WebMARS: A Multimedia Search Engine

Michael Ortega-Binderberger,[a] Sharad Mehrotra,[b]
Kaushik Chakrabarti[a] and Kriengkrai Porkaew[a]

[a]University of Illinois at Urbana Champaign

[b]University of California at Irvine

ABSTRACT

The Web provides a large repository of multimedia data, text, images, etc. Most current search engines focus on textual retrieval. In this paper, we focus on using an integrated textual and visual search engine for Web documents. We support query refinement which proves useful and enables cross–media browsing in addition to regular search.

Keywords: Multimedia Analysis and Retrieval System, MARS, Multimedia Retrieval, Image Retrieval, Multimedia Web Search Engine

1. INTRODUCTION

With advances in digital multimedia and Web technologies, there is an increasing trend in documents over the Web to incorporate several media types such as text, images, audio and video. While some prototype systems[1–3] as well as commercial Web search engines[4] have started to incorporate image search extensions, few systems have properly explored the use of all available information in different media types for retrieval into a single integrated framework. To address this limitation, in the MARS project, we have developed a web search engine, WebMARS, that exploits both textual and visual information for HTML document retrieval. The integrated retrieval framework allows WebMARS to exploit both the textual (e.g., words) and visual content (e.g., images) of Web pages in answering user's queries. WebMARS not only searches over multiple media types but also utilizes cross-media information to improve the search performance. Answers returned to the user by the system on evaluation of a query may include good and bad results. To enhance retrieval performance, query refinement (QR)[5] is used. QR changes the original query with user input to improve its results. QR is exceedingly important since the user may initially submit a coarse query using one media type and then use successive refinement iterations to search related material in a different or composite media. This effectively bridges the gap between different media types, allows queries to spread from one media type to another, and fuses querying and browsing into a single integrated framework. Our preliminary experiments show that for both small (2000 documents) and medium (20,000 documents) sample HTML collections, retrieval effectiveness (precision and recall)[6] improve significantly when both image and text properties are used for content search instead of just using a single media type.

This document is developed as follows. Section 2 describes the multimedia object and query models used, including the query refinement and document matching models. Section 3 presents some experimental results, sect. 4 discusses some related work and finally, sect. 5 offers some conclusions.

2. MODEL

A document is represented as a hierarchical collection of objects which themselves are represented as collections of features (also referred to as terms). Different features are extracted from different media types and at different levels; e.g. images have regions and features are extracted for individual regions and the image as a whole. Examples of textual features include words in a document, citations, etc. Conversely, visual features refer to either low level image properties such as color in an image, textures or patterns, layout or organization of colors and textures, etc. Higher level visual features include descriptions of salient regions in an image using visual properties such as shape, color and textures. Users specify their information need (IN) in the form of a query. Given a representation of the users IN and a document collection, WebMARS estimates the likelihood that a given document matches the users IN. The representation of documents and queries, and the metrics used to compute the similarity among them constitute the *retrieval model* of the system.

Send correspondence to Sharad Mehrotra, 444 Computer Science, University of California, Irvine, Irvine, CA, 92697-3425.
E-mail: M.O.B.: miki@acm.org, S.M.: sharad@ics.uci.edu, K.C.: kaushik@ics.uci.edu, K.P.: nid@ics.uci.edu

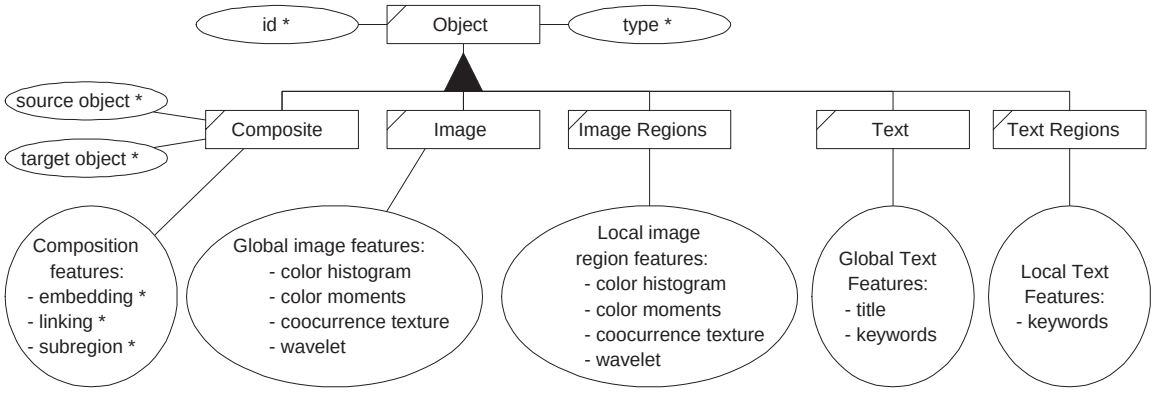


Figure 1. Multimedia Object Model

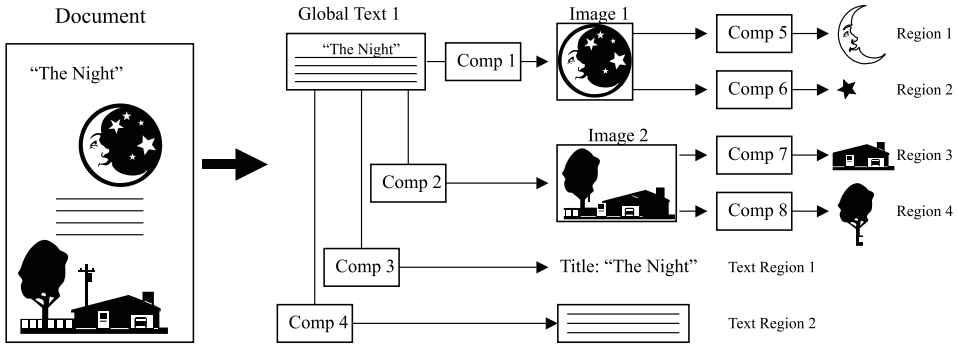


Figure 2. Multimedia Object Model Example

2.1. Multimedia Object Model

The multimedia object model used is illustrated in fig. 1 which shows an object model with five types: composite, textual, textual regions, images and image regions. *Object* is the major entity that defines properties common to all objects: identification number and type. *Composite* objects exist to capture the many to many relationship between multimedia objects. For example, an HTML page with an embedded image may result in many objects: the textual page, several textual regions, an image and image regions, and a number of composite objects that capture the relationships between objects in the page. *Image* objects are represented through image features described below and are also segmented into several regions for which features are also extracted allowing for region matching. *Textual* objects are sub-entities of Object and further described below. A document is a HTML text page and all images such that the text page is related to that image through a *Composite* relationship. Figure 2 shows how an example document is represented: 17 objects are used, one for the text of the document, 2 for main text regions, 2 for the images, 4 to represent image regions and 8 to represent the relationships between the textual and visual objects.

2.1.1. Text objects

For textual data, we support two granularities, at the region level and the complete document. Regions can be e.g. titles, citations or paragraphs. We use the keywords feature in a *vector space* model[6] to represent the textual content. Recent models[7] provide slightly better performance at a much higher implementation cost. Since our purpose is to explore techniques to retrieve documents consisting of multiple media types and the relative improvement in retrieval effectiveness by considering multiple media types in the same retrieval framework, a simplistic text retrieval model suffices. In the vector space model, a textual object is considered as a collection of words. The *term frequency* (tf) is the number of times a word appears in the object. *Document frequency* (df) is the number of objects in which the same word appears at least once. To avoid noise in the representation of text objects, commonly used words such as prepositions are discarded given their lack of content. Tf correlates positively with the importance of words

in the text object while *df* correlates negatively. Commonly the inverse of the logarithm of *df* is used $log(\frac{N_{txt\ obj}}{df})$, where $N_{txt\ obj}$ is the total number of textual objects, and named *inverse document frequency (idf)*. All terms i in an object are then assigned weights: $w_i = tf_i \times idf_i$,[8,6,9] words not present have a weight of 0. Let N_{word} be the number of distinct words in the textual object collection, each object is then viewed as a point (vector) in an N_{word} dimensional space. Queries and objects are represented the same way. The similarity between objects is defined over pairs of vectors, a similarity of 1 indicates a good match. The similarity function used in this model is the cosine of the angle between the vectors formed by the two points in the N_{word} dimensional space[6]: $sim(D,Q) = \frac{D \cdot Q}{||D||\ ||Q||}$. Textual objects can be complete documents or regions of documents, e.g. titles are stored as separate regions.

2.1.2. Image objects

The retrieval performance of an image database is inherently limited by the nature and the quality of the features used to represent the image content. For this prototype we used the following subset of features and representations: *Color histogram* (CH),[10] and *color moments* (CM),[11] were proposed in the past few years; we chose both features in the Hue, Saturation, Value (HSV) color space since the HSV color space has uniform, de-correlated coordinates, better matching the human perception of color. The V coordinate is discarded given its sensitivity to the lighting condition. The HS coordinates are used to form a two-dimensional histogram by dividing the H and S dimensions into $N \times M$ bins. Each bin contains the percentage of image pixels that have corresponding H and S values for that bin; note that the sum of all bins equals one.[10] The similarity between two color histograms is computed by histogram intersection which captures the amount of overlap between two histograms: $sim_{color} = \sum_{i=1}^{N} \sum_{j=1}^{M} min(H_1(i,j), H_2(i,j))$ where H_1 and H_2 are the two histograms; and N and M are the number of bins along the H and S coordinates. Since the histograms are normalized to add to one, this measure ranges from 0 (not similar) to 1 (identical). For CM[11] the first (average), second (standard deviation) and third moments of the H, S and V coordinates of each pixel are computed and then normalized. Euclidean distance is used to determine the distance and appropriately normalized to yield a similarity value.[12] *Texture Features* used include the *coocurrence matrix method* (CO)[13] and *wavelet based* (WV).[14] In CO, four matrices are formed on the pixel intensities to measure variations between adjacent pixel values in the vertical, horizontal, and the two diagonal directions. For each matrix four metrics are computed: second angular moment, contrast, inverse difference moment and entropy,[13] resulting in a 16 dimensional vector. The similarity between two texture feature vectors is defined as the Euclidean distance in this 16 dimensional feature space. WV is based on a two dimensional three level wavelet decomposition where the texture vectors are the average and standard deviation of the wavelet coefficients in each of the 10 generated fields. Image regions can be obtained through several methods. Human segmentation of visual objects is one such method, or e.g. automatic visual object segmentation algorithms with their known limitations. An alternative approach presented in ref. 15 obtains regions based on clustering. After extraction of the features for the collection, a domain specific post-processing step is required that normalizes the features to make their similarities meaningful. This process has three objectives: a) put equal emphasis on each feature element within a feature vector, i.e. account for different physical properties that affect their magnitudes, thereby biasing distance measures, b) simplify computation of distance functions, c) map natural distance values from each atomic feature into the range [0,1] to facilitate interpretation as similarity values.[12] The feature vectors are then indexed using Hybrid trees, which are a high dimensionality indexing structure.[16]

2.2. Weighted Summation Query Model

We chose to combine similarity values using a weighted summation model given its monotonicity properties and intuitive operation which naturally lead to query refinement discussed later.

2.2.1. Query tree

A query is a tree where nodes can be of two types: 1) leaf nodes n_L represent feature vectors, and 2) internal nodes n_I which have their similarity computed as the weighted summation of the similarity values for all children of n_I. Each child carries an associated weight and the sum of the weights for all children of a node is 1, initially the weights for all children of a node are equal, this will change in sect. 2.3. For example, a user can select an image region as a simple query where the query tree is formed by a single internal node (the image region) with its features as leaf nodes. I.e. the similarity of arbitrary image regions or images I to the query Q will be the weighted summation of the similarities for their component features: $sim(I,Q) = \sum_{i \in features} w_i \times sim_{feature\ i}(I,Q)$. The similarity of an internal node n_I in the query tree is computed recursively according to $sim(object, n_I) = \sum_{i \in children(n_I)} w_i \times sim(object, i)$ until the root of the tree is reached. The similarity value for the root of the query tree is the final similarity of the object with the query.

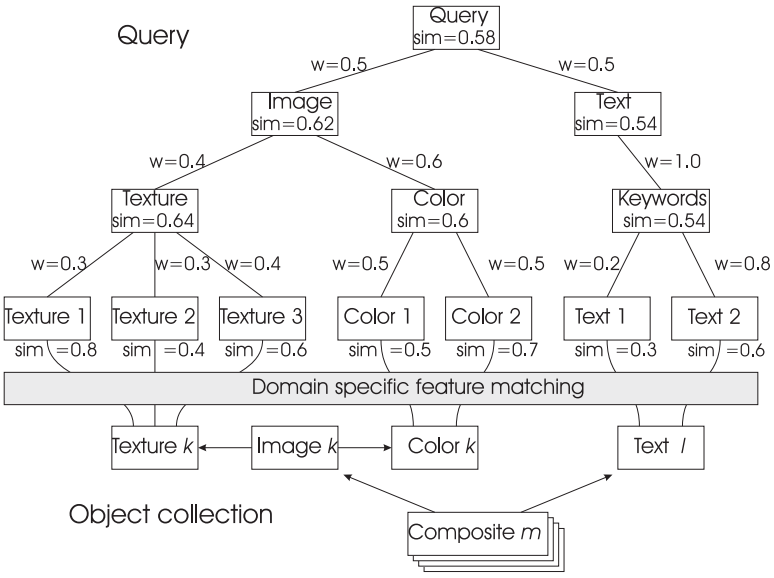


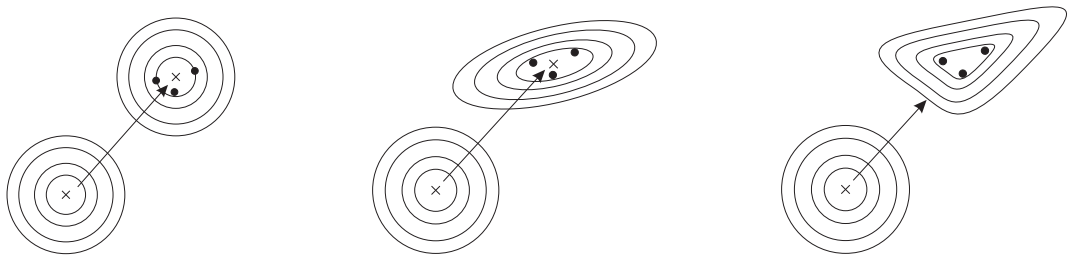
Figure 3. Example Query Tree

2.2.2. Construction of the query tree

We now turn to the construction of the query tree. The tree is organized in four levels: a) *Query level* is the root of the tree and represents the final similarity between an object and a query, b) *Type level* nodes group media types together, i.e. media of the same type is handled below this level, c) *Feature level* nodes group features of the same type together i.e. images or image regions are at this level, and d) *Feature instance level* are leaf nodes (n_L), all features of the same type, but belonging to different objects in the query, are organized under the same parent. Figure 3 shows an example query tree formed with text and two image elements and the computation of similarity values along the tree. Matching of a composite object is described next and shown in the figure.

2.2.3. Document matching

Conceptually, the user wants to match complete documents based on all information and all granularities available; sect. 2.3 describes how the user may later assign more importance to certain regions, media or parts of the query tree. The document collection contains textual and image objects at global and local granularities which are linked by composite objects. To enable document retrieval, conceptually, all objects in the collection are compared to the query to determine the degree to which they satisfy the users IN, specifically, all "chains" of objects that fit the query tree have their similarities computed. I.e. if the query tree contains a text object, an image and an image region, then (from fig. 2): "Global Text 1", "Comp 1", "Image 1", "Comp 5" and "Image Region 1" are said to fit the query tree and this set of objects are used to compute a final similarity. The chain "Global Text 1", "Comp 1", "Image 1", "Comp 7" and "Image Region 3" is not valid, and thus no evaluation is done using this combination. Preference is given to those combinations that have the most complete fit in the query tree, i.e. those that do not leave any low level nodes empty. Once no more full length matches can be performed, partial matches are taken, setting unused features to a similarity value of 0. The motivation for this is to include documents that are purely textual or purely visual in the retrieval process. Since the user expects whole documents in response to her queries, this imposes an additional restriction on the retrieval in which all composite objects that share the same document (main text unit derived from the retrieved HTML page) are collapsed together into a text object with multiple associated text region, image and image region objects. I.e. only one document including all associated components will be returned as a unit for fig. 2, not several individual components. For each document, there might be several "chains" that can match the query tree. For fig. 2 the complete set of chains is: ("Global Text 1", "Comp 1", "Image 1", "Comp 5" and "Image Region 1"), ("Global Text 1", "Comp 1", "Image 1", "Comp 6" and "Image Region 2"), ("Global Text 1", "Comp 2", "Image 2", "Comp 7" and "Image Region 3"), ("Global Text 1", "Comp 2", "Image 2", "Comp 8" and "Image Region 4"), ("Global Text 1", "Comp 3" and "Text Region 1"), and ("Global Text 1", "Comp 4" and "Text Region 2"). To determine the ranking of the complete document, the similarities for all the component chains are included.



a) Centroid only (QPM) b) Changed similarity function (QSF) c) Query expansion (QEX)

Figure 4. Intra Feature Feedback Models

The document similarity is the maximum similarity among all the chains, and is the one used for ranking purposes. This aggregation process is performed only to present results to the user, and does not modify the structure and semantics of the query tree.

2.3. Query Refinement

Query refinement (QR) has two objectives, a) improve the quality of the results, and b) to allow searching and browsing in an integrated framework bridging multiple media types. The query model has two main phases. First, individual features are compared to yield feature to feature similarities which are combined at higher levels to fix the final similarity value. The user may receive good and bad results. To enhance retrieval performance, QR is used. QR consists of changing the original query to improve its results. One way to perform QR is to present results to the user and ask her to state which are relevant and which are not; this is known as relevance feedback (RF), the process of automatically adjusting an existing query using information fed-back by users about the relevance of retrieved documents. Following the two stages of queries described above, refinement can occur at two levels: inside a feature (intra feature feedback), and at internal nodes in the query tree (query re-weighting).

2.3.1. Intra feature relevance feedback

Retrieval performance can improve by using RF techniques at the feature level.[17] Instead of specifying exact values for the query, the user submits a coarse initial query to start the retrieval process. The feature values of the query will be *automatically* refined according to the users RF, such that the refined query better approximates the users IN. Several RF iterations are done until the user is satisfied or the refinement reaches a saturation point. The techniques discussed, query point movement (QPM), its enhancement query similarity function (QSF) and query expansion (QEX), work for features represented as vectors in an N dimensional vector space.

In the *QPM* model, a query in each feature space is represented by a single point. Using RF information from the user, the point in the vector space is moved closer to the ideal position. If the sets of relevant and non-relevant feature vectors are known, an optimal query vector can be determined. In practice these are not known in advance, however the RF obtained from the user provides reasonable approximations to these sets. RF information is used to construct a new query vector closer to the optimum query. Experiments show that the retrieval performance improves considerably with this technique.[8,9] The technique described moves only the center-point of the query space to a new centroid. Figure 4a) shows a two dimensional example, where the query point ($\times$) is moved to a location central to the three relevant vectors. The query vector changed, but not the similarity function. In QPM, by applying a scaling and rotation transformation to the query space in addition to the point movement we obtain a modified query similarity function (QSF), the space around the query point is warped; this flexible strategy is shown in figure 4b).[5,18]

The *QEX* model uses several points for the query, instead of a single point. New relevant points are added to the query and old points not found to be useful are removed. Figure 4c) shows an example of this method. Note that the black dots represent relevant vectors that are now part of the query. Many policies to add or remove points from the query exist.[17] A simplistic approach is to add all points judged relevant by the user and remove those that are not. Other policies including a cluster based approach in which the points deemed relevant (and similarly not relevant) are clustered and cluster representatives (centroids) of the relevant set are added (points within a non-relevant cluster are discarded) can be found in reference 17. Note that query expansion may add leaf nodes to the query tree.

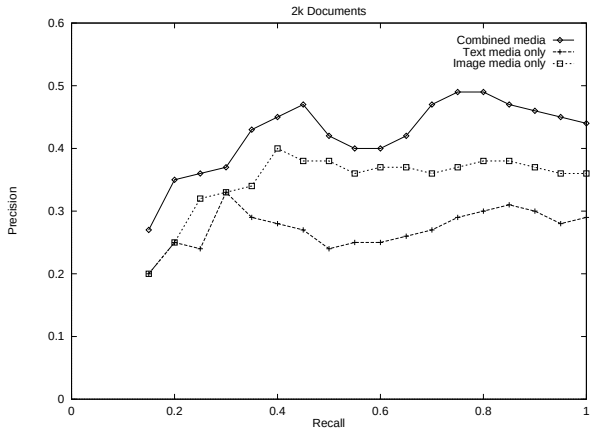


Figure 5. Precision Recall, 2k Documents

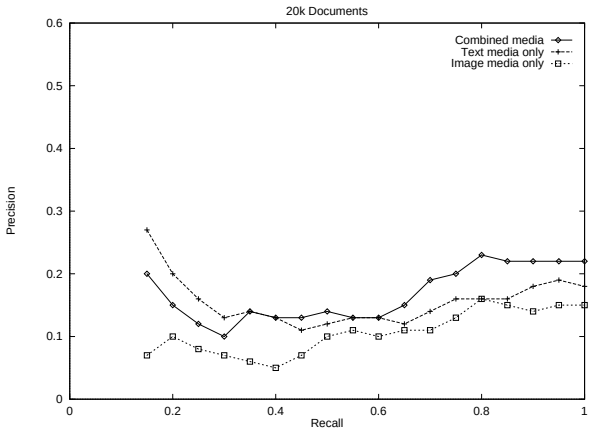


Figure 6. Precision Recall, 20k Documents

2.3.2. Query tree reweighting

In Query Tree Reweighting (QTR) the weights on the parent–child link in the query tree are initially equal for all children of the same parent and add up to 1. Without modifying the structure of the query tree the weights are changed to better reflect the users IN. To update the weights, first a list of objects for each child is built such that all objects returned are restricted to have a higher similarity value than some threshold. Then all the relevant objects indicated by the user are searched in these lists. The new weight for a link is selected by one of the following strategies: a) *count* of the number of relevant objects in the list, b) *minimum similarity value* among all objects in the list, and c) *sum* the similarity values for all relevant objects in the list. Since this process does not yield weights that sum to 1, the weights are normalized to add up to 1 for all children of a node. This process is propagated up in the query tree until the root is reached. Once this process is complete, the query is re–evaluated and shown to the user for another iteration of RF. The rationale why the above query reweighting strategies converge to the optimal representation can be found in reference 5.

3. EXPERIMENTS

We have implemented the core of the described model in our MARS server. The dataset consists of hypertext documents from the WWW including HTML and images collected by our web crawler which populates the multimedia object collection. A subset of the collection was used to perform some experiments with query refinement and to measure how multiple media types improve retrieval performance.

3.1. Multiple Media Retrieval

In this experiment we focus on the contributions of performing a query using multiple media types. We do this on a subset of the collection by measuring the *precision* and *recall*[6] of the queries. Precision is the percentage of relevant documents in the retrieved set, and recall is the percentage of relevant documents in the retrieved set vs. all relevant documents in the collection. To determine the relevant set for a query, we took advantage of the source of the HTML documents we used; the documents are grouped already according to content in sets of about 100 at a time, thus for each query document, the set of relevant documents was considered to be its associated category. Three sets of experiments follow the alternatives: a) using only text objects for retrieval, b) only images, and c) using both. Figure 5 and 6 show the results for subsets of the collection with 2,000 and 20,000 documents. Sometimes image only retrieval outperforms text only, and sometimes this is reversed. However, using both media gives the best performance confirming that using multiple media types for retrieval results in better overall performance.

3.2. Query Refinement

The focus of our experiments to evaluate QR is to test the ability of the mechanism to converge. An initial query formed with objects chosen randomly from the collection is executed and the ranked result saved as the ground truth (relevant list, RL). Some objects are chosen from RL for a new query, which is executed and produces a retrieved list (RT). The difference between RL and RT indicates the improvement of the retrieval process. In the i_{th} iteration,

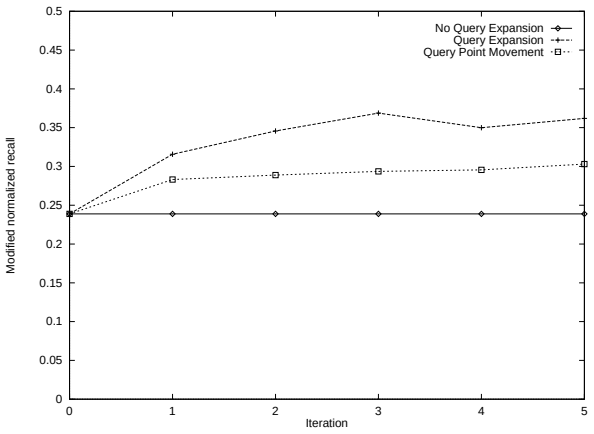


Figure 7. QEX vs. QPM

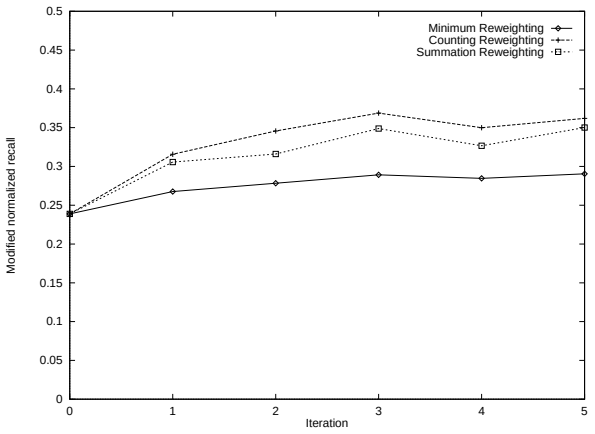


Figure 8. QR approaches

good answers in RT_i are marked based on RL and sent to the system which refines and reexecutes the query to create RT_{i+1}. The effectiveness of QR is measured by the increase in the similarity between the lists at each iteration. To compare two ranked lists (RL and RT) we use a modified normalized recall metric (NR).[6] Our modified NR computes the rank difference of the lists, and normalizes based on the number of retrieved documents, not the whole collection and puts more emphasis on highly ranked responses through weights that linearly decrease from the top ranked to $\mid RL \mid$. Experiments include all combinations of proposed strategies. For each combination, query objects are taken from five distances in the RT list to see how fast the query refinement converges. Figure 7, shows that QEX outperforms QPM. Figure 8, shows that the counting approach outperforms the other two approaches for reweighting. Experiments show faster convergence to the goal when we start a query close to the goal which confirms the intuition. Among all approaches explored, counting combined with query expansion outperforms all other models.

4. RELATED WORK

Similar to MARS, most existing content-based image retrieval systems also extract low-level image features like color, texture, shape, and structure.[1,2,19,20] QBIC,[2] is the first commercial content-based Image Retrieval system, its system framework and techniques had profound effects on later Image Retrieval systems. Virage[21] is a content-based image search engine developed at Virage Inc. and supports visual queries based on color, composition (color layout), texture, and structure (object boundary information). It also supports weighted combinations of the above four atomic queries. Photobook[1,22] is a set of interactive tools for browsing and searching images, consisting of sub-books, with shape, texture, and face features extracted respectively; users can query based on features in each sub-book. Picard et al. proposed to include the user in the image annotation and retrieval loop[22] motivated by the observation that no single feature best models images from all domains. VisualSEEk[23] and WebSEEk are visual feature search engines, from Columbia University concentrating on spatial relationship queries of image regions where users sketch their queries. ImageRover[24] is a web based image search engine from Boston University, it produces a vector representation of the image and text features creating a unique vector for each image content. A project focused on multiple media types in a single retrieval framework is AMORE.[25]

5. CONCLUSIONS

WebMARS is a multimedia retrieval system that concentrates on the retrieval of whole documents based on all their component media types. This is a departure from the more traditional text or image centric approaches to retrieval. Retrieving whole documents is also advantageous since now the user can pick whole documents or parts of them to perform QR, which also helps to bridge the gap between different media types; thus the browsing and querying functions are fused providing a flexible interface to the user. The use of multiple media types in formulating the query also helps the retrieval performance of the system as shown in our results.

ACKNOWLEDGMENTS

This work was supported in part by the Army Research Laboratory under Cooperative Agreement No. DAAL01-96-2-0003; in part by NSF/DARPA/NASA Digital Library Initiative Program under Cooperative Agreement No. 94-11318; in part by NSF CISE Research Infrastructure Grant CDA-9624396. Michael Ortega-Binderberger is supported in part by CONACYT grant 89061, Mavis Fellowship and an IBM Fellowship.

REFERENCES

1. A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook: Tools for Content-Based Manipulation of Image Databases," in *Proc. Storage and Retrieval for Image and Video Databases II*, vol. 2(185), pp. 34–47, SPIE, (Bellingham, Wash), 1994.
2. M. Flickner *et al.*, "Query by Image and Video Content: The QBIC System," *IEEE Computer* , September 1995.
3. J. R. Smith and S.-F. Chang, "Searching for images and video on the world-wide web," Tech. Rep. 459-96-25, Columbia Univ., 1996.
4. "The altavista search engine." http://www.altavista.com/.
5. K. Porkaew, S. Mehrotra, and M. Ortega, "Query reformulation for content based multimedia retrieval in MARS," in *Proc. IEEE Conf. on Multimedia Computing and Systems*, (Florence, Italy), June 1999.
6. G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*, McGraw Hill, 1983.
7. J. P. Callan, W. B. Croft, and S. M. Harding, "The INQUERY retrieval system," in *Proceedings of the Third International Conference on Database and Expert Systems Applications*, (Valencia, Spain), 1992.
8. C. Buckley and G. Salton, "Optimization of relevance feedback weights," in *Proc. of SIGIR'95*, 1995.
9. W. M. Shaw, "Term-relevance computations and perfect retrieval performance," *Info. Proc. and Mgmt.* .
10. M. Swain and D. Ballard, "Color Indexing," *International Journal of Computer Vision* **7**(1), 1991.
11. M. Stricker and M. Orengo, "Similarity of Color Images," in *Proc. SPIE Conf. on Vis. Commun. and Image Proc.*, 1995.
12. M. Ortega, Y. Rui, K. Chakrabarti, K. Porkaew, S. Mehrotra, and T. S. Huang, "Supporting ranked boolean similarity queries in mars," *IEEE Tran. on Knowledge and Data Engineering* **10**, November–December 1998.
13. R. M. Haralick, K. Shanmugam, and I. Dinstein, "Texture Features for Image Classification," *IEEE Trans. on Sys, Man, and Cyb* **SMC-3**(6), 1973.
14. A. Laine and J. Fan, "Texture Classification by Wavelet packet Signatures," *IEEE Trans. Patt. Recog. and Mach. Intell.* **15**(11), pp. 1186–1191, 1993.
15. A. Natsev, R. Rastogi, and K. Shim, "Walrus: A Similarity Retrieval Algorithm for Image Databases," in *ACM SIGMOD*, pp. 395–406, (Philadelphia, PA), June 1999.
16. K. Chakrabarti and S. Mehrotra, "The hybrid tree: An index structure for high dimensional feature spaces," March 1999.
17. K. Porkaew, K. Chakrabarti, and S. Mehrotra, "Query refinement for multimedia similarity retrieval in MARS," in *ACM International Multimedia Conference*, (Orlando, Florida), Nov 1999.
18. Y. Ishikawa, R. Subramanya, and C. Faloutsos, "MindReader: Querying Databases through Multiple Examples," in *Proceedings of VLDB*, 1998.
19. B. Manjunath and W. Ma, "Texture Features for Browsing and Retrieval of Image Data," Tech. Rep. TR-95-06, CIPR, July 1995.
20. J. R. Smith and S.-F. Chang, "Tools and Techniques for Color Image Retrieval," in *IS & T/SPIE proceedings*, vol. 2670, 1994. Storage & Retrieval for Image and Video Databases IV.
21. J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C. fe Shu, "The Virage Image Search Engine: An open framework for image management," in *SPIE Storage and Retrieval for Still Image and Video Databases IV*,
22. T. P. Minka and R. W. Picard, "Interactive Learning Using a "society of models"," Tech. Rep. 349, MIT Media Lab, 1996.
23. J. R. Smith and S.-F. Chang, "Visualseek: A fully automated content-based image query system," in *Proc ACM Multimedia 96*, 1996.
24. S. Scarloff, L. Taycher, and M. La Cascia, "Imagerover: A content-based image browser for the world wide web," *Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries* , June 1997.
25. "Advanced Multimedia Oriented Retreival Engine (AMORE)." http://www.ccrl.neclab.com/amore/index.html.