

Chapter 4 - Image

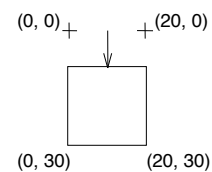


Vector Graphics

- Raw data:
 - set (!) of lines and polygons with coordinates and attributes (line width, color)
- Registration data:
 - coordinate system (cartesian, polar)
 - definitions for colors, textures
- Description data:
 - grouping related lines and polygons into geometric, higher order objects: squares, rectangles, projections of 3D-objects

start point end point width

0	30	20	30	2
9	8	10	10	1
20	10	20	30	2
0	10	0	30	2
10	0	10	10	1
11	8	10	10	1
20	10	0	10	2



a) set of line definitions

b) resulting graphics

Vector Graphics (2)

- Operations:
 - focus on output
 - manipulation by special editors (not part of the repository)
 - "simple" manipulations in the repository without replacing the complete graphics
- Input: via file or interchange format
- Output:
 - to a file, on a device (display, plotter)
 - line by line (numberOfLines, getAllLines)
- Modification:
 - add/delete line
 - translate, rotate, resize
- Analysis, aggregation, extraction
 - extract rectangular section (clipping)
 - reconstruction of geometric objects

Search

- Comparison (for search)
 - using description data (-> text)
 - using raw data
 - similarity of geometric figures (for very simple graphics)
- Significant extensions required for
 - curves
 - surfaces
 - segments (collection of arbitrary related elements)
- Subtypes:
 - technical drawing (CAD)
 - bar charts, flow charts, pie charts
 - street maps
 - ...

(Raster) Image

- Raw data
 - pixel matrix (pixel = picture element, or "pel")
- Registration data
 - # of bits/pixel (pixel depth, usually 1, 8, or 24)
 - # of pixels/line (picture width)
 - # of lines (picture height)
 - linearization: by line or column
 - pixel semantics: grayscale, color definition, index into colormap
 - (optional) colormap with specific number of entries, length of entries (e.g., 24 bit)
 - (optional) definition of color space (RGB, IHS, ...)
 - and more
- Descriptive data
 - text, keywords, knowledge representation describing content
 - recognized lines, areas appearing in the image
 - resulting 2-D objects such as circles, ellipses, polygons, ...

Image Operations

- Input/Output
 - file (SUN Rasterfile, GIF, TIFF, JPEG,)
 - main memory data structure (matrix)
 - device (scanner, camera / display, printer)
- Modification:
 - set individual pixel(s)
 - change color map
 - for artificial colors (tomography)
 - for analysis (contrast)
 - bitmap operations
 - overlay with other images
- Analysis, aggregation, extraction
 - increasing contrast
 - line recognition
 - resize, zoom, crop (window)
 - convert to grayscale

Image Search

- Significant research efforts over the last 15 years
 - a number of techniques and systems available
- Alternative approaches:
 - attribute-based
 - text description (annotation)
 - content-based image retrieval (CBIR)
 - elementary features (low-level)
 - semantic features, object recognition (high-level)

Text-based Image Retrieval

- Image description
 - unrestricted (free) text
- Queries
 - keyword or free text
 - with/without boolean operators
- Search
 - uses conventional IR techniques (see chapter 3)
- Differences to text search
 - manual annotation required (unless captions are already present in the image file)
 - needs to be efficient, complete, consistent
 - requires domain knowledge, thesaurus
- Advantages
 - abstractions, concepts can be used ("smile", "happiness")
 - hard to achieve with other techniques
- Disadvantages
 - visual appearance (e.g., texture) is hard to describe textually
 - query-by-example-image is not supported

Color-based Search

- Main idea
 - given an image, find images with a similar color appearance
 - three primary colors, or color channels (e.g., RGB)
 - each channel is divided into m discrete intervals
 - results in m^3 different color combinations (bins)
- Color histogram
 - $H(M)$ for image M
 - vector $(h_1, h_2, \dots, h_y, \dots, h_n)$ with
 - n = number of bins
 - h_j = number of pixels for image M that fall into bin j
- Query is mapped to a histogram as well
 - compute from example image, or "guess" from description
- Search
 - compute distance between histograms of stored images and query image
 - return images below a given threshold, or first k (sorted ascending by distance)



© Prof.Dr.-Ing. Stefan DeBloch

Digital Libraries and Content Management

Color Histograms

- Distance metrics
 - numerous proposals
 - simplest: L-1
- $$d(H_1, H_2) = \sum_{l=1}^n |h_{1,l} - h_{2,l}|$$
- Limitations of basic approach
 - similarity of colors (by human perception) and bins is ignored:
 - assumption: all images have N pixels
 - if not, then perform a normalization step
 - maximum distance of two images: $2N$
 - there is no bin in which both images have pixels
 - what if images do not have exactly the same colors, but colors perceived to be similar?
 - queries only give an approximation of the desired colors
 - colors may be slightly different due to noise, lighting conditions, etc.
 - problem is made worse by the exact definition of bins



© Prof.Dr.-Ing. Stefan DeBloch

Digital Libraries and Content Management

Accounting For Color Similarity

- Account for color perception in distance computation
 - method by Niblack et. al. (used in QBIC)
 - query histogram X, database image histogram Y
 - normalized: $0 \leq x_i, y_i \leq 1; \sum x_i = \sum y_i = 1$
 - bin-by-bin distance histogram $Z = X - Y$
 - $-1 \leq z_i \leq 1; \sum z_i = 0$
 - distance between X and Y

$$d_{hist}^2(X, Y) = (X - Y)^t A (X - Y) = \sum_i \sum_j a_{ij} (x_i - y_i)(x_j - y_j)$$

- A is the symmetric color similarity matrix with
 - $a(i, j) = 1 - d(c_i, c_j) / d_{max}$
 - c_i and c_j are the colors of bin i and j in the histogram,
 - $d(c_i, c_j)$ is the distance of colors,
 - d_{max} is the maximum distance of all colors
- for similar colors, $d(c_i, c_j)$ is small, so $a(i, j)$ is close to 1
 - presence of similar colors will result in reducing the distance

Example: bins for red, orange, blue

$X = [1.0, 0.0, 0.0]$
 $Y = [0.0, 1.0, 0.0]$
 $W = [0.0, 0.0, 1.0]$

$$A_{red,orange,blue} = \begin{bmatrix} 1.0 & 0.9 & 0.0 \\ 0.9 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$$

$$d_{hist}^2(X, Y) = 0.2$$

$$d_{hist}^2(X, W) = 2.0$$



© Prof.Dr.-Ing. Stefan DeBloch

Digital Libraries and Content Management

Perceptually Weighted Histograms

- Determine representative colors of a color space
- For each color, create a color bin
 - equal distribution across color space
- Histogram computation:
 - for each pixel, determine the ten most similar representative colors
 - determine the color distance
 - assign a weight proportional to the inverse distance and add to the corresponding color bin
 - pixel counts (fractionally) in multiple bins
- Most important difference to previous approaches:
 - histogram computation already considers color similarity
- Apparently results in performance advantage



© Prof.Dr.-Ing. Stefan DeBloch

Digital Libraries and Content Management

Image Regions

- Consider position of colors within the image's spatial dimensions
- Partition the image into regions and compute histograms for each region
 - raster-based (e.g., 4 x 4 regions of fixed size)
 - separation of foreground and background
 - background often dominate regarding image area
 - foreground usually more important for search
 - separate histograms should be computed
 - segmentation does not need to be precise
 - minimum bounding rectangle, can be determined automatically based on horizontal and vertical pixel variations, or can be manually assigned
- Search may exploit the presence of regions (or foreground/background) in different ways
 - global search, considering region-specific similarity measures
 - focused search, referring to a specific region (e.g., foreground only)
 - weighting regions differently
- Experiments show significant improvements



© Prof.Dr.-Ing. Stefan DeBloch

Digital Libraries and Content Management

Color Histograms – Additional Aspects

- Color distribution
 - so far, we partitioned the color space into bins in an equi-distant manner, without considering which colors are actually used
 - alternative approaches: use finer-grained partitioning for color intervals that receive more pixels
- Color variability
 - which color spaces are best suited for distance computation?
 - when should pixels in different pictures be considered "equivalent"?
 - color spaces
 - **RGB** is device-dependent, requires White and three primary colors for reference, is very sensitive regarding lighting conditions, surface reflections, etc.
 - **Munsell** or Lab spaces achieve a better perceptual uniformity
 - **HSV** (hue, saturation, value) – hue is invariant regarding illumination



© Prof.Dr.-Ing. Stefan DeBloch

Digital Libraries and Content Management

Shape-based Search

- Requires segmentation
 - semi-automatic methods exist
- Shape representation and similarity
 - every shape should have a *unique* representation, invariant to translation, rotation, scaling
 - similar shapes should have similar representations to allow for search based on distance measures
- Query
 - query-by-example, sketch
- Terminology
 - **major axis**: straight line between the two most distant points on the boundary of the shape
 - **minor axis**: straight line perpendicular to the major axis such that a minimum bounding rectangle can be constructed which is parallel to the major and minor axis and width/height correspond to the lengths of the major/minor axis
 - **base rectangle**: the minimum bounding rectangle described above
 - **excentricity**: ratio of major to minor axis (≥ 1)



© Prof.Dr.-Ing. Stefan Deßloch

Digital Libraries and Content Management

Shape Representation

- Simple shape representation
 - based on the four measures defined above
 - may be used for representation, search
 - are only a "weak" characterization of the shape
 - usually combined with additional characteristics
 - example (QBIC system by IBM): area, circularity, major axis orientation, invariant moments
- Region-based shape representation
 - closer to perceived shape by including regions
 - promises better retrieval quality



© Prof.Dr.-Ing. Stefan Deßloch

Digital Libraries and Content Management

Shape Regions - Idea

- Raster definition
 - quadratic cells of equal size
 - raster is just large enough to cover the shape
- Cells
 - may be completely or partially "filled" by the form, or may be "empty"
 - are labeled with "1" when filled at least 15%, otherwise 0
 - ordered left to right, top to bottom within the raster: bitstring for the shape
 - compact, easy to compute, invariant to translations
- Raster size
 - smaller cells result in more accurate shape representation, but require more resources
 - compromise: cell size from 10x10 up to 20x20 pixels

Region Normalization

- Achieves uniform shape orientation and scale
- Rotation
 - major axis parallel to x-axis
 - still two possible positions
 - results in two bitstrings
 - not for stored objects (would double storage space requirements)
 - but for queries
- Scale
 - proportionally scale the shape to a fixed length of the major axis (e.g., 192 pixels)
- Results in unique shape representation
 - provided that the major axis is unique
- Bitstring length
 - raster is just large enough for normalized shape: #cells along x-axis is always the same
 - example: 8 cells with cell size 24x24, major axis length 192 pixels
 - number of cells along the y-axis may vary (difference in excentricity), but is always smaller than along the x-axis
 - has to be accounted for by the similarity measure

Similarity Measure

- Idea: distance = number of unequal cells
- Excentricity needs to be accounted for
 - equal raster size: bitwise comparison
 - significant difference in #cells along y-axis:
 - shapes are regarded as different
 - threshold depends on cell size and application
 - typical example: 3
 - small difference for y-axis: fill with zeroes, do bitwise comparison
- Reflections (horizontal/vertical)
 - should be considered similar
 - two additional bitstrings for each query
- Multiple possible major axis
 - need to store all possible major axes, determine all possible major axes for query
 - pairwise distance comparison, keep minimum

Indexing and Retrieval

- For every shape in the database
 - determine major/minor axis, excentricity
 - normalize regarding rotation and scale
 - apply raster, determine bitstring
 - store bitstring, length of minor axis
- For every query
 - determine bitstring in the same manner
 - but include bitstrings for 180 rotation, reflections
 - search for bitstrings in the database with (almost) equal minor axis length (similar excentricity)
 - compute distance for all matches
 - return sorted ascending by distance

Texture-based Search

- Texture is hard to characterize
- Approach by Tamura et.al: six different features
 - coarseness
 - most important feature
 - size of distinguishable picture elements
 - contrast
 - grayscale levels, edge sharpness, periodicity of repeated elements
 - directionality
 - shape and position of elements
 - line-likeness
 - shape of elements
 - regularity
 - variation in placement of elements
 - roughness

Systems

- Have to support combinations of different kind of search
 - especially for elementary features *and* text
- QBIC
 - query image (color, shape, texture) plus keywords
 - partly included in DB2
 - www.qbic.almaden.ibm.com
- Virage
 - Features: color, shape, texture
 - www.virage.com
- WebSEEK
 - www.ctr.columbia.edu/webseek

Summary

- Vector graphics
 - media object – raw/registration/description data, operations
 - search
- Raster image
 - media object – raw/registration/description data, operations
 - text-based image retrieval
- Content-based image retrieval (CBIR)
 - mostly based on elementary (low-level) features:
 - color-based
 - histograms, color similarity matrix, perceptually weighted histograms, regions
 - shape-based
 - representation – raster, normalization, similarity
 - texture-based