Unit 7: Content-Based Retrieval in Digital Libraries

CS 3570

Shang-Hong Lai

Chapter 21
Content-Based Retrieval in Digital Libraries

How Should We Retrieve Images?

- For Bosch's painting in Fig. 21.1, a text-based search will likely do the best job, provided the multimedia database is fully indexed with proper keywords.
- Most multimedia retrieval schemes, however, have moved toward an approach favoring multimedia content itself ("content-based").
- Many existing systems retrieve images with the **image features**, e.g., color histogram, color layout, texture, etc.

Hieronymus Bosch, *The Garden of Earthly Delights*, oil on oak panels, 205.5 cm × 384.9 cm (81 in × 152 in), Museo del Prado, Madrid

C-BIRD: An Early Experiment

- **C-BIRD** (Content-Base Image Retrieval from Digital libraries): an image database search engine. It contains approximately 5,000 images, many of them keyframes from videos.
- The database can be searched using a selection of tools: text annotations, color histograms, illumination-invariant color histograms, color density, color layout, texture layout, and model-based search.

Color Histogram

- The most prevalent feature that is utilized in image retrieval is the color histogram.
- A color histogram counts pixels with a given pixel value in Red, Green, and Blue (RGB).
- Image search is done by matching **feature-vector** (color histogram) for the sample image with feature-vector for images in the database.
- In C-BIRD, a color histogram is calculated for each target image, and then referenced in the database for each user query image.

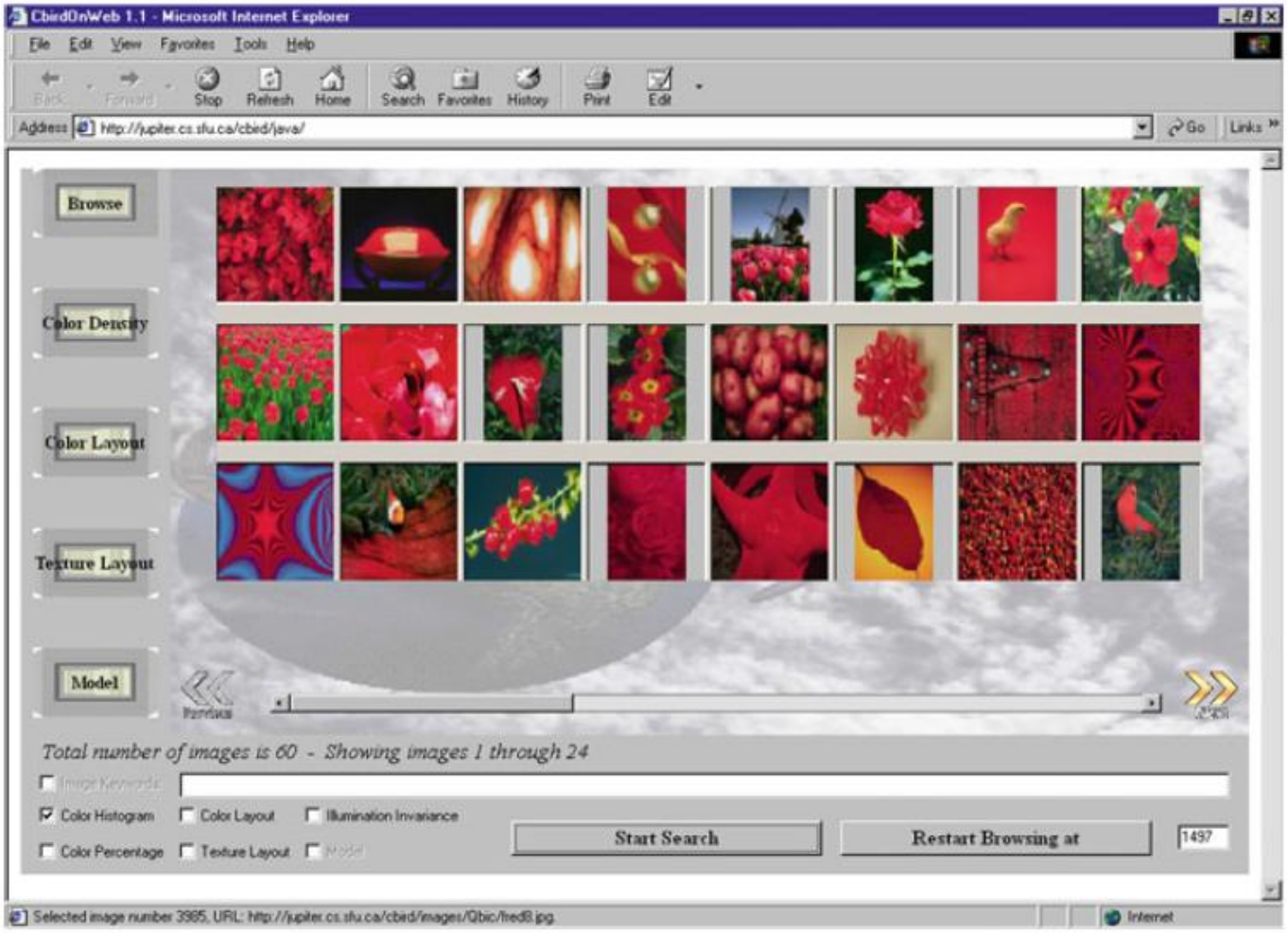

Fig. 21.2: Search by color histogram results.

Histogram Intersection

- **Histogram intersection**: The standard measure of similarity used for color histograms:
  - A color histogram $\mathbf{H}_i$ is generated for each image i in the database and then *normalized*
  - The new image will match against all possible targets in the database.
  - Its histogram $\mathbf{H}_m$ is intersected with all database image histograms $\mathbf{H}_i$ according to the equation. 1 is perfect match.

$$\text{intersection} = \sum_{j=1}^{n} \min(\mathbf{H}_i^j, \mathbf{H}_m^j)$$

Color Density and Color Layout

- For color density, what percentage of the image having any particular color or set of colors is selected by the user, using a color-picker and sliders. User can choose from either conjunction (ANDing) or disjunction (ORing) a simple color percentage specification.
- For color layout, the user can set up a scheme of how colors should appear in the image, in terms of coarse blocks of color. The user has a choice of four grid sizes: 1 × 1, 2 × 2, 4 × 4 and 8 × 8.

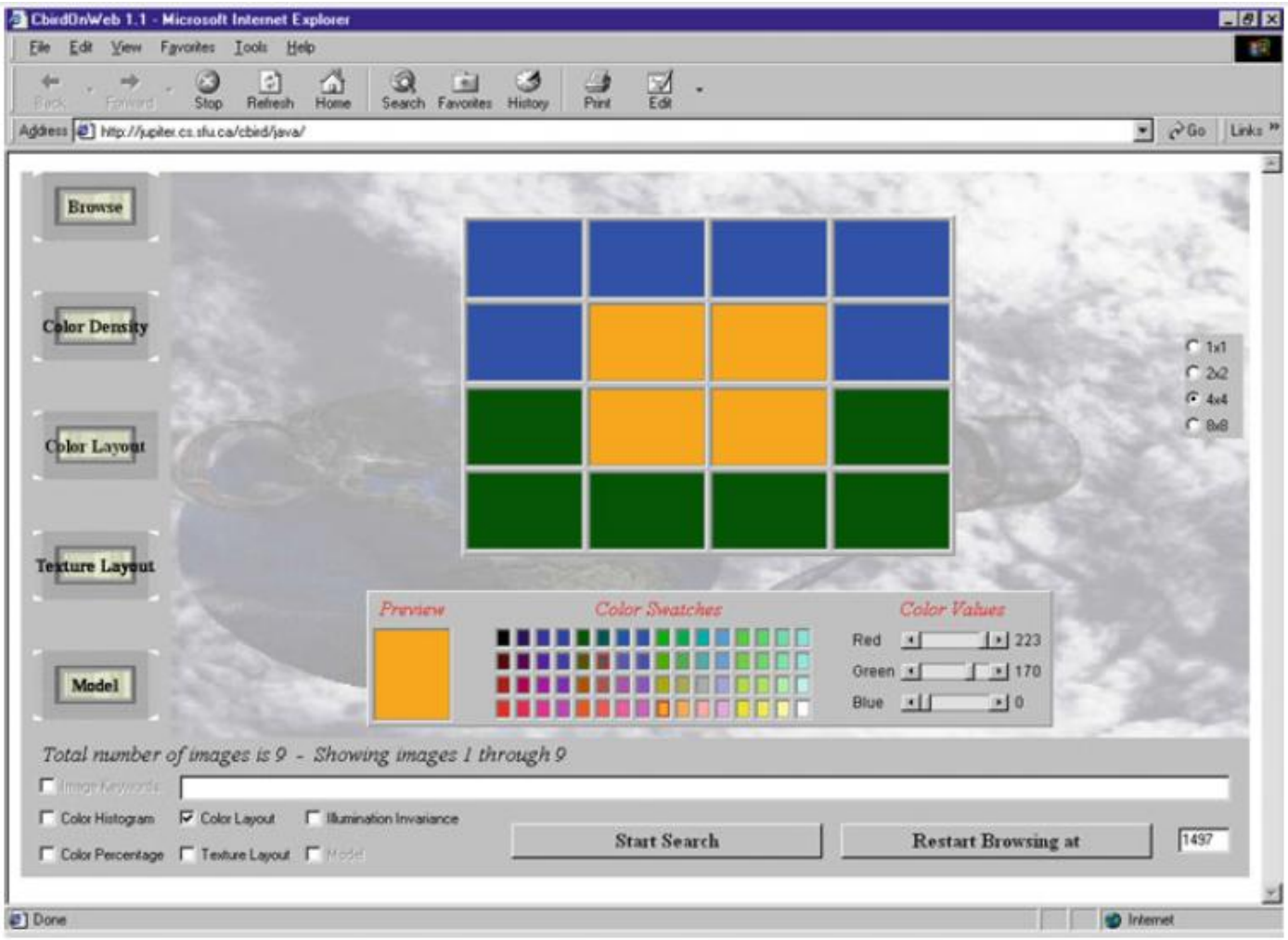

Fig. 21.3: Color layout grid

Texture Layout

- This query allows the user to draw the desired texture distribution.
- **Available textures**: zero edge density, medium or high density edges in four directions (0°, 45°, 90°, 135°) and combinations of them.
- **Texture matching** is done by classifying textures according to directionality and density (or separation), and evaluating their correspondence to the texture distribution selected by the user.

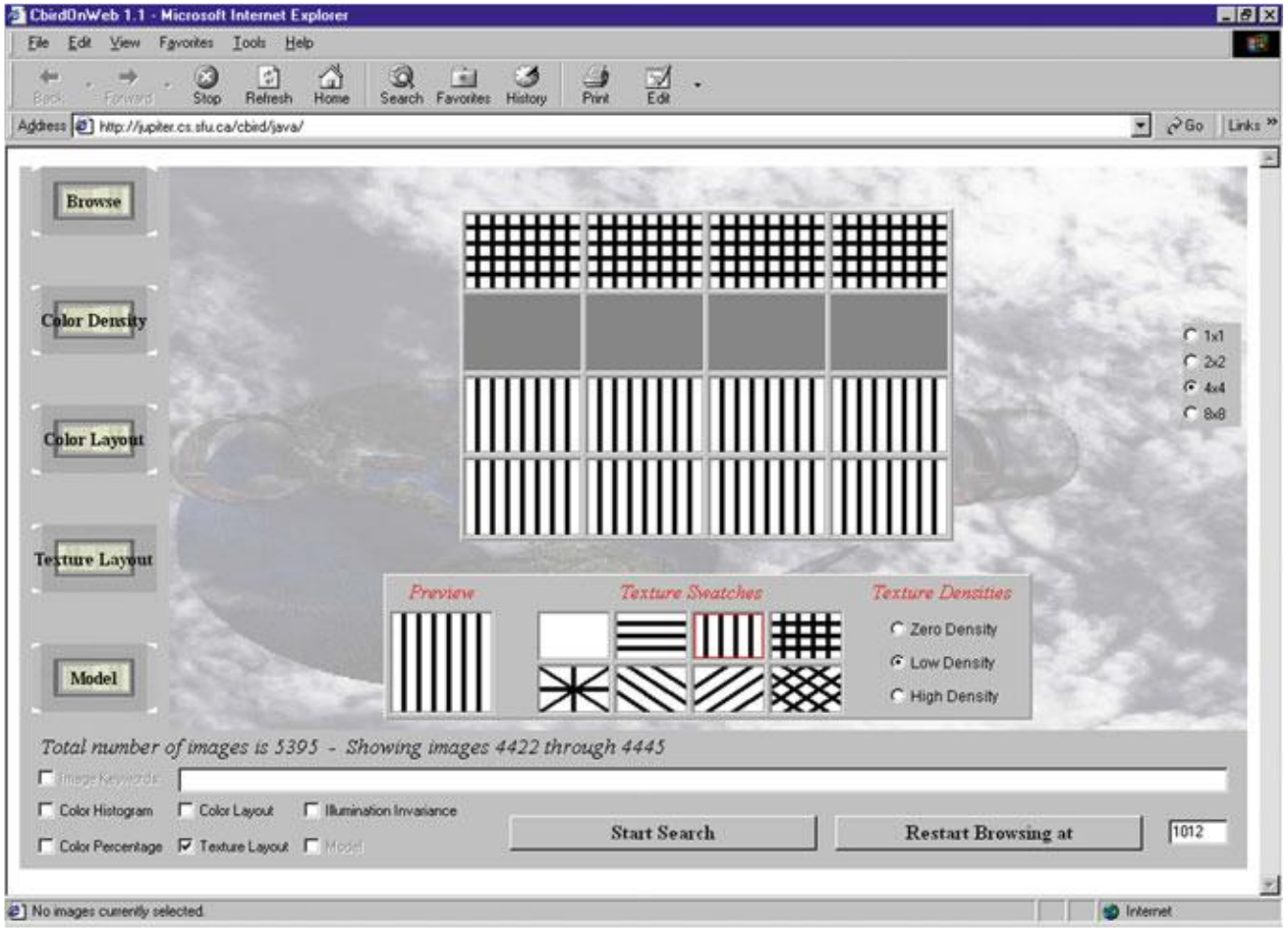

Fig. 21.4: Texture layout grid.

Texture Analysis Details

- Edge-based texture histogram
- A 2-dimensional texture histogram is used based on edge directionality ϕ, and separation ξ
- A *Sobel edge operator* is applied to the Y -image by sliding the following 3×3 weighting matrices over the image.

$$d_x: \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad d_y: \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

- The edge magnitude D and the edge gradient ϕ are given by

$$D = \sqrt{d_x^2 + d_y^2}, \qquad \phi = \arctan \frac{d_y}{d_x}$$

Texture Analysis Details (Cont'd)

- Preparation for texture histogram

- If a pixel i with edge gradient ϕ_i and edge magnitude D_i has a neighbor pixel j along the direction of ϕ_i with gradient $\phi_j \approx \phi_i$ and edge magnitude $D_j > D_i$ then pixel i is suppressed to 0.

- To make a binary edge image, set all pixels with D greater than a threshold value to 1 and all others to 0.

- For edge separation ξ, for each edge pixel i we measure the distance along its gradient ϕ_i to the nearest pixel j having $\phi_j \approx \phi_i$ within 15°. If such a pixel j doesn't exist, the separation is considered infinite.

Texture Analysis Details (Cont'd)

- 2D texture histogram of ξ vs. ϕ is constructed.

- The initial histogram size is 193 × 180, where separation value ξ = 193 is reserved for a separation of infinity (as well as any ξ > 192).

- The histogram is "smoothed" by replacing each pixel with a weighted sum of its neighbors, and then reduced to size 7 × 8, separation value 7 reserved for infinity.

- Finally, the texture histogram is normalized by dividing by the number of pixels in the image segment. It will then be used for matching.

Search by Illumination Invariance

- Each color channel band is normalized and compressed to 36-vector.
- A 2-dimensional color histogram is then created by using the *chromaticity*, which is the set of band ratios $\{R,G\}/(R+G+B)$.
- A 128 × 128–bin 2D color histogram is treated as an image and compressed using a wavelet-based compression scheme.
- The DCT coefficients for the smaller histogram are calculated and placed in zigzag order.
- Matching is performed in the compressed domain by taking the Euclidean distance between two 36-component feature vectors.

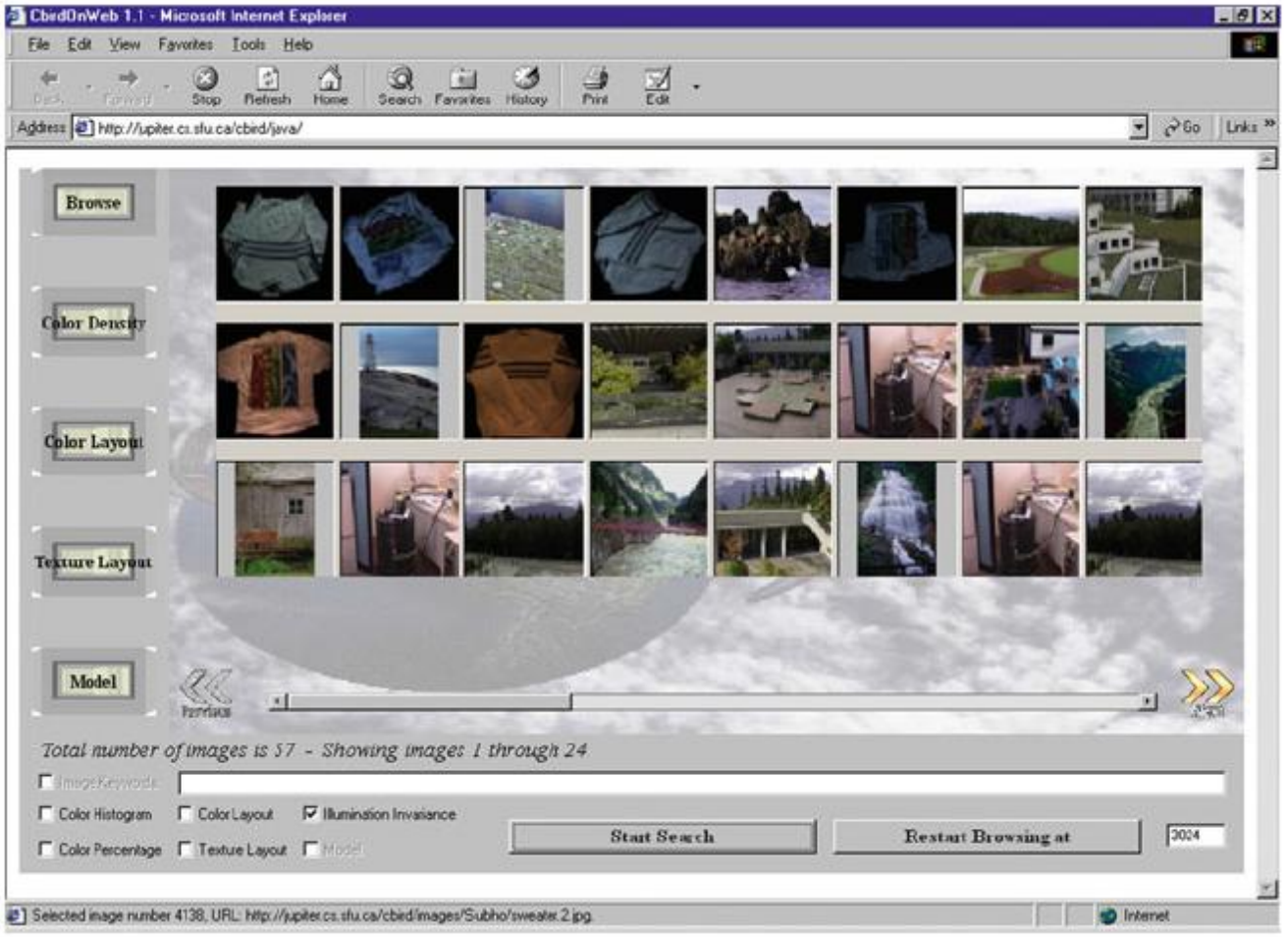

Fig. 21.5: Search with illumination invariance.

Search by Object Model

- This search type proceeds by the user selecting a thumbnail and clicking the *Model* tab to enter object selection mode.
  - An image region can be selected by using primitive shapes such as a rectangle or an ellipse.
  - An object is then interactively selected as a portion of the image.
  - Multiple regions can be dragged to the selection pane, but only the active object in the selection pane will be searched on.

Search by Object Model (Cont'd)

- The user-selected model image is processed and its features localized.
- Color histogram intersection is applied as a first *screen*.
- Estimate the pose (scale, translation, rotation) of the object inside a target image.
- Verification by intersection of texture histograms, and then a final check using an efficient version of a Generalized Hough Transform for shape verification.

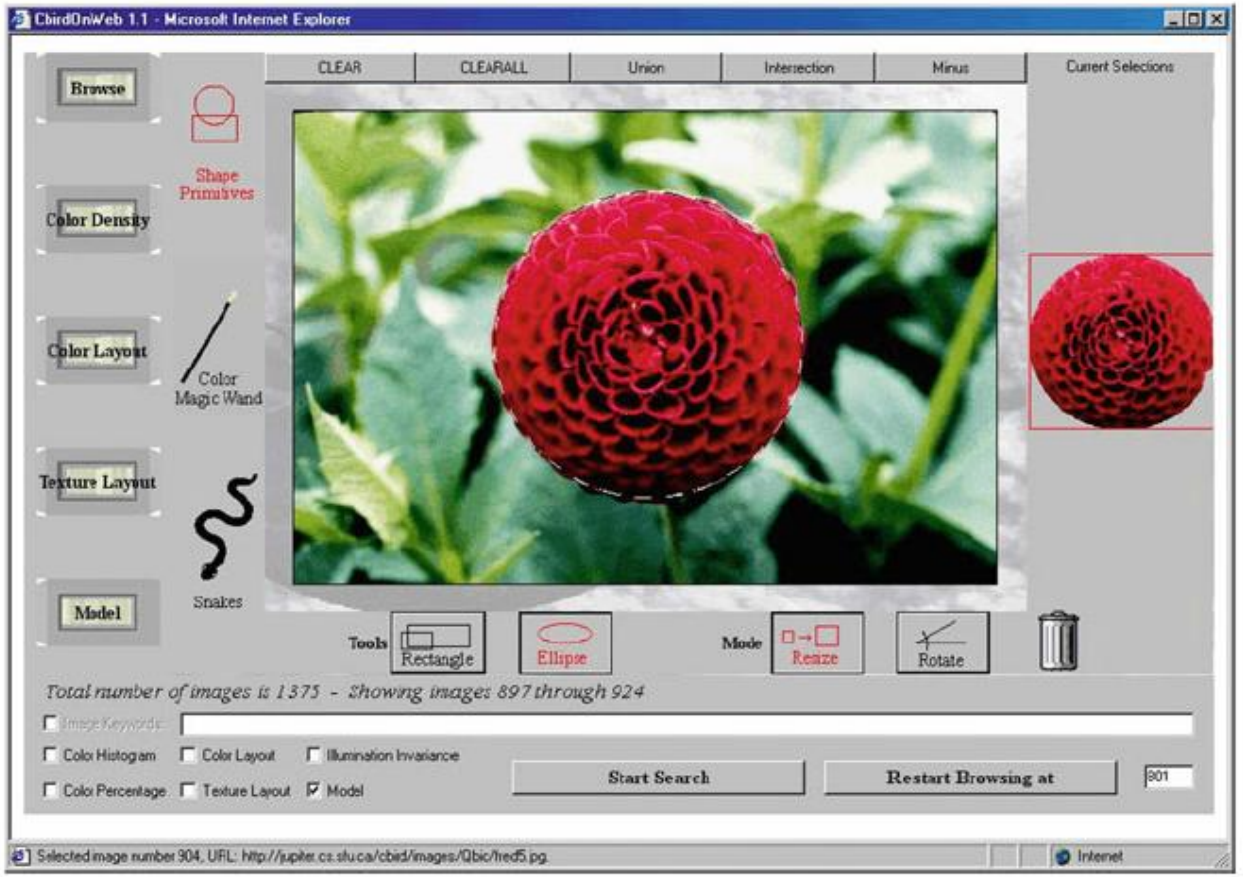

Fig. 21.6: C-BIRD interface showing object selection using an ellipse primitive.

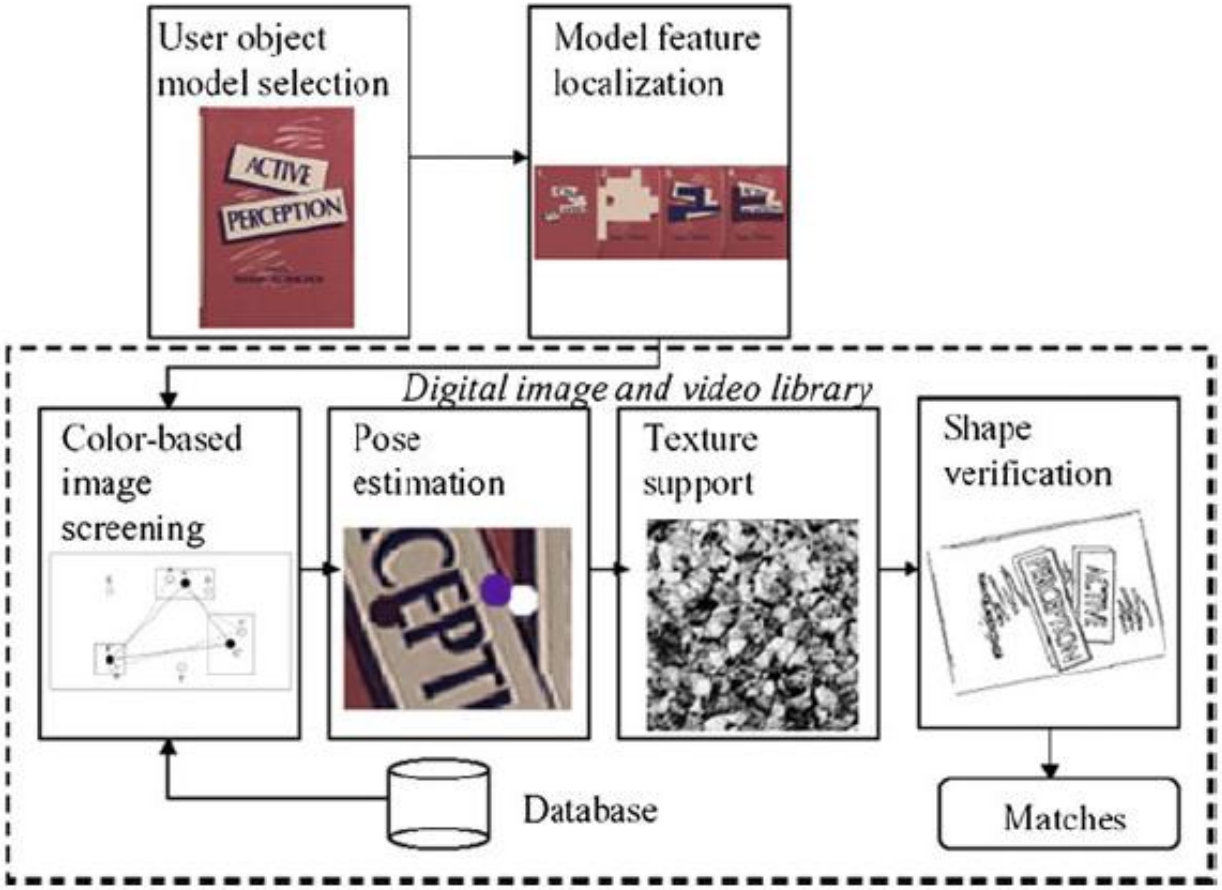

Fig. 21.7: Block diagram of object matching steps

Quantifying Results

- **Precision** is the percentage of relevant documents retrieved compared to the number of all the documents retrieved.
- **Recall** is the percentage of relevant documents retrieved out of all relevant documents.

$$Precision = \frac{Relevant\ images\ returned}{All\ retrieved\ images}$$

$$Recall = \frac{Relevant\ images\ returned}{All\ relevant\ images}$$

Quantifying Results (Cont'd)

- Alternatively, they may also be written as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

where TP (True Positives) is the number of relevant images returned, FP (False Positives) is the number of irrelevant images returned, and FN (False Negatives) is the number of relevant images not returned.

Quantifying Results (Cont'd)

- The Average Precision of a single query q is defined, the Precision(n) is the one after the nth relevant image was retrieved, N_R is the total number of relevant images

$$AP(q) = \frac{1}{N_R} \sum_{n=1}^{N_R} \text{Precision}(n) \tag{21.8}$$

- *Mean Average Precision (MAP)* is the mean of Average Precisions over all query images, where Q is the query image set, and N_Q is its size.

$$MAP = \frac{1}{N_Q} \sum_{q \in Q} AP(q) \tag{21.9}$$

Quantifying Results (Cont'd)

- *Receiver Operating Characteristic (ROC)* curve is a general statistical measure for various classifiers. It evaluates the TPR (True Positive Rate) vs. FPR for any systems, so that we can get the whole picture of the performance.

- ROC has its applications in many disciplines, e.g., psychology, physics, medicine, and increasingly in machine learning and data mining.

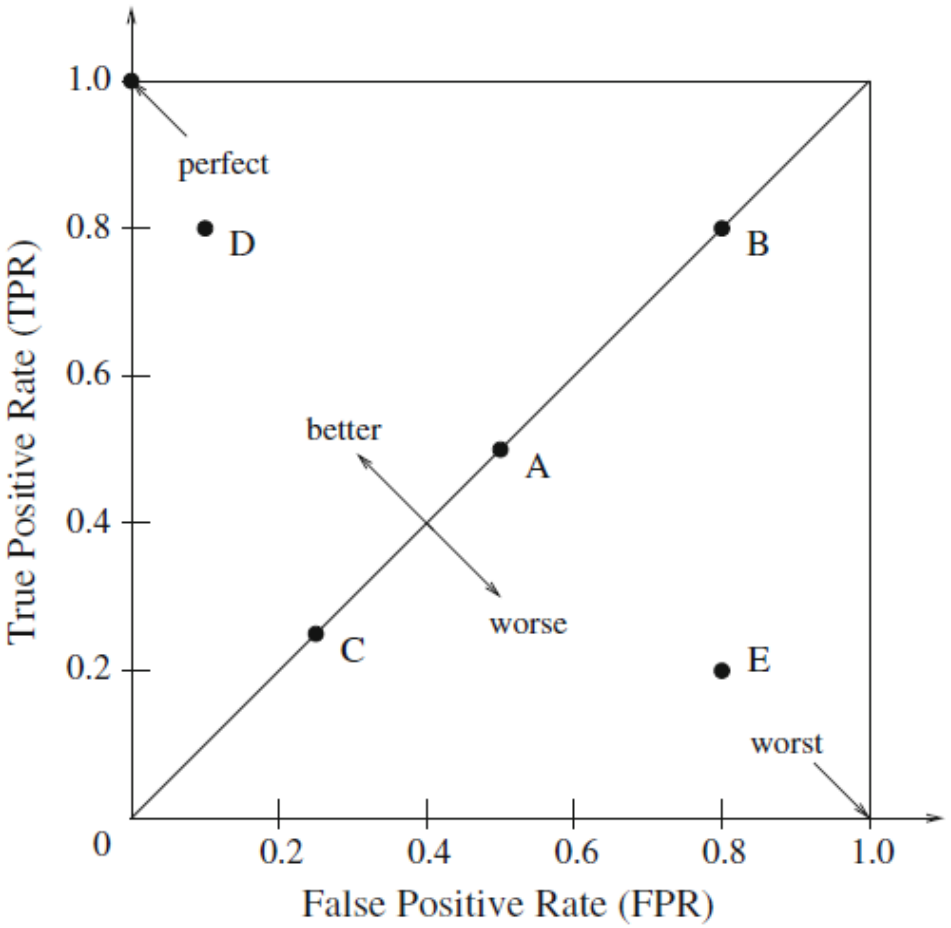

Fig. 21.10: The ROC space

Key Technologies in CBIR Systems

- Robust Image Features
- Relevance Feedback
- Other Post-processing Techniques
- Visual Concept Search
- The Role of Users in Interactive CBIR Systems

Robust Image Features

- **SIFT** (*Scale Invariant Feature Transform*)
  - Build a multi-resolution scale space and applying Gaussian smoothing.
  - Generates Difference of Gaussian (DOG) images.
  - Key point detection.
  - Generate a 128-dimensional SIFT descriptor from the local patch (16×16 pixels) at key point. 128 dimensions are derived from 4x4 sub-windows and 8 gradient orientation bins.

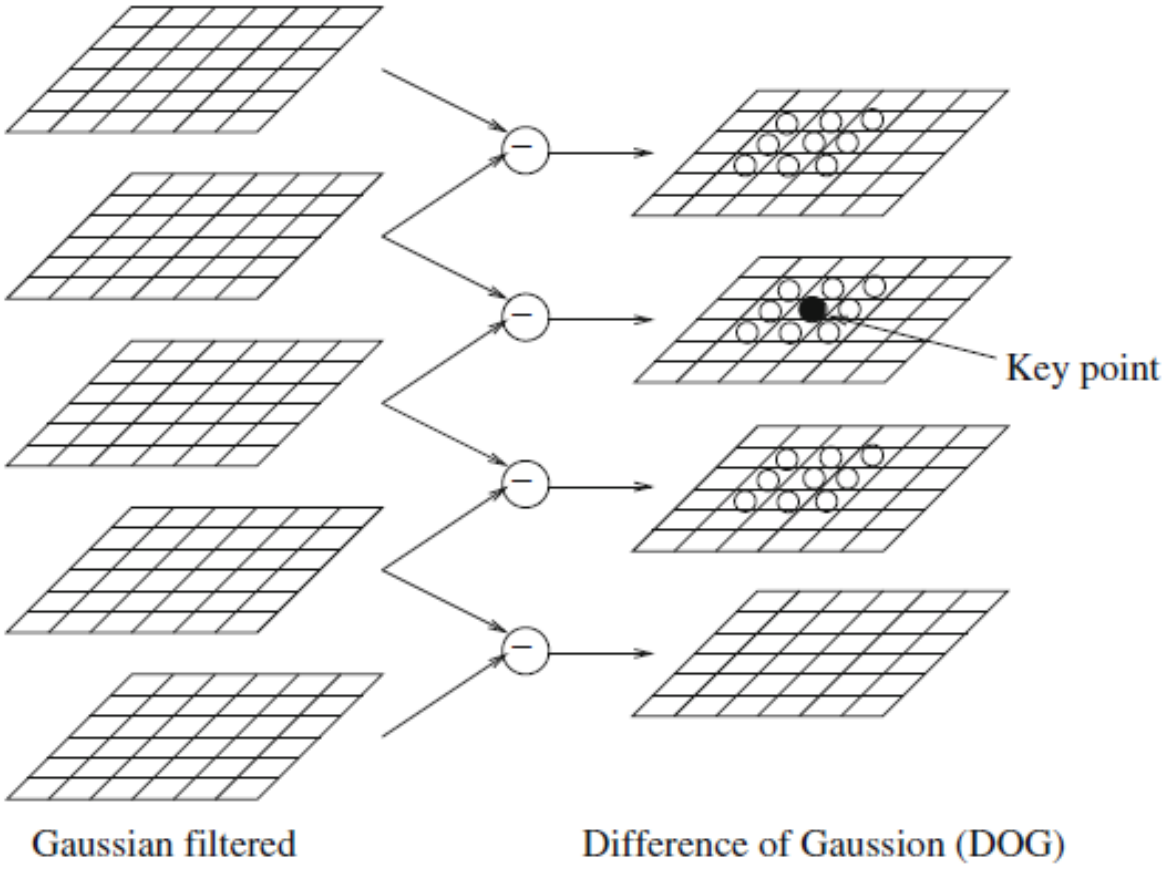

Fig. 21.11: SIFT key point extraction in one octave

Robust Image Features (Cont'd)

- Visual Words

- A mixed bag of *words* can be extracted from a query sentence and be used for a query. It could also be used to extract image features.

- In CBIR, a common way of generating Visual Words is to use SIFT because of its good properties discussed above. This can be (a) object based or (b) video frame based.

- Visual Words are rich in encapsulating essential visual features. However, compared to words from text, Visual Words are even more ambiguous.

User Feedback and Collaboration

- *Relevance feedback* is to involve the user in a loop, whereby images retrieved are used in further rounds of convergence onto correct returns.
- Relevance feedback establishes a more accurate link between low-level features and high-level concepts, somewhat closing the semantic gap.

User Feedback and Collaboration (Cont'd)

- Study what forms our basis of perception of image similarity. The function used in this approach can be a type of "perceptual similarity measure" and is *learned* by finding the best set of features (color, texture, etc.) to capture "similarity".
- In addition to *content-based*, we need to be *context-based*, because the user's interpretation of the content is often influenced (or even determined) by the context.

Post-processing Techniques

- Spatial Verification
  - Check for consistency or relevance in certain query results if the user is only interested in images from certain locations.
- Query Expansion
  - Some combination of high-ranked relevant documents can be presented as a new query to boost the performance of the retrieval.
- Question-answering (QA)

Visual Concept Search

- Typically, after local features are turned into words, the words are converted to semantic concepts by machine learning algorithms.
- The TREC Video Retrieval Evaluation (TRECVID) benchmark
  - Semantic indexing
  - Interactive surveillance event detection
  - Instance search
  - Multimedia event detection
  - Multimedia event recounting

Feature Learning with Convolutional Neural Networks

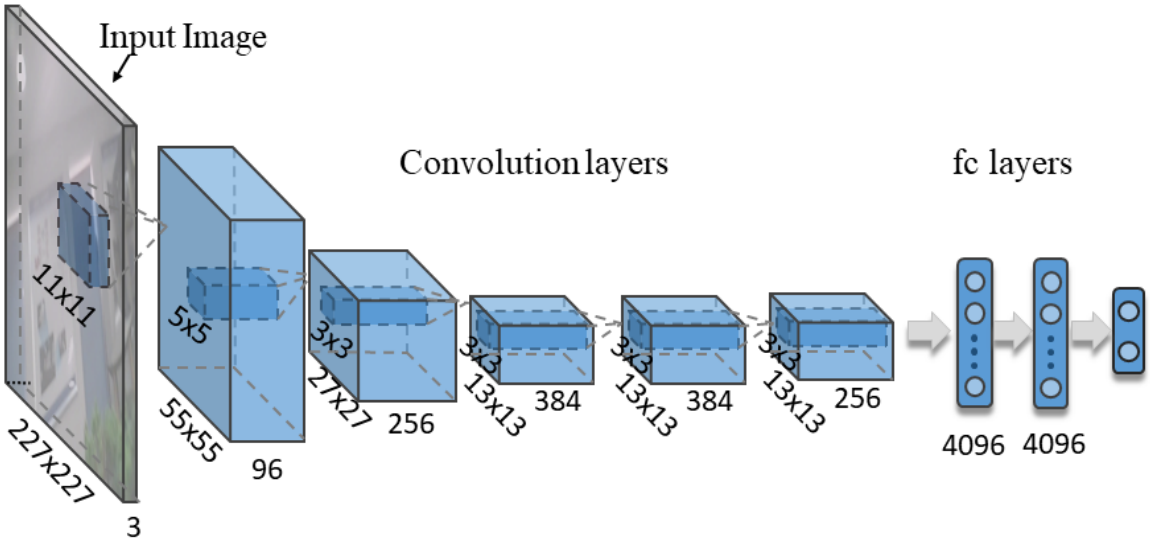

Fig. 21.12 An Illustration of a Typical Convolutional Neural Network Architecture.

- The generic feature descriptors extracted from the CNNs are very powerful and discriminative for a diverse range of visual recognition tasks.
- They can be fine-tuned by first initializing the network weights with the pre-trained CNN and then re-train the model on the target dataset.
- Other schemes: (a) Use similarity learning based on the extracted features from the pre-trained model. (b) Fine-tune the CNN with loss function based on cosine similarity.

Database Indexing

- The scale of image database has been growing explosively and the response time becomes a key issue in content-based image retrieval.
- Simple nearest neighbor search algorithm that matches the query with all images is considered too time consuming. Even optimized search with structures such as K-D tree is not enough.
- Approximate nearest neighbor search methods are proposed to tackle this problem.
- Database organizing and indexing are required to enable fast approximate retrieval.

Inverted Indexing:

- The Inverted Indexing file structure can be considered as a matrix whose rows and columns denote images and visual words, respectively.
- To retrieve similar images from the database, only those that contain the same visual words as the query are checked.
- For very large databases, *Inverted Multi-index* is proposed.

Hashing Based Indexing:

- Partition the database by using multiple hash functions to project the features into several buckets.
- The Locality Sensitive Hashing (LSH) can be used.
- The hash functions can also be learned using deep convolutional neural networks.

Deep Incremental Hashing Network

- Hash functions learned by convolutional neural networks significantly outperform conventional hashing in many applications.
- *Deep Incremental Hashing Network* (DIHN), is proposed for learning hash codes in an incremental way. DIHN incrementally learns the hash codes for new images while holding those of the original ones invariant.
- An incremental hashing loss is devised to preserve the similarities between images.