https://doi.org/10.48009/2_iis_2009_444-450

WEB IMAGE INFORMATION MINING SYSTEM USING UUP FOR E-BUSINESS INTELLIGENCE

Seong-Yong Hong, Korea Advanced Institute of Science and Technology (KAIST), gosyhong@kaist.ac.kr

ABSTRACT

Recently, the web sites such as e-business sites and shopping mall sites deal with a lot of image information. To find a specific image from these image sources, we usually use web search engines or image database engines which rely on keyword only retrievals or color based retrievals with limited search capabilities. This paper presents an intelligent web e-catalog image retrieval system using metadata and user log. We propose the system architecture, the texture and color based image classification and indexing techniques, and representation schemes of user usage patterns. The query can be given by providing keywords, by selecting one or more sample texture patterns, by assigning color values within positional color blocks, or by combining some or all of these factors. The system keeps track of user's preferences by generating user query logs and automatically adds more search information to subsequent user queries. To demonstrate the usefulness of the proposed system, some experimental results showing recall and precision are also explained.

Keywords: Data Mining**,** Image Retrieval System, E-Business Intelligence, Image Information Mining.

INTRODUCTION AND BACKGROUND

Data and Metadata Mining is becoming a mainstream technology used in e-business intelligence applications supporting industries such as financial services, retail, healthcare, telecommunications, and higher education, and functions of business such as marketing, manufacturing, customer experiences, customer service, and sales [1]. Many of the business problems that data mining can solve cut across industries such as customer retention and acquisition, cross-sell, and response modeling. In recent years, however, data mining products have simplified data mining considerably by automating the process-making the fruits of the technology more widely accessible. New algorithms and heuristics have been evolved to provide good results with little or no experimentation or data preparation. In addition, the availability of data mining has increased with in-database data mining capabilities [2, 3, 4].
More recently, the concept of e-business intelligence is being expanded to include data mining techniques to extract knowledge and generated predictions for business problems, thereby enabling companies to make better use of costly cooperate asset, their organizational data.

Data Mining Techniques

Data mining is the process of finding patterns and relationships in data. At its core, data mining consists of developing a model, which is typically a compact representation of patterns found using historical data, and applying that model to new data [5]. Data Mining offers different techniques that can be used depending on the type of problem to be solved. For example, a marketing manager who wants to find out which customers are likely to buy a new product can use the classification function. Similarly a supermarket manager who wants to determine which products to put next to milk and eggs, or what coupons to issue to a given customer to promote the purchase of related items can use the association function [6, 7].
Java technology, specifically as leveraged within the scalable J2EE architecture, facilitates integration with existing application such as B2B and B2C web sites, customer care centers, campaign management, as well as new applications supporting national security, fraud detection, bioinformatics and life sciences. JDM (Java Data Mining) allows users to draw on the strengths of multiple data mining vendors for solving business problems, by applying the most appropriate algorithm implementations to a given problem without having to invest resources in learning each vendor's proprietary API [8].

Image Information Retrieval System

With the rapid development of the Internet technology, the number of internet users and the amount of multimedia information on the Internet is ever increasing. Recently, the web sites such as e-business sites and shopping mall sites deal with lots of image information. To find a specific image from these image sources, we usually use image database engines or web search engines. But, the feature based retrieval capabilities of these systems are quite limited, especially for the web images.
Many image retrieval systems have been developed, such as QBIC [9], Safe, VisualSEEK, Photobook, WBIIS, SIMPLIcity, Blobworld [10,11,12,13]. Some

systems rely on keyword only retrievals and others support image content based retrievals. In the latter approach, they support image retrievals based on the image feature information, such as average colors, color histograms [17,18,19,20], texture patterns [15], and shape objects [13]. But, most of them are developed for image database applications [14,21,22].

This paper is an effort to develop an intelligent web image retrieval system. The approach of this paper is a little bit different. We are currently developing the agent-based image search engine which supports the content-based retrieval on web images. We are trying to support various access paths on web images with customized feedback according to internet user's preferences. To support this, we have applied both data mining [16] and web mining techniques [2].

This system supports hybrid image retrievals based on query keywords, colors, and textures. For a given web image, our system classifies it whether it is textured or non-textured and, for the textured image, assigns the appropriate texture pattern(s) to the image. The proposed system also generates positional color information. The query can be given by providing keywords, by selecting one or more sample texture patterns, by assigning colors within positional color blocks, or by combining some or all of these factors.

The proposed system is intelligent since it remembers user's preferences and adds more feature information to the given query. For example, if the user provides search keyword “shirts”, the system automatically adds texture patterns (e.g., check pattern) and color information according to user's previous preferences. As a result, the system can keeps track of user's query logs and applies data mining techniques to determine user usage patterns on colors and textures.

IIM SYSTEM ARCHITECTURE

The architecture of the agent-based image search engine, IIMS (Image Information Mining System), which supports the content-based retrieval on web image is shown in Figure 1.

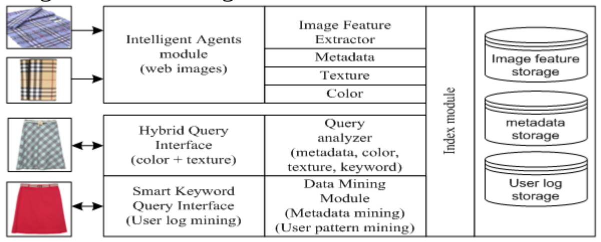
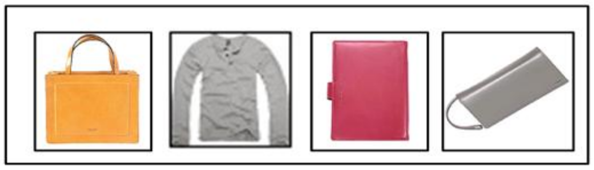

Intelligent Agents module (web images)
Image Feature Extractor
Metadata
Texture
Color
Hybrid Query Interface (color + texture)
Query analyzer (metadata, color, texture, keyword)
Smart Keyword Query Interface (User log mining)
Data Mining Module (Metadata mining) (User pattern mining)
Index module
Image feature storage
metadata storage
User log storage

Figure 1. System architecture of the image metadata mining system IIMS

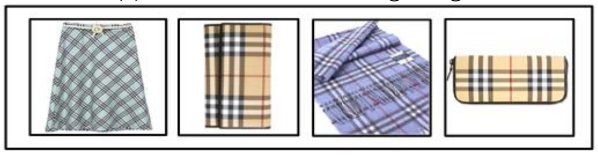
Web images are collected by the Intelligent Agents. The Feature Extractor preprocesses web images to extract metadata, color and texture information. Here, image metadata represents image size, image type, etc. The Index Module generates texture bitmap indexes and color region indexes for web images. The IIMS Interface provides various search forms for content based retrievals. The image search logs are collected and used by the Knowledge Generation Module to build user usage pattern (UUP)s. The Query Analyzer utilizes user usage patterns as well as image feature information to return more accurate search results, thus increasing user's satisfaction.

TEXTURE AND COLOR BASED INDEXING

Users of e-business shopping mall sites are usually interested in the texture, color, shape, and size of products. These web catalog images can be classified as textured (Figure 2a) and non-textured (Figure 2b).



(a) Non-textured E-catalog Image



(b) Textured E-catalog Image

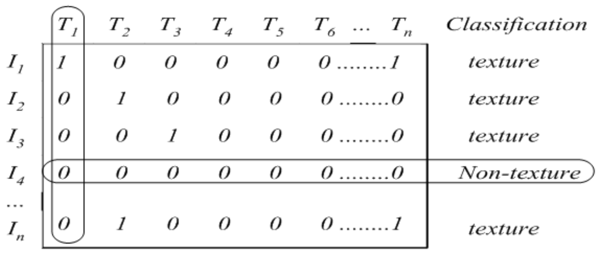
Figure 2. Image types

We represent set of stored images in the equation (1) and set of texture patterns in the equation (2).

$$\sum_{x=1}^{n} I_x = \{I_1, I_2, ..., I_n\} \text{ ....(1)}$$

$$\sum_{x=1}^{n} T_x = \{T_1, T_2, ..., T_n\}. \text{ ....(2)}$$

An image can have zero or many texture patterns, so that we represent the set of textures of the I_x image as texture$(I_x) \subseteq \{T_1, T_2, ..., T_n\}$ or texture$(I_x) \subseteq \{\}$, where n is the number of textures included in the I_x image. The empty set means that the given image is non-textured. We represent the texture information of an image as n-bit bitmap vector. The bit 1 of the bitmap vector for an image represents the presence of the texture within it. Figure 3 shows an example of bit vector index for n images. In Figure 3, the image I_4, having all zero bits, represents a non-textured image. The first and last rows represent texture $(I_1) = \{T_1, T_n\}$ and texture $(I_n) = \{T_2, T_n\}$. This means that I_1 and I_n have one common texture T_n. Therefore, these two images will be seen to users as similar images. Similarly, we can observe that I_2 and I_n have one common texture T_2. These bit vector indexes are used to support fast texture based retrievals.



	T_1	T_2	T_3	T_4	T_5	T_6	...	T_n	Classification
I_1	1	0	0	0	0	0	....	1	texture
I_2	0	1	0	0	0	0	....	0	texture
I_3	0	0	1	0	0	0	....	0	texture
I_4	0	0	0	0	0	0	....	0	Non-texture
...									
I_n	0	1	0	0	0	0	....	1	texture

Figure 3. Bit vector index for textured images

Indexing on Image Colors

Web images have various colors. The average RGB/HSI values or their color histograms are usually used for content based retrievals on colors. The histogram-based retrieval is quite slow because of the computing overhead by dimensionality curses. To lessen the computing time, it is essential to lower the histogram dimensions. Figure 4 shows an example lowering the image dimension from 256×256 to 8×8.

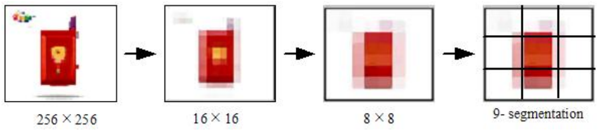


Figure 4. Transforming the image into the lower dimensions

We divide the resulting low resolution image into m×m positional blocks (segments) to provide queries on color regions. For each C_j (1≤j≤m×m), the average RGB value and average HSI value of the corresponding color regions are stored. Figure 5 shows the set of colors of the I_x image, color (I_x) = {C_1, C_2, ..., C_9}, where $m = 3$. The color search ranges are expressed as (x-range, y-range). For example, to find images having color search conditions on region C_4, C_5, and C_6, the color search range is expressed as (2, 1:3).

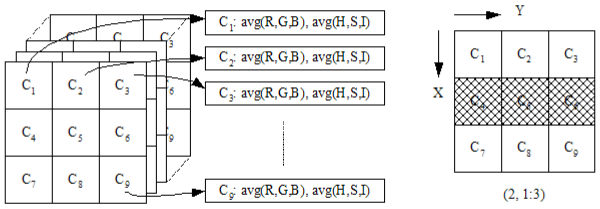

C1: avg(R,G,B), avg(H,S,I)
C2: avg(R,G,B), avg(H,S,I)
C3: avg(R,G,B), avg(H,S,I)
C9: avg(R,G,B), avg(H,S,I)
C1 C2 C3
C4 C5 C6
C7 C8 C9
Y
X
(2, 1:3)

Figure 5. Color information on positional image blocks

For an image, its image identifier, the average RGB value, the average HSI value, and average RGB/HSI values of each color regions are stored in the database as show in Figure 6. The search conditions can be specified using the appropriate color positions. The system searches the matching images by comparing the average color of the whole image and the average colors of the specified color regions simultaneously.

INTEGRATED WEB IMAGE RETRIEVAL

The IIMS supports various search interfaces on web images. It is possible to present queries by texture, color, keyword, or by combining some or all of these factors. The search patterns of individual users are recorded in the system log table.

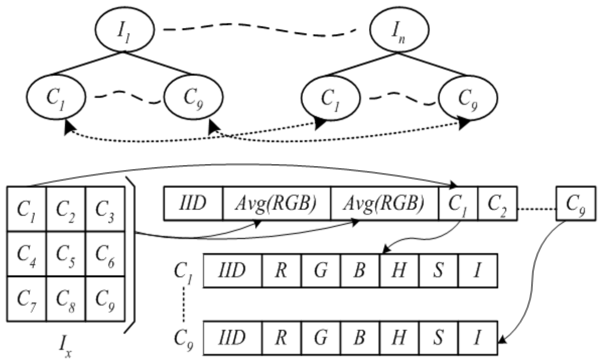


Figure 6. Storage structure of color image information

Retrieval Technique by Texture

Users can select zero or any texture patterns from the query by texture interface. The user id and his/her selected texture values are recorded in the log table for UUP processing. We apply the exclusive-OR operation on the texture bit vectors of query image and each stored image. In this operation, we exclude the case where the values of both bit positions are 0. Then, we complement its result, and apply bitwise OR operations on the complemented result. The final value 1 means two images are matched. Figure 7 summarizes this procedure. To retrieve the matched images, the appropriate SQL query is generated and its results are returned to users. Figure 8 shows an example of query by texture.

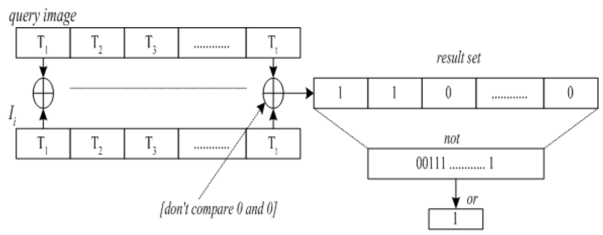


Figure 7. Matching by texture image information

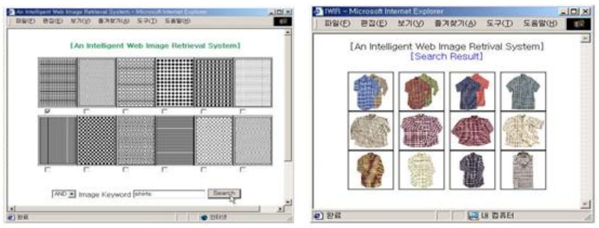

[An Intelligent Web Image Retrieval System]
Image Keyword
Search
[An Intelligent Web Image Retrival System]
[Search Result]

Figure 8. An example of query by texture

Retrieval Technique by Color

Users can specify RGB or HSI color values on the interested color regions through the query by color interface. Figure 9 shows an example of query by color. Alternatively, the color query can be given by uploading a sample image. The user id and his/her selected color values are recorded in the log table. The average values of the whole image or each color region is extracted and the appropriate SQL query is generated. The retrieval procedure for query by color is as follows (here, m=3).

Procedure 1. Query_by_color (var r_color:avg(R); g_color:avg(G); b_color:avg(B); h_color:avg(H); s_color:avg(S); i_color:avg(I));

```
   var rowcount : integer; x : integer;
   database connection.open;
 BEGIN
   FOR  x :=1 TO 9 DO
   IF ($C_x$ =Null)
     THEN  set R,G, B and H,S,I values as null;
   ELSE   /*  set of color range */
    open.recordset
    Rcolor_p := Cint (rs("r_color")) + 15; Rcolor_n := Cint (rs("r_color")) - 15;  /* tolerance ±15% */
    Gcolor_p := Cint (rs("g_color")) + 15; Gcolor n := Cint (rs("gcolor")) - 15;
    Bcolor_p := Cint (rs("b_color")) + 15; Bcolor_n := Cint (rs("b_color")) - 15;
    Hcolor _p := Cint (rs("h_color")) + 15; Hcolor_n : = Cint (rs("h_color")) - 15;
    Scolor _p := Cint (rs("s_color")) + 15; Scolor n := Cint (rs("s_color")) - 15;
    Icolor _p := Cint (rs("i_color")) + 15; Icolor n := Cint (rs("i_color")) - 15;
    close.recordset
  END IF
     /* Retrieve IID from color index */
    IF (rs.EOF := true) THEN query_result := Null;
    ELSE
    DO WHILE NOT rs.EOF
                  rowcount := rowcount + 1;
                  rs.movenext
         LOOP
         END IF
     database connection.close; /* database disconnection */
     set rs.NULL
    END;
END Query_by_color
End of procedure 1
```

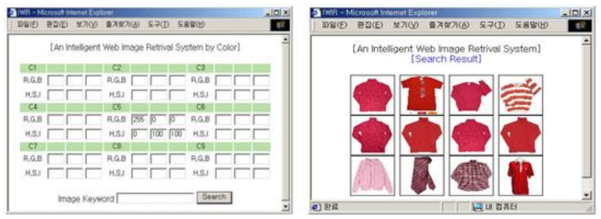

[An Intelligent Web Image Retrival System by Color]
Image Keyword
Search
[An Intelligent Web Image Retrival System]
[Search Result]

Figure 9. An example of query by color

Integrated Web Image Searching

Users can specify queries having search conditions related with keyword, texture, and color. Figure 10 shows the query processing procedure. The images are matched using texture and color indexes and the appropriate SQL query is generated using texture ids, color ids, and keywords. Figure 11 shows an example of query having search conditions on keyword, texture, and color.

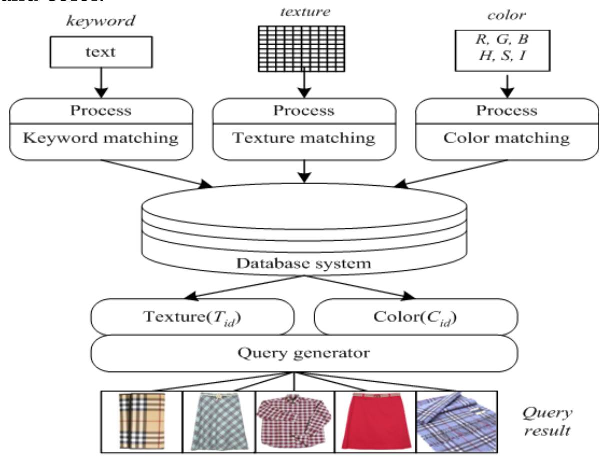


Figure 10. Integrated search procedure

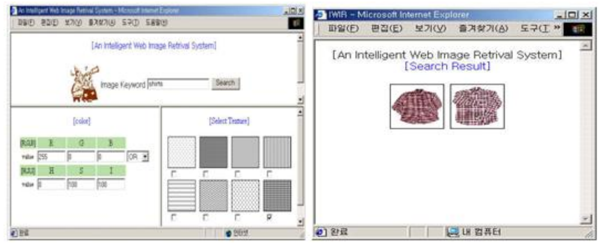

[An Intelligent Web Image Retrival System]
Image Keyword
Search
[color]
[Select Texture]
[An Intelligent Web Image Retrival System]
[Search Result]

Figure 11. An example of query by hybrid (keyword + texture + color)

MINING OF USER USAGE PATTERNS

The IIMS remembers user's preferences, so called user usage patterns. Some users may be interested in "red" shirts having "check" patterns. We utilize the log information on the previous queries to improve user's satisfaction on the later query results. The previous query logs are accumulated and used for texture and color mining. Figure 12 shows the outline of query log generation and mining procedure.

For the texture mining, user id and retrieved texture ids for each search transaction are stored in the texture log table. By applying the association mining procedure on the log table, the frequent item sets with their referenced count values can be generated. We choose the texture having the maximum count value as the preferable texture of the user. The texture mining procedure is as follows.

```
Procedure 2.  Texture_log_analysis ( $T_i := T_{id}$);
    var  i :=integer;
    BEGIN
       database connection.open;
       set of Log_T ;   /*  texture log table */
             FOR i :=1 TO n DO
/* scanning the $T_i$ item */
  select $T_{id}$ from Log_T where $T_{id}$ like $T_i$ and
($U_{id}$ :=session(Uid)) ;
  IF ($T_i \neq$ Null) THEN
       create temp_table_log_t
       finding frequent item sets;
  FOR i :=1 TO n DO
/* increment count */
  update temp_table_log_t set count := count +1 where
$T_{id}$={$T_i$};
  END IF
/* find maximum number */
   select Max(count) from temp_table_log_t
   IF (Count(rs(0)) >1) THEN
       select random item;
   ELSE
        select $T_i$ item;
    END IF
       database connection.close;
  set rs.NULL;
  END;
 END Texture_log_analysis
End of procedure 2
```

Figure 13 shows an example of texture based mining. For the user U_1, his previous queries show the texture patterns *[{T_1}, {T_1,T_2}, {T_2,T_3}, {T_1,T_3}, {T_1}]*. By choosing the maximum count value from the frequent item sets, we can find the user U_1 prefers the texture T_1.

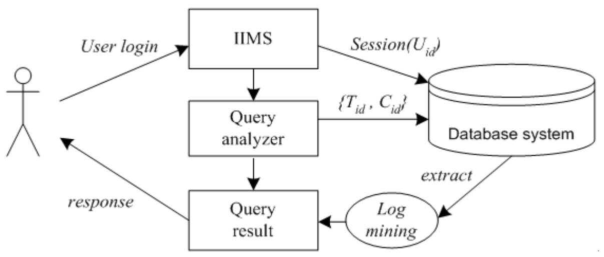
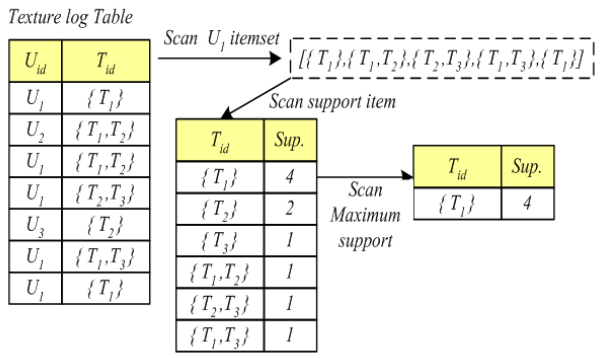


Figure 12. Query log generation and mining procedure



U_{id}	T_{id}
U_1	{T_1}
U_2	{T_1,T_2}
U_1	{T_1,T_2}
U_1	{T_2,T_3}
U_3	{T_2}
U_1	{T_1,T_3}
U_1	{T_1}

T_{id}	Sup.
{T_1}	4
{T_2}	2
{T_3}	1
{T_1,T_2}	1
{T_2,T_3}	1
{T_1,T_3}	1

T_{id}	Sup.
{T_1}	4

Figure 13. An example of texture mining

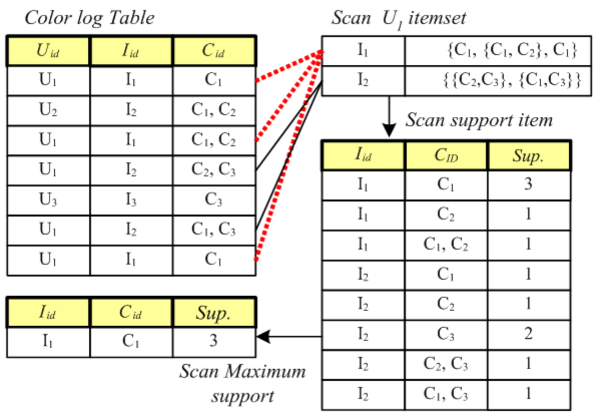
The color mining procedure is similar with the texture mining procedure. This procedure is as follows.

```
Procedure 3. Color_log_ analysis ( $C_j := C_{id}$; $I_i := I_{id}$);
   var j :=integer ; i :=integer;
   BEGIN
      database connection.open;
        set of Log_C ;  /*  color log table */
             FOR j :=1 TO n DO

/* scanning the $I_i$, $C_j$ items */
  select $I_{id}$, $C_{id}$ from Log_C where $U_{id}$ :=session(Uid) ;
  create temp_table_log_c
  IF ($C_j \neq$ Null) THEN
                  result into temp_table_log_c ;
                  finding frequent item sets;
                  FOR j :=1 TO j DO
                  FOR i :=1 TO i DO
/* increment count */
   update temp_table_log_c set count := count +1 where
($C_{id}$ :={$C_j$} and $I_{id}$ := $I_i$);
  END IF

/* to find maximum number */
   select Max(count) from temp_table_log_c
   IF  (Count(rs(0)) >1) THEN
/*  more than 1 item */
        select random item;
   ELSE
         select $C_j$ item;
   END IF
         database connection.close;
         set rs.NULL
   END;
END Color_log_ analysis
End of procedure 3
```

Figure 14 shows an example of color based mining. For the user U_1, his previous queries show the color patterns *{C_1, C_1, C_2, C_1}* on the image I_1 and *{C_2, C_3, C_1, C_3}* on the image I_2. By choosing the maximum count value from the frequent item sets, we can find the user U_1 prefers the color C_1.



U_{id}	I_{id}	C_{id}
U_1	I_1	C_1
U_2	I_2	C_1, C_2
U_1	I_1	C_1, C_2
U_1	I_2	C_2, C_3
U_3	I_3	C_3
U_1	I_2	C_1, C_3
U_1	I_1	C_1

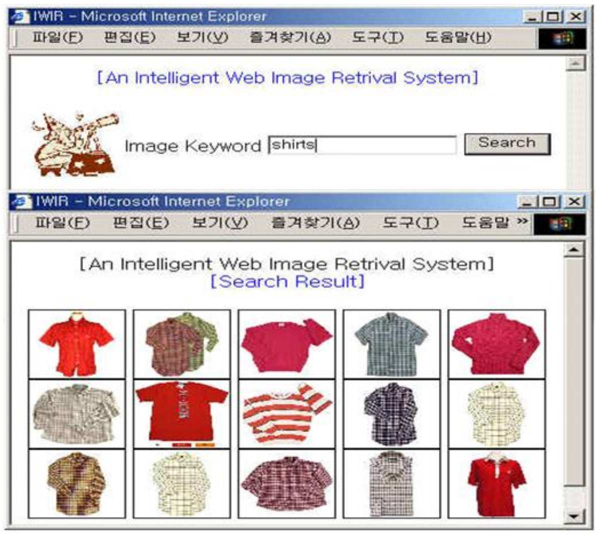
I_1	$\{C_1, \{C_1, C_2\}, C_1\}$
I_2	$\{\{C_2,C_3\}, \{C_1,C_3\}\}$

I_{id}	C_{ID}	Sup.
I_1	C_1	3
I_1	C_2	1
I_1	C_1, C_2	1
I_2	C_1	1
I_2	C_2	1
I_2	C_3	2
I_2	C_2, C_3	1
I_2	C_1, C_3	1

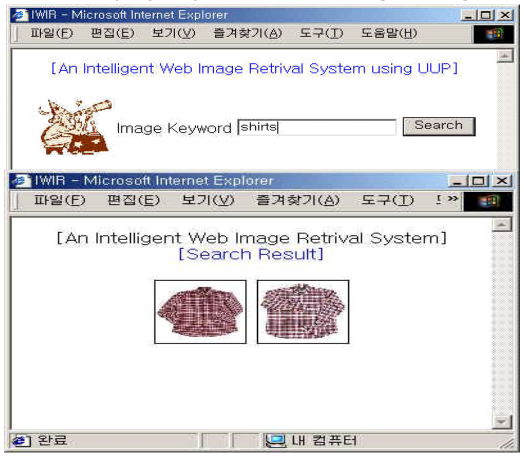
I_{id}	C_{id}	Sup.
I_1	C_1	3

Figure 14. An example of color mining

These mining results are used to filter the query results. Figure 15 compares the results of query by keyword "shirts" before and after the user log mining is applied. After the query log mining is applied, the number of matched images is reduced, thus resulting in shorter response time. This will be quite useful to the e-commerce site users, suffering from a lot of uninterested search results.


IWIR - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
[An Intelligent Web Image Retrival System]
Image Keyword shirts
Search
IWIR - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말
[An Intelligent Web Image Retrival System]
[Search Result]

(a) Query by keyword without log mining


IWIR - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
[An Intelligent Web Image Retrival System using UUP]
Image Keyword shirts
Search
IWIR - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T)
[An Intelligent Web Image Retrival System]
[Search Result]
완료
내 컴퓨터

(b) Query by keyword reflecting user's preferences

Figure 15. Effects of user log mining

EXPERIMENTAL RESULTS

We have developed the prototype system on the Pentium CPU 2.80GHz and 2GB of RAM, Windows 2003 platform using SQL Server 2005. This system was implemented by Active Server Pages (ASP) and Java technology. We have used 1000 web e-catalog images. Table 1 shows performance results on various search combinations measured in terms of recall and precision. This table also shows that the results combining keyword, texture, and color search conditions are better than single feature queries. Also, by applying the user usage pattern, the degree of retrieval correctness has been quite improved even for the case of query by keyword.

CONCLUSION

In this paper, we presented an intelligent e-catalog image retrieval system IIMS. We have proposed the texture and color based image indexing techniques. To

Table 1. Performance measures in recall and precision

no	Keyword	Color(RGB,HIS)	Texture	UUP	Recall	Precision
1	*"shirts"*	-	-	-	*72%*	*42%*
2	-	*(253,0,0) (100,0,0) ±1.5%*	-	-	*83%*	*86%*
3	-	-	*100…000*	-	*75%*	*90%*
4	-	*(253,0,0) (100,0,0) ±1.5%*	*100…000*	-	*83%*	*96%*
5	*"shirts"*	*(253,0,0) (100,0,0) ±1.5%*	*100…000*	-	*83%*	*99%*
6	*"shirts"*	-	-	o	*83%*	*99%*

support fast retrieval, we utilized the bit vector indexing for textures representing the presence of each texture pattern by 1 bit. For positional color search, we divided given images into region blocks and stored the average color of each block in system databases. The query can be given by providing keywords, by selecting one or more sample texture patterns, by assigning colors within positional color blocks, or by combining some or all of these factors. The system is designed to remember user's preferences by mining user query patterns, so that it can add more feature information automatically to subsequent user queries.

We are planning to develop an integrated e-catalog image search engine by combining the image crawling agent and automatic image caption extractor module. There should be further researches to improve the performance of the feature extractor modules and the matching modules on massive amount of web images.

REFERENCES

1. M.S. Chen, J. Han, and P.S. Yu. (1996). Data Mining: An Overview from a Database Perspective, *IEEE Trans. Knowledge and Data Engineering*, pp. 8:866-883.
2. O. Zaiane and J. Han. (1998). WebML: Querying the World-Wide Web for Resources and Knowledge,' *in Proc. Int'l Workshop on Web Information and Data Management (WIDM'98)*, pp. 9-12.
3. J. Han and Y. Fu. (1994). Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Database, *in Proc. AAAI'94 Workshop on Knowledge in Databases (KDD'94)*, Seattle, WA, pp.157-168.
4. R. Agrawal and R. Srikant. (1995). Mining Sequential Patterns, *in Proc. of the Int'l Conference on Data Engineering (ICDE)*, Taipei, Taiwan, pp.3-14.
5. R. Agrawal and R. Srikant. (1994). Fast Algorithms for Mining Association Rules, *in Proc. VLDB, Santiago, Chile*, pp.487-499.
6. D.-H. Lee, D.-Y. Seo, N.-H. Kim, J.-Y. Lee. (1998). Discovery and Application of User Access Patterns in the World Wide Web, *in Proc of the 4th World Congress on Expert Systems*, pp.321-327.
7. R. Agrawal, T. Imielinski, and A. Swami. (1993). Mining Association Rules Between Sets of Items in Large Databases. *In Proceedings of the 12th ACM SIGMOD International Conference on Management of Data*, pp.207-216.
8. Mark Hornick. (2006). Java™ Specification Request 247: Java™ Data Mining(JDM)2.0, *JSR-247 Expert Group*. Available: http://www.jcp.org/en/jsr/detail?id=247
9. Niblack, W., Barber, R., Equitz, W., Flickner, M., Glasman, E., Petkovic, D., Yanker, P, & Faloutsos, C. (1993). The QBIC Project: Querying images by content using color, texture, and shape, *in Proc. SPIE Storage and Retrieval for Image and Video Database*, pp.173-187. Available: http://Iibra.ucdavis.edu/cgi-bin/QbicStable/
10. V. N. Gudivada and V. V. Raghavan. (1995). Content-Based Image Retrieval Systems, *IEEE Computer, 28(9)*, pp.18-22.
11. C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equiz. (1994). Efficient and Effective Querying by Image Content, *Journal of Intelligent Information System(JIIS), 3(3)*, pp.231-262.
12. Smith, J. R. & Chang, S.-F. (1996). VisualSEEk: A Fully Automated Content-Based Image Query System, *ACM Multimedia 96*, pp. 87-98.
13. Li, Jia., Wang, James Z., & Wiederhold, Gio. (2000). IRM: Integrated region matching for image retrieval, *Proc. ACM Multimedia*, Los Angeles, ACM, pp. 147-156.
14. Jia. Wang, et.al. (1997). Color Clustering Techniques for Color Content-B Image Retrieval from Image Database, *in Proc. of the International Conference on Multimedia Computing and Systems*, pp. 442-449.
15. J.Li, J.Z. Wang, G. Wiederhold. (2000). Classification of texture and non-textured images using region segmentation, *in Proc. of the Seventh International Conference on image Processing, Vancouver, BC, Canada*, pp. 754-757.
16. J. Han, Y. Huang, N. Cercone and Y. Fu. (1996). Intelligent Query Answering by Knowledge Discovery Techniques, *IEEE Transactions on Knowledge and Data Engineering*, *8(3)*, pp.373-390.
17. Stricker, M. & Orengo, M. (1995). Similarity of color images, *Proc. SPIE on Storage and Retrieval for Image and Video, Databases, Vol. 2420*, San Jose, USA, pp. 381-392.
18. Swain, M. J. & Ballard, S. H. (1991). Color Indexing, *Int. Journal of Computer Vision, Vol.7, No. 1*, pp. 11-32.
19. Rickman, R. & Stonham, J. (1996). Content-based image retrieval using color tuple histograms, *SPIE proceedings*, 2670: 2-7.
20. Smith, J. & Chang, S.-F. (1996). Tools and techniques for color image retrieval, *SPIE proceedings*, 2670: 1630-1639.
21. Ogle, V.E. and Stonebraker, M., (1995). Chabot: Retrieval from a Relational Database of Images, *IEEE Computer*, Sept, pp.40-48.
22. Smith, J. R. & Chang, S.-F. (1995). Single Color extraction and image query, *in Proc. ICIP, vol. 3*, pp. 528-531.