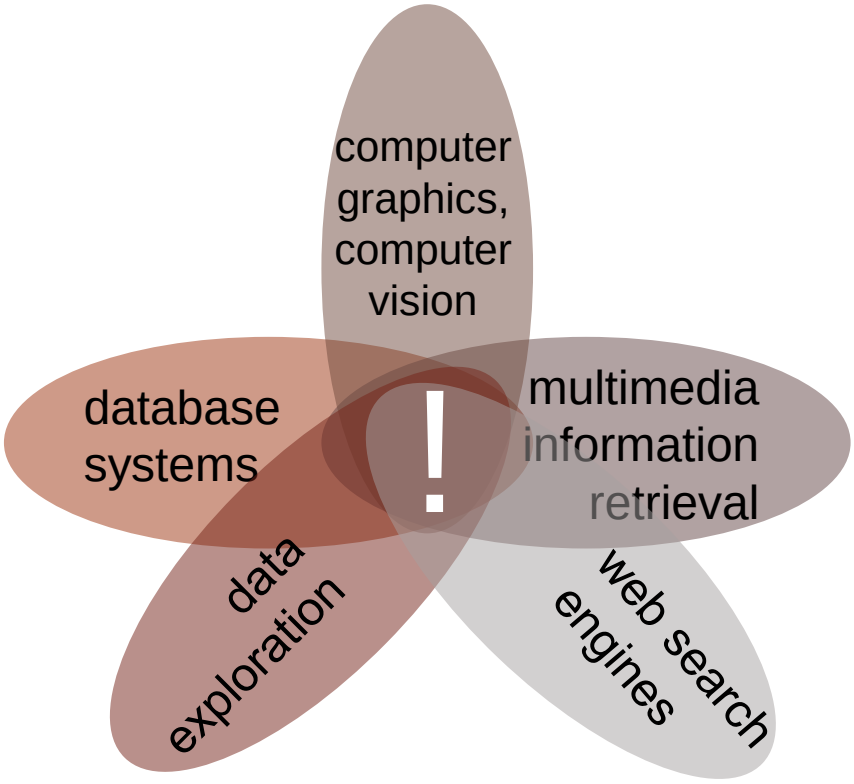
Introduction to Similarity Search in Multimedia Databases

Tomáš Skopal
Charles University in Prague
Faculty of Mathematics and Phycics
SIRET research group
http://siret.ms.mff.cuni.cz

March 23rd 2011, DCC, Universidad de Chile

The context of the research topic

Similarity Search in Multimedia Databases



Talk outline

- motivation
- context vs. content-based search
- similarity search
- examples of applications in various domains
  - images
  - shapes
  - audio
- metric indexing for fast search

Multimedia content on the Web

- more than 95% (99.9%?) of web space is considered to store multimedia content
  - 100 billions of photographs are expected to be taken every year

Multimedia content on the Web

- more than 95% (99.9%?) of web space is considered to store multimedia content
  - 100 billions of photographs are expected to be taken every year
- factors
  - high-speed internet, increasing computational power
  - cheap digital devices
    - cameras, camcoders, all-in-one devices (PDA, smartphones)
    - everyone is producer of multimedia
  - human activities move to internet in a large extent
    - social networks (FaceBook, Twitter, MySpace)
    - industry (e-banking, e-commerce, e-services)

What is multimedia content?

- **multi–media** = more than one type of **digital media**
- traditional interpretation
  - **image, audio**, **video** content

What is multimedia content?

- **multi–media** = more than one type of **digital media**
- traditional interpretation
  - **image, audio**, **video** content

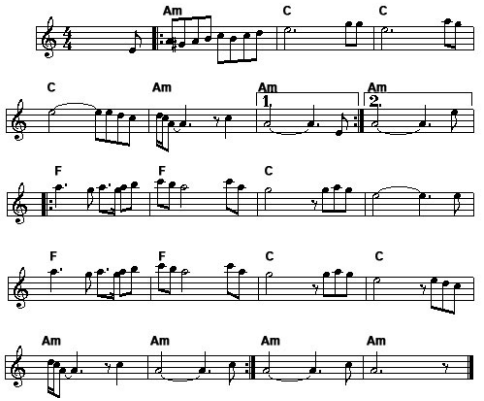
- extended interpretation of multimedia
  - any media type with **unstructured content**
  - mostly **“digitized nature”**
    - sensory data
  - also “**human-processed”** data
    - text, XML, geometry & illustration, application (flash), etc.

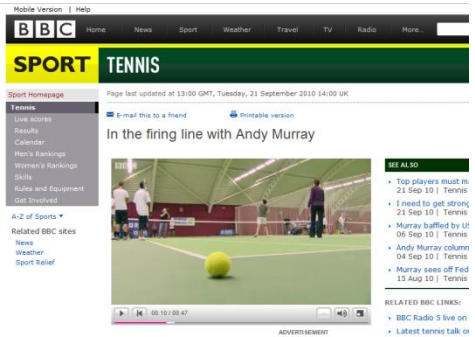
Producers of multimedia data

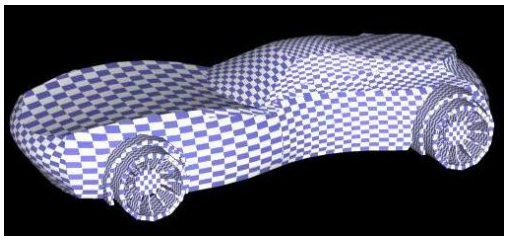
- individuals
  - recording whatever using phones, PDAs, cameras

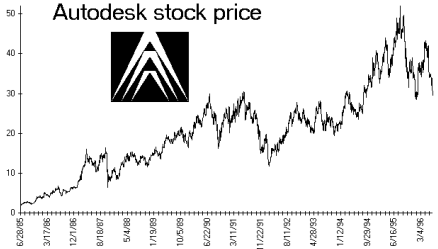
Producers of multimedia data

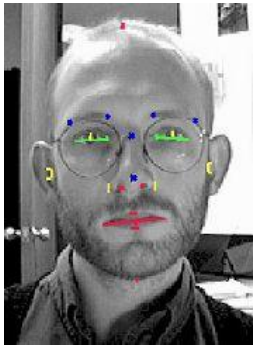
- industry
  - entertainment
  - financial
  - internet news media
  - manufacturing
  - biometrics

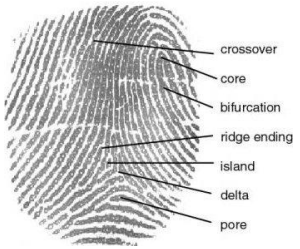


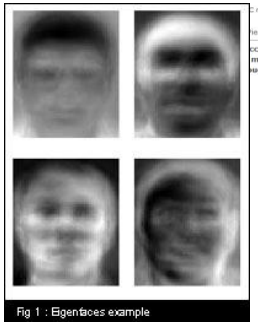


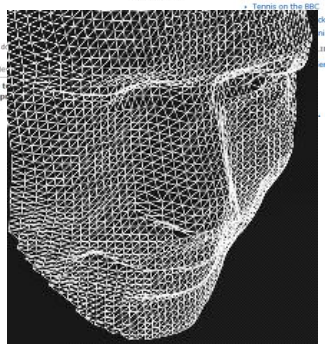






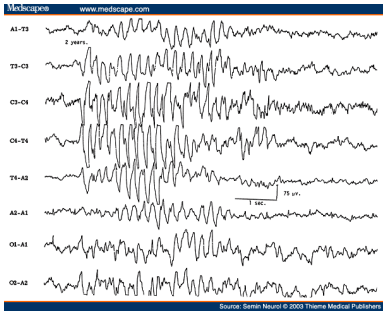


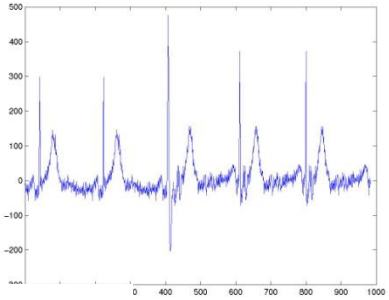


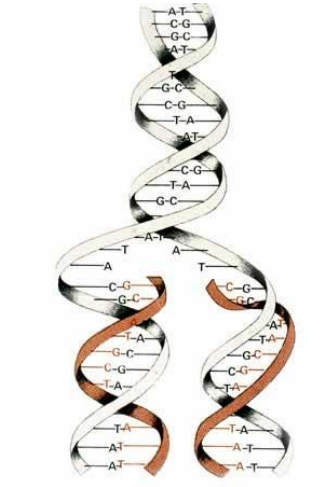


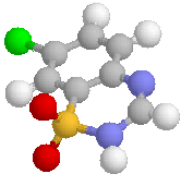
Producers of multimedia data

- research
  - medicine
  - material engineering
  - biochemistry









Context and content

- context – description/annotation/neighborhood of content
  - high-level semantics (e.g., image of a ship)
  - keywords/tags, full-text, URL, categorization
    - mostly human-annotated
    - some can be automated
      - e.g., Google images and the embedding web page
  - GPS, context of social network (links, groups)

Context and content

- context – description/annotation/neighborhood of content
  - high-level semantics (e.g., image of a ship)
  - keywords/tags, full-text, URL, categorization
    - mostly human-annotated
    - some can be automated
      - e.g., Google images and the embedding web page
  - GPS, context of social network (links, groups)

- content – the actual data content
  - raw content (e.g., color pixels, audio wave)

Content and context

Example:
a photograph hosted at Flickr
found by keyword query “vacation”

raw content

manual annotation

automated annotation

Text-based search pros and cons

- **pros**
  - text/keyword annotation allows to implement semantically high-level queries
    - high-level keyword query vs. high-level keyword annotation
  - easy to implement a fulltext index
    - the same as web search engine, i.e., Boolean or vector model
  - context-based search will be a powerful concept in the future due to the omnipresent social networks

Text-based search pros and cons

- **pros**
  - text/keyword annotation allows to implement semantically high-level queries
    - high-level keyword query vs. high-level keyword annotation
  - easy to implement a fulltext index
    - the same as web search engine, i.e., Boolean or vector model
  - context-based search will be a powerful concept in the future due to the omnipresent social networks
- **cons**
  - requires human labor
    - Ask yourselves,

      how many of your photos from vacations did you keyworded? ☺
  - highly subjective (two persons keyword differently)
  - incomplete (some possibly relevant content is not keyworded)

Structure and semantics of data

- relational databases
  - strong structure – typed attributes shared by all rows
  - strong semantics – user knows meaning of attributes

Structure and semantics of data

- relational databases
  - strong structure – typed attributes shared by all rows
  - strong semantics – user knows meaning of attributes
- full-text databases
  - loose structure – plain sequence of words
  - strong semantics – query and text share the model

Structure and semantics of data

- relational databases
  - strong structure – typed attributes shared by all rows
  - strong semantics – user knows meaning of attributes
- full-text databases
  - loose structure – plain sequence of words
  - strong semantics – query and text share the model
- **multimedia databases**
  - **loose structure**
    – digitized signal (e.g., pixels)
  - **loose semantics**
    – how to query based on pixels?

Content-based similarity search

- if no context/annotation → content-based search
  - it is suitable also in case of existing annotation, because of the cons of text-based search

Content-based similarity search

- if no context/annotation → content-based search
  - it is suitable also in case of existing annotation, because of the cons of text-based search

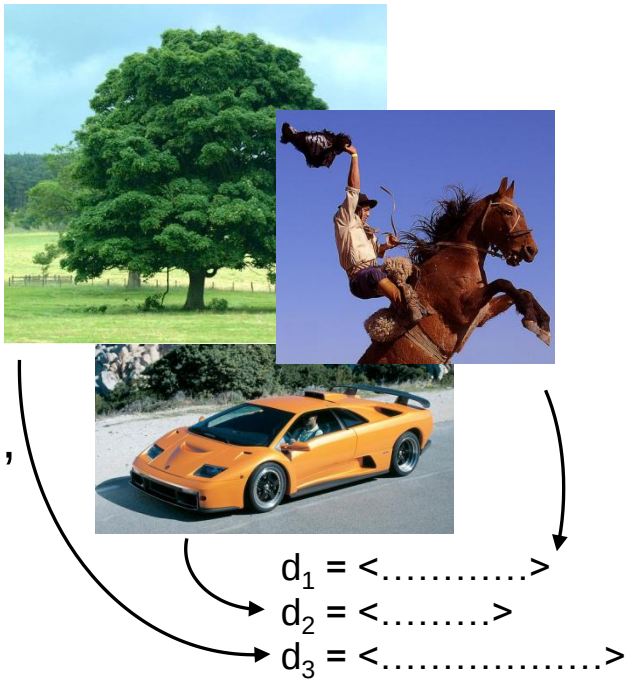
- multimedia content has loose structure + semantics,
  - these have to be **revealed** (or **interpreted**) by establishing a content-based similarity search model
  - condensing the raw content into a higher-level information
    - analogy to text search, as to the extreme case, i.e., a few-byte information (e.g., “sunset”) vs. 5MB of mess (a lot of pixels)

Content-based similarity search

- similarity search model

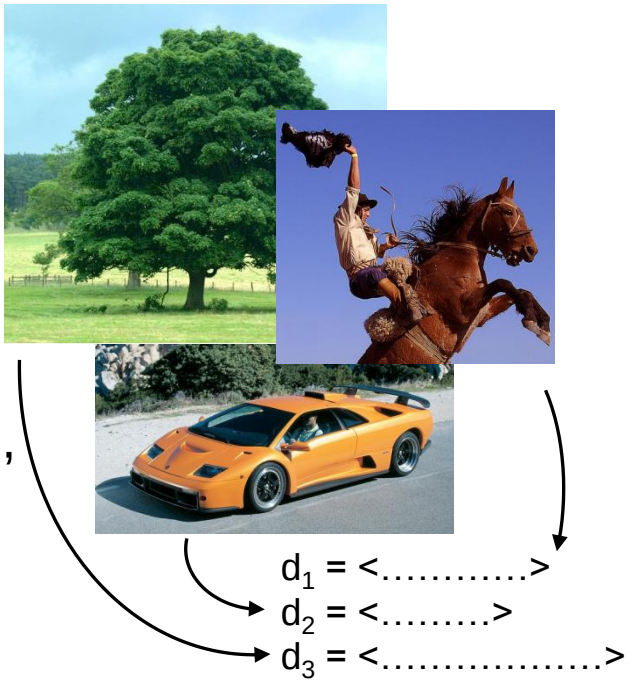
Content-based similarity search

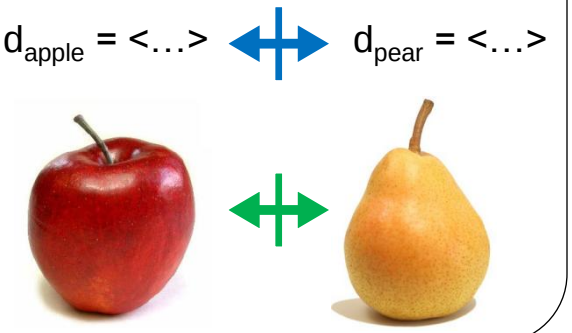
- similarity search model
  - **feature extraction** procedure
    - definition and production of concise **structured descriptors**
    - we obtain a structured database, however, unlike relational databases, the descriptor structure is **hidden** to the end user



Content-based similarity search

- similarity search model
  - **feature extraction** procedure
    - definition and production of concise **structured descriptors**
    - we obtain a structured database, however, unlike relational databases, the descriptor structure is **hidden** to the end user
  - **similarity** function
    - suitable function for **comparison** of the descriptors
    - should mimic the **semantic** similarity of the original multimedia objects





Content-based similarity search

- querying
  - the hidden structure of descriptors
    cannot be used in a query language (as in SQL)

Content-based similarity search

- querying
  - the hidden structure of descriptors
    cannot be used in a query language (as in SQL)
  - for querying we can only utilize the model tools, i.e.,
    - feature extraction procedure
    - similarity measure

Content-based similarity search

- querying
  - the hidden structure of descriptors
    cannot be used in a query language (as in SQL)
  - for querying we can only utilize the model tools, i.e.,
    - feature extraction procedure
    - similarity measure
  - **query-by-example concept**
    - a multimedia object is obtained, its descriptor **q** is extracted
    - using the similarity function,
      q is compared to all descriptors in the database
    - which results in ordering of the most similar ones to **q**
      - **range query** – returned objects above a similarity threshold
      - **k nearest neighbors** (kNN) query – the **k** most similar ones

Content-based similarity search

Content-based similarity search

query object

Content-based similarity search

query object

ordering of database descriptors
according to their similarity to query descriptor

Content-based similarity search

query object

0.9

ordering of database descriptors
according to their similarity to query descriptor

Content-based similarity search

query object

0.9

0.85

ordering of database descriptors
according to their similarity to query descriptor

Content-based similarity search

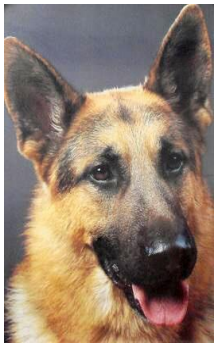
query object

ordering of database descriptors
according to their similarity to query descriptor

0.9

0.85

0.7



Content-based similarity search

query object

ordering of database descriptors
according to their similarity to query descriptor

0.9

0.85

0.7

0.6

Content-based similarity search

query object

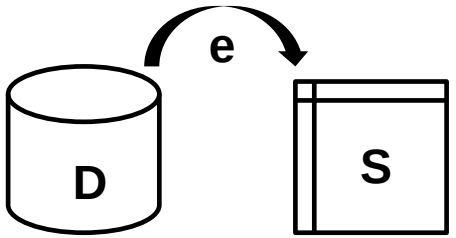
ordering of database descriptors
according to their similarity to query descriptor

0.9 **0.85** **0.7** **0.6** **0.4**

range query (q, 0.5) or 3NN query

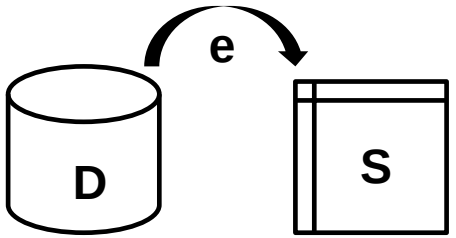
Content-based similarity search

- in summary, we have the following formalized model
  - feature extraction procedure **e: X → U**
    - transforming a multimedia object from database universe **X** into a descriptor in descriptor universe **U**
      - the original database **D ⊂ X**,
      - the descriptor database **S ⊂ U**



Content-based similarity search

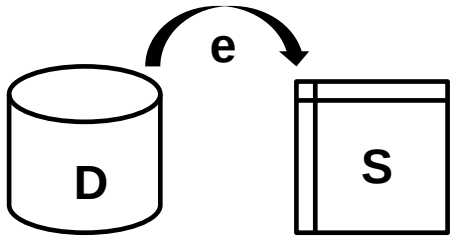
- in summary, we have the following formalized model
  - feature extraction procedure **e: X → U**
    - transforming a multimedia object from database universe **X** into a descriptor in descriptor universe **U**
      - the original database **D ⊂ X**,
      - the descriptor database **S ⊂ U**



  - similarity function **δ: U x U → R**
    - better a dissimilarity (distance), i.e., **close means similar**

Content-based similarity search

- in summary, we have the following formalized model
  - feature extraction procedure **e: X → U**
    - transforming a multimedia object from database universe **X** into a descriptor in descriptor universe **U**
      - the original database **D** ⊂ **X**,
      - the descriptor database **S** ⊂ **U**



  - similarity function **δ: U x U → R**
    - better a dissimilarity (distance), i.e., **close means similar**
  - query-by-example types
    - range query $(\mathbf{q}, \mathbf{r}) = \{x \in S \mid \delta(\mathbf{q}, x) \leq \mathbf{r}\}$
    - **k** nearest neighbor query (**k**NN)

$$(\mathbf{q}, \mathbf{k}) = \{C \mid C \subseteq S, |C| = \mathbf{k}, \forall x \in C, y \in S\text{-}C \Rightarrow \delta(\mathbf{q},x) \leq \delta(\mathbf{q},y)\}$$

Feature extraction

- how to obtain a descriptor of a multimedia object?

Feature extraction

- how to obtain a descriptor of a multimedia object?
  1) using **low-level features** as the basic building blocks
     - competence of the research areas outside the data engineering (e.g., computer vision, cognitive psychology)
     - often providing just a **local information**

Feature extraction

- how to obtain a descriptor of a multimedia object?
  1) using **low-level features** as the basic building blocks
     - competence of the research areas outside the data engineering (e.g., computer vision, cognitive psychology)
     - often providing just a **local information**
  2) **combining low-level features** to obtain an **expressive** yet **concise** descriptor
     - competence of data exploration, multimedia retrieval

Feature extraction

- how to obtain a descriptor of a multimedia object?
  1) using **low-level features** as the basic building blocks
     - competence of the research areas outside the data engineering (e.g., computer vision, cognitive psychology)
     - often providing just a **local information**
  2) **combining low-level features** to obtain an **expressive** yet **concise** descriptor
     - competence of data exploration, multimedia retrieval
- MPEG7, standard ISO/IEC 15938 (launched in 1998)
  - definitions of visual, audio and motion descriptors
  - selection of some successful content-based descriptors

Feature extraction

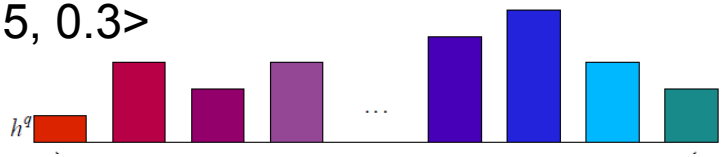
- what could be the descriptor structure?

Feature extraction

- what could be the descriptor structure?
  - **vector** (for feature histograms)

    e.g.,

    x = <0.1, 0.4, 0.3, 0.4, …, 0.6, 0.7, 0.5, 0.3>

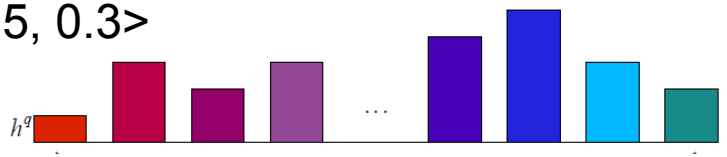


Feature extraction

- what could be the descriptor structure?
  - **vector** (for feature histograms)

    e.g.,

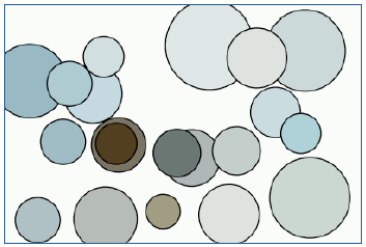
    x = <0.1, 0.4, 0.3, 0.4, …, 0.6, 0.7, 0.5, 0.3>



  - **set** (for feature signatures)

    e.g.,

    x = (<c_1, w_1>, <c_2, w_2>, …, <c_k, w_k>)

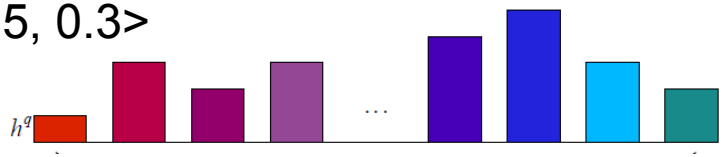


Feature extraction

- what could be the descriptor structure?
  - **vector** (for feature histograms)

    e.g.,

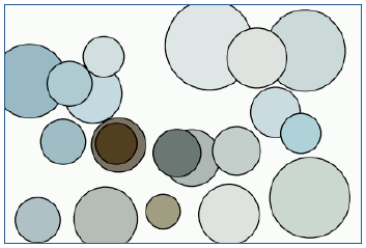
    x = <0.1, 0.4, 0.3, 0.4, …, 0.6, 0.7, 0.5, 0.3>



  - **set** (for feature signatures)

    e.g.,

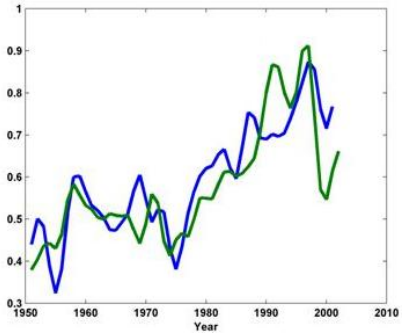
    x = (<c_1, w_1>, <c_2, w_2>, …, <c_k, w_k>)



  - **ordered set** (for time series, sequences, strings)

    e.g.,

    x = <t_1, t_2, …, t_k>



Modeling similarity

- how to obtain a **suitable similarity function** for comparing descriptors?

Modeling similarity

- how to obtain a **suitable similarity function** for comparing descriptors?
  - depends on the descriptor structure, and the semantics of low-level features

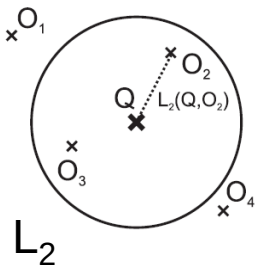
Modeling similarity

- how to obtain a **suitable similarity function** for comparing descriptors?
  - depends on the descriptor structure, and the semantics of low-level features
  - based on descriptor structure, we can use, among others

Modeling similarity

- how to obtain a **suitable similarity function** for comparing descriptors?
  - depends on the descriptor structure, and the semantics of low-level features
  - based on descriptor structure, we can use, among others
    - **vectorial distances** (for general vectors, histograms)
      - Minkowski L_n distances (independent dimensions)

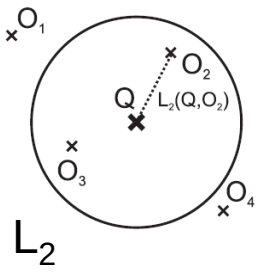
$$L_p(v_1, v_2) = (\sum_{i=1}^{D} |v_1[i] - v_2[i]|^p)^{\frac{1}{p}} \qquad (p \geq 1)$$



Modeling similarity

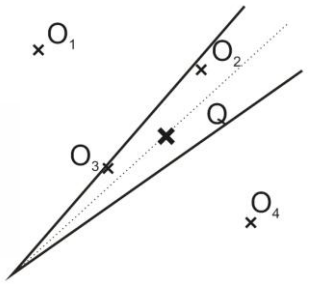
- how to obtain a **suitable similarity function** for comparing descriptors?
  - depends on the descriptor structure, and the semantics of low-level features
  - based on descriptor structure, we can use, among others
    - **vectorial distances** (for general vectors, histograms)
      - Minkowski L_n distances (independent dimensions)

$$L_p(v_1, v_2) = (\sum_{i=1}^{D} |v_1[i] - v_2[i]|^p)^{\frac{1}{p}} \qquad (p \geq 1)$$



      - cosine distance (angle matters)

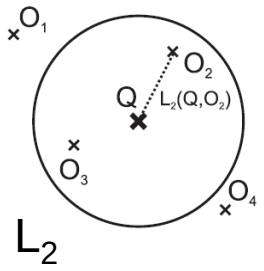
$$\mathrm{SIM}_{cos}(v_1, v_2) = \frac{\sum_{i=1}^{D} v_1[i] v_2[i]}{\sqrt{\sum_{i=1}^{D} v_1[i]^2 \cdot \sum_{i=1}^{D} v_2[i]^2}}$$



Modeling similarity

- how to obtain a **suitable similarity function** for comparing descriptors?
  - depends on the descriptor structure, and the semantics of low-level features
  - based on descriptor structure, we can use, among others
    - **vectorial distances** (for general vectors, histograms)
      - Minkowski L_n distances (independent dimensions)

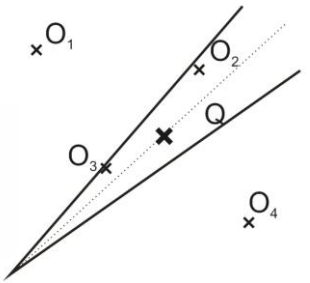
$$L_p(v_1, v_2) = \left(\sum_{i=1}^{D} |v_1[i] - v_2[i]|^p\right)^{\frac{1}{p}} \qquad (p \geq 1)$$



L_2

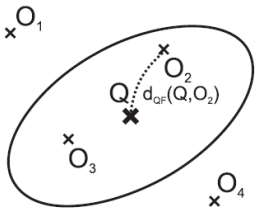
      - cosine distance (angle matters)

$$\mathrm{SIM}_{cos}(v_1, v_2) = \frac{\sum_{i=1}^{D} v_1[i] v_2[i]}{\sqrt{\sum_{i=1}^{D} v_1[i]^2 \cdot \sum_{i=1}^{D} v_2[i]^2}}$$



      - quadratic form distance (histograms)

$$d_{QF}(v_1, v_2) = \sqrt{(v_1 - v_2) M (v_1 - v_2)^T}$$



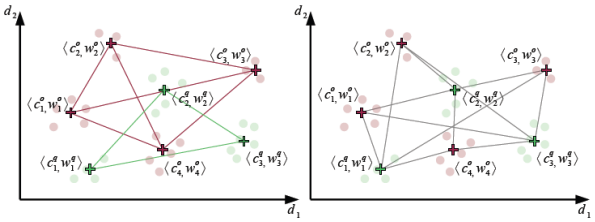
Modeling similarity

- **adaptive distances** (signatures, gen. sets)

Modeling similarity

- **adaptive distances** (signatures, gen. sets)
  - signature quadratic form distance

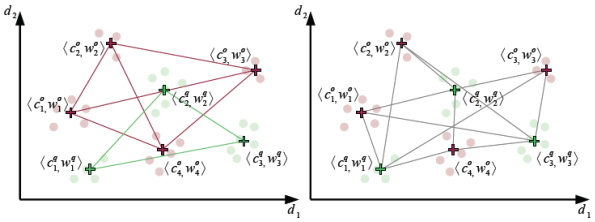
$$\mathrm{SQFD}_{f_s}(S^q, S^p) = \sqrt{(w_q \mid -w_p) \cdot A_{f_s} \cdot (w_q \mid -w_p)^T}$$



Modeling similarity

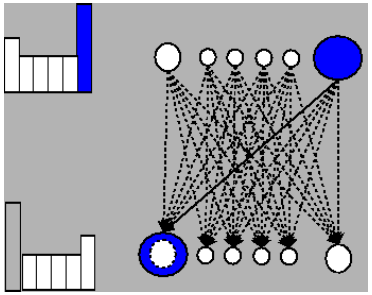
- **adaptive distances** (signatures, gen. sets)
  - signature quadratic form distance

$$\mathrm{SQFD}_{f_s}(S^q, S^p) = \sqrt{(w_q \mid -w_p) \cdot A_{f_s} \cdot (w_q \mid -w_p)^T}$$



  - earth mover's distance

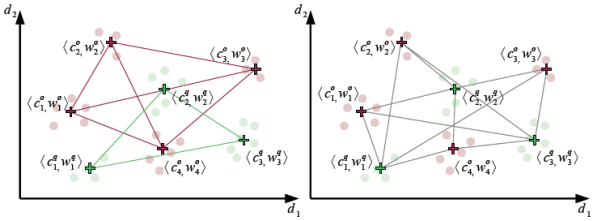
$$\begin{aligned} \delta_{EMD}(x,y) &= \min_f \left\{ \textstyle\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} f_{ij} \right\} \\ \text{subject to } f_{ij} &\geq 0 \\ \textstyle\sum_{i=1}^{m} f_{ij} &= y_j \quad \forall j = 1, \ldots, n \\ \textstyle\sum_{j=1}^{n} f_{ij} &= x_i \quad \forall i = 1, \ldots, m \end{aligned}$$



Modeling similarity

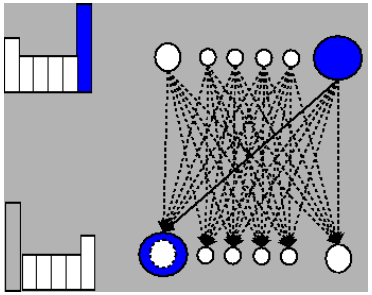
- **adaptive distances** (signatures, gen. sets)
  - signature quadratic form distance

$$\mathrm{SQFD}_{f_s}(S^q, S^p) = \sqrt{(w_q \mid -w_p) \cdot A_{f_s} \cdot (w_q \mid -w_p)^T}$$



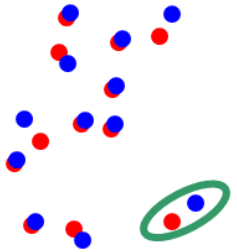
  - earth mover's distance

$$\begin{aligned} \delta_{EMD}(x,y) &= \min_f \left\{ \textstyle\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} f_{ij} \right\} \\ \text{subject to } f_{ij} &\geq 0 \\ \textstyle\sum_{i=1}^{m} f_{ij} &= y_j \quad \forall j = 1, \ldots, n \\ \textstyle\sum_{j=1}^{n} f_{ij} &= x_i \quad \forall i = 1, \ldots, m \end{aligned}$$



  - Hausdorff distance

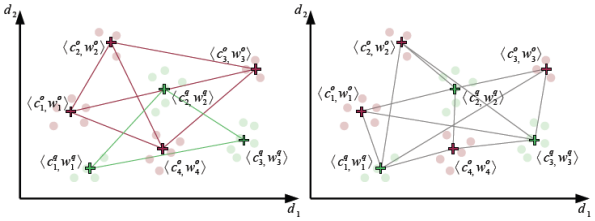
$$d_H(A,B) = \max(h(A,B),\ h(B,A))$$
$$h(A,B) = \max_i \min_j \delta(A_i, B_j)$$



Modeling similarity

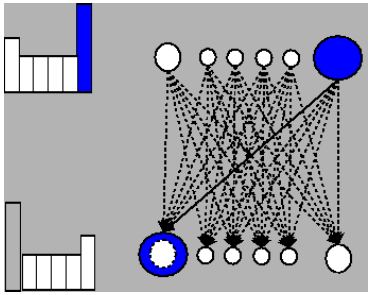
- **adaptive distances** (signatures, gen. sets)
  - signature quadratic form distance

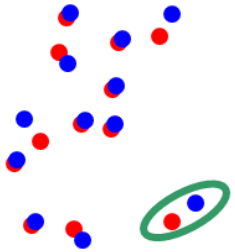
$$\mathrm{SQFD}_{f_s}(S^q, S^p) = \sqrt{(w_q \mid -w_p) \cdot A_{f_s} \cdot (w_q \mid -w_p)^T}$$



  - earth mover's distance

$$\begin{array}{rcll} \delta_{EMD}(x,y) & = & \min_f \left\{ \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} f_{ij} \right\} & \\ \text{subject to } f_{ij} & \geq & 0 & \\ \sum_{i=1}^{m} f_{ij} & = & y_j & \forall j = 1, \ldots, n \\ \sum_{j=1}^{n} f_{ij} & = & x_i & \forall i = 1, \ldots, m \end{array}$$





  - Hausdorff distance

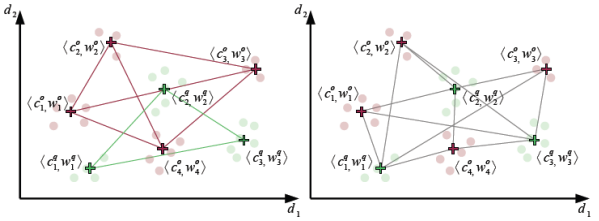
$$d_H(A,B) = \max(h(A,B),\ h(B,A))$$
$$h(A,B) = \max_i \min_j \delta(A_i, B_j)$$

- **sequence distances** (strings, time series)

Modeling similarity

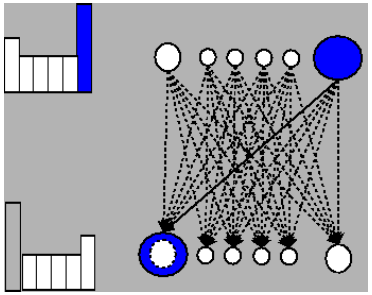
- **adaptive distances** (signatures, gen. sets)
  - signature quadratic form distance

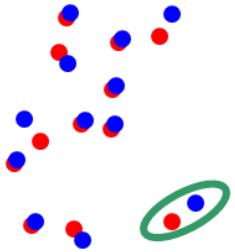
$$\mathrm{SQFD}_{f_s}(S^q, S^p) = \sqrt{(w_q \mid -w_p) \cdot A_{f_s} \cdot (w_q \mid -w_p)^T}$$



  - earth mover's distance

$$\begin{aligned} \delta_{EMD}(x,y) &= \min_f \left\{ \textstyle\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} f_{ij} \right\} \\ \text{subject to } f_{ij} &\geq 0 \\ \textstyle\sum_{i=1}^{m} f_{ij} &= y_j \quad \forall j = 1, \ldots, n \\ \textstyle\sum_{j=1}^{n} f_{ij} &= x_i \quad \forall i = 1, \ldots, m \end{aligned}$$



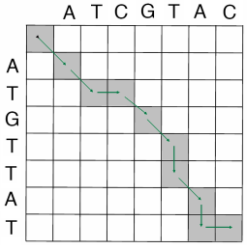


  - Hausdorff distance

$$d_H(A,B) = \max(h(A,B),\ h(B,A))$$
$$h(A,B) = \max_i \min_j \delta(A_i, B_j)$$

- **sequence distances** (strings, time series)
  - edit distance

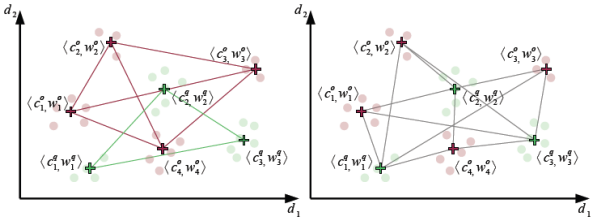
$$\delta_{edit}(x,y) = \min\{W(P)\}$$
$$W(P) = \sum_{k=1}^{m} \gamma(x_{i_{k-1}} + 1 \ldots i_k \rightarrow y_{j_{k-1}} + 1 \ldots j_k)$$



Modeling similarity

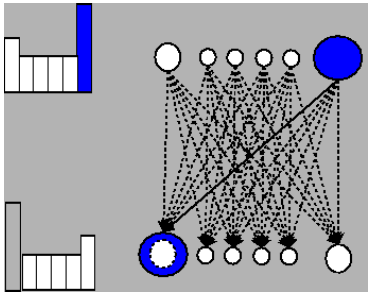
- **adaptive distances** (signatures, gen. sets)
  - signature quadratic form distance

$$\mathrm{SQFD}_{f_s}(S^q, S^p) = \sqrt{(w_q \mid -w_p) \cdot A_{f_s} \cdot (w_q \mid -w_p)^T}$$



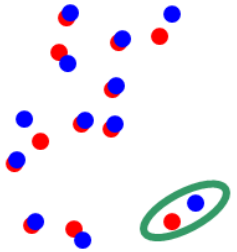
  - earth mover's distance

$$\begin{aligned} \delta_{EMD}(x,y) &= \min_f \left\{ \textstyle\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} f_{ij} \right\} \\ \text{subject to } f_{ij} &\geq 0 \\ \textstyle\sum_{i=1}^{m} f_{ij} &= y_j \quad \forall j = 1, \ldots, n \\ \textstyle\sum_{j=1}^{n} f_{ij} &= x_i \quad \forall i = 1, \ldots, m \end{aligned}$$



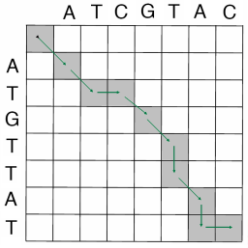
  - Hausdorff distance

$$d_H(A,B) = \max(h(A,B),\ h(B,A))$$
$$h(A,B) = \max_i \min_j \delta(A_i, B_j)$$



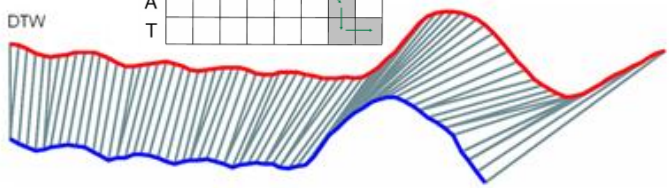
- **sequence distances** (strings, time series)
  - edit distance

$$\delta_{edit}(x,y) = \min\{W(P)\}$$
$$W(P) = \sum_{k=1}^{m} \gamma(x_{i_{k-1}} + 1 \ldots i_k \rightarrow y_{j_{k-1}} + 1 \ldots j_k)$$



  - dynamic time warping distance

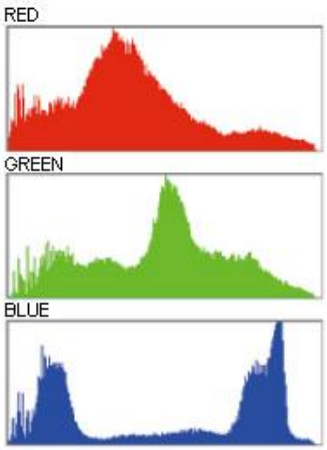
$$\delta_{DTW}(x,y) = \min_W \left\{ \sqrt{\sum_{k=1}^{t} w_k} \right\}$$

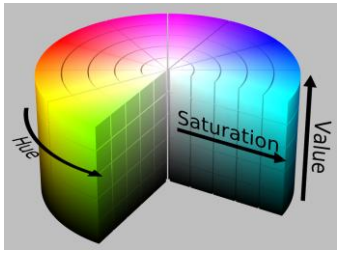


Applications

- image retrieval using **global features**, e.g.,

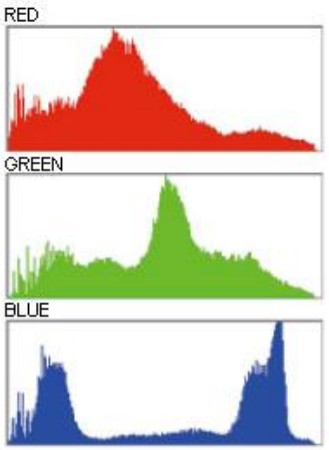
Applications

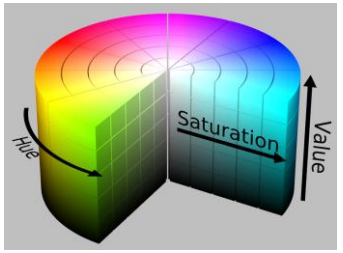




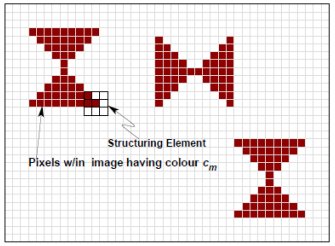
- image retrieval using **global features**, e.g.,
  - **Scalable color** (MPEG7)
    - histogram extracted and converted into HSV color space, quantized to 256 bins, passed through 1D Haar transform, resulting in 16-128 dimensional histogram (the descriptor)
    - L_1 distance used

Applications

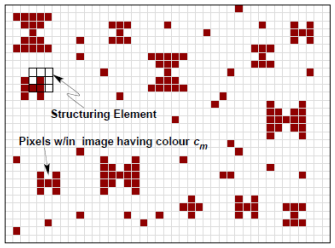




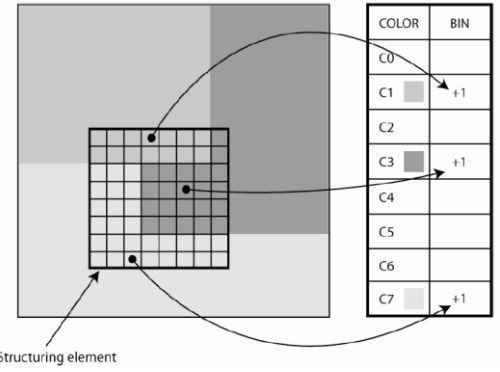
- image retrieval using **global features**, e.g.,
  - **Scalable color** (MPEG7)
    - histogram extracted and converted into HSV color space, quantized to 256 bins, passed through 1D Haar transform, resulting in 16-128 dimensional histogram (the descriptor)
    - L_1 distance used
  - **Color structure** (MPEG7)
    - specific color histogram that includes local information
    - moving structuring element (grid), adding its content to histogram (the descriptor) after each movement
    - L_1 distance used



high color structure



low color structure

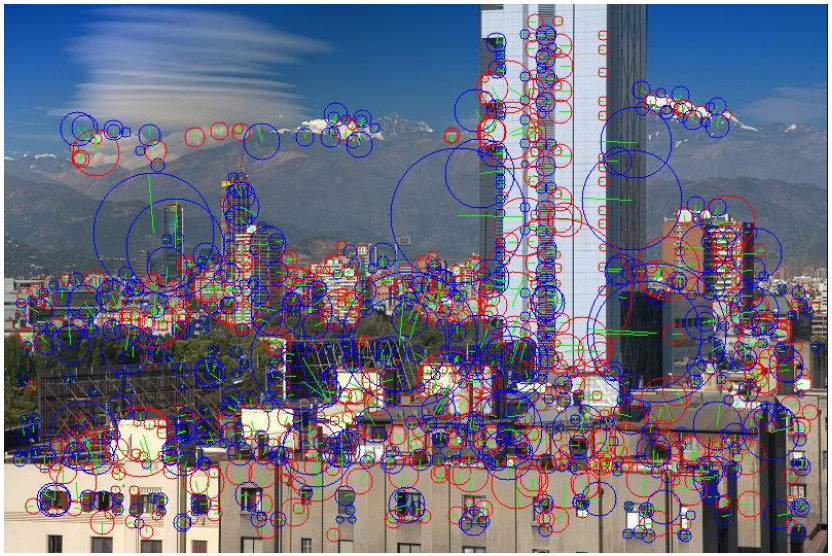


Applications

- image retrieval using **local features**, e.g.,

Applications

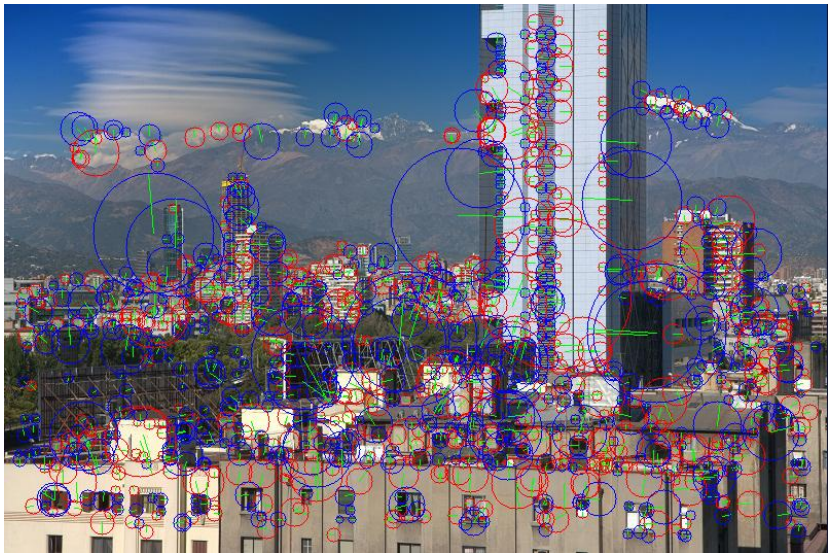
view at Santiago, 1174 interest points (SURF)



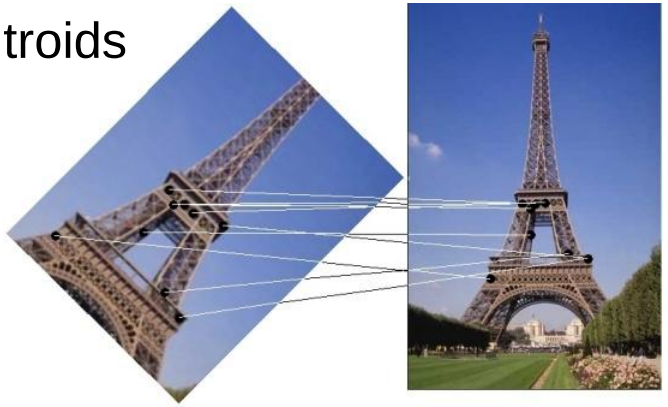
- image retrieval using **local features**, e.g.,
  - SIFT or SURF
    - interest points/blobs
    - local SIFT feature vector = 128D vector consisting of gradient orientation histograms around an interest point
    - image descriptor – signature
      - SIFT feature vectors are clustered
      - signature consists of weighted centroids

Applications

view at Santiago, 1174 interest points (SURF)



- image retrieval using **local features**, e.g.,
  - SIFT or SURF
    - interest points/blobs
    - local SIFT feature vector = 128D vector consisting of gradient orientation histograms around an interest point
    - image descriptor – signature
      - SIFT feature vectors are clustered
      - signature consists of weighted centroids
    - set distance
      - (signature) quadratic form dist.
      - Hausdorff distance
      - using an L_p ground distance for the SIFT features

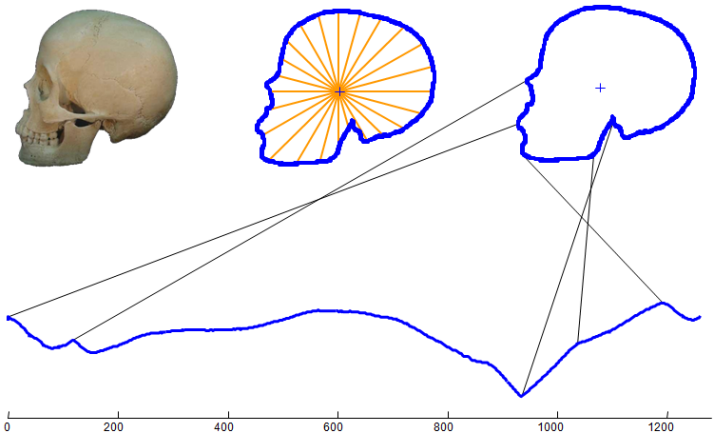


Applications

- shape retrieval using **time series**, e.g.,

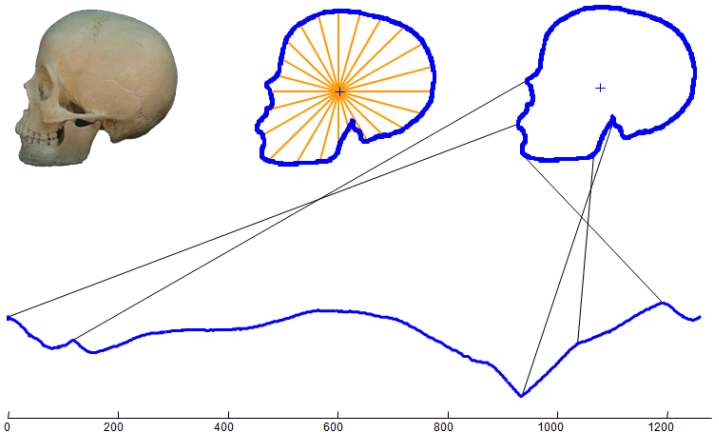
Applications

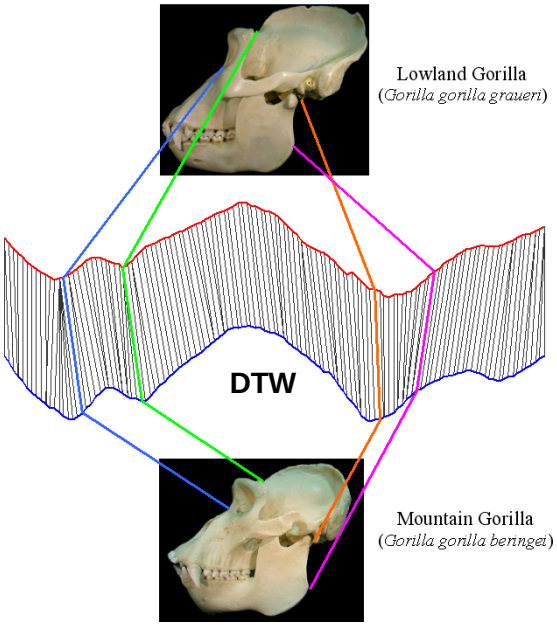
- shape retrieval using **time series**, e.g.,
  - having a closed polygon, centroid-to-contour distances are put into a time series (the descriptor)



Applications

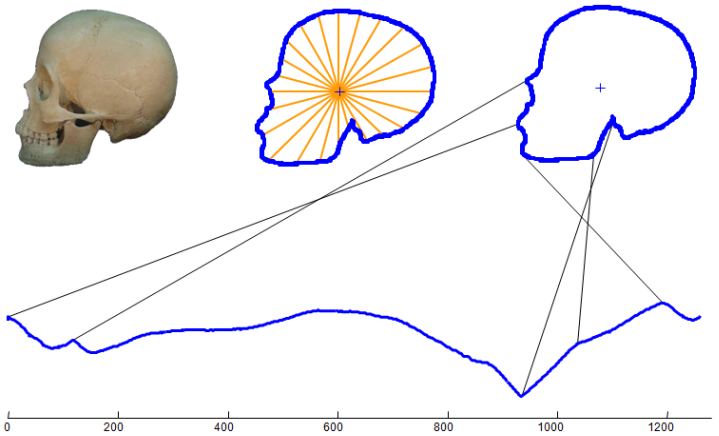
- shape retrieval using **time series**, e.g.,
  - having a closed polygon, centroid-to-contour distances are put into a time series (the descriptor)
  - **dynamic time warping distance** (DTW) or

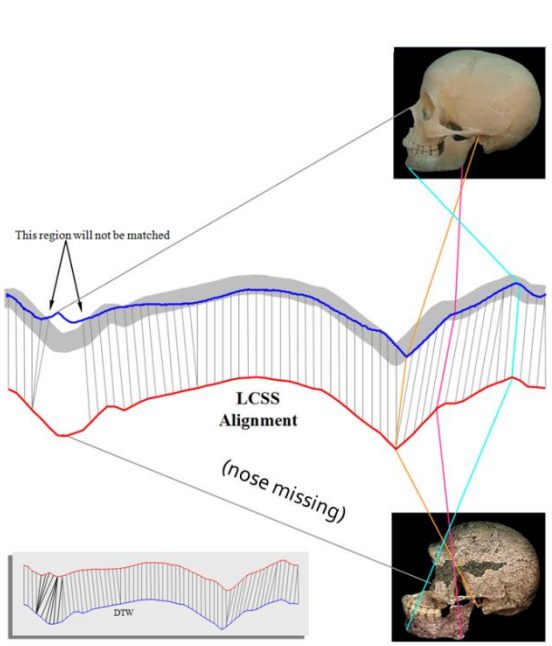


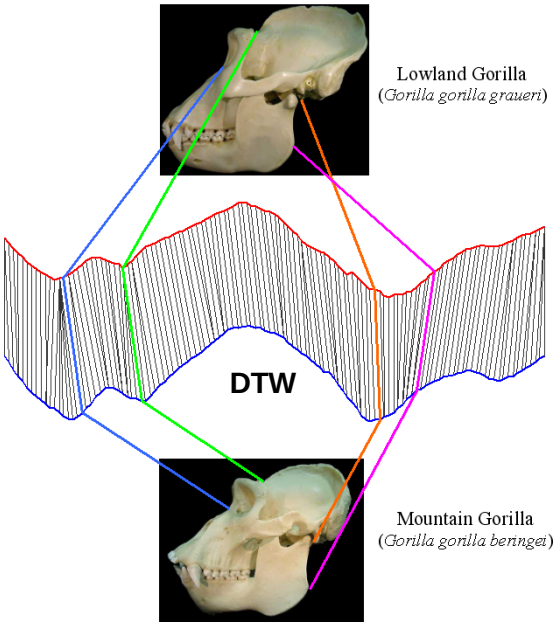


Applications

- shape retrieval using **time series**, e.g.,
  - having a closed polygon, centroid-to-contour distances are put into a time series (the descriptor)
  - **dynamic time warping distance** (DTW) or **longest common subsequence** (LCSS)

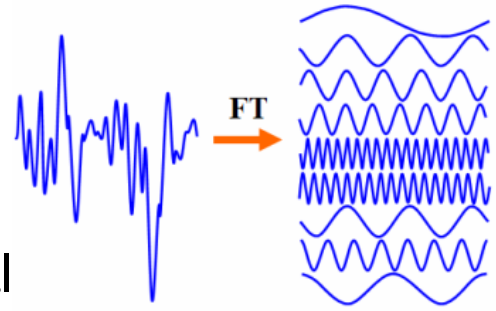




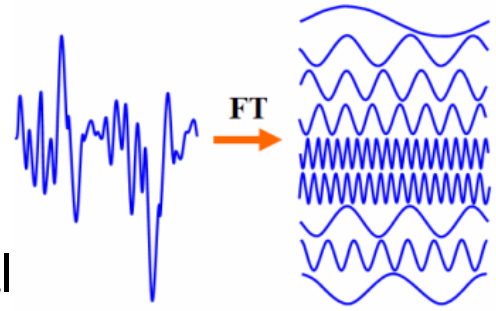


Applications

- audio retrieval using **waveform spectral information**
  - discrete Fourier transformation of the signal into the frequency domain – the power spectrum

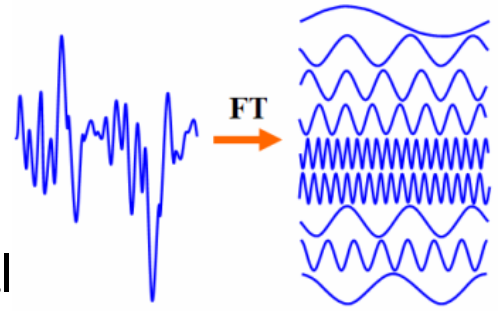


Applications

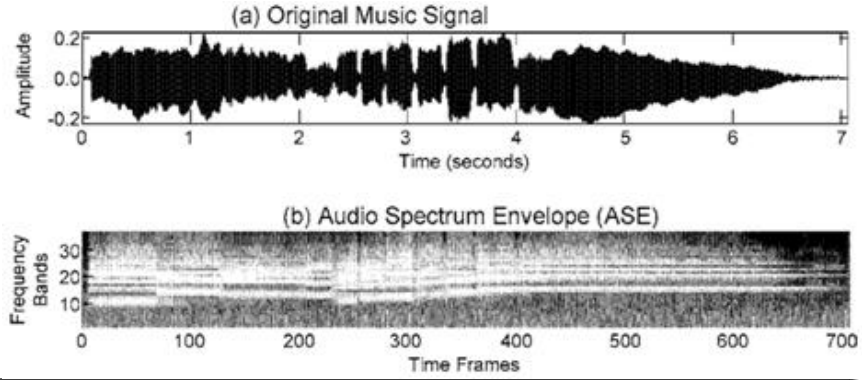


- audio retrieval using **waveform spectral information**
  - discrete Fourier transformation of the signal into the frequency domain – the power spectrum
  - **Audio Spectrum Envelope** (MPEG7)
    - local feature (for a time frame)
      - obtained by summing the energy of the original power spectrum within a series of logarithmically distributed frequency bands (log. due to human ear)

Applications



- audio retrieval using **waveform spectral information**
  - discrete Fourier transformation of the signal into the frequency domain – the power spectrum
  - **Audio Spectrum Envelope** (MPEG7)
    - local feature (for a time frame)
      - obtained by summing the energy of the original power spectrum within a series of logarithmically distributed frequency bands (log. due to human ear)
    - concatenation of local features into a spectrogram (the descriptor)
    - some distance for multidimensional time series

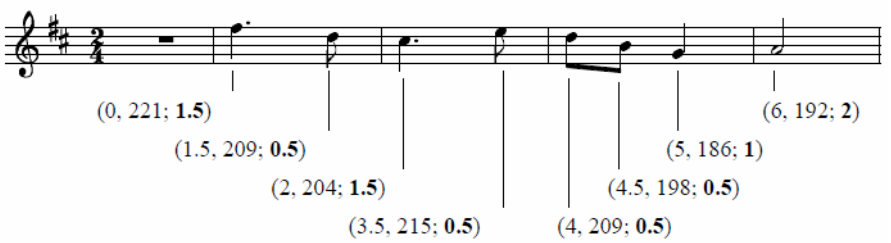


Applications

- audio retrieval using **melody/score**

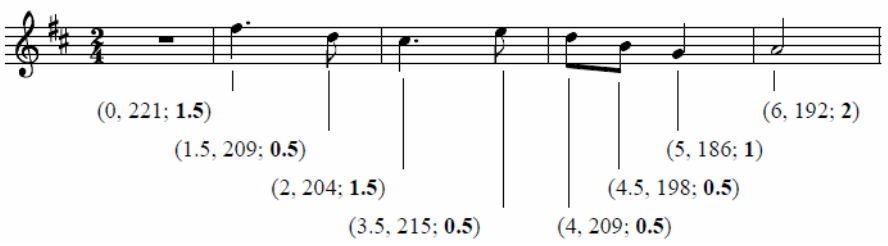
Applications

- audio retrieval using **melody/score**
  - notes (musical symbols) transformed into a set of 2D points
    - point location determined by the pitch and timing, while the weight of point was determined by a musical property

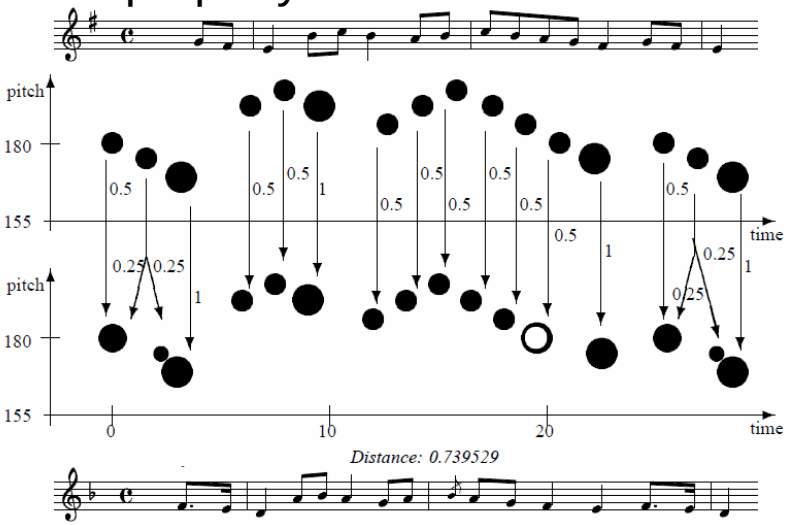


(images from: Typke et al., Using Transportation Distances for Measuring Melodic Similarity, 2003)

Applications



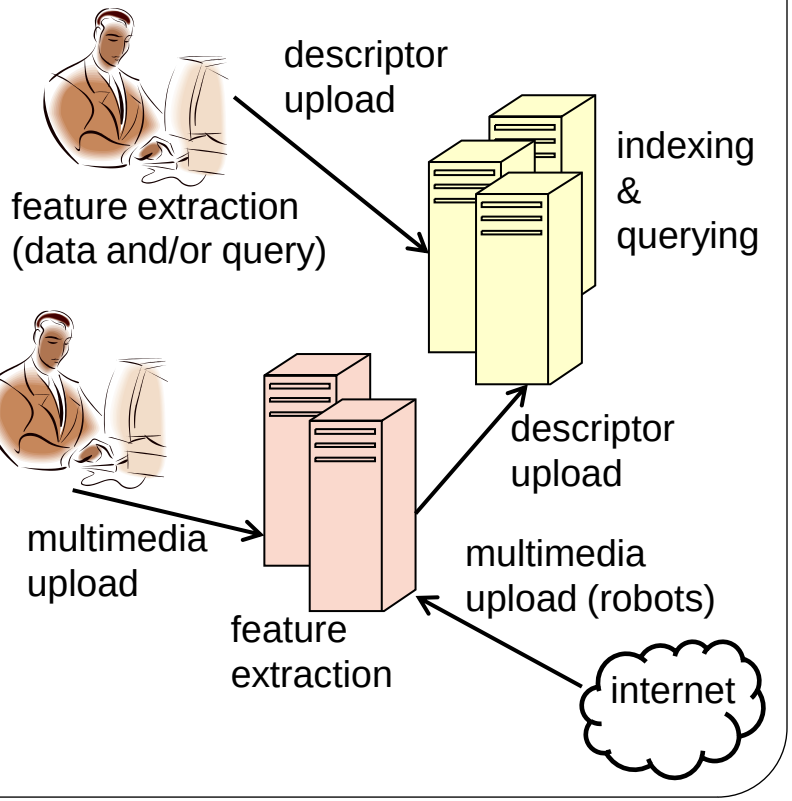
- audio retrieval using **melody/score**
  - notes (musical symbols) transformed into a set of 2D points
    - point location determined by the pitch and timing, while the weight of point was determined by a musical property
  - distance
    - Earth mover's distance
    - Proportional Transportation distance



(images from: Typke et al., Using Transportation Distances for Measuring Melodic Similarity, 2003)

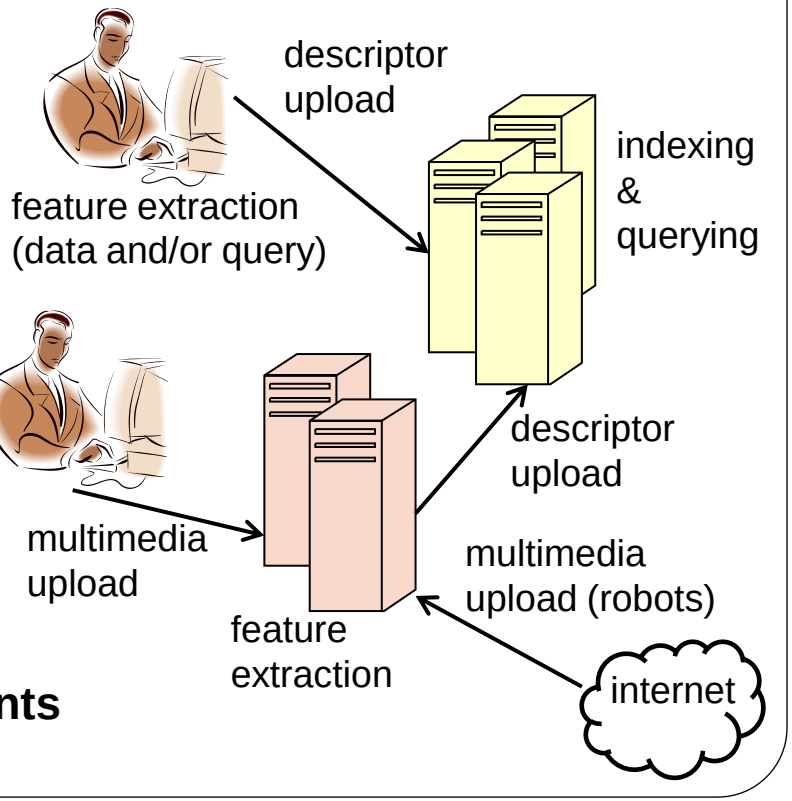
Performance of similarity search

- search engine subsystems
  - feature extraction – expensive, but
    - could be forwarded to the clients or to dedicated servers
      - both the data and the query (example)
    - well-parallelizable
      - just the multimedia object needed to extract descriptor, not the entire database
    - updates less frequent than querying



Performance of similarity search

- search engine subsystems
  - feature extraction – expensive, but
    - could be forwarded to the clients or to dedicated servers
      - both the data and the query (example)
    - well-parallelizable
      - just the multimedia object needed to extract descriptor, not the entire database
    - updates less frequent than querying
  - similarity indexing
    - the computational **complexity of distance function is crucial**
    - centralized or distributed index
      - **cannot be forwarded to the clients**



Computational complexity of distance functions

- cheap **O(n)**
  - L_p distances, cosine distance, Hamming distance

Computational complexity of distance functions

- cheap **O(n)**
  - L_p distances, cosine distance, Hamming distance
- more expensive **$O(n^2)$**
  - quadratic form distance, edit distance, DTW, LCSS, Hausdorff distance, Jaccard distance

Computational complexity of distance functions

- cheap **O(n)**
  - L_p distances, cosine distance, Hamming distance
- more expensive **$O(n^2)$**
  - quadratic form distance, edit distance, DTW, LCSS, Hausdorff distance, Jaccard distance
- even more expensive **$O(n^k)$**
  - tree edit distance

Computational complexity of distance functions

- cheap **O(n)**
  - L_p distances, cosine distance, Hamming distance
- more expensive **$O(n^2)$**
  - quadratic form distance, edit distance, DTW, LCSS, Hausdorff distance, Jaccard distance
- even more expensive **$O(n^k)$**
  - tree edit distance
- very expensive **$O(2^n)$**
  - (general) earth mover's distance (transportation problem, resp.)

Metric approach to efficient similarity search

- **assumption:** the distance δ is computationally expensive, so that querying by a sequential scan over the database of **n** objects is way too expensive

Metric approach to efficient similarity search

- **assumption:** the distance δ is computationally expensive, so that querying by a sequential scan over the database of **n** objects is way too expensive
- **the goal**: minimizing the number of distance computations $\delta(*,*)$ for a query, also I/Os

Metric approach to efficient similarity search

- **assumption:** the distance δ is computationally expensive, so that querying by a sequential scan over the database of **n** objects is way too expensive
- **the goal**: minimizing the number of distance computations $\delta(*,*)$ for a query, also I/Os
- **the way**: using **metric distances**

Metric approach to efficient similarity search

- **assumption:** the distance δ is computationally expensive, so that querying by a sequential scan over the database of **n** objects is way too expensive
- **the goal**: minimizing the number of distance computations $\delta(*,*)$ for a query, also I/Os
- **the way**: using **metric distances**
  - general, yet simple, model

Metric approach to efficient similarity search

- **assumption:** the distance δ is computationally expensive, so that querying by a sequential scan over the database of **n** objects is way too expensive
- **the goal**: minimizing the number of distance computations δ(∗,∗) for a query, also I/Os
- **the way**: using **metric distances**
  - general, yet simple, model
    - metric postulates

$$\begin{aligned} \delta(x,y) &= 0 && \Leftrightarrow x = y && \text{reflexivity} \\ \delta(x,y) &> 0 && \Leftrightarrow x \neq y && \text{non-negativity} \\ \delta(x,y) &= \delta(y,x) && && \text{symmetry} \\ \delta(x,y) + \delta(y,z) &\geq \delta(x,z) && && \text{triangle inequality} \end{aligned}$$

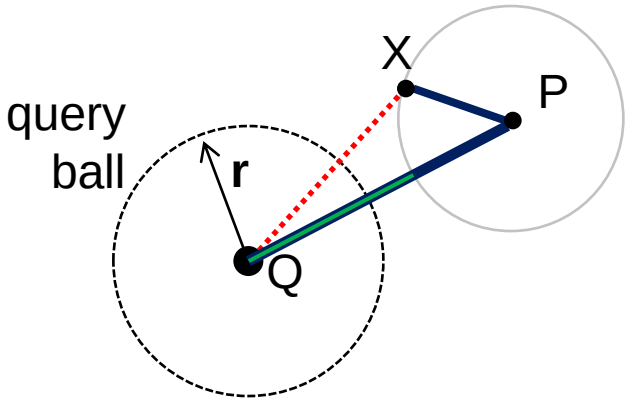
  - allows to partition and prune the data space (triangle inequality), resulting in a **metric index**
  - the search is performed just in several partitions → **efficient search**

Using lower-bound distances for filtering database objects

- a cheap determination of tight **lower-bound distance** of δ**(*,*)** provides a mechanism how to quickly filter irrelevant objects from search

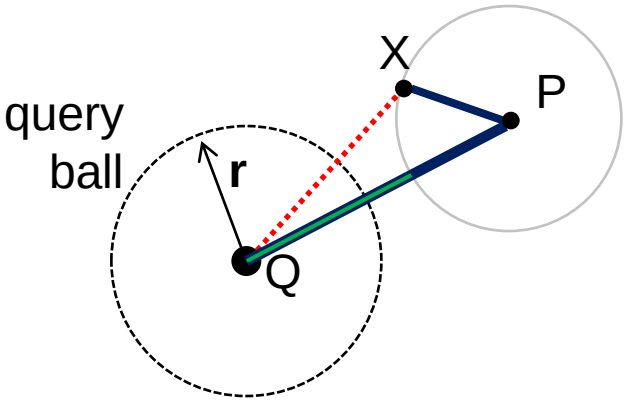
Using lower-bound distances for filtering database objects

- a cheap determination of tight **lower-bound distance** of **δ(*,*)** provides a mechanism how to quickly filter irrelevant objects from search



Using lower-bound distances for filtering database objects

- a cheap determination of tight **lower-bound distance** of **δ(*,*)** provides a mechanism how to quickly filter irrelevant objects from search

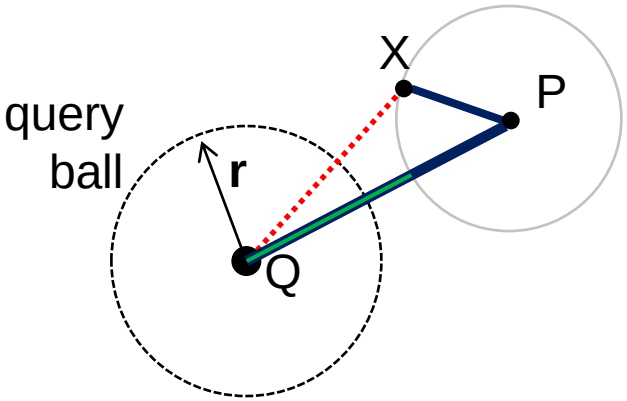


The task: check if **X** is inside query ball

- we know **δ(Q,P)**
- we know **δ(P,X)**
- we do not know **δ(Q,X)**
- we do not have to compute **δ(Q,X)**, because its lower bound **δ(Q,P)-δ (X,P)** is larger than **r**, so **X** surely cannot be in the query ball, so **X** is ignored

Using lower-bound distances for filtering database objects

- a cheap determination of tight **lower-bound distance** of **δ(*,*)** provides a mechanism how to quickly filter irrelevant objects from search

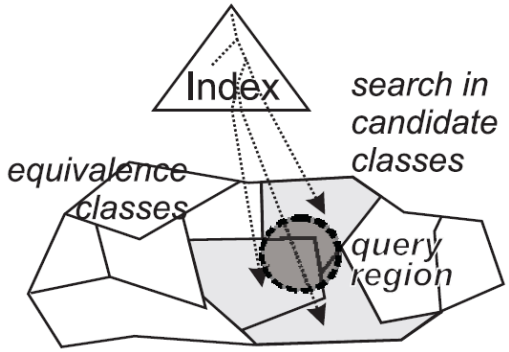


The task: check if **X** is inside query ball
- we know **δ(Q,P)**
- we know **δ(P,X)**
- we do not know **δ(Q,X)**
- we do not have to compute **δ(Q,X)**, because its lower bound **δ(Q,P)-δ (X,P)** is larger than **r**, so **X** surely cannot be in the query ball, so **X** is ignored

- this filtering is used in various forms by metric access methods, where **X** stands for a database object and **P** for a pivot object

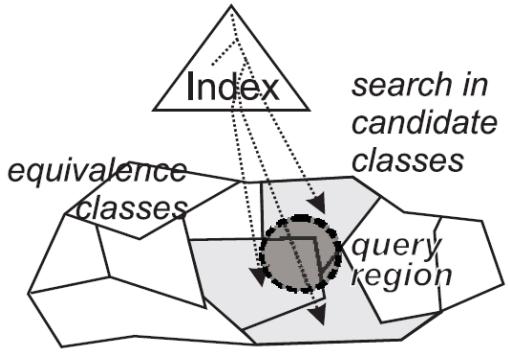
Metric access methods (MAMs)

- indexes for **efficient similarity search in metric spaces**
  - just the distances are used for indexing (structure of universe **U** unknown)
  - database **S** is partitioned into equivalence classes
  - index construction usually takes between O(k**n**) to O($\mathbf{n}^2$)



Metric access methods (MAMs)

- indexes for **efficient similarity search in metric spaces**
  - just the distances are used for indexing (structure of universe **U** unknown)
  - database **S** is partitioned into equivalence classes
  - index construction usually takes between O(k**n**) to O(**n**^{2})
- using the **lower-bounding** filtering when searching
  - pruning some equivalence classes, the remaining classes are searched sequentially



Metric access methods (MAMs)

- various structural designs
  - flat pivot tables (LAESA)
  - trees
    - ball-partitioning (M-tree)
    - hyperplane-partitioning (e.g., GNAT)
  - hashed indexes (D-index)
  - combinations of the above (PM-tree, M-index)
  - index-free approach (D-file)

Pivot tables

- set of **m** pivots p_i
- mapping individual objects into pivot space by use of the **m** pivots – distance matrix
  - $v_i = [\delta(o_i, p_1), \delta(o_i, p_2), ..., \delta(o_i, p_m)]$
  - contractive: $L_{max}(v_i, v_j) \leq \delta(o_i, o_j)$
    - different view of the pivot-based lower-bounding
- 2 phases
  - sequential scan of the distance matrix + filtering
  - the non-filtered candidates must be refined in the original space

Pivot tables

- set of **m** pivots p_i
- mapping individual objects into pivot space by use of the **m** pivots – distance matrix
  - $v_i = [\delta(o_i, p_1), \delta(o_i, p_2), ..., \delta(o_i, p_m)]$
  - contractive: $L_{max}(v_i, v_j) \leq \delta(o_i, o_j)$
    - different view of the pivot-based lower-bounding
- 2 phases
  - sequential scan of the distance matrix + filtering
  - the non-filtered candidates must be refined in the original space

	p_1	p_2
o_1	5	2.6
o_2	7.2	1.4
o_3	6.7	1.6
o_4	4.8	2.7
o_5	2.6	3.2
o_6	3.6	3.5
o_7	3.6	4.5
o_8	2.5	5.5

Pivot tables

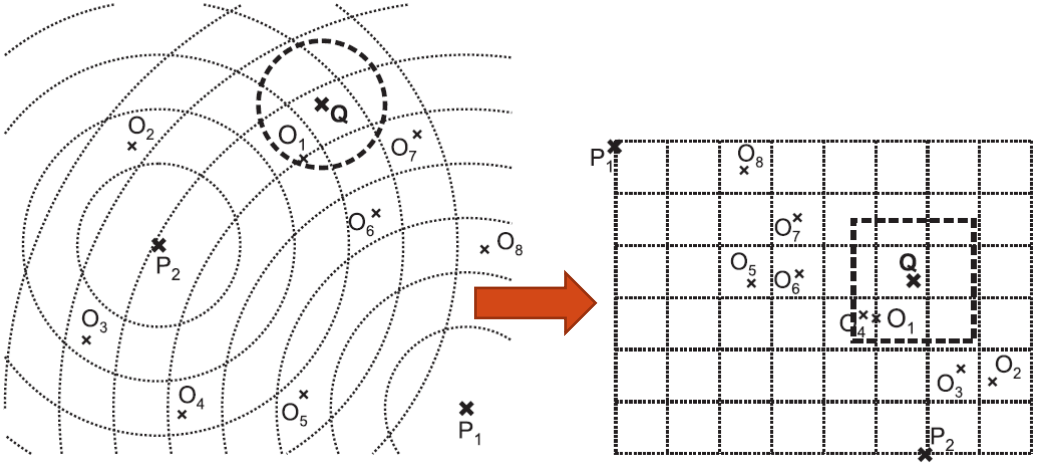
- set of **m** pivots p_i
- mapping individual objects into pivot space by use of the **m** pivots – distance matrix
  - $v_i = [\delta(o_i, p_1), \delta(o_i, p_2), ..., \delta(o_i, p_m)]$
  - contractive: $L_{max}(v_i, v_j) \leq \delta(o_i, o_j)$
    - different view of the pivot-based lower-bounding
- 2 phases
  - sequential scan of the distance matrix + filtering
  - the non-filtered candidates must be refined in the original space

	p_1	p_2
o_1	5	2.6
o_2	7.2	1.4
o_3	6.7	1.6
o_4	4.8	2.7
o_5	2.6	3.2
o_6	3.6	3.5
o_7	3.6	4.5
o_8	2.5	5.5

Pivot tables

- set of **m** pivots p_i
- mapping individual objects into pivot space by use of the **m** pivots – distance matrix
  - $v_i = [\delta(o_i, p_1), \delta(o_i, p_2), ..., \delta(o_i, p_m)]$
  - contractive: $L_{max}(v_i, v_j) \leq \delta(o_i, o_j)$
    - different view of the pivot-based lower-bounding
- 2 phases
  - sequential scan of the distance matrix + filtering
  - the non-filtered candidates must be refined in the original space

	p_1	p_2
o_1	5	2.6
o_2	7.2	1.4
o_3	6.7	1.6
o_4	4.8	2.7
o_5	2.6	3.2
o_6	3.6	3.5
o_7	3.6	4.5
o_8	2.5	5.5



Conclusion and future trends

- content-based similarity search is an important paradigm for searching multimedia content

Conclusion and future trends

- content-based similarity search is an important paradigm for searching multimedia content
- feature extraction and similarity models allow to focus on domain-specific semantics, which leads to more precise retrieval

Conclusion and future trends

- content-based similarity search is an important paradigm for searching multimedia content
- feature extraction and similarity models allow to focus on domain-specific semantics, which leads to more precise retrieval
- emerging phenomena
  will affect the research in the future
  - search aided by social networks (linked media)
  - more complex models, including non-metric similarities
  - powerful indexes for huge multimedia repositories

Thank you for attention!