewing images on each page or the variations through rotation respond quickly, which suggests that these operations are run client-side. Once the random image is selected, the script would proceed to the next tab. Because of the similarity of the Eyes, Hairs, and Clothing tab with the Skin tab, the same method of collection was used for each.

After the random hair selection, an image of the avatar's face would be captured in PNG format on a preview window, opened by the zoom button in the preview. The resolution used to capture the image depended on the region dimensions. The avatar was rotated once toward the left side in order to capture a direct face-to-face image of the avatar. For the Body tab, the same method of image selection used in the Face tab was also applied in this tab because of the similarity between the interface of the Body tab and the Face tab.

After the images for the clothing tab, which contains sub-tabs of the top, bottom, and shoe tabs, the image of the avatar's body was captured. However, instead of the default position, which is a slightly angled arms out position, a face-to-face stand position was chosen in the dropdown box in the zoomed preview window. Finally, the script would select "New Avatar" located near the top right hand corner of the web page in order to repeat the process.

RESULTS

The program was run as continuously as possible over the course of seven days, and approximately 1030 successful images were captured, more of which were of the avatar body than the avatar face. The initial days were primarily test runs that were used to find and resolve unforeseen bugs. There were more than 100 face images that were unsuccessful, yet the script managed to continue with the process and collect other potentially successful samples. One interesting point to note is that there were a significantly larger number of unsuccessful images of faces than there were of bodies, which totaled about 10-20 errors at most. This is primarily because the method for adjusting the avatar's face was more problematic than that of the avatar's body. Adjusting the avatar's head relied on clicking 'rotate left arrow' icons, which were often ignored by the server. Some of these major problems are longer load times or Sikuli's failure to recognize loading signals.

The source of most of these errors was primarily due to server latency. Between every selection in the avatar customization process, there is a variable loading time and a loading box, which appears in the center of the screen. Surprisingly, the loading time differs for each attribute image applied to the avatar, not necessarily differing for one type of physical attribute. For example, the loading time for two different but similarly categorized physical attributes would vary. One could conjecture that the amount of polygons, depending on the attribute, added upon the avatar may cause the variable loading time.

These problems haven't occurred very frequently, but their potential occurrences require hourly checkups else the script could fail if left unchecked. Overnight data collection was risky because of the lack of checkups, but in the cases that they were successful, a partial number of defunct images were present in the dataset. For example, previews of avatars that still have the loading text in the center of the preview.

However, the other problem that exists is executing the script on a different computer. Because of the differing monitor resolutions, adjustments to the program were required. The minimum changes made were of the regions under the each tab because each has a defined upper left coordinate that is correct only in the previous computer's resolution. The coordinate as a result must be redefined because it would throw off the entire collection process.

CONCLUSION

Besides the unsuccessful captured images, there were very rare occurrences where abnormal morphing would occur (Figure 6). However, server latency was a higher concern, since images were often captured too early (Figure 5 and 8), because the loading text image wasn't detectable when it was overlaid in the center of the avatar preview. As a result, the program would instead rely on detecting when the black area within the loading box disappears before undertaking any action. However, instructions to rotate the avatar head weren't recognized, and as a result most unsuccessful captured face images were in the default angle (Figure 3). Comparatively, changing body positions were mostly recognized by the server resulting with more successful body images (Figure 7) than face images (Figure 4).

Collected avatar images from previous research are of relatively lower quality than of those in this data set. This is primarily due to the implementation of low graphics requirements to open the virtual community to as many online users without hardware being a limiting factor. Although low graphics result with basic avatar models, simple facial expressions can still be easily noticed and identified (Parke 1972). In more complex avatar models, such as those in this data set, there are thousands of more polygons that allow more complex expressions to be shown.

Furthermore, gathering the data from virtual communities is incredibly tedious, considering there are a number of other problems that exist, such as involuntary self-movement of the avatar (Oursler, Price, and Yampolskiy 2009). For example, the head or eyes of the avatar would always shift in place, and often times the captured image wouldn't be in the position displayed by the avatar in Figure 2. Considering comparing Figures 2 and 4, the images in this data set are of higher quality and of larger resolution.

Overall, this dataset could be repurposed not only for imaged-based CATPCHA tests, but for facial expression recognition and analysis. With the addition of thousands of polygons, creating more complex facial expressions is possible. As such, biometric analysis could be performed on such avatars as a precursor to the analysis of real user biometric data.

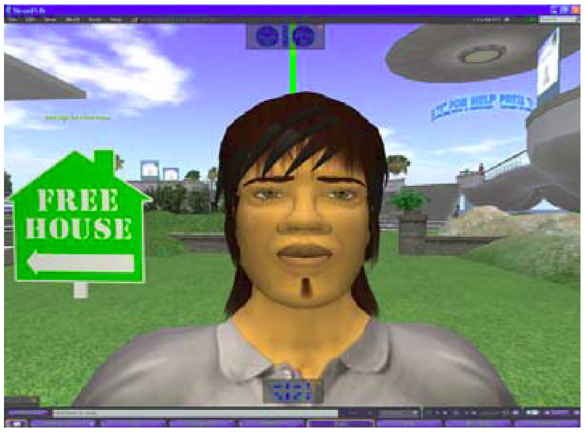

Figure 2. Avatar image captured from Second Life

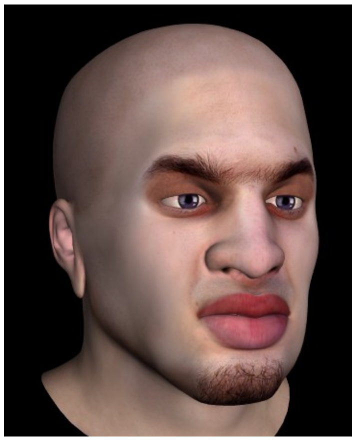

Figure 3. Unsuccessful Captured Face Image - Default Angle Unchanged

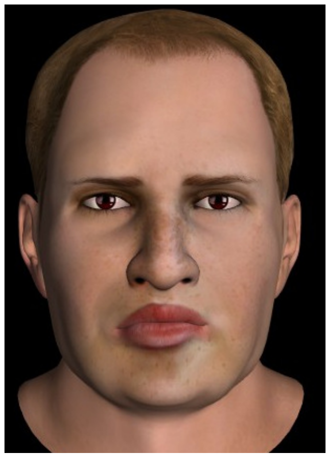

Figure 4. Successful Captured Face Image

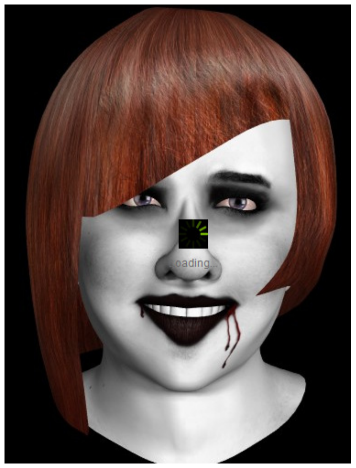

Figure 5. Unsuccessful Captured Face Image - Loading Box in Center

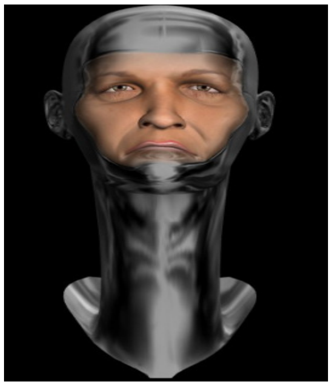

Figure 6. Unsuccessful Captured Image – Abnormal Morph

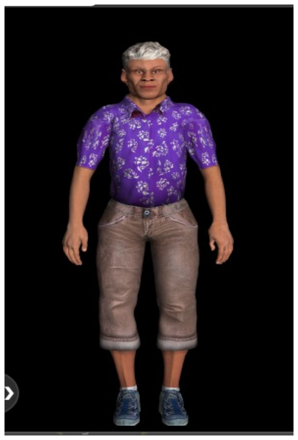

Figure 7. Successful Captured Body Image

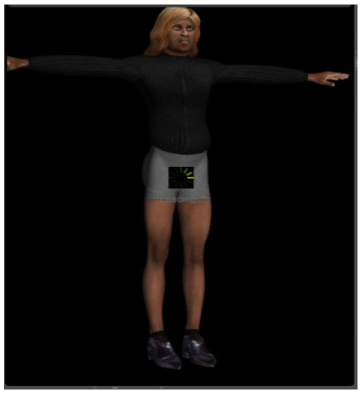

Figure 8. Unsuccessful Captured Body Image – Loading Box in Center

References

Boulay, Jacques-Andre. (2006) "Modeling: Evolver Character Builder." Virtual Reality News and Resources By/for the New World Architects. http://vroot.org/node/722. Retrieved on February 21, 2012.

Bursztein, Elie, Matthieu Martin, and John C. Mitchell. (2011) "Text-based CAPTCHA Strengths and Weaknesses" ACM Conference on Computer and Communications Security (CSS'2011). October 17-21, 2011. Chicago, USA.

Bursztein, Elie, Steven Bethard, Celine Fabry, John C. Mitchell, and Dan Jurafsky. "How Good Are Humans at Solving CAPTCHAs? A Large Scale Evaluation. " *2010* IEEE Symposium on Security and Privacy. pp. 399-413. May 16-19, 2010. Berkeley, CA.

Geetha, Dr. G., and Ragavi V. "CAPTCHA Celebrating Its Quattuordecennial – A Complete Reference." IJCSI International Journal of Computer Science Issues 2nd ser. 8.6 (2011): 340-49. Print.

Lawton, George. (2010) "Screen-Capture Programming: What You See Is What You Script. "Computing Now. IEEE Computer Society, Mar. 2010. Retrieved on February 22, 2012. <http://www.computer.org/portal/web/computing now/archive/news054>.

Oursler, Justin N., Mathew Price, and Roman V. Yampolskiy. Parameterized Generation of Avatar Face Dataset. 14th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGames'09), pp. 17-22. Louisville, KY. July 29-August 2, 2009.

Parke, Frederick I. "Computer Generated Animation of Faces." ACM '72 Proceedings of the ACM Annual Conference. Vol. 1. NY: ACM New York, 1972. 451-57. Print.