A Dual-Layer Clustering Scheme for Real-Time Identification of Plagiarized Massive Multiplayer Games (MMG) Assets

W. Raffe, J.Hu, F. Zambetta, K. Xi
School of Computer Science and IT
RMIT University
Melbourne 3001, Australia
(wraffe@cs.rmit.edu.au), (Jiankun.Hu, Fabio.Zambetta, Kai.Xi@rmit.edu.au)

Abstract**—***Theft of virtual assets in massive multiplayer games (MMG) is a significant issue. Conventional image based pattern and object recognition techniques are becoming more effective identifying copied objects but few results are available for effectively identifying plagiarized objects that might have been modified from the original objects especially in the real-time environment where a large sample of objects are present. In this paper we present a dual-layer clustering algorithm for efficient identification of plagiarized MMG objects in an environment with real-time conditions, modified objects and large samples of objects are present. The proposed scheme utilizes a concept of effective pixel banding for the first pass clustering and then uses Hausdorff Distance mechanism for further clustering. The experimental results demonstrate that our method drastically reduces execution time while achieving good performance of identification rate, with a genuine acceptance rate of 88%.*

Keywords-component; patter recognition, clustering, MMG security)

I. Introduction

In recent years, interactive media has become a multi-billion dollar industry [1], with global video game sales rivaling other top contending media such as movies and music. A subsection of these games are ones that can be played over the Internet. Multiplayer online games (MOGs) present a virtual realm in which players can play together, compete against each other, or simply provide a new means for friends to communicate and express themselves all through the use of Internet based technology.

Massively multiplayer games (MMGs) are yet an even stricter classification. MMGs are becoming increasingly popular and more software developers are investing in creating their own virtual worlds. It is not uncommon to see online communities with populations in the thousands, sometimes even hundreds of thousands, subscribed to a single virtual world at any one time. This raises a few opportunities and challenges. With such large populations, there exist business potentials that as of yet have been largely neglected. At the same time, there exist increasing security concerns, such as those outlined by Jiankun and Fabio [1],

The theft of assets in a virtual world is of pressing concern. MMGs are typically categorized as having at least some small form of persistence which remains in the virtual world even after the player has logged out of the system. This can also be seen as players having ownership over certain aspects of the game. Game characters and objects, virtual currency, and virtual real-estate, along with user created content such as 3D models, 2D textures, programming scripts, and audio are all examples of persistent objects that may exist in or be uploaded to a game server and may all be grouped under the title of *virtual assets.* The real world value of virtual assets can be surprisingly high and with the advent of real-money trade (RMT) [2] it is possible to acquire economic gain though the sale of these assets. It is evident then that measures must be taken to safeguard these assets from theft and other malicious acts.

In our research we focus on protecting 3D models that may be created by a user and shared through a multiplayer game. These assets are typically hard to protect because in order for a 3D game object to be displayed in the virtual world, all data that defines the objects construction must be shared with all players who may be able to see it. Thus, it is easy to capture this data, either during communication of the model or throughout the graphical pipeline, and then replicate the 3D model for redistribution. An active example of this issue has occurred within the virtual world Second Life [3] and has robbed individuals and businesses of economic gain. Our goal is to aid in the detection of this copying technique. We hope that when a new 3D object is introduced to the virtual world, it can be checked against all other existing 3D models in the game; if the new model is too similar to an existing one and cannot be considered unique, and then its creation is denied or reported for inspection by an administrator.

Pattern recognition and clustering techniques are potentially suitable for addressing the above issue. Some clustering techniques already exist for use in content based image retrieval systems. These techniques are generally known as statistical classification methods [4] which use visual features to categorize images. Some examples of these are the SemQuery system [5] that analyzes images based on heterogeneous features such as texture and color, Vailaya et. al [6] and the SIMPLIcity [7] system that both reduce the search space of images in specific databases into broad logical groups based upon the context within an image and the composition of image, and the ALIP [8] that is designed to work with multiresolution Markov models. Our proposed method is different in that we do not use image features or human defined categories but rather we group images based on size and shape. In our method we first count effective pixels in an image in order to group objects of roughly the same size, which is explained in Section III, and then we use Hausdorff Distance to cluster objects of similar shape, as described in Section IV.

Our clustering algorithm is designed to protect 3D models that can be claimed as virtual assets in a virtual world or that creators feel they could have some form of copyright over. The most developed field that this process can be related to is content based 3D model retrieval systems. Working within the framework of a single game allows for certain assumptions to be made about the rotation, scale and standards of the 3D models and our system has been tailored to work in that environment. We consider the case of 2D projection of a 3D object based on the following considerations: (i) As rotation and scaling are synchronized in our environment, if two 3D objects are similar, their 2D projections should also be similar. Also, 2D projections are the dominate visual features in this games environment. (ii) Projecting a 3D object into 2D can simplify the problem significantly by allowing us to work with pixel information rather than 3D mesh vertex data. (iii) Detection of plagiarizing arts article cannot entirely depend on technology. In many cases an expert's view is required. Therefore we aim to provide a narrowed list of base objects that are similar to the object to be detected. We develop our proposed system and use a database of roughly three thousand 3D models to test its effectiveness. Through our experiments we received a genuine acceptance rate (GAR) of 88%, which indicates that, given a suspect image, the original 3D model that it is plagiarizing can be found within our clustering structure 88% of the time. We also demonstrate that the clustering reduces the number of images that are tested for similarity to a suspect image and also that the clustering can even improve the ranking of original images that the suspect may be a plagiarism. Please provide your experimental setup and results here. The organization of the remaining paper is as follows. Section II is to introduce our dual-layer clustering algorithm. Section III and Section IV provide details of the two clustering algorithms used. Section V presents the experiments we have undertaken and the results gathered. Finally, Section VI gives a brief conclusion and a look at what future work is required.

II. DUAL-LAYER CLUSTERING ALGORITHM

A. Overview of Algorithm

In our algorithm we have two major stages, build time and search time. Build time is done offline and therefore we are not concerned with its efficiency. Search time is considered as being online and may be expected to be executed in a real time system; therefore we not only look for combined effectiveness

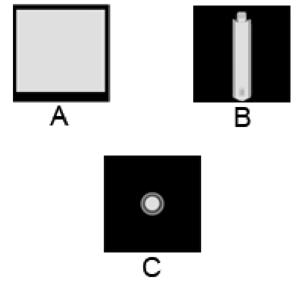

Figure 1. Three different 3D models projected into 2D images

with the build time algorithm but also the individual efficiency of this stage. The overall structure of our final system is as follows:

- Build Time
  - Get 2D projections of all dataset objects
  - Downsample dataset images
  - Divide dataset into effective pixel Bands
  - Divide Bands into Hausdorff Distance Clusters (HD Clusters) by selecting representatives and using Hausdorff Distance on them
- Search/Run Time
  - Get 2D projection of search object
  - Downsample search image
  - Select effective pixel Band(s) that the search image falls into
  - Select HD Cluster(s) that has the best Hausdorff Distance value(s) between the search image and the HD Cluster's representative.
  - Calculate desired pattern recognition metric between search image and dataset images in the HD Cluster

B. Gathering 2D Projections and Downsampling

The process of gathering a 2D projection is used to get pixel data from a 3D game model. We do this simply by rendering the 3D model in an empty black environment and then, rather than displaying the object onto a monitor, we save the pixel data into a popular image format such jpeg or bitmap.

After capturing a 2D image of the 3D model, we are then able to downsample it if need be. Downsampling is a process of lowering the resolution of an image. The algorithm we used for this was, for a sample rate of n, we keep every n^{th} pixel, starting with the first element of the pixel matrix, and dispose of all pixels in between. We first do this along the y-axis and then along the x-axis. We use this process to reduce the complexity of the images and increase the performance of the Hausdorff Distance calculations. In early experiments, we tested the difference between higher resolution images and downsampled. For the experiments in Section V we use images of the size 64x64 pixels as we found that this resolution gave a good balance between detail being retained in the image and efficiency of processing during the Hausdorff Distance calculations. This downsampling step may not be needed if the images are already at a suitable resolution. Alternatively, if all images are at different resolutions, this step can be generalized to resolving the resolutions of all images to a common size.

III. *First Round Clustering: Banding*

A. Banding Defined

The first stage of clustering is to separate images based on effective pixels. We call these clusters 'Bands' and they are used to group images that contain objects with similar proportions. Our definition of effective pixels is any pixel with a value greater than zero. All of the 2D projections in our implementation have black backgrounds; therefore an effective pixel is part of the model shown in the image.

When getting 2D projections of 3D models, we are able to enlarge or minimize all models to fit a coherent scale. However, this does not mean that the proportions are the same. For example, after being scaled to a coherent range, two objects may have the same height but that does not mean they will have the same width. Therefore, a thin object will have a lot less effective pixels than a thick object when projected into a 2D image. This is demonstrated in Fig. 1. Object C in Fig. 1 has the same scale as objects A and B but is longer in the z-axis and very thin in the x and y-axis. By clustering images into effective region bands, we perform an initial, efficient, and somewhat obvious check to make sure the models are of roughly the same proportions, preventing models that are thin in the x and y dimensions from being compared with those that are large in these dimensions.

For counting effective pixels, we simply use a binary check. If a pixel has a value greater than one, then we add one to a counter. Thus, for an image of size 64x64 pixels, an all white image will have a value of 4096 and an all black image will have a value of zero.

Effective pixels do not give any indication to the shape or detail of the image. For example, two completely different objects, such as an image of a car and an image of a human, may in fact have the same number of effective pixels, though they are clearly different objects to the human eye. This is why we use a second layer of clustering which is described in Section IV.

B. Creating and Searching Bands

When creating and searching the Bands we use the equation

$$P \pm (P * \Delta) \qquad (1)$$

where P is the number of effective pixels that was responsible for creating the Band and Δ is a percentage value that determines the range of the Band.

When building the Bands from a given dataset, we randomly choose a starting image, count its effective pixels, and then create a new band with the range specified by the equation above. When we add a second image, we again count its effective pixels and see if it lies within in the range of the existing band. If the second image does lie within this range, then it is added to this Band, otherwise a new Band is created. These steps are repeated, comparing the third image to the Bands created by the first and possibly second image.

For example, a Δ value of 20% is chosen. An initial image with an effective pixel count of 1000 creates a new Band. A second image with an effective pixel count of 500 is then added. As 500 does not lie in the range (900 – 1100), a new Band is created around the second image. The Bands that were used for the results in this paper were created and searched using a Δ value of 20%.

The above equation that we have chosen allows for simple overlapping of the Bands. That is, if a dataset image can fall into two Bands then it is placed in both and if a search image could belong in more than one Band, then both bands are searched. This gives a benefit over using Bands with fixed sized ranges, which cause issues with boarder cases.

When utilizing the Bands during our Search Time we follow the same equation as above; gathering the effective pixel count of the search image and evaluating it against that of the Bands. If a corresponding Band is identified, then we continue into this Band proceed to use our second round clustering method.

IV. *Second Round Clustering: Hausdorff Distance Clustering (HD Clustering)*

A. Hausdorff Distance Background

Hausdorff Distance is a shape discovery and matching algorithm commonly used in computer vision. At its core, Hausdorff Distance is used to determine the distance between two polygons on a coordinate plane. When used in image processing, however, we can think of it as overlaying the two shapes on top of one another and then determining their similarity by calculating the distance between the pixels on one shape and pixels on another. The two images used in a comparison should be the same resolution and the 3D models should be centered within the image space. We then calculate the edges of the model in the image by using an edge detection algorithm such as Sobel [12]. This is done so that we do not need to consider every effective pixel in the image, rather just the pixels that define the edge of the model, reducing computation time significantly. We then calculate pixel distance from each point of one model to each point of another and follow the Hausdorff Distance algorithm. Thus, if the two images are exactly the same, then all of the points identified in each image will be in the same pixel positions, resulting in distances of zero for every point and therefore resulting in a Hausdorff Distance of zero.

Other edge detection algorithms, such as Canny [11] can be used if the images used have high amounts of noises or if a higher granularity of detail in the model is to be detected. Fig. 2 shows how the Canny edge detection algorithm produces less detailed edges. Fig. 2 also shows how downsampling not only reduces the resolution of the image but also simplifies the edges as well, thus creating less points for a Hausdorff Distance algorithm to calculate. There also exist approaches that gather Hausdorff Distance by utilizing line segments [10] rather than points. This can increase time efficiency as some shapes will have very few lines but many pixel points that those lines pass through.

For our experiments throughout this paper though, we use the basic point based Hausdorff Distance. However, as final point of mention, we have modified the basic Hausdorff Distance measure by getting the average distance from A to B and B to A, rather than taking the maximum. Equation 1 shows how our final Hausdorff Distance calculations are computed.

$$H(A,B) = mean(h(A,B), h(B,A)) \qquad (2)$$
$$\text{where } h(A,B) = \max_{a \text{ in } A} (\min_{b \text{ in } B} (||a-b||)$$
$$\text{where } ||a-b|| \text{ is the distance between two points } a \text{ and } b$$

For our dataset we found that this provided more evenly distributed and logically correct HD Clusters. Choosing the maximum rather than the average causes one direction of Hausdorff Distance testing to be chosen to represent the total distance and discards the other value, which we felt did not accurately represent the relationship between two images for our dataset.

B. Creating the HD Clusters

To create the HD Clusters we use Hausdorff Distance during build time, immediately after we have completed the construction of the Bands. The first step in creating the HD Clusters is to randomly choose an image in a Band, this will be the representative of our first HD Cluster. We then use this representative as a search image of sorts, comparing it to every other image within the Band. Any image that is less than σ distance to the representative is grouped into this initial HD Cluster. The image that has the furthest distance from our first representative is chosen as our second representative and the process is repeated. Again, the image that is furthest away from our second representative is chosen to be our third. We should point out that a representative cannot exist in any other cluster. For example, for a set of distance values *X*, a new representative will be at position *max(X)*. If the corresponding image has already been placed in a HD Cluster then we use *max(X)-n,* increasing the value of *n* until we find the index of an image that has not already been placed in a HD Cluster or until the value at position *max(X)-n* is less than σ, indicating that all images in this Band have been placed in a HD Cluster.

While an image cannot become a representative if it has already been placed in a HD Cluster, the image can exist as a member of multiple HD Clusters. It can even be a representative of one HD Cluster and a member of another, if and only if it was chosen as a representative before it became a member of another HD Cluster. This ensures that the HD Clusters properly encapsulate all images in the Band that are similar to their respective representatives and allow for overlapping of HD Clusters.

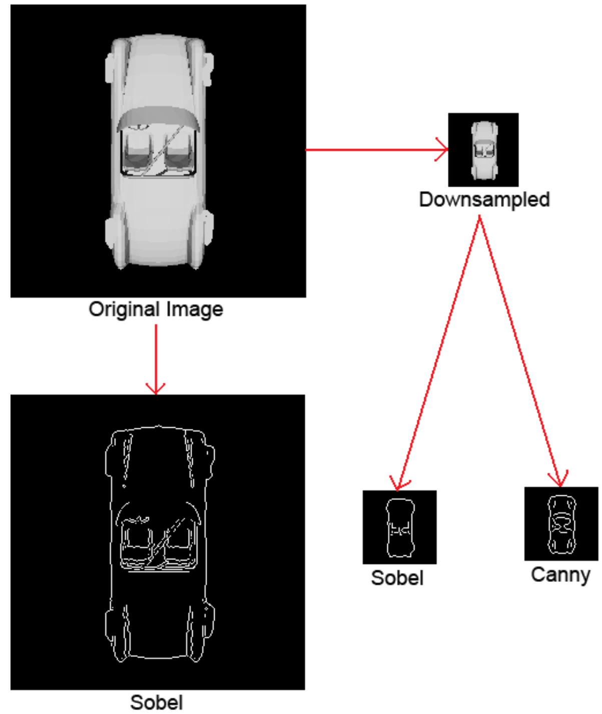

Figure 2. The effects of downsampling and alternative edge detection algorithms

We have mentioned above that a value σ is used during the construction process. This value is the maximum Hausdorff Distance value, beyond which two images are considered to be different and thus should not co-existing in the same HD Cluster. This value will differ depending on the dataset used and the granularity required of the searching algorithm. For example, to what extent must two models have the same features and details to be considered similar? We calculated our σ value by modifying a subset of images from our dataset in a way that we deemed the resulting images to be similar to, but not a duplicate of, their originals. We then calculated the Hausdorff Distance between each of the modified images and their paired original images and took the highest distance as our σ value. The resulting value that we utilized caused a HD Clusters to contain models that are roughly the same shape and orientation but that allowed for varying detail and minor appendages. If we wanted a stricter similarity measure, we would reduce σ value towards zero.

C. Searching the HD Clusters

To utilize this structure, we first find the Band(s) that the search image belongs to, using the process described in Section III. We then calculate the Hausdorff Distance between the search image and the representatives of each of the HD Clusters. All members of the HD Cluster with the smallest distance from the search image should be examined for direct similarity with the search image.

If we want to relax restraints on HD Cluster selection, we can introduce a variation variable that dictates how far a representative's Hausdorff Distance must be from the best ranked representative distance value in order for its cluster to be considered. For example, if HD Cluster A has a Hausdorff Distance of 6 between its representative and a search image, B has a value of 7, C has a value of 11, and our variation value is 2, then all of the images in HD Clusters A and B will be examined for similarity to our search image but C will not be examined. For all of the experiments below, we use a variation value of 0, this means that if two or more representatives have exactly the same Hausdorff Distance to the search image, then all of these HD Clusters are examined. Taking into account that our Hausdorff Distance values are calculated in floating point precision, this occurrence is rare but still possible considering the relatively limited range of Hausdorff values, usually between 0 and 150 when using an image resolution of 64x64 pixels and preliminary filtering through Banding.

V. EXPERIMENTAL SETUP AND RESULTS

A. Experiment Setup

We use the dataset that was created by Funkhouser et. al. [9] for the use of benchmarking content based 3D model retrieval systems. The dataset has 1833 models in it, though due to errors in gaining 2D projections of the models we utilize only 1829 of these. The dataset is freely available online. All 2D projections of the models were gathered using the OpenGL graphics API along with the Corona image capturing library for C++. All images have a resolution of 64x64 pixels.

All tests were carried out on a Intel Core2 2.40GHz CPU. For our implementation the CPU is the major hardware component that determines execution times. To implement the algorithm in this paper we have used MATLAB version 7.5.0 (R2007b).

B. Genuine Acceptance Rate

We use Genuine Acceptance Rate (GAR), commonly used in the field of biometrics, to judge the performance of our complete structure, which includes both Banding and HD Clustering. To help evaluate this metric, we randomly chose one hundred images from the complete dataset of models and manually modified them in a way that we believed the new image and the original image were similar but not duplicates of each other. To relate this to the field of biometrics, we consider one modified image as a genuine suspect of one corresponding original image. We do not test the original image itself because the overlapping nature of our Banding and HD Clustering algorithms means that we will always be able to find a duplicate model.

Our test was then to simply use each one of these modified images as a search image and test whether or not the name of the original file could be found in our clustering structure with our search algorithm. That is to say, no pattern recognition metrics were used once a HD Cluster was identified during search time, rather just a file name search is carried out. However, the search images are modified such that if we were to do these pattern recognition calculations we would expect to see the original image to appear high within a set of rankings.

Using the above approach we found that our Bands yielded a GAR of 92%. This means that, with overlapping our Bands and with an effective pixel Δ value of 20%, we are able to find 92% of the original images. The 8% of search images that did not succeed are most likely a result of modifying the test images to a greater extent than is allowed by our search algorithm. That is, the number of effective pixels is too different for our algorithm to classify the two images as being similar. This GAR value could be increased if we modified the 8% of search images that did not succeed such that they more closely resemble their original images or by increasing the effective region Δ value in order to make our Bands larger. However, we chose to keep this value and report it in this paper in order to show that there exist discrepancies between our algorithm and the expectation of human intuition.

The HD Cluster structure scores a GAR of 88%. This is on top of the GAR of 92% that we received for the Banding structure, meaning that the introduction of the HD Cluster reduces the GAR by 4%. We believe this to be an acceptable result. It primarily shows that the approach that we use to select HD Cluster Representatives is working well during both build time and search time.

C. Ranked Results

In this test we utilize the two clustering structures we have created but, unlike the GAR tests, we also continue to use a similarity metric to compare a search image with all of the images in a selected HD Cluster. For our experiments we simply used the Hausdorff Distance metric again. Once we gather a set of distance values from our search image, we rank the results from most similar to least

Fig. 3 and 4 show before and after comparisons. The results in Fig. 3 were gathered by using the Hausdorff Distance metric between the search image and every image in the entire dataset. The results in Fig. 4, however, were gathered by following our Banding and HD Clustering algorithms. For reference, both sets of results can be viewed as if this were a content driven information retrieval system for 3D models, returning the top ten results given a query of the search image.

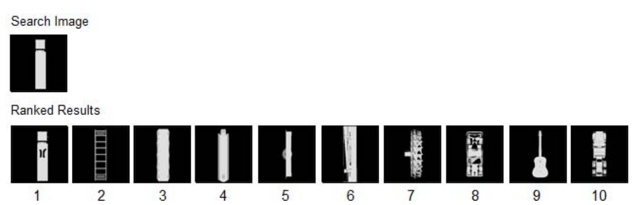

Figure 3. Ranked results by comparing search image with entire dataset

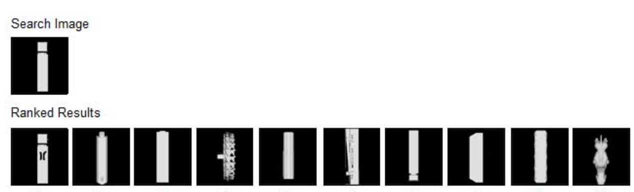

Figure 4. Ranked results while using dual layered clustering algorithm

Human intuition identifies that the ranked results in Fig. 3 show more objects that are similar to our search object. That is, that more objects within the top ten ranked results of this figure are roughly rectangular and solid as oppose to Fig. 4. This is because the objects in Fig. 3 that appear not to be greatly similar to our search image are actually erroneous results due to the similarity metric used. Our dual layer clustering algorithm prevents these errors from occurring. This means that our clustering algorithms in fact improve results.

In terms of efficiency, our finalized HD Cluster approach computes a chosen pattern recognition algorithm results against select models within two Bands for this search image. Hausdorff Distance is calculated against all of the HD Cluster Representatives of both Bands and then, for this search image, proceeds to execute the Hausdorff Distance algorithms on all images in one HD Cluster from each Band. In total, including the Representative calculations and removing repeat calculations for images that may be in both HD Clusters, we end up comparing our search image with 132 files. To give a comparison, the Bands that are examined in this experiment have roughly 320 images each. Table 1 shows the execution times for these. It also shows how long it takes for one search image to be compared against all 1829 files in the dataset, at an original resolution of 256x256 pixels and a downsampled resolution of 64x64 pixels. The Banding and HD clustering times use the downsampled images. Also, the experiment that uses HD clustering also uses the Banding structure; therefore the time stated represents the efficiency of our completed dual-layer clustering algorithm.

VI. CONCLUSION

Protecting virtual assets in massively multiplayer online games from theft or copy, especially where copyrights may be claimed, gives individuals and businesses financial security in virtual worlds. This in turn encourages online communities to grow and share in creativity. In this paper we have focused on the protection of 3D model virtual assets, attempting to identify when a newly introduced model is too similar to an existing one in the virtual world.

TABLE I. EXECUTION TIMES FOR ONE SEARCH IMAGE AGAINST THE ENTIRE DATASET

	Files Checked	Time Taken
256x256 Images	1829	4019 seconds
64x64 Images	1829	206 seconds
Using Banding	640	63 seconds
Using HD Clusters	132	21 seconds

We have created a method to cluster dataset images prior to using a similarity metric between them and a search image. This drastically reduces the number of calculations that need to be computed during search time, instead utilizing an offline build time to complete the bulk of the calculations. This is especially useful for object recognition metrics that may be computationally expensive.

Our algorithm used two layers of clustering. The first, Banding, was a basic grouping based upon the number of effective pixels in an image. The second used Hausdorff Distance to build clusters around representatives. Our results show that not only does our clustering algorithm reduce the number of calculation required at search time but also that while there is a chance for genuine suspects to be lost due to the clusters, as shown by our GAR value, there is also a chance for overall performance to be improved by removing erroneous results, such as non-similar objects from being ranked highly.

We originally designed this clustering algorithm to be used in aiding us in the copyright protection of 3D models in massively multiplayer online games. While this algorithm was tailored for use in these games and some assumptions were made in the design of the algorithm, we believe that extension of this work to other image based identification problem is possible.

REFERENCES

[1] J. Hu and F. Zambetta, “Security issues in massive online games”, in Security and Communication Networks, vol. 1, no. 1, 2008

[2] Y. Guo and S. Barnes, “Why people buy virtual items in virtual worlds with real money”, in ACM SIGMIS Database, vol. 38, no. 4, pp. 76, 2007

[3] M. Rymaszewski, W.J. Au, M. Wallace, and C. Winters, “Second Life: the official guide”, 2007

[4] Y. Chen, J.Z. Wang, R. Krovetz, “Content-based image retrieval by clustering”, in Proceedings of the 5th ACM SIGGM int. workshop on multimedia information retrieval, pp. 193-200, 2003

[5] G. Sheikholeslami, W. Chang, and A. Zhang, “SemQuery: semantic clustering and querying on heterogeneous features for visual data”, IEEE Trans. Knowledge and Data Engineering, vol. 14, no. 5, pp. 988-1002, 2002

[6] A. Vailaya, M.A.T. Figueiredo, A.K. Jain, and H.J. Zhang, “Image classification for content-based indexing”, IEEE Trans. Image Processing, vol. 10, no. 1, pp. 117-130, 2001

[7] J.Z. Wang, J. Li, and G. Wiederhold, “SIMPLIcity: semantics-sensitive integrated matching for picture libraries”, IEEE Trans. Pattern Anal. Machine Intell., vol. 23, no. 9, pp. 947-963, 2001

[8] J. Li and J.Z. Wang, “Automatic linguistic indexing of pictures by a statistical modeling approach”, IEEE Trans. Pattern Anal. Machine Intell., vol. 25, no. 10, 2003

[9] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A.Halderman, D. Dobkin, and D. Jacobs, “A search engine for 3D models”, ACM Transaction on Graphics, vol. 22, no. 1, pp. 83-105, 2003

[10] Y. Gao and M.K.H Leung, “Line segment Hausdorff distance on face matching”, Pattern Recognition, vol. 35, no. 2, pp. 361-371, 2002

[11] J. Canny, “A computational approach to edge detection”, Readings in Computer Vision: Issues, Problems, Principles, and Paradigms, pp. 184-203

[12] M Heath, S. Sarkar, T. Sanocki, and K. Bowyer, “Comparison of edge detectors. A methodology and initial study”, Computer Visionion and Image Understanding, vol. 69, no. 1, pp. 38-54, 1998