course:
Multimedia Retrieval

lecture 1:

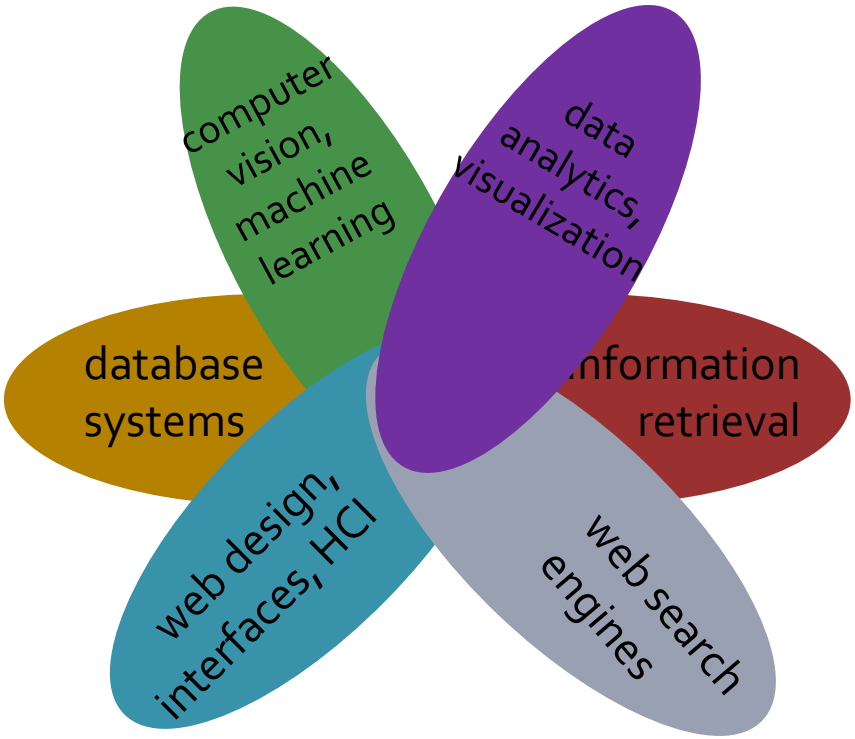
Web platforms for retrieval and sharing of multimedia content

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

The course focus

Multimedia
Retrieval

(on the Web)



What is the course about?

- web platforms for multimedia retrieval 3
- vocabulary-based image search 1
- image retrieval using neural networks 1
- content-based similarity search 2
- feature extraction techniques 4
- multimedia indexing 2

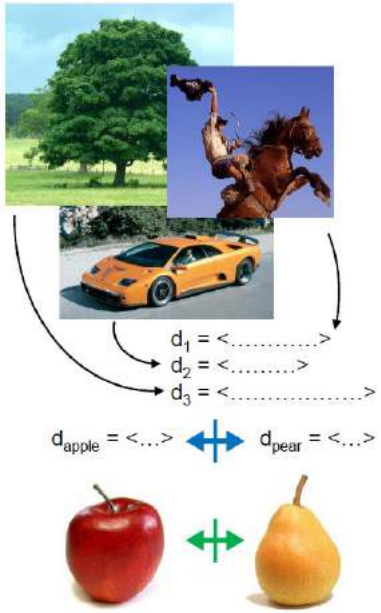
Software architecture view

GUI

web platforms
web interfaces
mobile apps

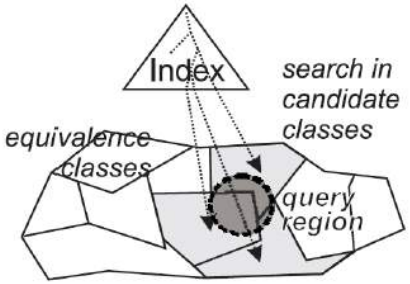
model

retrieval modalities
vocabulary-based search, NN
content-based similarity search
feature extraction models
visualization models

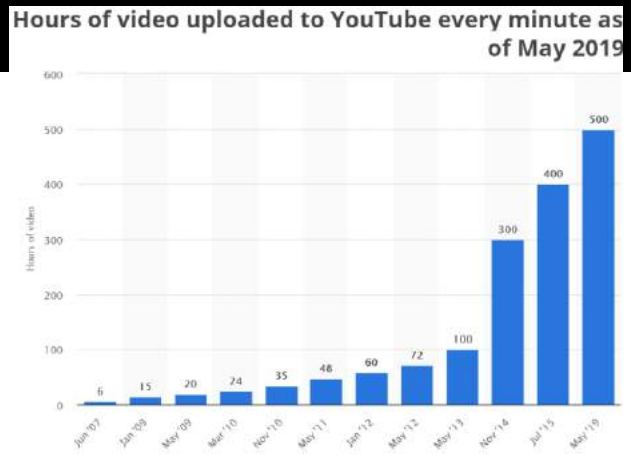


implementation

multimedia indexing
visualization



Multimedia content on the Web

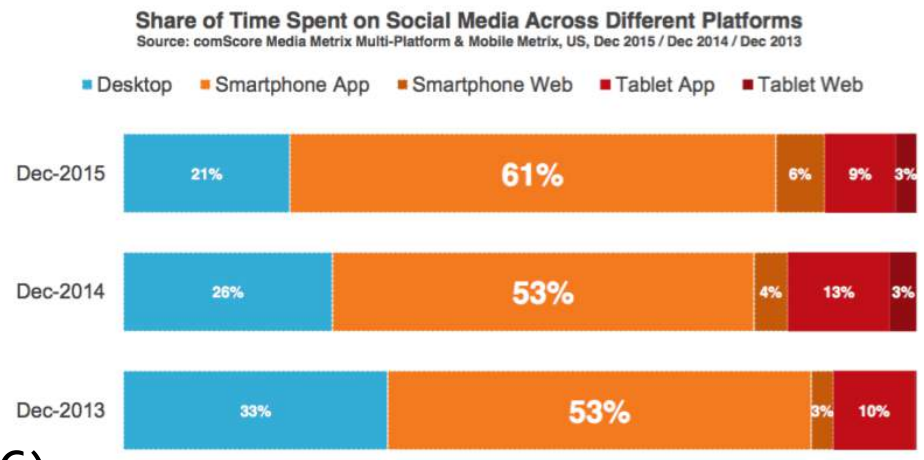


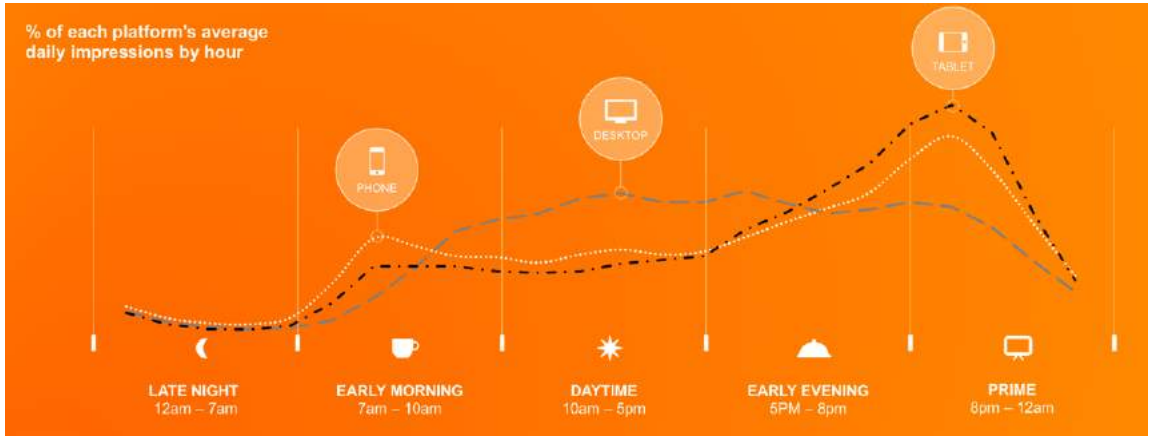
- > 99% of web space stores multimedia content,

  mostly at social networks
  - **billions** of photos **per day** uploaded
    - 100M photos+videos/day @ Instagram, **50 billion** in total (2019)
    - 350M photos/day @ Facebook, **250 billion** in total (2019)
  - **500 hrs** of video **per min** uploaded @YT, **5 billion** watched daily (2019)
- factors
  - high-speed internet, increasing computational power, clouds, cheap capturing devices (digital cameras/mobile phones/tablets)
    - everybody is data/information producer (Web 2.0)
  - human activities move to internet (cloud) in a large extent
    - entertainment (social networks), services (e-banking, e-shops, e-gov, …)

Client platforms

- desktop/notebook
  - on decline
- mobile
  - 70% YouTube (2017)
  - 80% social networks (2016)





David Chaffey, 2017

What is multimedia content?

- **multi–media**
  = more than one type of **digital media**

- traditional interpretation
  - **audio** and/or **video** content

- ambiguous terminology
  - one media type in a document
    - audio record, still image
  - multiple media types in a document
    - video = motion picture + audio record (+ text subtitles)

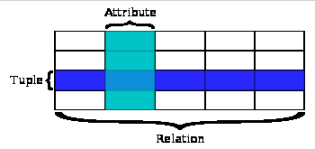
What is multimedia content?

- combinations
  - document **containers** (multiple tracks of data)
    - audio, video, presentation
  - extreme interpretation – the whole web is multimedia
- extended interpretation of multimedia
  - any **semantically unstructured** media type
  - mostly **"digitized nature"**
    - sensory data
  - also "**human-processed"** data
    - fulltext, XML, geometry & illustration, application (flash), etc.

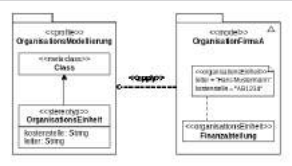
Structured vs. unstructured data

structured data
(with schema)

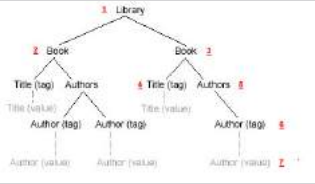
relational (SQL)



object(-relational)



XML (XPath, XQuery)



RDF (SPARQL)

key-value (NoSQL)

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

time

1950

1970

1990

2000

Big Data era

today

unstructured data
(schemaless)

fulltext

strings/
time series

sensory data
- gen. multimedia
- biometrics
- medical/scientific
- industrial

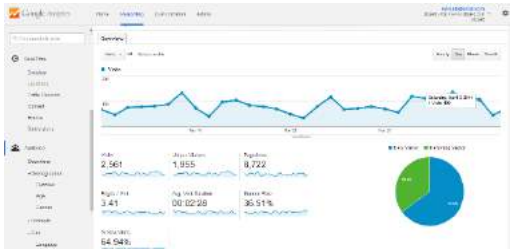
similarity search

stream/cloud processing
- 500 hrs/min of video uploaded @ YT
- 10 GB/sec from LHC in CERN

Unstructured data

fulltext
(plain text, web pages, PDF, ...)

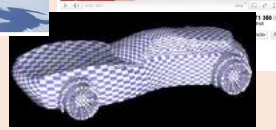
strings
(bioinformatics, ...)



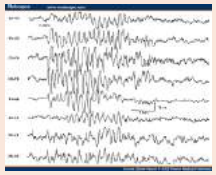
time series
(feature in time – sensory, clickstream, ...)

sensory data

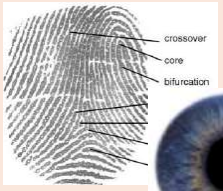
general multimedia
(image, audio, video, 3D models)



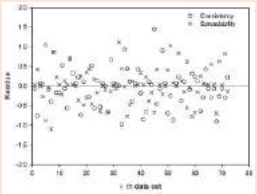
medical/scientific
(radiology, EEG, bio, ...)



biometrics
(fingerprint, iris, gait, face, ...)



industrial
(task-specific sensors)

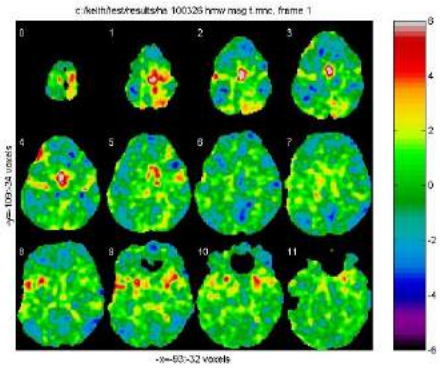


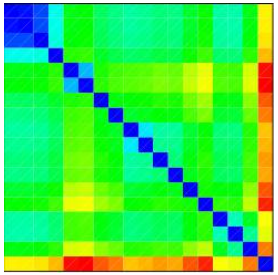
Multimedia domains (producers)

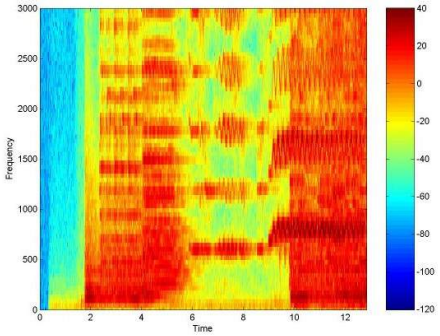
- free time end-users (billions)
  - humans create & share private content
  - generally not restricted domain
    - but mostly limited technically and semantically
      - photography taken by smartphones in real world indoor/outdoor (much less often drawings, charts, etc.)
      - widely shared semantics – people, city, nature, sports, …
      - visually complex scenes (e.g., background not separated)
  - non-critical usage
    - imprecise or no annotation
      - e.g., social context of hosting application
    - entertainment, socializing

Multimedia domains (producers)

- industry, research
  - professionals create content
  - many narrow subdomains
    - photography & other imaging
      - visible or invisible spectrum, X-rays, MRI, ultrasound, …
    - artificial images
      - processed photography, data visualizations
    - other sensory data
  - critical usage
    - precise annotation
    - part of decision chain (e.g., medical diagnosis)







Multimedia descriptors

- annotation – external description, high-level semantics
  - explicit: keywords, full text, URL, classification, GPS, …
  - contextual: e.g., relations in social network (comments, likes, shares)
- content-based feature descriptors
  - features extracted from the visual content (image pixels)
  - rather low-level features lacking semantics (semantic gap)
    - using deep learning, visual features become semantic as well
- multi-modal descriptors
  - combination of multiple descriptors
    - using deep learning, the content and annotation fuses together

Multimedia descriptors

Example:
a photograph hosted at Flickr using keyword "vacation"

visual content (image pixels)

explicit annotation

contextual annotation

Multimedia descriptors

Example:
a photograph found by Google Images using keyword "finance"

raw content

manual annotation

automated annotation

www.economywatch.com/finance-advertising-economy-advertising.html

EconomyWatch
Economy, Investment & Finance Reports

Follow us on Facebook, Twitter or Email

This site The web

Home News World Economy Investment Industries Banking Credit Card Credit Report Insurance Loans Jobs

Economic Statistics Beta

News Letter Subscription
email address
Submit

Country Economic Profiles
World Economy
North America Economy
US Economy
California Economy
Canada Economy
Mexico Economy
Asia Economy
India Economy
China Economy
Japan Economy
Australia Economy
Pakistan Economy
Korea Economy
Malaysia Economy
Singapore Economy
Vietnam Economy
Philippines Economy
Europe Economy
UK Economy
England Economy
Russia Economy
Germany Economy
France Economy
Ireland Economy
Italy Economy
Spain Economy
Africa Economy
South Africa Economy
Nigeria Economy
Middle East Economy
Iraq Economy
Egypt Economy
Dubai Economy
South America Economy
Brazil Economy
more...
International Trade
World Trade
US Trade
Canada Trade

About Us Advertise With Us Contact Us

Advertising on EconomyWatch.com

Hit Your Targets with Finance, Economics & Investing Advertising on EconomyWatch.com

Why Advertise on EconomyWatch.com?

EconomyWatch.com is the largest independent economics community online - in the world. **Every month, over 700,000 users consume over 1.5 million pageviews of specific targeted content in the finance, investing and economics categories.**

Since our beginnings as a small portal in 1999, we have become a trusted reference source used by everyone from professional investors to the World Bank and Economics Network of academics in the UK.

Advertising on EconomyWatch.com is one of the best ways to promote finance or investing products to a specialist audience.

Annotation vs. content features

- annotation
  - pros: high semantics, easy to search by
  - cons: subjective, incomplete, missing
    - varies from domain to domain
  - now also generated by deep learning

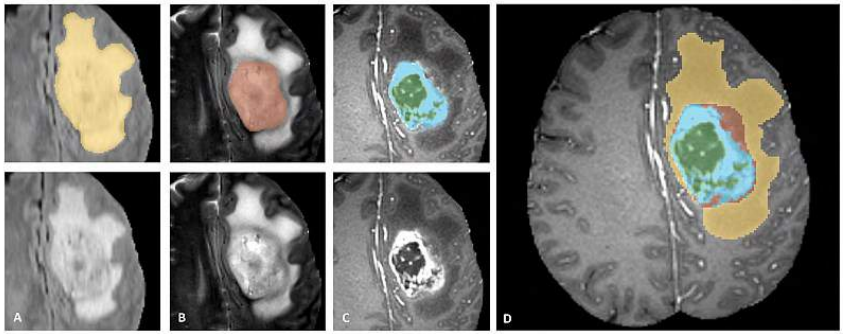
    e.g., keywordsready.com, cloud.google.com/vision/

    (but not perfect)
- content features
  - pros: objective, always present
  - cons: mostly low semantics, then low precision for semantic searches – semantic gap
    - but again, deep learning changes the game

basket | wicker | nature | pasture | wood | summer | fruit | garden | food | rustic | farming | grass | leaf | decoration | outdoors | vertical | color image | no people | love - emotion | healthy lifestyle | healthcare and medicine | body conscious | freshness | healthy eating | non-urban scene | day

Annotation vs. content features

- narrow domain
  - e.g., brain MRI scans
    - keywords, localization
- broad domain
  - object-level
    - "desk, laptop, chair, screen, window"…
  - concept-level
    - "office, clean, light, empty, business"…



gliome, tumor, tumor core, enhancing tumor structures, necrotic components

Software architectures for multimedia retrieval

- data-centric applications
  - search engines
  - hosting servers
    - amateur vs. professional (stock)
    - some provide middleware – web service, API
  - P2P multimedia sharing
- service-centric applications
  - social networks
  - map servers (infrastructure services)
  - interactive (mobile) platforms

Search engines

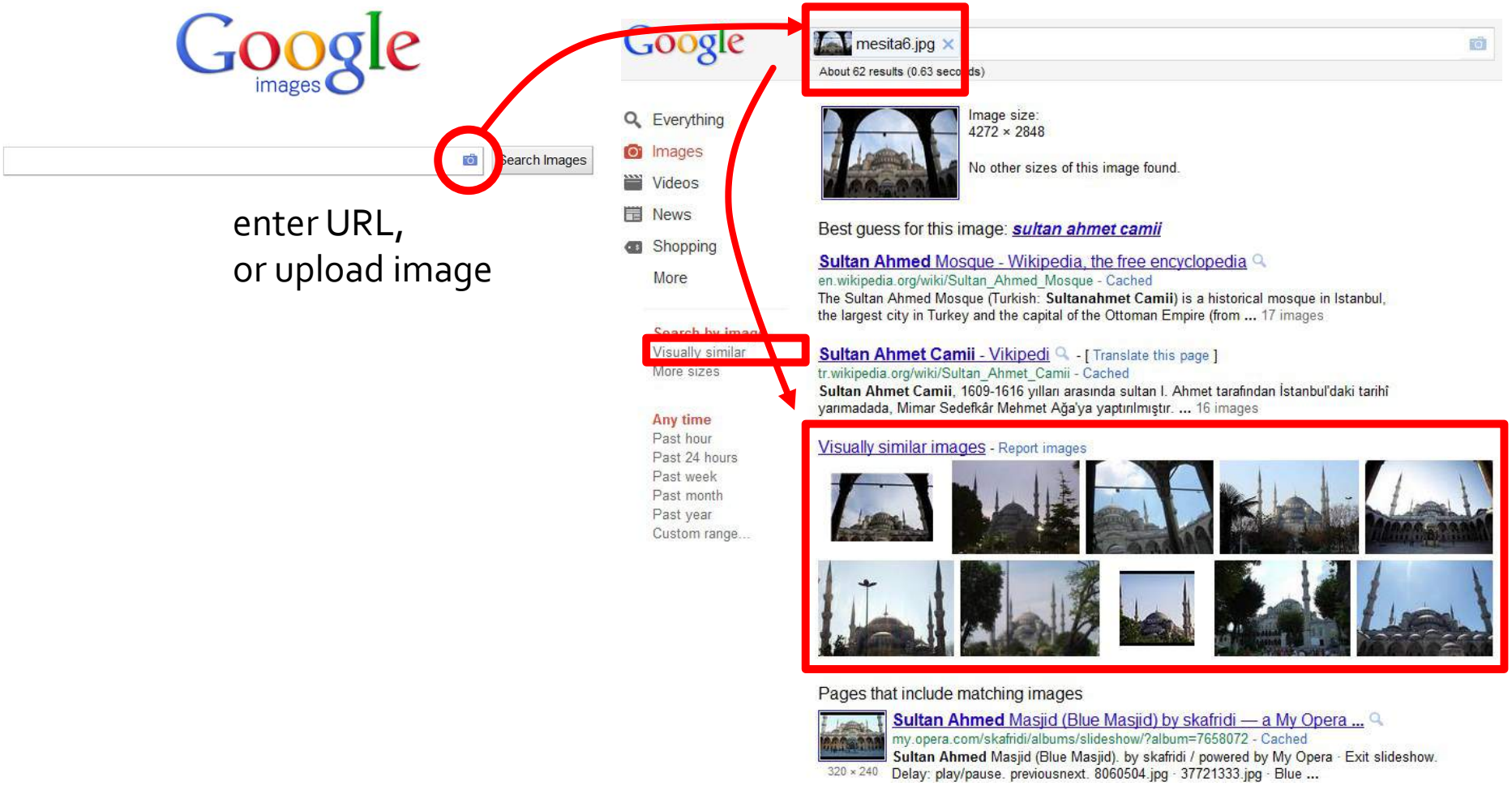
- analogy to traditional (text-based) web search
  - a crawler part (bots) download all the available web content (both multimedia + text)
  - indexing needed
    - data preprocessing allowing fast retrieval/queries
- search model
  - metadata-based (annotation-based)
    - keywords, categories, context in social network
  - content-based
    - the features of the content are used

Search engines

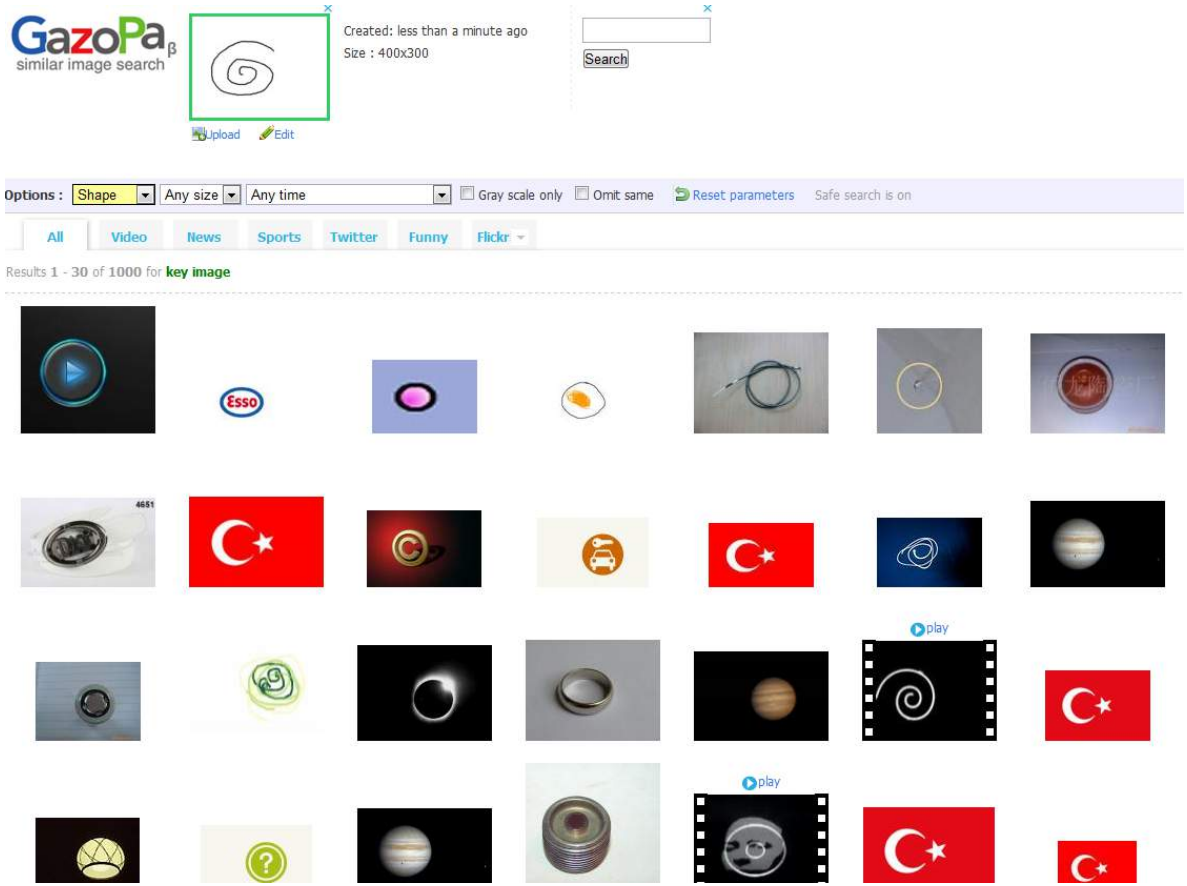
- mainstream multimedia search engines
  - mostly keyword search, some content-based functionality
  - **images:** Google Image Search, Bing Image Search, AllTheWeb, PicSearch
  - **video:** Bing Video Search, Lycos, AOL Video Search, SearchForVideo, BlinkX
  - **audio:** KaZaA, FindSounds, Skreemr, Yahoo Music Search
- content-based image retrieval engines
  - commercial & research prototypes/demos
  - Elastic Vision, Gazopa, Imense, Imprezzeo, Incogna, Like.com, MiPai, idee Visual Search Lab, Empora, Shopachu, TinEye, Tiltomo, eBay More Like This, ALIPR, Anaktisi, BRISC, Caliph & Emir, CIRES, FIRE, GNU Image Finding Tool, ISSBP, img(Rummager), imgSeek, IKONA, MUVIS, PIRIA, RETIN, Retrievr, SIMBA, TagProp, MUFIN
  - since 2011 Google started content-based image search

Search engines – keyword search

Search engines – content-based



Search engines – content-based



Hosting servers

- storage of content
  +
  multimedia search engine (crawler not needed)
  - keyword search
- trends towards social networks
  - huge multimedia container (mix of video, audio, text, application), sharing = networking
- embedded content vs. browsing hosting site
  - newspapers, blogs, personal web site

Hosting – Images

- general image hosting servers
  - albums, photoblogs, galleries
  - comments by users
    - social networks
  - mostly provide embedding (direct URL)
  - Flickr, PhotoBucket, ImageShack, Google Picasa, DeviantArt

Hosting – Audio & Video

- streaming vs. off-line
- broadcast vs. pre-recorded
  - live TV/radio broadcast
- mostly HTML5 (flash is dead)
  - generally application, not just video
  - allows advertising, recommendation system, user preferences
- general video hosting servers
  - **video:** YouTube, DailyMotion, Yahoo Video, MySpace, MetaCafe, MSN Video

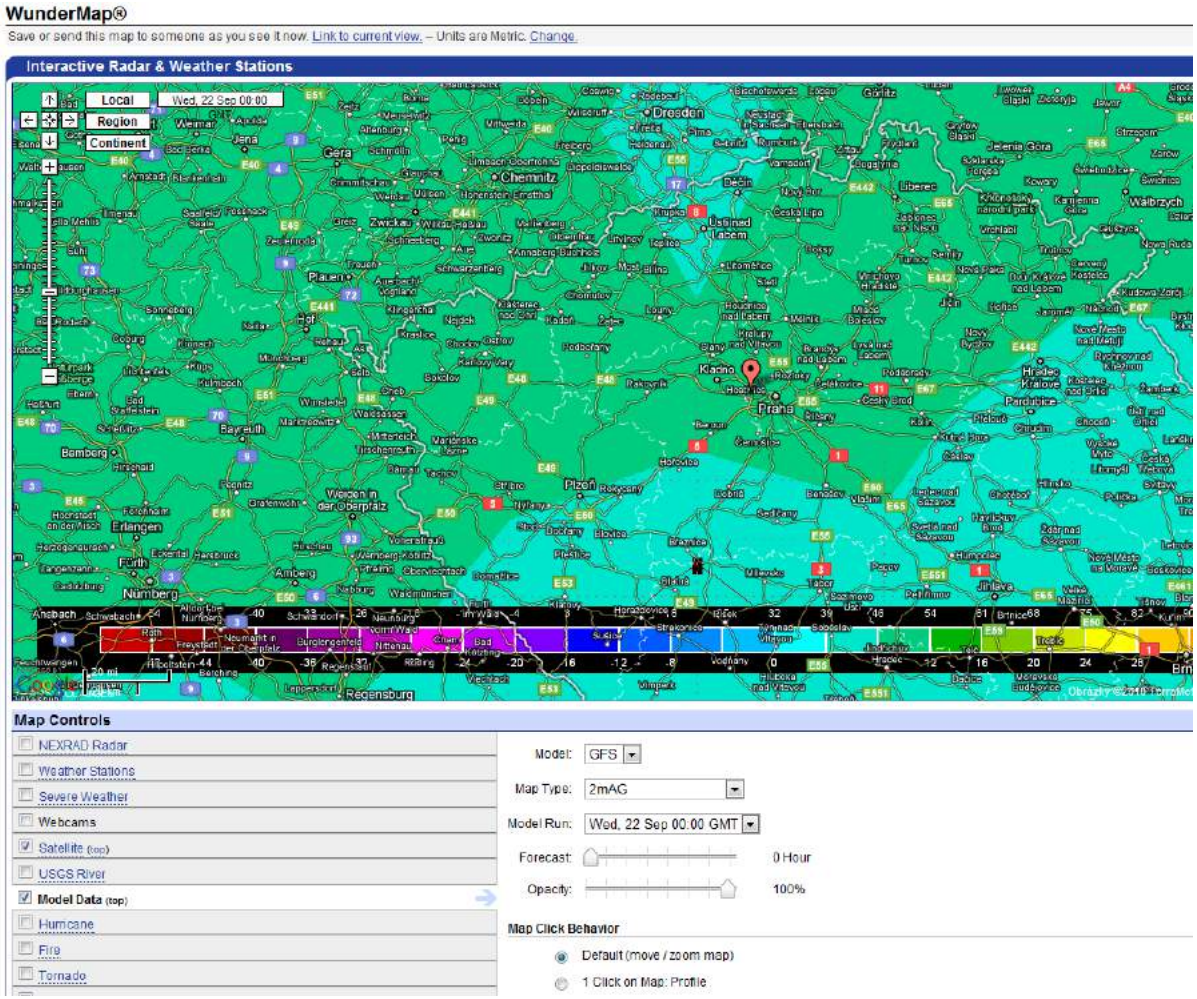
Hosting – stock servers

- cliparts for professional designers
  - paid content
  - image/video/audio/vector/flash content
  - each site up to 5-20 millions hosted objects
  - annotated content: keywords, categories
  - controlled quality (reviewing)
  - Corbis, Getty, iStockPhoto, ShutterStock, Fotolia, DreamsTime, Alamy, Veer

Hosting – API for retrieving multimedia

- GUI-based retrieval is limiting
- service (middleware) for internet applications
  - useful for aggregating applications (mash-ups)
  - more general than just embedding multimedia document on a web page
- examples
  - Google Maps API (customizable maps)
  - Flickr API (images)
  - YouTube APIs (videos)

Hosting
– API for retrieving multimedia



Service-centric applications

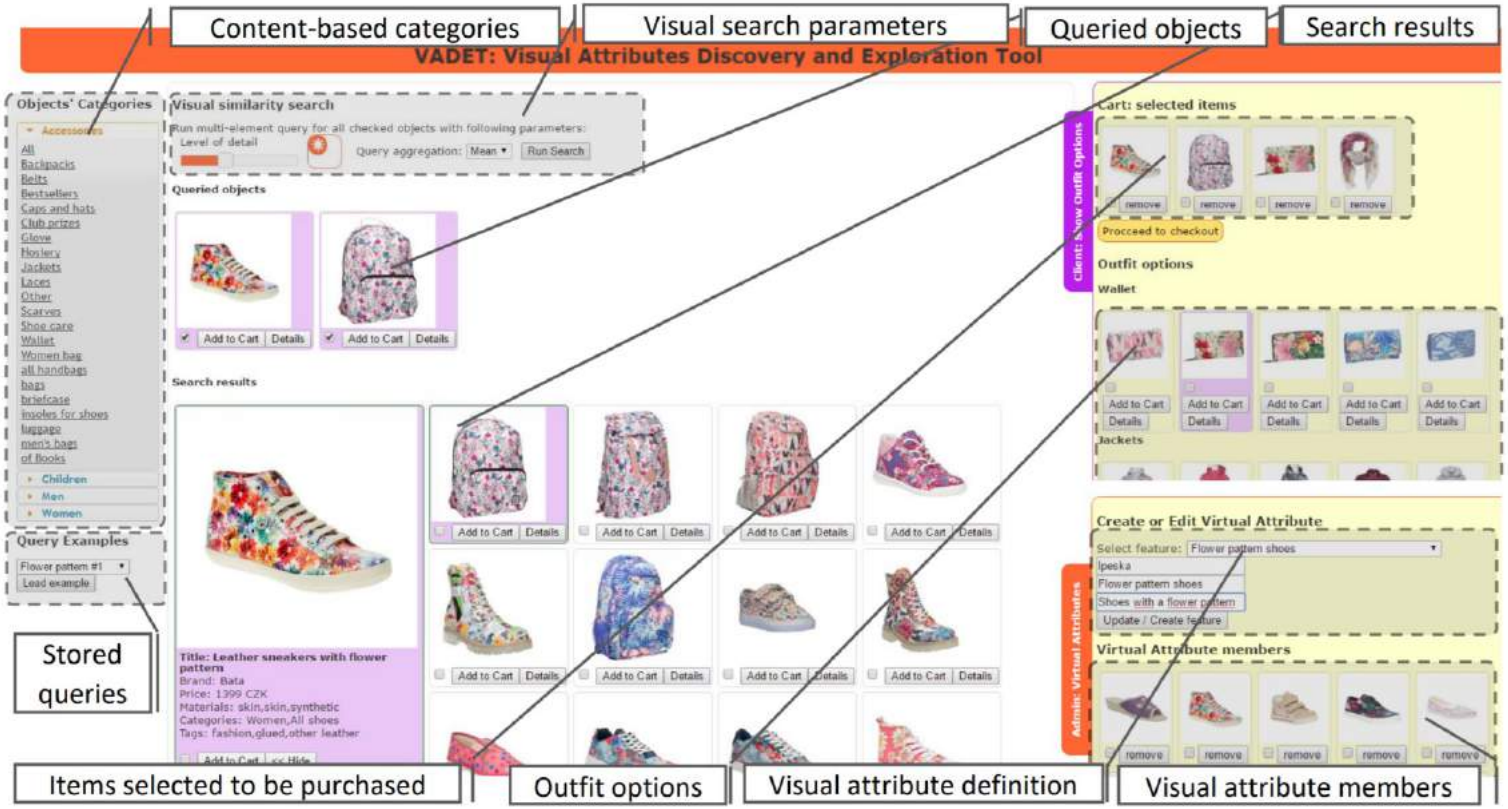
- next stage of multimedia retrieval
  - where multimedia is not the **target**, but the **means**
  - similarity search as hidden technology
    - querying mechanism and even the data type hidden to the user
- service is the final product
  - much wider applicability of similarity search?
    - e.g., e-commerce, "audiovisual wikipedia", etc.
  - multimedia object as connector
      - direct
      - image = entity (e.g., product to buy)
    - indirect
      - two persons in a photograph might be in relationship
      - increasing the power of social networks

Service-centric applications – social networks

- combine/link multimedia content, user profiles, groups, blogs, etc.
  - advanced annotation (links, memberships)
  - content annotation (people in photographs)
  - applications (flash)
  - FaceBook, MySpace, YouTube…
- generating social relations (edges in the social graph)
  - sharing multimedia content
  - identifying persons in image/video

Service-centric – e-shops

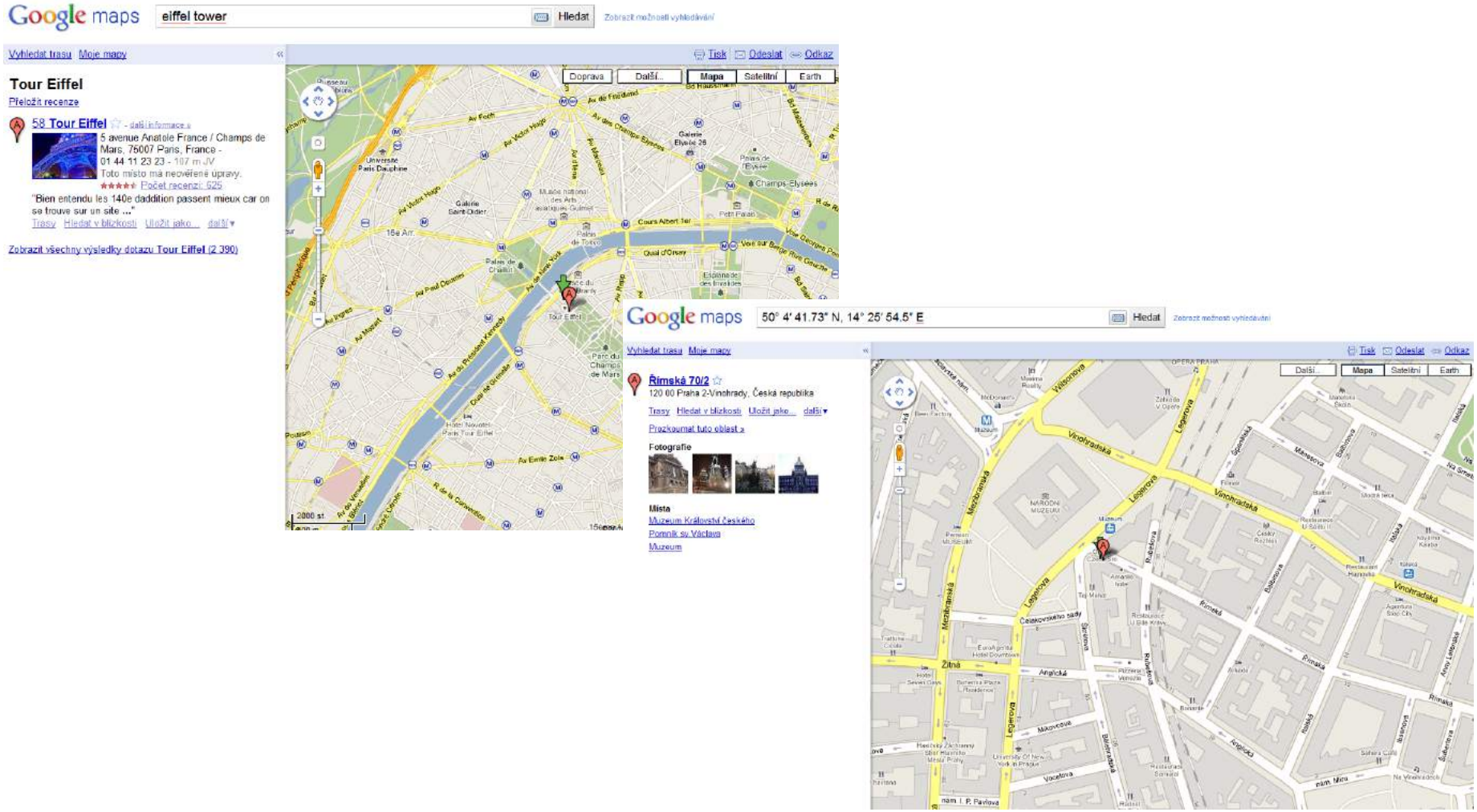
- multi-model and multi-modal search



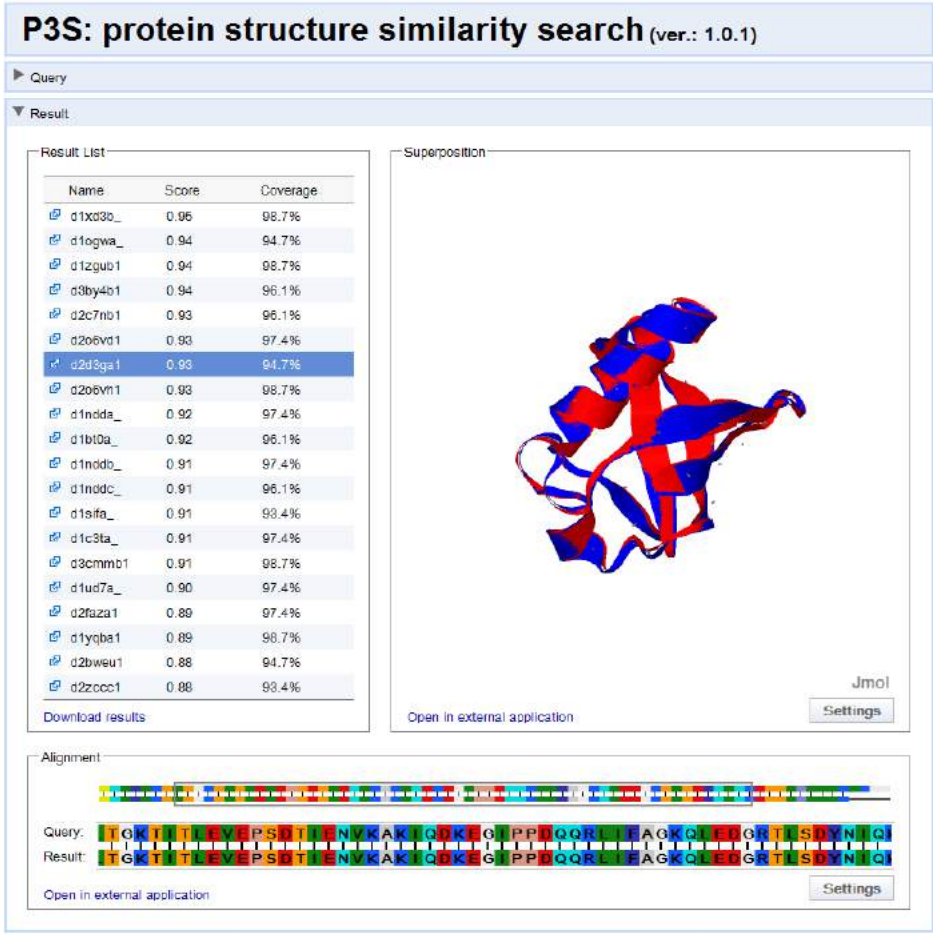
Service-centric applications – map servers

- general maps
  - specific multimedia object (single large image)
  - browsing/navigation rather than retrieval
  - text-based search (address), also GPS coordinates
- advanced applications (specific map layers)
  - weather
  - web-based geographic information systems (GIS)
    - tourist info, transport navigation
  - generally infrastructure services (electronic guide)

Service-centric applications – map servers



Service-centric applications – scientific tasks



www.siret.cz/p3s

Service-centric applications – interactive (mobile) platforms

- capturing + retrieval by multimedia as a mean of general information retrieval
- desktop retrieval architectures are not convenient (hardware-based limitation)
- mobile devices (smartphones, tablets) enable easy retrieval based on multimedia
- the "killer application" for content-based multimedia retrieval
  - idea:
    take picture (record a sound) of something and the device will answer **what it is**, **where to buy it**, **what is the price**, **who is that person**, **what is the song's name** …
  - i.e., universal answering machine (what do I see/hear)

Mobile platforms – mobile Google

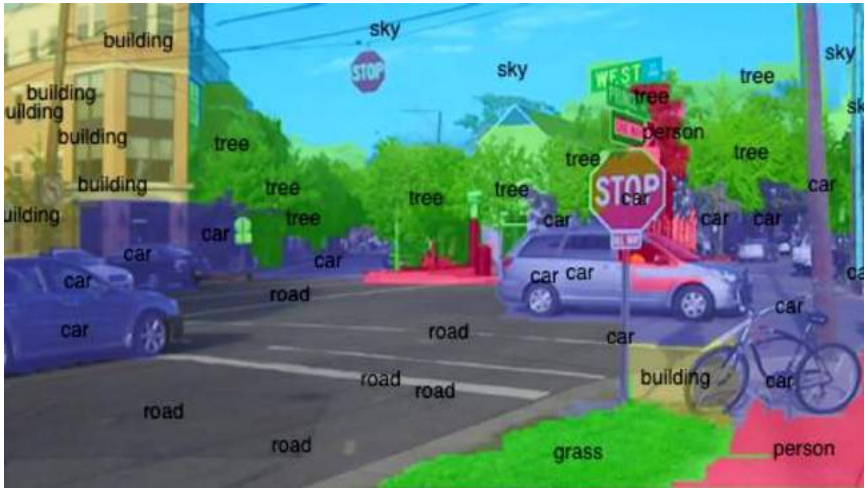
- mobile version of Google using content-based image retrieval
  - combination of OCR and pattern matching (similarity-based) techniques
  - Android, iOS (iPhone, iPad)
- nowadays provides classification of:

Mobile platforms – mobile Google

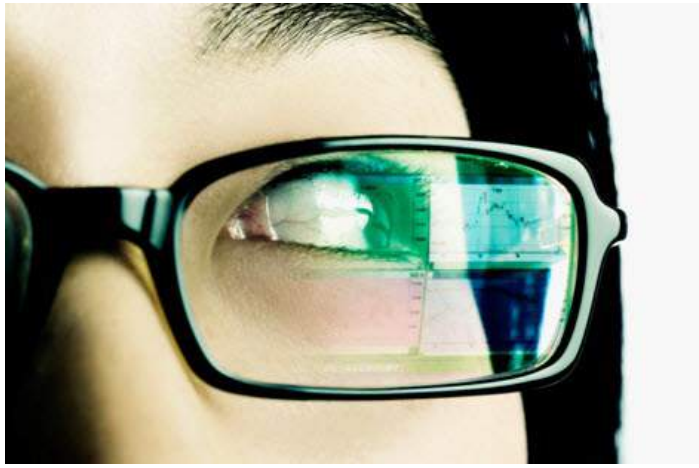
(credit: Google)

Extended mobile platforms

- augmented reality
  - i.e., real-time image/audio classification



Google's MobileNetV2



course:
Multimedia Retrieval (NDBI034)
© Tomáš Skopal, 2018

lecture 2:

Web interfaces, means and quality of retrieval

doc. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

- means of retrieval
  - queries
  - browsing
  - exploration
  - recommendation
- retrieval effectiveness
  - semantic gap
  - how to measure the quality of retrieval

Models of relevance in retrieval

- binary relevance
  - an object ***is*** *or* ***is not*** relevant to my retrieval intent
    - classic set theory (crisp sets)
- multi-value relevance
  - object is ***somehow*** relevant to my retrieval intent
    - geometry, fuzzy set theory, probabilistic theory, …
  - pair-wise similarity function
    - subject of the next lecture (for now a black box)

Means of retrieval

	querying	browsing	exploration	recommendation
using annotation	**keyword search,** form query, full-text search, …	graph browsing, sequence of keyword queries, …	navigating in hierarchy of concepts (ontology)	explicit vs. implicit filter
content-based	**query by example,** query by sketch, query by humming, etc.	graph browsing, sequence of content queries	**navigating in hierarchy of concepts** (or just representatives)	explicit vs. implicit content filter

Means of retrieval – Query

- assumption
  - we are able to specify our search intent
- query = explicit formulation of one-shot search intent
- query models
  - text-based (annotation based)
  - content-based
    - file upload, URL, sketch
- query result
  - binary relevance – unordered set of objects
  - multi-value relevance – ranking on database objects
- relevance feedback, re-ranking

Queries

Query formulation:

text-based

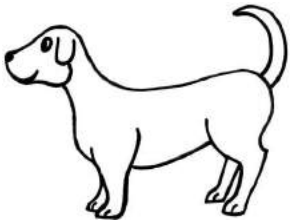
keywords

"dog, labrador"

content-based

example

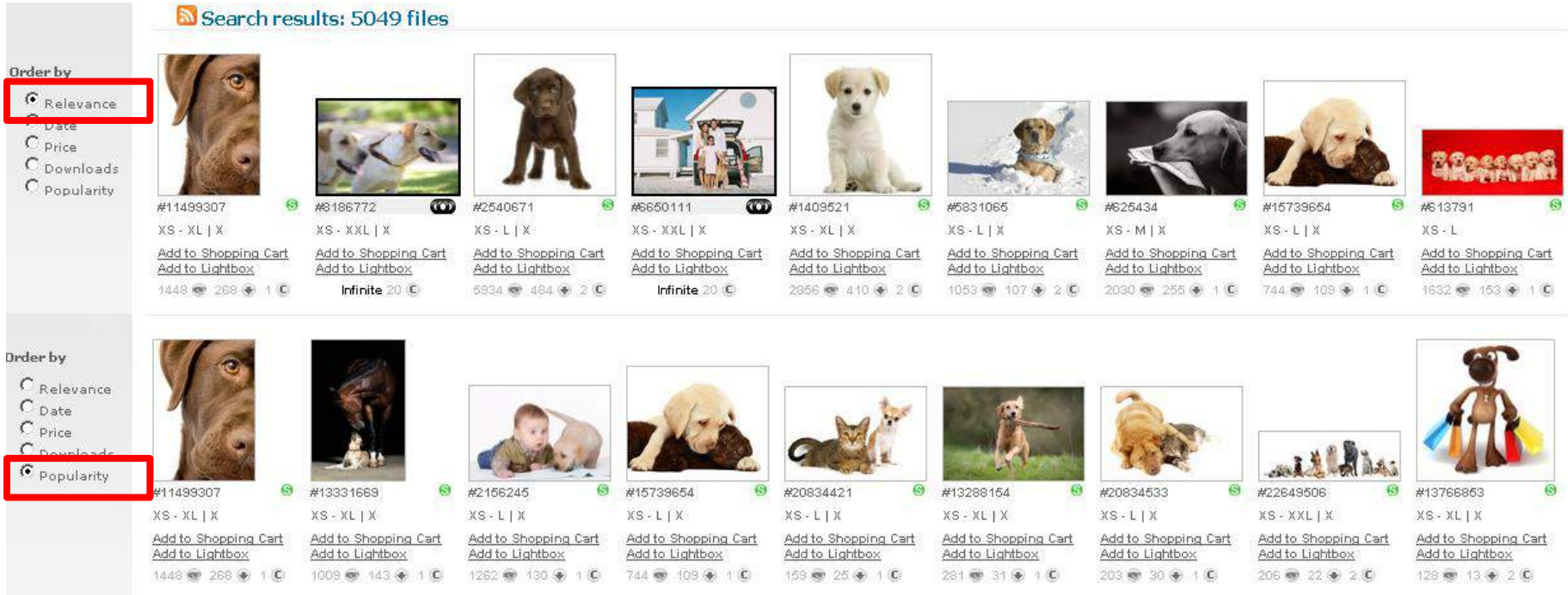
sketch



Query result:

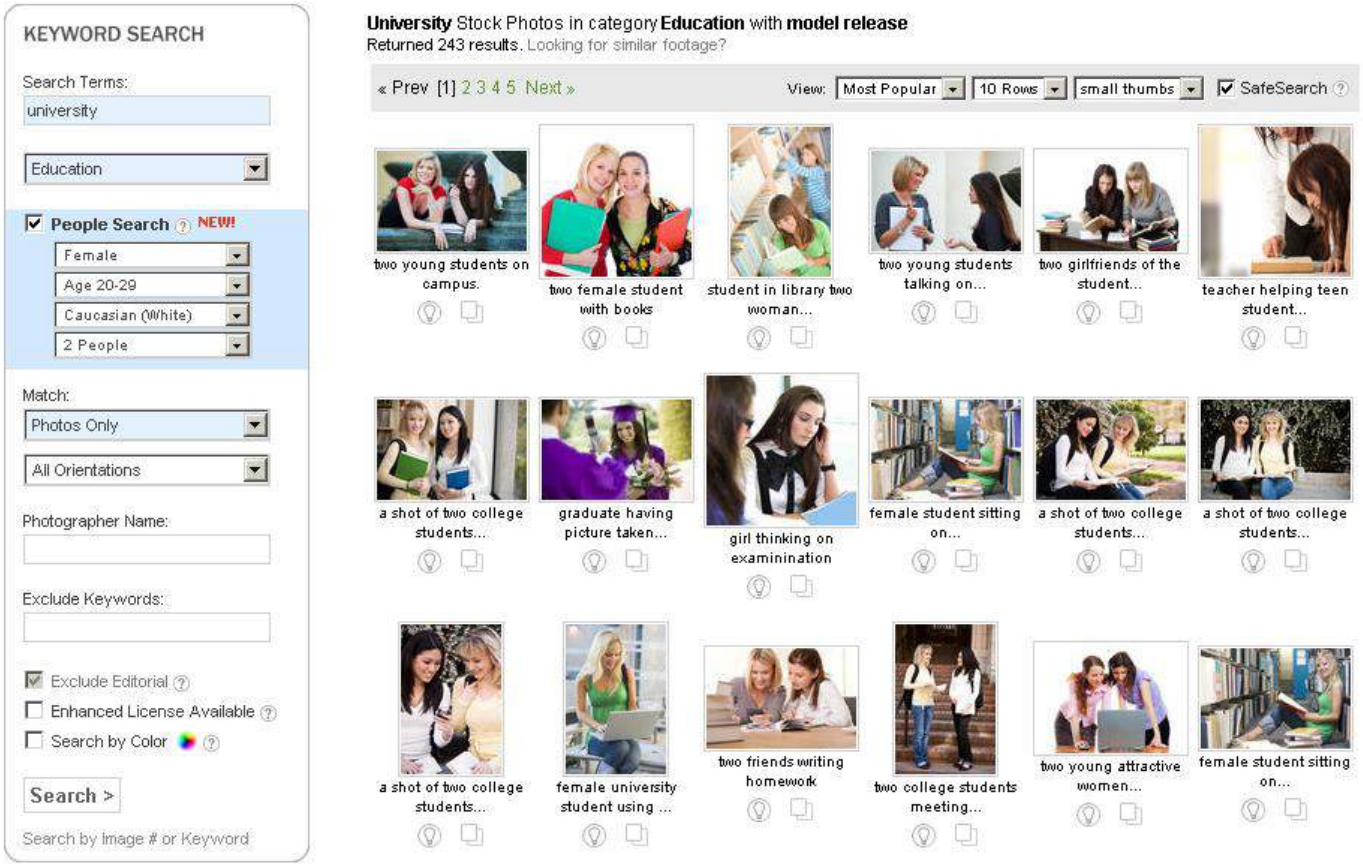
0.9 | 0.85 | 0.7 | 0.6 | 0.4 **ranking**

Example – ranked query result



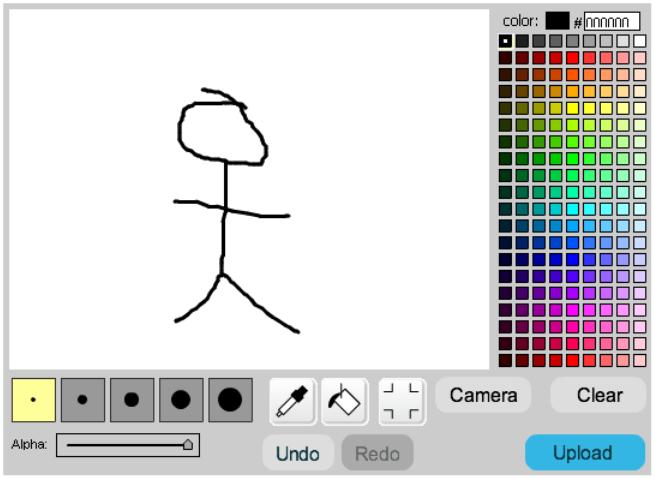
(credit: fotolia.com)

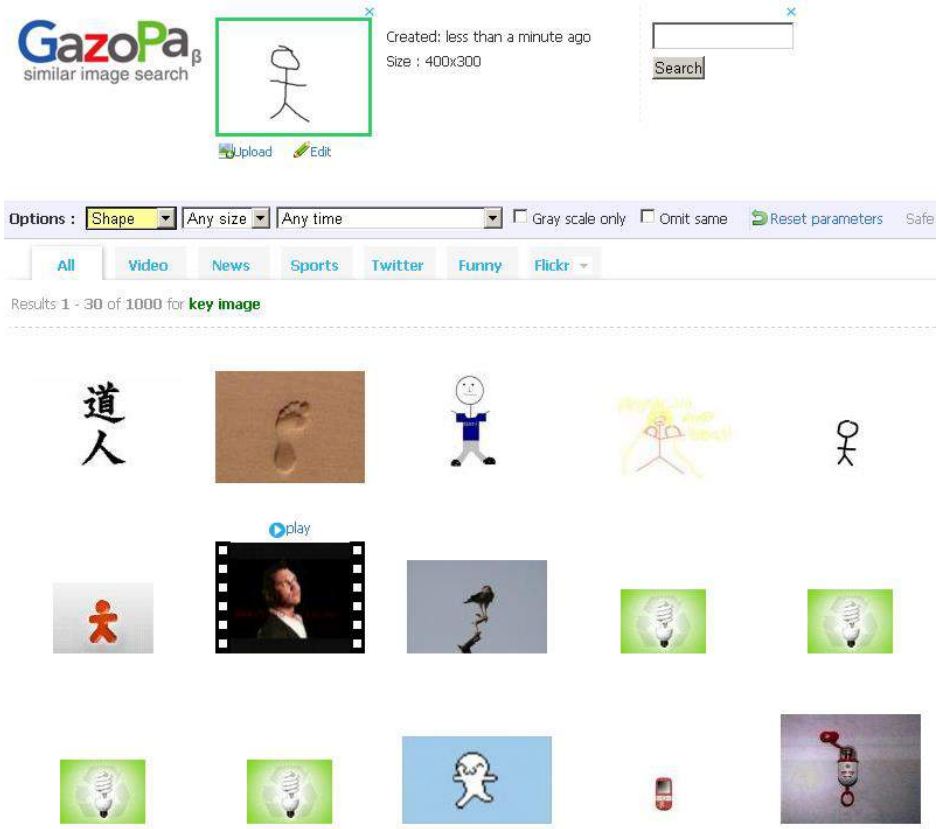
Example – query by form + keywords



(credit: shutterstock.com)

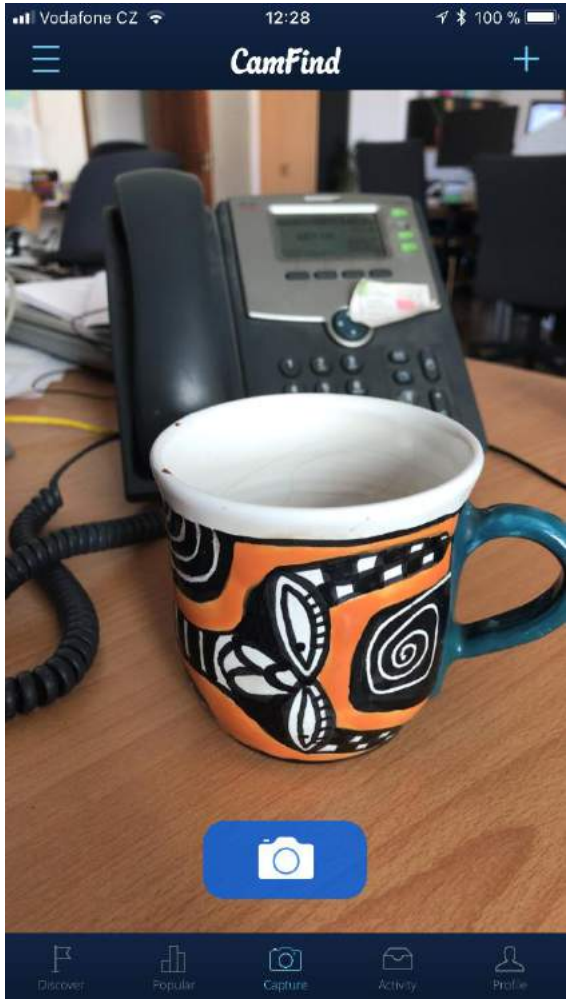
Example – query by sketch/example

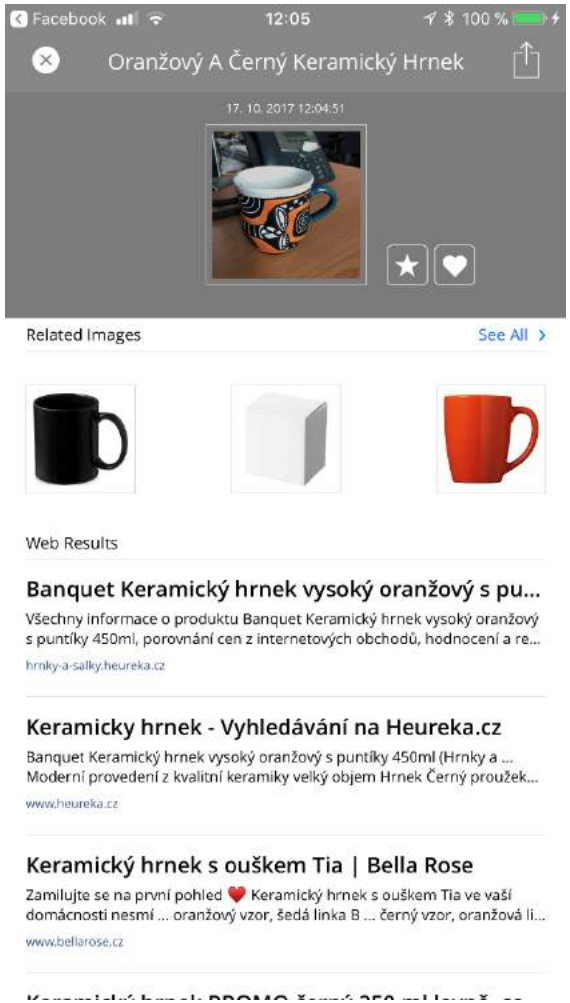




(credit: gazopa.com)

Example – query by example using mobile devices (e.g.,CamFind)

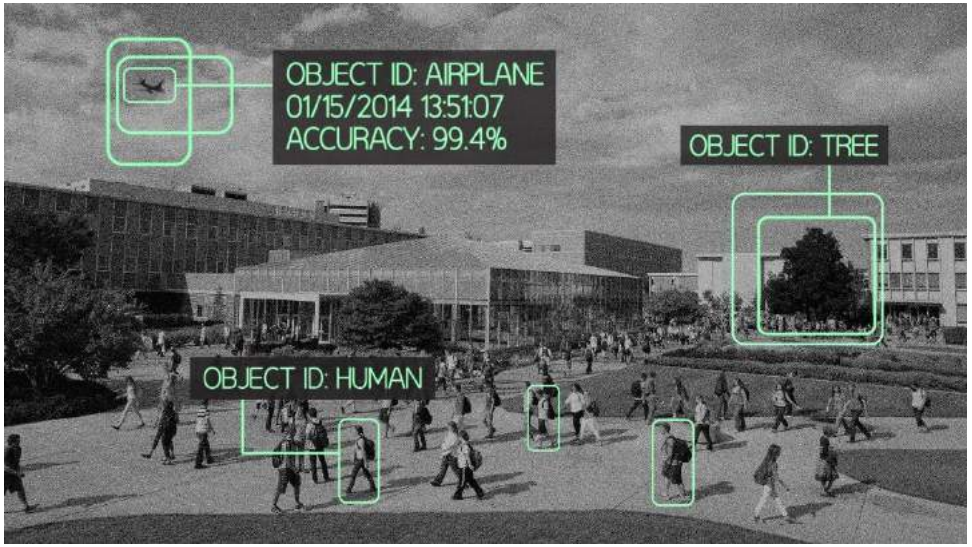




Example – smart camera (realtime)

- by camera localization (GPS)
  - services, landmarks, dinning, shops, etc.

- by image
  - faces localization, QR codes recognition, recently general objects recognition



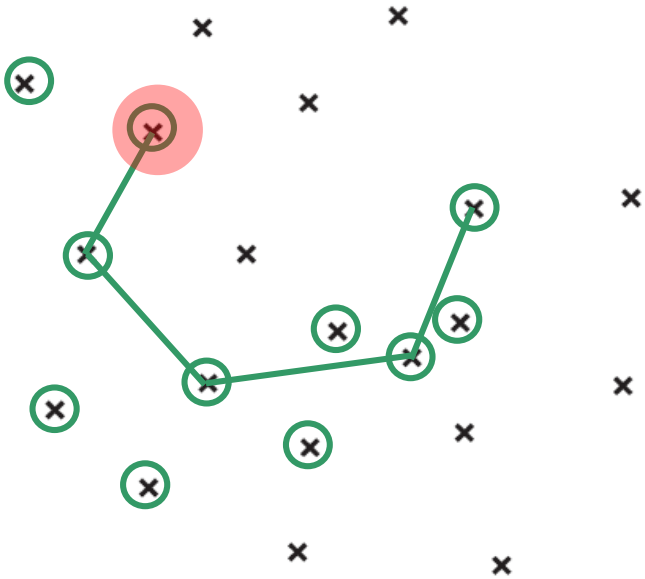
Akshika Wijesundara, Object Recognition with OpenCV on Android

Means of retrieval – Browsing

- assumption
  - we are not able to (well) specify our search intent
- browsing = iterative navigation in the database
- browsing models
  - explicit graph (links between entities)
  - virtual graph (series of queries)
- browsing result
  - subset of database objects
  - set of clusters (representatives)
  - hierarchy (ontology)

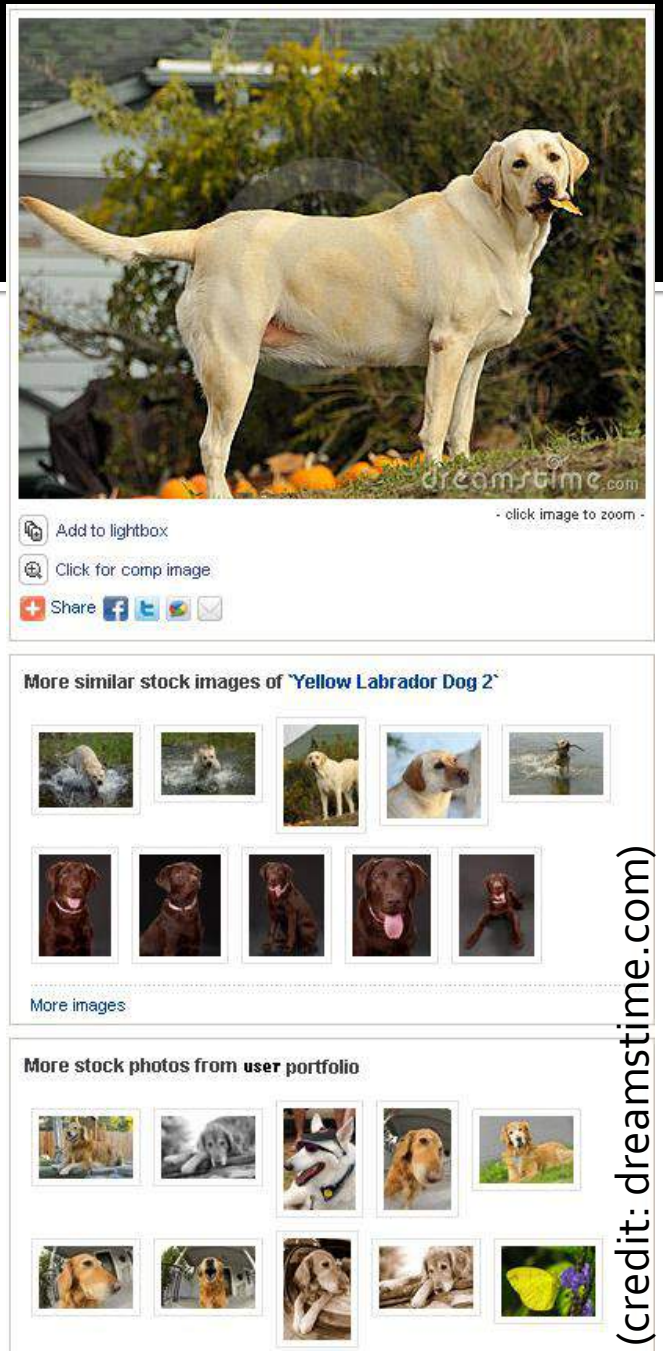
Means of retrieval – Browsing

- browsing explicit graph
  - static hierarchy of categories
  - dynamic network
- browsing virtual graph
  - series of queries
  - an object in query result is used to specify the new query



Example – browsing images

- explicit graph
  - links to images in user portfolio
- virtual graph
  - links to similar images
    - query by keywords obtained from the image annotation
    - weighted keywords
      - weight = font size

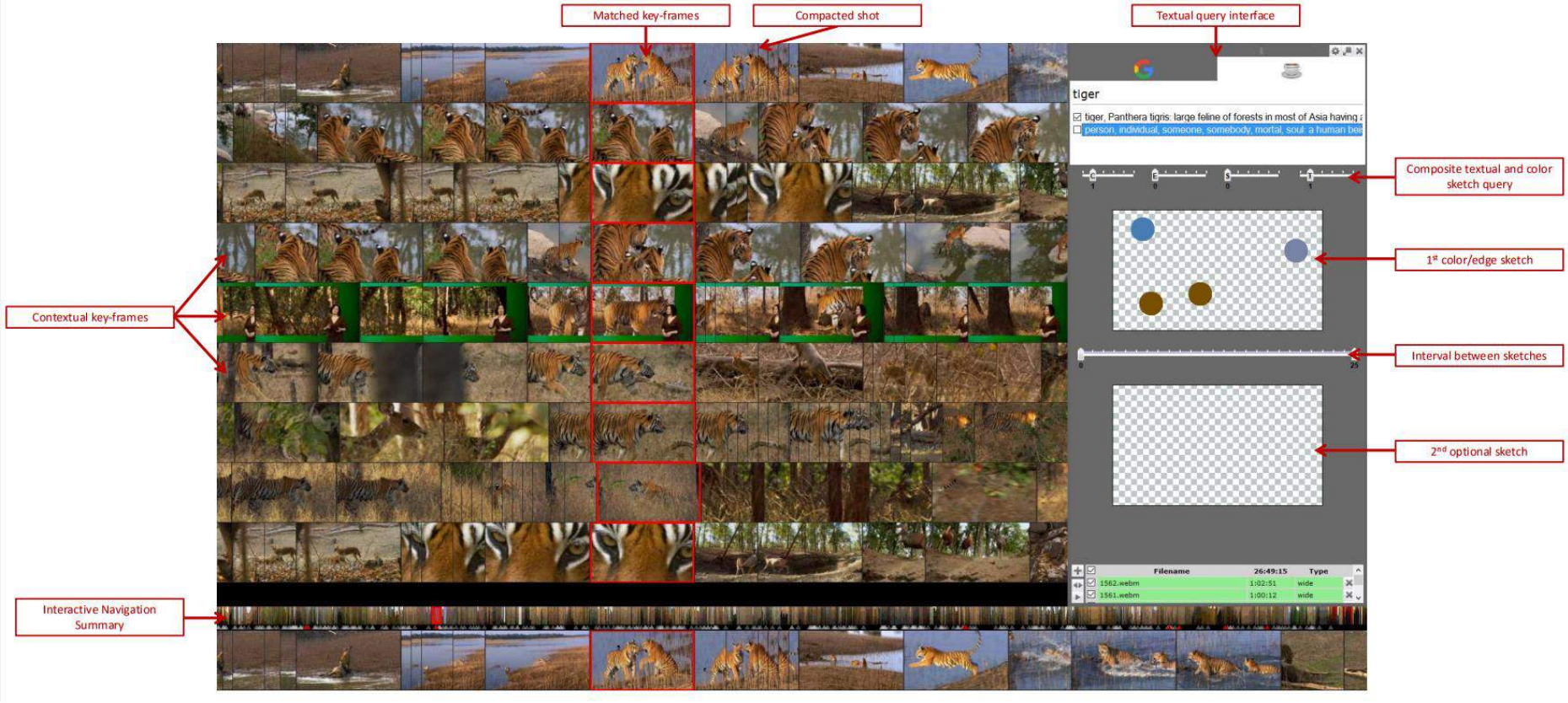


(credit: dreamstime.com)

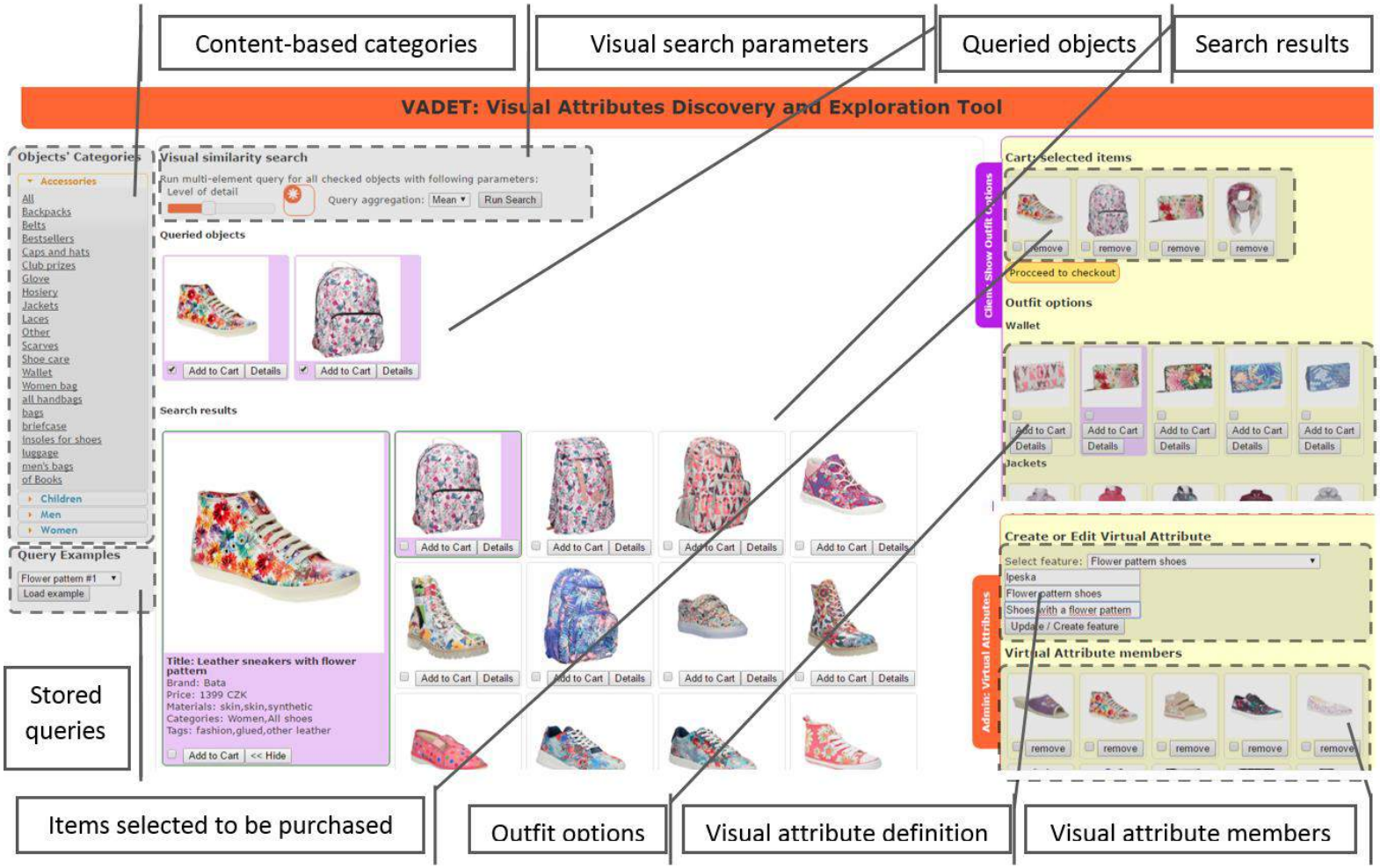
Keywords (Report | Suggest)
active animal attentive breed canine carnivore command companion cute daylight dog energentic face fur furry golden lab labby labrador male mammal outdoor pedigree pet portrait pose predator protector pure purebred rescue retriever retrieving side social stand trained watchdog yellow

Example – browsing videos

- query by sketch + keywords, borowsing



Example – querying + browsing products (photos)



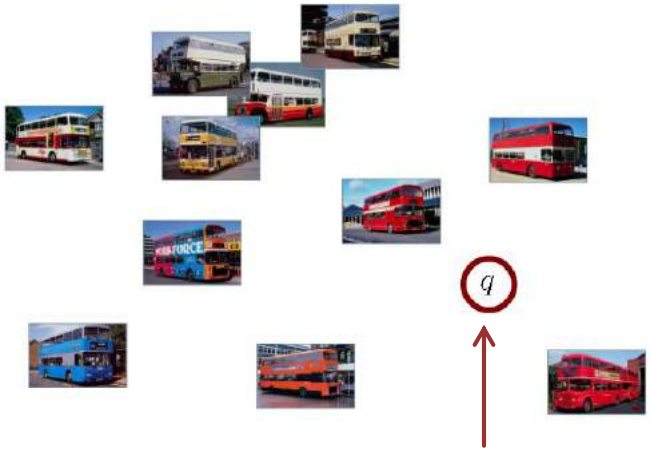
Means of retrieval – Exploration

- get an idea of the entire database content
  - aggregated browsing
  - multimedia data (visual) analytics
- multimedia exploration – complex framework involving
  - visualizing (a part/summary of) the database
  - intensive visual interaction with the user
  - underlying triggering of similarity queries based on the interaction
  - real-time processing necessary (not necessary for queries)

Means of retrieval – Exploration

(more in a Prezi presentation)

- exploration queries
  - zoom in/out
  - slide, rotate
  - add/remove/move
- similarity-based layouts
  - preserving the similarity in the visualization space
- suitable for modern GUI
  - smart phones & tablets with multi-touch displays

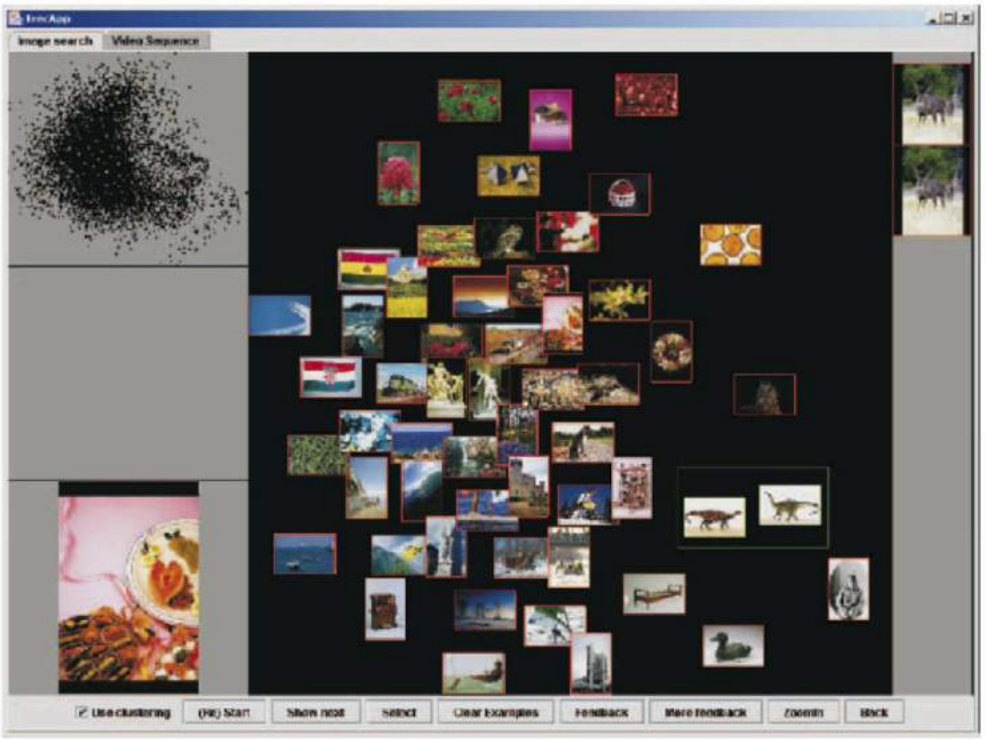


what's behind there?

$$\phi(o) = \sum_{o_i \in X} w_i \cdot (\delta_{o,o_i} - d_{q,X(o_i)})^2$$

Example – image exploration

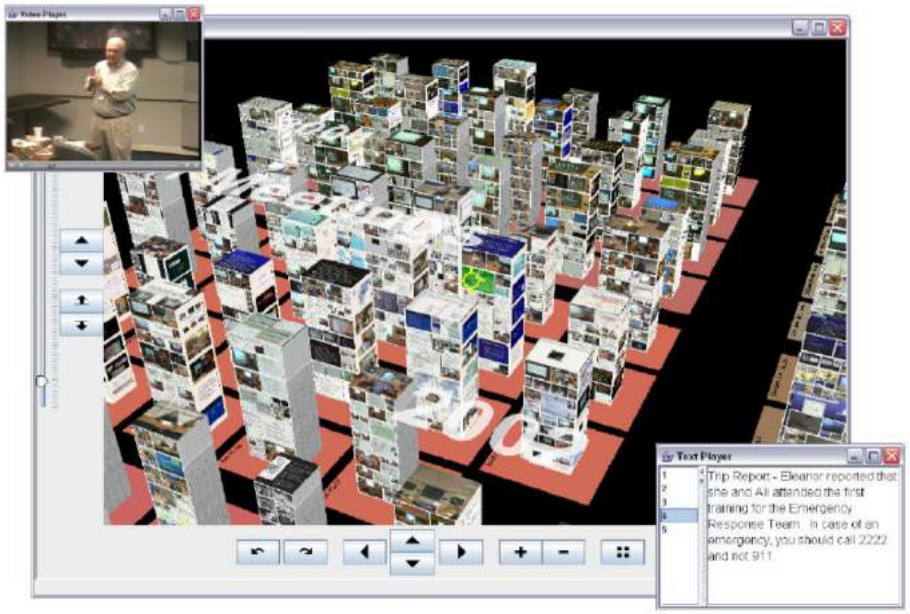
Exploration using similarity-based layouts:



Giang P. Nguyen, Marcel Worring: Interactive access to large image collections using similarity-based visualization. J. Vis. Lang. Comput. 19(2): 203-224 (2008)

Example – video & image exploration

Exploration using other visualization layouts:

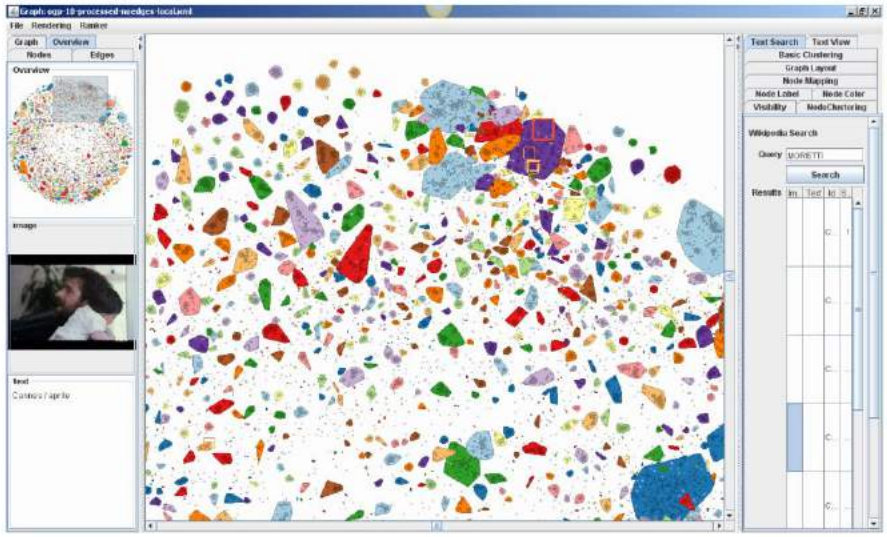


Patrick Chiu, Andreas Girgensohn, Surapong Lertsithichai, Wolfgang Polak, Frank M. Shipman III: MediaMetro: browsing multimedia document collections with a 3D city metaphor. ACM Multimedia 2005: 213-214

Gerald Schaefer: A next generation browsing environment for large image repositories. Multimedia Tools Appl. 47(1): 105-120 (2010)

Example – image exploration

Kan Ren, Risto Sarvas, Janko Calic: Interactive search and browsing interface for large-scale visual repositories. Multimedia Tools Appl. 49(3): 513-528 (2010)



Marie-Luce Viaud, Olivier Buisson, Agnès Saulnier, Clement Guenais: Video exploration: from multimedia content analysis to interactive visualization. ACM Multimedia 2010: 1311-1314

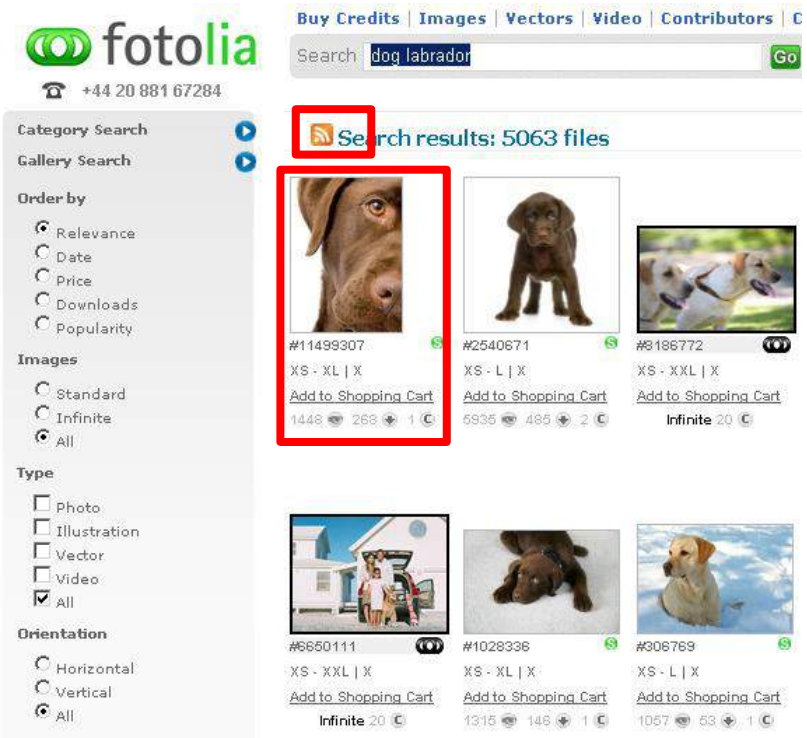
Means of retrieval – Recommendation

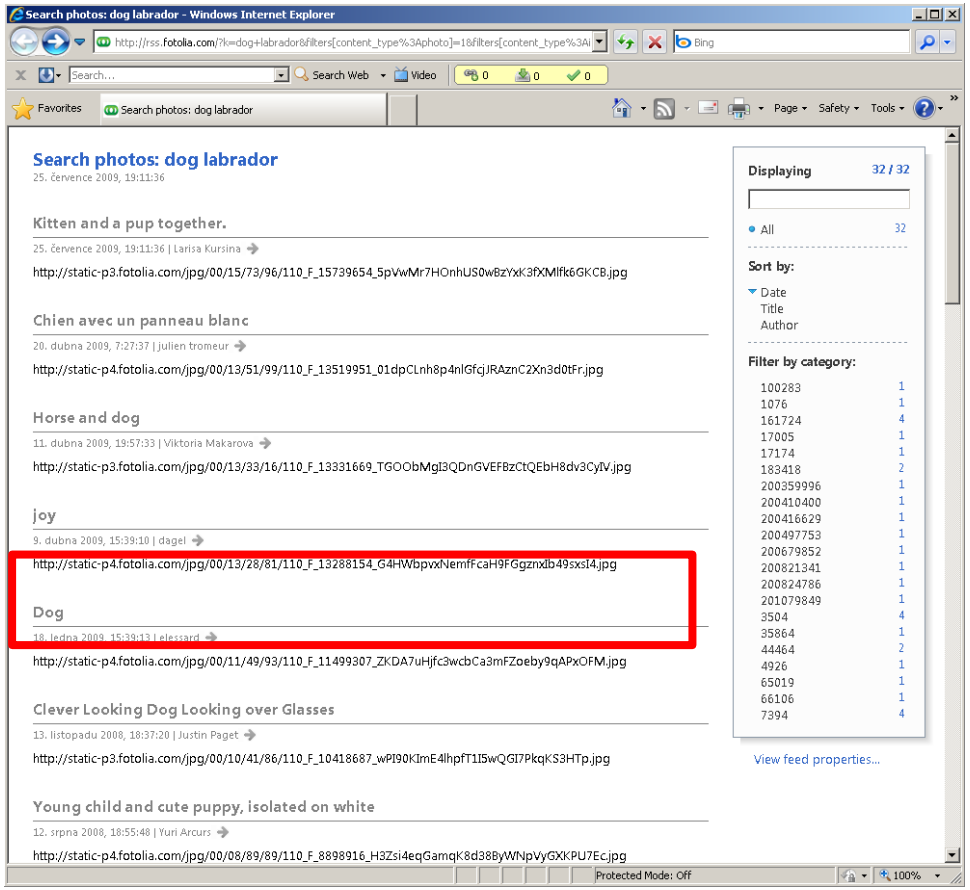
- explicit = based on static request
  - query (aka database view), subscription
- implicit = based on user behavior/context
  - user preferences, profile, search history, click streams, collaborative filtering, etc.
  - context of (web) applications, social networks
- recommendation result
  - dynamic, changing in time
  - application-specific

Example – explicit recommending

- RSS channel + reader
  - filtering/recommendation tool
- keyword query

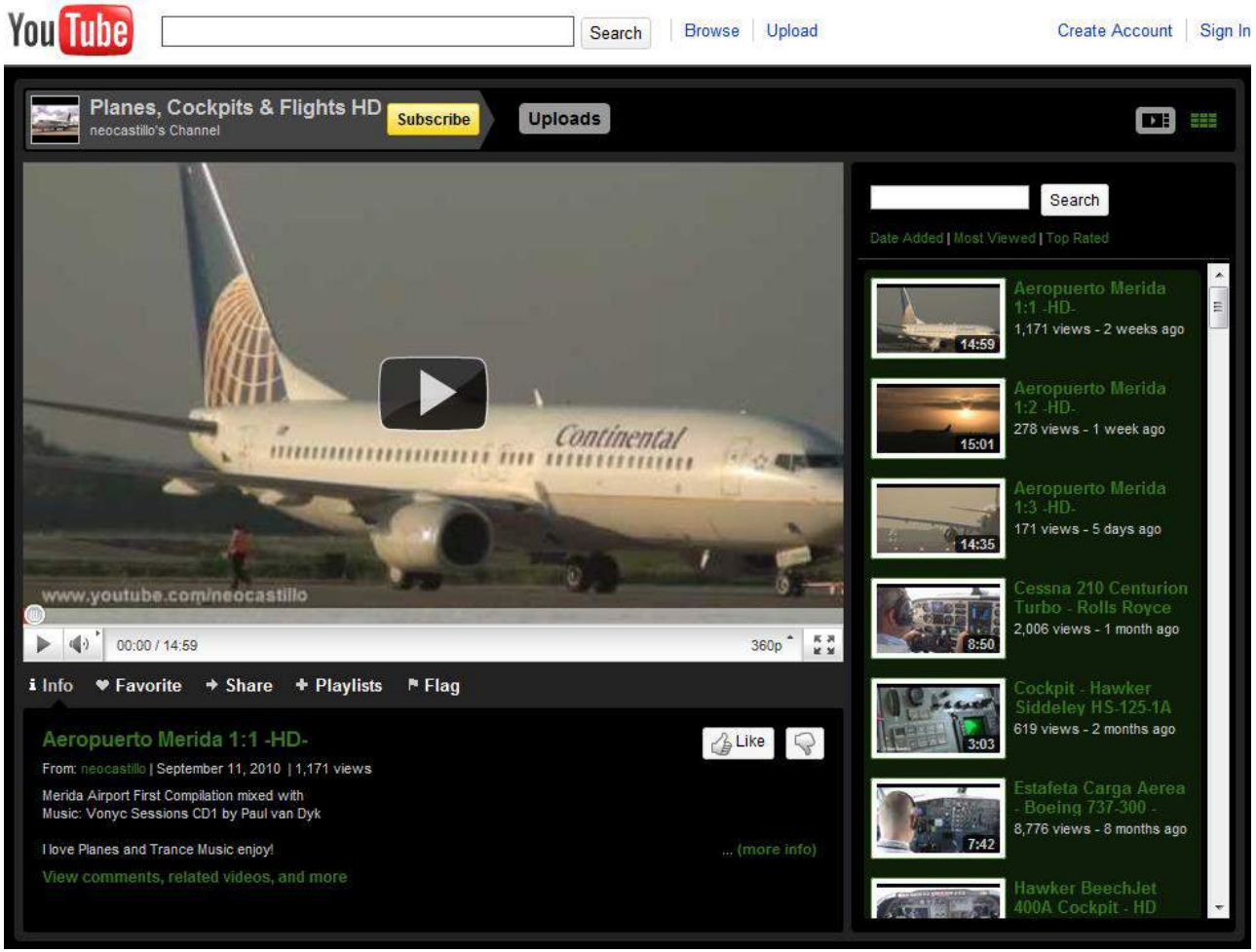
(credit: fotolia.com)





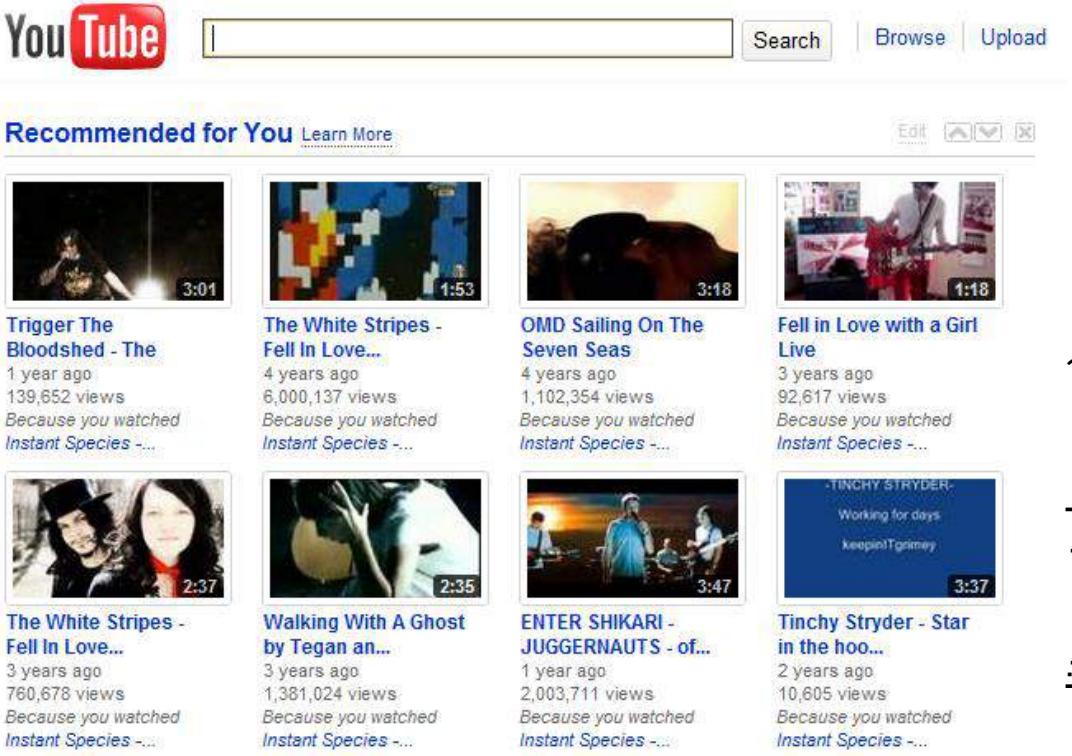
Example – explicit recommending

- subscription to a "channel"
- membership in a group
- etc.



Example – implicit recommending

- implicit recommending as an automatic query based on user's previous searches
  - history of viewing content serves as relevance feedback

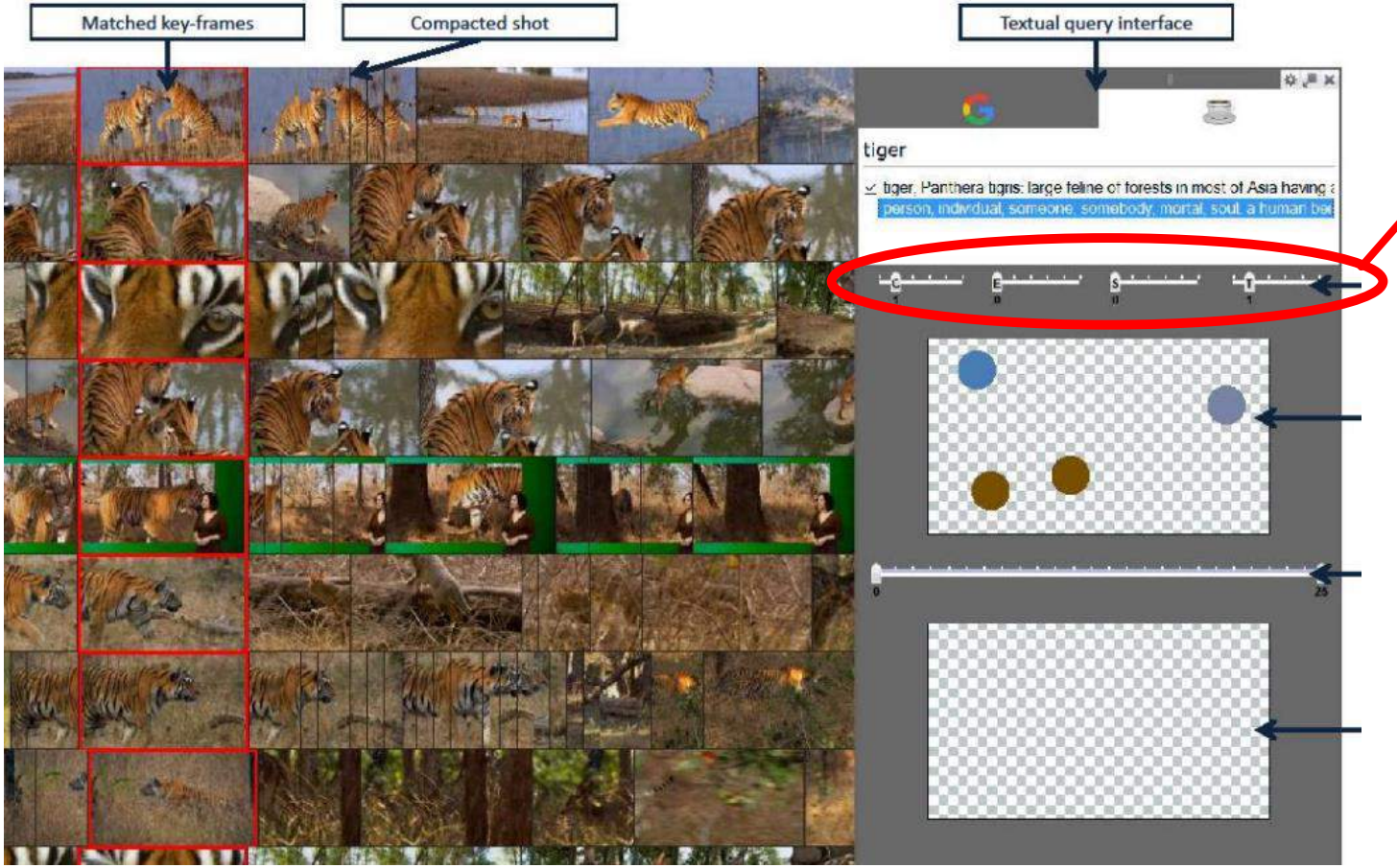


(credit: youtube.com)

Relevance feedback

- 3 steps
  - initial set of objects (by query or filter)
  - user relevance feedback
    - expert fine-tuning
      - user directly adjusts parameters (weights) of a retrieval model
    - non-expert user feedback
      - explicit - user directly selects positive and/or negative examples
      - implicit - user gives more attention to some objects (positive examples)
  - adjusted query or filter
- several iterations
  - kind of browsing
  - synergy with recommender systems

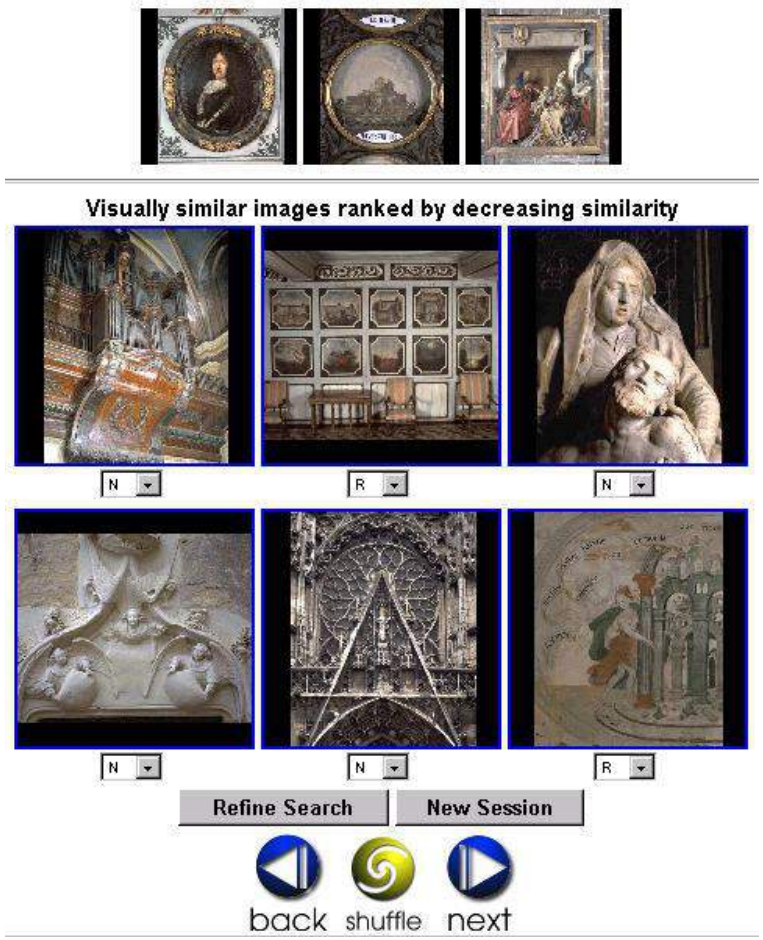
Example – expert fine-tuning

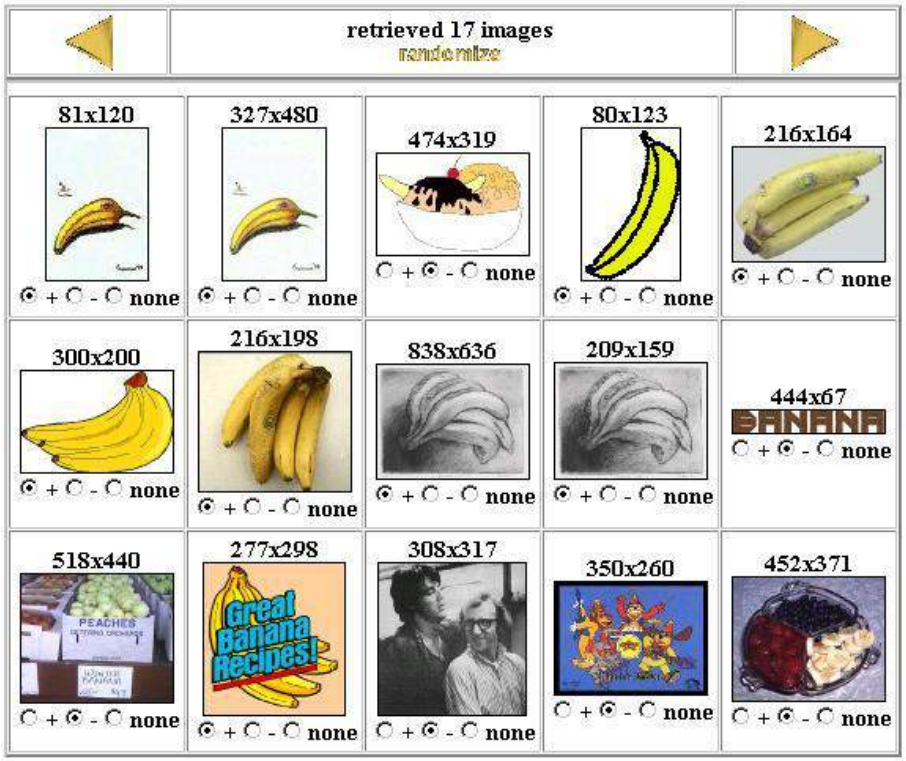


user-specified weights:

Color layout
Edge histogram
Semantic features
Text

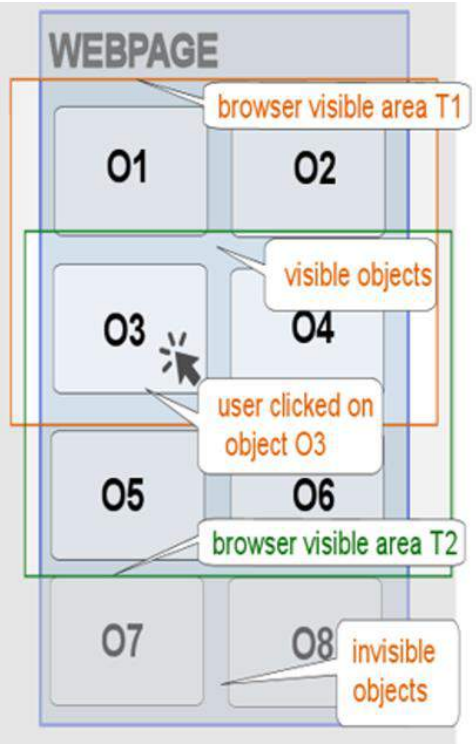
Example – explicit user feedback

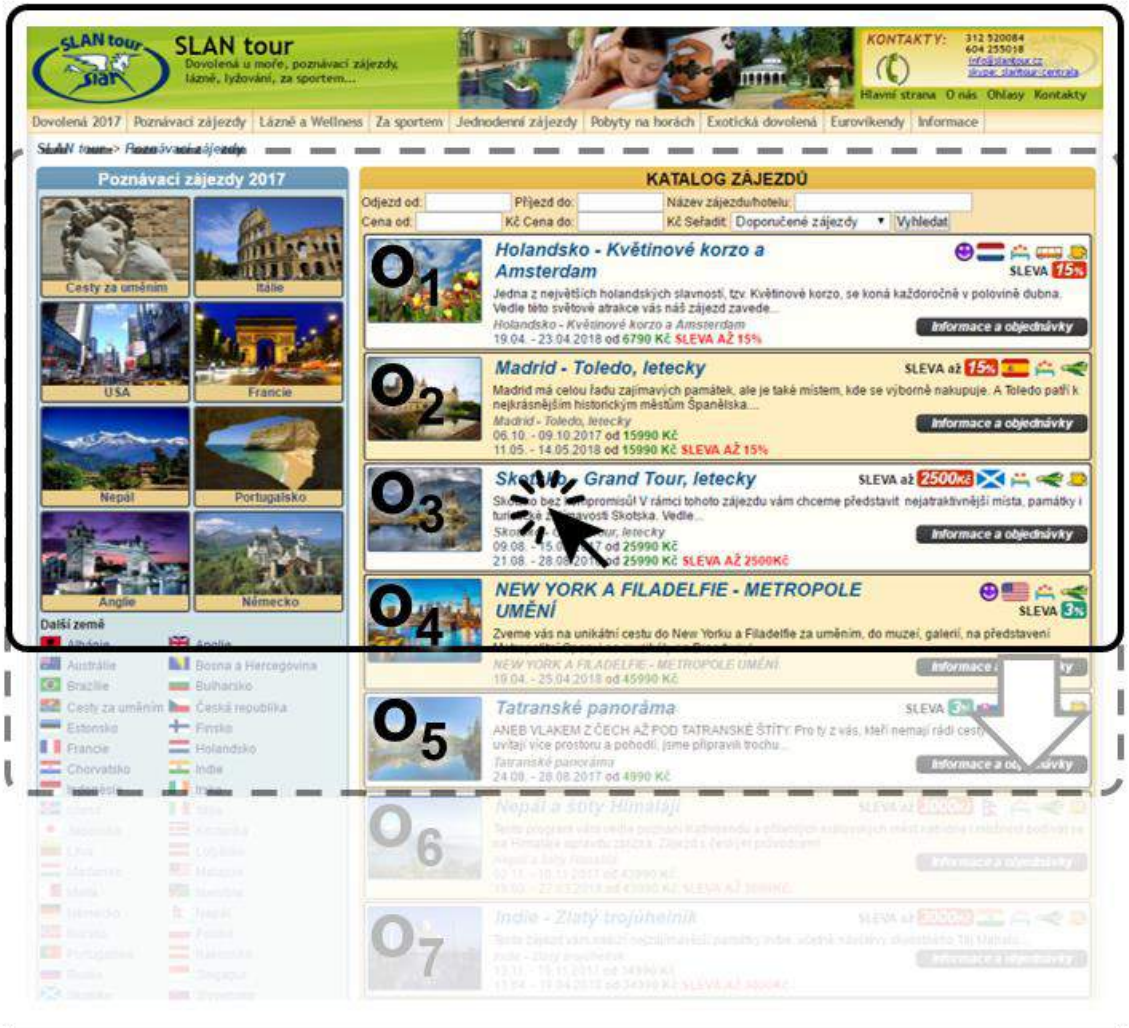




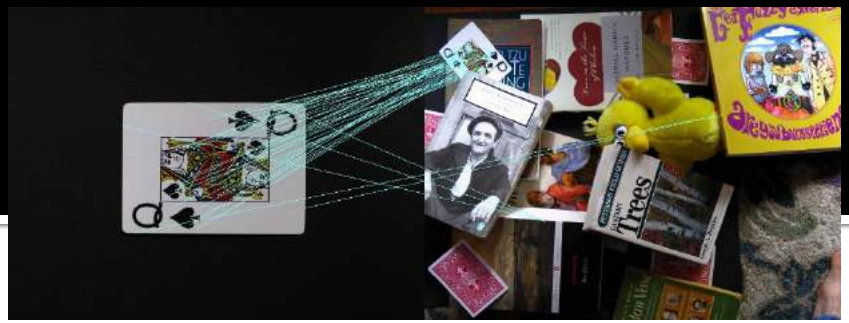
(credit: surfimage, webseek)

Example – implicit user feedback

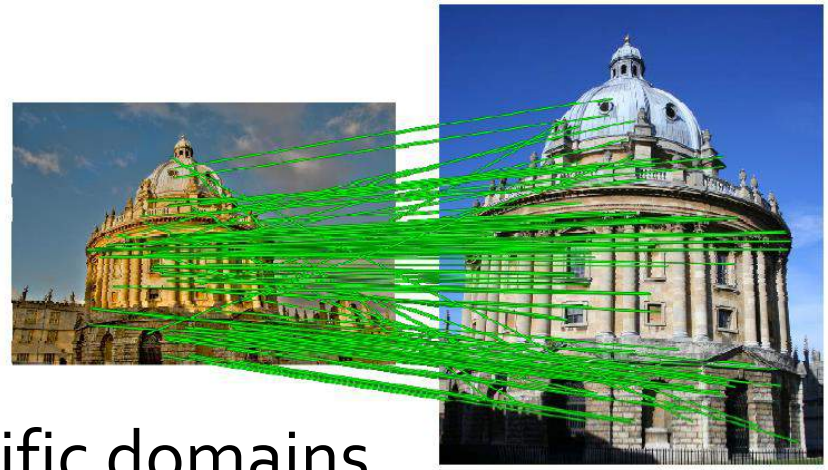




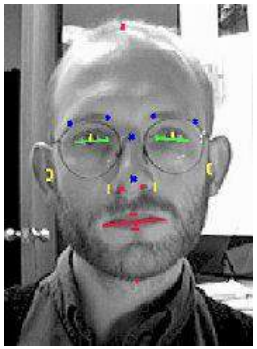
Semantic gap

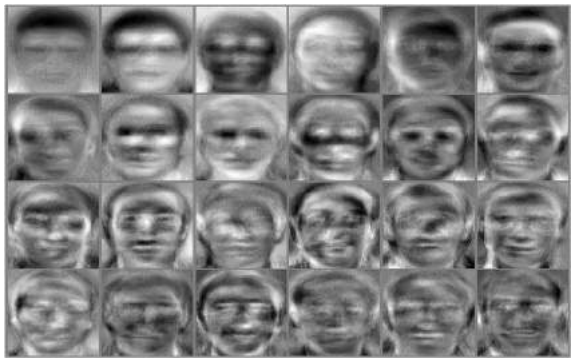


- local features are applicable for limited set of domains
  - e.g., SIFT, SURF for visual pattern matching (rigid object matching)



- higher-level features in specific domains
  - e.g., face descriptor





- or general using deep CNN
  - huge labeled data needed
  - learned, not explainable (yet)

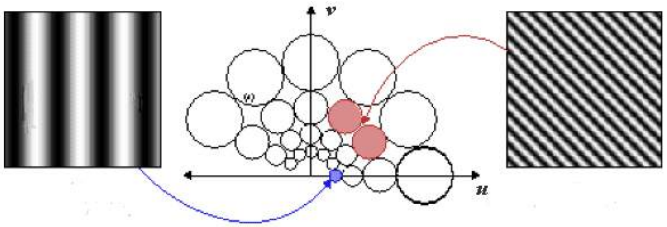
Semantic gap

- general low-level visual features do not capture the semantics at all

- example
  - color layout
  - texture
  - shape

Semantic gap

- general low-level visual features do not capture the semantics at all
- why?
  - designed to describe any image
  - static visual model
    - not suitable for adaptation based on user feedback/classification
    - e.g., histogram of colors, histogram of texture gradients



Semantic gap

- consequence of semantic gap:

  limited retrieval effectiveness

Retrieval effectiveness

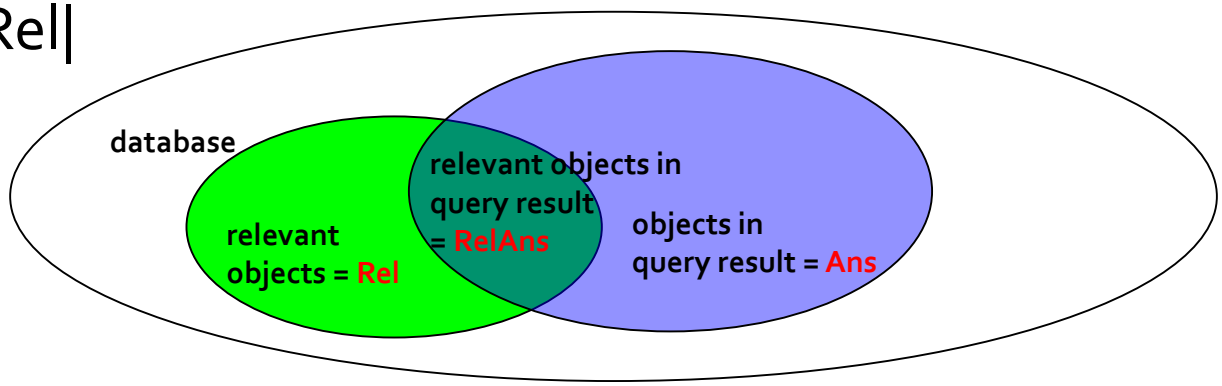
- quality of retrieval system
  - effectiveness = the measure of user satisfaction with the result
    - based on determining relevant objects in the query result
- relevant objects
  - given a particular query and database, relevant objects should be within the query result of a examined system
  - relevancy of an object could be determined when
    - annotated collection is used
    - referential retrieval system is used
- extrapolation assumption
  - a system that is effective on an evaluated collection should be effective also on an unknown collection

Retrieval effectiveness

- subjective relevance
  - collection annotated by human experts
  - golden standard, tightly connected to users
- relative (objective) relevance
  - relative comparison of two systems, one referential
  - transitively also subjective
    - given a subjective evaluation of the referential system
    - just a lower bound, not directly connected to users

Precision and recall

- two complementary measures
  - **precision** = probability that an object in the result is relevant
  - **recall** = probability that a relevant object is in the result
- formally
  - P = |RelAns| / |Ans|
  - R = |RelAns| / |Rel|



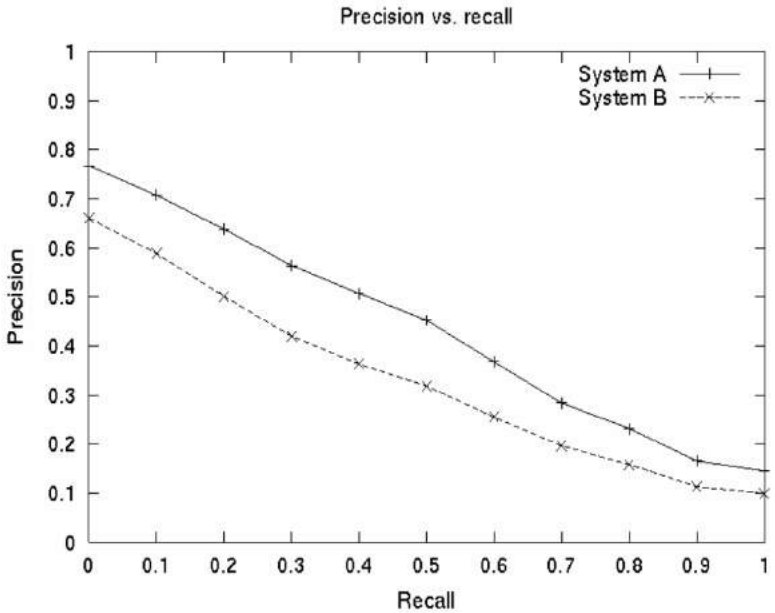
Relative precision/recall

- let QR_{ref} is the desired query result of a referential system
  - e.g., directly desired answer (annotated queries)
- let QRsys is query result obtained by the evaluated system

- **false alarm** (false hit)
  - an object $Oi \in QR_{sys}$ and $Oi \notin QR_{ref}$
    - i.e., should not be retrieved but it was
  - the more false alarms, the lower **relative precision**
- **false dismissal** (false drop)
  - an object $Oi \notin QR_{sys}$ and $Oi \in QR_{ref}$
    - i.e., should be retrieved but it was not
  - the more false dismissals, the lower **relative recall**

P-R curve

- the precision-recall curve shows the dependence of precision and recall
  - gives more complex characteristics of the system
  - enlarging query result is considered (not just single result)
- 11 standard levels of recall
  1) let's have a ranked query result
  2) enlarging the query result till a certain level of recall is reached
  3) precision is computed
  4) goto 2)
- typically a compromise



Alternative measures

- precision, recall and P-R are defined for a single query result (fixed or enlarging)
- F-score = 2*P*R/(P+R)
  - harmonic average of precision and recall
  - single-query measure
- mean precision
  - averaged precision over all levels of recall
  - proportional to the surface below P-R curve
  - single-query measure

Alternative measures

- R-precision
  - the precision when considering the first |Rel| objects in the ranked result
  - correlated to the mean precision
  - single-query measure
- Bull-Eye percentage
  - the recall when considering the first |Rel|/2 objects in the ranked result
  - single-query measure
- Mean average precision
  - the mean precision averaged over multiple queries
  - standard single-value measure

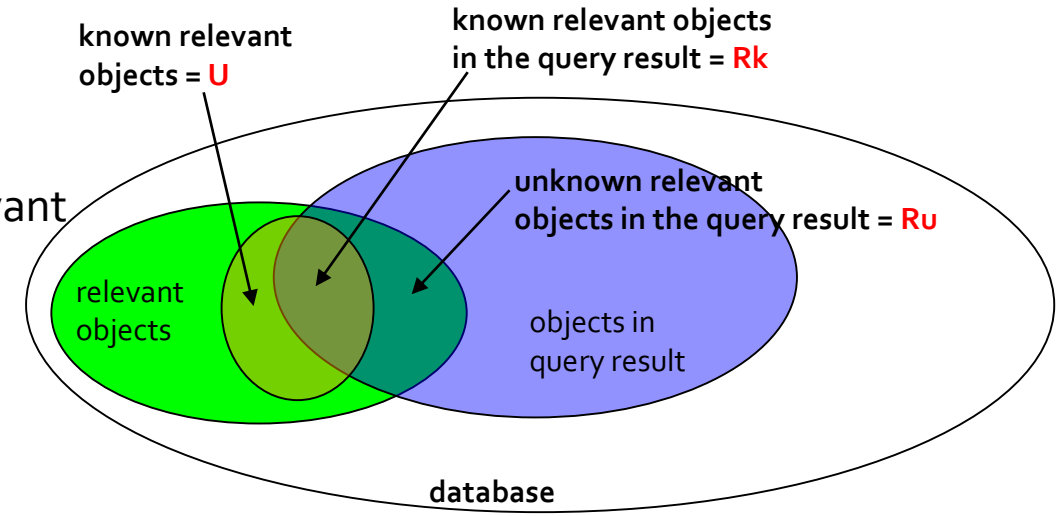
Alternative measures

- Harmonic mean and E-measure
  - $F(j) = 2/(1/R(j) + 1/P(j))$
  - $E(j) = 1 - ((1+b^2)/(b^2/R(j) + 1/P(j)))$
    - $R(j)$ = recall on the j-th object in the ranked query result
    - $P(j)$ = precision on the j-th object in the ranked query result
    - b is user-defined parameter

Alternative measures

- all the mentioned effectiveness measures were aimed at one-shot queries
- iterative retrieval processes (like browsing) require more appropriate measures
  - previous query results should be taken into account
  - known object = object already seen (in previous query result)

- Coverage and Novelty
  - Coverage = |Rk|/|U|
    - proportion of known relevant objects in the query result
  - Novelty = |Ru|/(Ru + Rk)
    - proportion of unknown relevant objects in the query result



Benchmarking

- Benchmarking the effectiveness of retrieval systems
  = establishing a platform for testing retrieval systems
- 3 components needed
  - database(s)
  - set of queries
  - for each query a set of relevant objects is selected (problems with subjectivity)

- ILSVRC (ImageNet Large Scale Visual Recognition Challenge), since 2010
  - object detection and image classification at large scale

- TREC trec.nist.gov (since r. 1992)
  - texts
  - video (TRECVID) www-nlpir.nist.gov/projects/trecvid
    - since 2003, content-based video retrieval
  - genomic databases ir.ohsu.edu/genomics

- Benchathlon www.benchathlon.net
  - content-based image retrieval

Benchmarking

- Cystic Fibrosis Database
  (www.sims.berkeley.edu/~hearst/irbook/cfc.html)
  - 1239 text documents about cystic fibrosis
  - 100 annotated queries
  - relevance scores 0..9 assigned by experts
- Princeton Shape Benchmark
  (shape.cs.princeton.edu/benchmark/index.cgi)
  - database and tools for 3D retrieval
  - 1814 3D models
  - training and testing classification of 3D objects (2x 90 classes and 907 models)

course:
Multimedia Retrieval

lecture 3:

Text-based and bag-of-features models

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

- text-based retrieval model
  - boolean model
  - vector model, LSI
  - implementation by inverted index
- automatic annotation
- bag of features (BoF)
- large-scale image search
  - VLAD
- word2vec

Text-based retrieval of multimedia

- multimedia object associated with text (document)
  - surrounding text on a web page
  - user annotation (set of keywords)
- visual words
  - content-based retrieval reusing the full-text search model/implement.
  - automated annotation by "visual words" = local visual categories

How To Guarantee Worker Safety

There are many ways you can contribute to improving worker safety on a construction site. You certainly do not want the next accident to happen on your worksite and as a result, get tied up in legal matters and need to pay a significant amount of compensation to the victim. Furthermore, you do not want to get fined or even be sent to prison for safety flaws on your worksite. This is what can happen if you do not comply with Occupational Health & Safety legislation.

Here are 2 great ways to go about improving your **Construction Health And Safety**.

- Checklist – One of the simplest ways to ensure that site safety is up to scratch is to utilise a Construction Safety Checklist. This simply involves ticking off a list of items on a list that contribute to adequate and sufficient safety on your site. The key is finding a comprehensive checklist so that all safety aspects are covered.
- Template – Another common way of covering your bases when it comes to Construction Safety is by using a Safety Plan Template. The refers to using pre-written and pre-structured safety plan where all you have to do is fill in the blanks with information regarding your own project. The key here is finding a template that complies with current legislation and is flexible enough to be used on any construction site.

Text-based retrieval

- input document = free text
- document preprocessing
  - remove unwanted characters/markup
    - HTML tags, punctuation, numbers, etc.
  - break into **terms** (words) on whitespace
  - simplify terms (stemming) – **index terms**
    - fishing, fished, fish, fisher → **fish**
  - remove **stop words**
    - is, a, the, was, because, …

Text-based retrieval

- collection = set of all documents
  - every document $\boldsymbol{d_j}$ has an id
  - **n** documents in collection
- vocabulary = set of all terms in collection
  - every term $\boldsymbol{t_i}$ has an id
  - typically **m** = 10^4-10^5 terms (English stems)
- document
  - typically 10^2-10^3 unique terms (i.e., below 1% of vocabulary)
- query
  - query document
  - set of keywords (extremely sparse, up to 10)

Boolean model

- document = (unordered) set of terms
  - set representation $\boldsymbol{d_j} = \{t_{i1}, t_{i2}, \ldots, t_{ik}\}$
    - set of term ids
  - binary vector representation $\boldsymbol{d_j} = < 0|1, 0|1, \ldots 0|1 >$
    - bitmap, signature
- collection
  - set of document sets
    $C = \{d_1, d_2, \ldots d_n\}$

Boolean model, matrix view

- could be modeled as a sparse binary matrix
  - more than 99% zeros

$$
\begin{array}{c|cccc}
 & t_1 & t_2 & \ldots & t_m \\
\hline
d_1 & 0|1 & 0|1 & \ldots & 0|1 \\
d_2 & 0|1 & 0|1 & \ldots & 0|1 \\
\vdots & \vdots & \vdots & & \vdots \\
\vdots & \vdots & \vdots & & \vdots \\
d_n & 0|1 & 0|1 & \ldots & 0|1
\end{array}
$$

Boolean model, querying

- Query: boolean expressions
  - terms connected by AND, OR, and NOT (+ brackets)
  - example: *(mountain OR forest) AND NOT nature*
- Answer: set of document ids
  - particular document in the collection relevant or not
  - no ranking on documents
  - no partial match

Boolean model, implementation

- inverted index
  - list of sorted document ids for each term (i.e., inverted)
  - searching only lists belonging to query terms
    - set operations appropriate to partial predicates
  - extremely efficient for sparse queries

	mountain	forest		nature
d_1	1	0	…	1
d_2	0	0	…	1
:	:	:		:
:	:	:		:
d_n	1	1	…	1

Query: *nature AND (NOT forest)*

mountain $\rightarrow$ d_1, d_n
forest $\rightarrow$ d_n
…
nature $\rightarrow$ d_1, d_2, d_n

Answer:
d_1, d_2

Boolean model, advantages

- popular retrieval model
  - easy to understand for simple queries
  - clean and simple formalism
- boolean models can be extended to include ranking (extended boolean model)
- reasonably efficient implementations
  - inverted index for sparse queries
  - signature index

Boolean model, disadvantages

- limited expressive power
  - AND means all, OR means any
  - difficult to express complex user requests.
- difficult to control the number of documents retrieved
  - all matched documents will be returned
- difficult to rank output
  - all matched documents logically/totally satisfy the query
- difficult to perform relevance feedback
  - how should the query be modified?

Vector model

- document = bag of terms (bag of words)
  - bag = multi-set = set allowing multiple occurrences of the same element
  - document modeled by a vector of dimension **m** (extremely high-dimensional)
- user specifies a set of desired terms with optional weights
  - user-weighted query terms

    q = < mountain(0.5); forest(0.8); nature(0.2) >
  - evenly weighted query terms

    q = < mountain; forest; nature >
  - no boolean conditions specified in the query

Vector model

- retrieval based on similarity between query and documents
- output documents are ranked according to similarity to query
- similarity based on occurrence *frequencies* of keywords in query and document
- automatic relevance feedback can be supported
  - shifting query vector
  - relevant documents "added" to query
  - irrelevant documents "subtracted" from query

Vector model, matrix view

- term-by-document matrix A
  - weights of terms in documents
  - zero = the term has no significance in the document or it doesn't occur in the document
    - sparse matrix

$$
\mathbf{A} = \begin{pmatrix}
 & t_1 & t_2 & \dots & t_m \\
d_1 & w_{11} & w_{21} & \dots & w_{m1} \\
d_2 & w_{12} & w_{22} & \dots & w_{m2} \\
\vdots & \vdots & \vdots & & \vdots \\
\vdots & \vdots & \vdots & & \vdots \\
d_n & w_{1n} & w_{2n} & \dots & w_{mn}
\end{pmatrix}
$$

Vector model, term weights

- more frequent terms in a document are more important

  f_{ij} = **frequency of term *i* in document *j***
- normalized term frequency of term *i* in document *j*

  $tf_{ij} = f_{ij} / max_j\{f_{ij}\}$
- terms present in many documents are less significant

  df_i = **document frequency of term *i***

  = **number of documents containing term *i***

  idf_i = **inverse document frequency of term *i*,**

  = $\log_2 (n/ df_i)$, where n is the total number of documents
  - logarithm used to inhibit the effect of term frequency

Vector model, term weights

- *tf-idf*, popular *weighting scheme*

$$w_{ij} = tf_{ij}\, idf_i = tf_{ij} \log_2 (n / df_i)$$

- high weight
  - a term occurring frequently in the document
  - but rarely in the rest of the collection
- advantage
  - experimentally, *tf-idf* is the best w.r.t. precision/recall
- disadvantage
  - using *idf* requires static collection for efficient query implementation (by inverted file)
- many other ways of determining term weights proposed

Vector model, term weights

Example:

Given a document **d** containing terms with given frequencies in **d** as

d = < mountain(3), forest(2), nature(1) >

Assume 10,000 documents and document frequencies of the terms

as mountain(50), forest(1300), nature(250)

Then the tf-idf weights of the terms in **d** are:

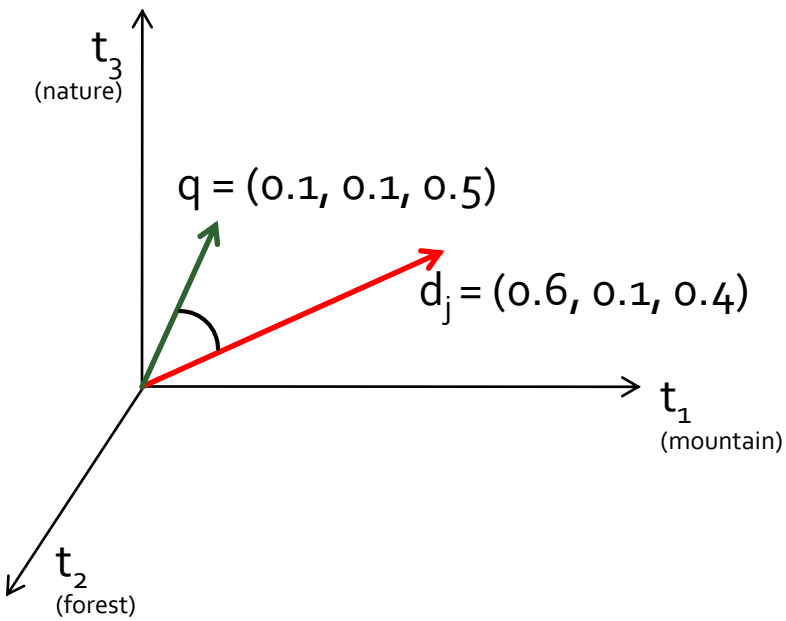
mountain:	tf = 3/3; idf = $\log_2(10000/50) = 7.6$;	**tf-idf = 7.6**
forest:	tf = 2/3; idf = $\log_2(10000/1300) = 2.9$;	**tf-idf = 2.0**
nature:	tf = 1/3; idf = $\log_2(10000/250) = 5.3$;	**tf-idf = 1.8**

Vector model, querying

- inner product
  - sim($\boldsymbol{d}_j, \boldsymbol{q}$) = $\boldsymbol{d}_j \bullet \boldsymbol{q} = \sum_{i=1}^{t} w_{ij} w_{iq}$
- cosine similarity (normed inner product)
  - CosSim($\boldsymbol{d}_j, \boldsymbol{q}$) =

$$\frac{\vec{d}_j \cdot \vec{q}}{\left|\vec{d}_j\right| \cdot \left|\vec{q}\right|} = \frac{\sum_{i=1}^{t} (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^{2} \cdot \sum_{i=1}^{t} w_{iq}^{2}}}$$

    - for binary vectors = size of intersection
    - for real vectors = cosine of the angle
- similarity query
  - query vector + similarity threshold



Vector model, implementation

- inverted index
  - list of sorted document ids for each term together with the weights $w_{i,j}$

$$
\begin{array}{cccccc}
 & \text{mountain} & \text{forest} & & \text{nature} \\
d_1 & 0.6 & 0 & \dots & 0.2 \\
d_2 & 0 & 0 & \dots & 0.1 \\
\vdots & \vdots & \vdots & & \vdots \\
\vdots & \vdots & \vdots & & \vdots \\
d_n & 0.2 & 0.5 & \dots & 0.3
\end{array}
$$

Query: <0, 0.5, 0.4>, similarity threshold = 0.05, inner product used

mountain $\rightarrow$ $d_1(0.6)$, $d_n(0.2)$

forest $\rightarrow$ $d_n(0.5)$

...

nature $\rightarrow$ $d_1(0.2)$, $d_2(0.1)$, $d_n(0.3)$,

Ranked answer: $d_n(0.37)$, $d_1(0.08)$

Vector model, LSI

- extension of the vector model
  - preprocessing of the term-by-document matrix A by singular value decomposition (SVD)

$$A = U \Sigma V^T$$

- the result is a dense *concept-by-document* matrix $\mathbf{\Sigma V^T}$
- concept bases (vectors) in **U**
  - concept = linear combination of terms
  - concepts ordered by significance
    $\rightarrow$ only ***k*** most significant concepts are considered

rank-***k*** SVD: $$A \approx U_k \Sigma_k V_k^T$$

Vector model, LSI

- dimensionality reduction
  - e.g. from hundreds of thousands to several hundreds
- LSI model reveals "latent semantics" in the collection
  - the search is term-independent, it is ***concept-based***
  - LSI model partially solves the problem of **synonymy** and **homonymy**

Vector model, LSI

- dense but low-dimensional matrix
  - dimensionality below 10^3
- inverted index not appropriate (dense query vector)

document term \	D_1	D_2	D_3	D_4	D_5
database	0	0.48	0.05	0	0.70
vector	0.23	0	0.23	0	0
index	0.43	0	0	0	0
image	0	0	0.10	0	0.54
compression	0	0	0	0	0.21
multimedia	0.12	0.52	0.62	0	0
metric	0	0	0.32	0.40	0
space	0.42	0	0	0.24	0

document concept \	D_1	D_2	D_3	D_4	D_5
$concept_1$	-0.21	0.48	-0.05	0.10	0.70
$concept_2$	0.23	0.20	-0.23	0.45	0
$concept_3$	-0.43	0.02	0.32	0.24	-0.06
$concept_4$	0.34	-0.01	0.10	0	0.54
$concept_5$	0.31	0.9	-0.78	0.52	0.21

Automatic annotation

- the text-based models are useful when annotation for multimedia data is available
- what if we have just the content?
  - content-based retrieval (further lectures)
  - automatic annotation (keyword generation)

Automatic annotation

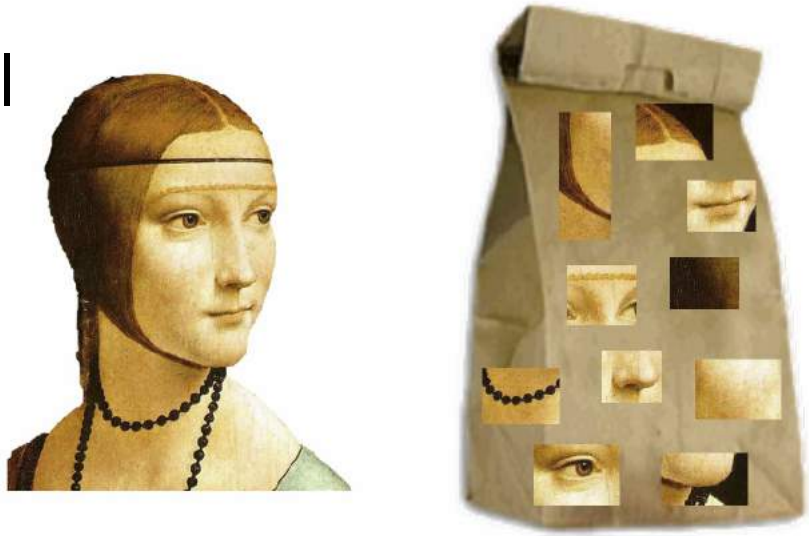
- pre-trained deep learning models
  - re-using a classification model (keywords restricted by classes)

```
---------- Prediction for ../../../examples/images/bike.jpg ----------
0.5661 - "n03792782 mountain bike, all-terrain bike, off-roader"
0.4000 - "n09193705 alp"
0.0094 - "n09468604 valley, vale"
0.0037 - "n09246464 cliff, drop, drop-off"
0.0031 - "n09472597 volcano"
0.0022 - "n03127747 crash helmet"
0.0019 - "n02835271 bicycle-built-for-two, tandem bicycle, tandem"
0.0017 - "n04509417 unicycle, monocycle"
0.0015 - "n03967562 plow, plough"
0.0013 - "n02950826 cannon"
```

- general approach
  - assume we have an annotated training collection
  - another **not annotated** (large) collection could be automatically annotated
  - take each document in this collection as a query and perform a content-based search in the retrieval collection
    - fetch the most similar document from the second collection and use (some of) the keywords for the annotation of the query document
  - now the collection is annotated and keyword-based querying is possible

Bag of features (bag of visual words)

- if the annotation is not available at all, what about create the keywords using the content?
- visual words instead of actual words
- the rest is the same as in the vector model of text retrieval
  - high-dimensional vectors (histograms)
  - cosine similarity
  - inverted index
- how to do that?



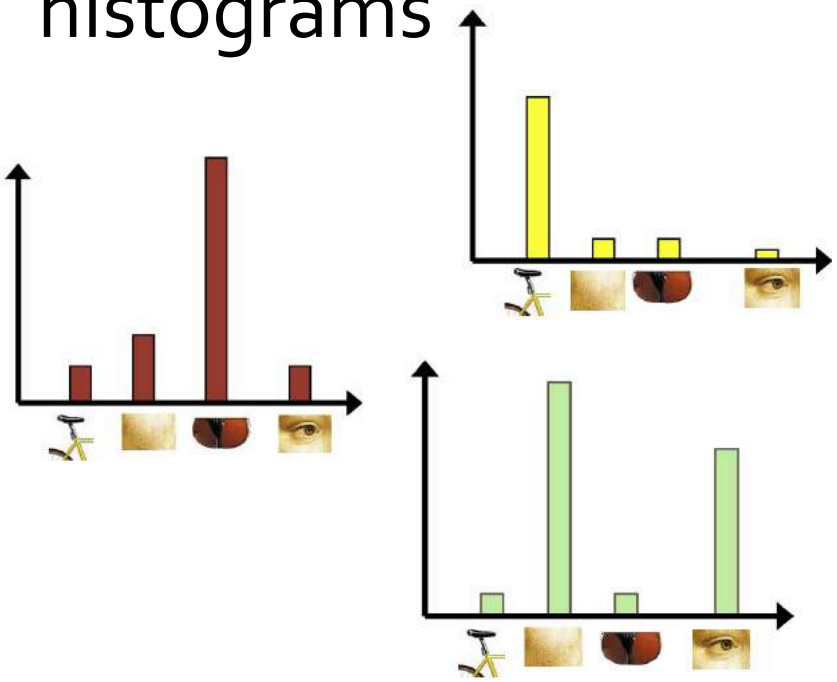
credit: Fei-Fei Li at al., 2005

Bag of features (BoF) – in short

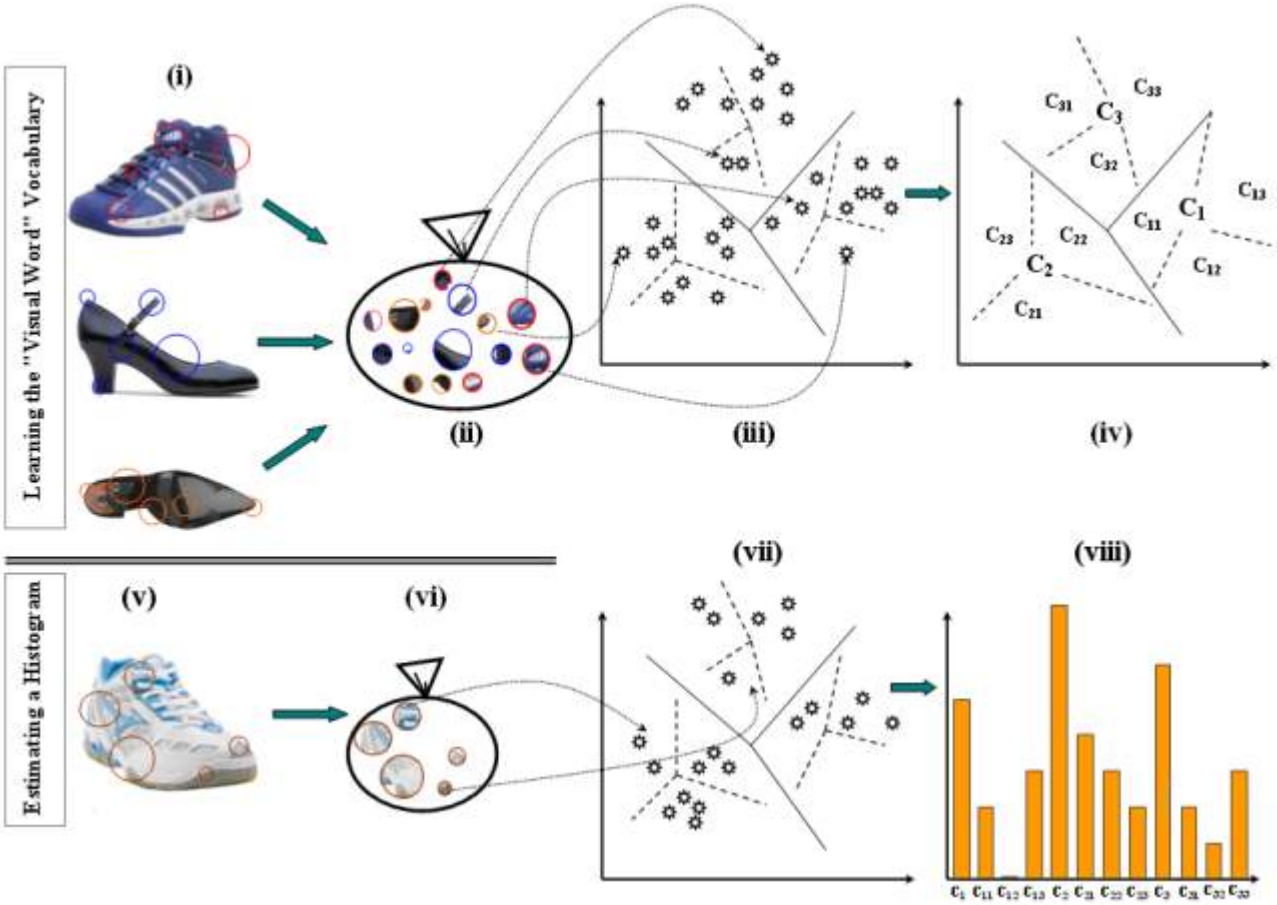
1. extract features (from many images)

2. learn visual vocabulary

3. quantize features
4. represent images by histograms



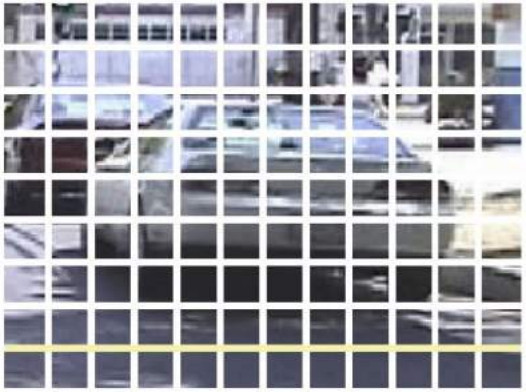
Bag of features (BoF) – in short

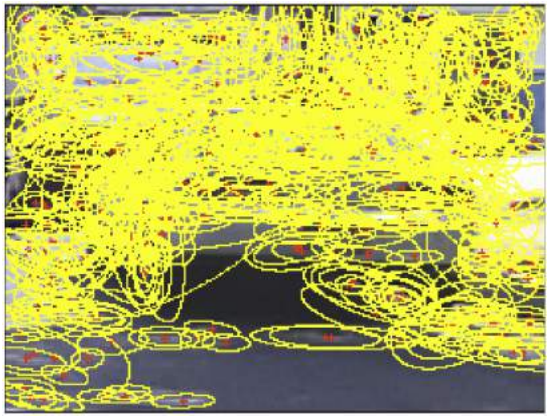


(image: http://www.codeproject.com/Articles/619039/Bag-of-Features-Descriptor-on-SIFT-Features-with-O)

Bag of features

- feature extraction
  - regular grid (ala JPEG)
  - interest points (SIFT)
  - other
    - random sampling
    - segmentation
- result is many features (vectors) per image





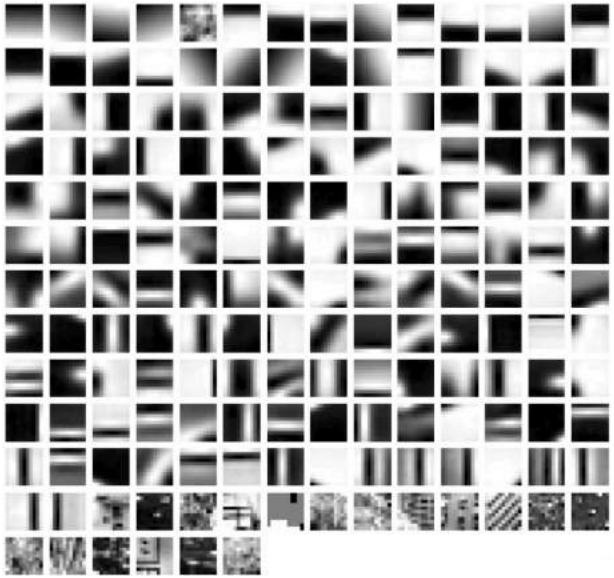
(more on feature extraction in further lectures)

Bag of features

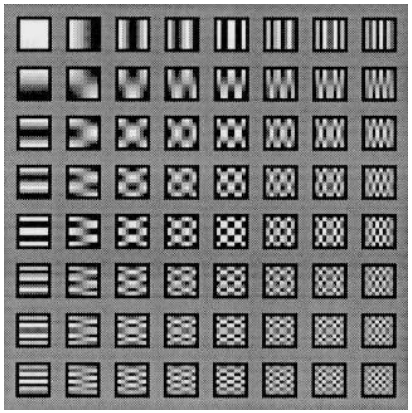
- visual vocabulary (codebook)
  - take all the feature vectors, cluster them, and use the centroids as the visual words

Bag of features

- examples of visual vocabulary
  - generic low-level local features

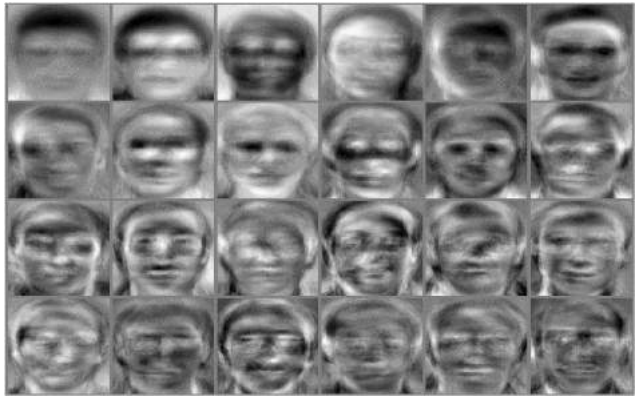


  - compare to fixed "vocabulary" used in JPEG coding

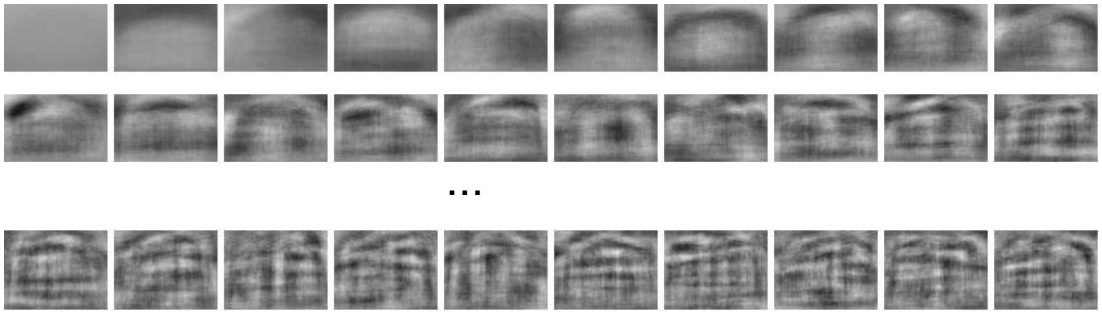


Semantic visual words

- domain-specific high-level
  - eigenfaces (PCA)



  - "eigenhouses" (SVD)



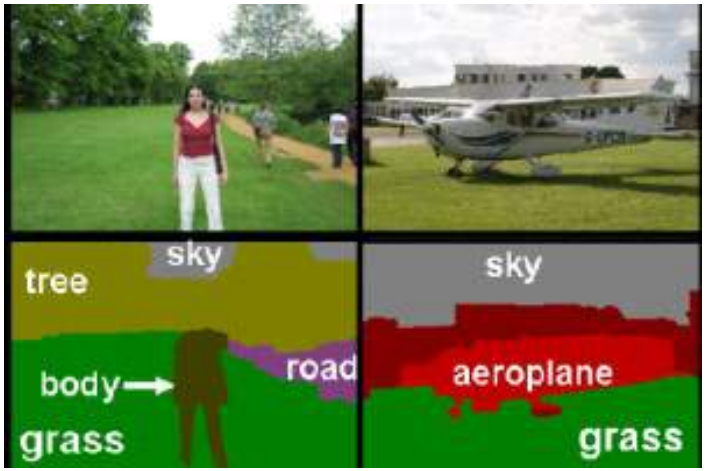
  - global match, restricted input (canonic form)

Semantic visual words

- general high-level visual words?
  - PCA, SVD don't work locally
  - SIFTs work only in limited domains
    - rigid patterns with edges (architecture, picture in picture, not query by sketch, nature, people, abstract)
  - MPEG7 and other global approaches not semantic
- decompose the problem
  - complex semantics into multiple common visual features

Semantic visual words

- often complex scene
  - unlimited camera settings, 3D projection, occlusion, abstract, sketch
- hierarchical decomposition
  - object-wise segmentation?
    - extremely difficult in general
    - good for 3D scene reconstruction, etc.
    - recent advances using machine learning
      - real words (classes) from visual segments, suitable for precise annotation
      - as in other ML applications, huge training data (e.g., DCNN models) needed



(source: http://www.cs.utexas.edu/~grauman/courses/fall2011/schedule.html)

Bag of features

- visual vocabulary (codebook)
  - vocabulary can be learned on separate training set
  - provided the training set is sufficiently representative, the vocabulary will be "universal"
- analogy to the text retrieval
  - vocabulary of keywords vs. visual words
  - vectors of terms weights vs. vectors of visual words weights
  - query text (keyword query) vs. query image

Bag of features

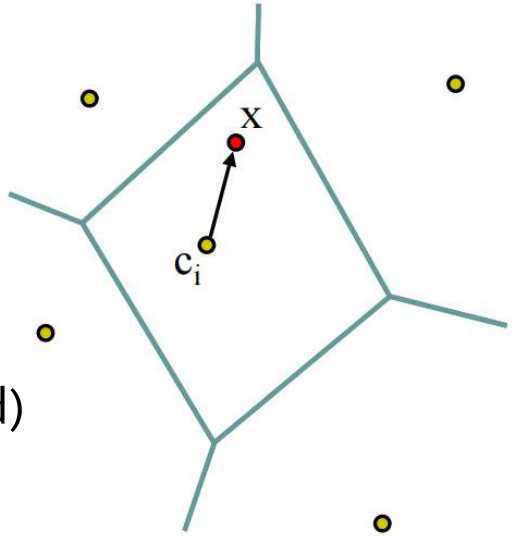
- querying
  - feature extraction from the query image
  - mapping (quantization) of features to visual words (using **similarity search** in the vocabulary, more in **next lecture**)
  - construction of the query vector (histogram)
    - also sparse, as in the text retrieval case
  - the rest is classic vector query (as in text retrival)
    - cosine similarity
    - inverted index implementation

Large-scale image search

- BoF + inverted index can handle ~ 10 millions images
  - with a limited number of descriptors per image
  - search: 2 seconds, RAM: 40GB
- web-scale = billions of images
  - with 100 M per machine
  - search: 20 seconds, RAM: 400 GB
  - not tractable
- solution: represent each image by one compressed vector
  - **VLAD**, Fisher vectors, GIST, feature signatures, etc.
  - different indexing needed (not inverted index)

VLAD

- VLAD = vector of locally aggregated descriptors
  - learning: a vector quantifier (k-means)
    - output: k centroids (visual words): $c_1,\ldots,c_i,\ldots c_k$
    - centroid c_i has dimension d
    - similar to BoF visual vocabulary construction
  - for each image
    - assign each feature x to closest center c_i
    - accumulate (sum) features per cell
      $v_i = v_i + (x - c_i)$
  - VLAD descriptor $[v_1, v_2, .., v_k]$ (dimension $D = k * d$)



- VLAD vs. BoF
  - no vocabulary used in VLAD
  - lower-dimensional dense vectors in VLAD
  - VLAD more robust to dimensionality reduction

VLAD

- evaluation* of VLAD vs. BoF: INRIA Holidays Dataset (mAP,%)
  - dimension is reduced to from D to D' dimensions with PCA

Aggregator	k	D	D'=D (no reduction)	D'=128	D'=64
BoF	1,000	1,000	41.4	44.4	43.4
BoF	20,000	20,000	44.6	45.2	44.5
BoF	200,000	200,000	54.9	43.2	41.6
VLAD	16	2,048	49.6	49.5	**49.4**
VLAD	64	8,192	52.6	**51.0**	47.7
VLAD	256	32,768	**57.5**	50.8	47.6

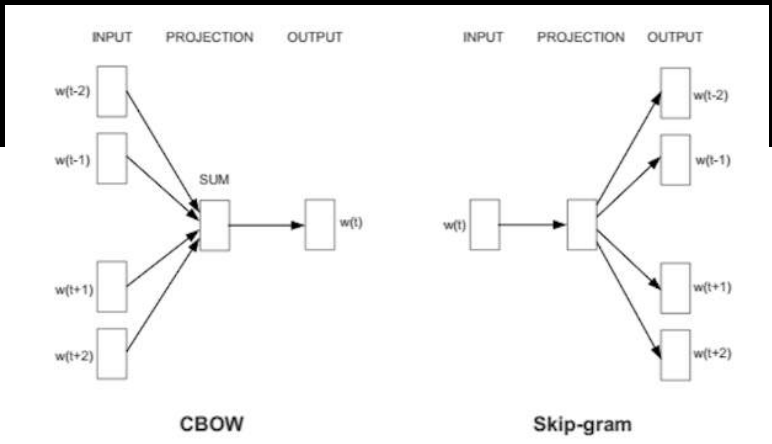
  - observations:
    - VLAD better than BoF for a given descriptor size
    - choose a small D if output dimension D' is small

*credit: slides by Cordelia Schmid, Visual Recognition and Machine Learning Summer School, 2013

Word2vec

- distributed word representations
- machine learning approach to word embeddings
  - by Tomáš Mikolov (2013), https://github.com/tmikolov/word2vec
  - extensible to genes, code, likes/follows, playlists, social media graphs and other verbal or symbolic series
  - unsupervised, just text corpus needed for training
  - useful for NLP, search, recommendation, sentiment analysis, etc.
  - 3-layer backpropagation (gradient descent) neural network

Word2vec



- two algorithms/network models
  - continuous bag of words (CBOW)
  - skip-grams
- N-gram context (window) for each (**center/focus**) word

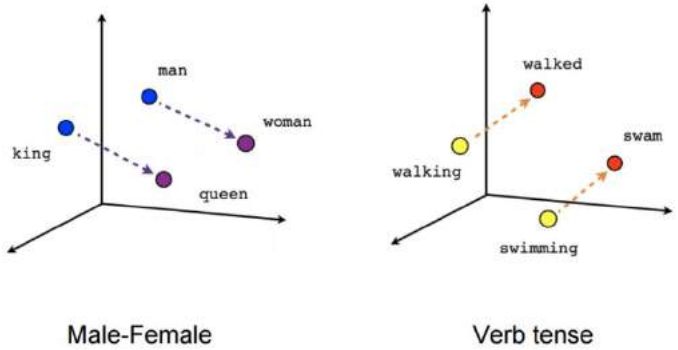
: Center Word
: Context Word

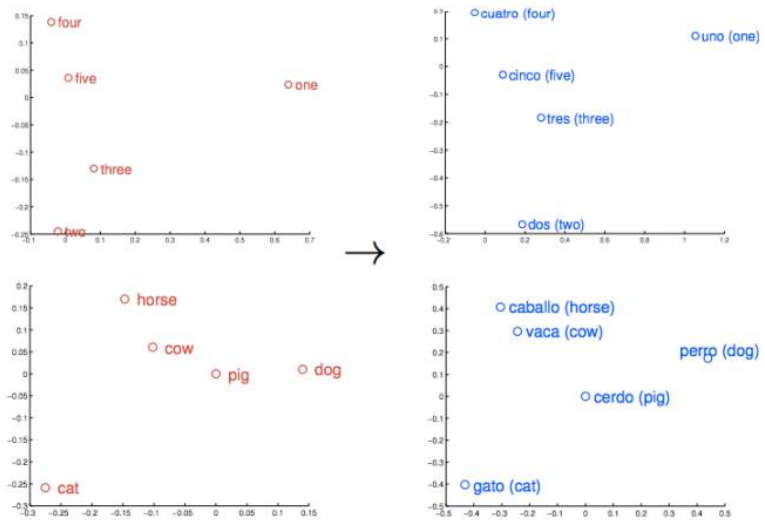
c=0 The cute cat jumps over the lazy dog.
c=1 The cute cat jumps over the lazy dog.
c=2 The cute cat jumps over the lazy dog.

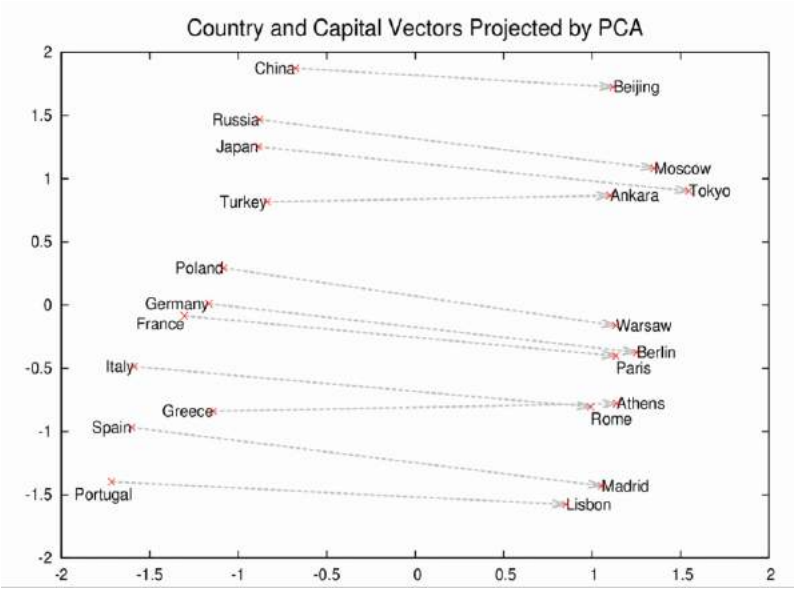
- CBOW
  - input layer: 1-of-V coding for **c** context words
    - a V-dimensional vector [0,0,0,…,1,…,0,0,0], where V is the vocabulary size
  - projection layer – continuous (target) representation of focus word
  - output layer – 1-of-V representation of focus word
- Skip-gram
  - the opposite of CBOW + order of words in context matters

Word2vec

- vector arithmetic for NLP and information retrieval
  - vec("king") – vec("man") + + vec("woman") ≈ vec("queen")







course:
Multimedia Retrieval

lecture 4:

Similarity search – models and queries

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

- the concept of similarity search
  - fundamentals
  - topological properties of similarity
  - smart descriptors vs. smart similarity
- similarity queries
  - simple queries (query "primitives")
  - advanced queries
  - re-ranking
- integrating similarity queries into SQL
  - legacy implementation
  - language extensions of SQL

The concept of similarity

- content-based (multimedia) retrieval
  - content-based annotation, then metadata search
  - direct similarity search
    - pair-wise similarity function = relevance measure
    - simple (low-level) queries
      - query-by-example
    - complex (high-level) queries
      - composition of low-level similarity queries/predicates
      - query language

(Dis)similarity space

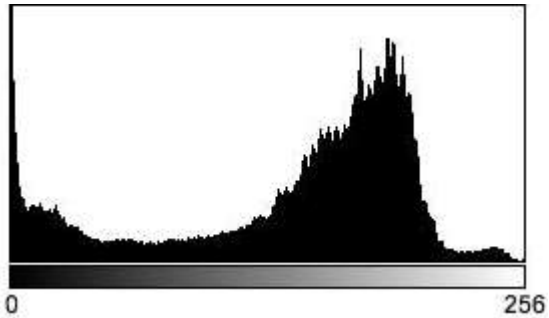
- (dis)similarity
  - similarity = higher score implies higher similarity
  - dissimilarity = distance = vice versa
- feature extraction
  - feature model + extraction procedure
  - produces descriptors of features (for multimedia objects)
- dissimilarity space
  - feature extraction $\rightarrow$ feature universe **U**
    - general set, vector space
    - database = set of descriptors $\mathbf{S} \subset \mathbf{U}$
  - (dis)similarity function $\delta: \mathbf{U} \times \mathbf{U} \rightarrow \mathbf{R}$
    - similarity
    - distance – „geometerization" of the search problem

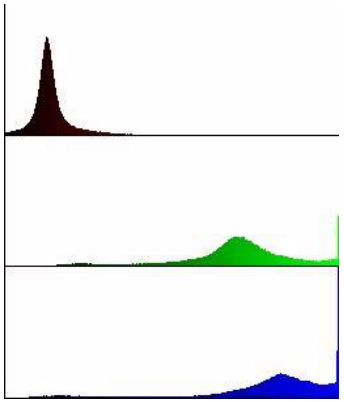
Feature extraction

- what features to extract into descriptor?
  - similarity is a part of features choice/modeling
  - descriptive features
  - relatively small number of features
  - compact representation
- geometric approach
  - narrow – vector space model + distance
  - more general – metric space model
  - general – dissimilarity (distance) space
- single- vs. multiple descriptor representation $f_{\text{single}} : \mathcal{D} \longmapsto \mathbb{U}$
  - single complex descriptor for an object
  - multiple (local) descriptors for an object $f_{\text{multi}} : \mathcal{D} \longmapsto 2^{\mathbb{U}}$

Feature extraction

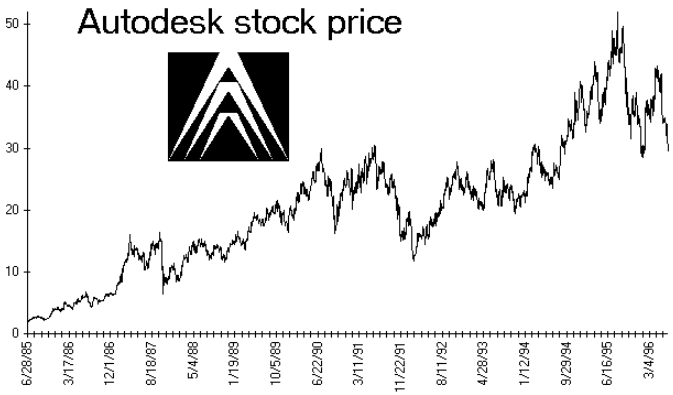
- vectors
  - correlated dimensions
    - histogram
  - independent dimensions
    - concatenation of independent features
    - numeric but also vector of general sets
  - combined
    - e.g., concatenated histograms

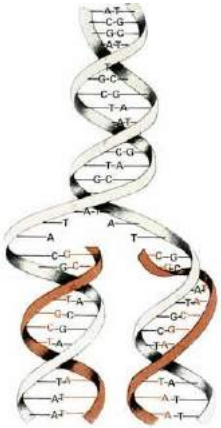




Feature extraction

- time series
  - vector of arbitrary dimension
    - dims. correlated in time/order
    - ordered set
- string/sequence
  - symbolic time series

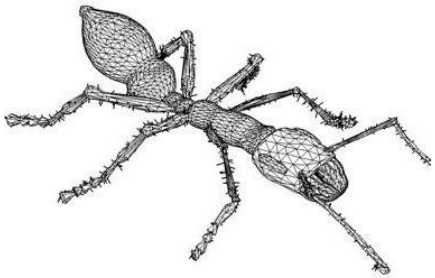


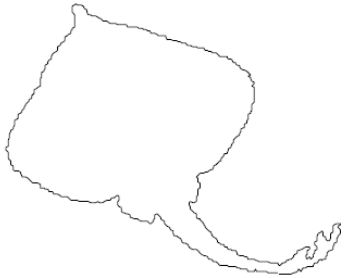


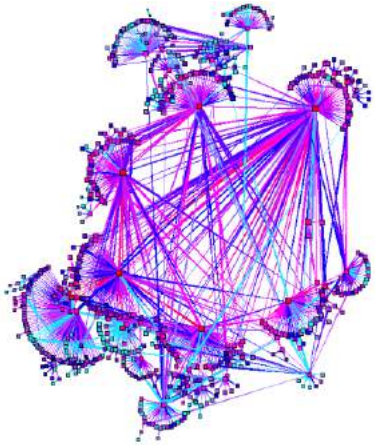
AATAGCAGCATA...

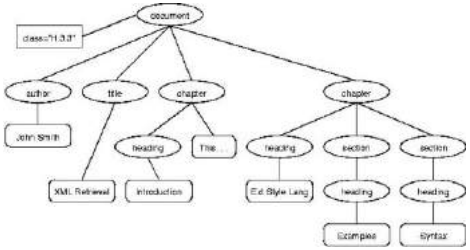
Feature extraction

- set
  - unordered set
- geometry
  - set/vector of spatial objects
- graph
  - set of edges
    + set of vertices









Properties of similarity function

- retrieval-specific properties
  - learning, adaptation, evolution
- topological properties
  - metric postulates
  - alternative (ultrametric, ptolemaic, data-specific)
- cost
  - computational complexity
  - **O**(**n**) to **O**(2^n)

Metric postulates

- motivation for metric similarity
  - well-studied in mathematics (many known metrics)
  - postulates support common assumptions on similarity
    - reflexivity – object is maximally similar to itself
    - non-negativity – two distinct objects are somehow dissimilar
    - symmetry – the direction of similarity judgement is not important
    - triangle inequality – similarity is transitive
  - allows efficient indexing and search (metric indexing)

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

Non-metric similarity

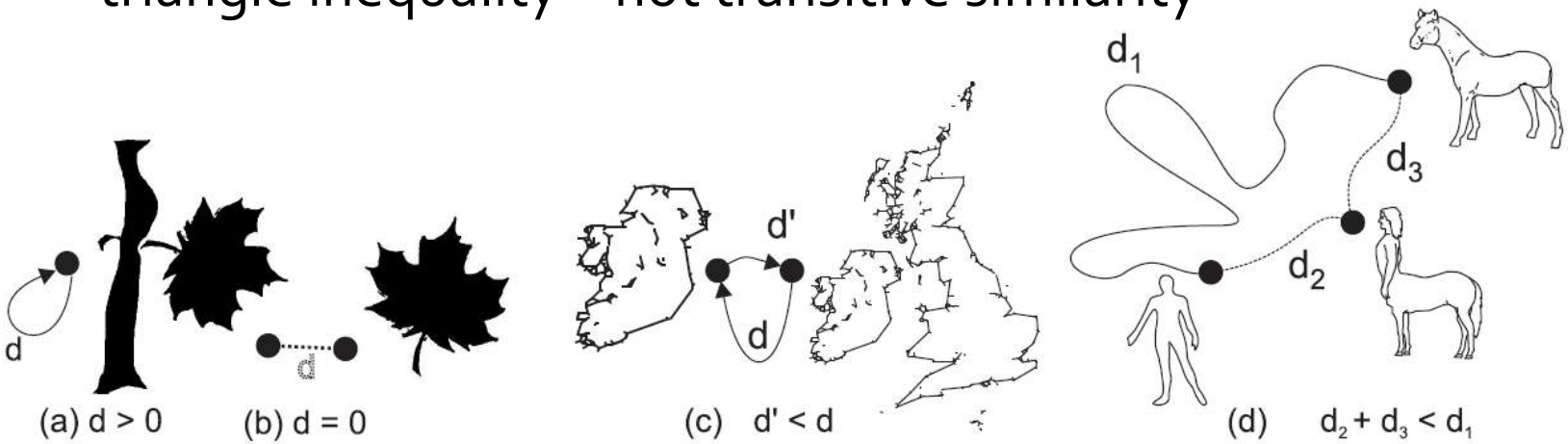
- motivation for non-metric similarity
  - simplicity of similarity modeling
  - arguments against metric postulates
  - robustness
  - black-box similarity (algorithm, HW device)
- disadvantages
  - lack of a priori information useful for indexing

Non-metric similarity = easy modeling

- the domain experts
  - practitioners, often outside computer science
  - they model similarity ad-hoc, often heuristic algorithms
  - requirements on similarity to be metric is annoying
    - trading efficiency (indexability) for effectiveness (similarity model)
- non-metric similarity – more comfort and modeling possibilities for domain experts

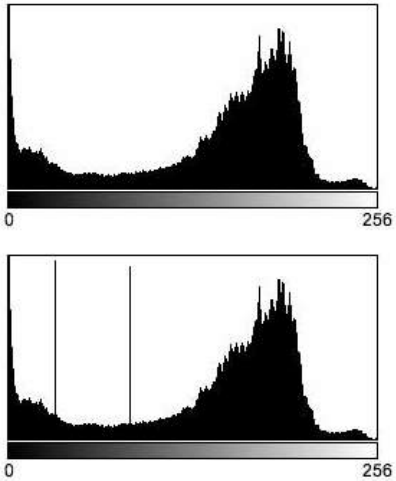
Arguments against metric similarity

- all of the metric postulates were questioned
  - reflexivity and non-negativity
    - single object could be viewed as self-dissimilar
    - two distinct object could be viewed as identical
  - symmetry – comparison direction could be important
  - triangle inequality – not transitive similarity



Robust similarity

- absence of metric postulates allows robustness
  - prefering similar portions
  - ignoring/inhibiting noise



Smart descriptors vs. distances

- where to put the "smart" retrieval logic?
  - high-level descriptor + simple distance
    - feature extraction does most of the logic
    - assumed canonized representation of features is possible
    - cheaper similarity - **O(n)**
  - low-level descriptor + complex distance
    - similarity aggregates relevant information at query time, i.e., information that cannot be extracted beforehand
    - more flexible but more expensive similarity $\geq$ **O(n^2)**

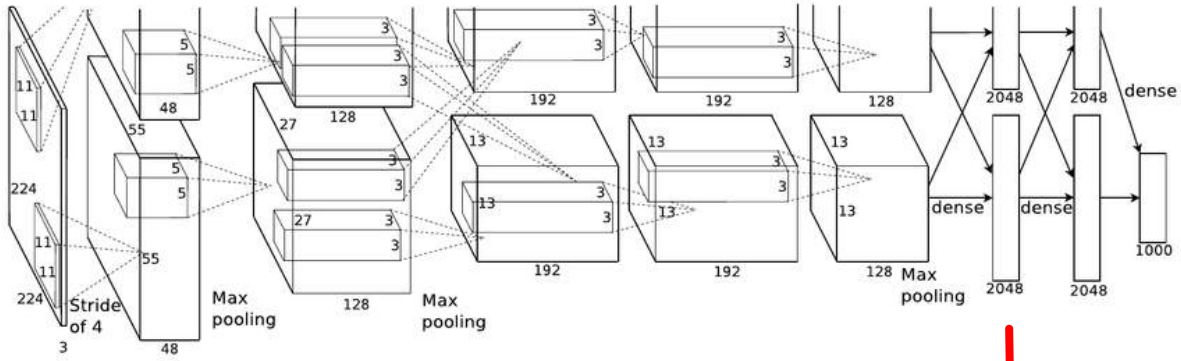
Smart descriptors vs. distances

- high-level descriptor + simple distance
  - bag of features models (high-dim euclidean space)
  - e.g., deep convolutional networks used as smart feature extractor

```
---------- Prediction for ../../../examples/images/bike.jpg ----------
0.5061 - "n03792782 mountain bike, all-terrain bike, off-roader"
0.4462 - "n09193705 alp"
0.0104 - "n09468604 valley, vale"
0.0079 - "n09246464 cliff, drop, drop-off"
0.0065 - "n09472597 volcano"
```

input: **color image** scaled to 224x224

AlexNet:

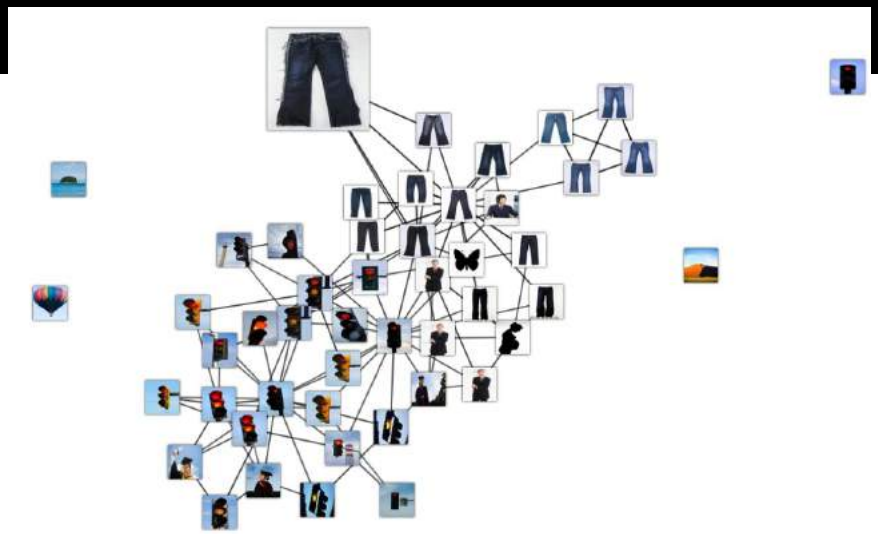


output: 1000D vector over **semantic classes**

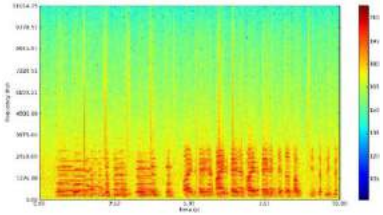
4096D visual-semantic feature vector (semantic descriptor)

Smart descriptors vs. distances

- deep convolutional networks used as smart feature extractor
  - real-world photography



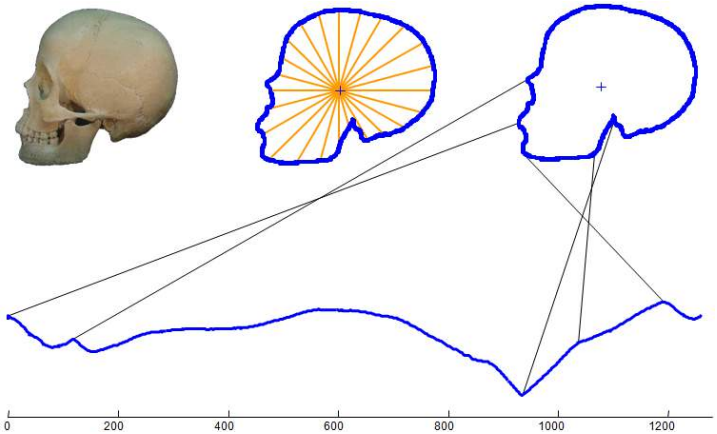
  - artificial images
    - non-image to image transformation
    - music genres
      - mp3 → spectrogram
      - fine-tuning the AlexNet CNN (17 classes instead of 1000)

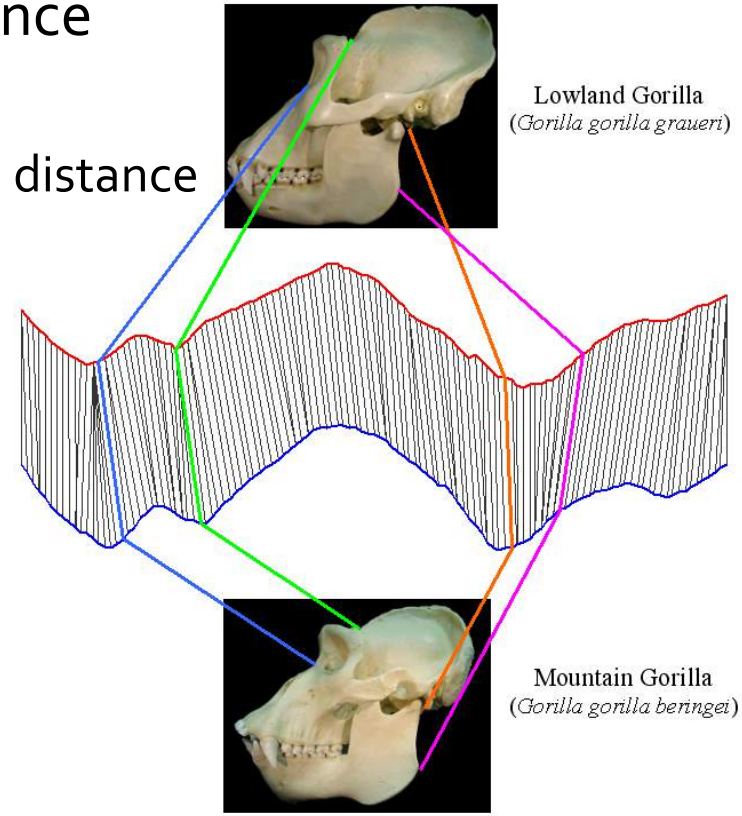


Smart descriptors vs. distances

(image © Eamonn Keogh, eamonn@cs.ucr.edu)

- low-level descriptor + complex distance
  - general non/metric spaces
  - e.g., time series + dynamic time warping distance
    - shape similarity example
    - **O(n²)** time complexity of DTW
    - elastic matching (nonmetric)



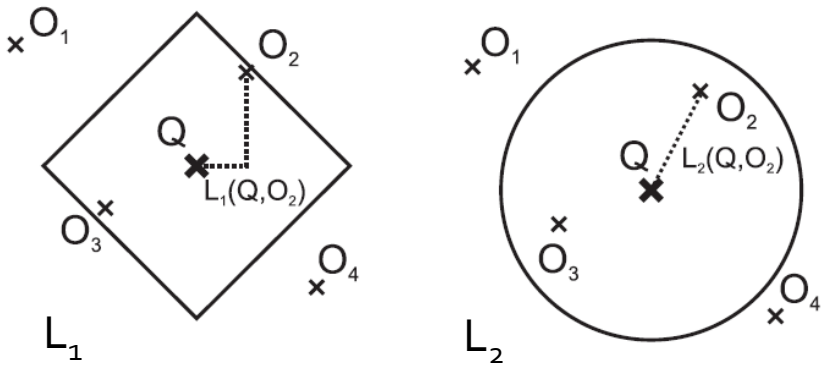


Similarity queries

- simple
  - range, kNN, RkNN
  - operators – similarity join
- composition of simple queries
  - aggregation (top-k operator)
- query languages for similarity search
  - predicates in SQL SELECT

Range query

- described by example object Q and query radius r_Q (or query threshold)
- returns all such objects $O_i \in \mathbf{S}$, for which $\delta(Q,O_i) \leq r_Q$
- geometric interpretation
  - a "ball region"
- requires user's knowledge of δ
  - size of result not known in advance
  - but 100% recall
  - sequential processing = comparison of all objects to Q



Range query – example

query
(Q, 1200)

Q:

dist = 0.0

1)

dist = 998.7

2)

dist = 1018.8

3)

dist = 1040.4

4)

dist = 1031.3

5)

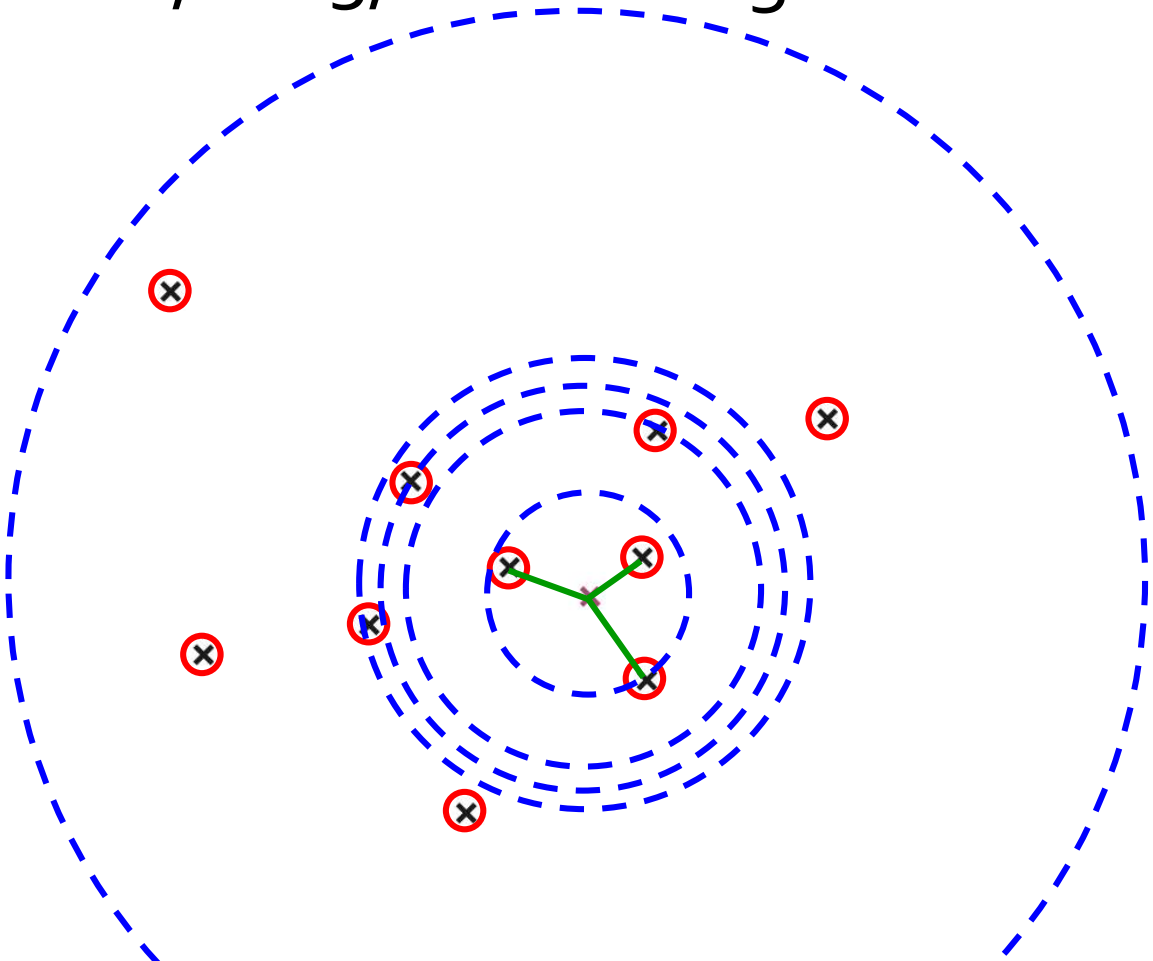
dist = 1166.2

k nearest neighbor (kNN) query

- described by example object Q and number k of most similar objects to be retrieved
- returns all such objects $O_i \in \mathbf{A} \subset \mathbf{S}$, that $|\mathbf{A}| = \mathbf{k}$ and $\forall Oj \in \mathbf{S} - \mathbf{A}$, $\delta(Q,O_i) \leq \delta(Q,Q_j)$
- geometric interpretation
  - again, "ball region", but unknown radius beforehand (distance to k-th NN)
- more suitable to end-user
  - known size of result (k)
  - but limited recall (is k large enough?)
- query processing
  - like range query – decreasing radius
  - incremental search – allows retrieving additional neighbors

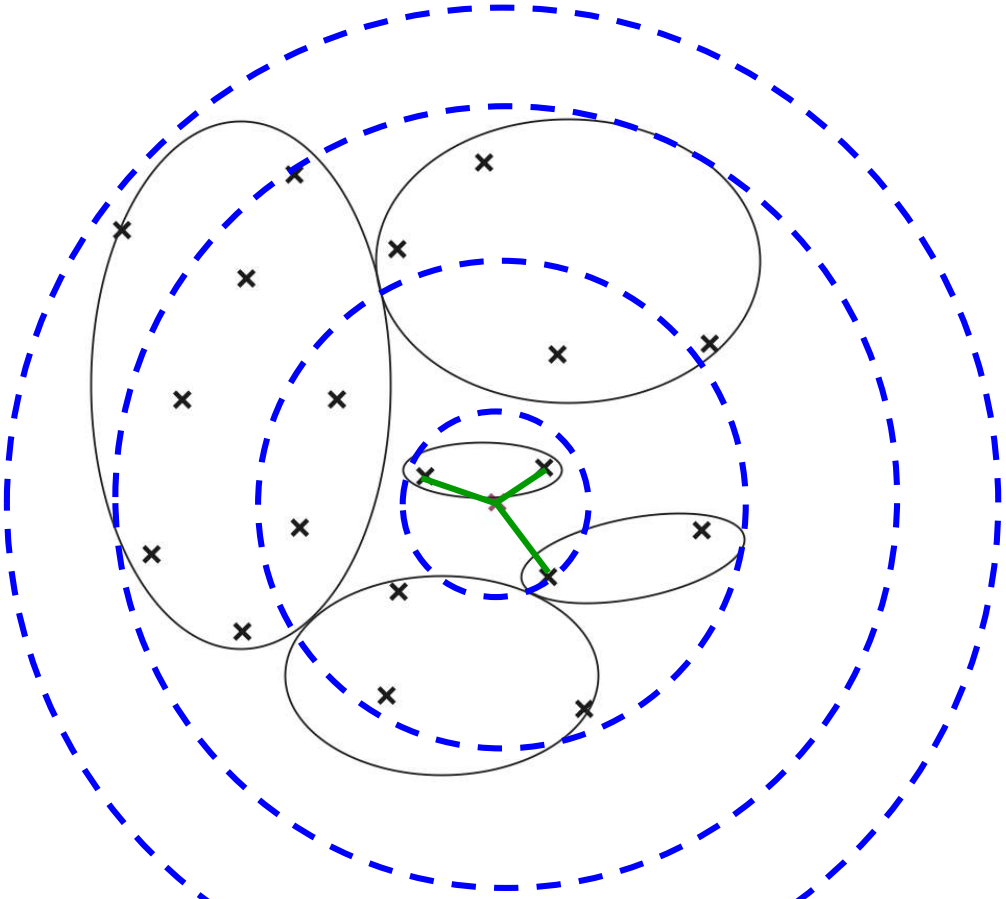
kNN query

- sequential, k = 3, decreasing radius



kNN query

- with index (faster radius decreasing), k=3



kNN query - example

1)

dist = 998.7

2)

dist = 1018.8

3)

dist = 1040.4

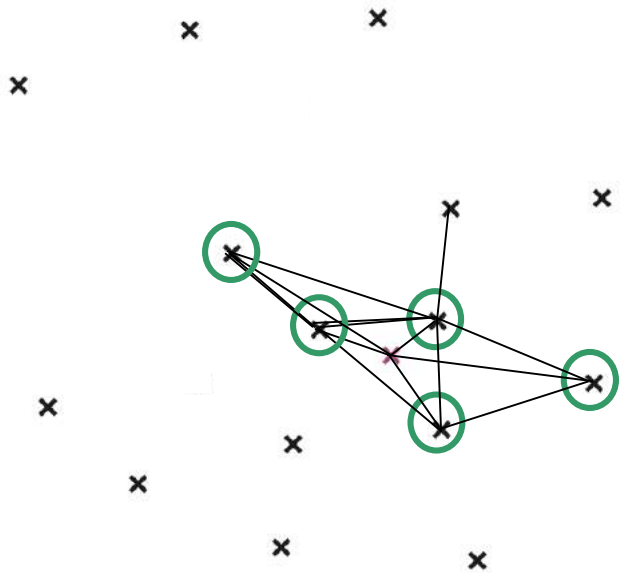
Q:

dist = 0.0

query
(Q, 3)

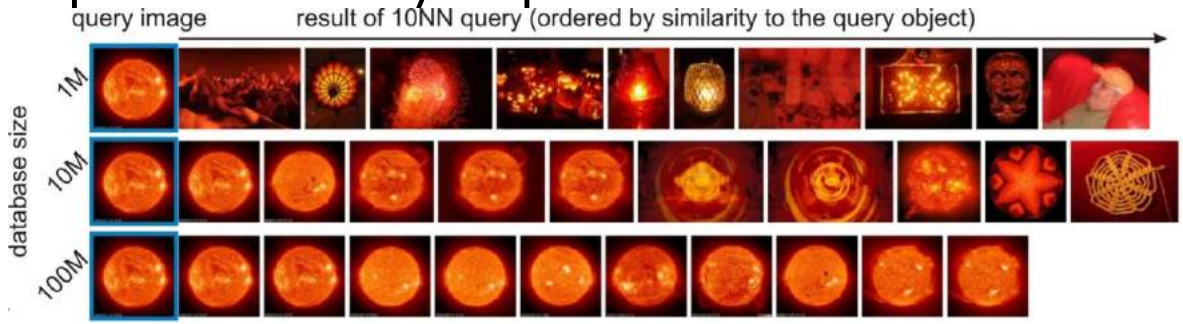
Reverse kNN (RkNN) queries

- described by example object Q and number k
- returns all such objects from **S**, for which Q is within their k nearest neighbors
  - i.e., $O_i \in \mathbf{A} \subset \mathbf{S}$ such that $Q \in kNN(O_i, k)$



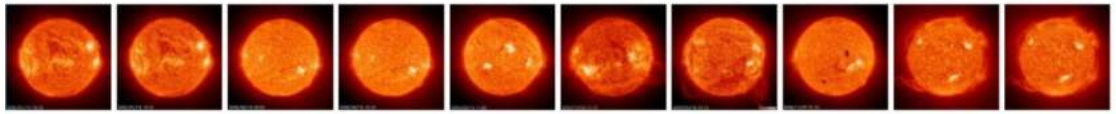
Advanced similarity queries

- in very large databases, simple kNN query might loose its expressive power – many duplicates

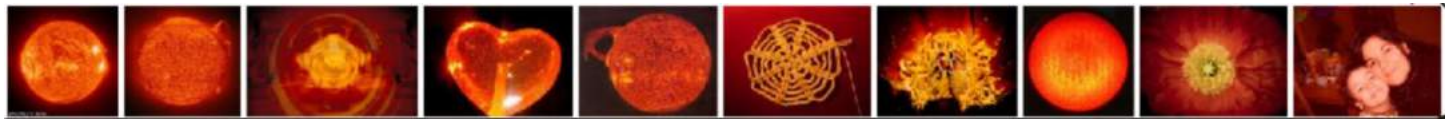


- distinct k nearest neighbors (DkNN)
  - selects k nearest, yet mutually distinct, neighbors
  - scalable expressive power (implemented to MUFIN)

10NN

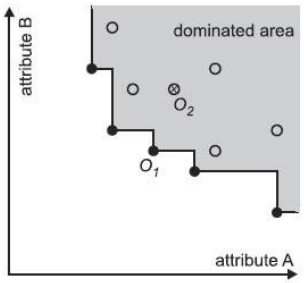


D10NN



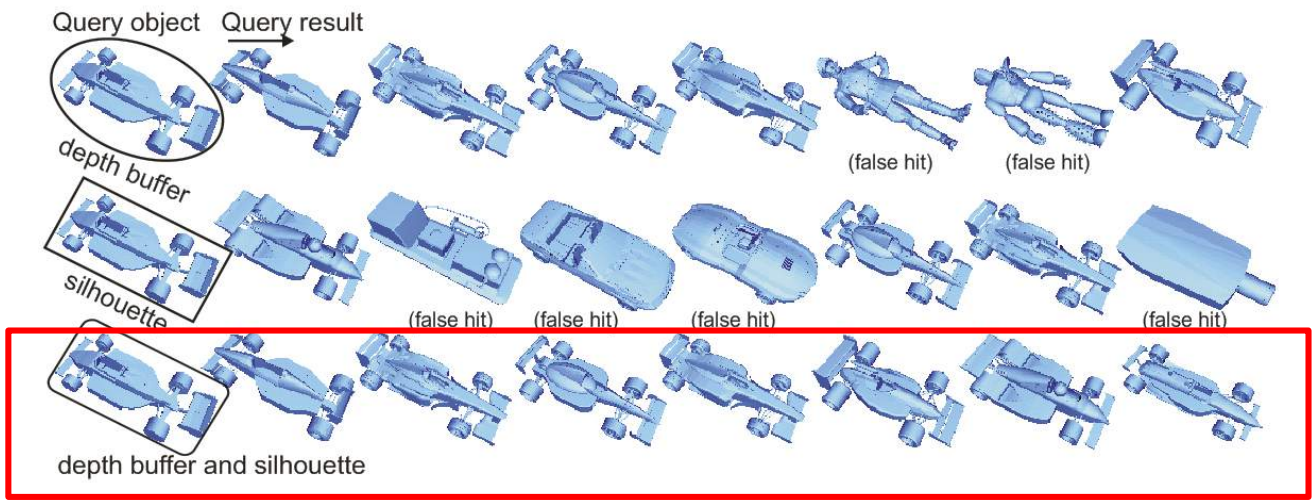
Advanced similarity queries

- finding a single "holy-grail" query example could be very hard
- multi-example queries can alleviate the problem
- metric skylines
  - multi-examples query, *k* examples $Q_1, \ldots, Q_k$
  - database objects are mapped into k-dimensional space $<\delta(Q_1, O_i), \ldots, \delta(Q_k, O_i)>$
  - then classic skyline operator is performed on the vectors



Aggregated queries

- differently ranked results could be independently aggregated
  - aggregating function
    - top-k operator



Re-ranking

- multi-phase querying (dependent aggregation)
  - applying different similarity model on part of the previous ranking
  - when single similarity is too complex to model

- example
  - first ranking as 4NN – color layout
  - re-ranking as 3NN – shape

Integrating similarity queries into SQL

- legacy implementation (embedded SQL)
  - descriptor attribute definition
  - feature extraction function
  - distance function
- language extensions
  - SIREN prototype

Integrating similarity queries into SQL

- legacy implementation
  - possible within SQL extension like PL/SQL, T-SQL, etc. using DDL a DML commands
  - descriptor attribute needed
    - descriptor attribute definition
      CREATE TABLE friend (name VARCHAR(20), photo BLOB)
      DDL AFTER trigger defined on the table for descriptor definition
      ALTER TABLE friend ADD (photoDescriptor BLOB)
      - instead of BLOB, we could use a user-defined type ImageDescriptor
    - descriptor update
      DML AFTER trigger defined on the table for feature extraction
      UPDATE friend SET photoDescriptor = **ImageFeatureExt**(photo)

Integrating similarity queries into SQL

- legacy implementation
  - distance function needed (user-defined function)
  - range query
    SELECT name,**Dist**(queryPhoto, friend.descriptor) AS d
    FROM friend WHERE d < range ORDER BY d ASC
  - kNN query (Microsoft SQL Server version)
    SELECT TOP k name,**Dist**(queryPhoto, friend.descriptor) AS d
    FROM friend ORDER BY d ASC

Integrating similarity queries into SQL

- legacy implementation
  - range similarity join

    SELECT Criminal.Id, Citizen.Id
    FROM Criminal INNER JOIN Citizen ON
    Dist(Criminal.Fingerprint, Citizen.Fingerprint) < 0.01

  - kNN similarity join

    SELECT Mammal.Id, Insect.Id FROM
    Mammal INNER JOIN Insect ON Insect.Id IN
    (SELECT Id FROM(
    SELECT TOP k Id,**Dist**(Mammal.DNA, Insect.DNA) AS d
    FROM Insect ORDER BY d ASC))

Integrating similarity queries into SQL

- legacy implementation
  - pros
    - no need to alter the existing DBMS
    - no restriction on distance function (metric/nonmetric)
  - cons
    - no indexing (slow)
    - no query optimization (slow)
    - not integrated for easy use (lot of programming needed)

Integrating similarity queries into SQL

- SIREN approach
  - extension of SQL supporting similarity queries
    (Barioni et al, Softw. Pract. Exper. 39:355–384, Wiley Interscience, 2009)

- new database objects (types, metric, index)
  - data types STILLIMAGE, AUDIO

```
CREATE TABLE friend (
name VARCHAR(30),
photo STILLIMAGE,
...
```

Integrating similarity queries into SQL

- new database objects (types, metric, index)
  - metric distance definition

```
CREATE METRIC <name> USING <function>
FOR STILLIMAGE | AUDIO | ...
    (<extractor>(<feature> AS <place holder>,...),
    <extractor>(<feature> AS <place holder>,...) ...);
```

example

```
CREATE METRIC Histogram USING LP1 FOR
            STILLIMAGE(HistogramEXT (HistogramFeature AS Histo));
```

Integrating similarity queries into SQL

- new database objects (types, metric, index)
  - metric index construction (Slim-tree used)

    CREATE INDEX Frontal ON Friend (photo)
    USING Histogram;

Integrating similarity queries into SQL

- native predicates in the WHERE clause
  <attr> NEAR <value> [STOP AFTER <k>] [RANGE <r>]

  example kNN

  ```
  SELECT * FROM friend WHERE photo
  NEAR 'c:\JohnFrontal.jpg' STOP AFTER 5;
  ```

  example kNN using multiple descriptors/metrics (intersection)

  ```
  SELECT * FROM friend WHERE
  (photo NEAR 'c:\img.jpg' BY Histogram STOP AFTER 5) AND
  (photo NEAR 'c:\img.jpg' BY Texture STOP AFTER 5);
  ```

Integrating similarity queries into SQL

- similarity joins
  - defined as selection from Cartesian product (std. SQL 89)

    example

```
SELECT * FROM Employee E1, Employee E2
WHERE E1.photo NEAR E2.photo STOP AFTER 5
AND E1.City = 'New York' AND E2.City = 'San Francisco';
```

course:
Multimedia Retrieval

lecture 5:

Similarity search – popular similarity functions

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

- vector distances
  - independent dimensions
  - histograms
- sequence distances
  - time series
  - strings
- set distances
  - general
  - signatures
- combined distances

Descriptor vs. similarity

- where to put the retrieval logic?
  - high-level descriptor + simple distance
    - feature extraction does most of the logic
    - assumed canonized representation of features is possible
    - cheaper similarity
  - low-level descriptor + complex distance
    - similarity aggregates relevant information at query time, i.e., information that cannot be extracted beforehand
    - more flexible but more expensive similarity

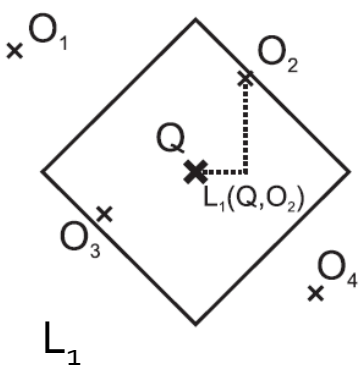
Vector similarities – Lp distances

- Minkowski (Lp) distances

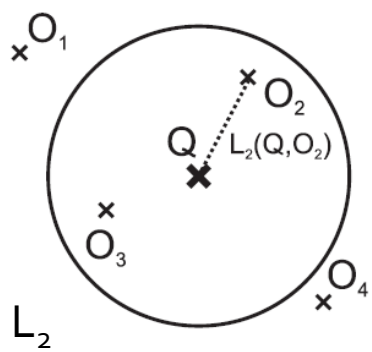
$$L_p(v_1, v_2) = \left(\sum_{i=1}^{D} |v_1[i] - v_2[i]|^p\right)^{\frac{1}{p}} \qquad (p \geq 1)$$

- very popular in many domains
- independent dimensions supposed
- cheap – O(D)
- metric

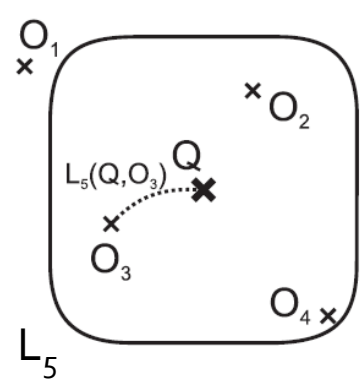
Manhattan distance (L_1)

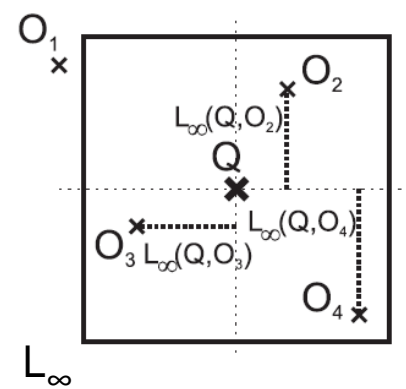


Euclidean distance (L_2)



Chessboard/Max distance (L_∞)





Vector similarities – cosine/angle

- cosine measure/angle distance
  - vector magnitude not important
  - the same as norming the vector + L_2
  - cheap O(D)
  - equivalent variants
    - **cosine similarity/measure**
      - non-metric (only symmetry)

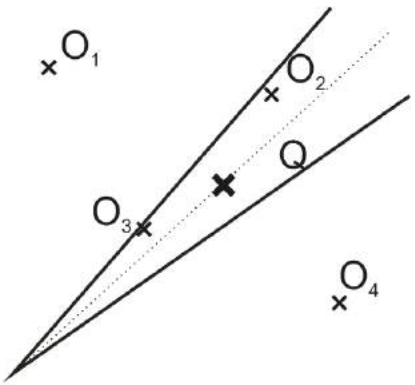
$$\mathrm{SIM}_{cos}(v_1, v_2) = \frac{\sum_{i=1}^{D} v_1[i] v_2[i]}{\sqrt{\sum_{i=1}^{D} v_1[i]^2 \cdot \sum_{i=1}^{D} v_2[i]^2}}$$

    - **cosine distance**
      - non-metric (only symmetry)

$$d_{cos}(v_1, v_2) = 1 - \mathrm{SIM}_{cos}(v_1, v_2)$$

    - **angle distance**
      - pseudo-metric (symmetry, triangle ineq.)

$$d_{angle}(v_1, v_2) = arccos(\mathrm{SIM}_{cos}(v_1, v_2))$$

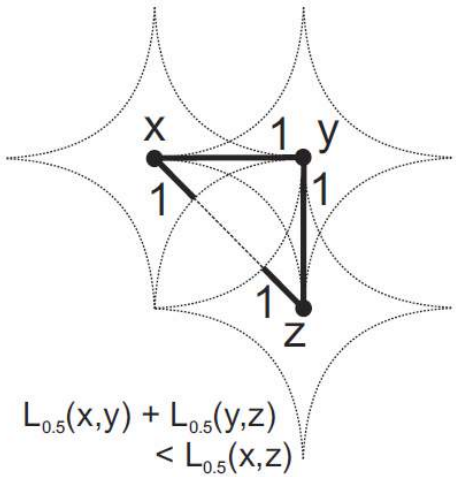


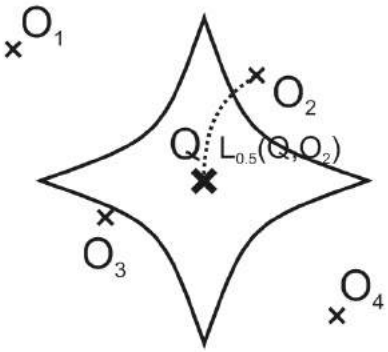
Vector similarities – frac. Lp dist.

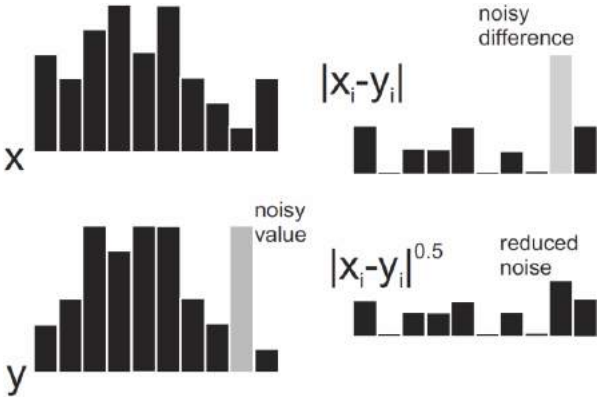
- fractional L_p distances

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

  - extension of Minkowski distances for 0<p<1
    - more robust to extremes
  - cheap O(D)
  - non-metric (triangle ineq.)
  - robust





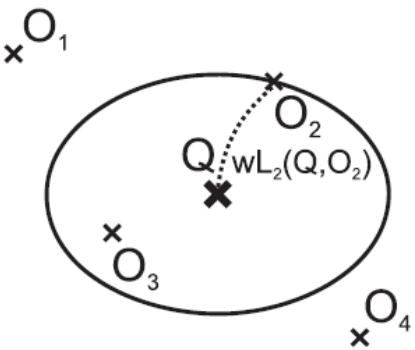


Histogram similarities – wL_2

- weighted Euclidean distance

$$wL_2(v_1, v_2) = \sqrt{\sum_{i=1}^{D} w_i (v_1[i] - v_2[i])^2}$$

  - more/less important dimensions
  - Euclidean distance special case ($w_i = 1$)
  - cheap O(D)
  - metric (if positive weights)



Histogram similarities – statistic distances

- Kullback-Leibler divergence

$$\delta_{KLD}(x,y) = \sum_{i=1}^{D} x_i \cdot \log\left(\frac{x_i}{y_i}\right)$$

  - based on information theory
  - measures how inefficient on average it would be to code one histogram using the other as the true distribution for coding
  - nonmetric
- Jeffrey-Divergence

$$\delta_{JD}(x,y) = \sum_{i=1}^{D} x_i \cdot \log\left(\frac{x_i}{\frac{x_i+y_i}{2}}\right) + y_i \cdot \log\left(\frac{y_i}{\frac{x_i+y_i}{2}}\right)$$

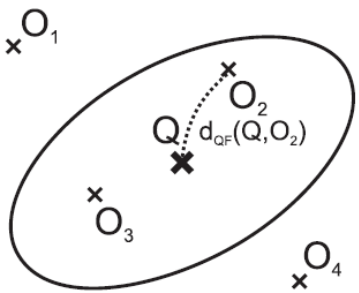
  - nonmetric
- χ^2 distance

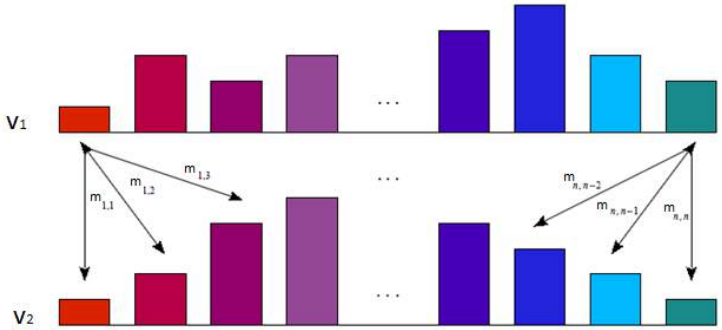
$$\delta_{\chi^2}(x,y) = \sum_{i=1}^{D} \frac{x_i - m(i)}{m(i)} \qquad m(i) = \frac{x_i+y_i}{2}$$

  - measures if two empirical distributions were produced from the same underlying true distribution
  - nonmetric

Histogram similarities – QFD

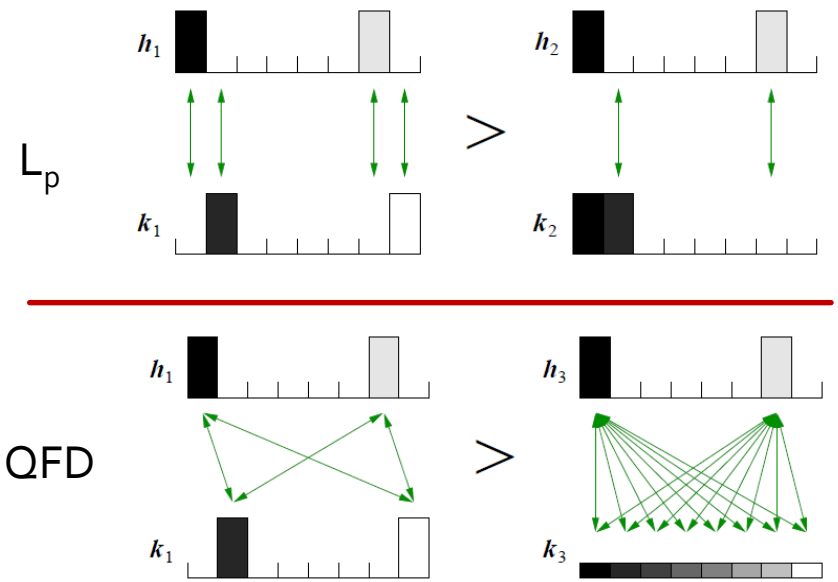
- quadratic form distance $$d_{QF}(v_1, v_2) = \sqrt{(v_1 - v_2)M(v_1 - v_2)^T}$$
  - correlated dimensions supposed
    - correlation matrix M
  - weighted L_2 special case (diagonal matrix M)
  - expensive, $O(D^2)$
  - metric (M must be positive-definite matrix)



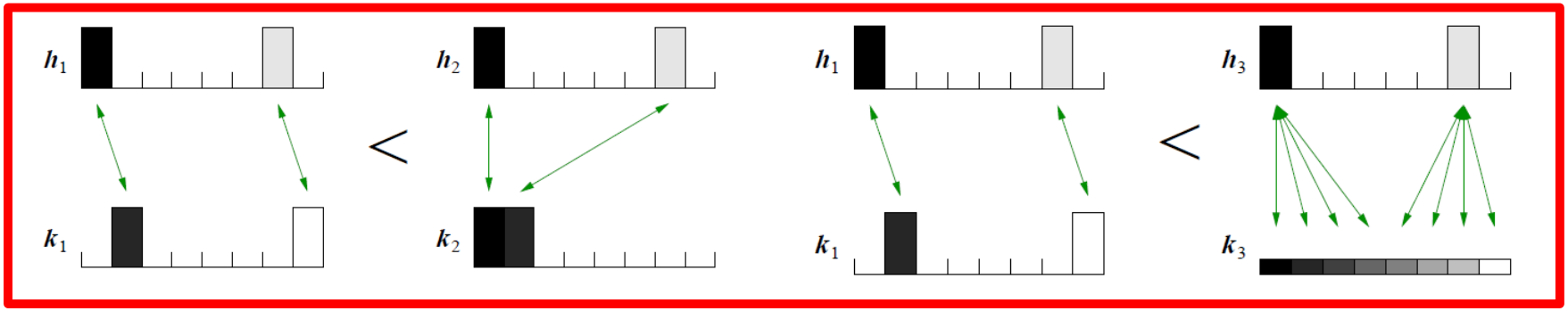


Histogram similarities – EMD

- motivation
  - L_p – dimensions strictly independent
  - QFD
    - correlated dimensions (structure) but data-independent similarity
  - so L_p and QFD do not match perceptual similarity



desired similarity (EMD):



(figures from Rubner et al., The Earth Mover's Distance as..., Int. J. of Comp. Vision, 2000)

Histogram similarities – EMD

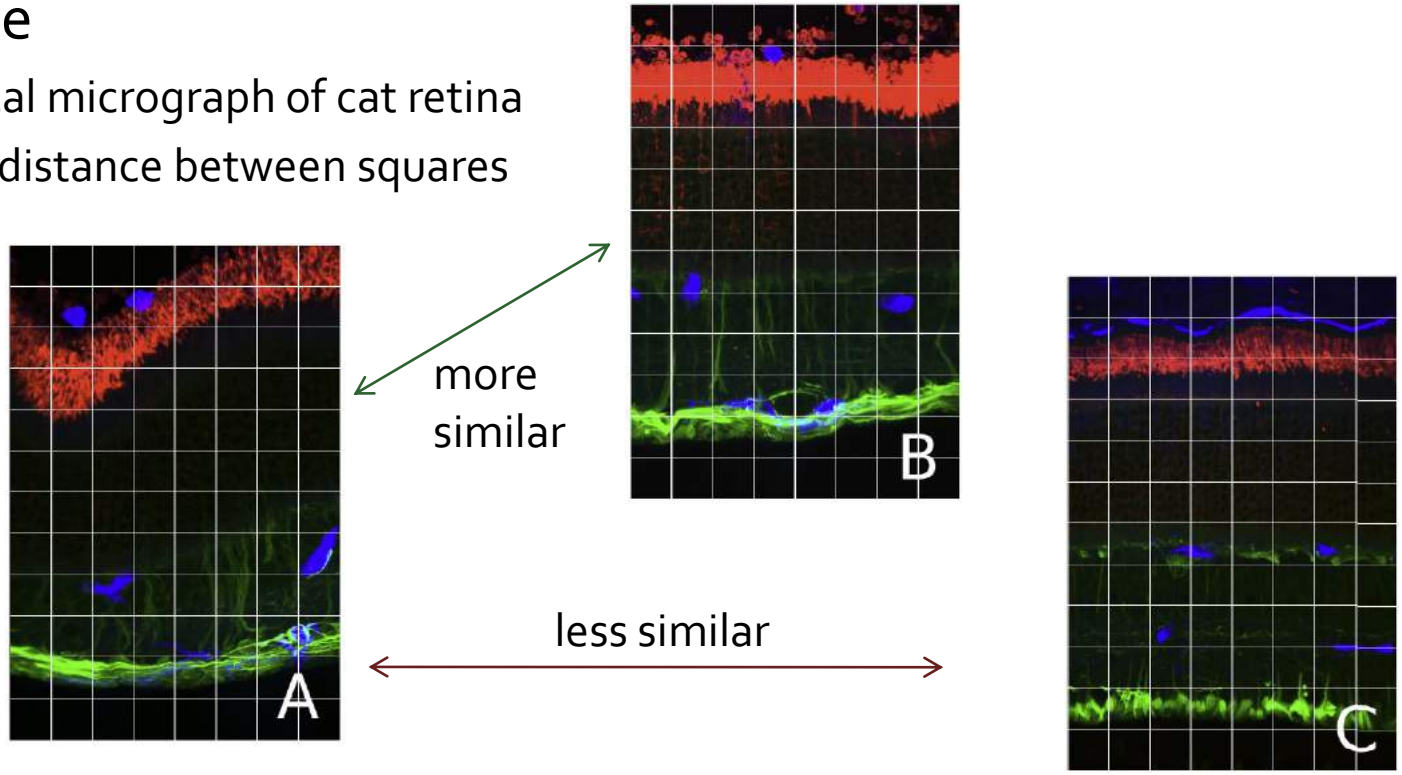
- earth mover's distance (EMD)
- two histograms = two "piles of dirt" (distributions of values)
- EMD(x,y) = minimum amount of work to turn pile x into pile y
  - f – flow (work plan)
    - f_{ij} – how much mass moving from x_i to y_j
  - c_{ij} – ground distance $d(x_i, y_j)$

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

- metric distance (if c_{ij} is metric)
- very expensive computation
  - $O(D^3 \log D)$ in case both histograms are of equal dimensionality $D = m = n$
  - $O(2^D)$ otherwise (by linear programming, simplex), $D = \max(m,n)$

Histogram similarities – EMD

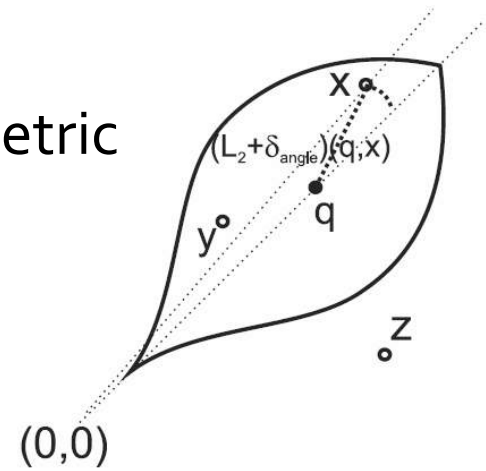
- suitable also to multi-dimensional histograms and signatures (due to general ground distance c_{ij})
- example
  - confocal micrograph of cat retina
  - $c_{ij} = L_2$ distance between squares



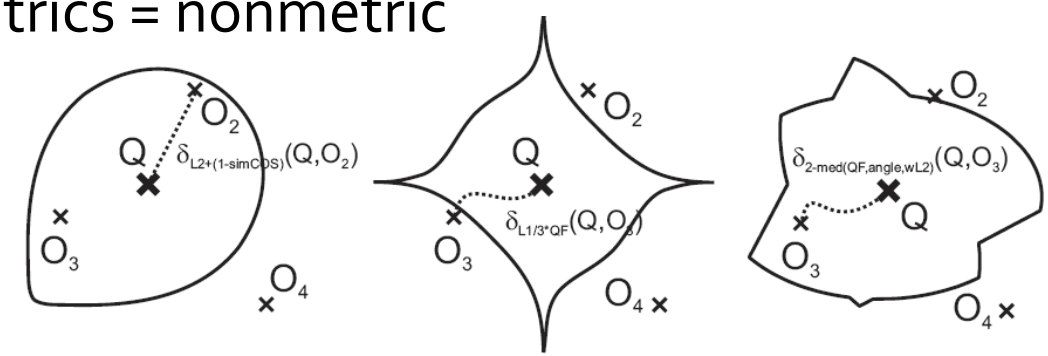
(figures from Ljosa et al., Indexing Spatially Sensitive Distance…, EDBT 2006)

Vector similarities – composed

- metric compositions
  - linear combination of metrics = metric

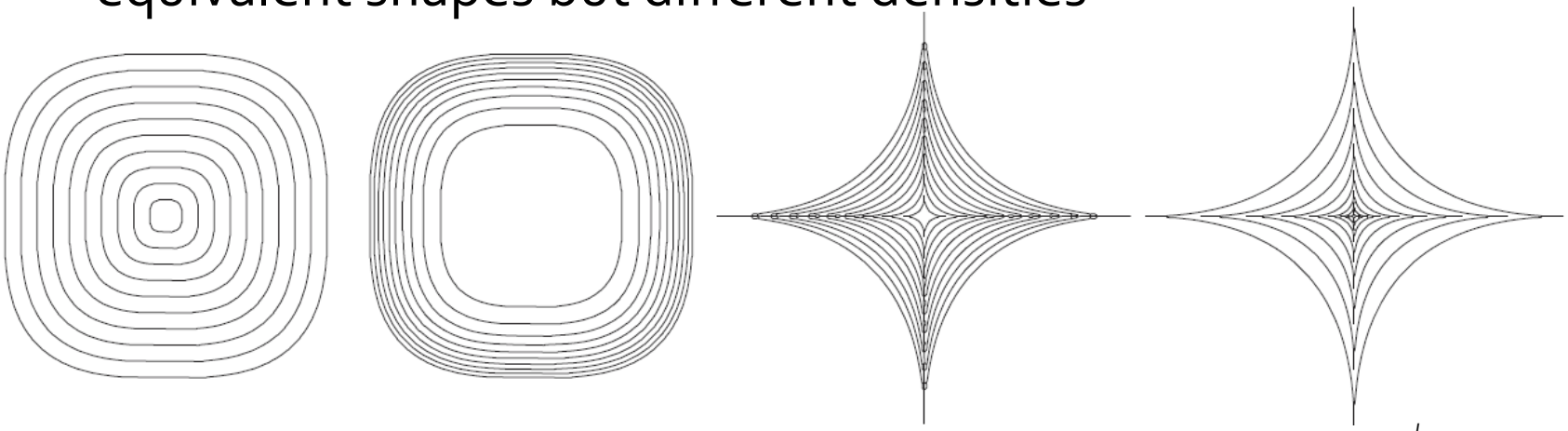


- non-metric compositions
  - multiplication of metrics = nonmetric
  - any combination with a nonmetric = nonmetric



Vector similarities – regions

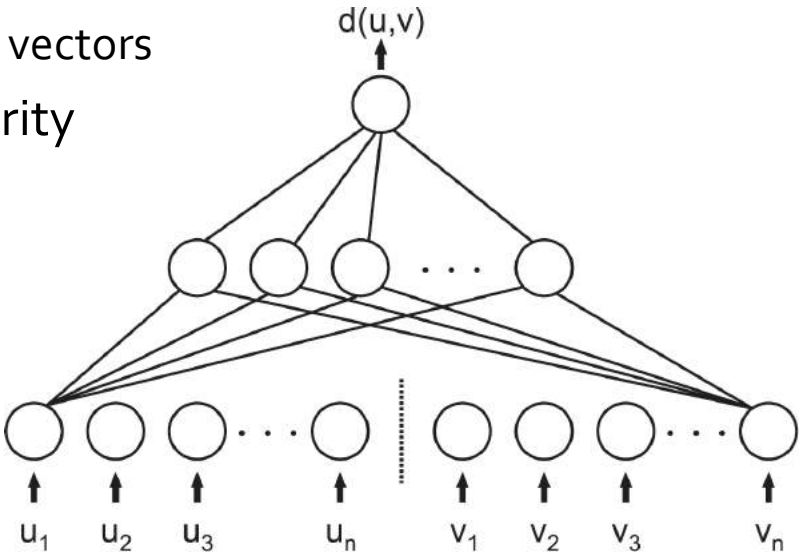
- convex/concave shape of ball region
  = no information on "metricity" of d
- density gives some information
- example
  - equivalent shapes but different densities



metric (L_5) nonmetric (L_5^2) nonmetric ($L_{0.5}$) metric ($\sqrt{L_{0.5}}$)

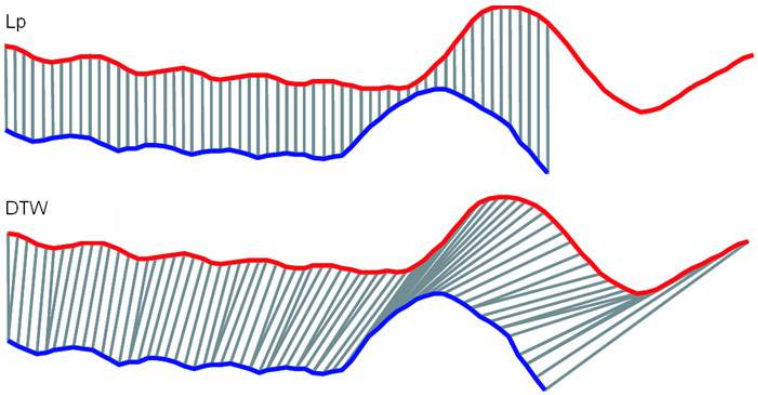
Vector similarities – neural network

- COSIMIR
  - three-layer backpropagation neural network
  - input – two data vectors u,v
  - output – distance/similarity value d(u,v)
  - learning by backpropagation
    - training set of user-valuated pairs of vectors
  - not very practical for general similarity
    - local extremes (not converging)
    - user could contradict himself (valuating the same differently)
    - the neural network model could be too rough (e.g., even L_2 is hard to model)



Sequence similarities – DTW

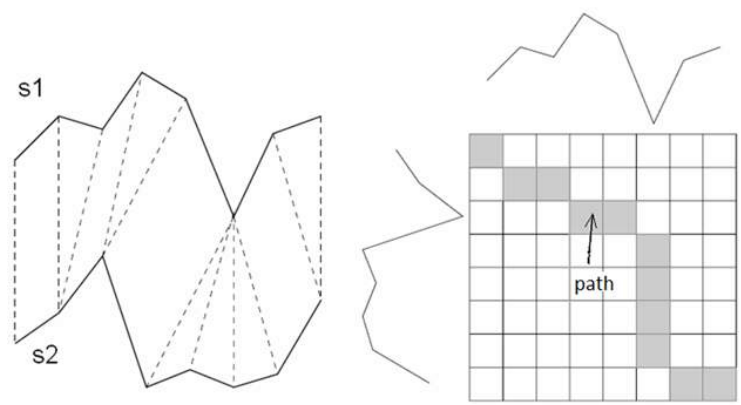
- sequences (time series) could be compared by vector similarities, e.g. L2, but
  - unsuitable for differently sized sequences
  - limited to real-value sequences
- dynamic time warping distance (DTW)
  - allows to stretch one series (even locally) to match the other
  - sequence alignment
  - allows arbitrary sequence elements
    - general ground distance
  - similar to EMD, but restricted to monotone distribution of mass
  - nonmetric



Sequence similarities – DTW

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

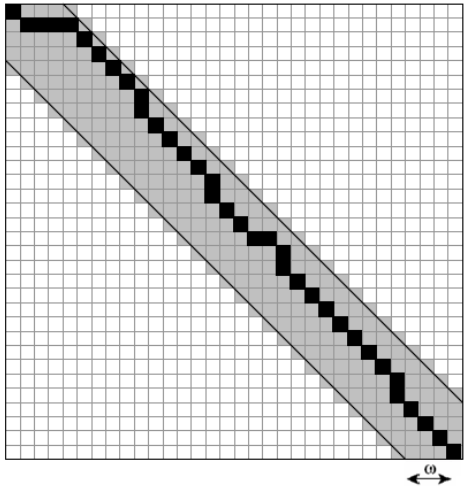
- sequences s_1, s_2
- m x n matrix M, where $m = |s_1|$, $n = |s_2|$
- matrix cell $M_{i,j}$ is partial distance $d(s_{1i}, s_{2j})$
- warping path $W = \{w_1, \ldots, w_t\}$, $\max\{m, n\} \leq t \leq m + n - 1$, is a set of cells from M that are contiguous
  - $w_1 = M_{1,1}$, $w_t = M_{m,n}$ *(boundary condition)*
  - if $w_k = M_{a,b}$ and $w_{k-1} = M_{a',b'}$, then
    - $a - a' \leq 1 \wedge b - b' \leq 1$ *(continuity)*
    - $a - a' \geq 0 \wedge b - b' \geq 0$ *(monotonicity)*
- DTW = L_2 distance on optimally aligned sequences (optimal warping path)



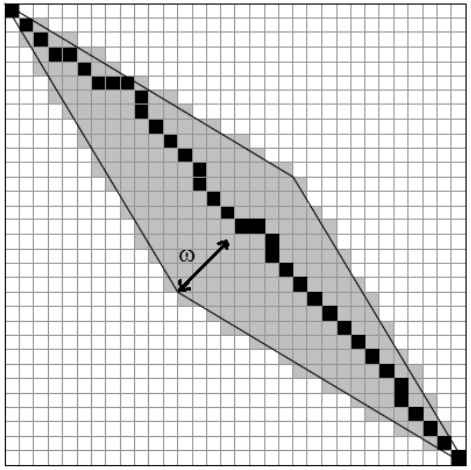
Sequence similarities – DTW

- exponentially many warping paths, but can be computed in O(mn)*O(ground dist.) time by dynamic programming
- constrained versions of DTW
  - avoiding pathological paths
    - a range parameter ω
    - by $\omega = 0$, m=n, d(x,y) = |x-y| we get the Euclidean distance (just the diagonal warping path allowed)
  - DTW reduced complexity to $O((m+n)\omega)$
  - Sakoe-Chiba band
    - warping paths are only allowed near the diagonal
  - Itakura Parallelogram
    - "time warping" in the middle of sequences is allowed, but not at the ends

Sakoe-Chiba band



Itakura Parallelogram



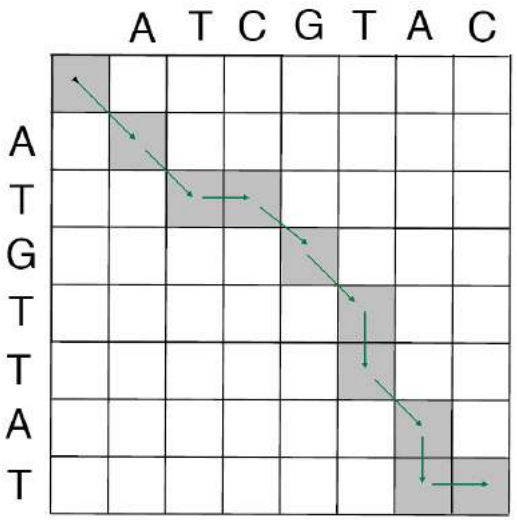
String similarities – edit, Hamming

- edit distance
  - the smallest number of operations needed to convert one string into another
  - character operations – insert, delete, replace
    - replacement could be removed (then in-del distance)
    - weighed edit distance (each operation weighted)
  - similar concept as DTW
    - paths in a matrix, sequence alignment, dynamic programming
  - metric distance
  - expensive O(mn)

indel(ATGTTAT
ATCGTAC) = 4

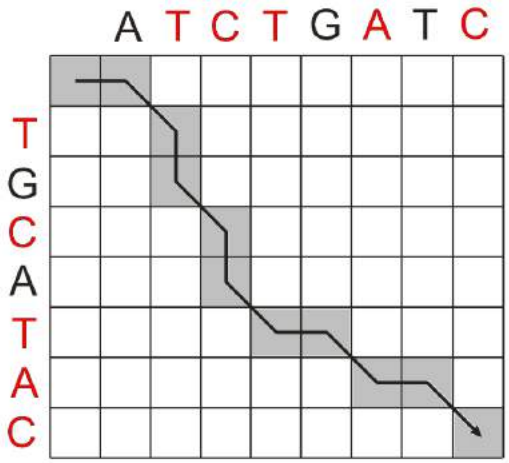
- Hamming distance
  - edit distance with only substitution allowed
    - on bit strings – XOR
  - strings of equal lengths
  - metric distance
  - cheap O(n)

hamming(ABCDEF
BCDEFA) = 6



String similarities – LCSS

- LCSS (longest common subsequence)
  - searching for longest subsequence (not only substring)
  - subsequence determined by any set of indexes in sequence
    - LCSS(ABCD, ACBD) = 3 (either ABD or ACD)
  - again, similar concept as DTW
    - paths in a matrix, sequence alignment, dynamic programming
    - just binary valuation of character pairs (match / mismatch)
  - used in DNA databases
  - nonmetric
    - similarity
    - local/robust behavior (triangle inequality violated)
  - expensive O(mn)



String similarities – biology

- edit distance adopted to biology
  - protein sequences (alphabet of 20 amino acids)
  - sequence similarity should mimic functional similarity of proteins
- biology-specific extensions
  - scoring matrix (probability that amino acid A will mutate into acid B)
  - gap penalty – opening gap is more penalized than continuing gap

- global alignment
  - Needleman-Wunch

N	P	H	G	I	I	M	G	L	A	E	→	-16
-7	-7	+8	+6	-7	-7	+2	+6	+4	-7	-7		
-	-	H	G	-	-	L	G	L	-	-		

- local alignment
  - Smith-Waterman

N	P	H	G	I	I	M	G	L	A	E	→	16
		+8	+6	+2								
		H	G	L								

Set similarities – Jaccard, Hausdorff

- Jaccard distance (normed overlap distance)
  - normed size of set intersection

$$d_{NO}(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

dNO({cat, dog, mouse}, {keyboard, mouse}) = 0.75

- Hausdorff distance
  - measuring the *furthest nearest neighbor*
    - for every element *a* in set A, determine the distance of *a* to its nearest neighbor in B (by a ground distance)
    - take the maximum distance
    - do the same again, switching the roles of A and B, and return the greater of the two maximum distances

$$d_H(A, B) = \max(h(A, B),\ h(B, A))$$

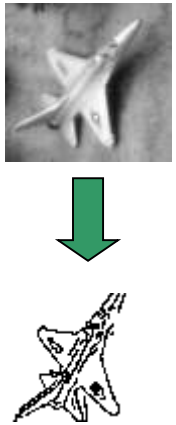
$$h(A, B) = \max_i \min_j \delta(A_i, B_j)$$

  - is metric if the ground distance is metric
  - expensive O(mn)*O(ground dist.)

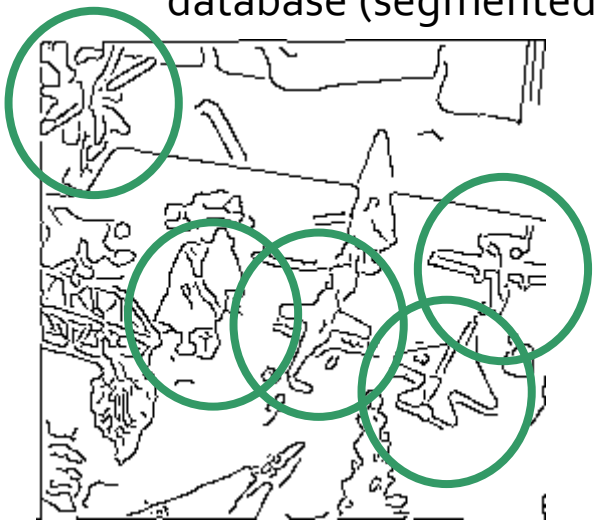
Set similarities – Hausdorff dist.

- application – shape retrieval
  - descriptor is a set of lines
  - ground distance is L_2 distance of two lines

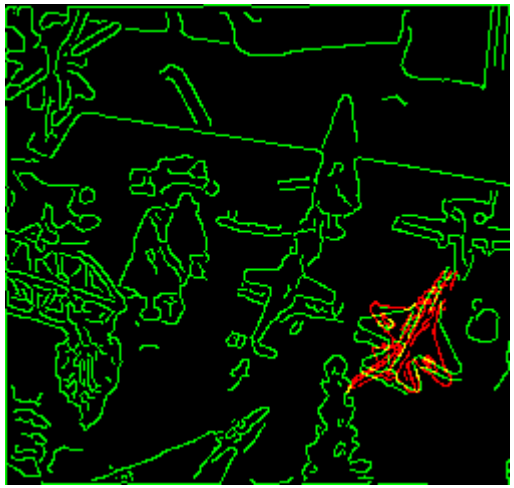
query



database (segmented image)



result – the closest descriptor

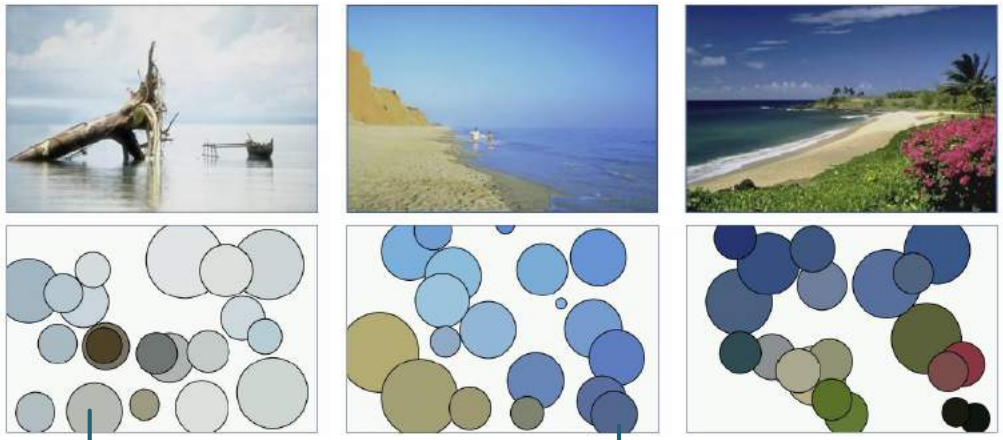


Set similarities – Hausdorff dist.

- application – fingerprint identification
  - descriptor is a cloud of points
  - ground distance is L_2 distance of two points

Set similarities – SQFD

- signature quadratic form distance
  - feature signature – set of weighted feature representatives (e.g., centroids of feature clusters + radius)



$$S^q = \{\langle r_i^q, w_i^q \rangle\}_{i=1}^n \qquad S^p = \{\langle r_i^p, w_i^p \rangle\}_{i=1}^m$$

centroid cluster weight

Set similarities – SQFD

- signature quadratic form distance
  - generalization of QFD for feature signatures (sets)

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

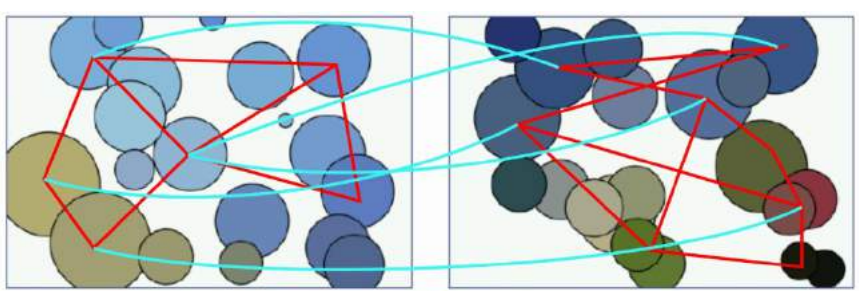
  - metric and ptolemaic distance (if A_{fs} is positive-definite)

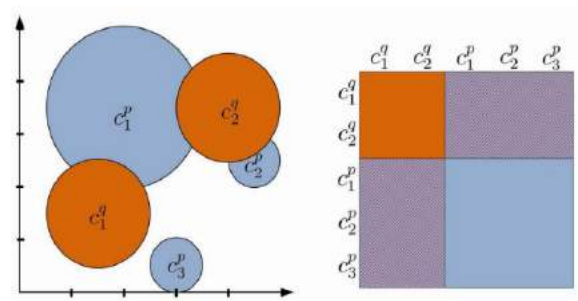
ground distance

$$f_-(r_i, r_j) = -d(r_i, r_j)$$ minus

$$f_h(r_i, r_j) = \frac{1}{\alpha + d(r_i, r_j)}$$ heuristic

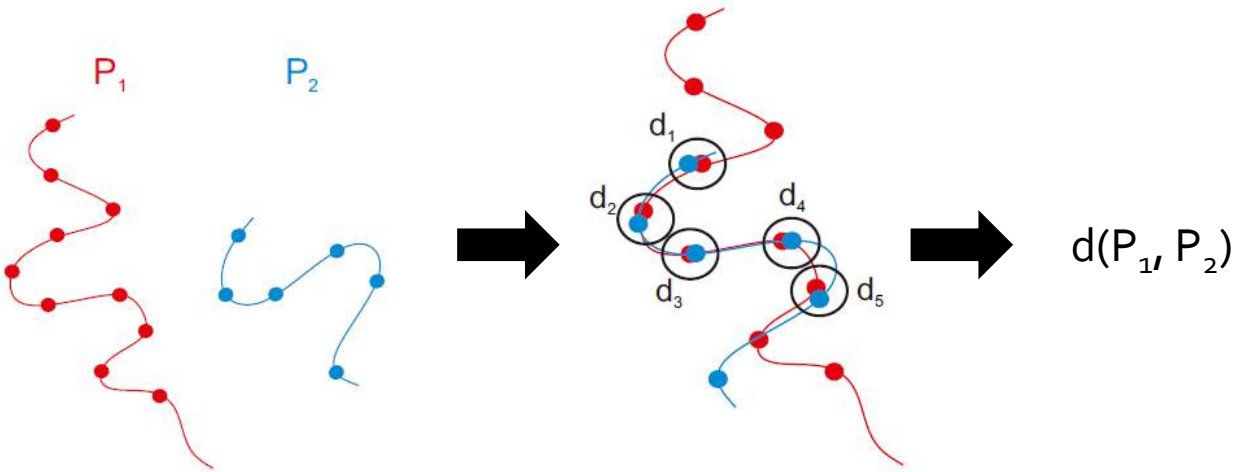
$$f_g(r_i, r_j) = e^{-\alpha \cdot d^2(r_i, r_j)}$$ gaussian





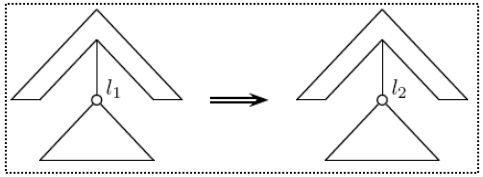
Set similarities – polygon similarity

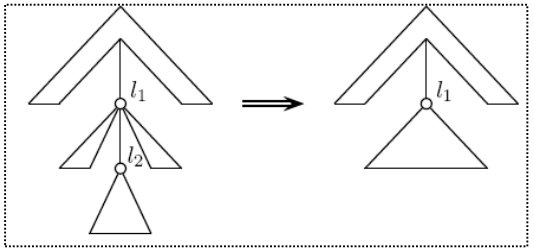
- general process
  - find optimal superposition (rotation, shift, scaling)
  - that minimizes the distance
- general geometry + biological applications (protein structure)
- nonmetric (local alignment)

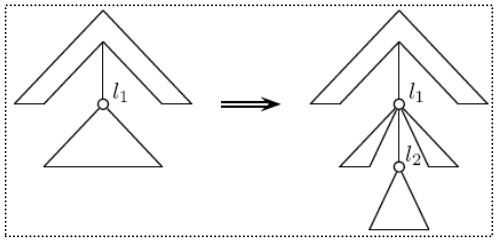


Graph similarity – tree edit distance

- structural similarity of trees
- tree edit distance
  - the smallest number of tree edit operations needed to transform one tree into another
    - node relabeling, node deletion, node insertion
  - analogy to string edit distance
  - structural similarity of XML documents
  - complexity $O(n^4)$, n number of nodes







Combined distances

- demand for flexible similarity
  - descriptor composed from multiple sub-descriptors
  - each sub-descriptor its own similarity model (metric space)
  - for each query object different contribution of particular sub-descriptor needed
- multi-metric
  - linear combination of distances on sub-descriptors
  - vector of weights W defined at query time

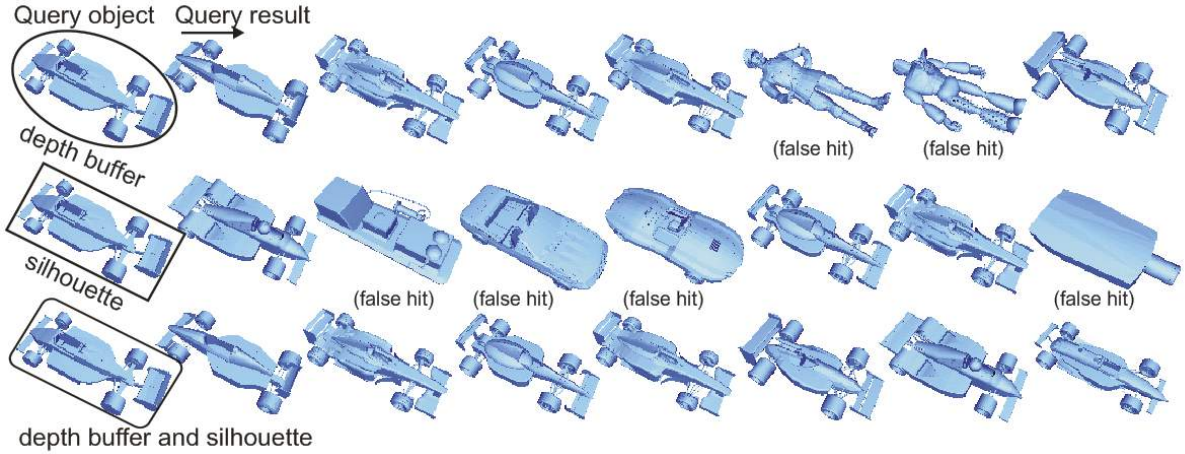
$$\mathbb{W} = \langle w_i \rangle$$

$$\Delta_{\mathbb{W}}(O_1, O_2) = \sum_{i=1}^{m} w_i \cdot \delta_i(O_1, O_2)$$

Combined distances

- multi-metric consisting of
  - depth buffer feature + distance δ_1
  - silhouette feature + distance δ_2

- results for query model "formula", when W set as
  - $w_1 = 1$, $w_2 = 0$
  - $w_1 = 0$, $w_2 = 1$
  - $w_1 = 0.5$, $w_2 = 0.5$



course:
Multimedia Retrieval

lecture 6:

Global image descriptors – analytic models

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

- Multimedia features and semantic gap
- MPEG-7 standard
- selected visual descriptors (analytic global descriptors)
  - color descriptors
  - texture descriptors
  - application-specific (high-level) descriptors
    - face descriptor

The presentation contains simplified facts – for more detailed reading see monograph

Manjunath et al, *Introduction to MPEG-7*, Wiley, 2003

Content-based retrieval

- late 1970s – text-based retrieval
  - problems with textual annotations such as subjectivity and human labor, therefore
- early 1990s – content-based retrieval (CBR)
  - large-scale collections, mostly images
  - automated feature extraction of low-level features
  - indexing for efficient search
  - query interfaces & interactive processing to bridge the "semantic gap"
- since 2012 – semantic descriptors (deep learning)
  - next lecture

Features of features

- feature locality
  - global (property of entire image/video)
  - local (related to certain segment)
- feature specificity
  - general features
    - low-level features (e.g., color, texture, shape, melody, sound envelopes)
    - high-level features (recognized objects, voices)
  - domain-specific features (e.g., face, fingerprint)

Semantic gap

- high-level features = concepts/objects
  - that is what typical user wants
    - query like: **"Give me images of a brown horse on a hill"**
  - extremely difficult to extract from general imagery

(credit: Google)

Semantic gap

- low-level features
  - visual characteristics in case of imagery
  - typical user cannot use in a query because of lacking meaningful taxonomy/ontology on such features
    - the same "horse" query reformulated to low-level description: **"Give me images mostly green on bottom, brown in the middle, and blue on top and sideways"**
      - could return "green apple with brown stalk put on a blue table"
  - easier to extract (color, texture, shape, audio spectrum)
- semantic gap = we are not able to bridge the two levels
  - except some domain-specific cases

Semantic gap

- funny example – both images share many low-level features (color, texture, shape), but not the semantics, indeed ☺

(credit: Google)

MPEG

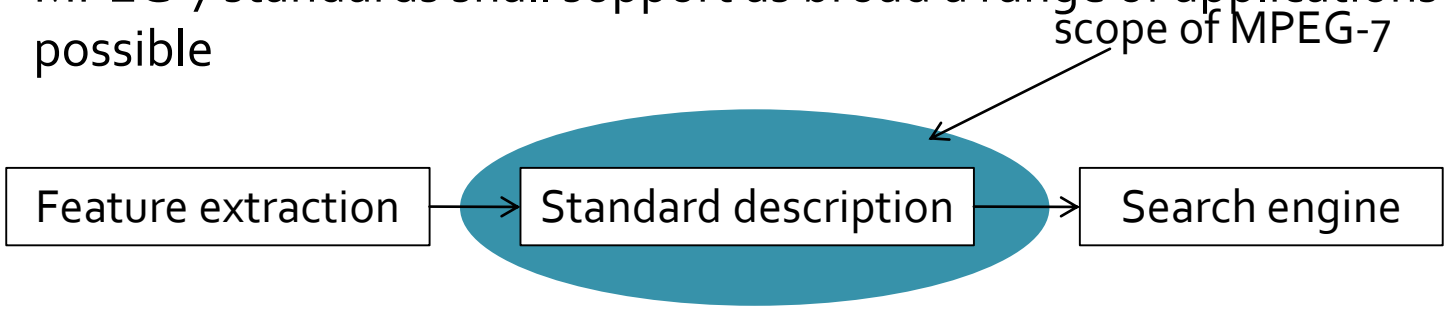
- Moving Picture Experts Group, est. 1988
- developing digital standards for representation of audiovisual (AV) content
- achievements
  - MPEG-1 – separated decoder (standard) and encoder (left to industry competition)
  - MPEG-2 – massive applications in TV broadcasting, and consumer electronics
  - MPEG-4 – even more usage in TV and also on the Internet and in mobile platforms

MPEG

- problem spotted?
  - tools needed that enable identification and navigation in the abundance of AV content
  - hence, works on the definition of "*audiovisual information representation standard that would describe or express the semantic meaning of the information and therefore enable people to discover what is in a set of AV objects without having to access the content*"
    started in 1996
  - result – MPEG-7 "Multimedia Content Description Interface"
    - not the only standard for AV description, but most general

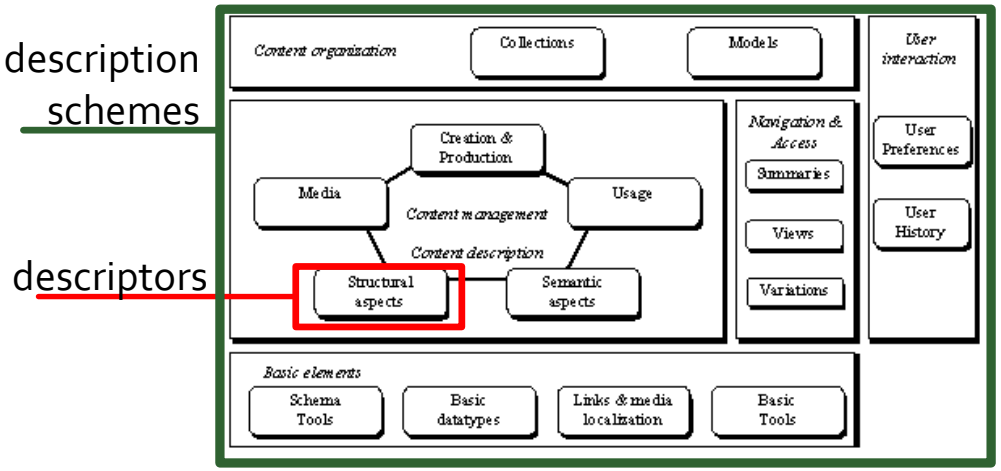
MPEG-7

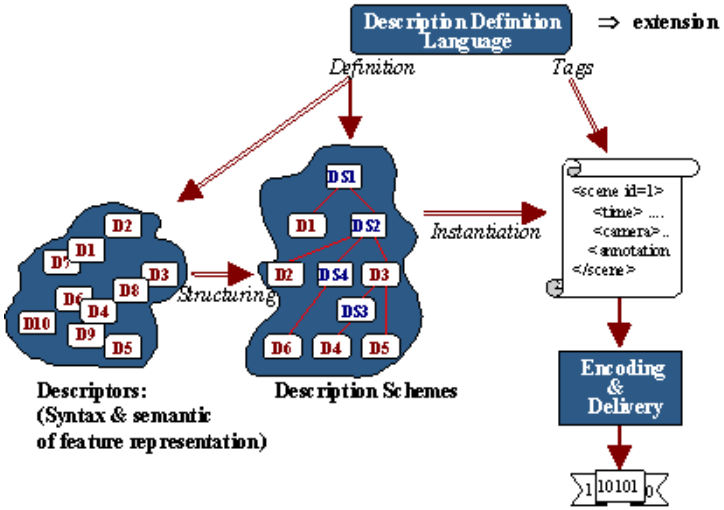
- objectives
  - standard for describing multimedia content by a rich set of tools for content management, organization, navigation and processing
    - different from MPEG-1/2/4 (not a standard for data, but for **metadata**)
  - AV information that MPEG-7 deals with are:
    audio, voice, video, images, graphs and 3D models
    - considered also textual metadata, camera parameters, etc.
  - independence between description and the information itself
  - MPEG-7 standards shall support as broad a range of applications as possible



MPEG-7

- MPEG-7 = standard ISO/IEC 15938
- tools
  - **DDL** – description definition language (XML-based)
  - **description schemes** – structure and semantics of relations between descriptors and description schemes
  - **descriptors** – textual and non-textual description of multimedia content





MPEG-7 – visual descriptors

- defined in Part 3 (ISO/IEC 15938-3)

- general visual descriptors
  - low-level descriptors based on color, texture, shape, motion
  - color: **DominantColor, ScalableColor, ColorLayout, ColorStructure,** Group-Of-Frame/Group-Of-Features (in lecture on video retrieval)
  - texture: **HomogeneousTexture, TextureBrowsing, EdgeHistogram**
  - shape: **RegionShape, ContourShape** (in lecture on shape retrieval)
  - motion, localization (in lecture on video retrieval)
- application-dependent (high-level) visual descriptors
  - face descriptor

MPEG-7 – visual descriptors

- implementation
  - algorithms/code for MPEG-7 feature extraction is not included in the MPEG-7 standard, left to the industry
  - some open-source implementations
    - img(Rummager), in C#
    - Caliph & Emir, in Java
    - MP7JRS C++ Library
  - XM Feature Extraction Web Service
    - former XM library in C++

MPEG-7 – visual descriptors

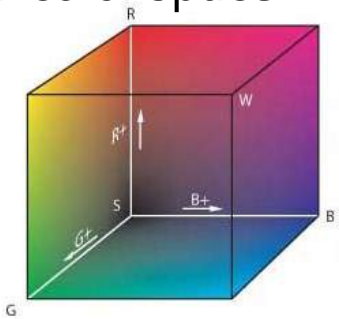
- quantitative evaluation during the MPEG-7 standardization process
  - evaluation image database with ground-truth
    - Common Color Dataset (CCD)
    - Common Color Queries (CCQ)
      D. Zier, J.R. Ohm. Common Datasets and queries in MPEG-7 color core experiments, TR#13-01, 1999
  - ranking function penalizing false dismissals, bad rank, etc.
    - ANMRR (average normalized modified retrieval rate)
  - the result set of descriptors passed ANMRR as the best

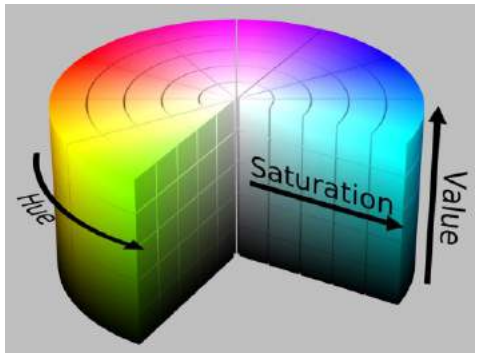
Color descriptors

- the most essential visual information perceived by human
- advantages
  - transformation invariant (rotation, scale, translation)
  - easy to extract
- supporting descriptors used with the content descriptors
  - color space
    - linear transformation matrix from RGB + monochrome color space
    - RGB, YCbCr, HSV, HMMD
  - color quantization
    - uniform, HMMD, lookup table
  - predefined by MPEG-7, but content descriptors can use specific color/quantization

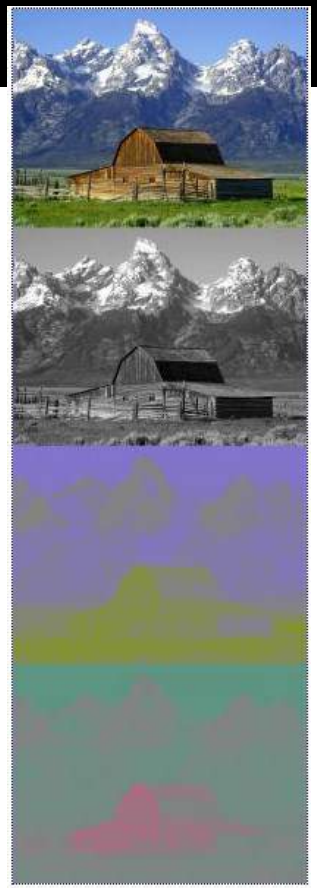
Fundamentals – Color Space

- MPEG-7 helper descriptor
- color model = mathematical model describing the way colors can be represented as tuples of numbers (e.g., RGB)
- color space (incl. gamut) = color model + mapping to physical color space
- four color spaces in MPEG-7 Color Space descriptor
  - RGB
    - red, green, blue
    - good for final reproduction (display), not for modeling
  - HSV (HSB)
    - hue, saturation, value (brightness)
    - suitable for artists (choosing hue, refining the rest)
    - symmetric with the RGB cube, so unrelated to human perception (improved by CIE Lab, HMMD)

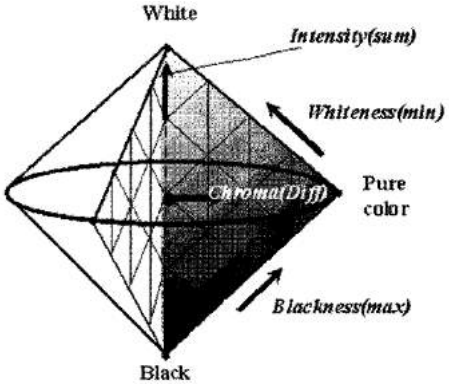




Fundamentals – Color Space



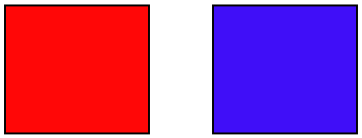
- YCbCr
  - luminance, chroma-blue, chroma-red
    - luminance dominant information (for human)
    - used in image compression – luminance high resolution/low compression, chromas low resolution/higher compression
  - perceptually more meaningful than RGB (luminance = B&W most important)
- HMMD
  - 4 components
    - hue (the same as in HSV space)
    - minimum and maximum over R, G, B values
    - difference between minimum and maximum (redundant)
  - histogram quantized to 256 (H), 128 (M), 64(B) and 32 (D) bins (8-bit)
  - very effective in perception – suitable for MPEG-7

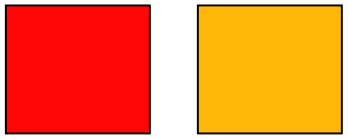


- additional spaces allowed in Color Space (transformation from RGB)

Fundamentals – Color Space

- the goal
  - perceptual closeness of colors corresponds to Euclidean distance in the color space





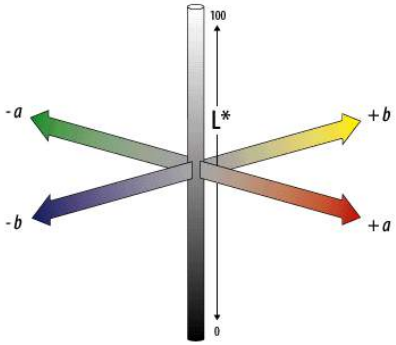
Fundamentals – Color Quantization

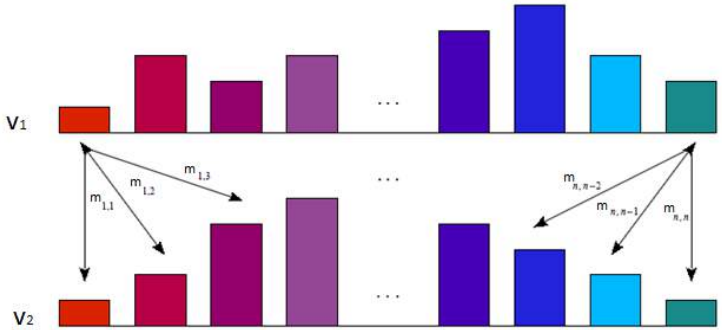
- MPEG-7 helper descriptor
- reduction of the number of unique colors in an image
- quantization types
  - linear/uniform – color value divided into equal intervals
  - non-linear (for HMMD)
  - lookup table – instant quantization
- only Dominant Color descriptor
  uses Color Space and Color Quantization
  - other descriptors use their own quantization and use Color Space only for user-defined transformation from RGB space

Fundamentals – color similarity

- quadratic form distance
  - matrix A – correlations of colors (histogram dimensions)
    - assuming quantized colors
    - each color bin has a representative point in CIE Lab space
    - correlation of color is d_{ij} = L2 distance between color representatives i,j

$$QFD_A(u,v) = \sqrt{(u-v)A(u-v)^T}$$





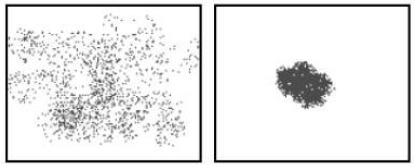
$$A_{ij} = 1 - \frac{d_{ij}}{d_{max}}, \text{where } d_{max} = max_{i,j}(d_{ij})$$

Dominant Color Descriptor (DCD)

- compact description of representative colors

$$\mathbf{DCD} = \{(c_i, p_i, v_i), s\}, \quad (i = 1,2,\ldots,N)$$

- N = number of dominant colors (max 8 observed as sufficient)
- for each color (*i*):
  - (c_i) vector of *C* color components, e.g., 3D vector for RGB color space
  - (p_i) percentage of pixels in the image corresponding to color c_i
  - (v_i) variance of the color values of the pixels in a cluster around the corresponding representative color (optional DCD component)
- (s) spatial coherency – spatial homogeneity of pixels associated with each representative color
  - high value means that pixels of similar color are co-located



Dominant Color Descriptor (DCD)

- extraction procedure – Generalized Lloyd Algorithm
  - determines clusters of pixels, N, etc.
- encoding of DCD in bits: (3,N*(C*(1-12,1),5),5) for $N,((c_i,v_i),p_i),s$
- measuring similarity given 2 DCDs F_1, F_2
  - when ignoring the variance and spatial coherence

$$D_{\mathrm{DC}} = \sum_{i=1}^{N_1} p_{1i}^2 + \sum_{j=1}^{N_2} p_{2j}^2 - \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} 2a_{1i,2j}p_{1i}p_{2j}$$

where $a_{k,l}$ = similarity of colors c_k,c_l

  - when including spatial coherence

$$D_{\mathrm{DC}}^{\mathrm{S}} = w_{\mathrm{a}}|s_1 - s_2|D_{\mathrm{DC}} + w_{\mathrm{b}}D_{\mathrm{DC}}$$

where w_1,w_2 are fixed weights

  - when including also the variance
    - using mixture of Gaussian distribs with params color values and color variance

$$D_{\mathrm{DC}}^{\mathrm{V}} = \sum_{i=1}^{N_1}\sum_{j=1}^{N_1} p_{1i}p_{1j}f_{1i1j} + \sum_{i=1}^{N_2}\sum_{j=1}^{N_2} p_{2i}p_{2j}f_{2i2j} + \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} p_{1i}p_{2j}f_{1i2j}$$

where f is really messy function on c_i,v_i ☺

DCD – example visualisation

DCD = $\{(c_i, p_i, v_i), s\}$, $(i = 1,2,\ldots,N=7)$

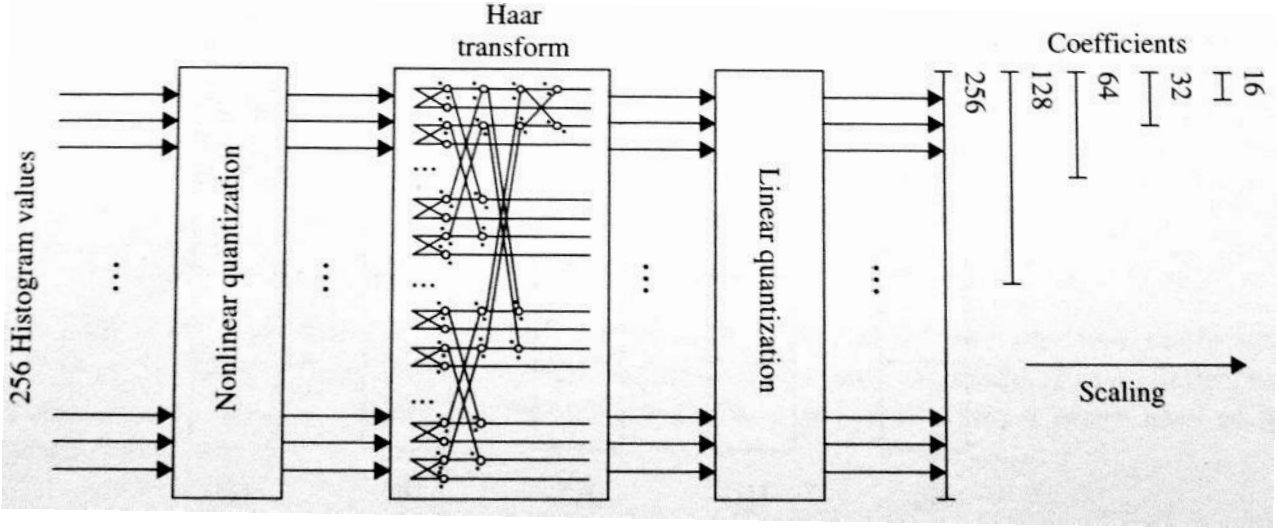
Scalable Color Descriptor (SCD)

- SCD = Haar transform-based encoding scheme applied on histogram in HSV color space
  - i.e., SCD is histogram represented in a compact and scalable way (16-256 bins)
- initial quantization of the 256-bin color histogram
  - H (hue) quantized to 16 bins, S (saturation), V (value) quantized to 4 bin each
    - i.e., 16*4*4 = 256 bins
- the encoding scheme is a series of 1-D Haar transforms
  - resulting in 16 low-pass coecients and 240 high-pass coefficients
  - high-pass coefficients tend to have low values, so they can be discarded
    - 256 bins → 128 or 64 or 32 or 16
    - and/or quantized to small number of bits
  - L1 distance used, but any other histogram distance can be used

$$D_{\mathrm{SC}} = \sum_{i=1}^{N} |\mathbf{H}_{\mathrm{A}}[i] - \mathbf{H}_{\mathrm{B}}[i]|$$

Scalable Color Descriptor (SCD)

- basic unit of Haar transform
  - high-pass coefficient = information at finer resolution levels – there is high redundancy here (small difference values),because of slight variation in colors (variable illumination and shadowing)
  - hence, truncating high-pass coefficients to a few bits is sufficient
- diagram of SCD generation



SCD – visualization example

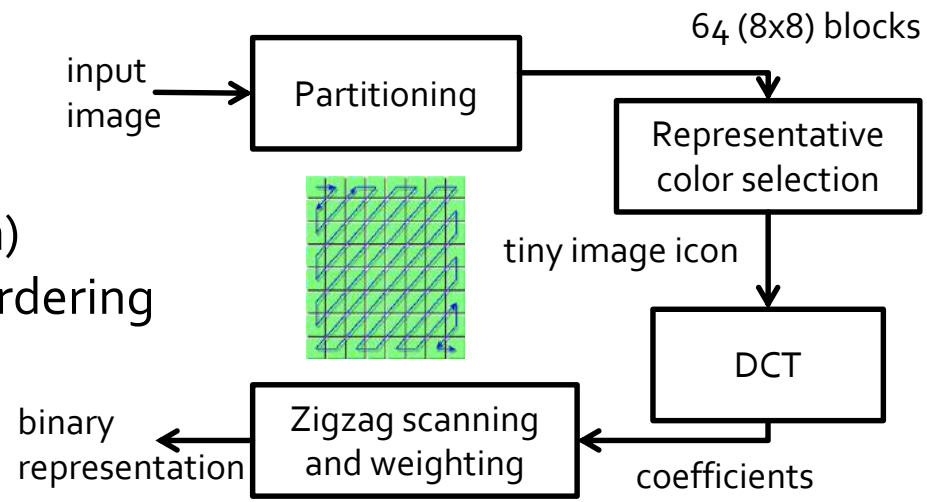
16 coefficients

64 coefficients

256 coefficients

Color Layout Descriptor (CLD)

- compact representation of the spatial distribution of colors within an image
  - especially useful for query-by-sketch and video segment identification (faster mathing and more compact descriptors than the others)
- four steps (inspired by JPEG encoding)
  - downsizing the image into 8x8 icon, i.e.,
    - partitioning the image to 8x8 grid (YCbCr color space used)
    - representative color selection (simple average in grid cell is ok)
  - DCT (discrete cosine transform) per color channel + zig-zag reordering of DCT coefficients
  - low-freq. DCT coeff. preserved and non-linearly quantized to form the descriptor



Color Layout Descriptor (CLD)

- **CLD = (*N*, *NY*, *NC*, *Y[]*, *Cb[]*, *Cr[]*))**, where
  N is the pattern of DCT coefficients used (1-2 bits),
  N = 1 → 6 coeffs. for luminance and 3 coeffs. for chrominances
  N = 2 → 6 coeffs. for luminance and also chrominances
  N = 3 → ***NY*** coeffs. for luminance and ***NC*** coeffs. for chrominances,
  while using 3-bit values ***NY***, ***NC*** ∈ {3, 6, 10, 15, 21, 28, 64}
  Y[],Cb[],Cr[] are arrays coefficients (quantized to 5-6 bits each per coeff.)
  - the descriptor could be just 64 bits, the most compact MPEG-7 descriptor

- measuring similarity given 2 CLDs {Y, Cr, Cb} and {Y′, Cr′, Cb′}

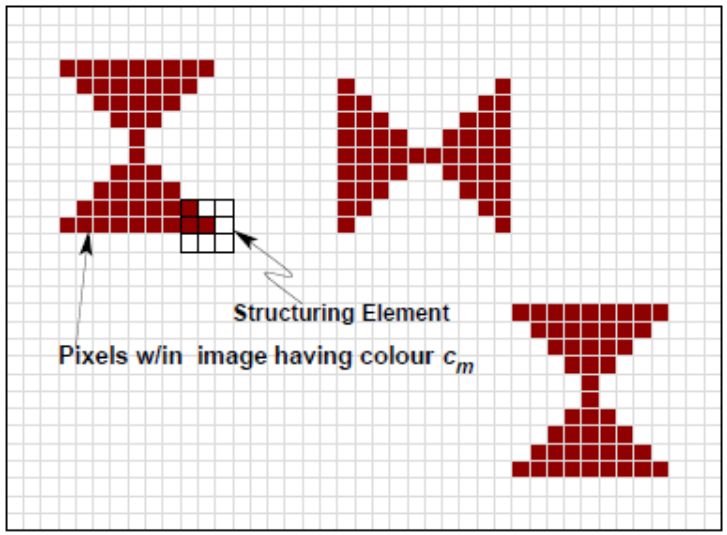
$$D_{\mathrm{CL}} = \sqrt{\sum_i w_i^y (Y_i - Y_i')^2} + \sqrt{\sum_i w_i^b (Cb_i - Cb_i')^2} + \sqrt{\sum_i w_i^r (Cr_i - Cr_i')^2}$$

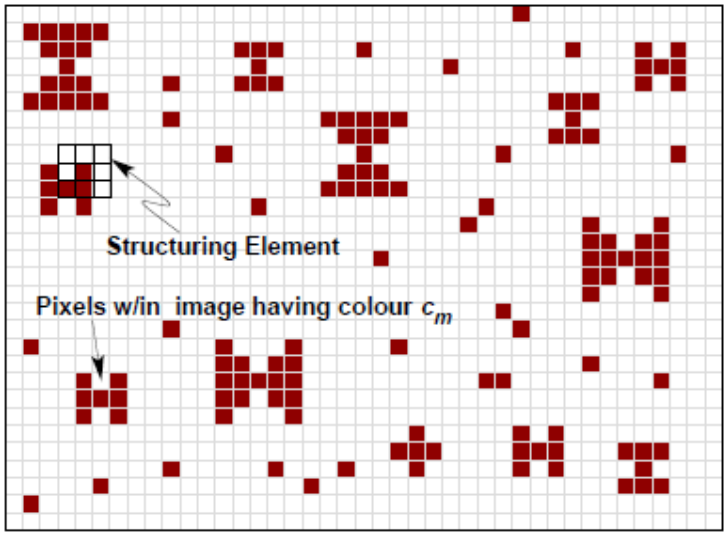
where *i* represents the zigzag-scanning order of coefficients

Color Structure Descriptor (CSD)

- CSD represents an image by both color distribution and local spatial structure of the color
  - sensitive to features for which histogram (i.e., also DCD, SCD,CLD) is blind
  - structure = degree to which pixels of a particular color are clumped together within a structuring element
    - structuring element = 8x8 window in the image

high color structure





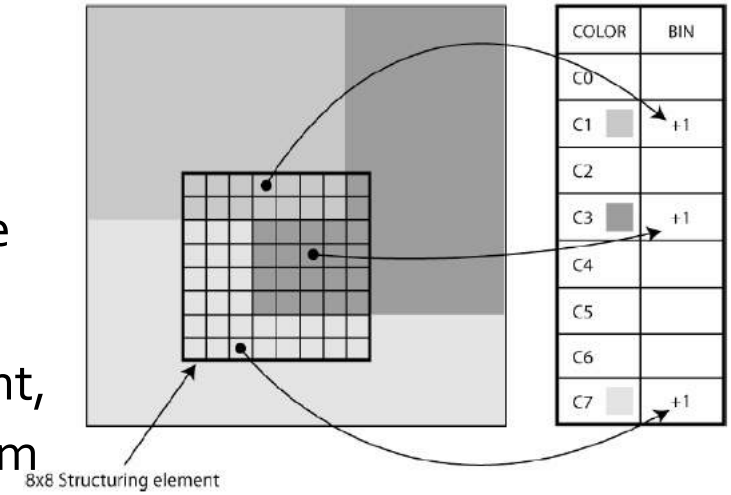
low color structure

Color Structure Descriptor (CSD)

- descriptor is specific color histogram

  $$CSD = h_s(m), \quad m \in \{1,\ldots,M\}$$

  where M is number of bins, s is the scale of square structuring element, hs
  - at each position of the structuring element, its content is used to update the histogram (**not quantity, but only presence of color is updated, i.e., +1**)
  - then normalization and non-linear quantization



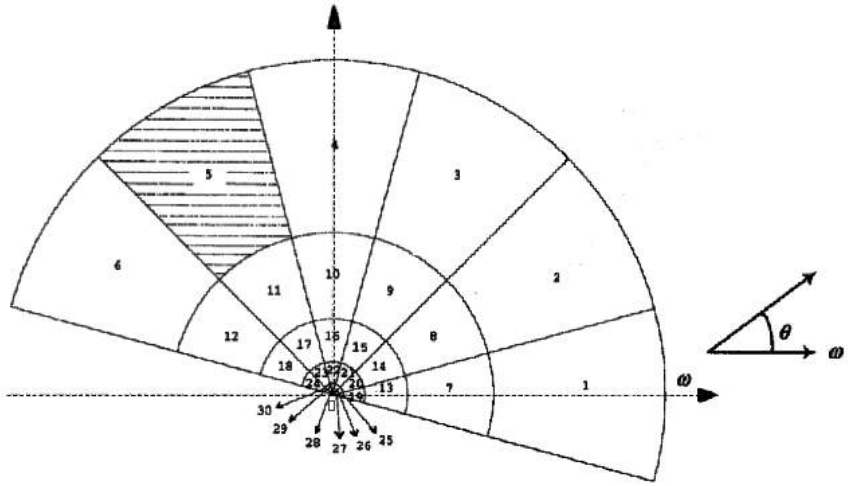
- CSD is sensitive to the quantization, so only standard procedure of descriptor extraction ensures interoperability
- measuring similarity of 2 CSDs
  - L1 by default, any other histogram distance could be used (QFD)

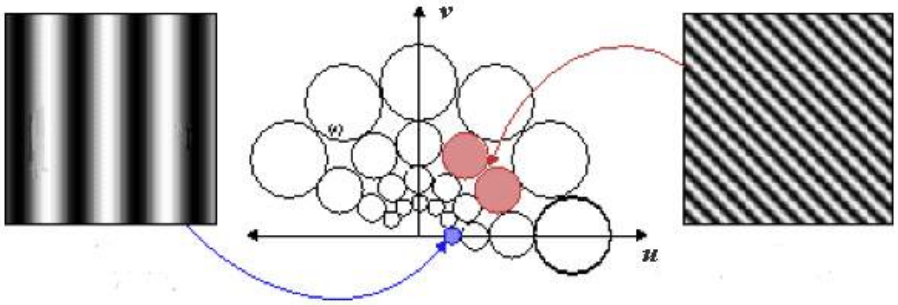
Texture features

- texture featuring corresponds well to some results from psychophysical experiments
  - the response of the visual cortex is turned to a band-limited portion of the frequency domain
  - human brain decomposes the spectra into perceptual channels that are bands in spatial frequency
- applications in
  - aerial photography (agriculture, geography)
  - fabric/material images
  - medical imaging
  - remote sensing
  - industrial inspection

Homogeneous Texture Descriptor (HTD)

- homogeneous texture is assumed
- image is filtered with a bank of orientation- and scale-sensitive Gabor filters
  - octave-bands (4-5) along the radial direction,
    equal-width angles along the angular direction (6 per 30 degrees)
  - multi-resolution decomposition of pattern frequencies





Homogeneous Texture Descriptor (HTD)

- descriptor components

  HTD = $\{f_{DC}, f_{SD}, e_1, e_2, \ldots, e_{30}, d_1, d_2, \ldots, d_{30}\}$
  - the means e_i and the standard deviations d_i of the filtered outputs in the spatial frequency domain (5 scales × 6 orientations per scale = 30) used as descriptor components
    - 2 x 30 = 60 features
  - also the mean intensity f_{DC} and the standard deviation f_{SD} of the image
  - hence, total 62-dimensional vector (8-bit coordinates)
- measuring similarity of 2 HTDs
  - L1 distance or histogram distance (closer banks = similar frequencies)

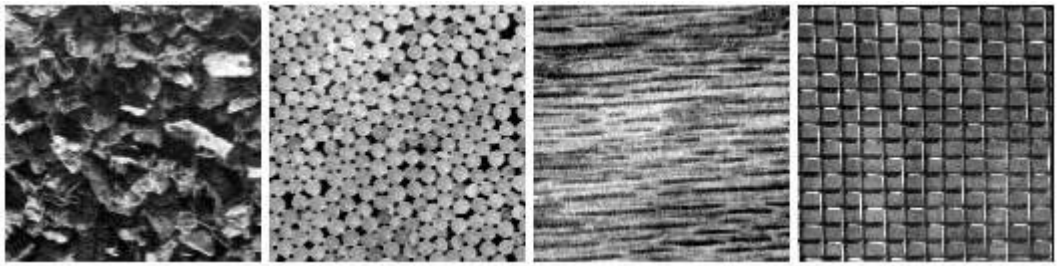
Texture Browsing Descriptor (TBD)

- 12-bit descriptor describing visual texture perception

  TBD = {v_1, v_2, v_3, v_4, v_5}

  - v_1 – regularity (2 bits) – four levels of regularity (irregular to periodic pattern)
  - v_2,v_3 – directionality (2 x 3 bits) – 6 levels per direction, 0° to 150° in steps of 30°
  - v_4,v_5 – coarseness (2 x 2 bits) – 3 levels per direction (fine grain to coarse texture)

- useful also for browsing (filtering), human user can specify query
- example



irregular

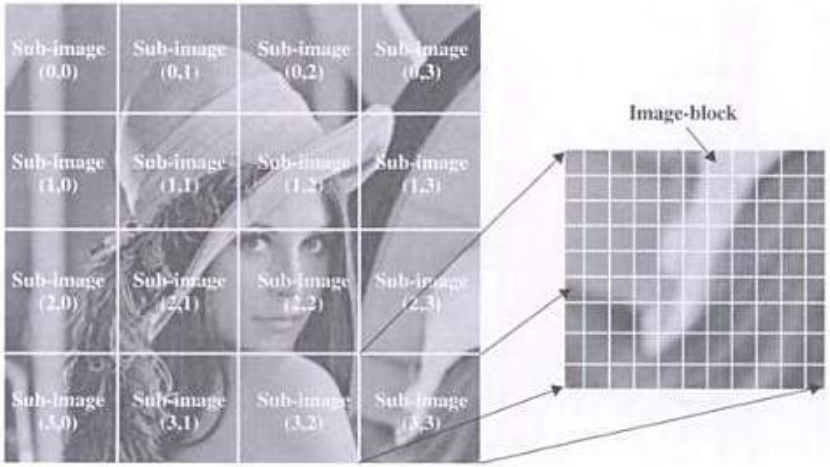
periodic pattern, horizontal + vertical directionality

Texture Browsing Descriptor (TBD)

- image filtered with a bank of orientation- and scale-tuned Gabor filters as in HTD
- dominant direction estimation
  - *S* directional histograms is computed (varying scale, e.g., S=4)
  - direction *k* is dominant if there is a peak in histogram at scale *s*, where there are peaks for the same direction also in histograms at scales *s-1* and *s+1*
  - among all the **dominant directions** the two are selected with sharpest peaks
- regularity and coarseness estimation
  - from the filtered images two projections are computed by Radon transformation using the two dominant directions (obtained in the previous step)
  - normalized autocorrelation function (NAC) computed for each projection
    - the peaks and valleys in NAC(*k*) are determined and a set of points [mean, std deviation] is obtained
    - agglomerative clustering of the points is performed and those the max **contrast** are selected for TBD
  - **regularity** is obtained by assigning credits to candidate projections

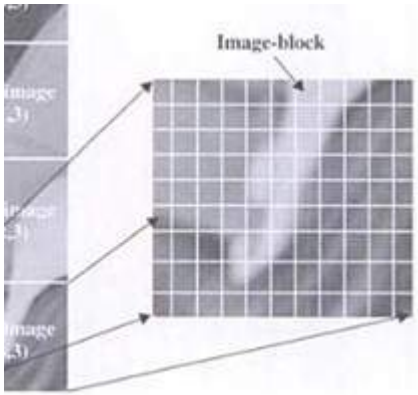
Edge Histogram Descriptor (EHD)

- local-edge distribution in image
- image partitioned into 4 x 4 subimages (grid)
  - local-edge distribution for each subimage
  - 5-bin histogram (5 directions)
    - no direction
    - vertical
    - horizontal
    - 45 degree
    - 135 degree
  - 80-bin (5*16) histogram whole image



Edge Histogram Descriptor (EHD)

- descriptor extraction
  - each subimage is divided into fixed number of blocks, while each block is treated as 2x2 pixel image (averaging each of the 2×2 partitions)
  - simple edge detectors (four directional filters and one isotropic filter) applied to these 2x2 image
    - detector with max edge strength identified, if not strong enough (threshold), the block is non-directional
    - appropriate bin for the subimage is incremented in the 80-bin hist.
  - non-linear quantization to 3 bits per bin, i.e., 240 bits (30 bytes) per descriptor



$$\begin{bmatrix} 2 & -2 \\ -2 & 2 \end{bmatrix}$$

Directional selective

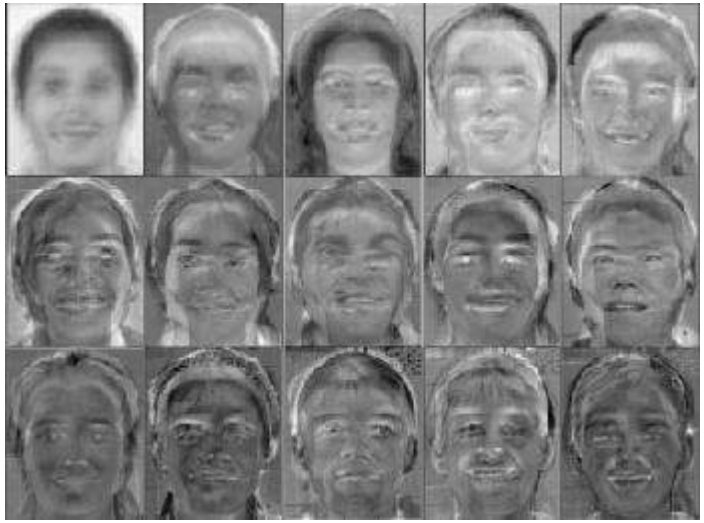
$$\begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} \sqrt{2} & 0 \\ 0 & -\sqrt{2} \end{bmatrix} \quad \begin{bmatrix} 0 & \sqrt{2} \\ -\sqrt{2} & 0 \end{bmatrix}$$

- scale-invariant descriptor
  - supports both rotation-sensitive and rotation-invariant matching
- L1 distance

Face descriptor

- application-specific descriptor included in MPEG-7
  - descriptor = vector of eigenface coefficients
    - eigenfaces = specific visual words
  - each image of a face is described as a linear combination of eigenfaces

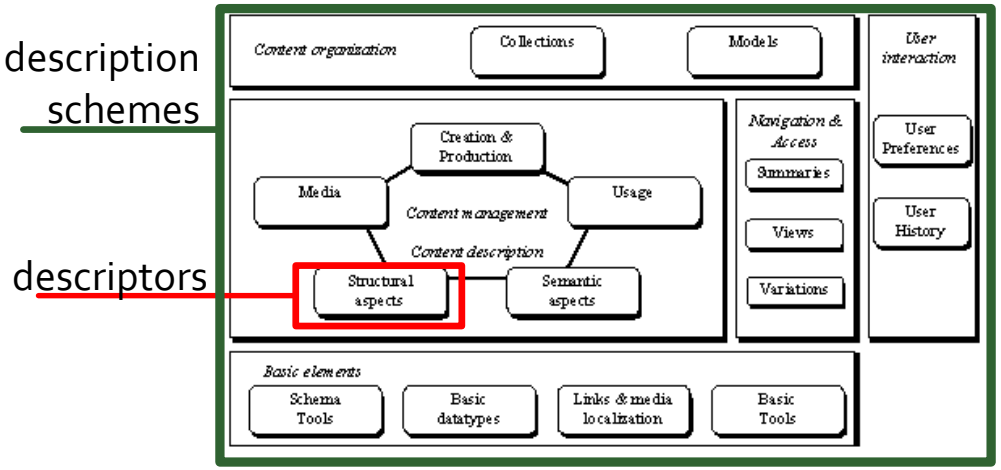


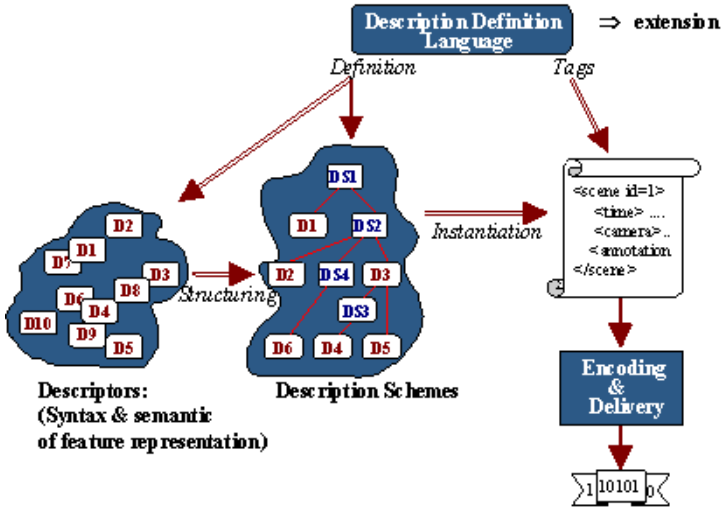
Face descriptor

- eigenface "vocabulary" construction
  - consider a training collection of *n* images
  - take every image in gray scale consisting of *m* pixels and form a column of a matrix, obtaining *m***n* matrix A
  - decomposition of A using principal component analysis (PCA), obtaining a new base of eigenvectors (eigenfaces)
- indexing/querying
  - consider a collection to be indexed/searched and a query image
  - each image from the collection (and also the query image) is projected from the pixel domain into eigenfaces domain (using the eigenvector matrix)
- measuring similarity
  - the same as in vector model/bag-of-features, i.e., cosine measure, L2

Combination of descriptors

- could be defined using description schemes





Combination of descriptors

CORRELATION BETWEEN THE MPEG-7 DESCRIPTORS.

Descriptor	Color Layout	Color Structure	Edge Histogram	Hom. Texture	Scalable Color
Color Layout	1.00	**0.23**	0.10	0.06	**0.45**
Color Structure	**0.23**	1.00	**0.24**	0.09	**0.67**
Edge Histogram	0.10	**0.24**	1.00	**0.23**	0.18
Hom. Texture	0.06	0.09	**0.23**	1.00	0.09
Scalable Color	**0.45**	**0.67**	0.18	0.09	1.00

source: Batko et al., CoPhIR Image Collection under the Microscope, SISAP 2009

- CoPhIR database
  - 100 million images downloaded from Flickr.com
  - 5 MPEG-7 descriptors extracted
    - Scalable Color, Color Structure, Color Layout, Edge Histogram, Homogenous Texture
    - overall distance function *D*, aggregating descriptor-specific distances d_i using a monotonous function *t*

$$D(x,y) = t(d_1(x,y), \ldots, d_m(x,y))$$

$$t(x_1, \ldots, x_m) \leq t(x'_1, \ldots, x'_m) \text{ when } x_i \leq x'_i$$

  - retrieval accuracy
    - trying different aggregating functions on the $m = 5$ distances d_i
    - user satisfaction evaluation (1 best, 6 worst user satisfaction) – evaluating accuracy of similarity queries by user satisfaction survey
    - ANMRR evaluation (0 all found, 1 none found)

Combination of descriptors

- two variants of *f*
  - weighted sum aggregation

$$t(x_1, ..., x_m) = w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_m \cdot x_m$$

MPEG-7 descriptor	Weight
Scalable Color	2.5
Color Structure	2.5
Color Layout	1.5
Edge Histogram	4.5
Homogeneous Texture	0.5

USER SATISFACTION EVALUATION.

Dataset	Satisfaction	Queries	Users
10M	2.63	446	31
100M	2.17	401	28

ANMRR EVALUATION.

Dataset	ANMRR	# improved	# worsen
10M	0.49	-	-
100M	0.41	62 %	28 %

  - logarithmic aggregation

$$t(x_1, ..., x_m) = w_1 \cdot \log_{b_1}(x_1+1) + \cdots + w_m \cdot \log_{b_m}(x_m+1)$$

ANMRR EVALUATION FOR LOGARITHMIC AGGREGATION.

Dataset	ANMRR	# improved	# worsen
10M	0.49	-	-
logarithms	0.43	55 %	25 %

course:
Multimedia Retrieval

lecture 7:

Global image descriptors – deep learning

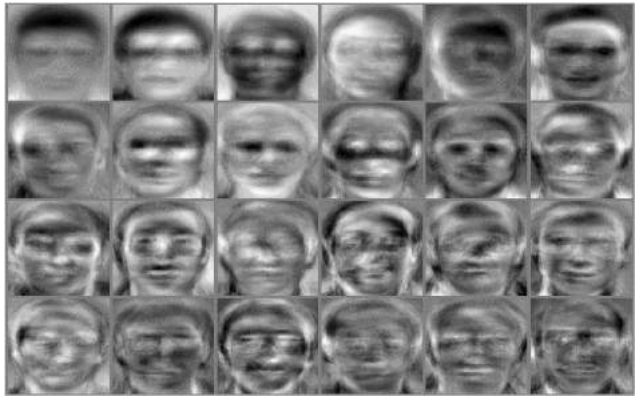
prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

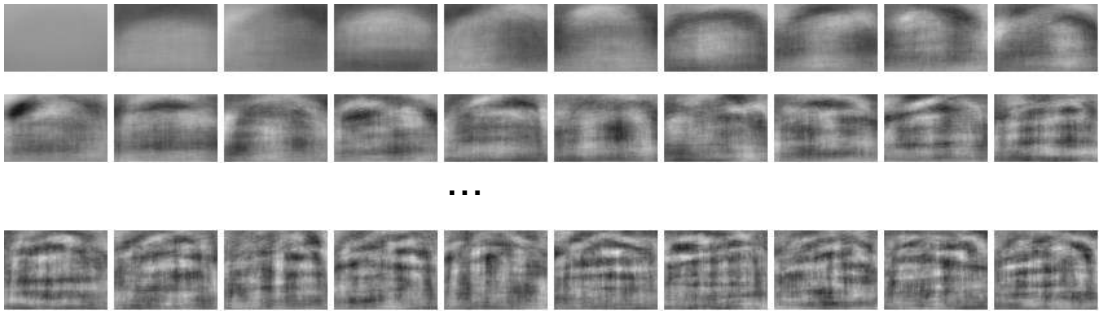
- semantic-visual words
- convolutional deep neural networks (CNN)
  - semantic-visual image descriptors
- transfer learning

Semantic-visual words

- domain-specific high-level
  - eigenfaces (PCA)



  - "eigenhouses" (SVD)



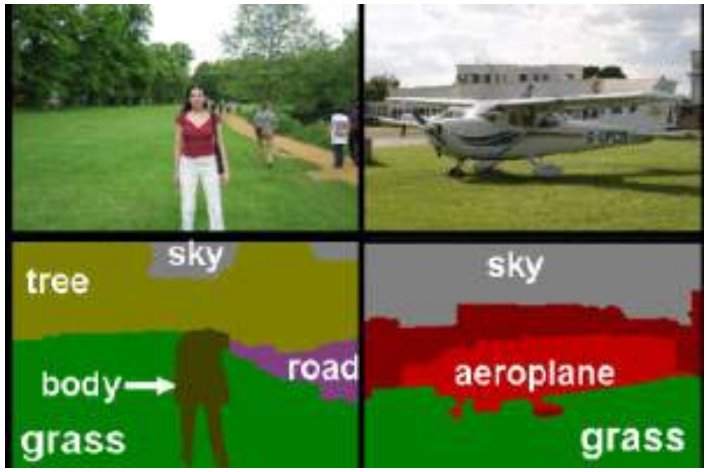
  - global match, restricted input (canonic form)

Semantic-visual words

- general high-level visual words?
  - PCA, SVD don't work locally
  - SIFTs work only in limited domains
    - good for rigid patterns with edges (architecture, picture in picture, solid objects)
    - bad for non-rigid patterns (query by sketch, nature, people, deformable objects, abstract)
  - MPEG7 and other analytic approaches not semantic (high-level)
- decompose the problem
  - complex semantics into multiple common visual features

Semantic-visual words

- often complex scene
  - unlimited camera settings, 3D projection, occlusion, abstract, sketch
- hierarchical decomposition
  - object-wise segmentation?
    - extremely difficult in general
    - good for 3D scene reconstruction, etc.
    - not a solution for retrieval



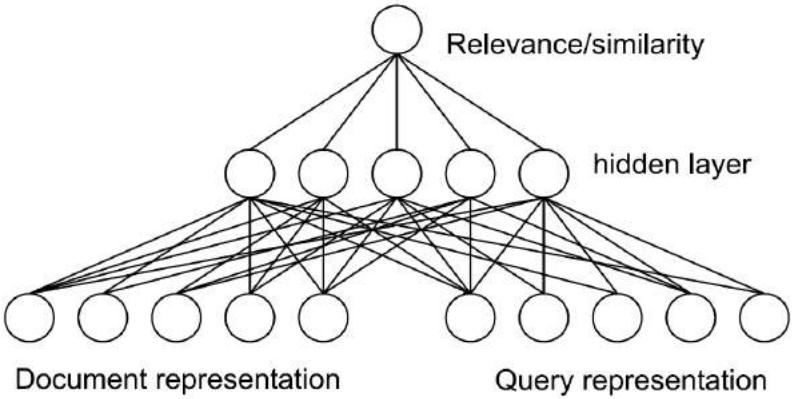
(source: http://www.cs.utexas.edu/~grauman/courses/fall2011/schedule.html)

Semantic-visual words

- where to get the high-level semantics?
  - human annotators → ground truth (data in classes)
  - how?
- let the neural networks do this dirty job "somehow"!
  - supervised learning (also unsupervised for some tasks)
  - a sort of perceptron neural network (NN)
    - back-propagation (gradient descent)
    - training image example + annotation provided by human user
      = the NN "magically" connects the semantics with visual features!
    - well, not that easy ☺

Learning in information retrieval

- early attempts in information retrieval
  - COSIMIR – **CO**gnitive **SIM**ilarity learning in **I**nformation **R**etrieval (T. Mandl, 1998)
    - 3-layer (i.e., shallow) back-propagation fully connected NN
  - not feature extraction, but learning the similarity function
  - pair of document descriptors + annotation (similarity score)
  - tends to overfitting, local extremes, inconsistency
  - and mainly, preprocessing needed: the descriptors must keep the semantics!



Learning in information retrieval

- history = failure@large-scale
  - fully-connected layers
  - limited number of neurons per layer (n^2 parameters)
    - because of computation power
  - therefore, shallow neural networks
  - small training data = overfitting
- today = success@large-scale
  - convolutional and fully-connected networks
  - more neurons per layer (but below n^2 parameters!)
  - GPU technology enabling fast vector parallelism
  - therefore, deep networks
  - large training data
    - e.g., ImageNet, 1.2M training images, 1000 classes

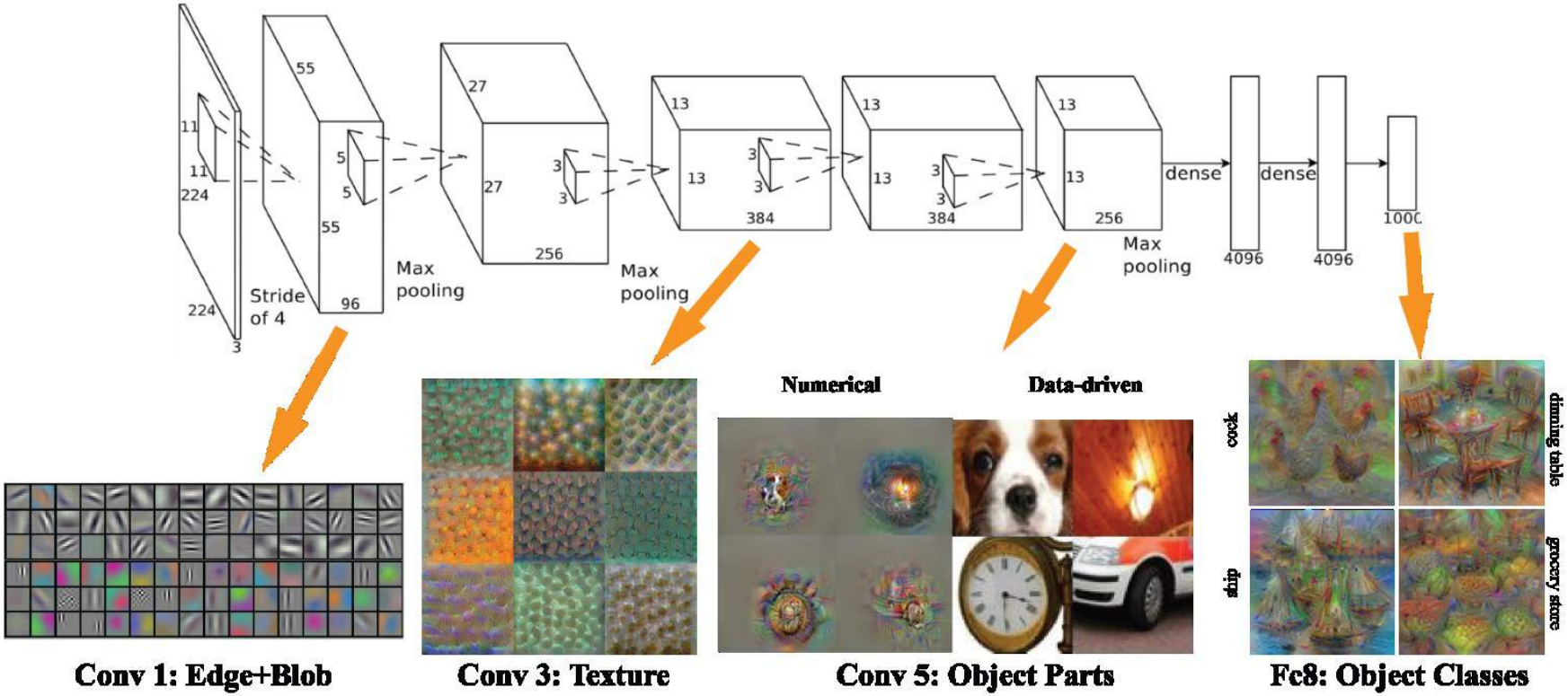
Semantic-visual words in CNN

- convolutional deep neural networks (CNN)
  - by LeCun 1989
  - convolution is hard-wired mechanism in human cognition (visual cortex)
    - where cells in retina ("pixels") is the input layer
  - inspired by this, CNN mimic the visual cortex
    - deep architecture: convolutional layers + fully connected layers
    - training example: raw image (no preprocessing, just scaling) + multiclass annotation
  - hierarchy of visual words
    - from low-level to high-level
    - fully connected layers on the top lead to semantic classes (as the output)

Semantic-visual words in CNN

- convolutional deep neural networks (CNN), cont.
  - the visual words (coded in neuron connection weights) arranged in the CNN such that they are activated regardless of
    - locality
    - scale
  - the last layer = semantic annotation
  - a $(\text{last} - k)_{th}$ layer = semantic-visual descriptor
    - the smaller k, the more semantic/less visual
    - the greater k, the more visual/less semantic

Convolutional deep network

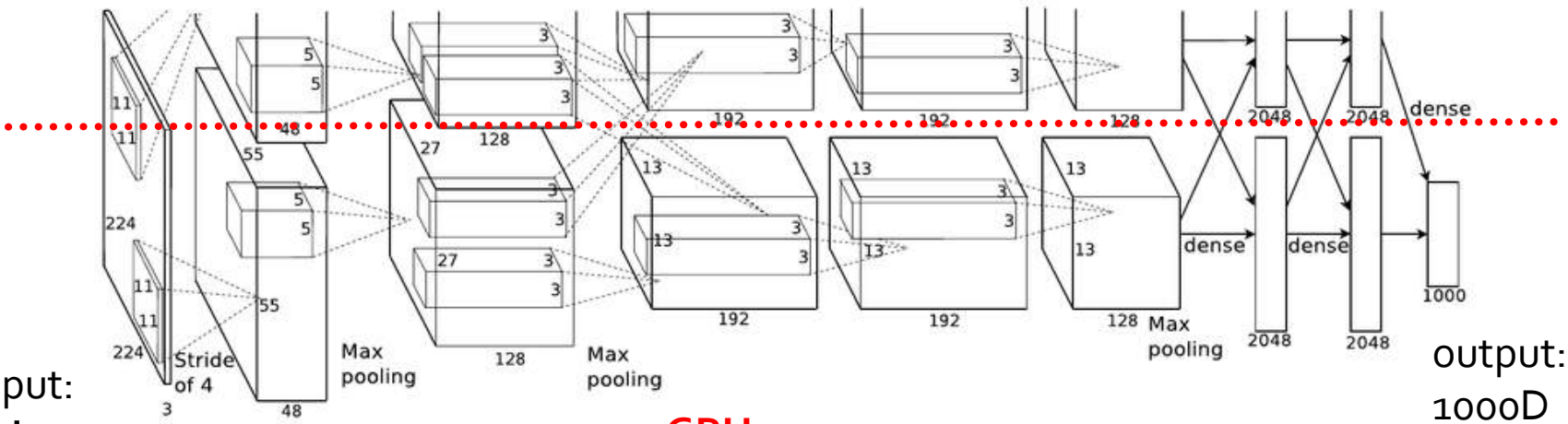


[Fig source: http://vision03.csail.mit.edu/cnn_art/index.html]

Convolutional deep network

example: AlexNet

GPU1



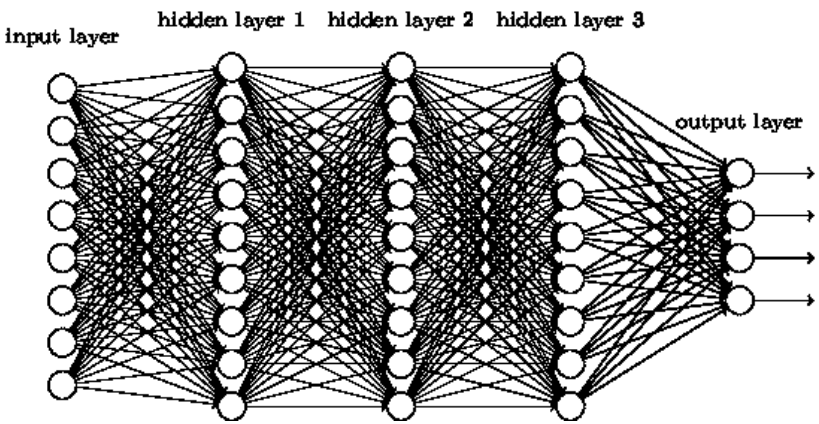
GPU2

input: **color image** scaled to 224x224

output: 1000D vector over **semantic classes**

Convolutional deep network

- why not traditional fully-connected perceptron layers?
  - not fit to image recognition needs
  - too many weights (parameters) vs. too few neurons (feature detectors)
    - prone to overfitting
    - bad recognition effectiveness/resolution
    - computationally expensive
  - global vs. local receptive field
    - not translation-invariant



- convolutional network
  - **actually IS the traditional multi-layer perceptron network!**
  - but constrained and designed for image recognition (annotation) task
    - not fully-connected, receptive fields introduced, shared weight banks (filters), etc.

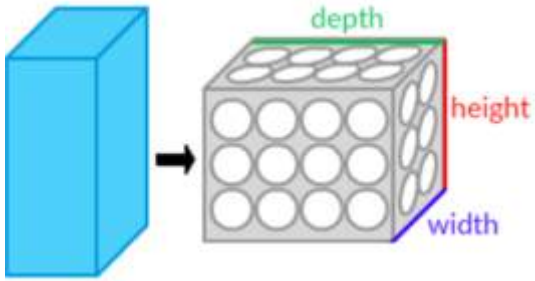
Convolutional deep network

- back-propagation (learning)
  - looks for the minimum of the error (loss) function in weight space using the method of gradient descent
    - applying partial derivative of the loss function w.r.t. w_i or bias
  - requires differentiable activation functions
    - like sigmoid, or treating the indifferentiable points (ReLU)
  - finds local extremes
  - note that in convolutional layers the convolution filters are learned!
    - compare to static convolution filters used for edge detection, etc.

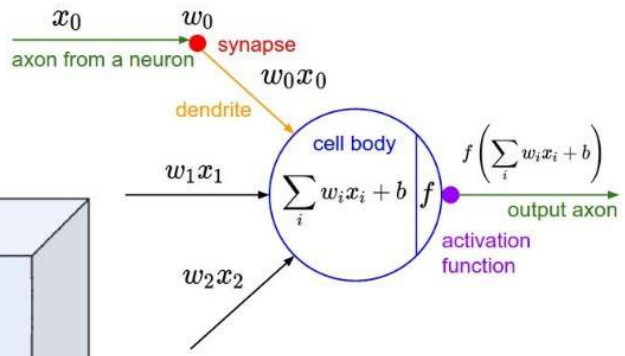
Convolutional deep network

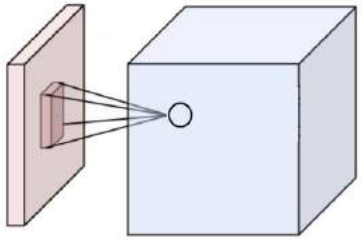
- types of layers
  - data layers (input, output)
  - vision layers
    - convolutional layers, pooling layers
  - activation layers
    - ReLU, Sigmoid, …
  - common layers
    - inner product (fully connected layer), …
  - loss layers
    - softmax, Euclidean, …

Convolutional layer



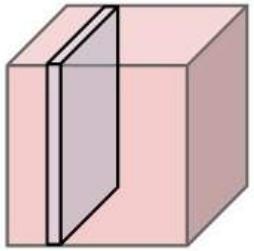
- neurons arranged in 3D block (3D layer)
  - width x height x depth
- each neuron is classic perceptron
  - connected to a set of neurons in the previous layer arranged in receptive field (3D block of neurons)
    - receptive field depth
      = previous layer depth
    - weights w_i of the connections
      = 3D block (filter) of the
      same volume as the receptive field



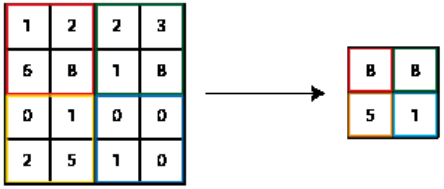


  - neuron is activated as usual
    - computing dot-product of activations x_i of neurons from the receptive field and the filter

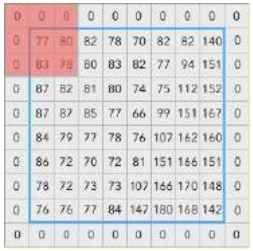
Convolutional layer

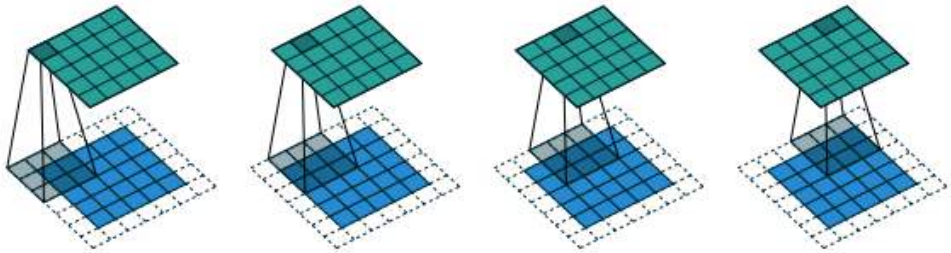


- depth slice
  - neurons in the same depth of layer
    - share the same filter
      - assuming same features may appear regardless of the position in image
  - for each neuron the receptive field slides (in width, height)
    - stride = parameter of sliding
      - 1 is by 1 pixel, 2 is by 2 pixels, so smaller output



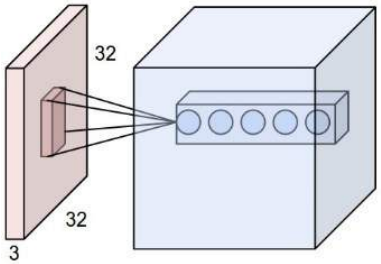
    - zero-padding = thickness of borders filled with zeros
      - extends the sliding region





Convolutional layer

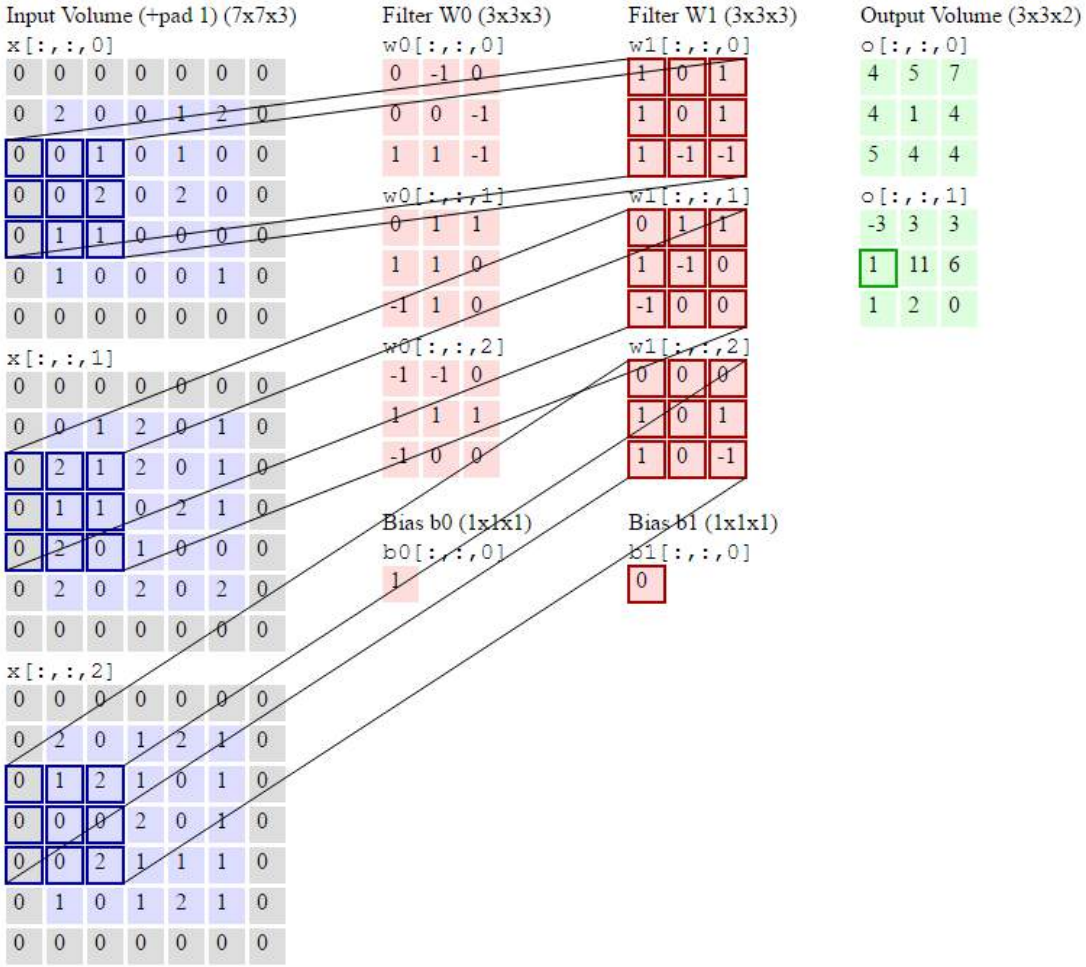
- depth column (fibre)
  - neurons at the same spatial (width, height) position but different depth
    - share the receptive field
      - i.e., different features can be recognized in the same receptive field



- backpropagation
  - also convolution, but with spatially flipped filters

Convolutional layer

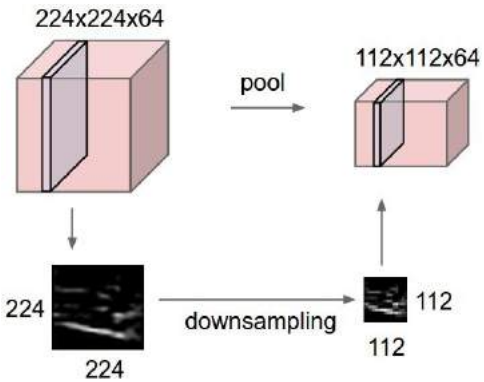
- example
  - stride 2, zero-padding 1

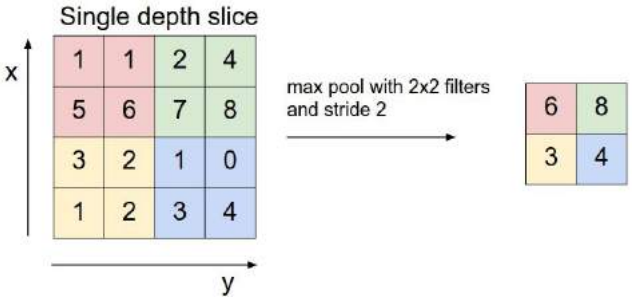


source: http://cs231n.github.io/convolutional-networks/

Pooling layer

- reduces the spatial size of input
  - reason: performance, overfitting
  - placed in between convolutional layers
  - could be omitted
- for every depth slice
  - usually 2x2 filter applied and stride 2
  - usually max pooling (max value taken), could be also avg, L2-norm
- backpropagation
  - routing the gradient to input with the highest value





ReLU (Rectified Linear Unit) layer

- elementwise activation function $f(x) = max(0,x)$
- most popular activation function for deep networks
  - computationally cheap
  - scale-invariant
  - efficient gradient propagation
  - biologically plausible

Fully connected and loss layers

- fully connected layers
  - as in regular networks, neurons connected to all neurons in previous layer
  - activation computed as inner product
- loss layer
  - usually the last layer (in very deep CNNs also in the middle)
  - determines the penalization of predicted and true labels
    - softmax – predicting one class out of *k* (sums to 1)
    - sigmoid cross-entropy – *k* independent probabilities in [0,1]

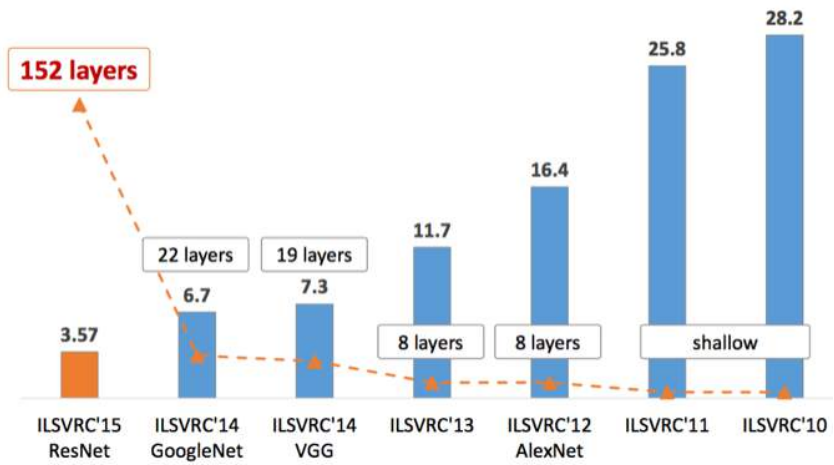
Basic use case – classification

vector of weights/
probabilities to classes
(loss layer)

```
---------- Prediction for ../../../examples/images/bike.jpg ----------
0.5061 - "n03792782 mountain bike, all-terrain bike, off-roader"
0.4462 - "n09193705 alp"
0.0104 - "n09468604 valley, vale"
0.0079 - "n09246464 cliff, drop, drop-off"
0.0065 - "n09472597 volcano"
```

Basic use case – classification

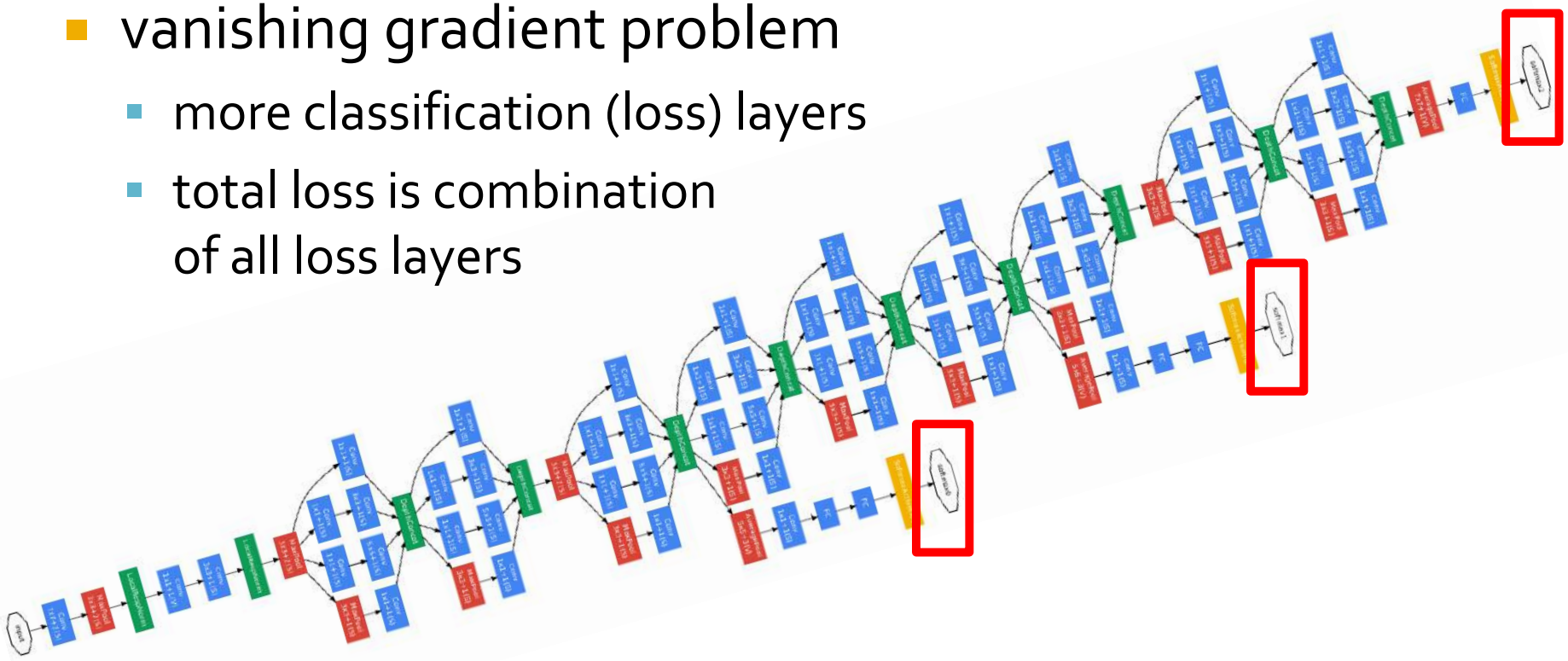
- models (trained CNN) rapidly evolving
  - ILSVRC (ImageNet Large Scale Visual Recognition Challenge)
  - going deeper…



Year	CNN	Developed by	Place	Top-5 error rate	No. of parameters
1998	LeNet(8)	Yann LeCun et al			60 thousand
2012	AlexNet(7)	Alex Krizhevsky, Geoffrey Hinton, Ilya Sutskever	1st	15.3%	60 million
2013	ZFNet()	Matthew Zeiler and Rob Fergus	1st	14.8%	
2014	GoogLeNet(19)	Google	1st	6.67%	4 million
2014	VGG Net(16)	Simonyan, Zisserman	2nd	7.3%	138 million
2015	ResNet(152)	Kaiming He	1st	3.6%	

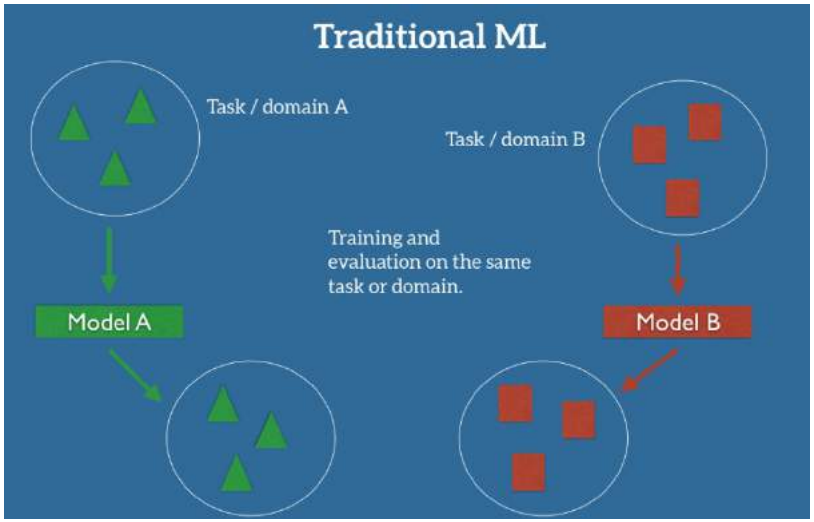
Very deep architectures

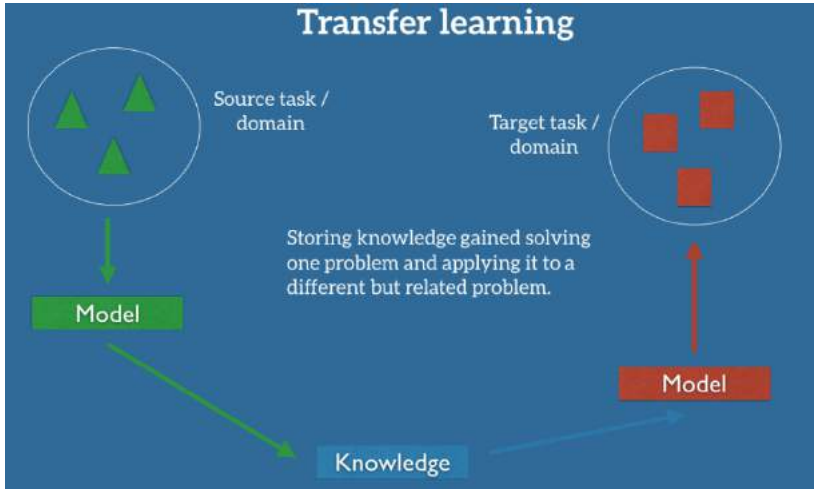
- e.g., GoogLeNet
- vanishing gradient problem
  - more classification (loss) layers
  - total loss is combination of all loss layers



Transfer learning

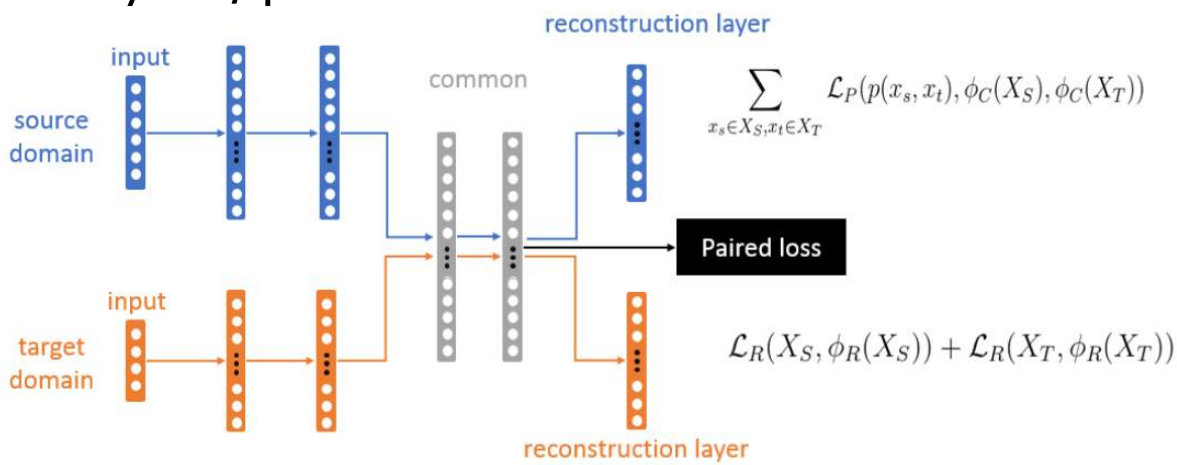
- reusing knowledge acquired in another domain
- why?
  - small data in target domain
  - reliability
  - personalization





Transfer learning in CNN

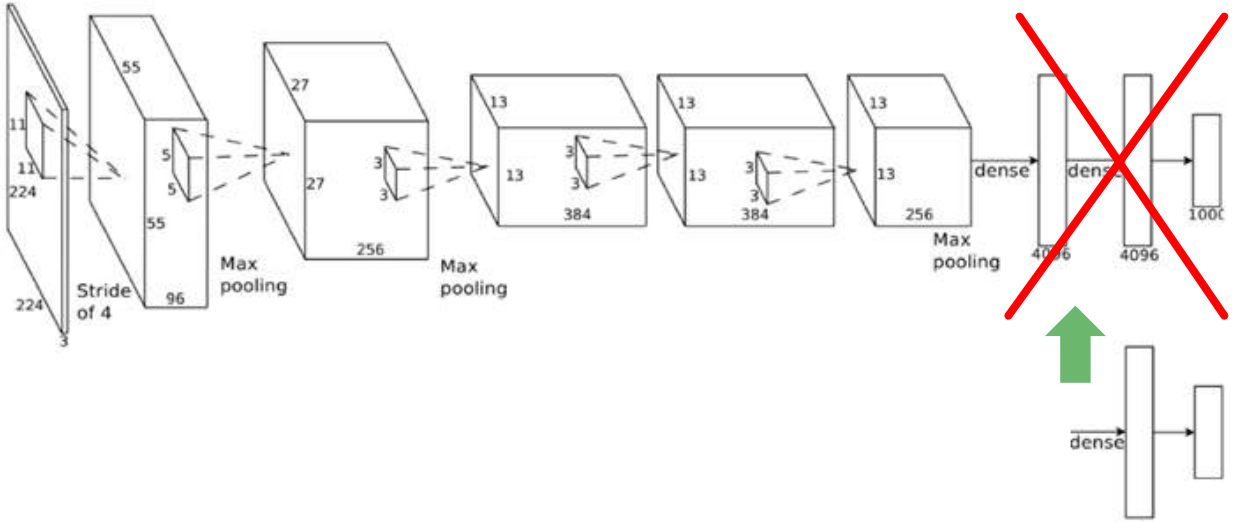
- lower CNN layers contain generic features
- heavy-weighed transfer learning
  - combined CNNs – combination of source domain (pretrained) and target domain CNN
  - shared layers, paired/twin loss



Multimodal Transfer Deep Learning with Applications in Audio-Visual Recognition, Seungwhan Moon, Suyoun Kim, Haohan Wang, arXiv:1412.3121

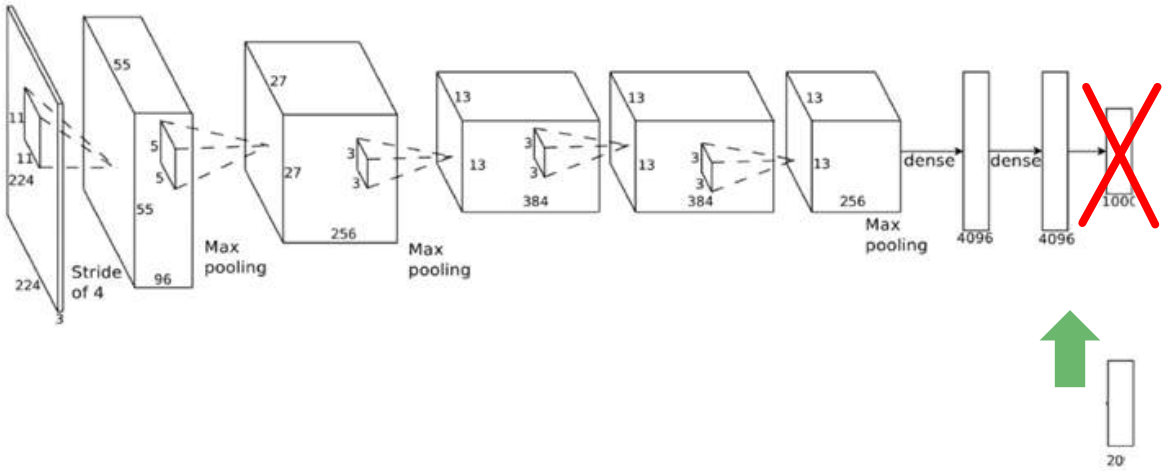
Transfer learning in CNN

- light-weighed transfer learning
  - fine-tuned CNN - taking an already learned model, adapting the architecture, and resuming training from the already learned model weights

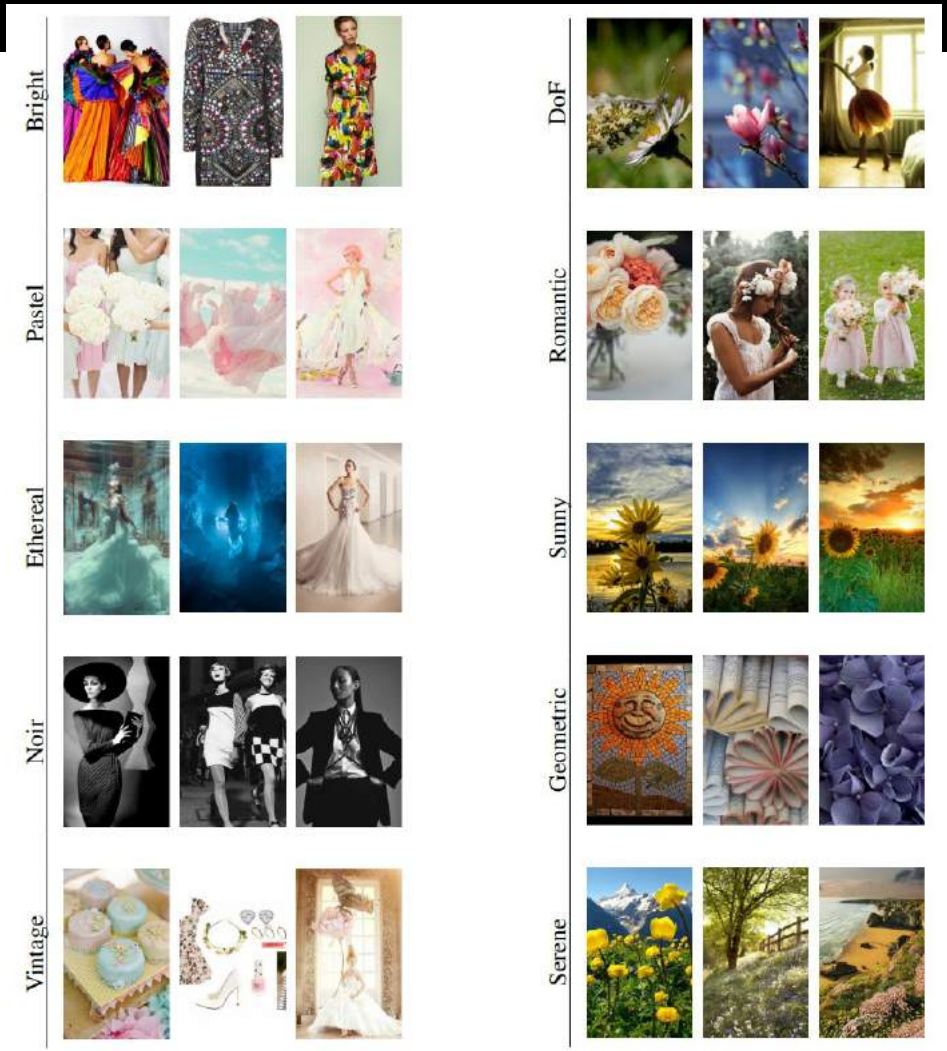


Transfer learning in CNN

- example Flickr style fine-tuning
  - take AlexNet model (trained on 1.3M generic images)
  - replace classification layer (1000 class neuron) by another (20 style class neurons) + random init.
  - train (fine-tune) by smaller training data (80K Flickr images, 20 style classes)



Transfer learning in CNN



Query: "dress". Query: "flower".

Transfer learning in CNN

Figure 4: Cross-dataset style. On the left are shown top scorers from the Wikipaintings set, for styles learned on the Flickr set. On the right, Flickr photographs are accordingly sorted by Painting style. (Figure best viewed in color.)

Frameworks for deep learning (2019)

Software	Creator	Initial Release	Software license[1]	Open source	Platform	Written in	Interface	OpenMP support	OpenCL support	CUDA support	Automatic differentiation[1]	Has pretrained models	Recurrent nets	Convolutional nets
BigDL	Jason Dai (Intel)	2016	Apache 2.0	Yes	Apache Spark	Scala	Scala, Python			No		Yes	Yes	Yes
Caffe	Berkeley Vision and Learning Center	2013	BSD	Yes	Linux, macOS, Windows[2]	C++	Python, MATLAB, C++	Yes	Under development[3]	Yes	Yes	Yes[4]	Yes	Yes
Chainer	Preferred Networks	2015	BSD	Yes	Linux, macOS	Python	Python	No	No	Yes	Yes	Yes	Yes	Yes
Deeplearning4j	Skymind engineering team; Deeplearning4j community; originally Adam Gibson	2014	Apache 2.0	Yes	Linux, macOS, Windows, Android (Cross-platform)	C++, Java	Java, Scala, Clojure, Python (Keras), Kotlin	Yes	No[6]	Yes[7][8]	Computational Graph	Yes[9]	Yes	Yes
Dlib	Davis King	2002	Boost Software License	Yes	Cross-Platform	C++	C++	Yes	No	Yes	Yes	Yes	No	Yes
Intel Data Analytics Acceleration Library	Intel	2015	Apache License 2.0	Yes	Linux, macOS, Windows on Intel CPU[11]	C++, Python, Java	C++, Python, Java[11]	Yes	No	No	Yes	No		Yes
Intel Math Kernel Library	Intel		Proprietary	No	Linux, macOS, Windows on Intel CPU[12]		CI[13]	Yes[14]	No	No	Yes	No	Yes[15]	Yes[15]
Keras	François Chollet	2015	MIT license	Yes	Linux, macOS, Windows	Python	Python, R	Only if using Theano as backend	Can use Theano, Tensorflow or PlaidML as backends	Yes	Yes	Yes[16]	Yes	Yes
MATLAB + Deep Learning Toolbox	MathWorks		Proprietary	No	Linux, macOS, Windows	C, C++, Java, MATLAB	MATLAB	No	No	Train with Parallel Computing Toolbox and generate CUDA code with GPU Coder[18]	No	Yes[20][21]	Yes[20]	Yes[20]
Microsoft Cognitive Toolkit (CNTK)	Microsoft Research	2016	MIT license[23]	Yes	Windows, Linux[24] (macOS via Docker on roadmap)	C++	Python (Keras), C++, Command line,[25] BrainScript[26] (.NET on roadmap[27])	Yes[28]	No	Yes	Yes	Yes[29]	Yes[30]	Yes[30]
Apache MXNet	Apache Software Foundation	2015	Apache 2.0	Yes	Linux, macOS, Windows,[34][35] AWS, Android,[36] iOS, JavaScript[37]	Small C++ core library	C++, Python, Julia, Matlab, JavaScript, Go, R, Scala, Perl, Clojure	Yes	On roadmap[38]	Yes	Yes[39]	Yes[40]	Yes	Yes
Neural Designer	Artelnics		Proprietary	No	Linux, macOS, Windows	C++	Graphical user interface	Yes	No	No	?	?	No	No
OpenNN	Artelnics	2003	GNU LGPL	Yes	Cross-platform	C++	C++	Yes	No	Yes	?	?	No	No
PlaidML	Vertex.AI,Intel	2017	AGPL	Yes	Linux, macOS, Windows	Python, C++, OpenCL	Python, C++	?	Some OpenCL ICDs are not recognized	No	Yes	Yes	Yes	Yes
PyTorch	Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan (Facebook)	2016	BSD	Yes	Linux, macOS, Windows	Python, C, C++, CUDA	Python, C++	Yes	Via separately maintained package[42][43][44]	Yes	Yes	Yes	Yes	Yes
Apache SINGA	Apache Software Foundation	2015	Apache 2.0	Yes	Linux, macOS, Windows	C++	Python, C++, Java	No	Supported in V1.0	Yes	?	Yes	Yes	Yes
TensorFlow	Google Brain	2015	Apache 2.0	Yes	Linux, macOS, Windows,[54] Android	C++, Python, CUDA	Python (Keras), C/C++, Java, Go, JavaScript, R,[55] Julia, Swift	No	On roadmap[56] but already with SYCL[57] support	Yes	Yes[48]	Yes[49]	Yes	Yes
Theano	Université de Montréal	2007	BSD	Yes	Cross-platform	Python	Python (Keras)	Yes	Under development[58]	Yes	Yes[51][52]	Through Lasagne's model zoo[53]	Yes	Yes
Torch	Ronan Collobert, Koray Kavukcuoglu, Clement Farabet	2002	BSD	Yes	Linux, macOS, Windows,[59] Android,[60] iOS	C, Lua	Lua, LuaJIT,[57] C, utility library for C++/OpenCL[58]	Yes	Third party implementations[59][60]	Yes[51][52]	Through Twitter's Autograd[53]	Yes[54]	Yes	Yes
Wolfram Mathematica	Wolfram Research	1988	Proprietary	No	Windows, macOS, Linux, Cloud computing	C++, Wolfram Language, CUDA	Wolfram Language	Yes	No	Yes	Yes	Yes[55]	Yes	Yes

https://en.wikipedia.org/wiki/Comparison_of_deep_learning_software

Demo, compare MPEG7 vs CNN

- DISA group
  Masaryk University in Brno, Czech Republic

  MUFIN project

  Visual search (5 MPEG7 descriptors) :
  http://disa.fi.muni.cz/imgsearch/similar

  Visual-Semantic search (CNN descriptors):
  http://disa.fi.muni.cz/demos/profiset-decaf

course:
Multimedia Retrieval

lecture 8:

Local image features

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

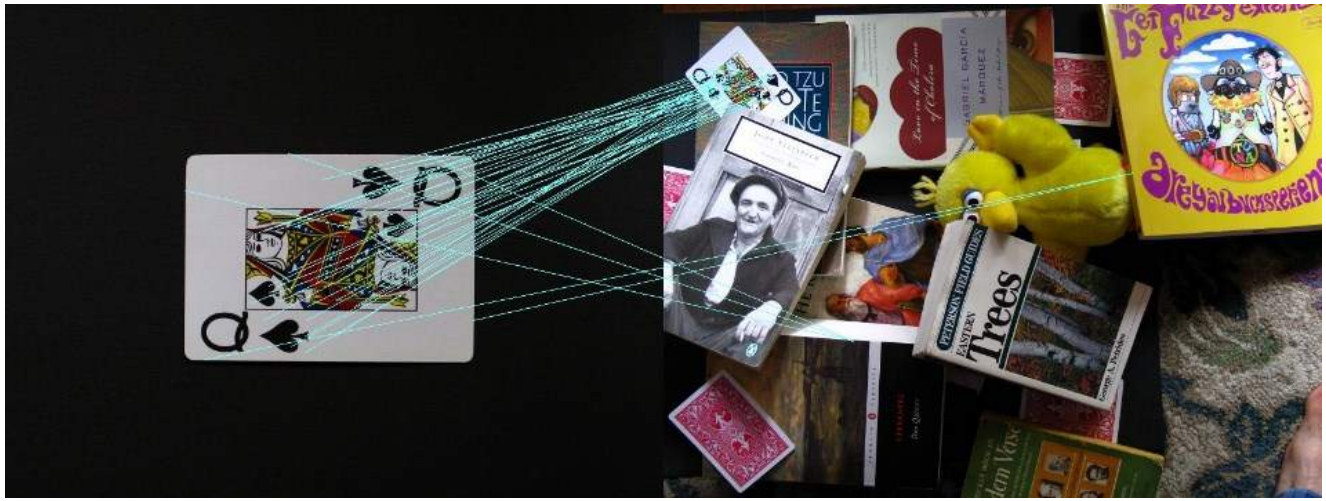
- descriptors of local image features – overview
- features based on interest points
  - edge/blob detection
  - interest points
  - SIFT
- image matching using BoF and SIFT

Local image features

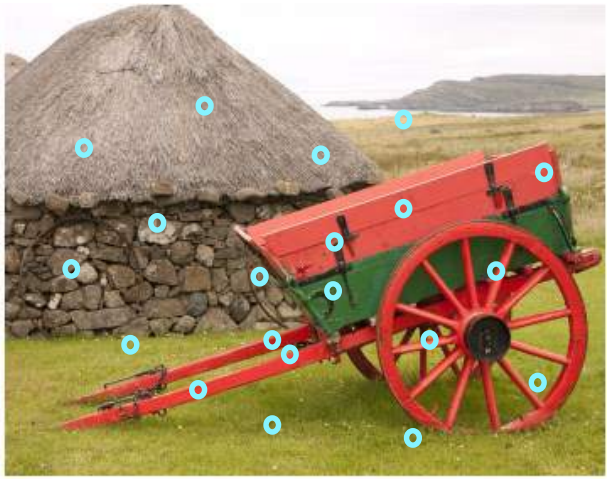
- MPEG-7 visual descriptors
  - rigid all-in-one solution (specific descriptors)
- descriptors of local image features – jigsaw puzzle
  - independent features localized in the image (at a pixel coordinate or as region/blob)
  - feature clustering
  - descriptor = set of feature clusters (feature signature)
  - applying a set similarity

Local image features

- asymmetric search problem – subimage (pattern) search
- rather identification than similarity
  - match invariant to translation, rotation, scale, color, …



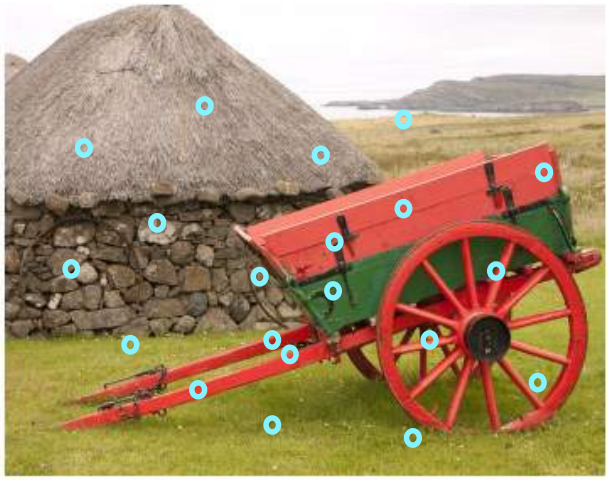
Local image features – feature localization



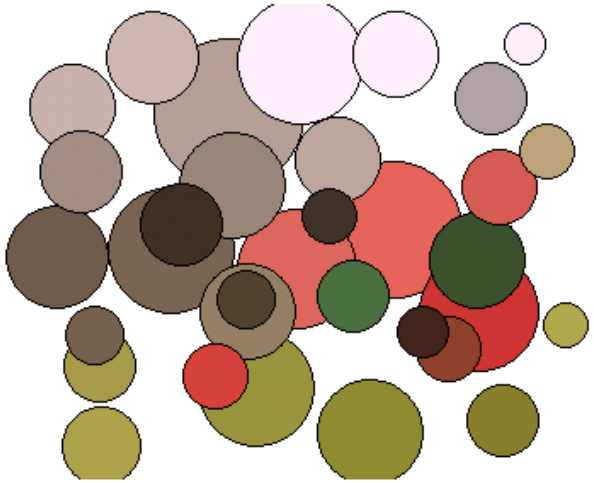
- a set of local features is localized (at a position in the image)
  - random sampling
  - interest point detection
  - segmentation
  - etc.
- variable number of local features per image
- each local feature described (by a vector)
- simple example that works [C. Beecks et al., Signature Quadratic Form Distances for Content-Based Similarity, ACM Multimedia 2009]
  - sampled pixels (uniform or normal distribution)
  - 7-dimensional feature vector $[x,y,L,a,b,\chi,\eta]$
    - pixel coordinates, color in Lab space, local contrast, local coarseness

Local image features – feature clustering

- clustering of features to identify homogeneous regions
  - robust representation
  - reduced number of features

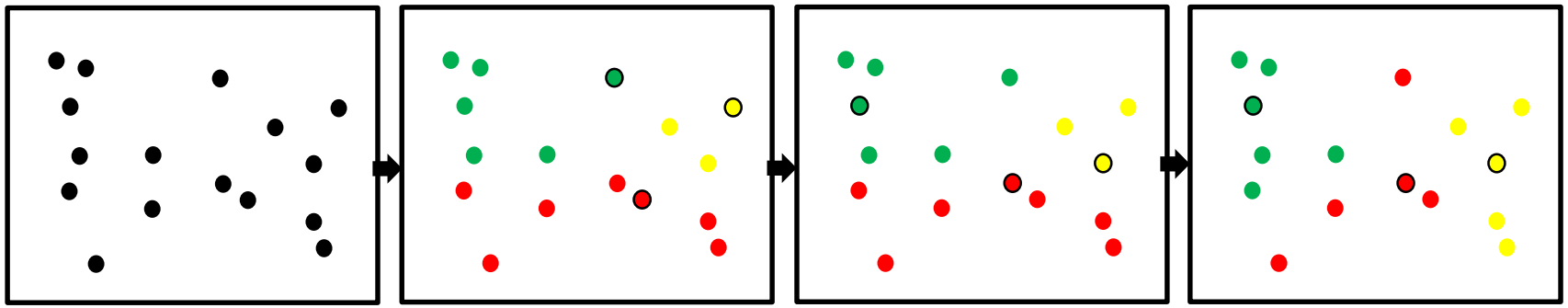


- various clustering algorithms
  - many algorithms proposed (over 100)
  - centroid-based, connectivity, etc.



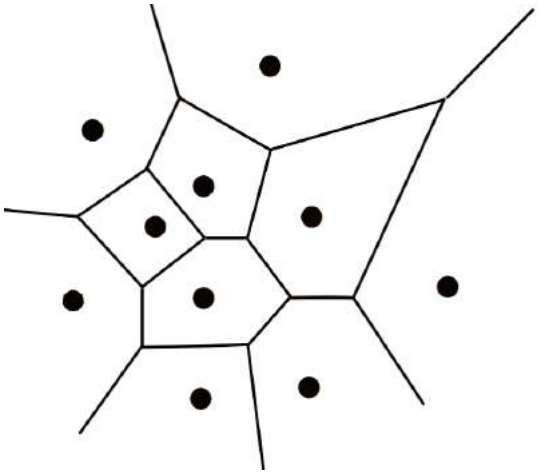
Local image features – feature clustering

- k-means clustering
  - centroid-based clustering technique, relatively fast
  - simple idea
    1. given a database of points (features), randomly select *k* "centroids"
    2. partition the whole database into *k* subsets based on the closest centroid
    3. recompute centroids of the clusters, forget the clusters, and goto step 2
    4. stop the iteration when a certain degree of convergence is reached
    5. return the centroids, cluster radii, and the partitioning

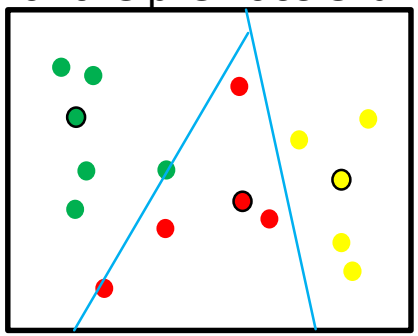


Local image features – feature clustering

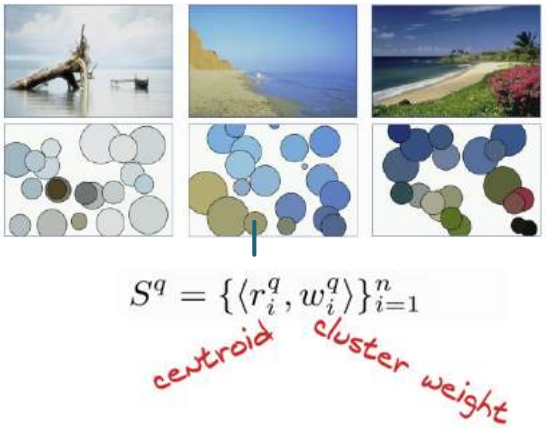
- Voronoi cells
  - k-means clustering gives us centroids and clustered data
  - partitions embedding the clusters are called Voronoi cells
    - border is defined as set of points in space of the same distance to the two closest centroids (usually using Euclidean distance)



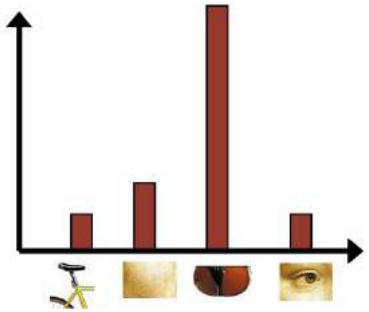
for the previous example:



Local image features – forming descriptor



- set of representatives
  - the descriptor is
    - directly the set of features (no clustering)
    - the set of centroids + cluster radii/weights
- bag of features model
  - the identified clusters are mapped to a vocabulary of representatives (e.g., visual words)
    - vocabulary = clustering of all features in all images (global clusters)
  - the descriptor is a histogram in the domain of visual words



- individual features as descriptors
  - each feature is treated separately
  - multiple features refer to single image

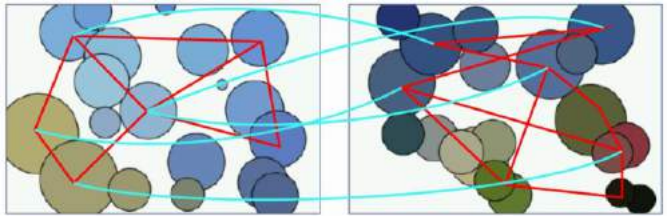
Local image features – measuring similarity

- set-based distances (for set descriptor)
  - global matching of features
    - quadratic form dist., Hausdorff dist., Earth mover's dist., etc.
  - local matching of features (nonmetric similarity)
    - asymmetric Hausdorff dist., local alignment techniques, etc.
- histogram distances (for histogram descriptor)
  - global matching
    - quadratic form distance, L_p distances, etc.
- vector distances (for feature in a role of descriptor)
  - additional aggregation needed (multiple descriptors for single image)

Local image features – set-based similarity

- global matching
  - signature quadratic form distance

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



  - Hausdorff- like distances, e.g.,
    - matching (*) of images d_i, d_j

$$C^d_{d_i,d_j} = \{(f_i, f_j) \mid f_i \in d_i, f_j \in d_j, \frac{\delta(\mathrm{f}_i, NN_1(\mathrm{f}_i, d_j))}{\delta(\mathrm{f}_i, NN_2(\mathrm{f}_i, d_j))} < c\}$$

    - the matching $C_{di,dj}$ (used for feature recognition) could be quantified into a normalized similarity function (**) as

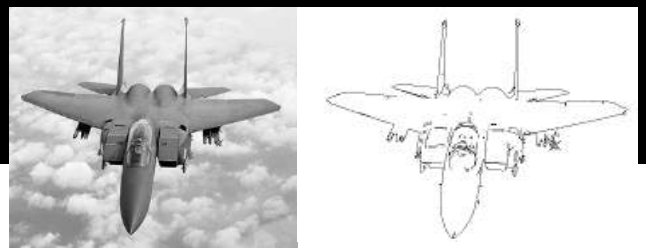
$$s(d_i, d_j) = \frac{|C_{d_i,d_j}|}{|d_i|}$$

* D. G. Lowe, Distinctive image features from scale-invariant keypoints. Int. J. of Computer Vision, 60(2), 2004
* Amato et al, Geometric consistency checks for kNN based image classification relying on local features, SISAP 2011

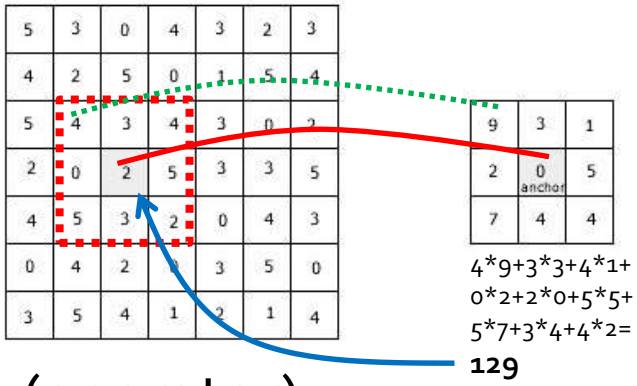
Local image features – points of interests

- features based on interest points/blobs
  - more sophisticated than just sampling pixels
  - could detect higher-level objects (edges/shapes)
  - designed primarily for specific image processing applications, i.e., not for general image retrieval
    - object recognition
    - 3D scene modeling (reconstruction)
    - object/person localization and tracking
    - image registration, alignment
    - robot navigation
    - etc.

Fundamentals – Edge/blob detection



- rapid change in brightness function – large derivative
- convolution of image
  - universal image processing operation (edge detection, blur, etc.)
  - kernel defines the image transformation



- edge detection by convolution mask (operator)
  - first-order derivative (Roberts, Sobel, Robinson, Kirch, ...)
    - e.g. Sobel
      - derivative approximations G_x, G_y (changes in the horizontal, resp. vertical direction)

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A}$$

$$\mathbf{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * \mathbf{A}$$

for each pixel:

gradient magnitude

$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}$$

gradient direction

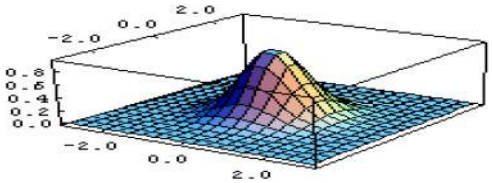
$$\Theta = \arctan\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right)$$

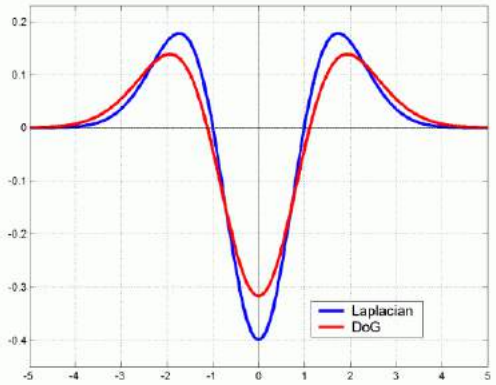
Fundamentals – Edge/blob detection

- second derivative of Gaussian
  - approximated by Difference of Gaussians (DoG)
    - $D(x, y, t) = L(x, y, k_i t) - L(x, y, k_j t)$
      - $k_i t$, $k_j t$ are scales
  - $L(\cdot,\cdot,t)$ = convolution of the image with Gaussian kernel *g*
    - blurred image at scale t (variance in Gaussian)
  - scale-space representation $L(x,y,t)$ at scale *t*
    - different-frequency edges are detected by $D(x, y, t)$
    - *t* is doubled (octave), k_i is selected to obtain fixed number of convolved images per octave

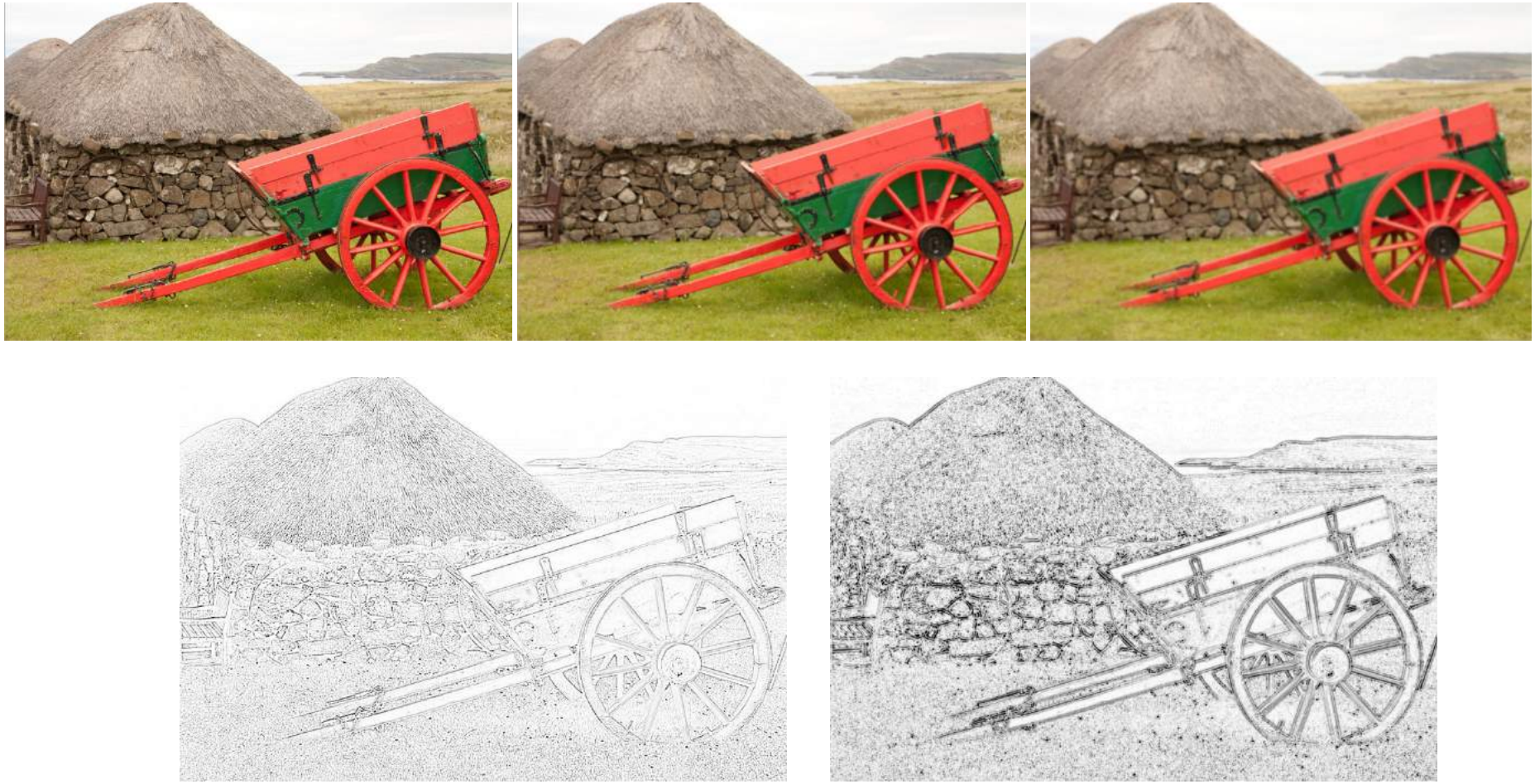
$$L(\cdot,\cdot;t) = g(\cdot,\cdot;t) * f(\cdot,\cdot)$$

$$g(x,y;t) = \frac{1}{2\pi t} e^{-(x^2+y^2)/2t}$$





Edge detection – differences of blurred images (Gaussians)

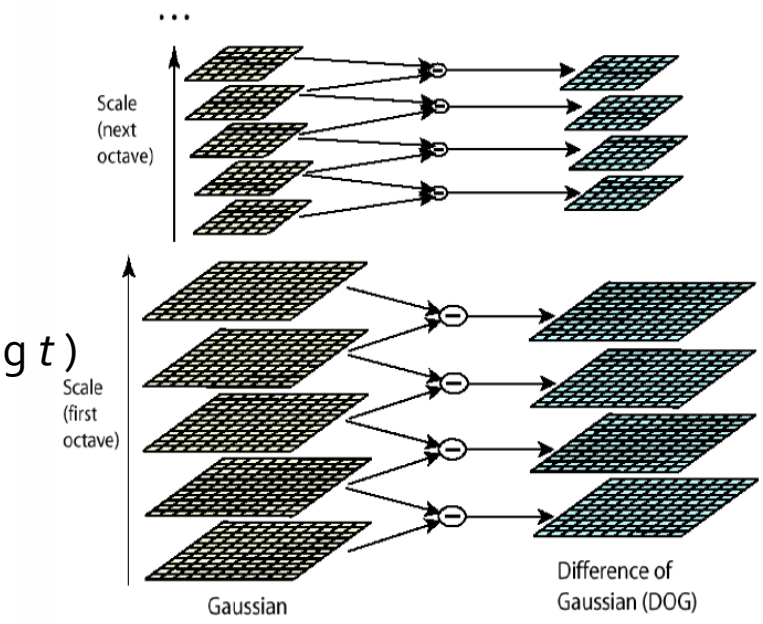


Fundamentals – Interest points

- we seek for local features in an image
- interest point = point in image that
  - is well-defined, well-positioned in image space
  - local information structure in the point neighborhood is rich
  - is stable (repeatable) with respect to perturbations in the images
    - changes in brightness, affine transformations, perspective deformations, etc.
- historically – corner detection (not sufficient)
  - junctions
- today – more general interest region (corner+blob) detection
  - points/regions in the image that are useful for image matching and visual object recognition
  - interest point is detected as a local 3D extreme having its absolute value greater than a threshold
  - can be used also for image segmentation, image retrieval

Fundamentals – Interest point detection

- DoG detector – space-scale extrema detection
- $D(x, y, t) = L(x, y, kt) - L(x, y, t)$
  - octave = group of blurred images (doubling t)
  - k images per each octave

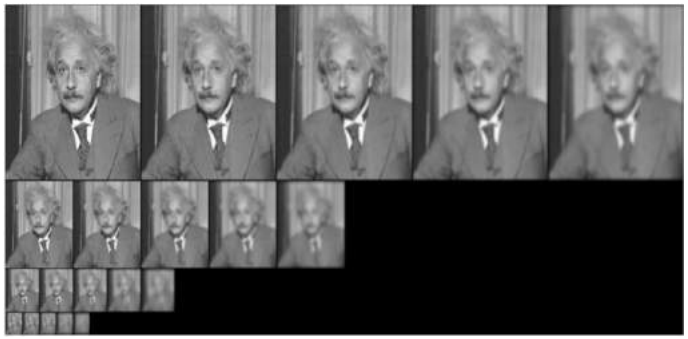


octave 1

octave 2

octave 3

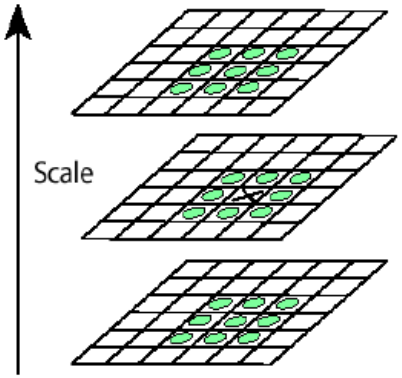
...



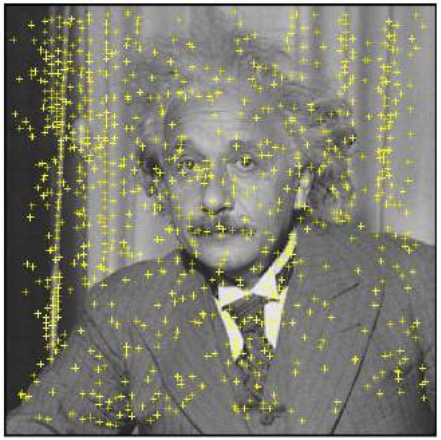
(images from Lowe, Distinctive Image Features from Scale Invariant Features, Int. Journal of Comp. Vision, 60(2), 2004 and Estrada et al., Local Features Tutorial, 2004)

Fundamentals – Interest point detection

- interest points identified as local maxima or minima of the DoG images across scales
  - consider a pixel in a DoG image as the center of cube across adjacent images
  - if the pixel is the local minimum/maximum, it is selected as a **candidate interest point**



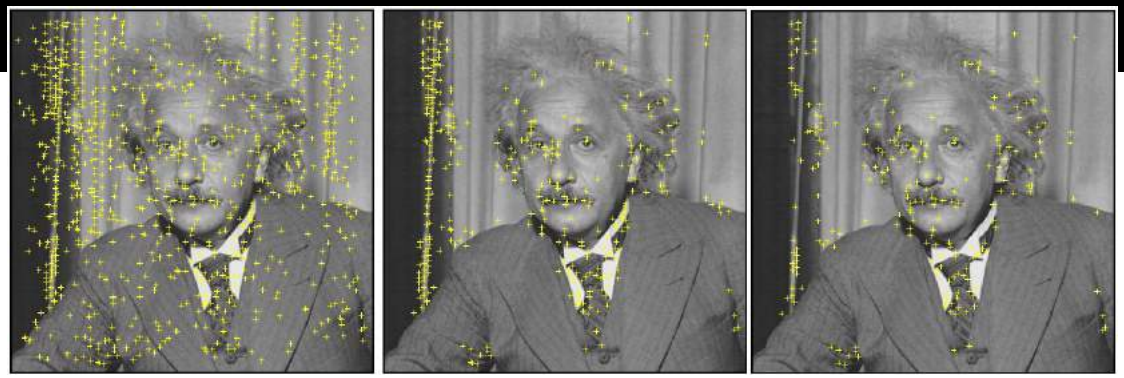
The pixel marked × is compared against its 26 neighbors in a 3 × 3 × 3 neighborhood that spans adjacent DoG images.



SIFT

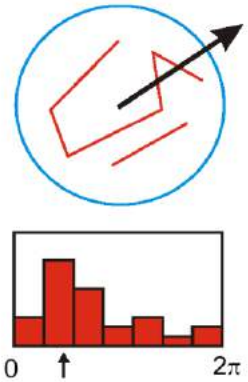
- SIFT = scale-invariant feature transform
- detector and descriptor of local features in image (algorithm, proposed in 1999)
  - scale-space extrema detection
  - keypoint (interest point) localization
  - orientation assignment
  - generation of keypoint descriptors
    - feature vector = 128-D vector belonging to a keypoint
    - desired property: repeatable and distinctive features
    - scale, translation, rotation, color invariant feature

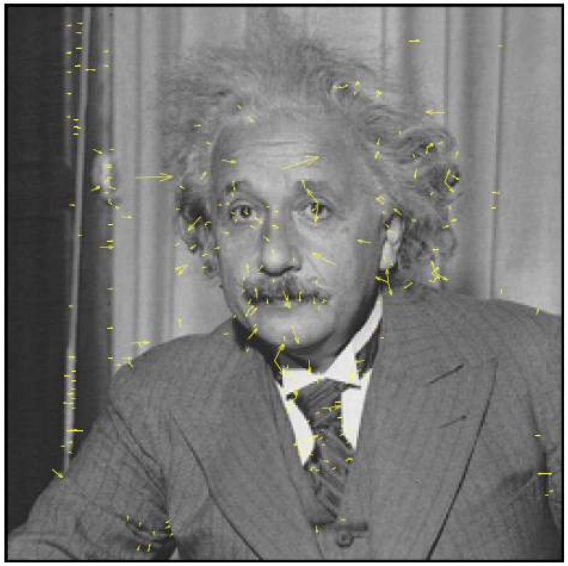
SIFT – keypoint detection



IP candidates | low-contrast IP removed | IPs on edges removed

- DoG detector – space-scale extrema detection, plus
  - removing "noise" keypoints (interest points)
    - low-contrast keypoints
    - keypoints along edges
  - orientation determined
    - gradient orientation histogram is computed in the neighborhood of the keypoint
    - peaks in the histogram correspond to dominant orientations

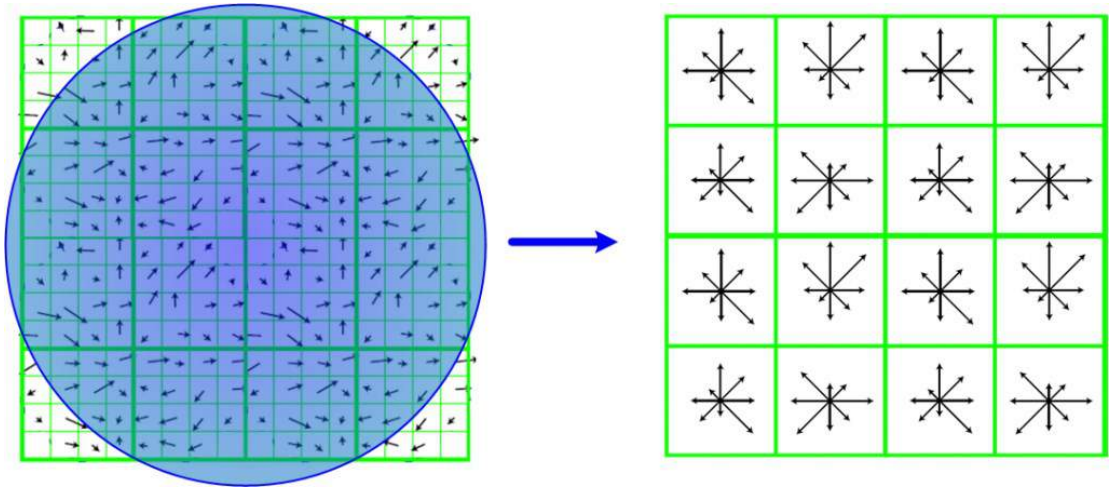




SIFT – feature descriptor

- keypoint descriptor
  - 4×4 histograms on 16×16 pixel patch around the keypoint (gauss-normalized gradients)
  - histograms contain 8 bins each
    - relative to the keypoint orientation (rotation invariance, max 50° in experiments)
  - SIFT feature vector = $4 \times 4 \times 8 = 128$ elements

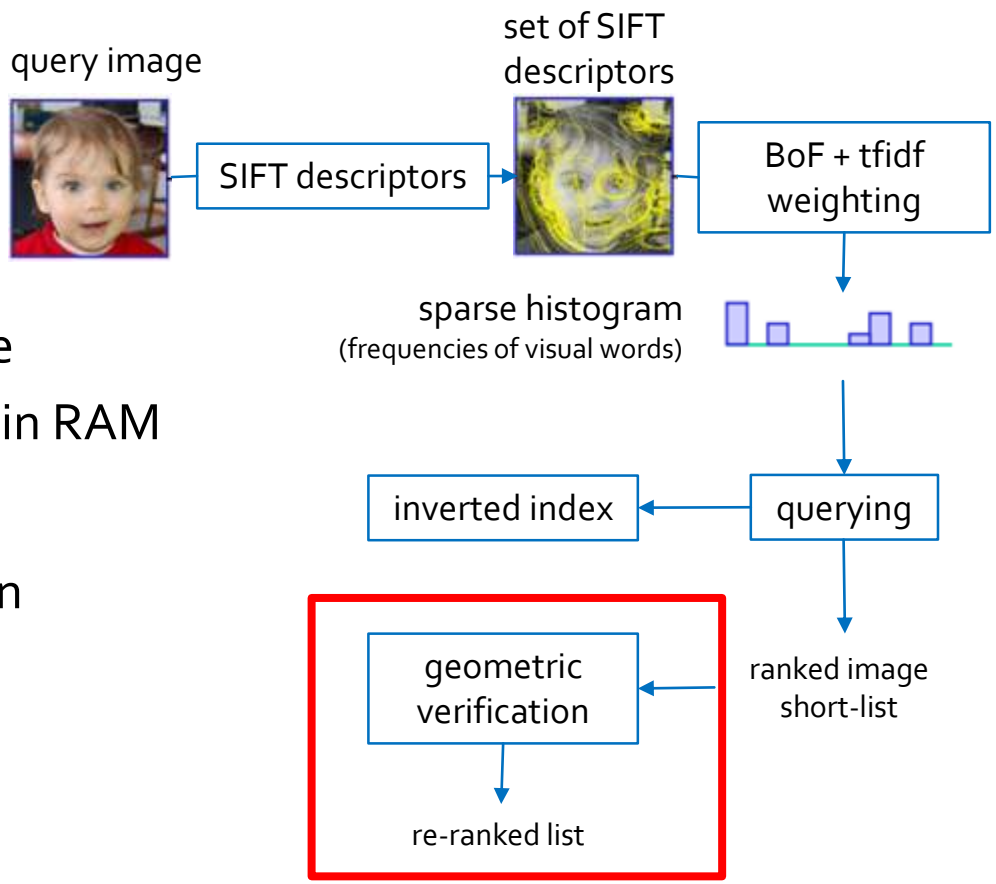
image gradients



keypoint descriptor

Image matching using SIFT features

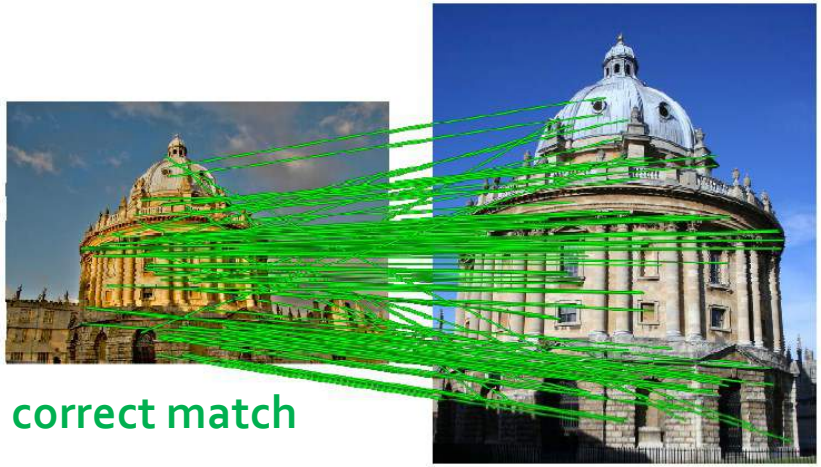
- bag of features
  - visual words
    - 1 word per SIFT
    - image ids in inverted file
    - 8GB for 1M images, fits in RAM
  - problem
    - matching approximation
  - solution
    - re-ranking using geometric verification



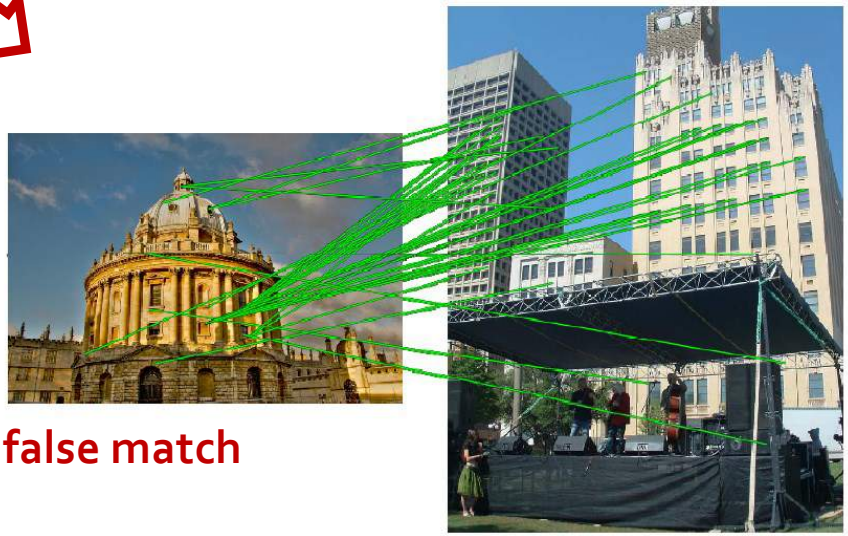
credit: images from Large-scale visual search by Cordelia Schmid, Visual Recognition and Machine Learning Summer School, Paris, 22-26 July 2013

Matching approximation

- visual words (local features) are matched 1:1
- many independent matches of local features
  ≠
  global match of images
- geometric verification needed, etc.



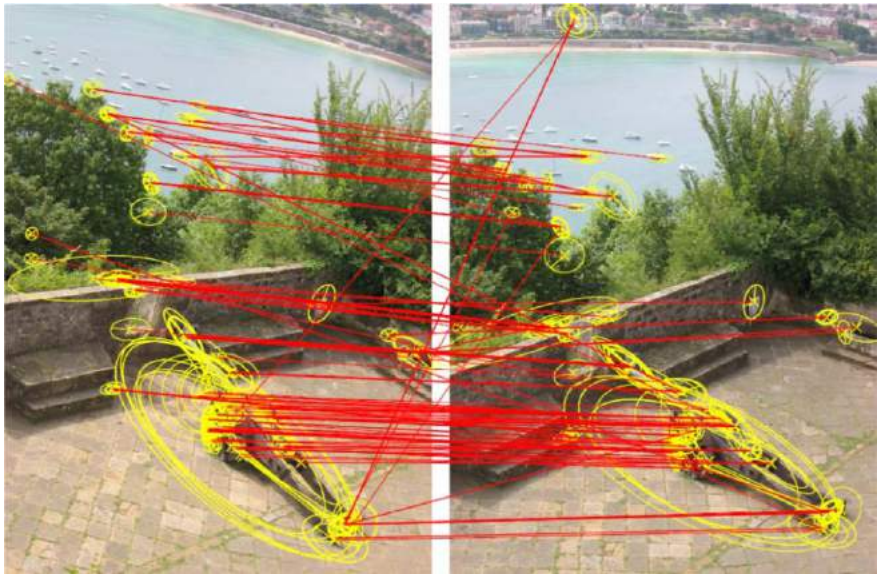
correct match



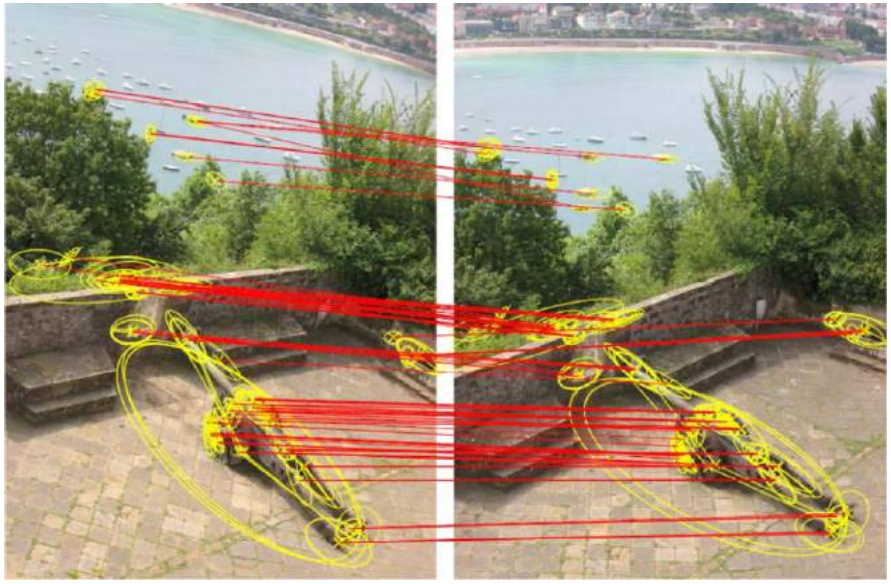
false match

Matching approximation

- increasing the BoF vocabulary reduces the incorrect matches, but introduces false dismissals
  - no good tradeoff



small vocabulary

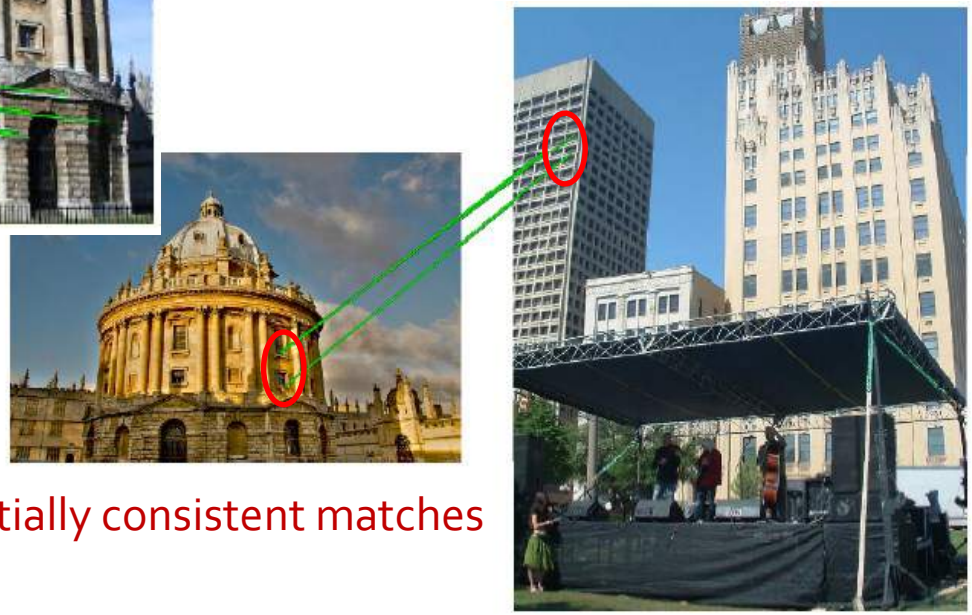


large vocabulary

Geometric verification

many spatially consistent matches

Remember, SIFT does not include
spatial positioning information,
just local properties on gradient changes



only few spatially consistent matches

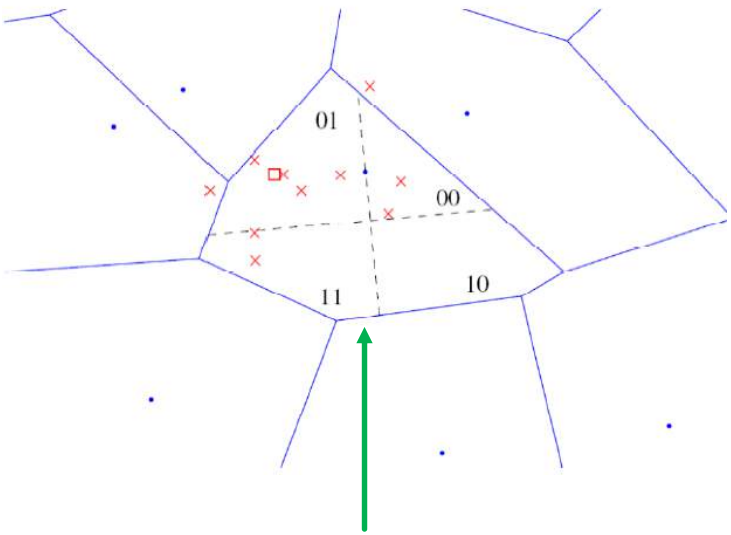
Geometric verification

- improving geometric consistency
  = enhance SIFT matching by including spatial position

- solutions
  - soft assignment (blurring the bins in histograms)
  - additional short codes (spatial information)
    - e.g. Hamming embedding

Geometric verification

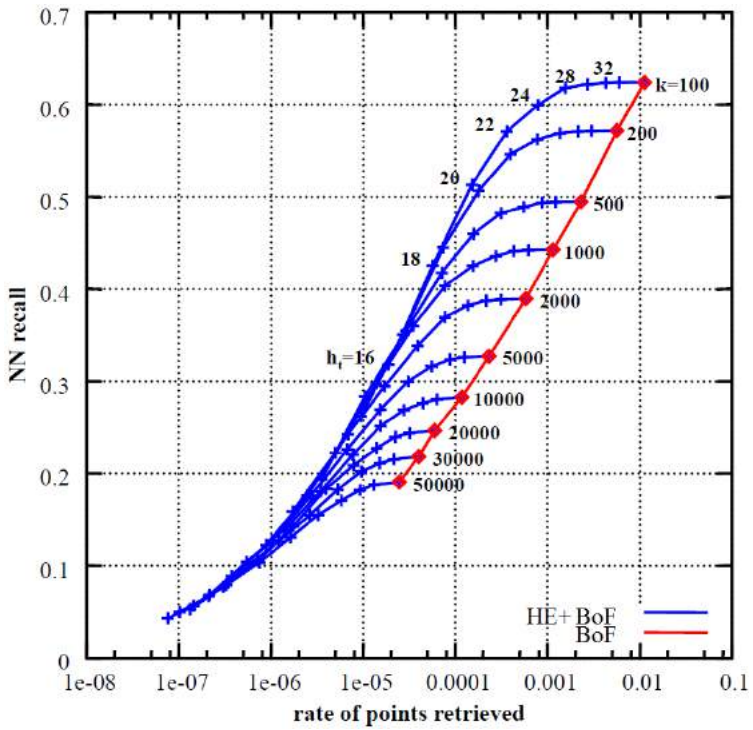
- Hamming embedding (HE)
  - descriptor x
    - q(x) = histogram in standard BoF
    - b(x) = short binary vector for localization in Voronoi cell
  - descriptors x and y match iff
    - $q(x) = q(y)$ and $h(b(x), b(y)) \leq h_t$, where $h(a,b)$ is Hamming distance and h_t is a distance threshold
  - efficient

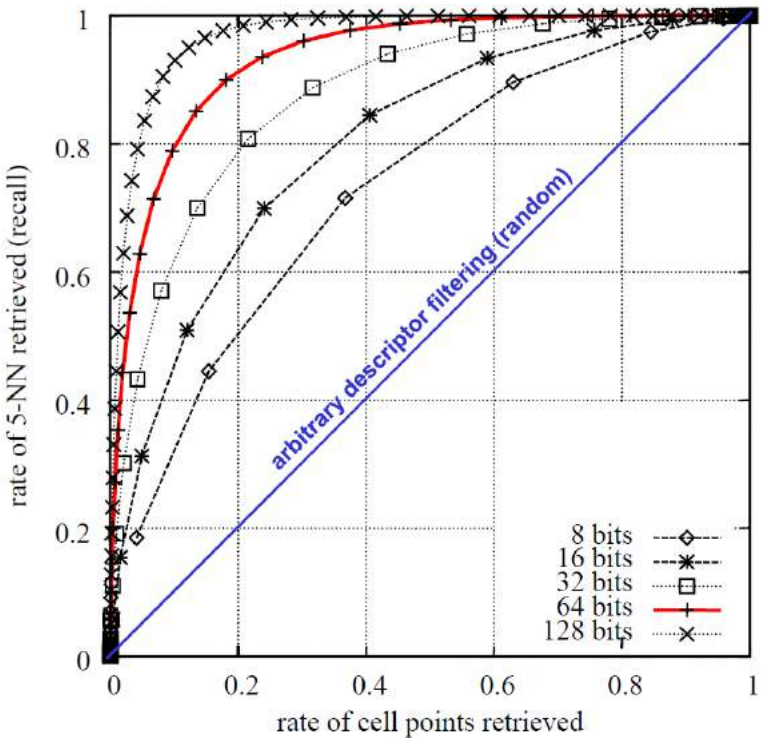


Voronoi cell of particular visual word in the BoF model

Geometric verification

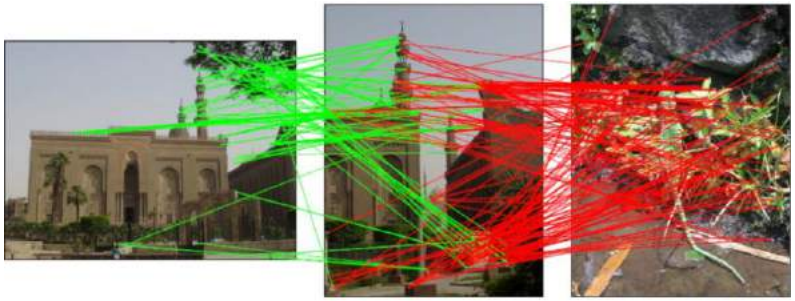
- impact of vocabulary size on NN recall = BoF + HE requires 10% of the short-list than BoF for the same recall
- impact of Hamming code length (memory requirement)



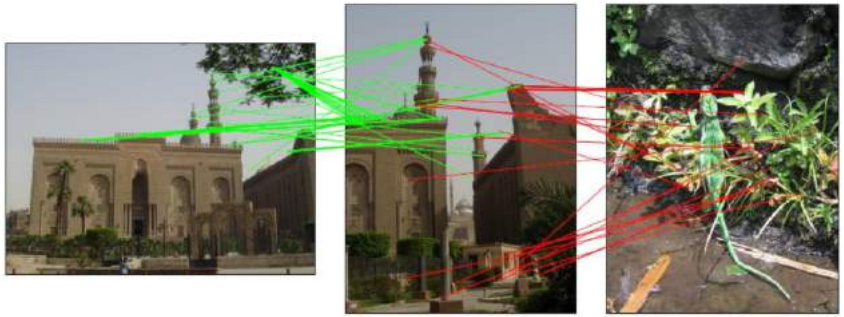


Geometric verification

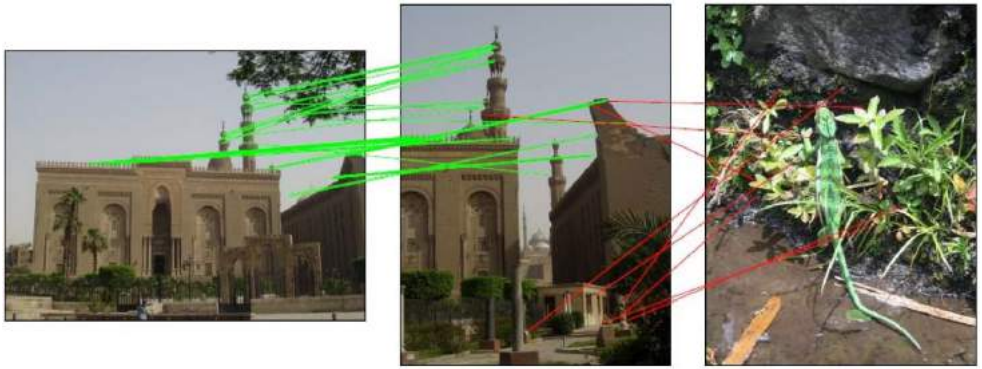
BoF (20K vocabulary)



BoF (200K vocabulary)

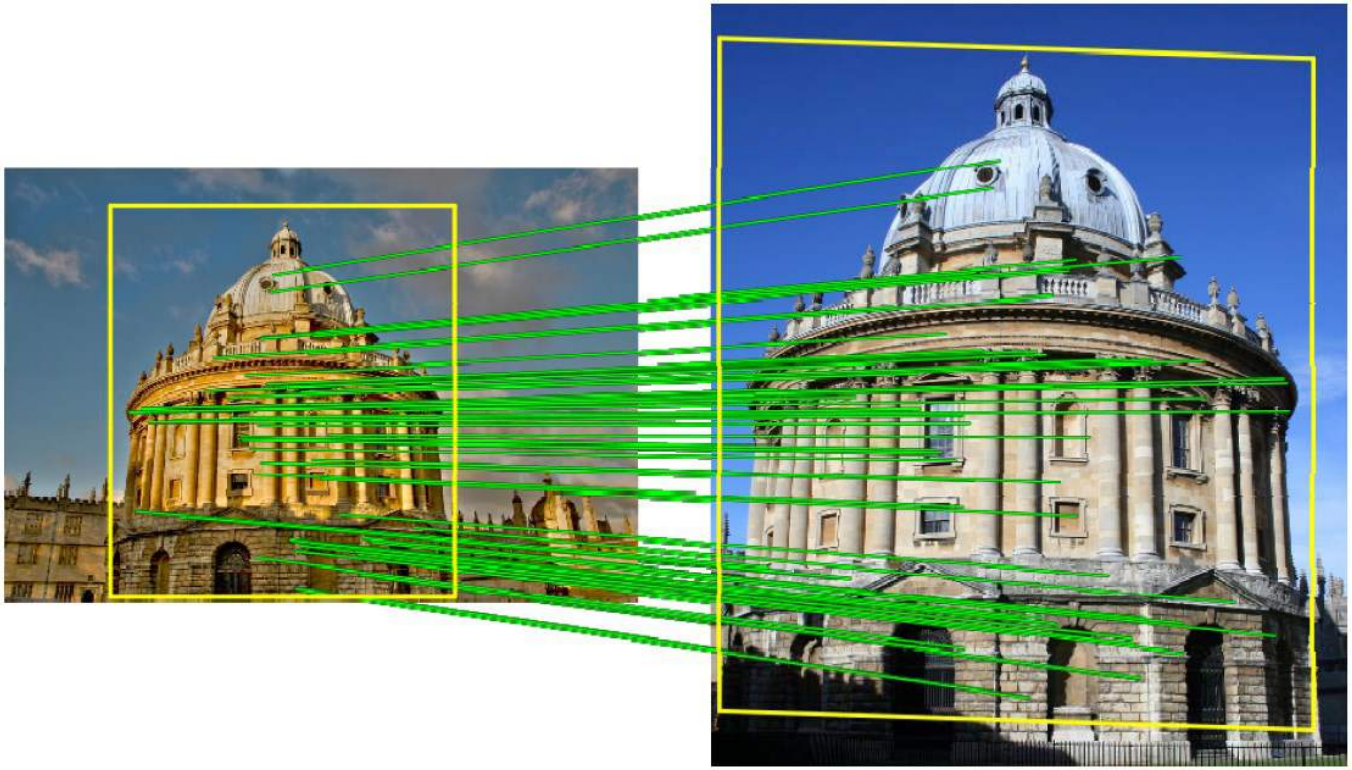


BoF + HE (20K vocabulary)



Geometric verification

- by-product: localization of objects



SIFT implementations

- algorithm patented in the US (the owner is the University of British Columbia)
- SIFT Keypoint detector – SIFT author's demo software
  http://www.cs.ubc.ca/~lowe/keypoints/
- C# library
  http://user.cs.tu-berlin.de/~nowozin/libsift/
- Matlab implementation
  http://www.vlfeat.org/~vedaldi/code/sift.html

SIFT competitors

- RIFT
  - rotation-invariant generalization of SIFT (beyond the 50°)
- G-RIF
  - Generalized Robust Invariant Feature
- PCA-SIFT, GLOH
  - gradient location-orientation histogram (GLOH)
  - increase its robustness and distinctiveness
  - descriptor = 128-eigenvalues obtained from 272 bin histogram reduced by PCA (given image patches collected from various images)
- SURF
  - Speeded Up Robust Features
  - high-performance scale and rotation-invariant interest point detector/descriptor

course:
Multimedia Retrieval

lecture 9:
Retrieval of 2D and 3D models

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

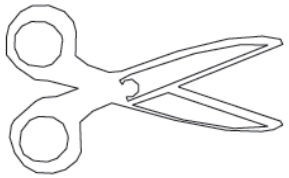
Lecture overview

- 2D models retrieval
  - MPEG-7 shape descriptors
  - time series approach
  - protein structure similarity (2.5D)
- 3D models retrieval
  - feature extraction
  - query-by-example
  - query-by-sketch

2D models retrieval

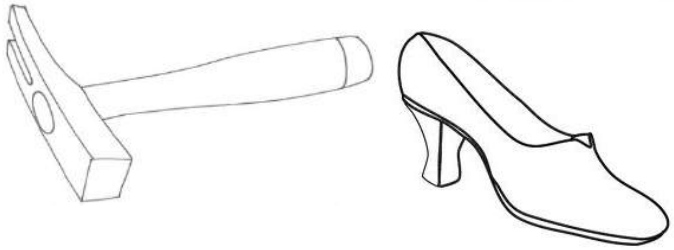
2D models/shapes

- 2D shape – narrow meaning as contour/silhouette of an object
  - open/closed directed polygons (geometry, not pixels)
  - other (cloud of points,…)



- database of 2D models (native)
  - outputs of CAD/CAM/vector editor
  - 2D projections of 3D models

- shapes obtained by vectorization of objects in raster images
  - assuming simple isolated images
    - e.g., product images



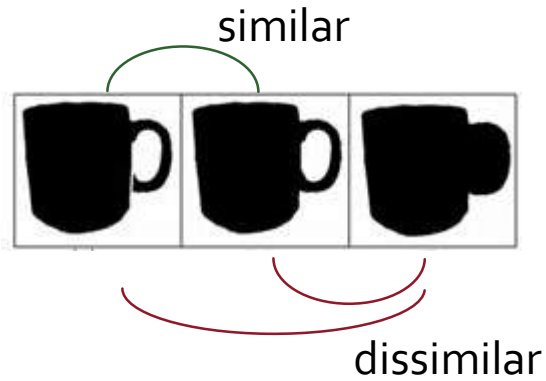
Shape features in images

- higher-level feature, intuitive concept for human
  - planar geometry objects
- 2D projections (onto an image plane) of the real-world 3D objects
- regions vs. contours
- descriptor should be invariant to scaling, rotation, translation, shape distortion (projection of 3D object under different angle)
- brand names or company logos in an image

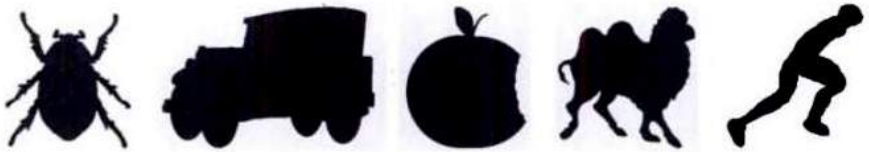
MPEG7 descriptor – Region Shape

- captures distribution of all pixels in a certain region
- based on both boundary and internal pixels
  - suitable for complex objects (disconnected regions) as well as simple objects (with or without holes)

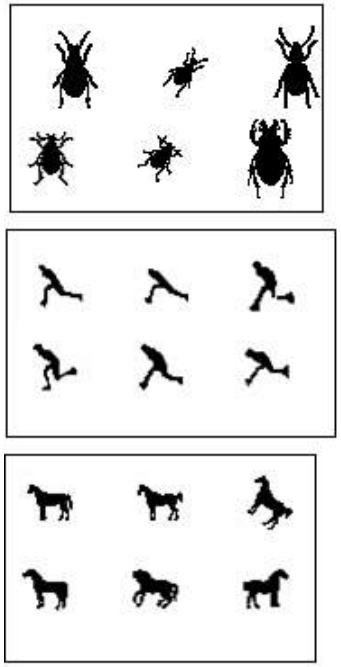
- set of coefficients produced by angular radial transformation (ART) – complex 2D transform
  - can cope with errors in segmentation
  - robust to segmentation noise (minor deformations along the boundary)
  - small size descriptor (17.5 bytes), fast extraction time and matching



MPEG7 descriptor – Contour Shape



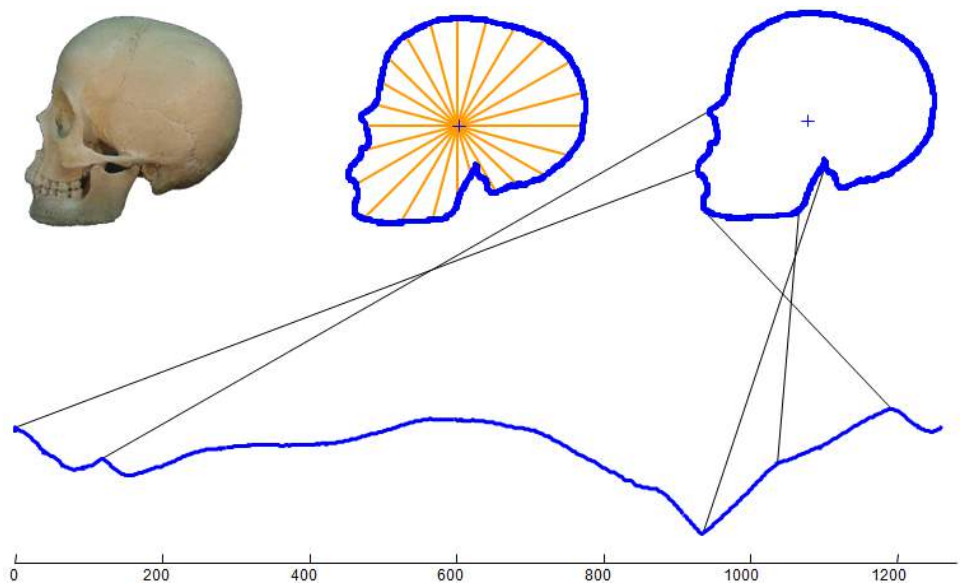
- captures closed contour of a 2D object in image
- based on the Curvature Scale Space (CSS) representation of the contour
  - reflects properties of the perception of human visual systems
  - robust to partial occlusion of shapes
  - robust to non-rigid motion (like changes of camera perspective)
  - compact representation (14 bytes on average)
  - captures very well characteristic features of the shape, enabling similarity-based retrieval
  - offers generalization



2D shapes – closed polygons

(images © Eamonn Keogh, eamonn@cs.ucr.edu)

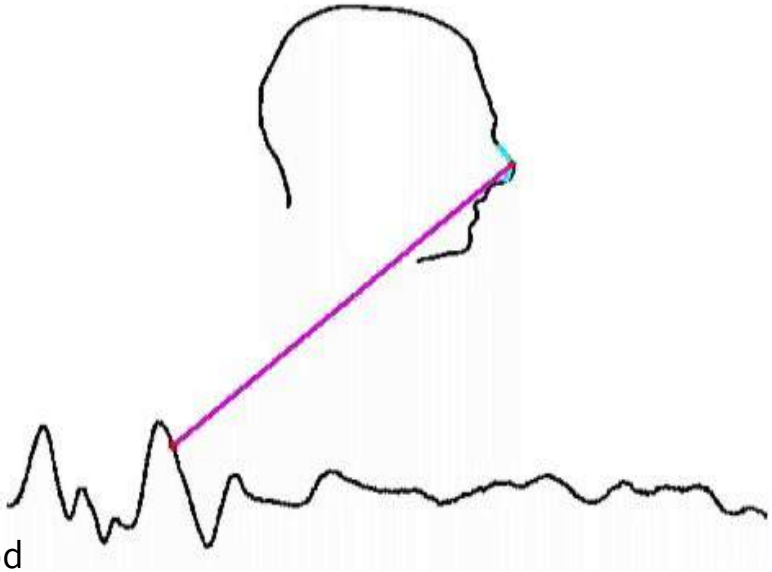
- approach for closed polygons
  - find the centroid of shape
  - walk with „touching hour hand" along the polygon by a fixed angular step, recording the length of the hand
- advantages
  - scale-invariant (if normed)
  - sampling-invariant
    - does not matter how fine-grained the polygon is
- problems
  - sensitivity to rotation (where to start with walking?)



2D shapes – open polygons

(images © Eamonn Keogh, eamonn@cs.ucr.edu)

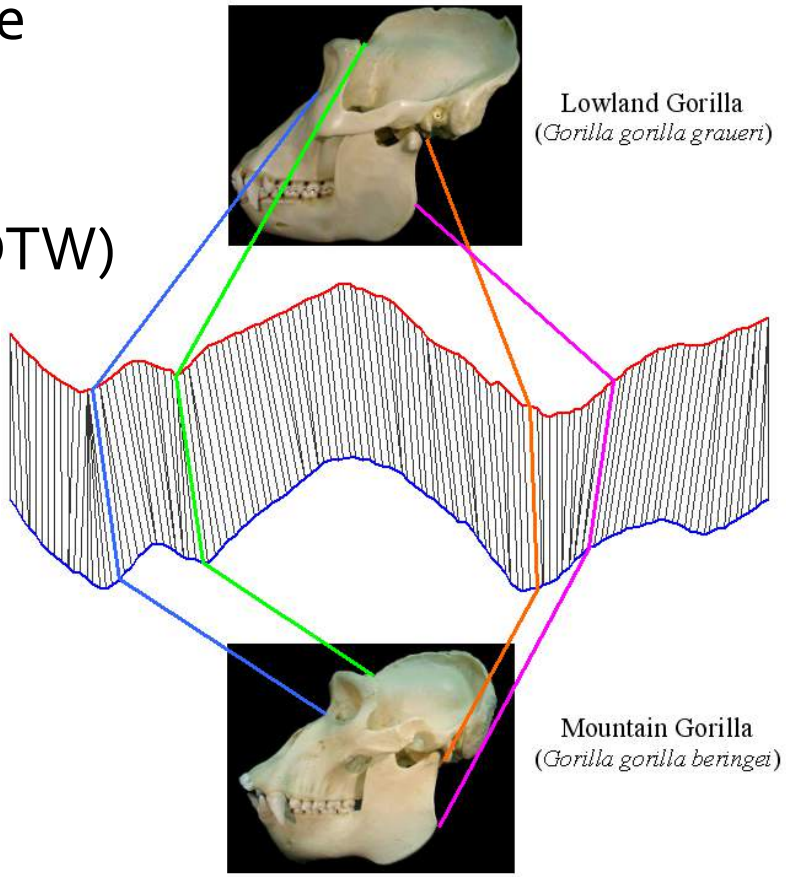
- approach for open polygons
  - centroid not appropriate
    - different centroid position for differently extended (cropped) polygon
  - instead, angles between adjacent edges recorded into the time series
- advantages
  - rotational invariance
    - the polygon has a start and an end
  - even more local information recorded w.r.t. the centroid (which itself is global)
- problems
  - scale- and sampling-dependent (dividing narrow line produces 180° angle)
    - could be improved by taking larger neighborhood
  - not very stable representation (sensitive to noise)



Shape matching by DTW

(images © Eamonn Keogh, eamonn@cs.ucr.edu)

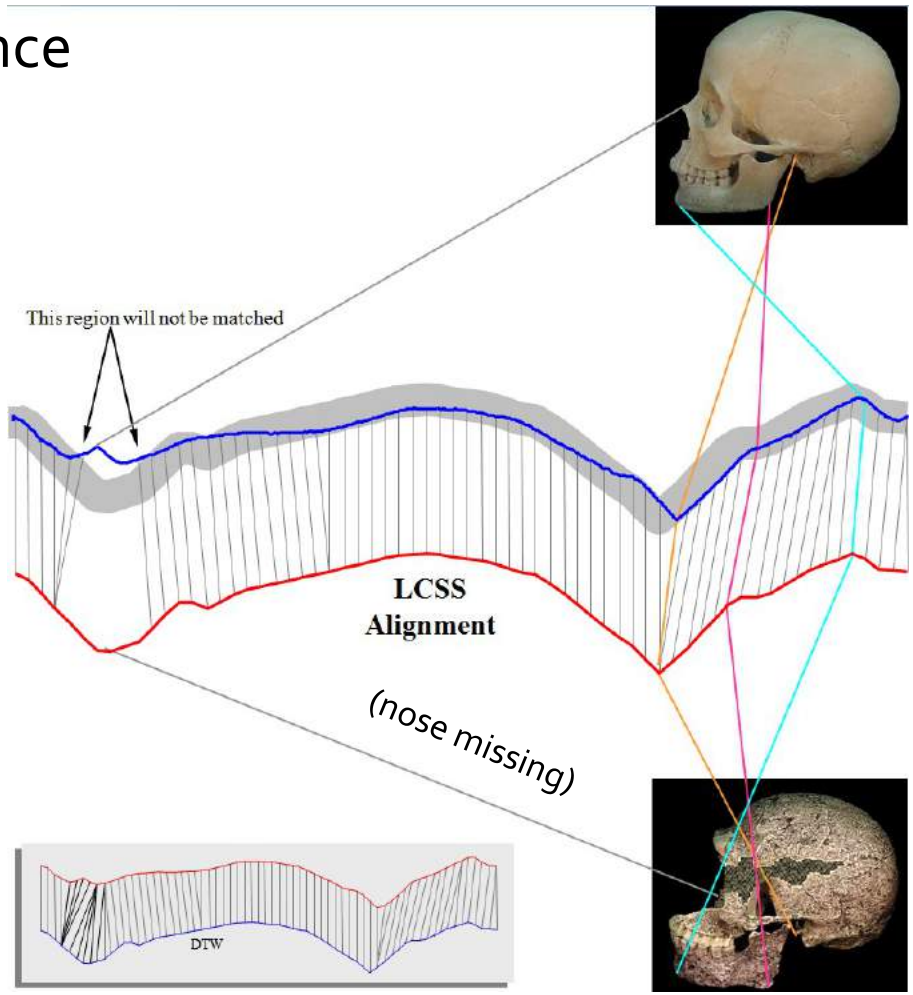
- Euclidean distance not appropriate
  - not sensitive to local deformations
  - requires series of the same length
- dynamic time warping distance (DTW)
  - elastic matching
    of time series (shapes)



Shape matching by LCSS

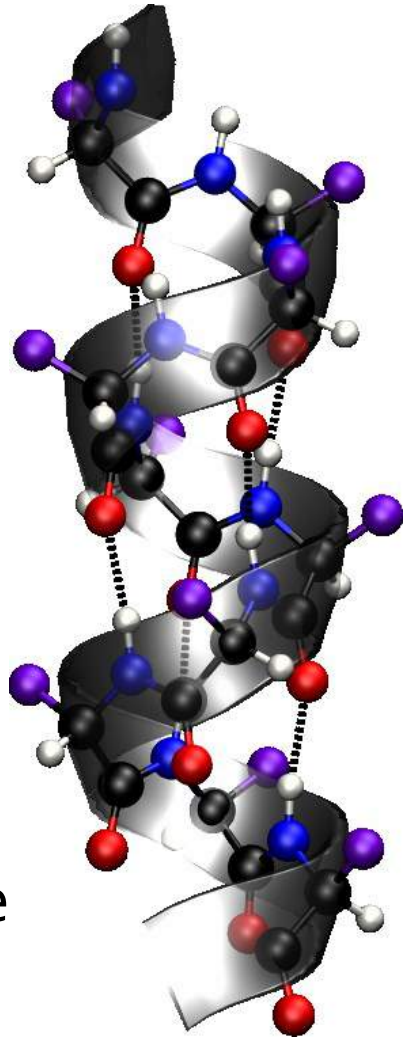
(images © Eamonn Keogh, eamonn@cs.ucr.edu)

- longest common subsequence
  - local alignment
  - allows to ignore gaps
    - missing or occluded shape parts

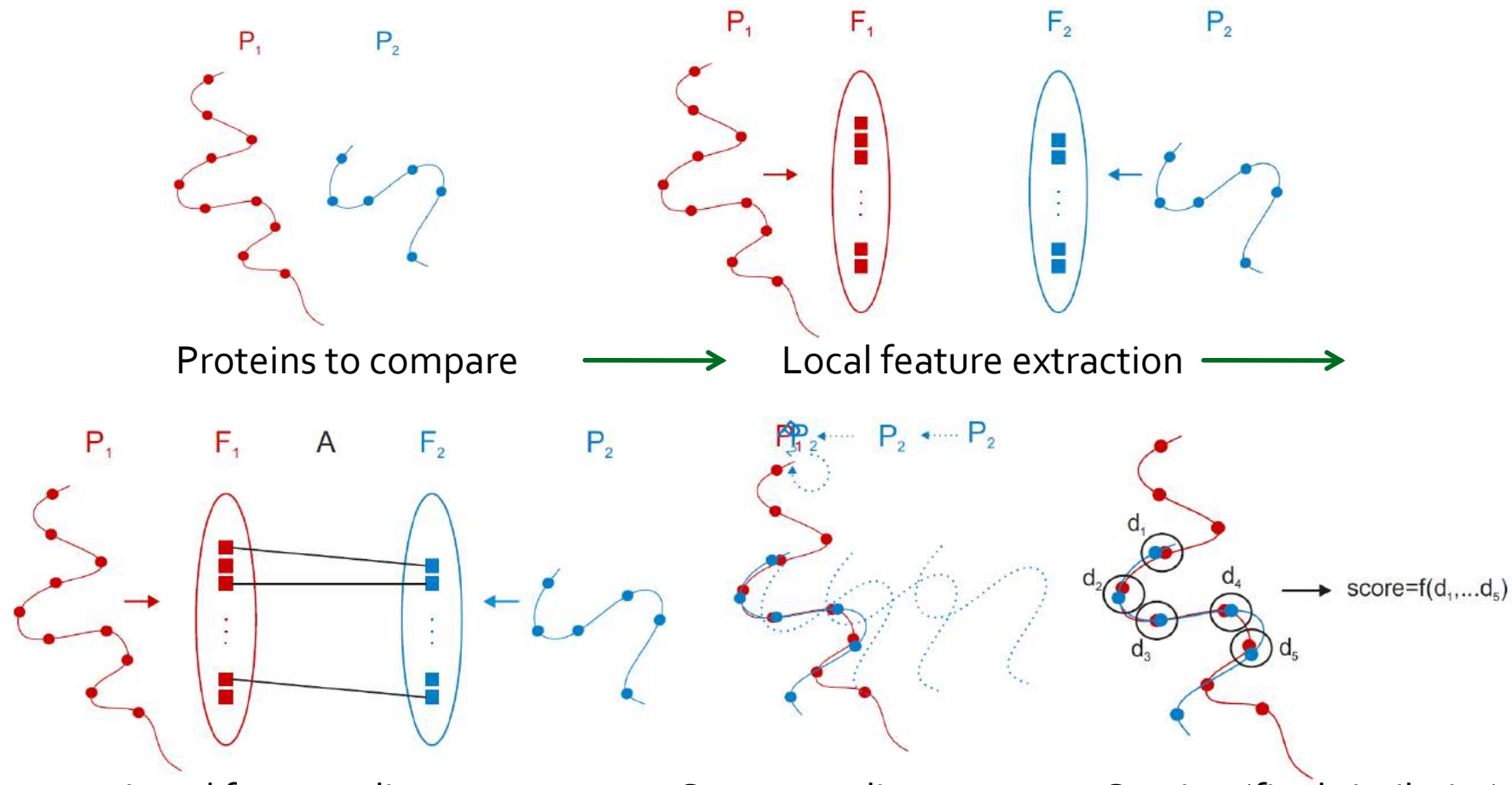


Protein structure similarity

- Protein structure
  - Structure is more correlated to biological function than sequence (but harder to obtain)
- Similarity – two-step optimization process
  1) Alignment of structures based on local properties/features
     - Shape properties (torsion angles between AAs, density of AAs, curvature, surface area)
     - Physico-chemical properties (hydrophobicity, AA volume)
  2) Aggregation measure on top of the alignment
     - RMSD, TM-score
- Existing top algorithms for function assessme
  - DDPIn+iTM, PPM, Vorometric, TM-align, CE



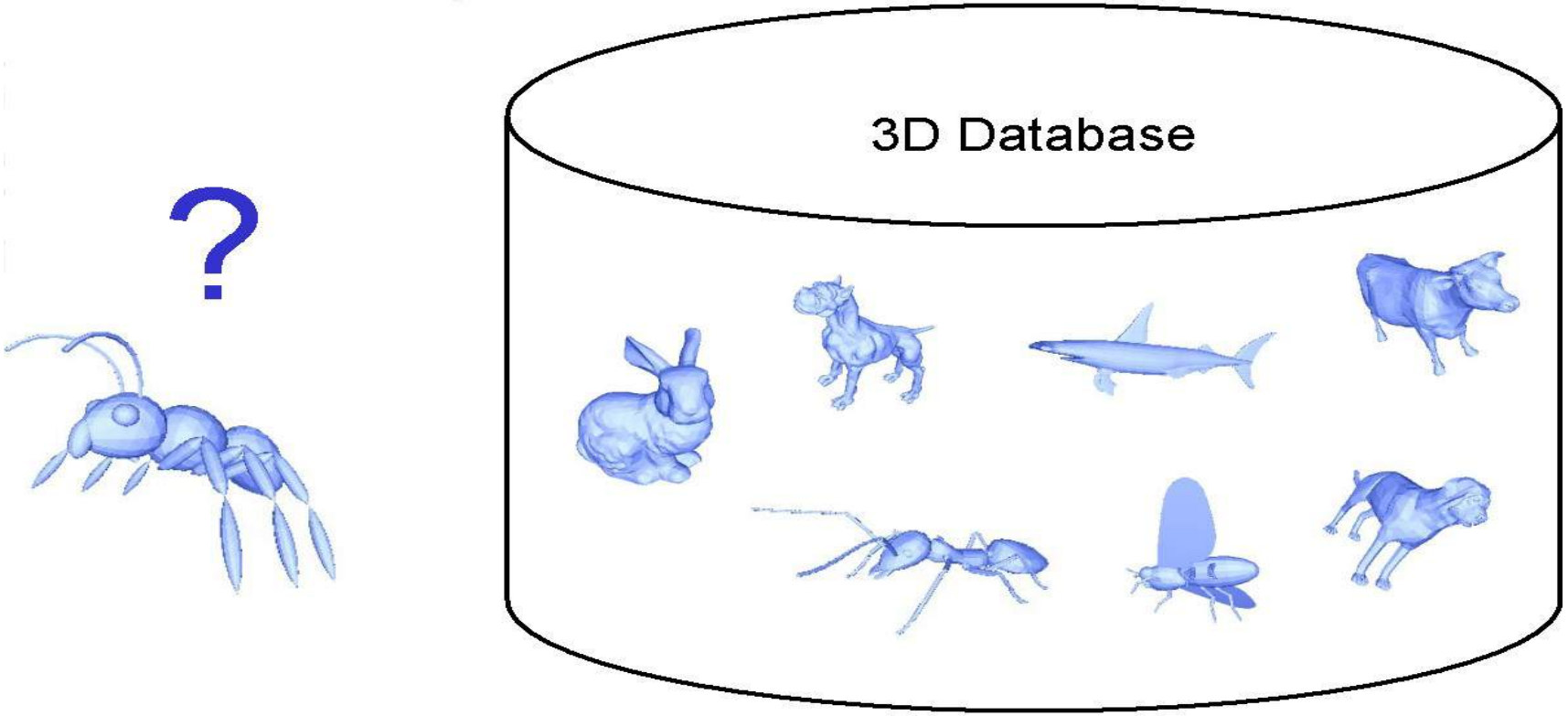
Protein structure similarity



3D model retrieval

selected slides from keynote at CBMI 2015 with permission of prof. Benjamin Bustos, entire keynote available here

Motivation



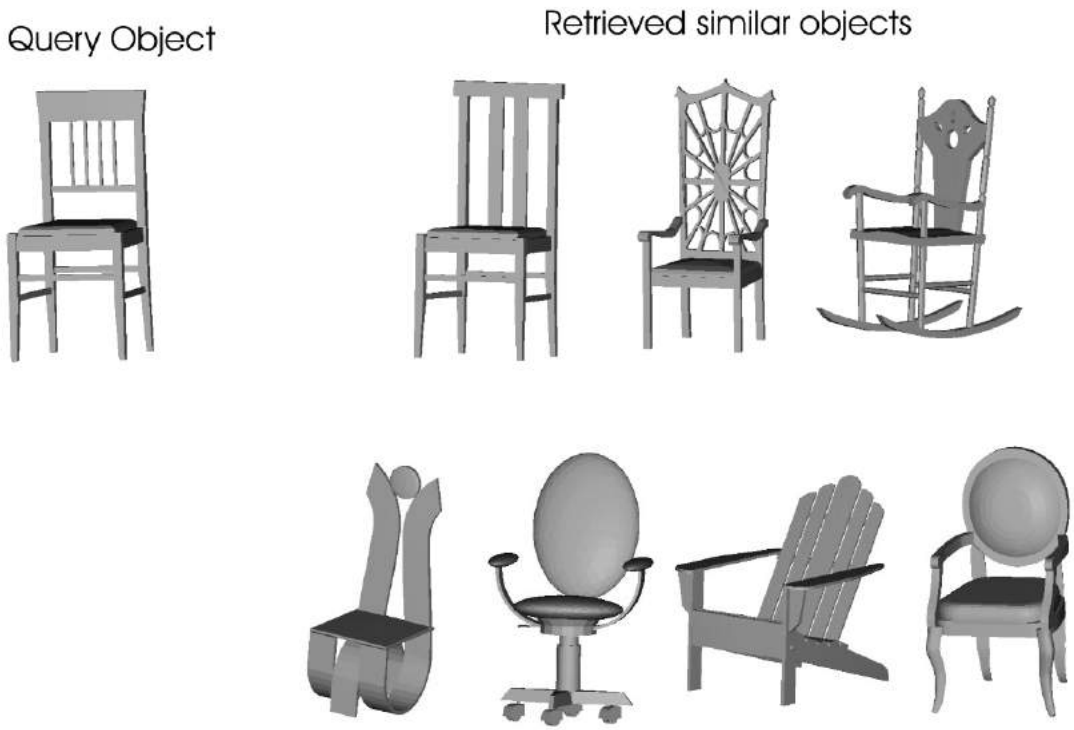
Lots of practical applications for this problem!

Motivation

- Main problems for solving 3D object retrieval
  - Representation of the 3D model
    - Objects at different scales / "resolution" / pose
    - Triangle meshes / Point clouds
    - Quality (holes, badly oriented triangles, etc.)
  - How to measure similarity?
    - Geometric or semantic?
  - What about partial similarity?
    - Whole-to-part / Part-to-whole

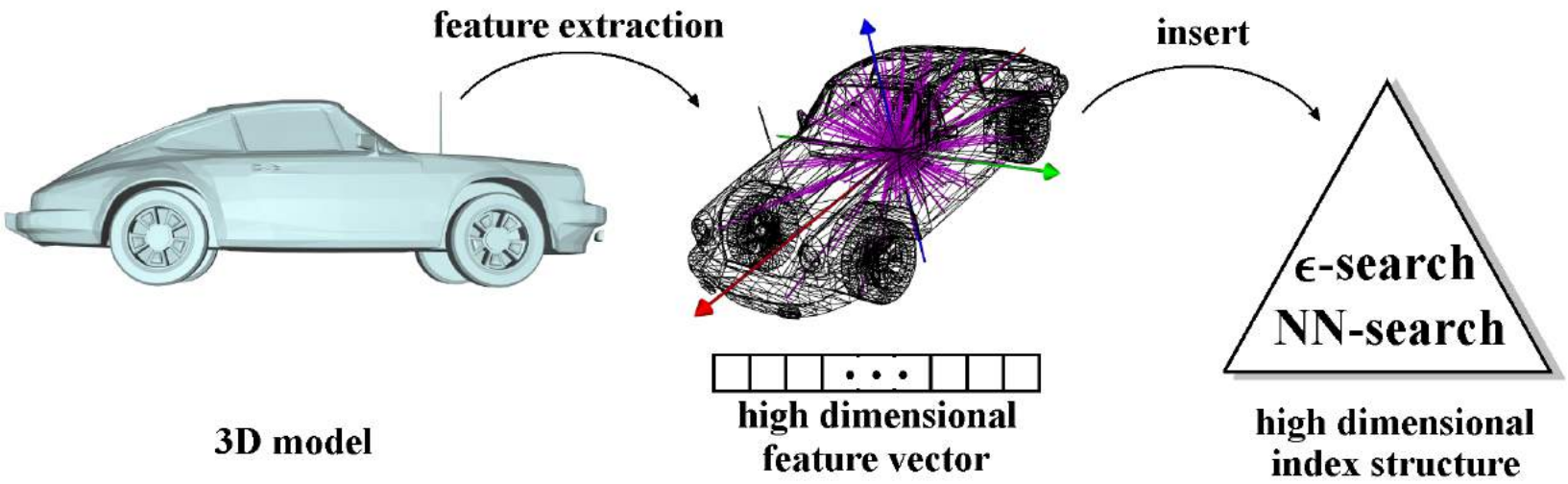
3D model similarity search

- **Geometric** vs semantic similarity



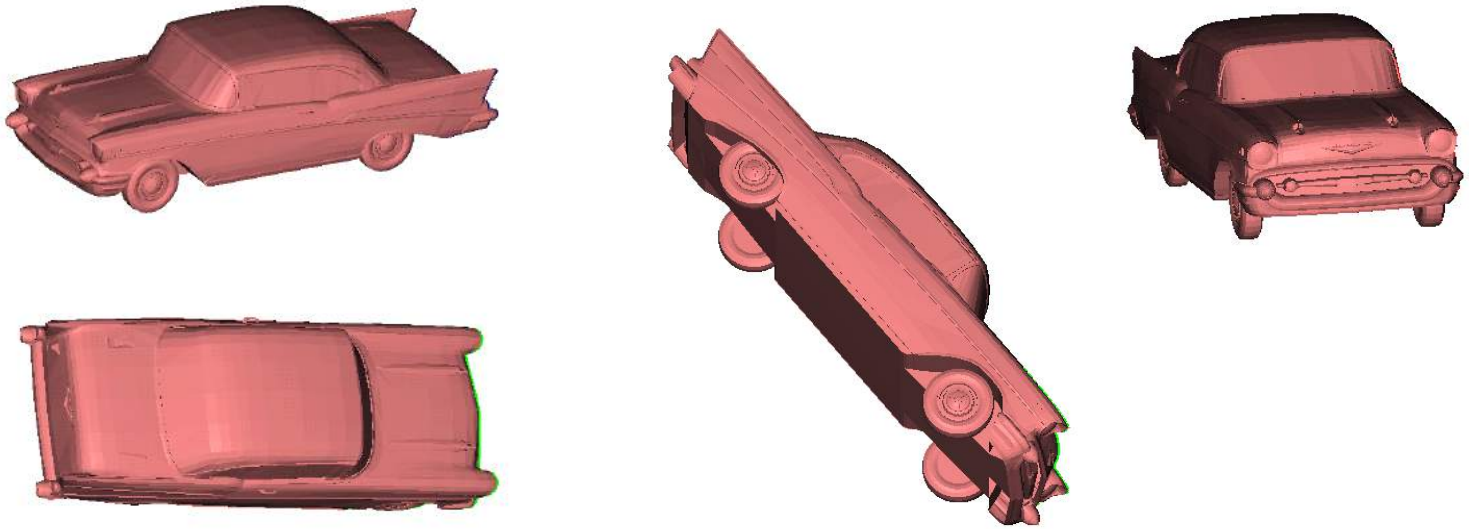
3D model similarity search

- General approach: feature vectors



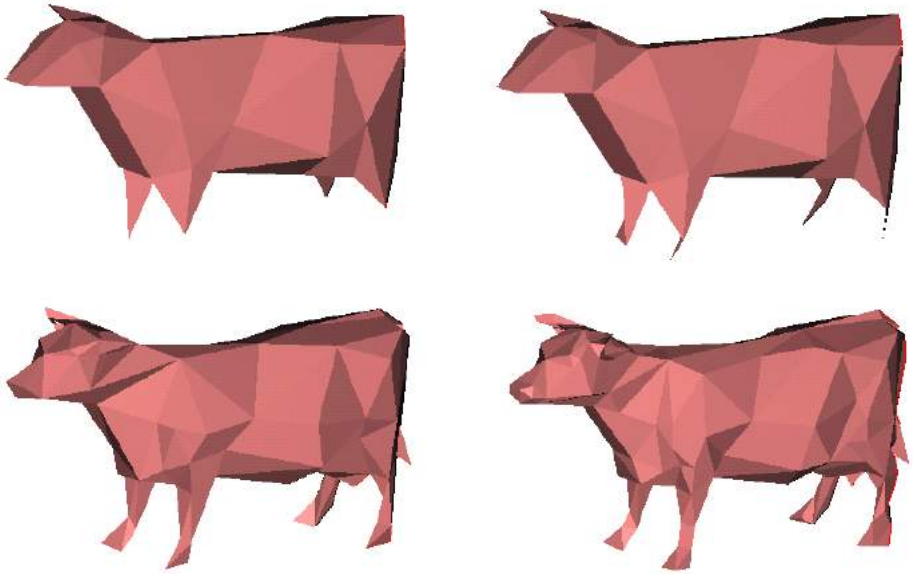
3D model similarity search

- Requirements for 3D descriptors
  - Invariance with respect to rotations, translations, scaling and (in some cases) reflections



3D model similarity search

- Requirements for 3D descriptors
  - Robustness with respect to level-of-detail (different *tessellations*)



3D model similarity search

- Requirements for 3D descriptors
  - Robustness with respect to *outliers*
  - Efficient feature extraction
  - Multiresolution representation
  - Discrimination of different geometric forms

Normalization of 3D objects

- Goal
  - Find a canonical coordinate system
- Result
  - Invariance against translation, rotation, scaling and reflections
- Tools*
  - Center of mass (translation)
  - Continuous PCA (rotation)
  - Test based on moments (reflection)
  - Scaling factor

* D. Vranic, D. Saupe, and J. Richter. Tools for 3D-object Retrieval: Karhunen-Loeve Transform and Spherical Harmonics. In Proc. IEEE Workshop Multimedia Signal Processing, pages 293—298, 2001

Normalization of 3D objects

- Global transformation of the object with PCA

$$\tau(v) = s^{-1} \cdot F \cdot R \cdot (v - c)$$

s: scaling factor

F: Flipping matrix (diagonal)

R: Rotation matrix

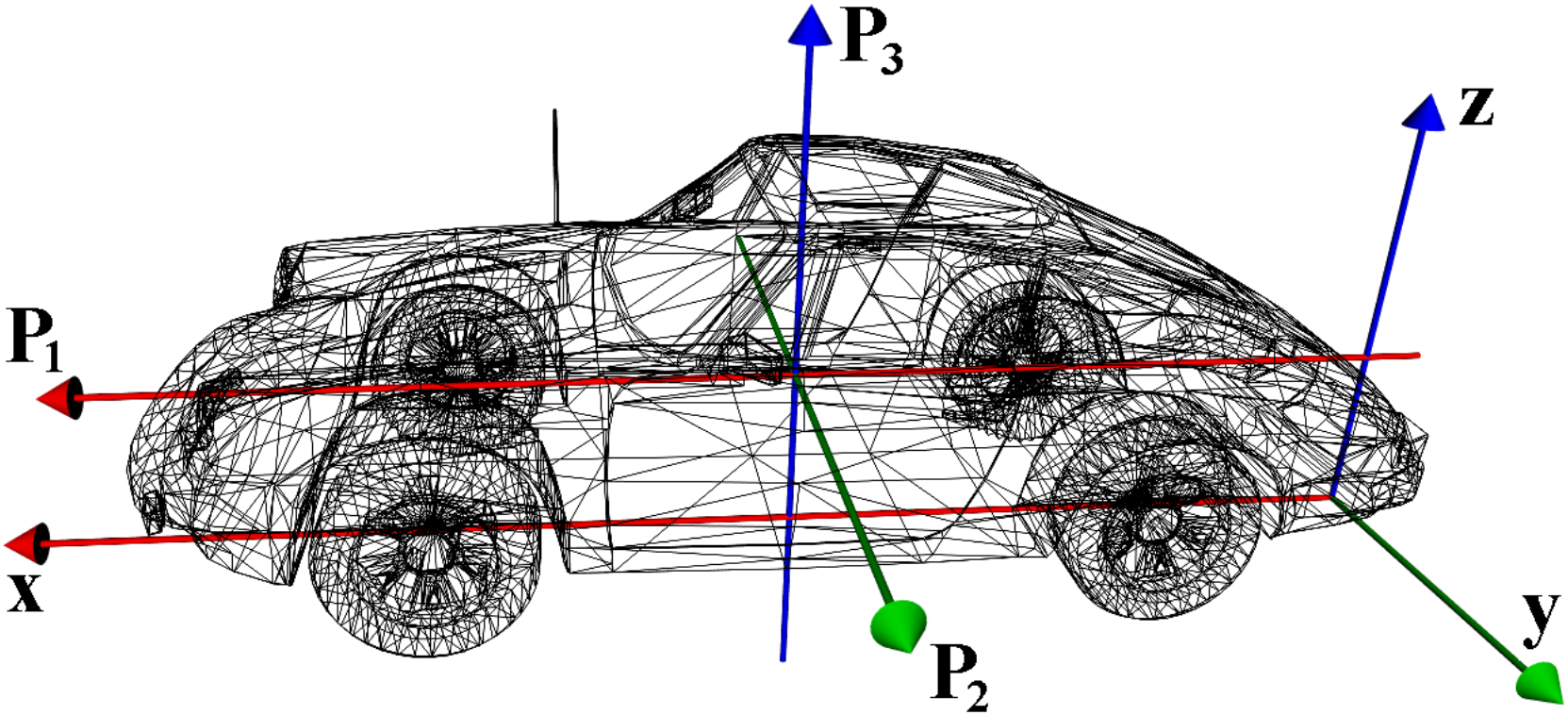
$(v - c)$: translated vertices, where

v: original vertices

c: center of mass

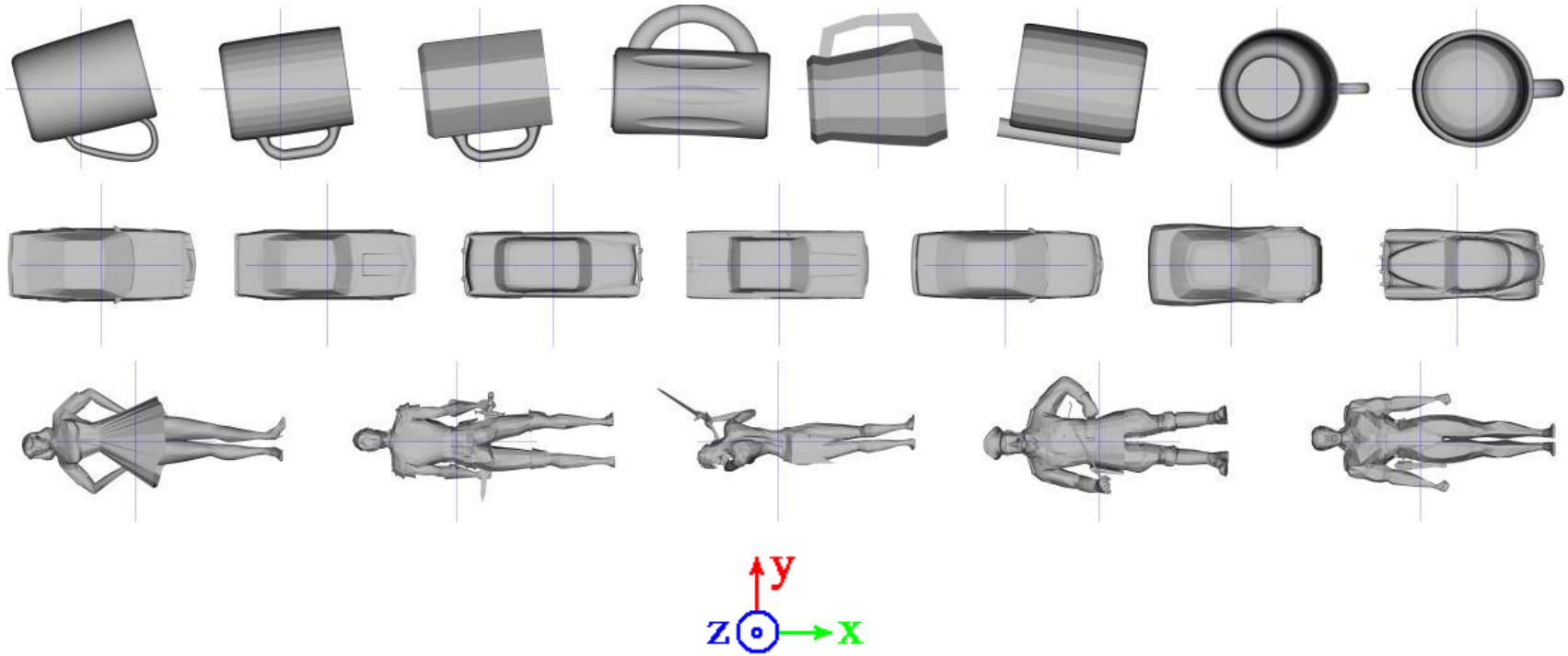
Normalization of 3D objects

- Example of normalization with PCA



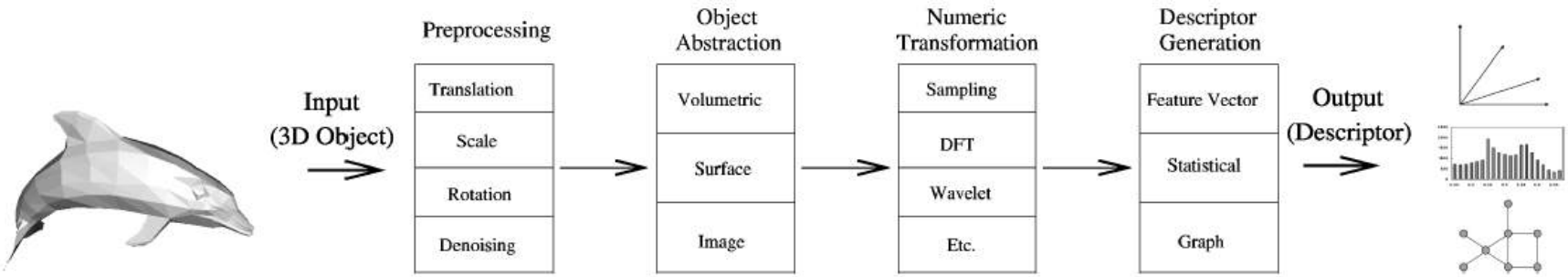
Normalization of 3D objects

- Examples of pose normalization with PCA



3D descriptors

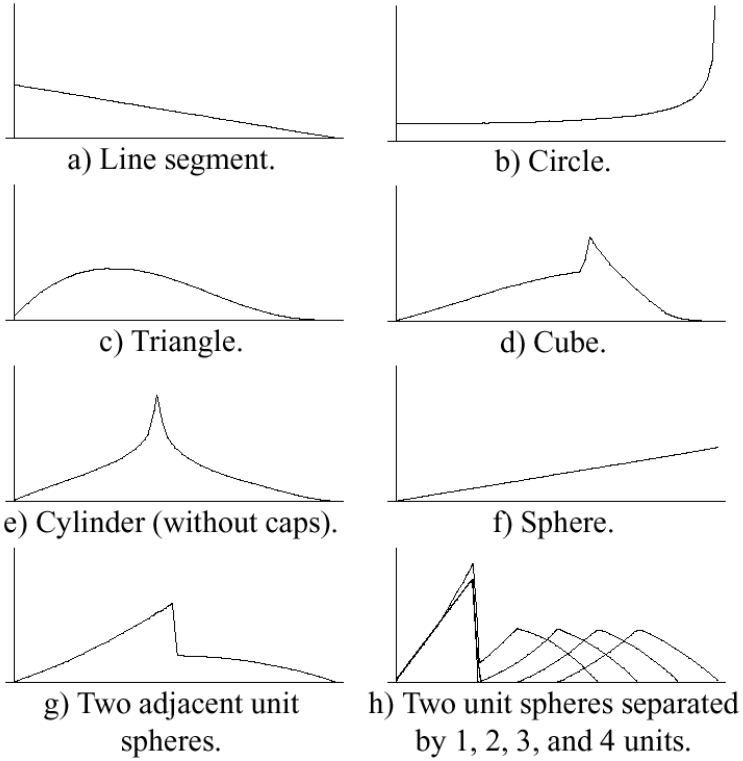
- Extracting 3D feature vectors*



* B. Bustos, D. Keim, D. Saupe, T. Schreck, and D. Vranic. Feature-based similarity search in 3D object databases. ACM Computing Surveys, 37(4):345-387, 2005

Shape distribution with D2*

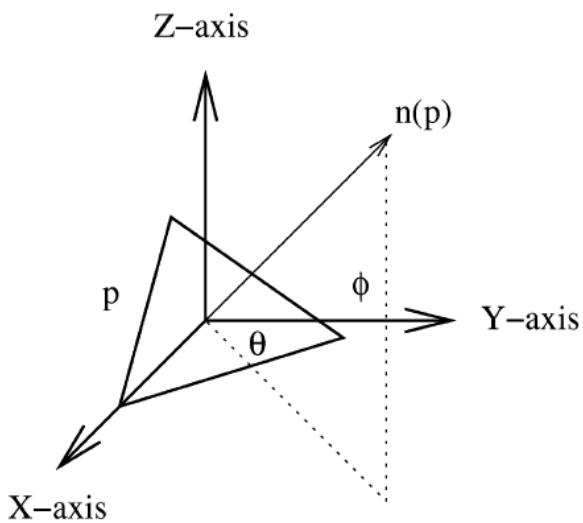
- Category: Statistical



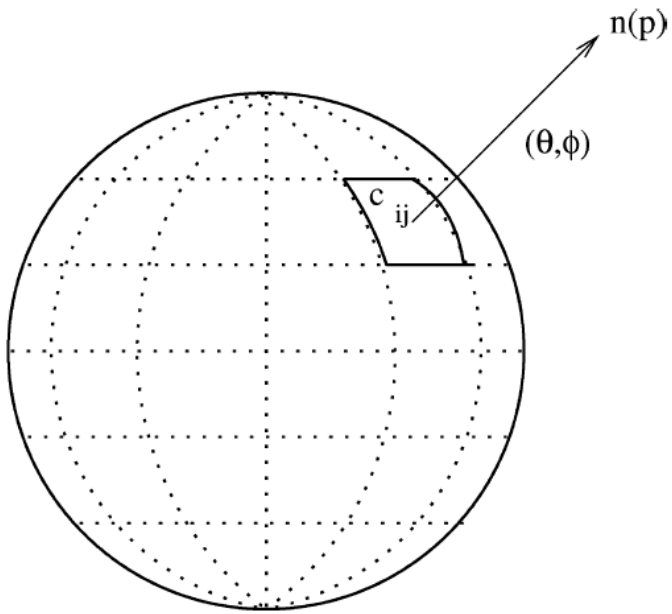
* R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Shape distributions. ACM Transactions on Graphics, 21(4):807—832, 2002

Extended gaussian image*

- Category: surface geometry



Polygon

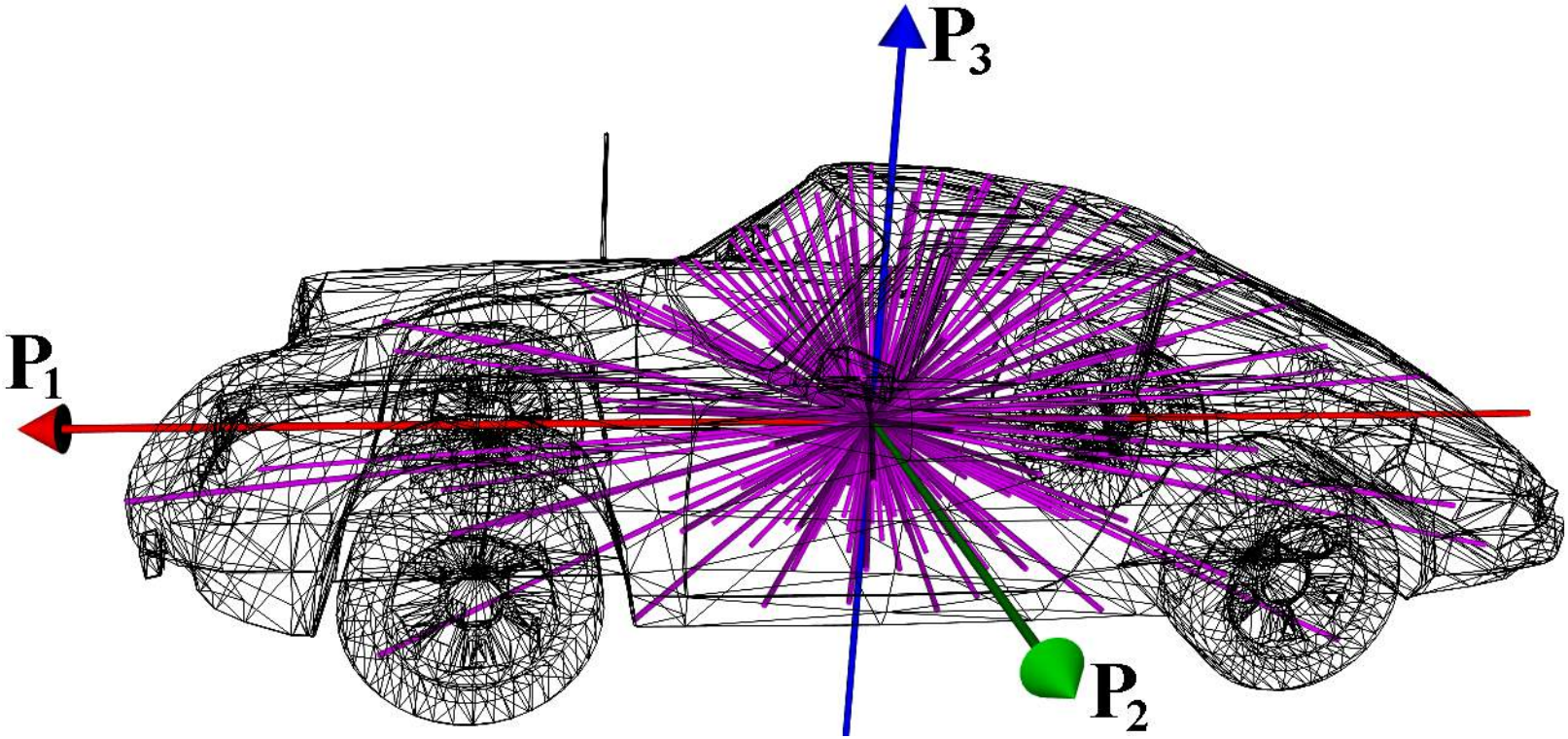


Gaussian sphere

* B. Horn. Extended Gaussian Image. Proceedings of the IEEE, 72(12):1671—1686, 1984

Ray-based*

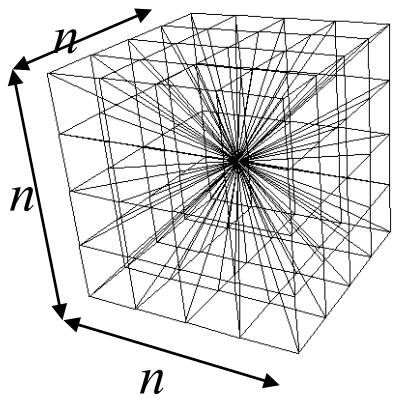
- Category: extension-based (dim=162)

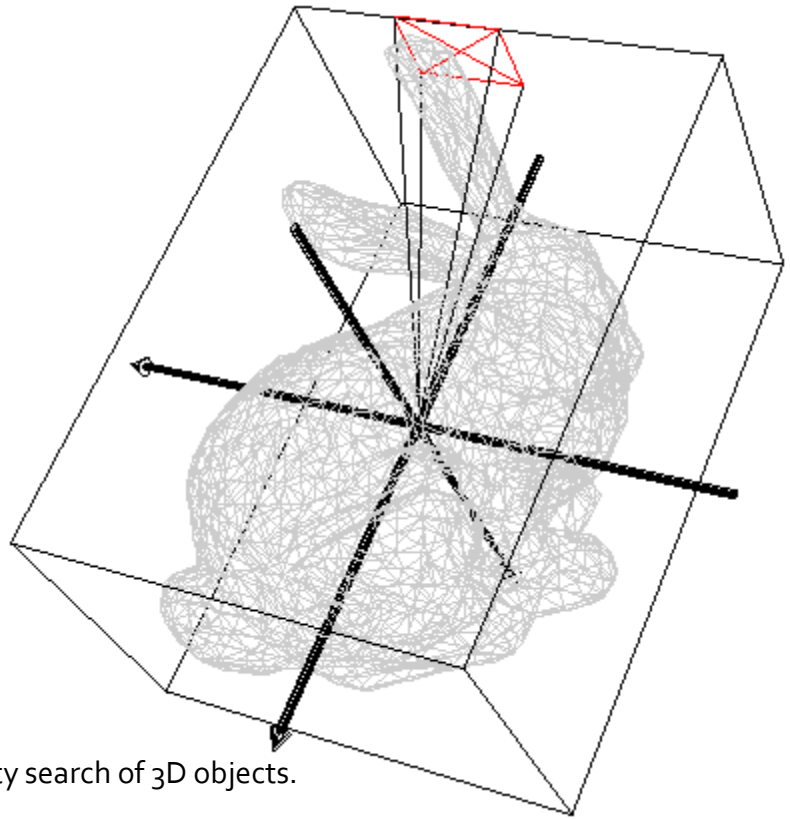


* D. Vranic and D. Saupe. 3D Model Retrieval. In Proc. Spring Conference on Computer Graphics and its Applications (SCCG'00), pages 89—93, 2000

Volume-based FV*

- Category: volume-based

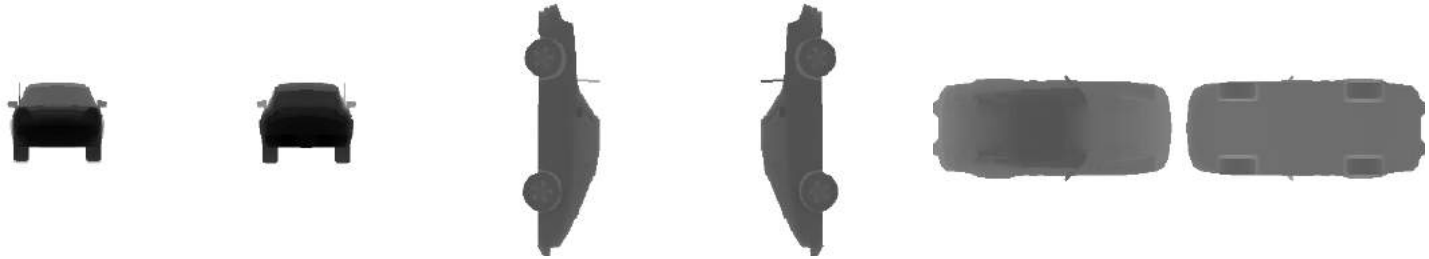




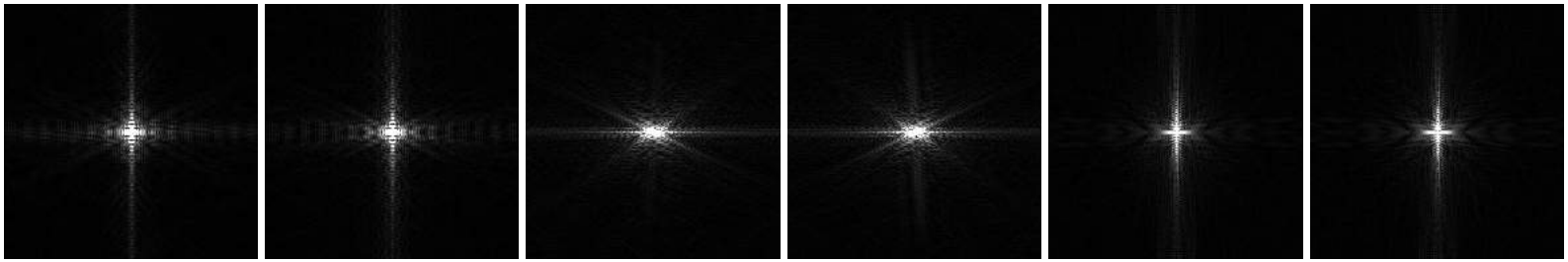
* M. Heczko, D. Keim, D. Saupe, and D. Vranic. Methods for similarity search of 3D objects. Datenbank-Spektrum, 2(2):54—63, dpunkt.verlag, 2002

Depth-buffer FV*

- Category: image-based



- Apply 2D discrete Fourier transform to the six images



* M. Heczko, D. Keim, D. Saupe, and D. Vranic. Methods for similarity search of 3D objects. Datenbank-Spektrum, 2(2):54—63, dpunkt.verlag, 2002

Disadvantages of global descriptors

- Many other global descriptors:
  - Image-based: best in practice
  - Combinations of descriptors (fixed, dynamic)
- But:
  - Global descriptors suffer from pose sensitivity
  - They are not well-suited for partial similarity
- New approach: Local descriptors!
  - Avoid the canonical pose problem
  - Support non-rigid and partial matching

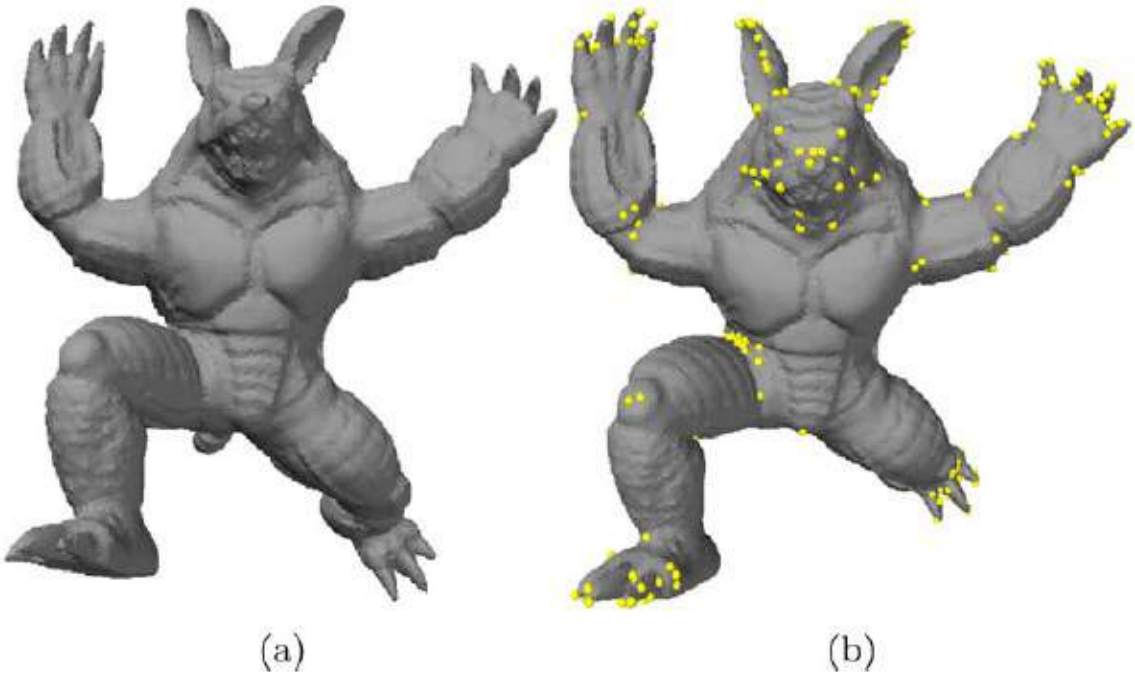
Local descriptors for 3DOR

- Main approach
  - Each shape is represented as a set of descriptors
  - The matching is performed by finding correspondences
- Example: Shape Google *
  - Heat kernel signatures
  - Bag-of-features approach

* Ovsjanikov, M., Bronstein, A.M., Guibas, L.J., Bronstein, M.M.: Shape Google: a computer vision approach to invariant shape retrieval. In: Proc. Workshop on Non-Rigid Shape Anal. and Deform. Image Alignment, 2009

Key-points

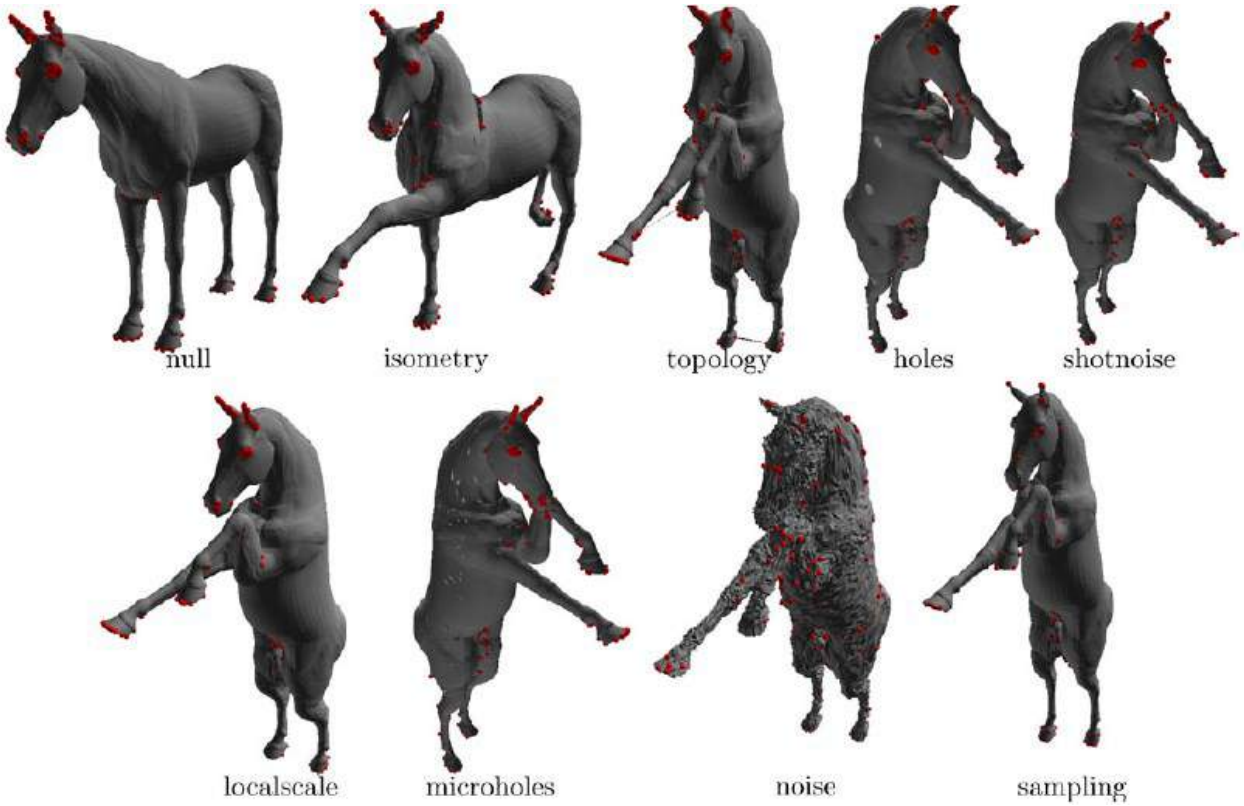
- Interest points on the 3D surface
  - Harris 3D method *



* I. Sipiran and B. Bustos. Harris 3D: A robust extension of the Harris operator for interest point detection on 3D meshes. The Visual Computer, 27(11):963-976, 2011

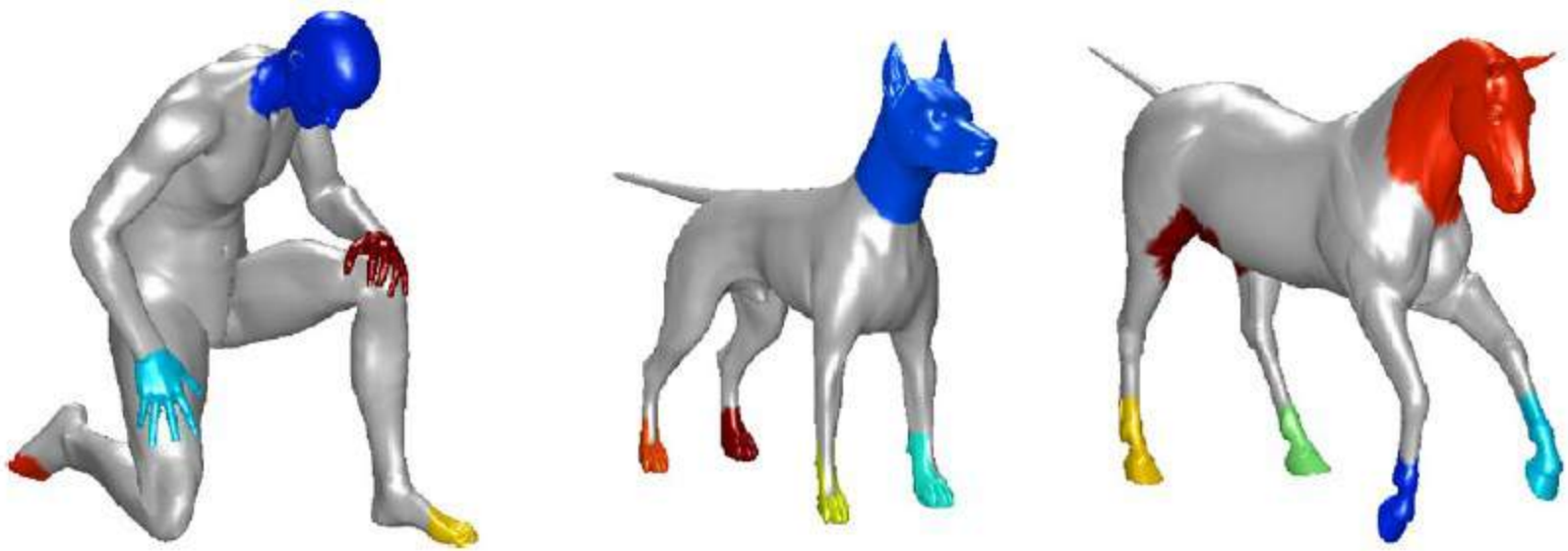
Key-points

- Harris 3D is a robust interest points detector



Key-components

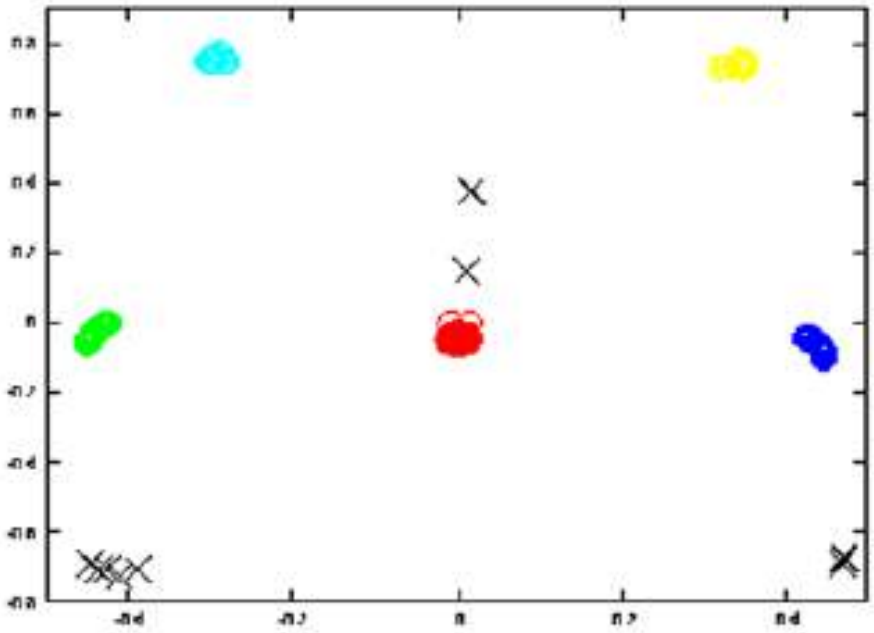
- Regions of interest on the 3D shape *
  - Based on key-points



* I. Sipiran and B. Bustos. Key-components: Detection of salient regions on 3D meshes. The Visual Computer 29(12):1319-1332, 2013

Key-components

- Main idea: clustering of key-points in geodesic space

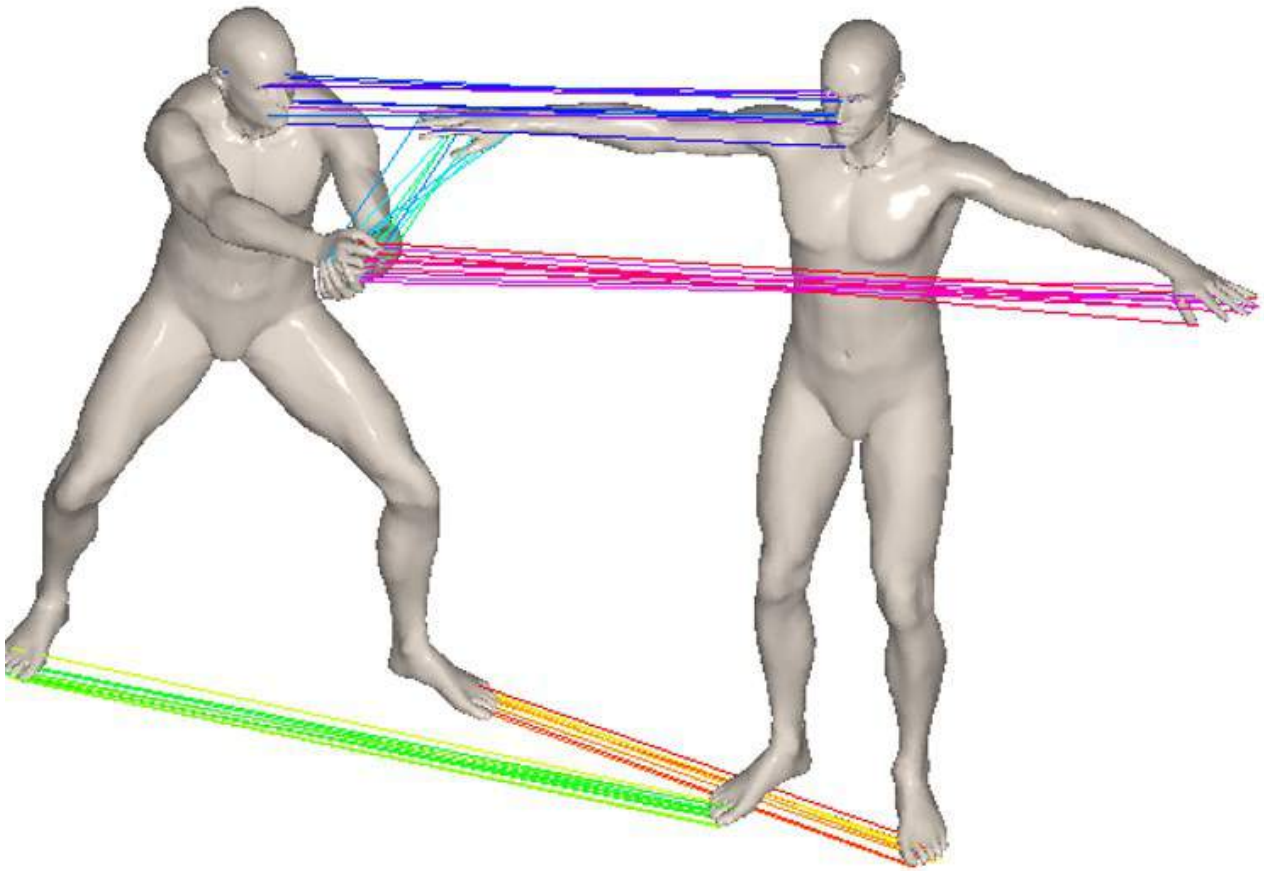


Key-components

- Key-components are robust

Fig. 5 Key-components detected on shapes with several transformations. From left to right: null shape, isometry, microholes, local scale, noise, topology, holes, sampling, and shot noise. Color are arbitrary

Finding correspondences *



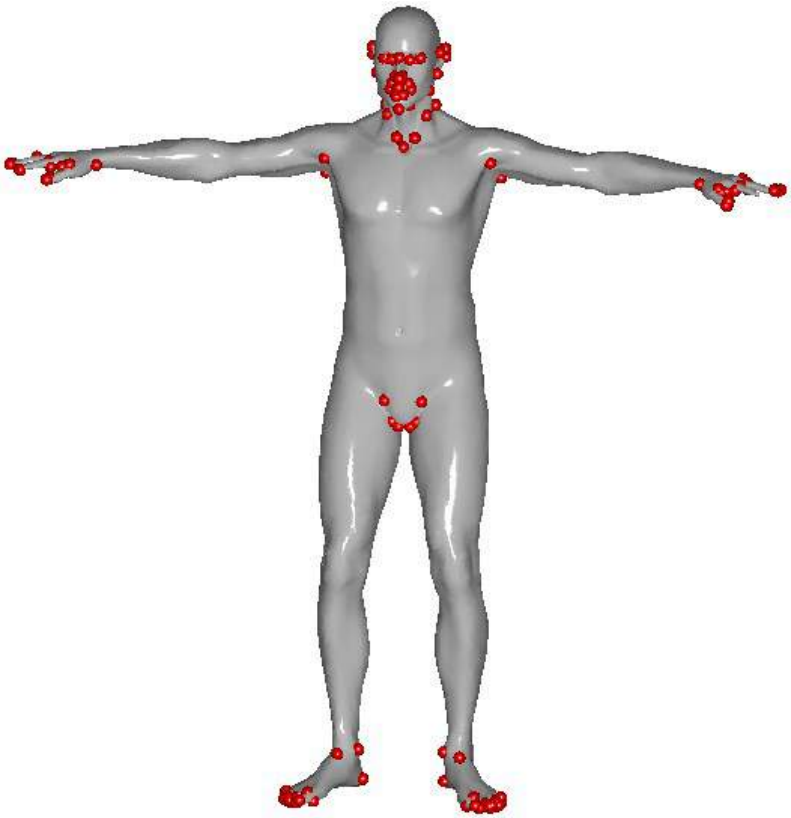
* I. Sipiran and B. Bustos. A fully hierarchical approach for finding correspondences in non-rigid shapes. In Proc. 14th IEEE International Conference on Computer Vision (ICCV'13), pages 817-824. IEEE, 2013

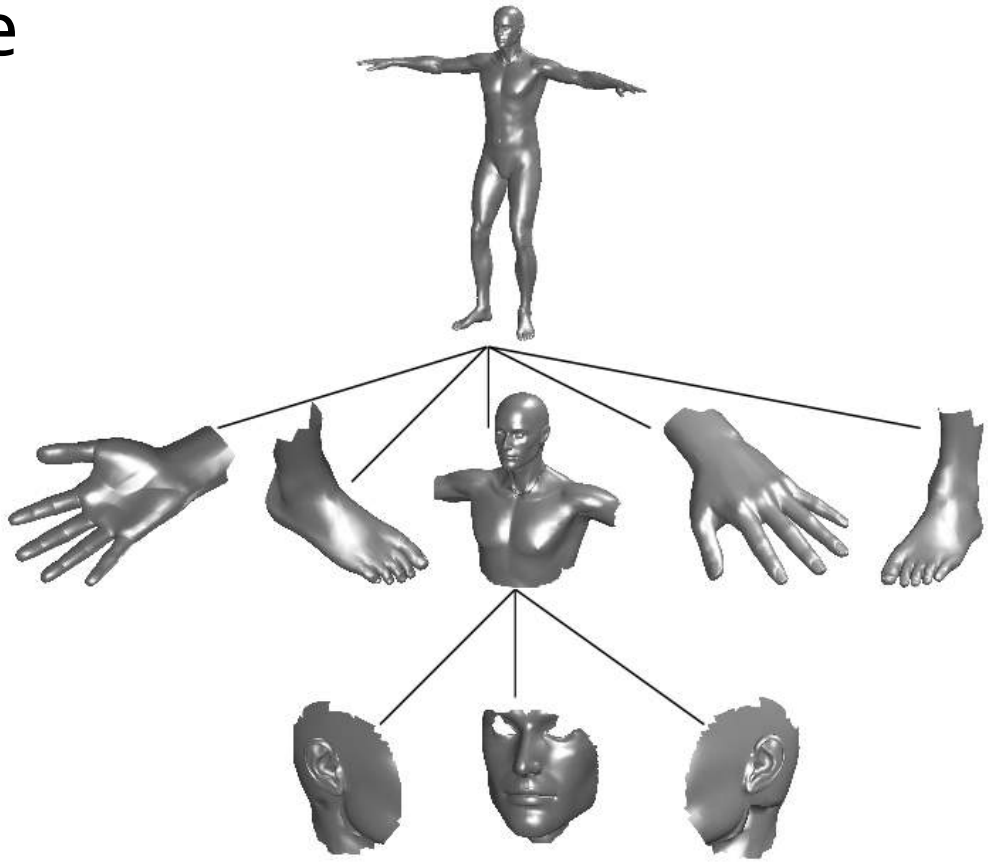
Finding correspondences

- Pre-processing
  - Key-point detection: Harris 3D algorithm
  - Compute geodesic distances between every pair of key-points
  - Discard key-points with low density
  - For each remaining key-point, compute descriptors
    - Internal nodes: Heat Kernel Signature [Sun et al. 2009]
    - Leaves: Wave Kernel Signature [Aubry et al. 2011]

Finding correspondences

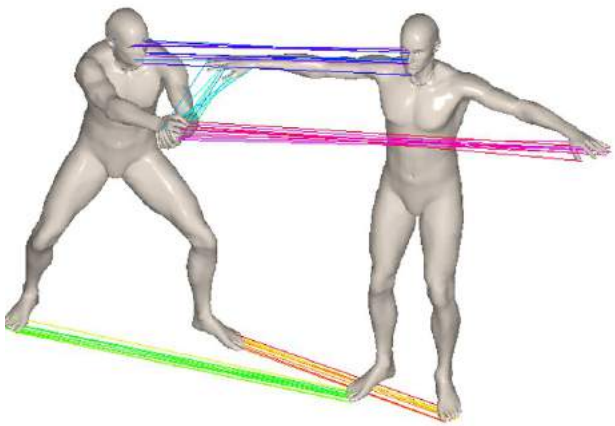
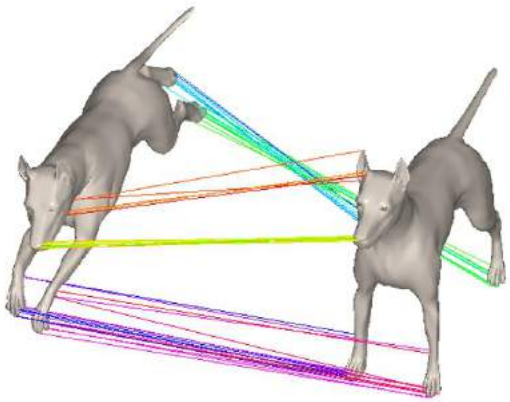
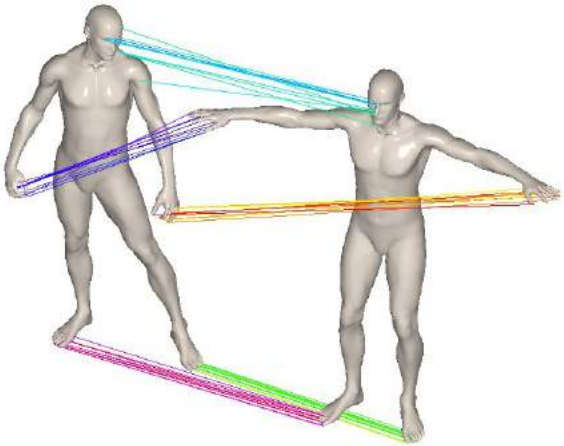
- Decomposition tree





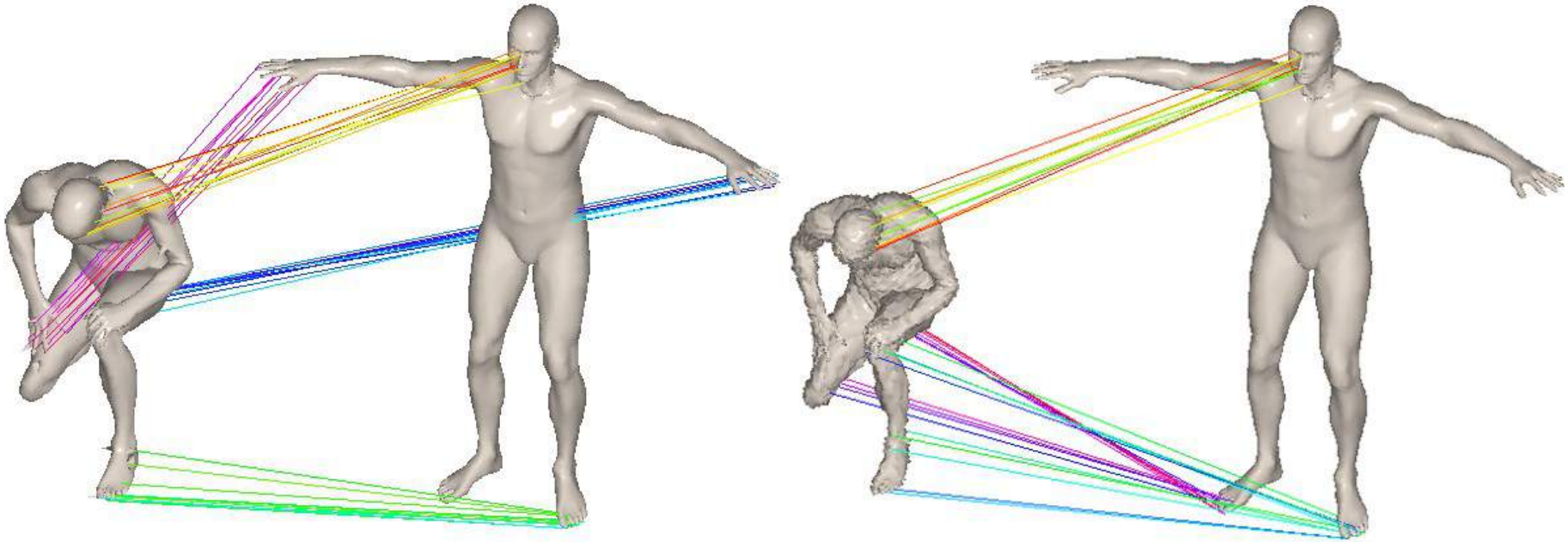
Experimental evaluation

- Some results

Experimental evaluation

- Strong perturbation of the data



course:
Multimedia Retrieval

lecture 10:
Feature extraction from video, audio, music and melody

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

- video retrieval
  - segmentation, boundary detection
  - keyframe extraction
  - specific MPEG7 descriptors for video
  - video copy detection
- audio & music retrieval
  - MPEG7 audio low-level descriptors
  - MFCCs – alternative audio descriptor
  - melody retrieval

Video retrieval

Video representation

- video record
  - true „multi-media" document
    - sequence of frames (still images)
    - audio track
    - text (subtitles)
  - for querying the visual (image) component is the most important
- video compression
  - independent images (mjpeg)
  - differences in images (mpeg)

Content-based Video Retrieval (CBVR)

- extension of content-based image retrieval (CBIR), specific in
  - evolution of objects in image, camera motion
  - more complex querying
- applications of CBVR
  - TV news
  - commercials (advertisment)
  - music clips
  - remote lecturing (e-learning)
  - medicine

Video segmentation

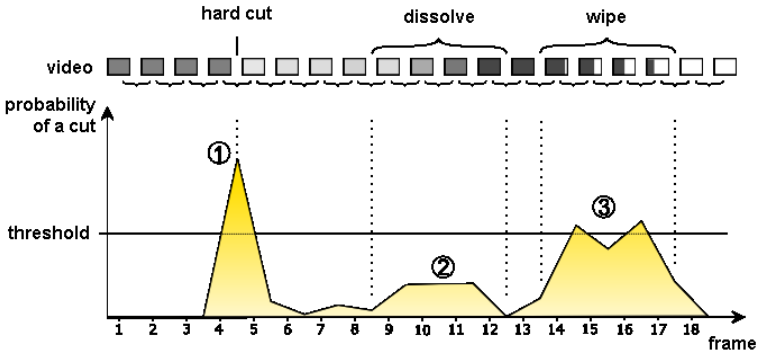
- 4 semantic levels of segmentation granularity in time
  - individual frames
    - no analysis
  - shot
    - sequence of frames continuous in time and space
    - suitable for further analysis (keyframes)
  - scene
    - set of semantically related shots
  - entire video

Shot boundary

- shot boundary
  - cut transition
    - sudden extreme change between two frames
  - gradual/continuous transition
    - fade – gradual transition between sequence and still image (fade in/fade out)
    - dissolve – gradual transition between two sequences (one fades out, one fades in)
    - wipe, etc. – visual cut edit
- boundary detection
  - image-based distance between adjacent frames exceeding a certain threshold

Shot boundary detection

- two phases
  - scoring – each pair of consecutive frames of a video is given a similarity score
    - sum of absolute differences
      - compared pixel by pixel (very sensitive)
    - histogram differences
      - difference between the histograms (more robust)
    - edge change ratio
      - comparing *edge pictures* (best performing algorithms for scoring)
  - decision – cut is detected from scores
    - fixed threshold
    - adaptive threshold
    - machine learning



Keyframes

- keyframe = basic element for construction of video descriptors
- keyframe extraction
  - the first frame of a shot (after boundary)
    - insufficient when the activity in shot becomes higher
  - clustering based on color features
    - entire video material is first grouped into clusters, each contains frames of similar visual content

MPEG7 for video

- within the class of visual MPEG7 descriptors
- reuse of descriptors for still images
  - color, shape, texture subclasses
    (see previous lecture)
- video-specific descriptors
  - GoF/GoP descriptor in the color descriptor subclass
  - localization subclass
  - motion subclass

MPEG7 – GoF and GoP descriptor

- group-of-frames/group-of-pictures (GoF/GoP)
  - GoP structure, MPEG7 GoF/GoP descriptor
- GoP structure
  - group of successive pictures within a coded video stream (e.g., in MPEG-1)
  - picture types
    - **I-frame** (intra coded picture) – fixed image independent of other picture types. Each GoP begins with this type of picture.
    - **P-frame** (predictive coded picture) – motion-compensated difference information from the preceding I- or P-frame.
    - **B-frame** (bidirectionally predictive coded picture) – difference information from the preceding and following I- or P-frame within a GoP
  - two numbers, M (distance between two anchor I- or P-frames) and N (distance between two I-frames), e.g., M=3, N=12, IBBPBBPBBPBBI

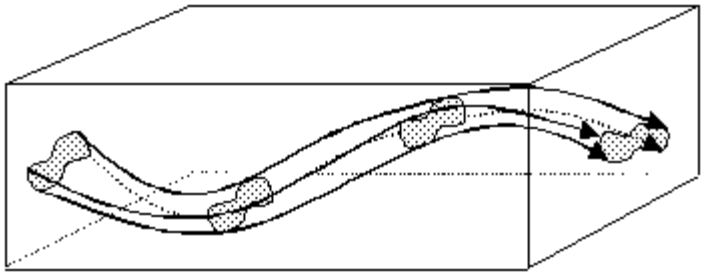
MPEG7 – localization descriptors

- subclass of visual descriptors (needed for motion descriptors)
  - Region Locator
    - localization of regions within images or frames by specifying them with a brief and scalable representation of a box or polygon
    - 2 steps
      - extraction of vertices of the region to be localized
      - localization of the region within the frame/image

(source: Charlie Dagli, Introduction to MPEG7, 2009)

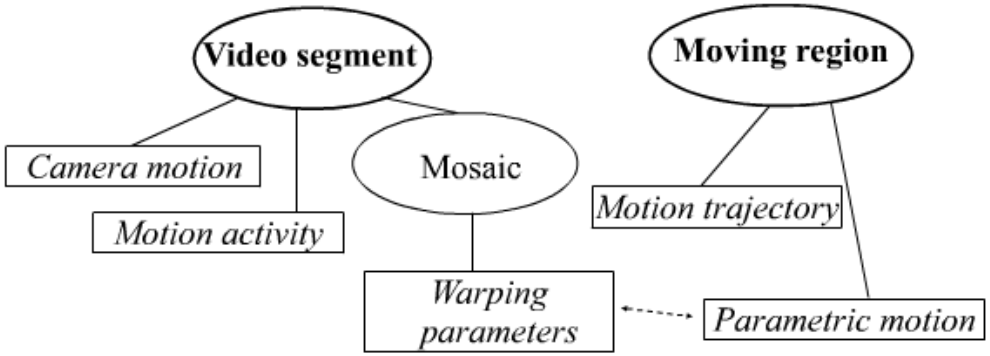
MPEG7 – localization descriptors

- Spatio-temporal Locator
  - spatial-temporal regions in a video sequence (moving object regions)
  - can describe both spatially connected and non-connected regions
  - provides localization functionality for video



MPEG7 – motion descriptors

- subclass of visual descriptors specific to video
  - Motion Activity
  - Camera Motion
  - Parametric Motion
  - Motion Trajectory



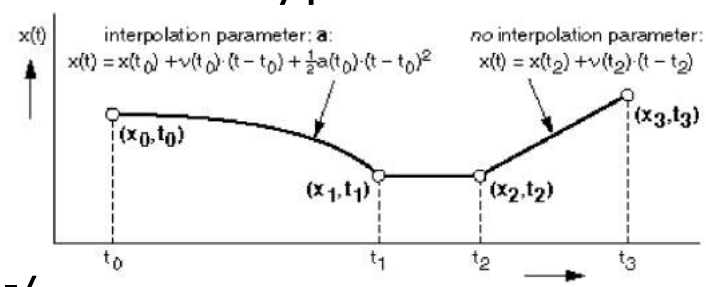
- high-level features
  - but cannot be (easily) extracted from the video content
    - easier when localization descriptors are provided
  - some supposed as kind of metadata

MPEG7 – Motion Activity Descriptor

- describes „intensity of action" in a video segment
  - intent: capture the gross motion characteristics of the video segment
- four attributes
  - intensity of activity – high/low score (1-5)
    - quantized standard deviation of motion vectors
      - e.g., goal scoring in soccer = high action vs. news studio shot = low action
  - direction of activity – dominant direction (if any)
    - spatial distribution of activity
    - temporal distribution of activity
- descriptor extraction
  - simple operations on the motion vector magnitudes – most often these are MPEG-1/2/4 motion vectors

MPEG7 – Motion Trajectory Descriptor

- high-level feature associated to a moving region
  - describes displacement of objects in time
- defined as the spatio-temporal localization of one of its representative points (e.g., center) as a list of keypoints (x,y,z,t)



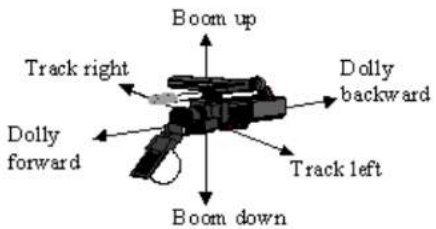
- descriptor extraction
  - either through the alpha channel of MPEG-4
  - or by carrying out a segmentation prior to computing the trajectory to mark the regions
    - if the segmentation is already done, the descriptor is quite easy to extract

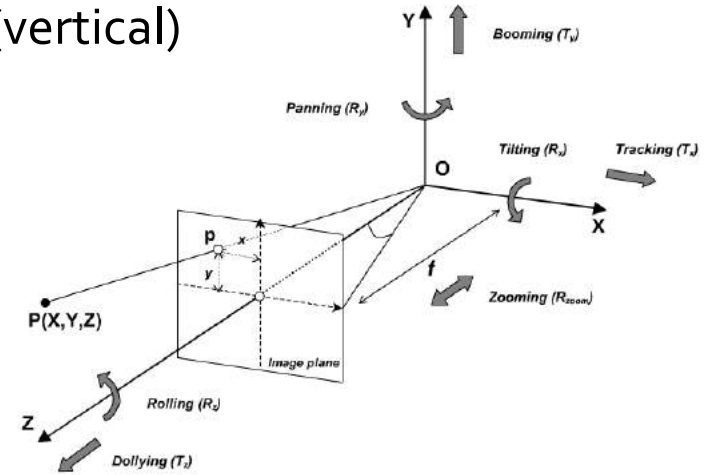
MPEG7 – Parametric Motion Descriptor

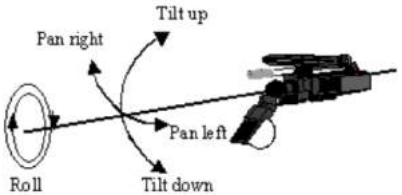
- describes the motion of objects as a 2D parametric model
  - affine models – translations, rotations, scaling (+ combinations)
  - planar perspective models – global deformations with perspective projections
  - quadratic models – more complex movements
- the descriptor characterizes the evolution of arbitrarily shaped regions over time in terms of a 2D geometric transform
- descriptor extraction
  - relies on computationally intense operations on the motion field to fit the various models
  - most computationally complex of the MPEG7 motion descriptors

MPEG7 – Camera Motion Descriptor

- camera operations
  - fixed operation (no motion)
  - translation operations
    - tracking (horizontal traverse), booming (vertical)
    - dollying (forward/backward)
  - rotation
    - panning (horizontal rotation)
    - tilting (vertical rotation)
    - rolling (rotation along optical axis)
  - zoom

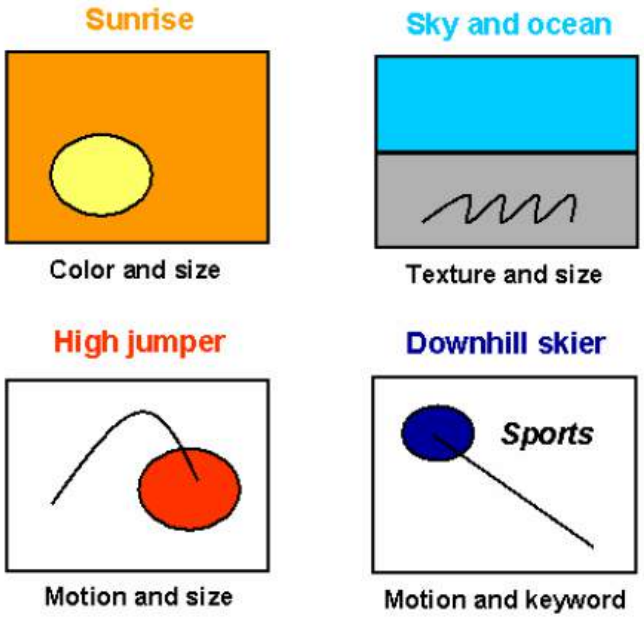






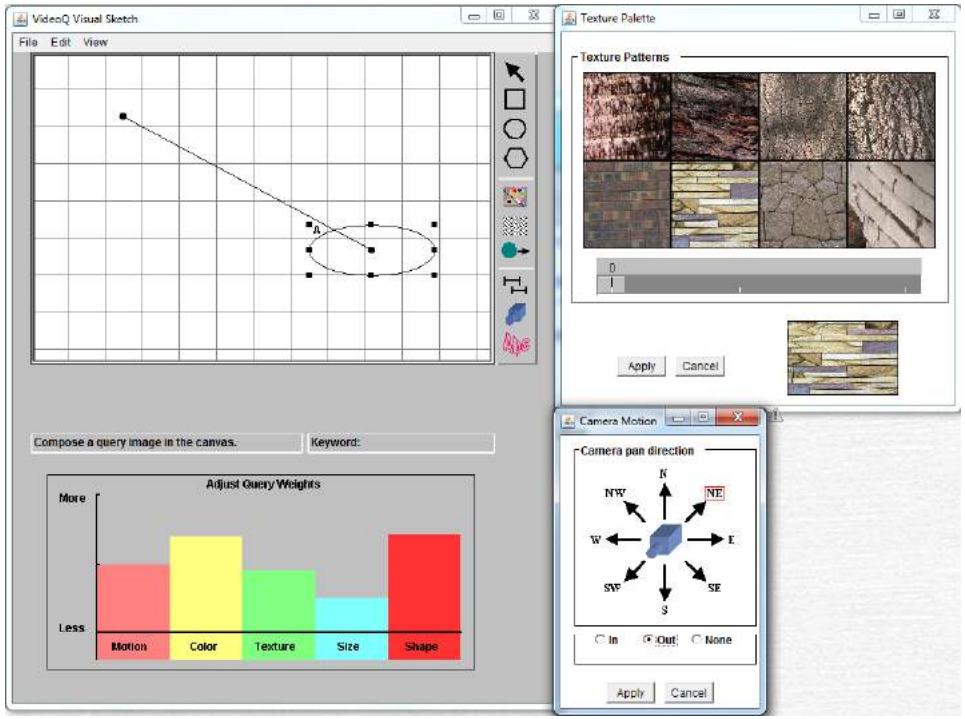
Advanced video querying – VideoQ

- combined query (weighted aggregation)
  - text
  - region size
  - color
  - texture
  - motion (region/camera)
- catalogue of template queries



Advanced video querying – VideoQ

- VideoQ, http://persia.ee.columbia.edu:8080/



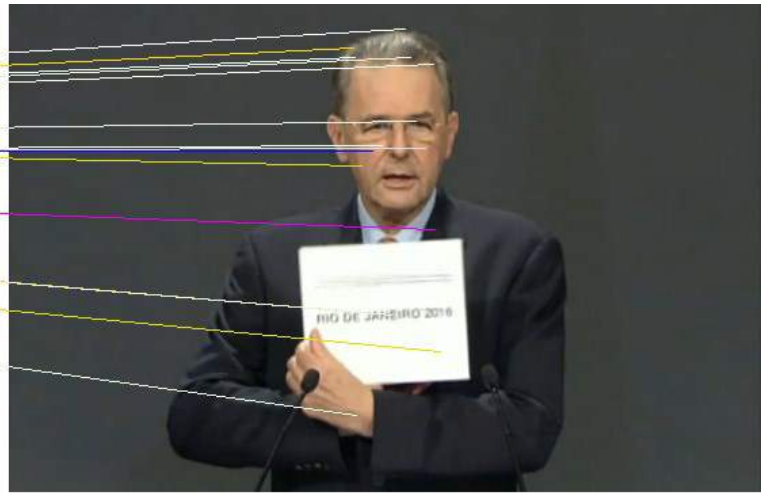
source: VideoQ, http://persia.ee.columbia.edu:8080/

Video copy detection

- retrieving all the modified versions of a video
  - video copy = tolerated transformations of source
    - quality transformations
      - blur, noise, re-encoding, change of brightness/contrast
    - post-production transformations
      - cropping, graphics/text insertion, picture in picture, background replacement
  - why?
    - video clustering, cleaning
      - e.g., youtube is full of edited video copies, collages, etc.
    - video editing
      - e.g., known commercials localization in a video stream (TV broadcasting)
    - included in TRECVID

Video copy detection

- example (picture in picture/cropping)



Video copy detection

- several approaches based on image retrieval
  - video = set of keyframes, aggregation
  - global descriptors
    - independent image features, contiguous stream of NN queries
    - descriptors include temporal activity (video = 3D image)
  - local descriptors
    - [AJ] features extracted in keyframes corresponding to extrema of the global intensity of motion
    - [VoCopT] space-time interest points correspond to points where the image values have significant local variation in both space and time

Audio retrieval

Fundamentals

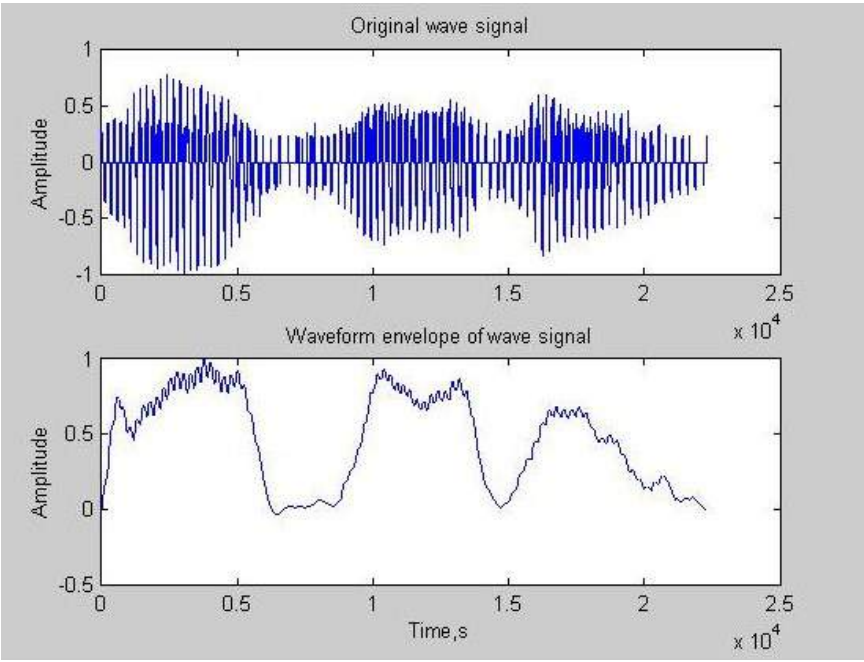
- acoustics
  - branch of physics concerned with the properties, production, and transmission of sound
  - sound = vibrations of a source (e.g., music instrument) which are transmitted through the air to the ear-drum

Fundamentals

- acoustic properties
  - pitch – speed of vibrations (frequency, 20 Hz - 20 kHz hearable)
    - logarithmic scale – notes (octaves), e.g., A4 = 440 Hz, A5 = 880 Hz, etc.
    - other scales (e.g., mel scale) – logarithmic at the high-frequency end, and linear at the low end
  - loudness – amplitude of the vibrations
    - logarithmic scale – decibels
      - logarithms of atmospheres – air pressure on ear-drum
  - sound quality
    - other properties, e.g., timbre (depending on periodicity of the signal, etc.)

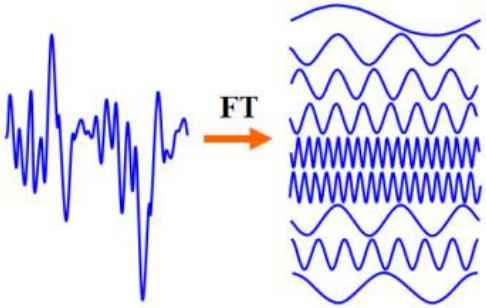
Fundamentals

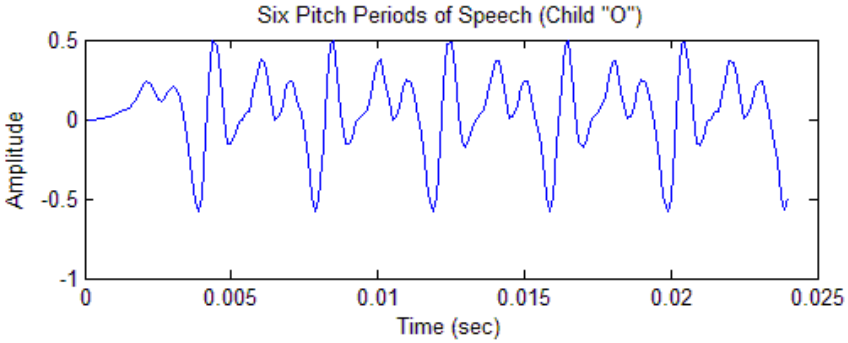
- sound waveform
  - complicated mixture of vibrations
  - digitized one-dimensional signal
- waveform envelope
  - maxima (+minima) in the (subsampled) signal

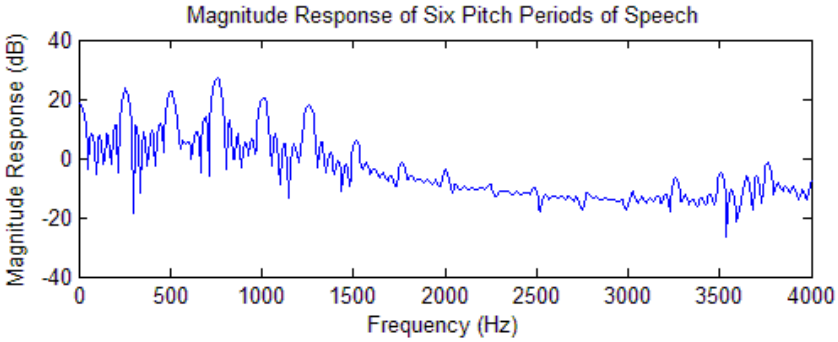


Fundamentals

- sound (power) spectrum
  - the amount of vibration at each individual frequency
  - sound spectrum as a sound „recipe"
    - take this amount of that frequency, add this amount of that frequency etc, until you have put together the whole, complicated sound.
    - reverse recipe: discrete **Fourier transform** (DFT)

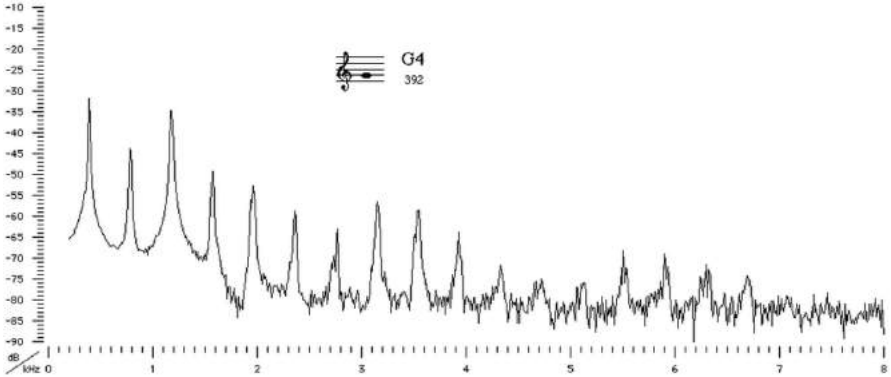






Fundamentals

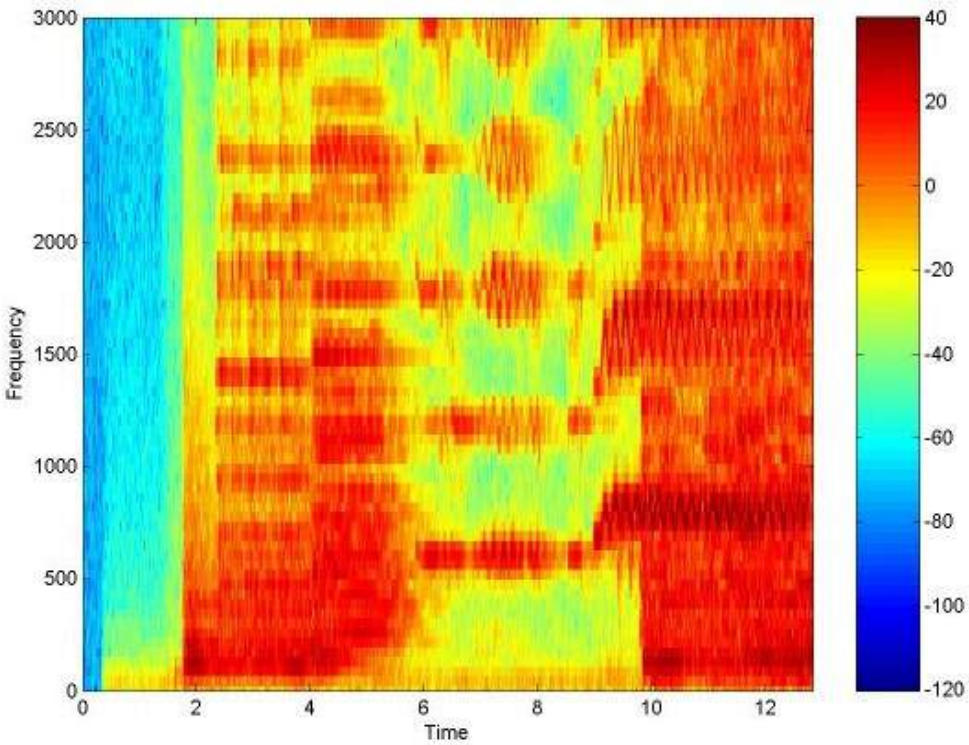
- harmonic series
  - spectrum with peaks at integral multiples of a base frequency
  - example, note G4 played by a flute (400 Hz base frequency)



- timbre
  - the human ear doesn't hear the harmonics as distinct pitches, but rather as the tone color (also called the *timbre*) of the sound, which allows the ear to distinguish one instrument or voice from another, even when both are playing the same pitch
  - about 60 ms needed to recognize timbre (below 4 ms – atonal click)

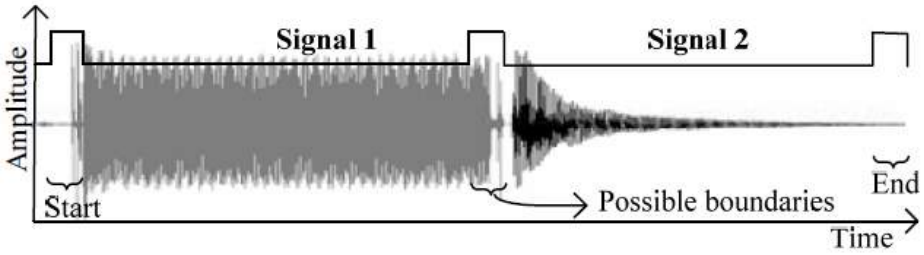
Fundamentals

- spectrogram
  - visual representation of an acoustic signal in both, frequency and time domain
  - multiple spectra represented as a pixel columns, where pixel color = amplitude

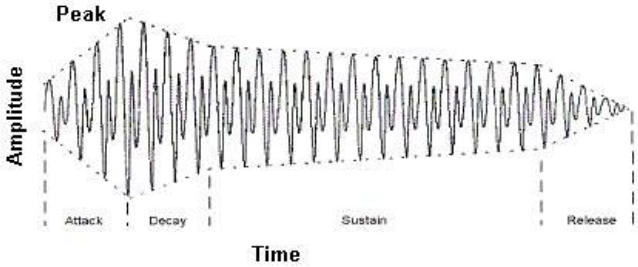


Fundamentals – audio segmentation

- segmentation of audio signal needed
  - unless short sounds assumed (door slam, dog bark, raining)
- based on acoustic analyses
  - identifying transitions
    - sudden changes in the signal's acoustic features
  - e.g., concert interleaved by applauses/silence

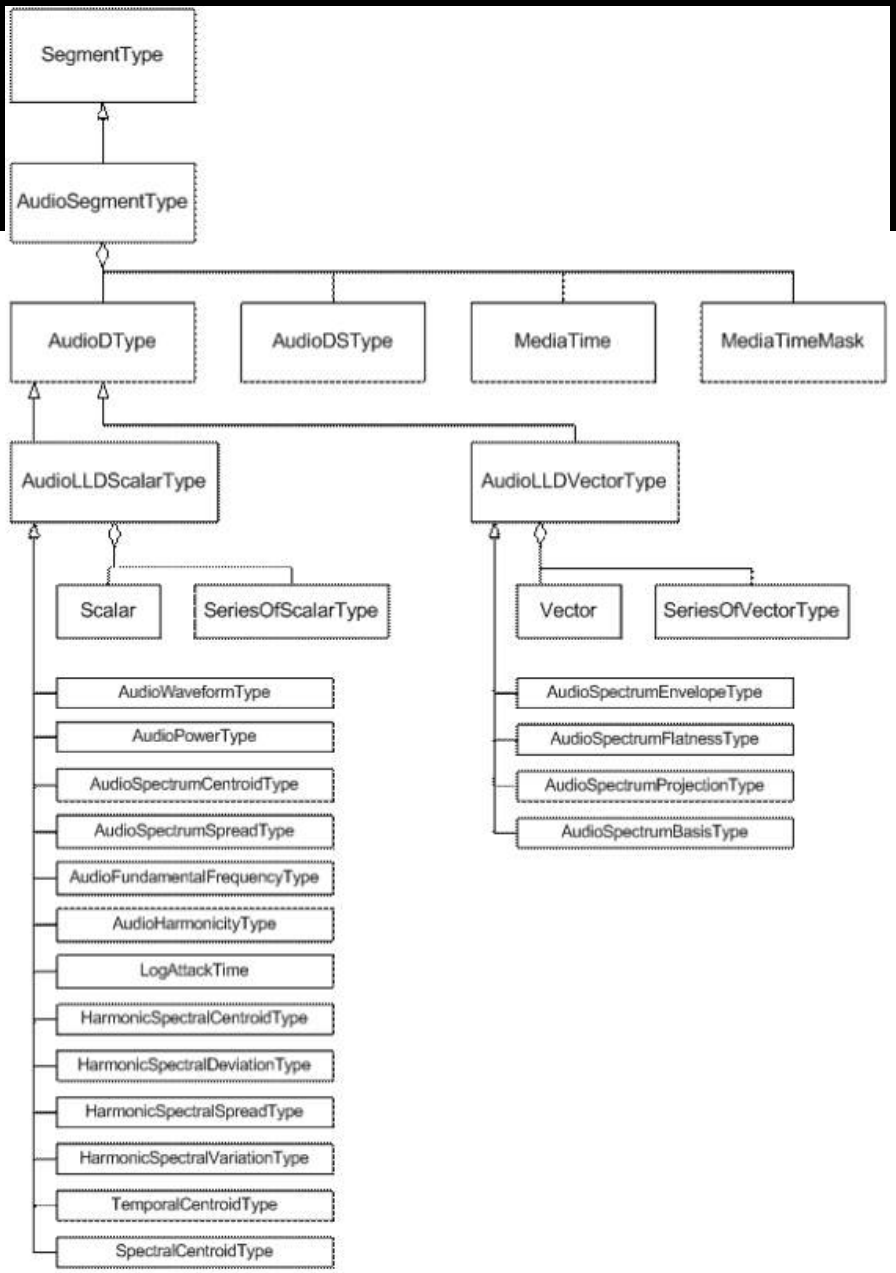


- sound of musical instrument has typical evolution
  - attack – initial run-up from silence to peak
  - decay – subsequent run-down from attack peak to sustain level
  - sustain – the main component (holding a key)
  - release – run-down to zero level (release of key)



MPEG7 audio descriptors

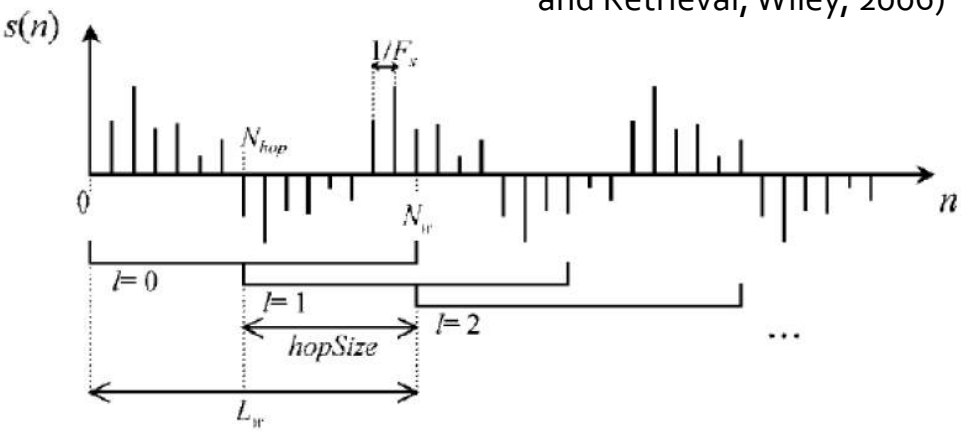
- 17 low-level descriptors →
  - each descriptor for an audio frame/segment
  - 2 basic descriptors
  - 4 basic spectral descriptors
  - 2 basic signal parameters
  - 2 temporal timbral descriptors
  - 5 spectral timbral descriptors
  - 2 spectral basis representations
- high-level (application-specific)
  - sound-effects
  - music instrument timbre
  - spoken content
  - melody contour
  - melody



MPEG7 audio descriptors

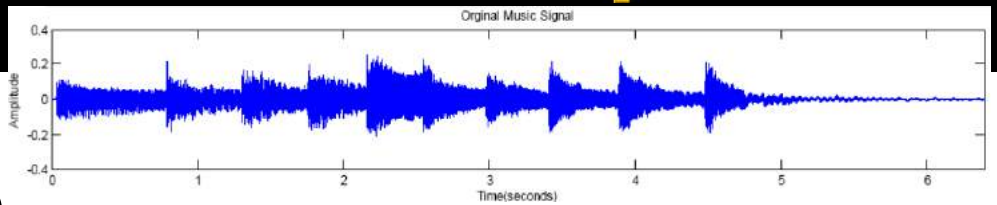
- frame (segment) based descriptors
  - n = index of time sample
  - s(n) = the signal input
  - l = index of frame
  - hopSize = interval between two frames
  - Lw = length of frame

(the following images from: Kim at al, MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval, Wiley, 2006)

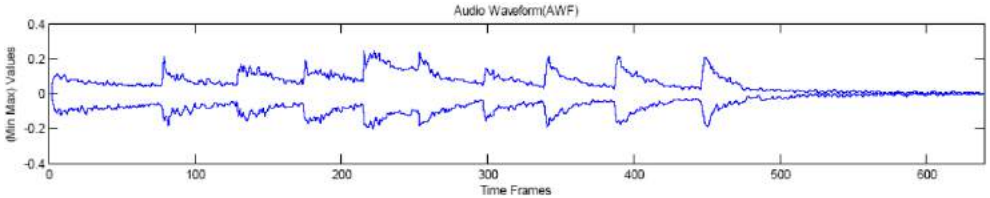


- silence segment descriptor
  - confidence [0, 1]
  - minDurationRef
    - minimum temporal threshold determining whether a signal portion is identified as a silent segment

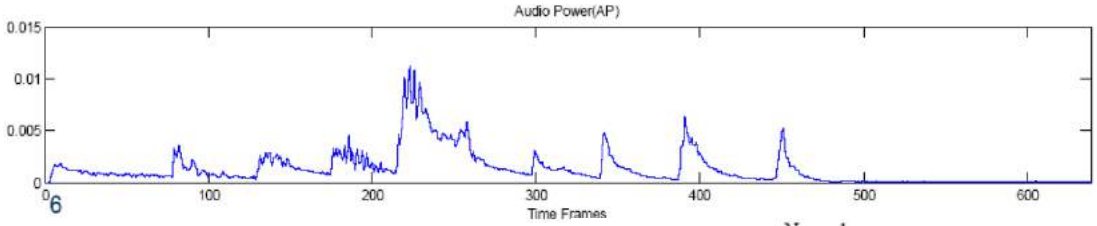
MPEG7 audio – Basic descriptors



- AudioWaveform (AWF)
  - shape of audio signal, in fact, waveform envelope
  - min/max samples within successive non-overlapping frames
    - Lw = hopSize = 10 ms



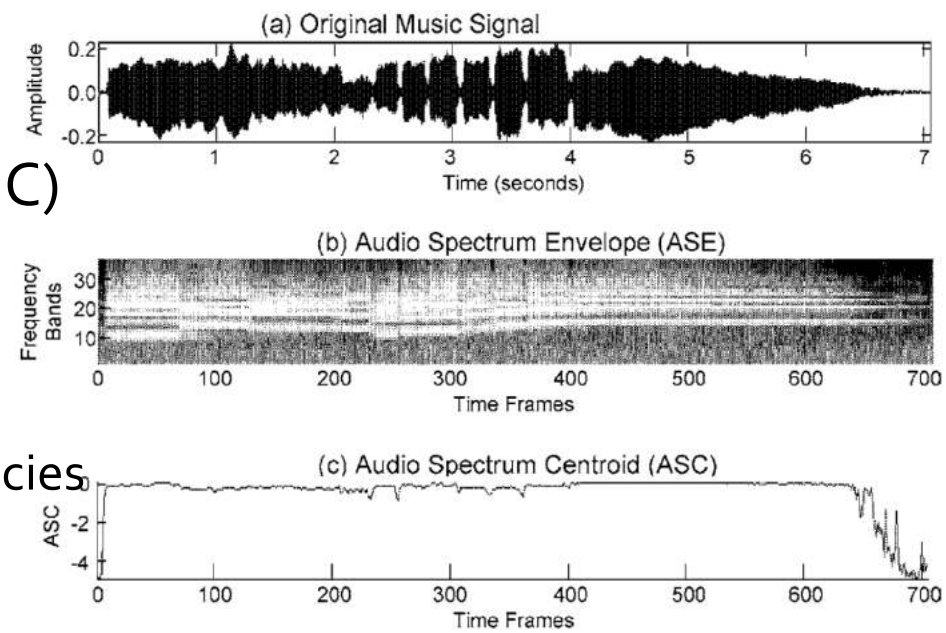
- AudioPower (AP)
  - temporally smoothed instantaneous power of the audio signal
  - measures the evolution of the amplitude as a function of time



$$AP(l) = \frac{1}{N_{hop}} \sum_{n=0}^{N_{hop}-1} \left| s(n + lN_{hop}) \right|^2$$

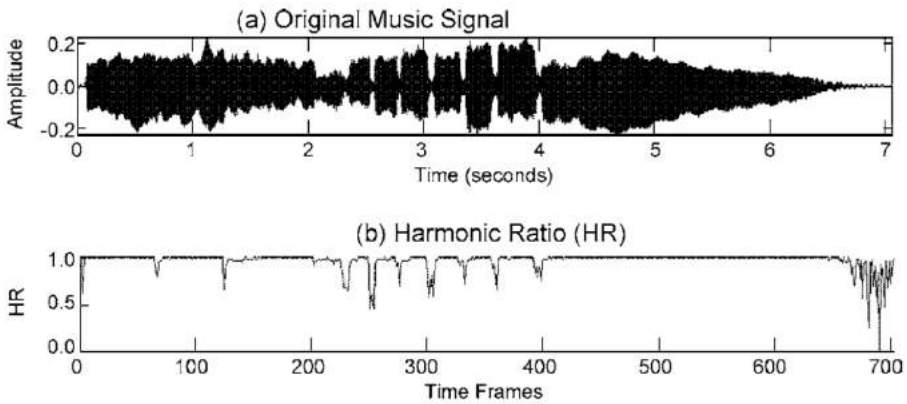
MPEG7 audio – Basic spectral

- AudioSpectrumEnvelope (ASE)
  - log-frequency power spectrum that can be used to generate a reduced spectrogram of the original audio signal
  - obtained by summing the energy of the original power spectrum within a series of **frequency bands**
  - bands are logarithmically distributed (human ear)
- AudioSpectrumCentroid (ASC)
  - the centre of gravity of a log-frequency power spectrum
  - indicates if a power spectrum is dominated by low/high frequencies
  - regarded as an approximation of the **perceptual sharpness**



MPEG7 audio – Basic signal parameters

- the basic signal parameters descriptors provide complementary information to basic spectral descriptors, by describing the degree of harmonicity of audio signals

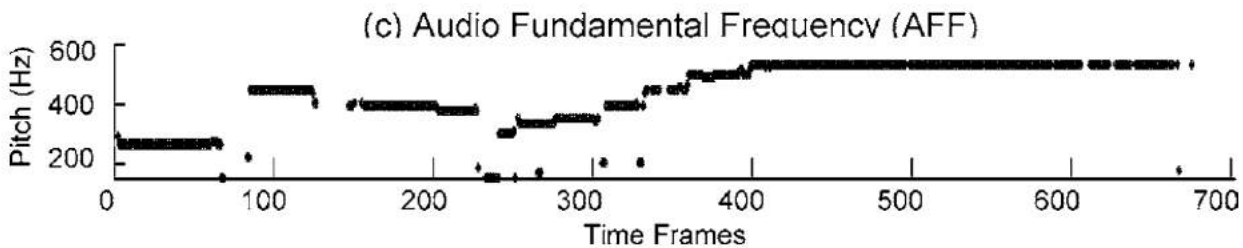


- AudioHarmonicity (AH)
  - HarmonicRatio (HR)
    - measure of the proportion of harmonic components in the spectrum
  - UpperLimitOfHarmonicity (ULH)
    - estimation of the frequency beyond which the spectrum no longer has any harmonic structure

- can be used for distinguishing between **harmonic sounds** *(*like musical sounds and voiced speech segments) and **non-harmonic sounds** (like noisy sounds and unvoiced speechsegments)

MPEG7 audio – Basic signal parameters

- AudioFundamentalFrequency (AFF)
  - provides estimations of the fundamental frequency in segments where the signal is assumed to be periodic
  - useful to get an approximation of the pitch of any music or speech signals
    - kind of monophonic musical score

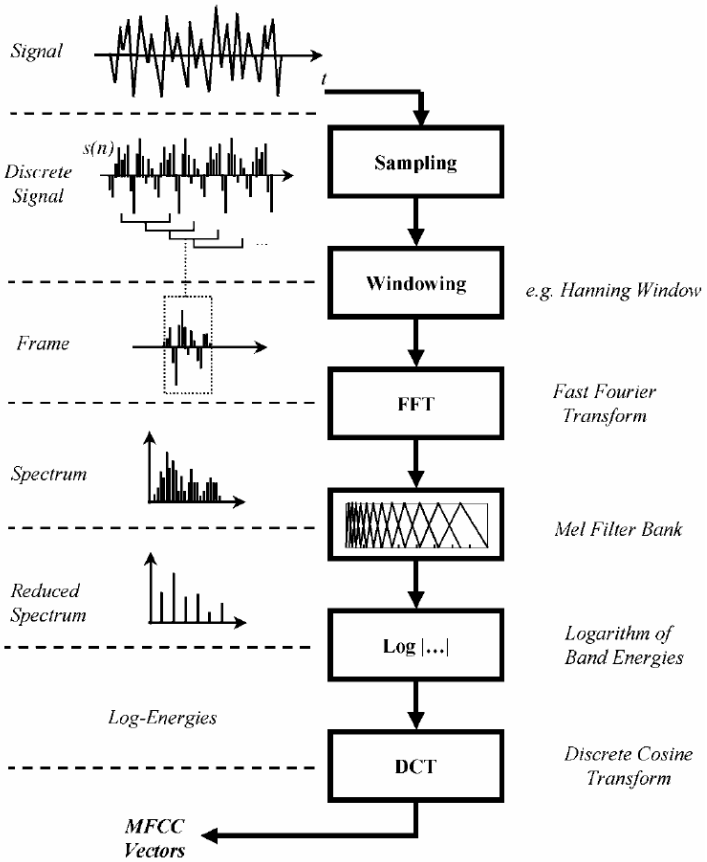


Alternative audio descriptors

- many classical low-level features used for sound are not included in the foundation layer of MPEG-7 audio
  - zero crossing rate (how many times the signal crosses zero axis)
  - spectral rolloff frequency (distinguishes voiced from unvoiced speech, e.g., whispering)
  - spectral flux (used for the separation of music from speech)
  - loudness
  - **Mel Frequency Cepstral Coefficients (MFCCs)**

Alternative audio descriptors - MFCCs

- Mel-Frequency Cepstrum Coefficients (MFCCs)
- excellent feature vector for representing the human voice and musical signals
  - especially speech recognition
  - music retrieval
    - genre classification
    - audio similarity
- mel-frequency cepstrum
  - linear cosine transform of a log power spectrum on a nonlinear mel-scale of frequency
- mel scale
  - scale of pitches judged by listeners to be equal in distance one from another
  - nonlinear scale – human frequency perception is more selective in the lower frequencies than in the higher ones
  - triangular filters



Applications – biometrics

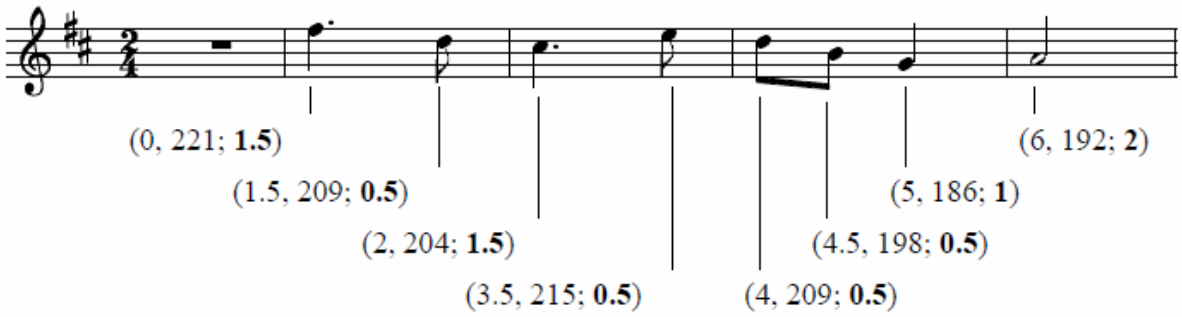
- talker identification
  - talker's utterance waveform is converted into MFCCs
  - MFCCs quantized + histogram produced
  - searching a database of talkers' utterances by a query utterance – nearest neighbor retrieved
  - similarity
    - Kullback-Leibler divergence
      - in case of single feature vector per waveform
    - nearest neighbor distance (similar to Hausdorff distance + L2 ground dist.)
      - in case of set of vectors per waveform

Applications – music retrieval

- music retrieval modalities
  - similarity query
    - query-by-example
    - query-by-humming
  - music genres classification
- audio signal vs. melody representation

Melody retrieval

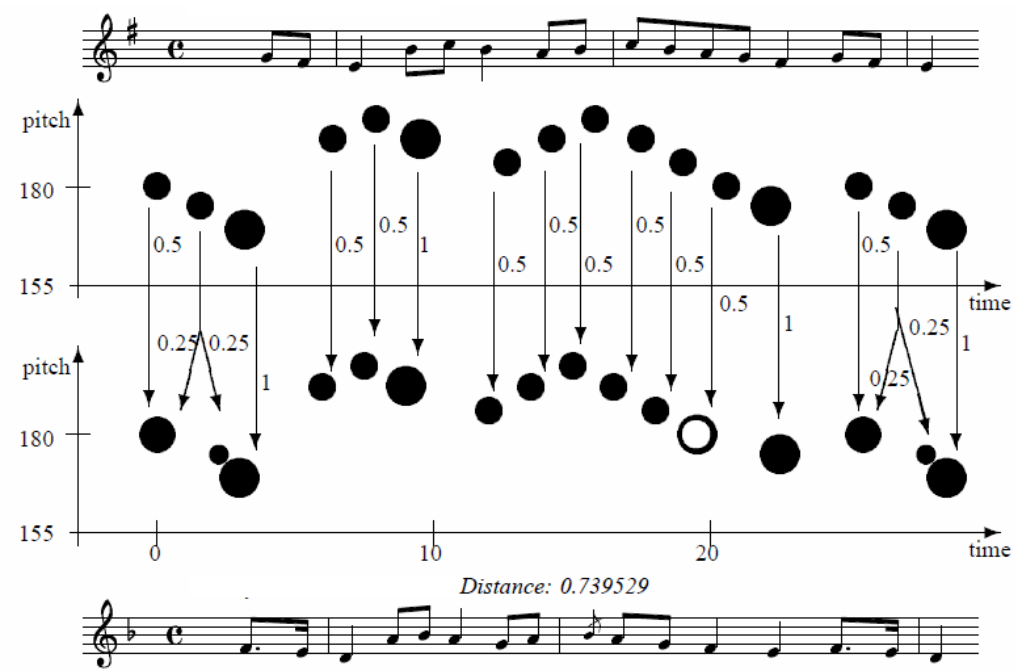
- musical score = symbolic notation of music/melody
  - does not depend on real music performance (audio waveform)
  - monophonic vs. polyphonic score
  - human-readable (score) vs. machine-readable (e.g., MIDI file)
- approach 1 – monophonic melody
  - notes (musical symbols) transformed into set of 2D points
  - point location determined by the pitch and timing, while the weight of point was deteremined by a musical property



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

Melody retrieval

- melody dissimilarity
  - Earth Mover's distance
  - Proportional Transportation distance



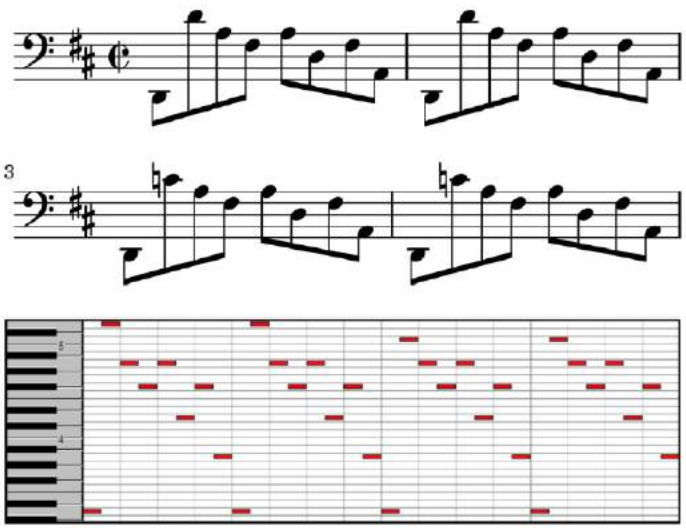
Melody retrieval

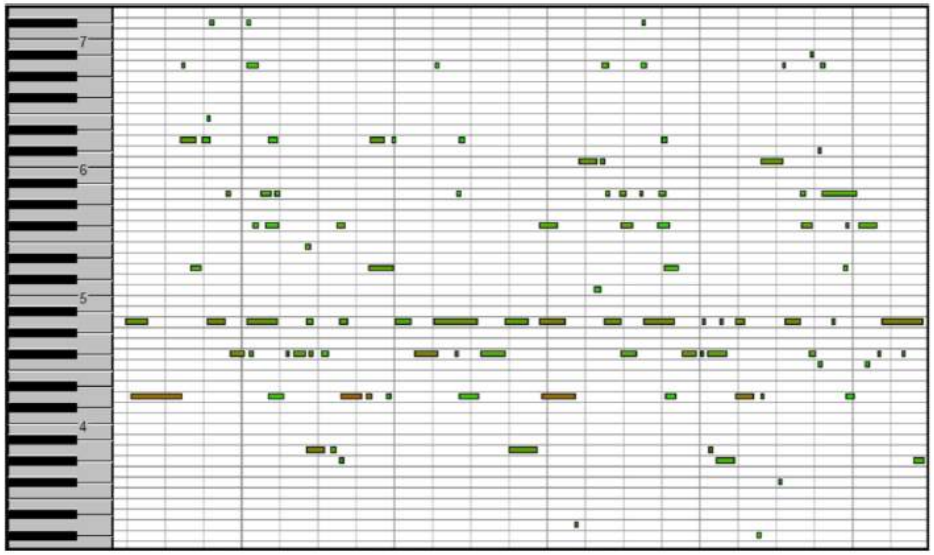
- approach 2 – monophonic melody
  - similarity
    - time-warped longest common subsequence similarity (TWLCS)
      - combines advantages of DTW and LCS – robustness w.r.t. variations in speed and rhythm (DTW) and not penalizing missing, redundant or noisy notes (LCS)
- approach 3 – polyphonic melody
  - query expected as MIDI file, while the scores in database are extracted from the respective audio waveforms
  - feature extraction produces up to 3x more notes than actually needed
  - similarity – longest common subsequence
    - beneficial due to ignoring noisy/redundant notes

Melody retrieval

- example

(images from: Suyoto et al, Effective Retrieval of Polyphonic Audio with Polyphonic Symbolic Queries, 2007)





correct representation vs. automatic transcription (many noise notes)

course:
Multimedia Retrieval

lecture 11:

Principles of multimedia indexing

prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

- motivation
- pivot-based metric lower-bounding
- metric space partitioning
- introduction to metric access methods
- indexability

Motivation

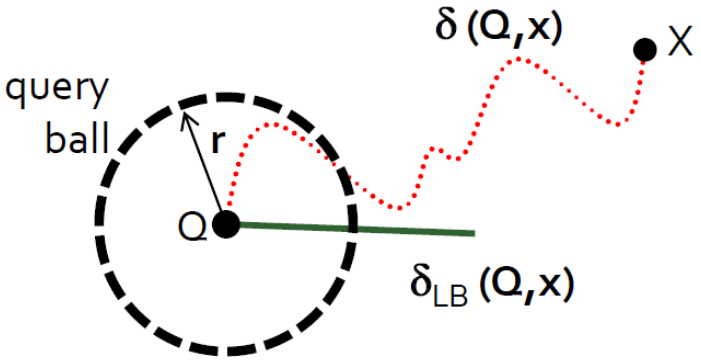
- similarity search
  - effective model (quality of similarity measure)
  - **efficient (fast) processing**
- assumption: **expensive** distances (superlinear time compl.)
  - most of the search time spent in computing distances
  - realtime reduced to the **number of distance computations**
    - platform-independent cost unit
  - I/O cost, internal cost, etc. relevant only for cheap distances

Motivation

- lower-bounding
  - general mechanism for efficient similarity search
  - using cheap lower bound instead of expensive distance
- metric space model
  - lower-bounding based on pivots and metric postulates
  - allows partitioning of the space
  - database indexing based on metric space model
    - metric access methods

General lower-bounding

- a cheap determination of **lower-bound distance** $\delta_{LB}(*,*) \leq \delta(*,*)$
- provides a mechanism how to quickly filter irrelevant objects from search
  - consider query ball **(Q, r)** and data object **x**
  - if $\delta_{LB}(Q,x) > r$, **x** is irrelevent
- **tight** lower bound needed
  - increasing probability that $\delta_{LB}(*,*) > r$
  - e.g., zero LB is useless



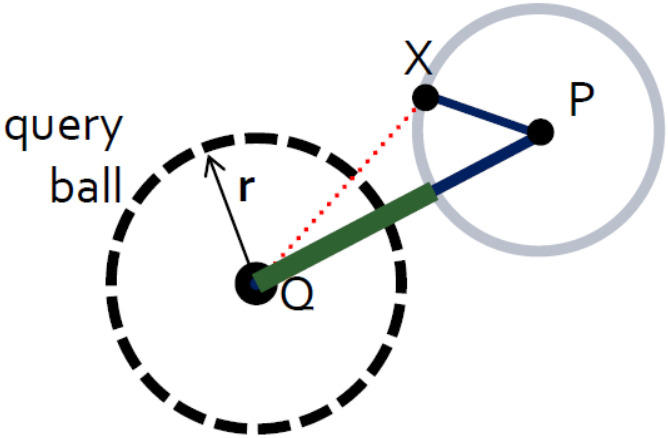
Metric postulates

- metric postulates support common assumptions on similarity
  - reflexivity – object is maximally similar to itself
  - non-negativity – two distinct objects are somehow dissimilar
  - symmetry – the direction of similarity judgement is not important
  - triangle inequality – similarity is transitive

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

Metric lower-bounding

- metric space model provides a specific mean of lower-bounding
  - **pivot objects** – static objects selected from the database
  - triangle inequality for lower bound construction (using a pivot)
  - precomputed distances to pivots
    - stored in a metric index

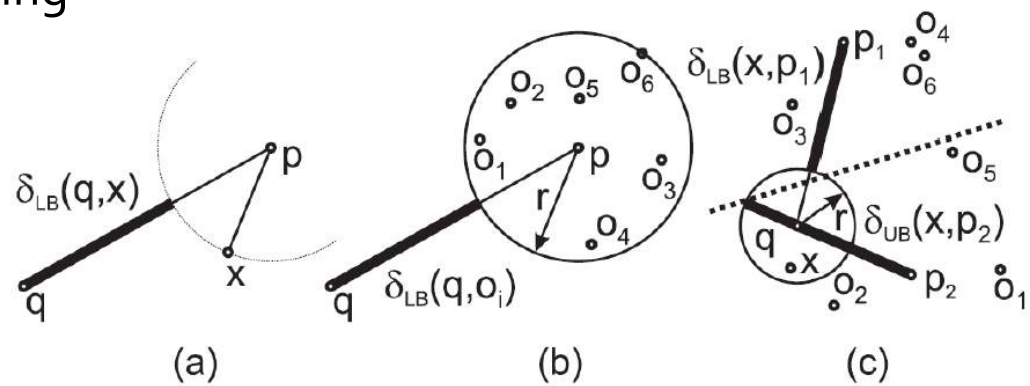


<u>The task:</u> check if **X** is inside query ball

- 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

Metric lower-bounding

- lower bound to a database object **(a)**
  - basic concept
- lower bound to a region **(b)**
  - ball-shaped regions
  - logical combinations of predicates
- combination of lower and upper bounds **(c)**
  - hyper-plane partitioning



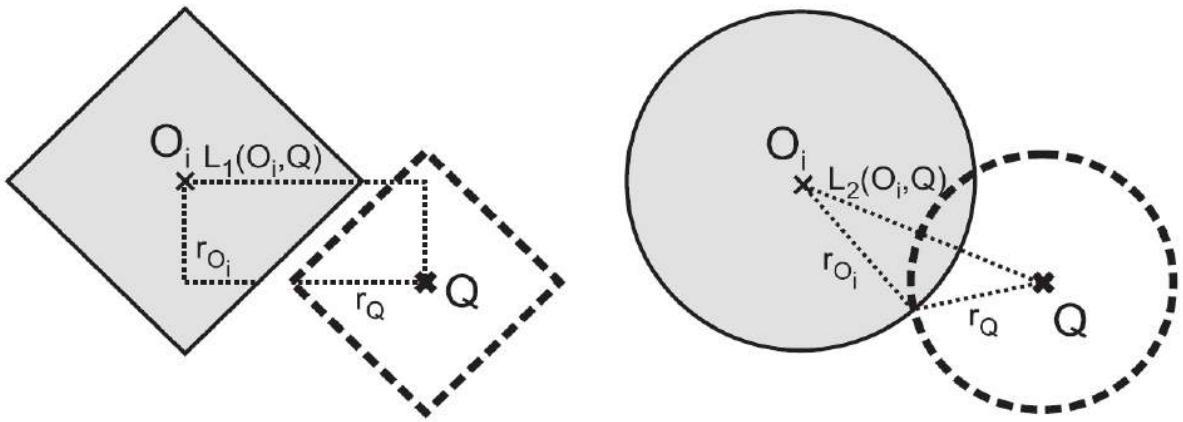
Metric space partitioning

- database organization in the metric space
  - equivalence classes – regions/partitions in the space
    - all objects share a property of particular closeness to some reference objects (pivots)
    - the search is then performed just in several partitions → **efficient search**
- region
  - information about location/distribution of objects in space
    - simple boundary
    - statistics
- testing overlap of query region and a data region
  - disjunction guarantees data inside the region are irrelevant to the query
  - overlap does not guarantee relevancy of data to the query (just possibility)
  - must be cheap operation (as cheap or cheaper as LB construction)

Ball-shaped regions

- for two balls (O_i, r_{Oi}) and (Q, r_Q) it holds:
  if $\delta(O_i, Q) > r_{Oi} + r_Q$ then balls do not overlap (and vice versa)
  Proof: let O_j is an object inside (O_i, r_{Oi})

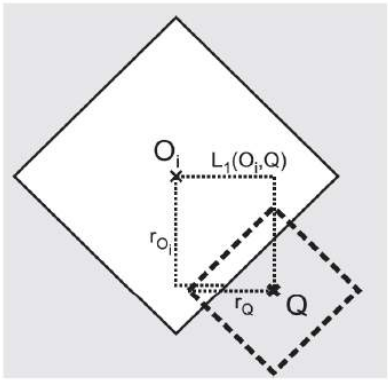
$$
\begin{aligned}
d(O_j, O_i) + d(O_j, Q) &\geq d(O_i, Q) && \text{(by triangular inequality)} \\
d(O_j, Q) &\geq d(O_i, Q) - d(O_j, O_i) \\
d(O_j, Q) &\geq d(O_i, Q) - r_{O_i} && \text{(covering radius upper bound)} \\
d(O_j, Q) &> r_Q && \text{(by hypothesis)}
\end{aligned}
$$

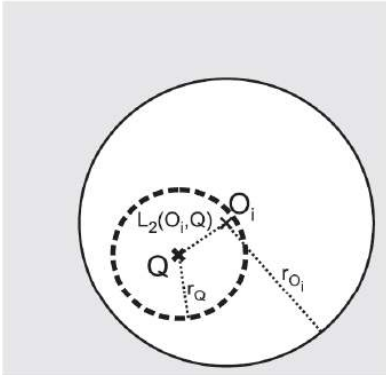


Ball-shaped regions

- for ball (Q, r_Q) and ball complement $\neg(O_i, r_{Oi})$,
  i.e. the whole space with hole in (O_i, r_{Oi}), it holds:
  If $\delta(O_i, Q) + r_Q < r_{Oi}$ then ball and the complement do not overlap (and vice versa)
  Proof: let O_j is an object inside $\neg(O_i, r_{Oi})$, then

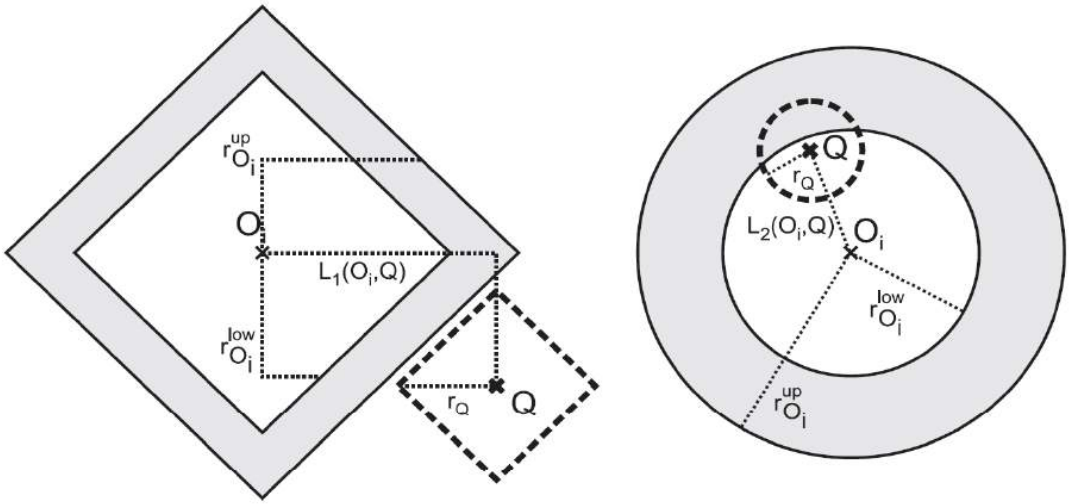
$$
\begin{aligned}
d(O_i, Q) + r_Q &< d(O_j, O_i) && \text{(by hypothesis)} \\
r_Q &< d(O_j, O_i) - d(O_i, Q) \\
d(O_j, Q) &\geq d(O_j, O_i) - d(O_i, Q) && \text{(by triangular inequality)} \\
d(O_j, Q) &> r_Q
\end{aligned}
$$





Ball-shaped regions

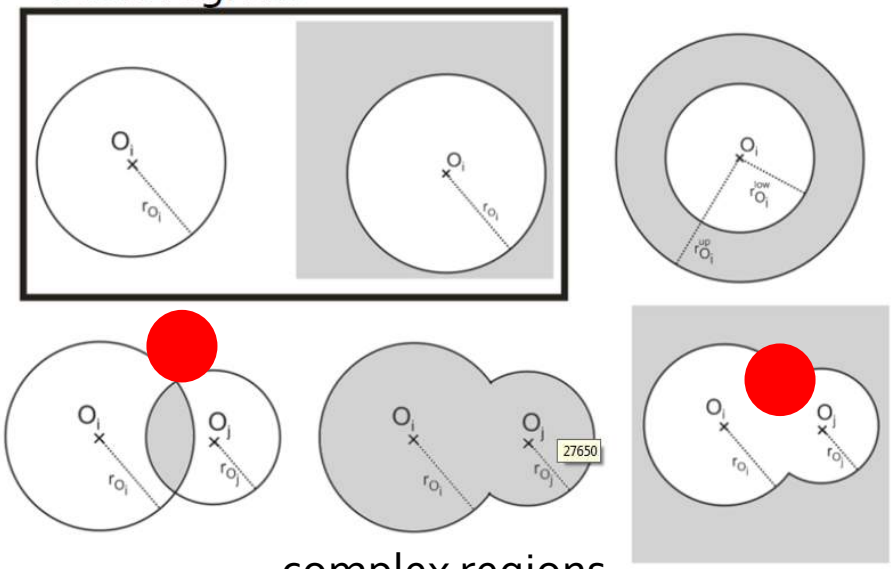
- for ring(O_i, r_{Oi}^{Up}, r_{Oi}^{Low}) and ball(Q, r_Q), it holds:
  if $\delta(O_i, Q) + r_Q < r_{Oi}^{Low} \vee \delta(O_i, Q) > r_Q + r_{Oi}^{Up}$ then
  they are not overlapping (and vice versa)
  - in other words: a query ball is entire inside the small ball or entire outside the big ball
    Proof: follows from the previous two lemmas



Ball-shaped regions

- the basic regions can be combined to form more complex regions
  - however, only in terms of logic, not geometry!
  - in metric space, there is no shape, volume, geometry
  - just combinations of predicates
    - "inflated geometry"

basic regions



complex regions

overlap of both balls, not overlapping the intersection

overlap of both holes, overlapping also the complement

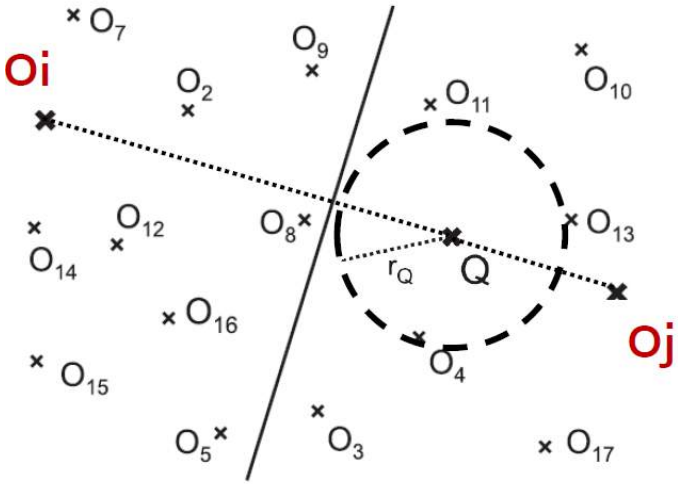
Hyperplane partitioning

- for two regions determined by a hyperplane between O_i,O_j, it holds:

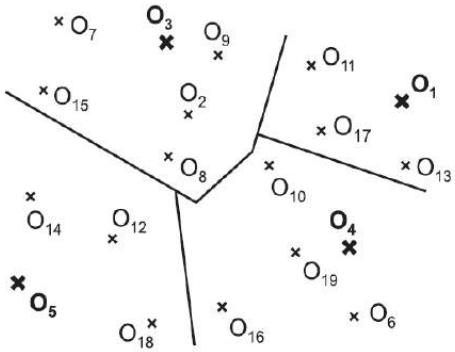
  if $\delta(O_i, Q) - r_Q > \delta(O_j, Q) + r_Q$ then first (O_i) region does not overlap query

  if $\delta(O_j, Q) - r_Q > \delta(O_i, Q) + r_Q$ then second (O_j) region does not overlap query

  - simply: if lower-bound distance to query from one pivot is greater than upper-bound distance to query from the second pivot, the first region does not overlap

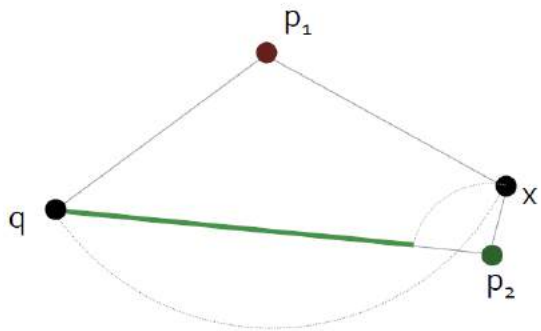


- by combination of the conditions, we can establish multiple partitions (Voronoi partitioning)



Pivots

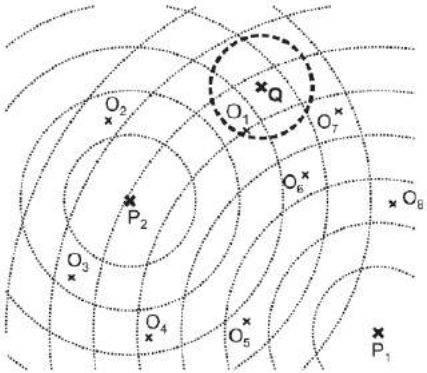
- pivots are important for the tightness of lower bounds
  - quality – a pivot should be either close to database object or to the query object
    - LB = $|\delta(q,p) - \delta(p,x)|$
    - p_1 is bad, LB(q,x) = 0
    - p_2 is good, LB(q,x) is high

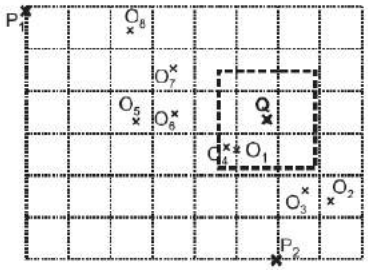


  - quantity – the more pivots, the greater probability of having a good one for a particular pair query object-database object
    - p_2 increased the LB obtained by p_1
  - quality + quantity – because one good pivot is enough, the pivots should not be close to each other (duplicate effect)
    - does not necessarily mean they should very far from each other

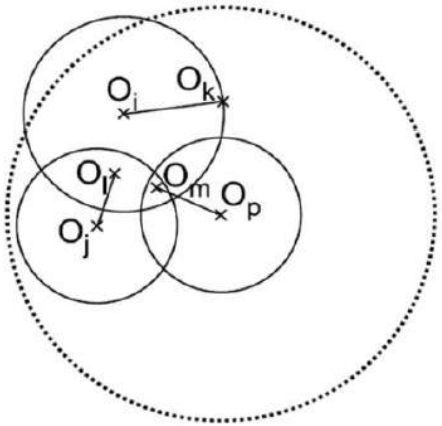
Pivots

- global pivots
  - static set, objects selected a-priori from the database
  - used uniformly – every lower bound constructed using the same set of pivots
    - mapping to a global pivot space





- local pivots
  - dynamic, objects selected during indexing
  - used specifically – every lower bound constructed using different set of objects
    - centers of clusters



Pivot selection techniques (global pivots)

- random
  - cheapest, just take random objects from a database
  - works good, especially for large number of pivots (avg. effect)
- random groups
  - randomly select some candidate sets of pivots (groups) from the database
  - for each group compute the mean of distance distribution in database projected into the pivot space (or just sum of distances among pivots)
  - select group with the maximal mean/sum of distances
  - a bit better than random, more expensive

Pivot selection techniques (global pivots)

- incremental
  - select the first pivot at random
  - from a random group of objects of size N, select the one that maximizes the mean of the distances among the database objects in the pivot space (including the candidate object as a possible new pivot), add this object to the set of pivots
  - repeat this until the desired number of pivots is reached
  - better than random, but very expensive
- SSS(sparse spatial selection)
  - optimal number of pivots is determined automatically
  - every database object is checked for a pivot
  - only the "distinct" ones are selected

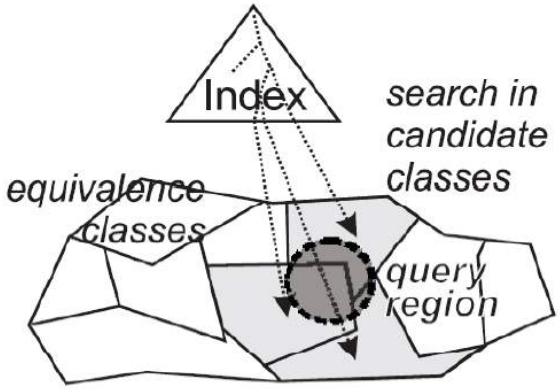
Algorithm 1: Sparse Spatial Selection

Input: $(\mathbb{X}, d)$, $\mathbb{U}$, α, M
Output: $Pivots$

```
Pivots ← {x_1};
foreach x_i ∈ U do
    if ∀ p ∈ Pivots, d(x_i, p) ≥ Mα then
        Pivots ← Pivots ∪ {x_i};
return Pivots
```

Metric access methods (MAMs)

- MAM = algorithms and/or data structure providing efficient (fast) similarity search under the metric space model
- mostly index-based
  - metric index = data structure that maintains precomputed distances (or aggregates on distances) related to a part of the database
    - persistent vs. main-memory
  - indexing needed = preprocessing step prior to querying
- various aspects
  - hierarchical vs. flat structure
  - static vs. dynamic
  - parallel vs. serial, centralized vs. distributed
  - exact vs. approximate
  - designed for continuous vs. discrete metrics

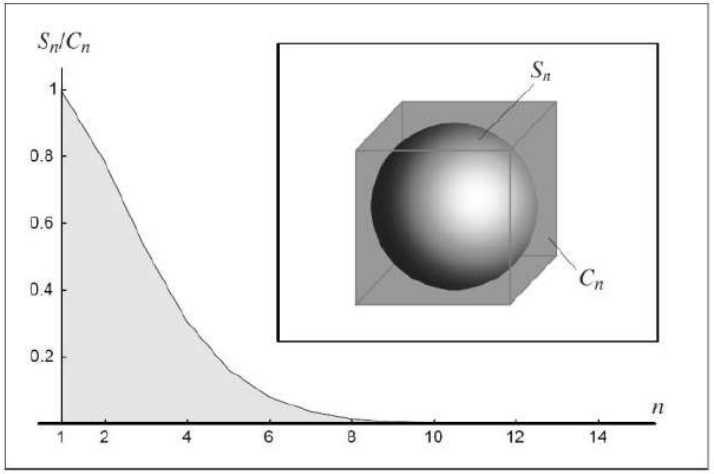


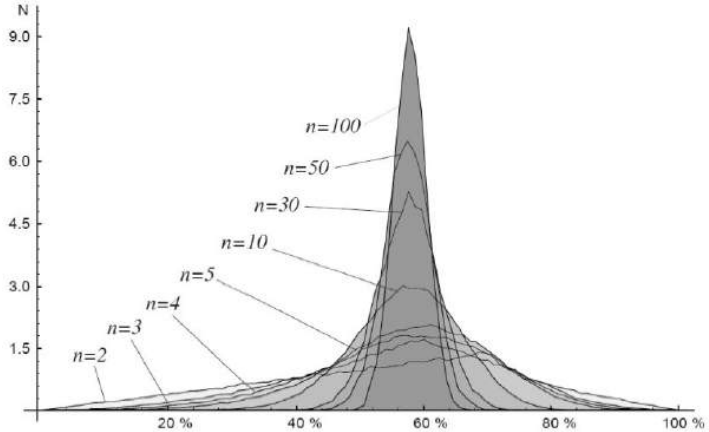
Indexability

- metric postulates alone do not ensure efficient indexing/search
  - also data distribution in space is very important
- indexability = ability of structuring the database into distinct classes, such as clusters
  - in metric indexing this means uneven distribution of distances
- indicators
  - intrinsic dimensionality
  - ball-overlap factor
  - others (fractal dimension)

Curse of dimensionality

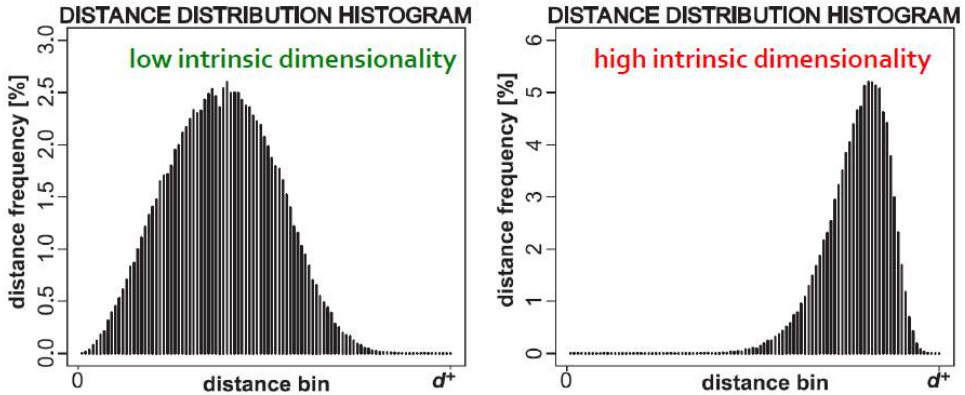
- phenomenon observed
  in high-dimensional vector spaces
  - negative impact for database indexing
  - data of dimensionality > 10
    not effectively indexable
- some effects
  - volume (and surface) of space and objects inside grows exponentially
    - consider ball inside cube, touching all cube sides
  - points in the space tend to be equidistant
    - consider distance distribution
    - whatever metric
    - distance becomes meaningless
      - nearest neighbor is almost as far as the furthest neighbor
  - points tend to gather near the axes
    - empty space (dead space)





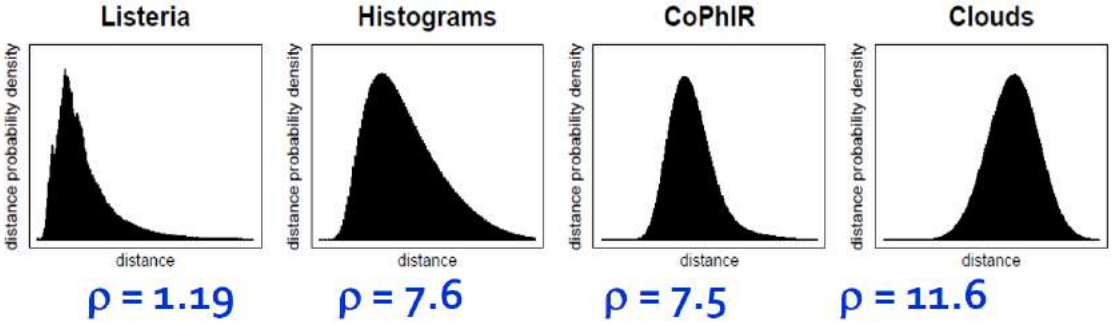
Intrinsic dimensionality

- **intrinsic dimensionality** $\rho(S,\delta) = \mu^2 / 2\sigma^2$
  - μ is **mean** and σ^2 is **variance** of discrete distance distribution in dataset **S**
- low **ρ** (e.g., below 10) means the dataset is **well-structured** – i.e. there exist tight clusters of objects
- high **ρ** means the dataset is **poorly** structured – i.e. objects are almost equally distant
  - kind of dimensionality curse
  - intrinsically high-dimensional datasets are hard to organize
  - querying becomes inefficient (sequential scan)
  - MAM-independent property



Intrinsic dimensionality

- example, 4 databases
  - Listeria–DNA sequences of Listeria monocytogenes of lengths 200–7,000
    - edit distance
  - Histograms –imagesfromFlickr.com, 512-dimensional histograms
    - quadratic form distance
  - CoPhIR–imagesfromFlickr.com, 282-dimensional descriptors (5 MPEG7 features)
    - Euclidean distance
  - Clouds –clouds (sets) of 60 6-dimensional points
    - Hausdorff distance (Euclidean ground distance on the points)



Ball-overlap factor

- the intrinsic dimensionality
  - applicable also on non-metric distances (to some extent)
  - but spacelessmeasure
    - the number is not defined precisely, just high or low
- ball-overlap factor (BOF)
  - applicable only to metric distances
  - the proportion of overlapping balls in space, where a ball is defined as centered in a database object of radius to its (kth) nearest neighbor
  - bounded measure
    - 0 – 1, zero means no overlap, one means every ball overlaps every other ball
  - utilizes an abstraction of MAM using ball regions – should be more appropriate as the indexability indicator for MAMs

$$\mathrm{BOF}_k(\mathbb{S}, \delta) = \frac{2}{|\mathbb{S}^*| * (|\mathbb{S}^*| - 1)} \sum_{\forall o_i, o_j \in \mathbb{S}^*, i > j} sgn(|(o_i, \delta(o_i, kNN(o_i))) \bar{\cap} (o_j, \delta(o_j, kNN(o_j)))|$$

course:
Multimedia Retrieval

lecture 12:

Metric access methods

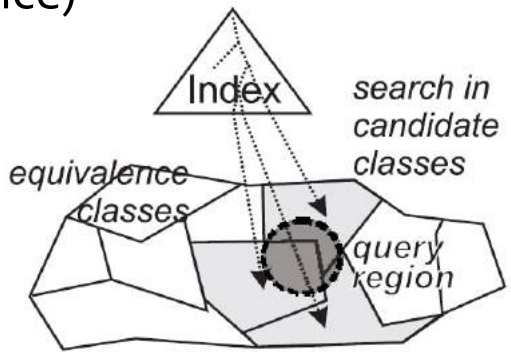
prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Lecture overview

- metric access methods (MAMs)
  - pivot tables (distance matrix methods)
    - + spatial access methods
  - tree-based indexes
  - hashed indexes
  - index-free MAM
- performance cost measures
  - realtime, distance computations, I/O, etc.

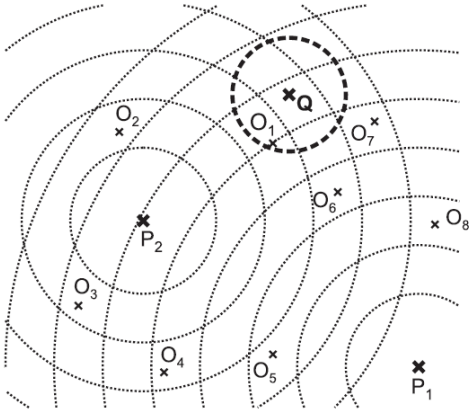
Metric access methods – motivation

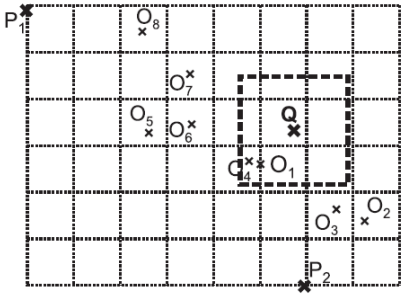
- MAMs objective: **fast similarity search in database S**
  - various types of cost measuring the "fast" (realtime, number of distance comp., I/O, internal, etc.)
- the means: **metric indexing**
  - based on lower-bounding using pivots
  - metric postulates needed
    (the similarity function must be metric distance)
- various structural designs
  - flat indexes (pivot tables)
  - hierarchical indexes (trees)
  - hashed indexes
  - index-free methods



Pivot tables

- class of indexes using mapping of data to the pivot space
  - each data object **O** maps to a vector **o**,
  - i.e., we get distance matrix
- originally AESA/LAESA
  (Linear) Approximating and Eliminating Search Algorithm
- AESA
  - every data object is a pivot, i.e., we have square matrix of size **O(|S|²)**
  - empirical average **O(1)** time for nearest neighbor search, expensive construction
- LAESA
  - only some **k** data objects selected as pivots, i.e., matrix size **O(|S|)**
- querying by L_∞ metric in the pivot space

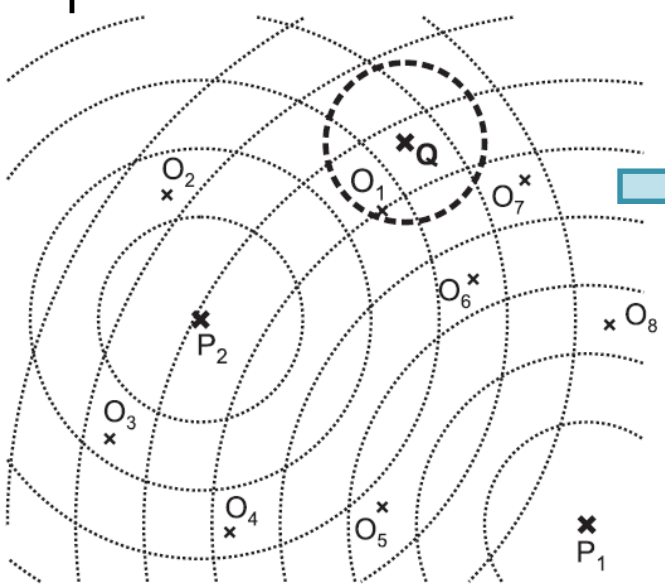


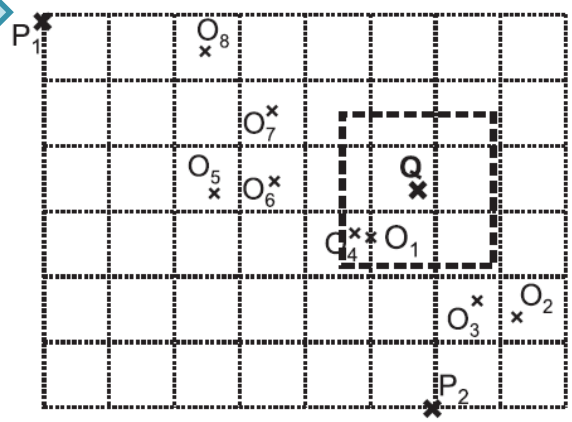


LAESA range query

- simple 2 phases
  - sequential processing 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





LAESA nearest neighbor query

- one-phase (as in AESA, see next slide)
  - suitable for large number of pivots
- two-phase
  1. the entire query vector **q** is determined from the query object
  2. the database objects $\mathbf{O}_i \in \mathbf{S}$ are sorted asc. w.r.t $\mathbf{L}_\infty(\mathbf{q}, \mathbf{o}_i)$
  3. in this order the $\delta(\mathbf{Q}, \mathbf{O}_i)$ is being computed, updating the NN candidate $\mathbf{O}_{nn}$
     - if $\delta(\mathbf{Q}, \mathbf{O}_{nn}) < \mathbf{L}_\infty(\mathbf{q}, \mathbf{o}_i)$, the filtering terminates (there is no better candidate than $\mathbf{O}_{nn}$)

AESA nearest neighbor query

- **problem:** cannot compute the entire query vector (the same as seq. search)
- **the idea:** increasing the lower bounds + filtering by mapping the query vector dimension by dimension

The algorithm:

1) (**initialization**) Let's have a NN query **Q** and a set of pivots **S'** = **S**.
   Let the current distance to $\mathbf{O_{nn}}$ is $\mathbf{d_{min}} = \infty$.

2) Select randomly a pivot **P** $\in$ **S'**

3) Compute $\delta(\mathbf{Q}, \mathbf{P})$, i.e., some coordinate of the query vector **q**.
   If $\mathbf{d_{min}} > \delta(\mathbf{Q}, \mathbf{P})$, then $\mathbf{d_{min}} := \delta(\mathbf{Q}, \mathbf{O_{nn}})$ and $\mathbf{O_{nn}} := \mathbf{P}$. $\mathbf{S'} := \mathbf{S'} - \{\mathbf{P}\}$.

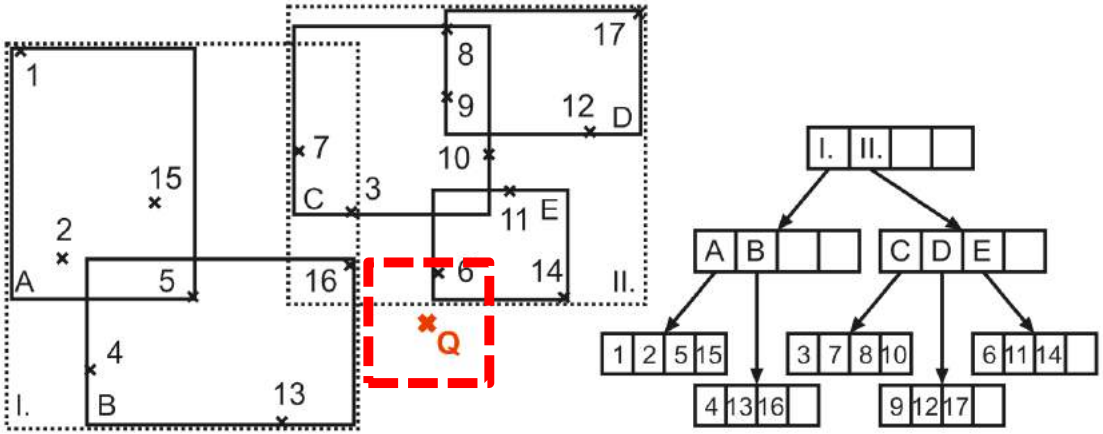
4) (**elimination**) Those objects $\mathbf{O_i}$ are filtered from **S**, for which $\mathbf{L_\infty}(\mathbf{q}, \mathbf{o_i}) > \mathbf{d_{min}}$.
   We consider only coordinates of $\mathbf{o_i}$ already computed also for **q**.

5) (**approximation**) If **S'** is empty, the NN was already found (it is $\mathbf{O_{nn}}$) and the algorithm terminates. Otherwise, we proceed with pivot **P** $\in$ **S'**, whose vector **p** has the lowest $\mathbf{L_\infty}(\mathbf{q}, \mathbf{p})$.

6) Repeat from step 3, until all objects in S are removed (filtered). Hence, only one NN candidate remains – $\mathbf{O_{nn}}$ – the final nearest neighbor.

Other methods based on pivot tables

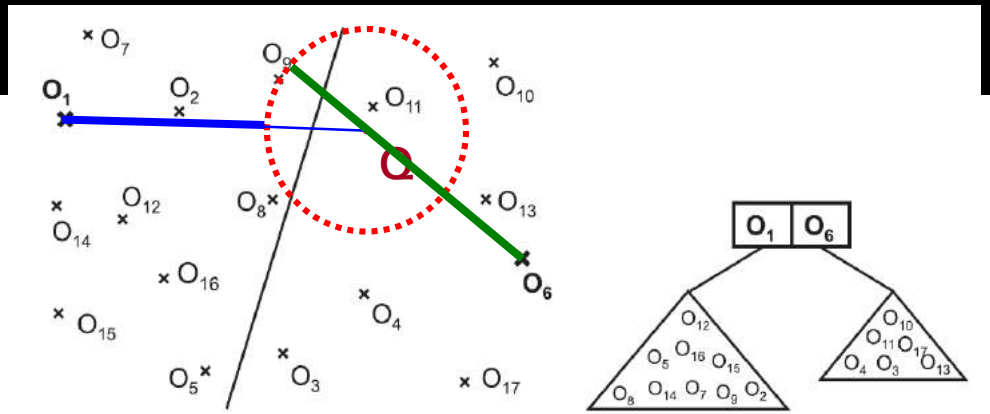
- TLAESA
  - optimization of I/O cost using a tree structure similar to gh-tree
- ROAESA
  - AESA + heuristics for reduction of the distance matrix traversal
- Spaghettis
  - optimization of I/O cost using sorted arrays of pairs object-pivot (i.e., indexing separately one-dimensional pivot spaces)
- OmniFamily
  - utilization of R-tree and other SAM



Tree-based indexes

- gh-tree, GNAT
  - hierarchy of regions defined by generalized hyperplane partitioning
- vp-tree, mpv-tree
  - hierarchy of spherical rings (+ intersections)
- M-tree, PM-tree
  - hierarchy of balls (+ intersections with rings)

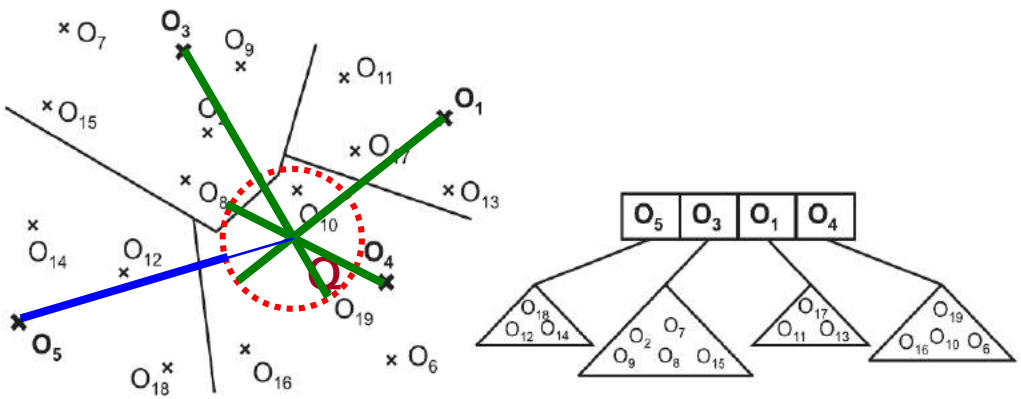
gh-tree



- binary tree
- static construction
  - select two pivots
  - partition the database in two groups/partitions
  - recursively repeat for the two partitions
- range query processing
  - test overlap of both partitions
  - recursively proceed down the hierarchy, accessing the overlapping partitions
  - filtering of left partition (subtree) in the example:
    - if the **closest possible object** inside the query (w.r.t $\mathbf{O_1}$) is more distant than the **furthest possible object** inside the query (w.r.t. $\mathbf{O_6}$), then the left subtree can be filtered out
    - in other words, if the lower bound from $\mathbf{Q_1}$ is greater than upper bound from $\mathbf{Q_6}$

GNAT

- generalization of gh-tree into n-ary tree
  - n pivots (not just two)
- some other extensions
  - distance ranges
- range query processing
  - test overlap of all partitions

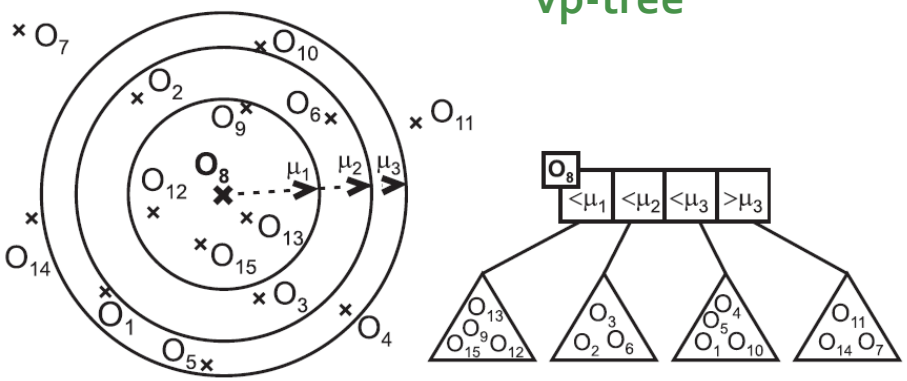


  - recursively proceed down the hierarchy in the overlapping partitions
  - filtering of partition/subtree belonging to $\mathbf{O_5}$ (in the example):
    - if the **closest possible object** inside the query (w.r.t $\mathbf{O_5}$) is more distant than the **furthest possible object** inside the query (w.r.t. any of $\mathbf{O_1}$, $\mathbf{O_3}$, $\mathbf{O_4}$), then the subtree can be filtered out

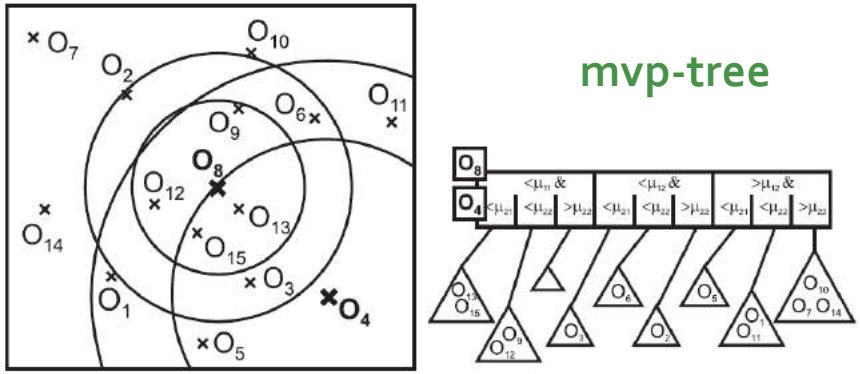
vp-tree, mvp-tree

- (multiple) vantage-point tree
  - vantage-point = pivot
  - n-ary tree
  - vp-tree can be balanced
    - mvp-tree cannot
- static construction
  - pivot(s) selected,
    number of ball rings selected
  - radii μ_i of the rings determined
  - objects partitioned into the rings
    (ring intersections
    in case of mvp-tree)
  - recursive repeating the partitioning

vp-tree

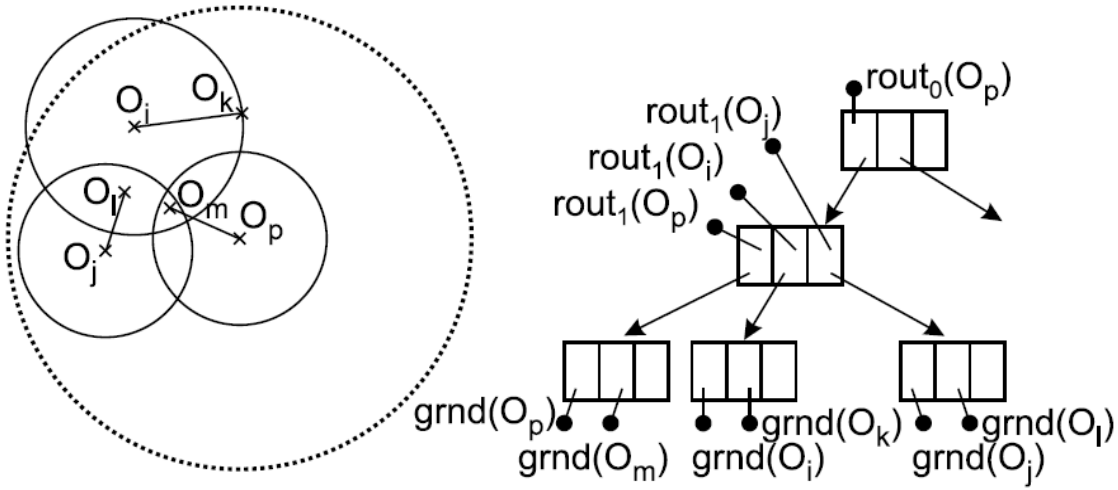


mvp-tree



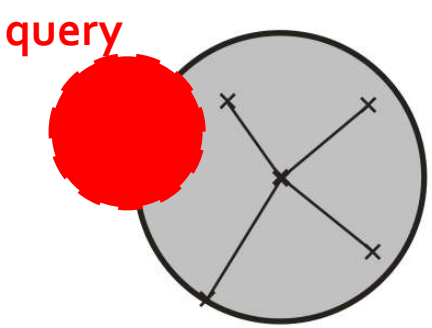
M-tree

- inspired by R-tree – modification into metric spaces
  - hierarchy of nested balls; balanced, paged, dynamic index
  - suitable for secondary storage
  - compact hierarchy is necessary, ball overlaps lead to inefficient query processing
  - many descendants – e.g., M*-tree, PM-tree, Slim-tree, etc.
- query processing
  - traversing the hierarchy, accessing just the overlapping nodes/subtrees

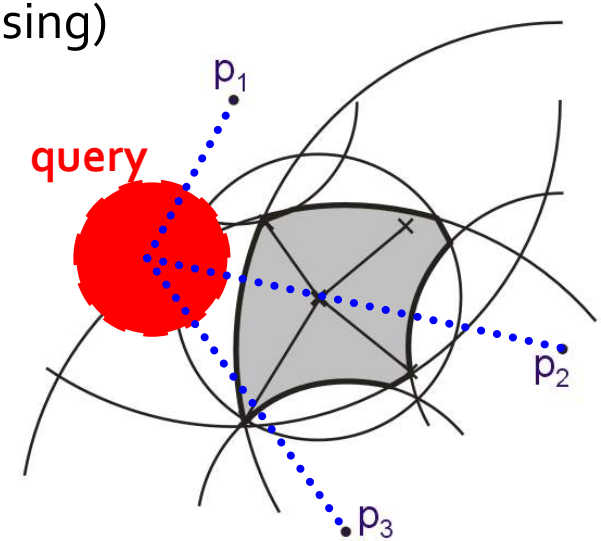


PM-tree

- PM-tree = M-tree additionally using a set of ***p*** global pivots
- each ball region is further reduced by ***p*** rings, centered in the pivots
- smaller volume of regions
  - better filtering (faster query processing)

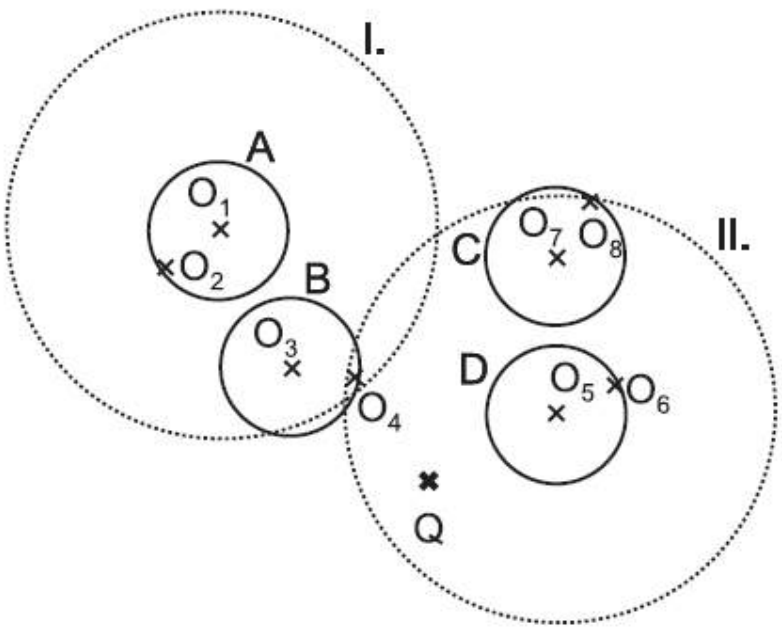


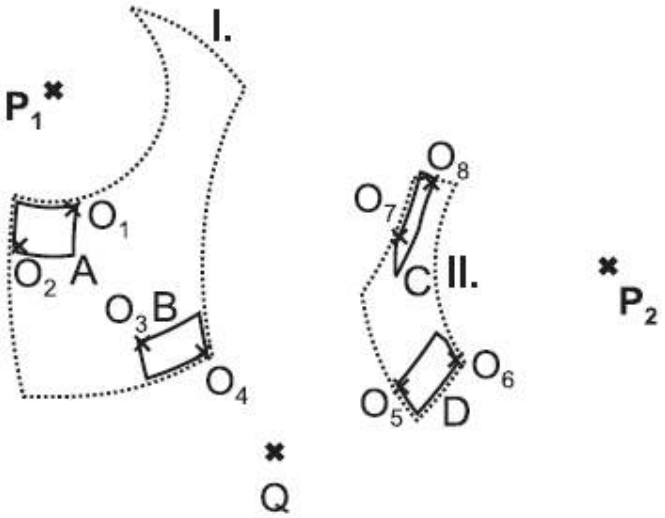
M-tree region



PM-tree region

M-tree vs. PM-tree hierarchy

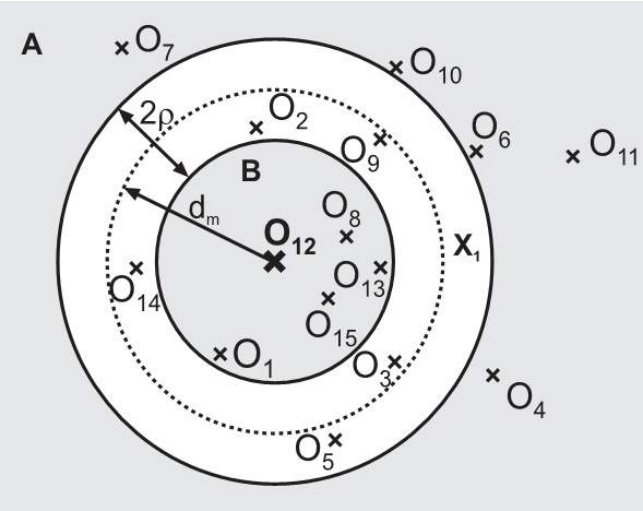




D-index

- D-index = hash-based index
- based on hashing functions *bps*$^{1,\rho,j}$ (ball-partitioning split functions), where $\mathbf{P_j}$ is pivot, $\mathbf{d_m}$ is median distance from $\mathbf{P_j}$ to the objects $\mathbf{O_i}$, and **ρ** is a splitting parameter
- the function assigns to an object O_i:
  - 2 , if it is inside the ring (P_j, d_m-ρ, d_m+ρ)
  - 1, if it is outside the greater ball (P_j, d_m+ρ)
  - 0, if it is inside the smaller ball (P_j, d_m-ρ)

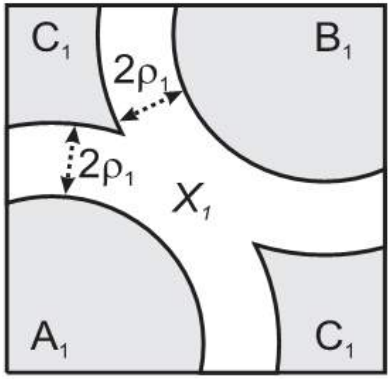
$$bps^{1,\rho,j}(O_i) = \begin{cases} 0 & \text{if } d(O_i, P_j) \leq d_m - \rho \\ 1 & \text{if } d(O_i, P_j) > d_m + \rho \\ 2 & \text{otherwise} \end{cases}$$



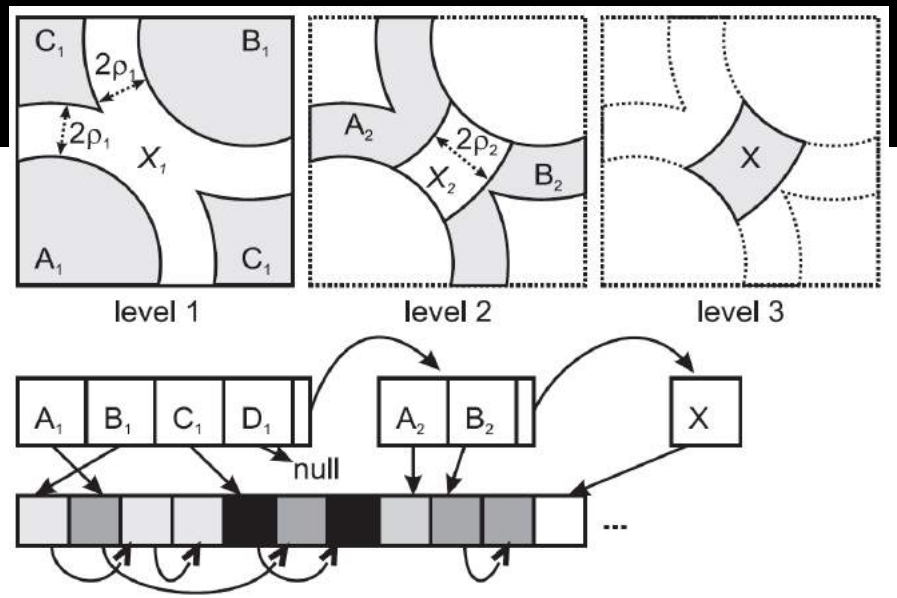
D-index

- the $bps^{1,\rho,j}$ functions can be combined
- if there **is no** 2 in the hash of the combined function, the string 0, 1 is a binary code of a partition (up to 2^n)
- if there **is a 2** in the hash code, the hashed object belongs to so-called exclusion set

- the motivation is to keep the partitions separated by the region of exclusion set
- the exclusion set is hashed again (based on different $bps^{1,\rho,j}$ functions), recursively, until it gets sufficiently small



D-index



- D-index structure
  - hashed regions are organized in buckets on the disk
  - the rehashing of exlusion set defines another level of D-index
- advantages
  - for small query radius (**r < ρ**), at most one bucket is accessed at each level
  - if the whole query lies inside a region, only that bucket is accessed
  - if the whole query lies inside the exclusion set, no bucket at that level is accessed and the query processing proceeds at the next level
- disadvantages
  - selection of the parameters ρ, d_m is difficult and/or D-index is not balanced
  - the ρ parameter should be very small, in order to not become everything into the exclusion set (i.e., suitable only for small and point queries)

Index-free MAM – D-file

- D-file = just the original database using sequential scan, BUT
- it uses D-cache
  - a memory-resident structure that maintains the distances computed during previous queries
  - **provides lower bounds** of requested distances that can be used to filter some of the database objects when querying
  - **O(1)** complexity for a lower bound retrieval

- no preprocessing step (indexing) needed to start querying the database
  - index-free MAM

Index-free MAM – D-file

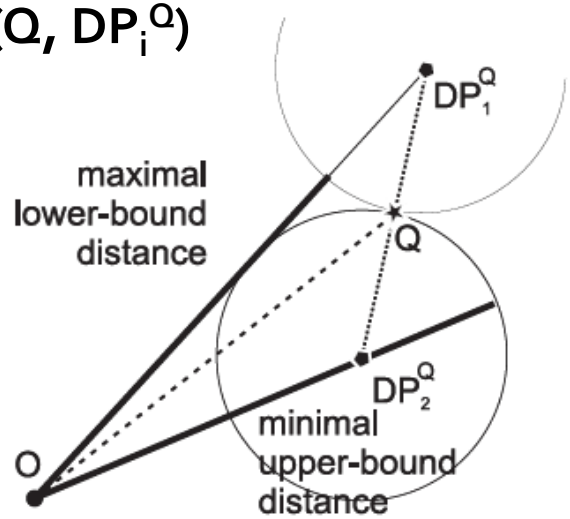
- every time a D-file computes a distance **δ(*,*)**, it is stored into D-cache
- the D-cache could be viewed as a sparse matrix, where queries denote rows, database object denote columns, and a cell contains a value of **δ(Q,O)**

$$D = \begin{array}{c} \\ Q_1 \\ Q_2 \\ Q_3 \\ \dots \\ Q_m \end{array} \begin{array}{c} \begin{array}{ccccc} O_1 & O_2 & O_3 & \dots & O_n \end{array} \\ \begin{pmatrix} & d_{12} & d_{13} & \dots & \\ d_{21} & & & \dots & d_{2n} \\ & & & \dots & \\ \dots & \dots & \dots & \dots & \dots \\ d_{m1} & & d_{m3} & \dots & \end{pmatrix} \end{array}$$

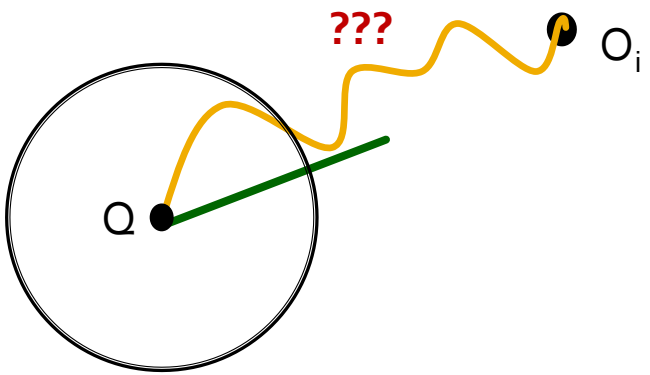
Index-free MAM – D-file

- D-cache has two functionalities
  - it allows to retrieve the exact distance **δ(Q,O)**, if it is there
  - the main functionality: it provides *tight lower bound* to δ(Q,O)

- how to obtain a lower bound?
  - prior to a new query Q, determine some old queries DP_i^Q (acting as **dynamic pivots**) and compute the distances $\boldsymbol{\delta(Q, DP_i^Q)}$
  - when a lower bound to **δ(Q,O)** is required, search for available distances $\boldsymbol{\delta(Q, DP_i^Q)}$ in the D-cache and obtain the $\boldsymbol{max(|\delta(DP_i^Q, O) - \delta(Q, DP_i^Q)|)}$; that is our tight lower bound distance



Index-free MAM – D-file



set **RangeQuery**(Q, r_Q) {
  Dcache.**StartQueryProcessing**(Q)
  for each O_i **in** database **do**
    if Dcache.**GetLowerBoundDistance**$(Q, O_i) \leq r_Q$ **then** // D-cache filtering
      compute $\delta(Q, O_i)$; Dcache.**AddDistance**$(Q, O_i, \delta(Q, O_i))$
      if $\delta(Q, O_i) \leq r_Q$ **then** add O_i to the query result } // basic filtering

sequential search enhanced by D-cache filtering

Performance measures

- performance of MAM in terms of
  - indexing
  - querying
- cost measures
  - number of distance computations
  - I/O cost
  - internal cost
  - realtime cost

Distance computations (DC)

- DC = number of distance **δ(*,*)** computations
- using DC as performance measure when
  - expensive distance is used
    - ≥ $O(n^2)$ and/or large descriptor size (n)
  - rather small database is used (e.g., fits in main memory)
  - other cost contributing to realtime is negligible
    - internal time/space cost, I/Os, networking, synchronization of parallel/distributed processing

I/O cost

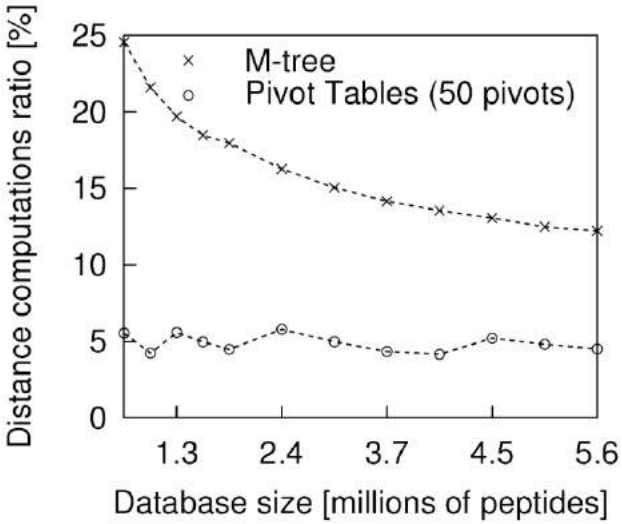
- I/O cost appropriate when
  - dominating the other cost (DC, internal, etc.)
  - the competitor MAMs share the same I/O platform
- example comparison HDD vs. SSD
  - 100 MB index, 4kB disk page, i.e., 25,600 pages
  - HDD (magnetic drive technology)
    - 1990
      - seek time + latency = **65ms**, transfer **600 kB/s** (Seagate ST 225 20MB)
      - sequential scan, **100% pages**, contiguous access = **167 sec**
      - a hierarchical MAM, **1% pages**, random access = **16.6 sec seek + 1.6 sec read = 18.2 sec** (**~9.2x speedup**)
    - 2019
      - seek time + latency = **6ms**, transfer **100 MB/s**
      - sequential scan, **100% pages**, contiguous access = **1 sec**
      - a hierarchical MAM, **1% pages**, random access = **1.5 sec seek + 0.01 sec read = 1.51 sec** (**~0.66x slowdown**)
  - SSD (solid-state drive)
    - sequential read **2150 MB/s**, random read **300k IO/s** (4kB IO, M.2 NAND NVMe), 2019
    - sequential scan, **100% pages**, contiguous access = **45 ms**
    - a hierarchical MAM, **1% pages**, random access = **0.85 ms** (**~52x speedup** vs. 100x theoretical best)
  - hierarchical indexing (random access) makes sense again in SSD (or RAM)!
    - optimized sequential scan could be a surprise in HDD

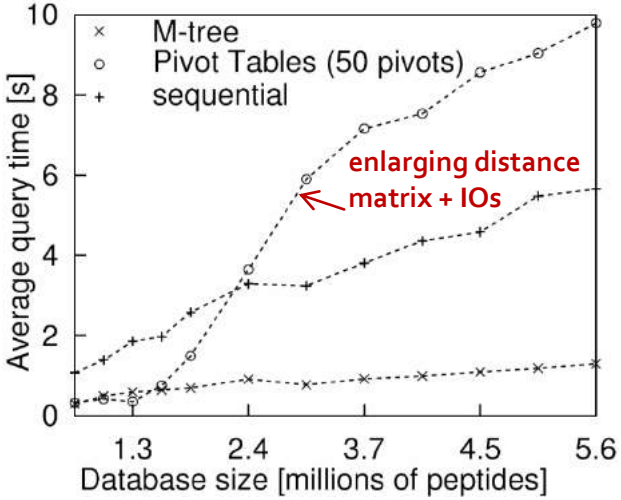
Internal cost

- more sophisticated MAM → more internal overhead
  - various auxiliary main-memory structures + processing
  - overhead data in the index + processing
- examples
  - incremental kNN processing (Hjaltason and Samet)
    - optimal in DC (w.r.t. equivalent range query), **but**
    - huge time/space overhead when managing the heap of requests
  - pivot tables (basic LAESA)
    - scanning the distance matrix
    - consider, e.g., 128 dimensional vector dataset + any L_p distance, 128 pivots → distance matrix processing means the **same or worse than simple sequential query (!)**

Realtime cost

- realtime cost (wall-clock time)
  - **cons:**
    - **optimization- and platform-dependent**
    - **harder to set up fair comparison**
  - **pros:**
    - **the only objective measure when it comes to real-world application!**
- real-world example
  - database of up to 5.6 million peptide spectra (pieces of proteins), dim ≈ 32 (intrinsic dim. ≈ 3)
  - **O(n)** variant of Hausdorff distance





course:
Multimedia Retrieval

lecture 13:

Approximate similarity search

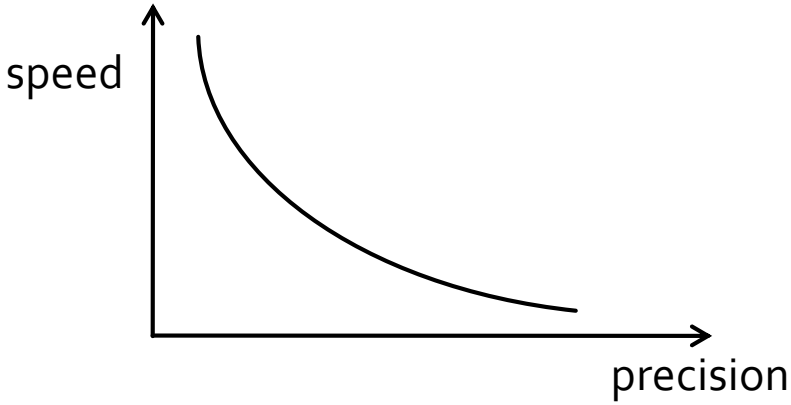
prof. RNDr. Tomáš Skopal, Ph.D.
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague

Today's lecture outline

- motivation
  - indexability
- mapping techniques
  - pivot space (simple usage of pivot tables)
  - FastMap
- approximate kNN search using M-tree
  - exact kNN algorithm
  - approximate kNN variants
- probabilistic queries using pivot tables
- permutation indexes

Motivation – exact search

- "exact" similarity search could be too slow
  - even when indexing technique is employed
- we might want to trade the retrieval precision for the speed
  - huge gain in speed usually expected (orders of magnitude) while only small loss in retrieval precision (several percent)

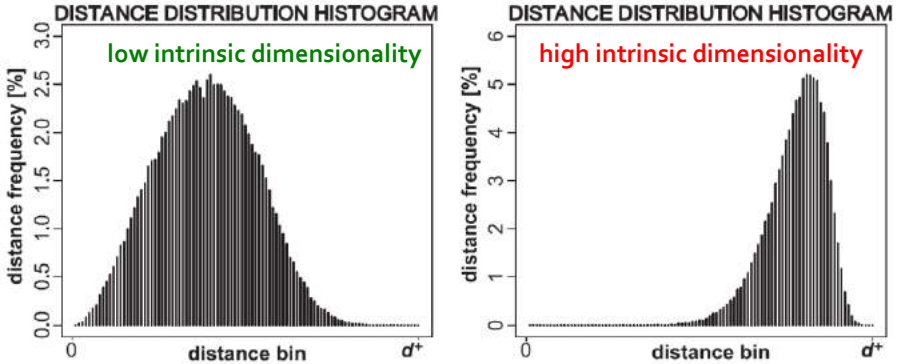


Motivation – exact search

- what is "too slow" exact search?
  - metric postulates alone do not ensure efficient indexing/search
  - also data distribution in space is very important
  - indexability = ability of structuring the database into distinct classes, such as clusters
    - **intrinsic dimensionality** – an indexability indicator

Motivation – exact search

- **intrinsic dimensionality** $\rho(S,d) = \mu^2 / 2\sigma^2$
  (μ is **mean** and σ^2 is **variance** of distance distribution in S)
- low ρ (e.g., below 10) means the dataset is **well-structured** – i.e. there exist tight clusters of objects
- high ρ means the dataset is **poorly** structured – i.e. objects are almost equally distant
  - intrinsically high-dimensional datasets are hard to organize
  - **exact search using any MAM becomes inefficient!** (the same as sequential scan)



Motivation – imperfect similarity

- moreover, similarity function often does not mimic perfectly the human perceived similarity (especially in multimedia databases)
  - even sequential search is thus subjectively approximate!
- hence, approximate search techniques only increase the subjective error of retrieval
  - i.e., just quantitative change, not a "change of category"

Motivation – approximate search

- what is "approximate"?
  - guarantee of a bound (strongest)
    - every object in query result is relevant to some extent
  - probabilistic estimates
    - every object in query result is relevant with some probability
    - i.e., on average, some percentage of objects is relevant, the rest could be completely wrong
  - no guarantee (weakest)
    - just experimental observation based on sampling, etc.

Motivation – approximate search

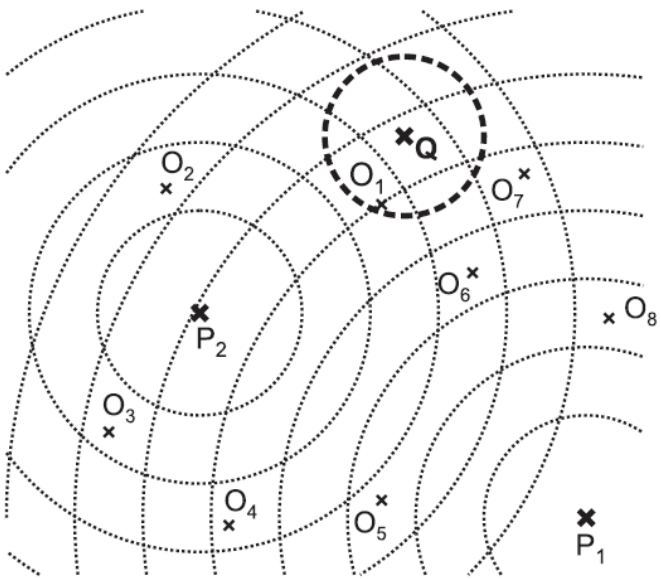
- how to speedup search?
  - realtime cost is crucial
    - indexing to reduce the distance computation (more aggressive than exact search)
    - transformation of the metric space into a simpler but only approximate target (vector) space
      - sequential search in the target space or a kind of indexing again

Mapping techniques

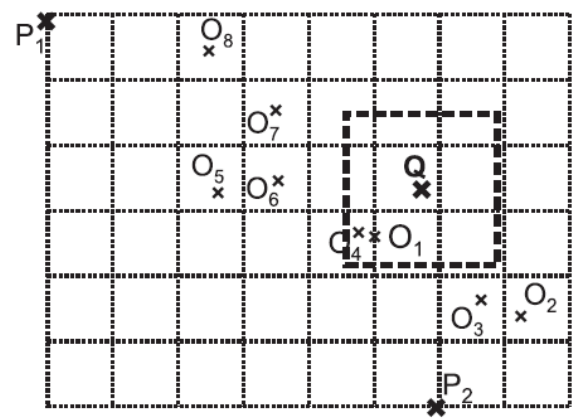
- expensive distance function δ in an arbitrary metric space could be replaced by cheap L_p distance
  - the metric space is embedded into a suitable vector space
- pre-processing of the database
  - mapping the database
    - mostly static (the embedding scheme is data-dependent)
    - some dynamic (the scheme is data-independent, e.g., simple **pivot tables**)
  - indexing the target space by metric access methods (or spatial access methods)
- many mapping techniques for metric spaces developed
  - mostly mapping to Euclidean vector space
  - metric multidimensional scaling (best but slow)
  - faster methods: **FastMap**, MetricMap, SparseMap

Mapping techniques – pivot tables

- the pivot tables (LAESA) could be used as a simple approximate method based on mapping (omitting the refinement step)
  - coordinate axis = distances to a pivot
  - using $L_\infty()$ as an approximation of $\delta()$



2NN query:
result = $\{O_1, O_4\}$



	P_1	P_2	$L_\infty(q,O_i)$
O_1	5	2.6	0.7
O_2	7.2	1.4	2.1
O_3	6.7	1.6	1.8
O_4	4.8	2.7	1
O_5	2.6	3.2	3.1
O_6	3.6	3.5	2.2
O_7	3.6	4.5	2.2
O_8	2.5	5.5	3.2

Mapping techniques – pivot tables

- will this work?
  - L_∞ in the pivot space lower-bounds δ in the metric space
  - the more (good) pivots, the tighter lower bound
    - and thus better approximation
  - limiting the need of distance $\delta(*,*)$ computations to
    - mapping the query object into the pivot space
    - optionally, re-ranking
      - e.g., 100NN query in the pivot space + re-ranking to 10NN result in the metric space
  - reasonable number of good pivots needed
  - no guarantee of retrieval precision (just empirical)

Mapping techniques – FastMap

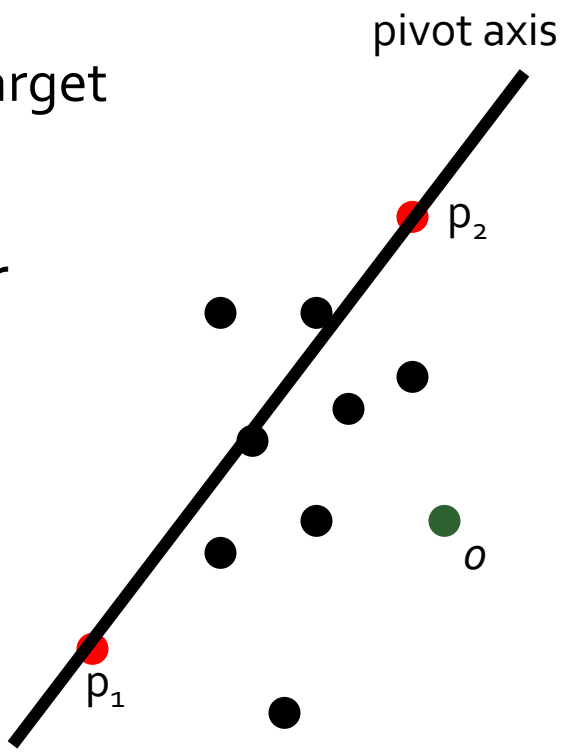
- considering an arbitrary metric space
  as an "unknown" Euclidean space
  (of user-defined dimensionality)

- the (assumed) coordinates are "recovered" by use of
  - establishing artificial coordinate system using pairs of pivots (each pair means one **pivot axis**)
  - **cosine law** and **Pythagorean theorem**,
    allowing to **project** the database objects onto the "pivot axis" (taking into account the orthogonal hyperplane formed by the already established pivot axes)

Mapping techniques – FastMap

the FastMap algorithm:

- step 0:
  - let *k* be the number of dimensions of the target Euclidean space we construct
- step 1: select first (next) best pivot pair
  - select object *o* at random
  - find p_1 being the furthest from *o*
  - find p_2 being the furthest from p_1
  - we have virtual "axis" (p_1, p_2), i.e., first (next) dimension was established



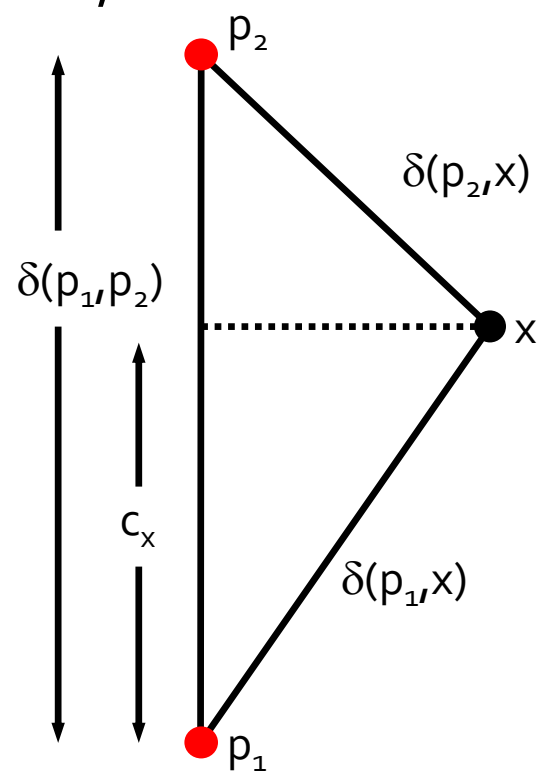
Mapping techniques – FastMap

- step 2:
  - given pivots (p_1, p_2), for any database point x, we use the **cosine law** to determine the relation of x along the axis p_1p_2

$$\delta(p_2,x)^2 = \delta(p_1,x)^2 + \delta(p_1,p_2) - 2c_x\delta(p_1,p_2)$$

$$c_x = \frac{\delta(p_1,x)^2 + \delta(p_1,p_2) - \delta(p_2,x)^2}{2\delta(p_1,p_2)}$$

c_x is the (next) coordinate of x in the target space, created by pseudo-projection

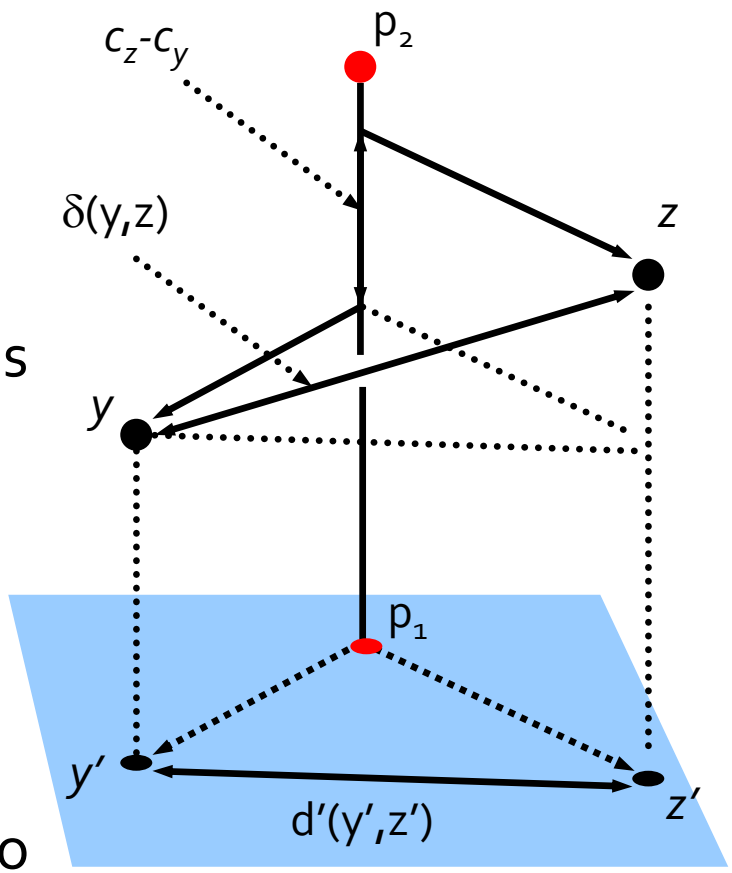


Mapping techniques – FastMap

- step 3:
  - we need to update the function $\delta(*,*)$ in order to include the already established coordinates of database objects
  - we can compute distances within the "orthogonal hyperplane" using the Pythagorean theorem

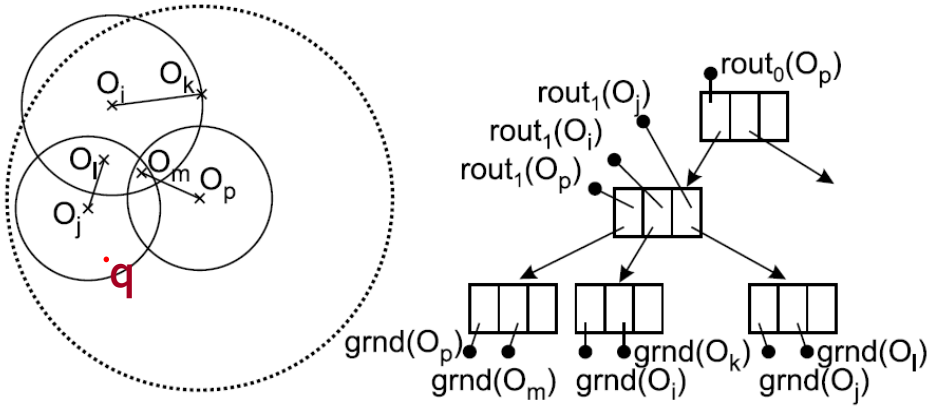
$$d'(y',z') = \sqrt{\delta(y,z)^2 - (c_z - c_y)^2}$$

  - hence, let **$\delta(*,*)$:= d'(*,*)** and goto step 1 (or terminate if all k dimensions have been established)



M-tree

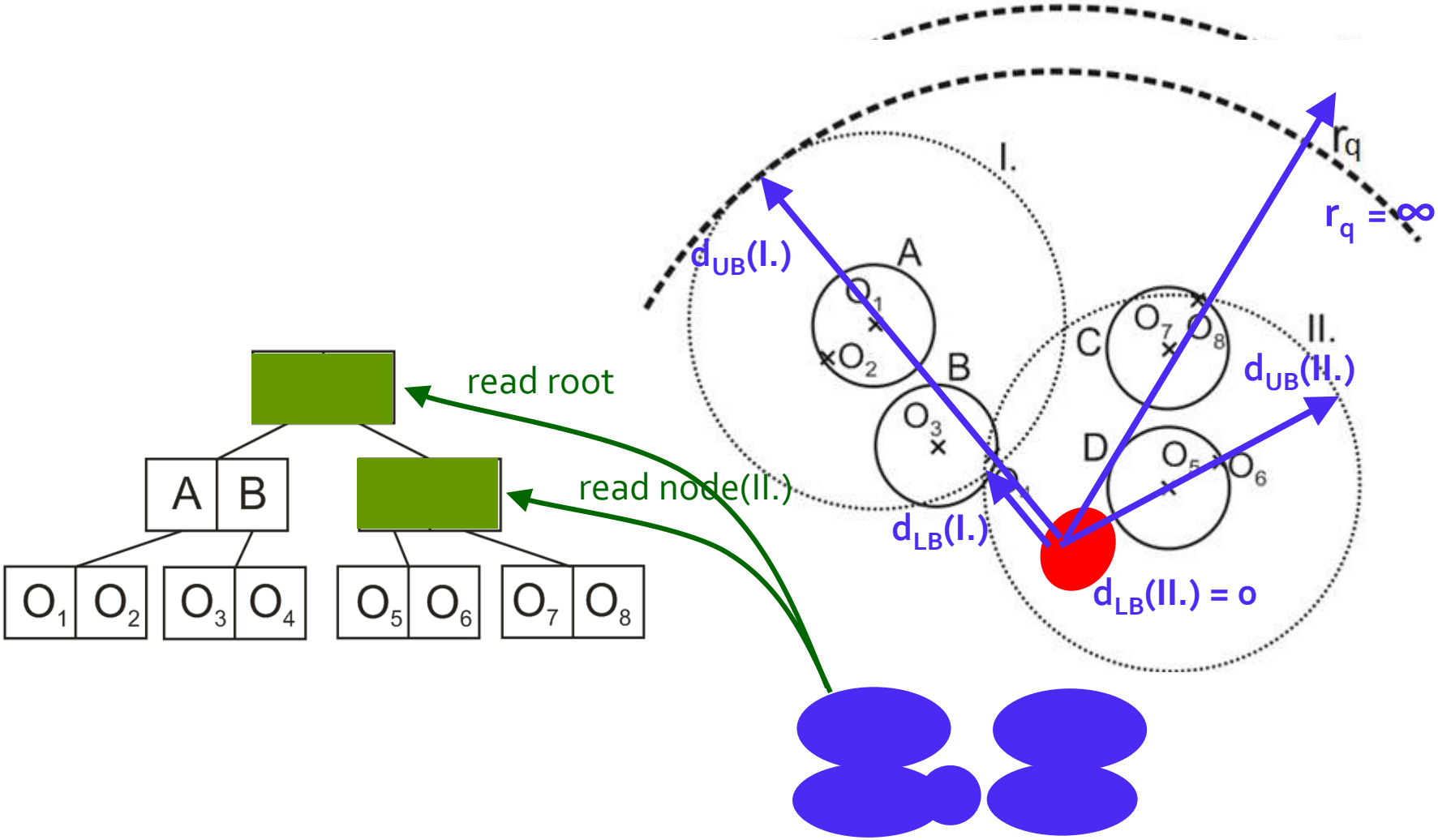
- M-tree is a metric index consisting of recursively nested ball regions (see previous lecture)
  - idea of R-tree generalized to metric spaces
- algorithm for exact kNN search developed (branch-and-bound)
  - based on a **priority queue PR** that maintains the nodes that cannot be excluded from the search
    - ordering of nodes by their (balls') lower-bound distances to q
  - kNN candidates + their distances to q are stored in a **NN array**
    - includes also candidate nodes (and upper-bound distances to q)
    - the kth entry of NN means the dynamic query radius



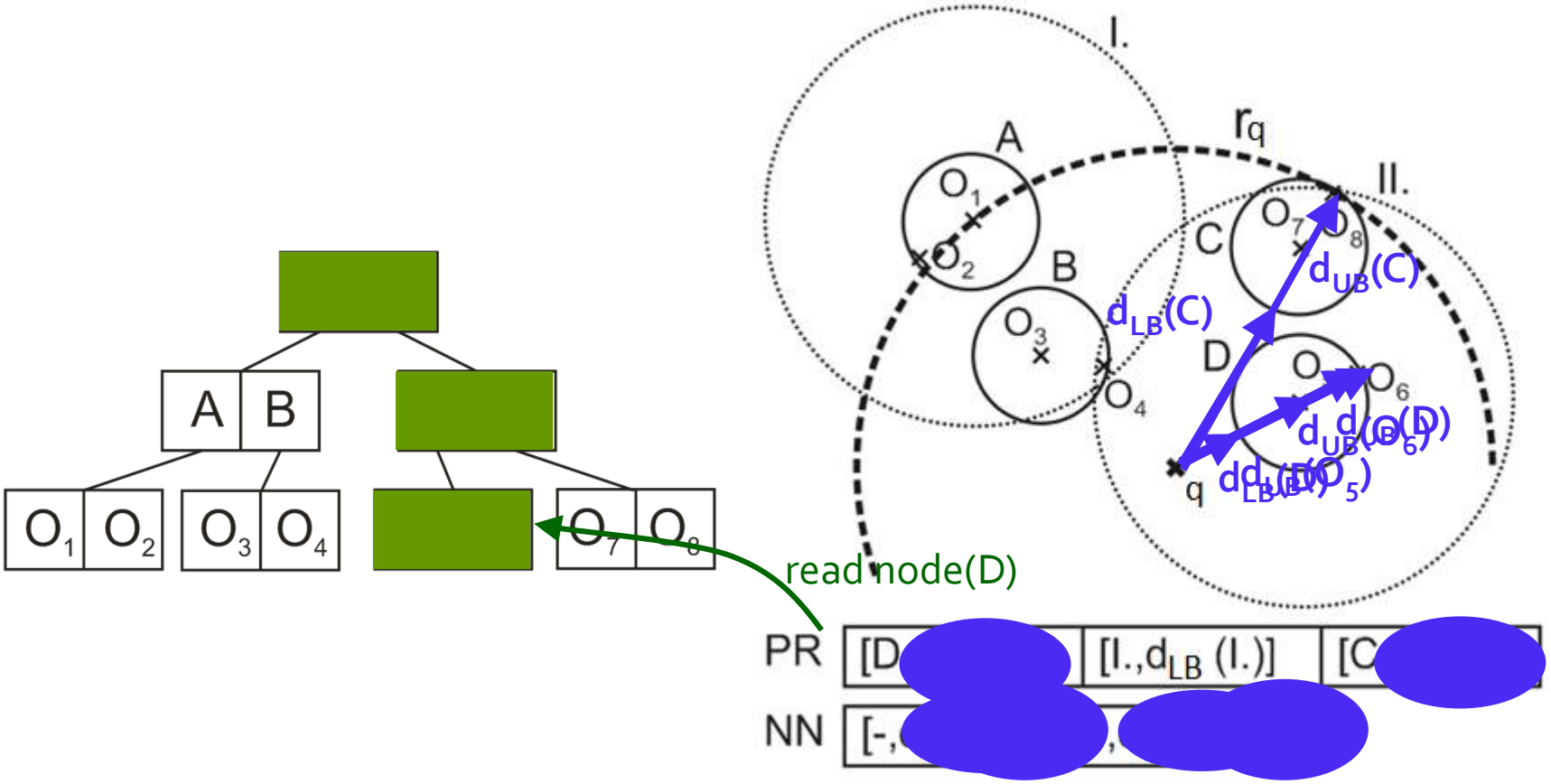
M-tree – exact kNN search

- the idea
  - consider a range query **(q, r_q)** processing, however the query radius $\mathbf{r_q}$ is dynamic
  - at the beginning the dynamic radius $\mathbf{r_q}$ is infinite, **PR queue** contains just the M-tree root, and **NN array** is empty
  - the M-tree is traversed, inserting into the priority queue PR nodes that have to be processed (i.e., the query ball overlaps the nodes' balls)
    - for each node in PR, its lower-bound distance (to the ball) is also stored
  - in the NN array, candidates to k nearest neighbors are stored
    - either directly objects (and distances to q) or candidate balls (and upper-bound distances to q)
    - NN[k] determines the dynamic query radius
  - during kNN processing, nodes are fetched from PR, updating PR and NN
    - note that NN[k] is decreasing, so that some nodes in PR are pruned prior to their fetch

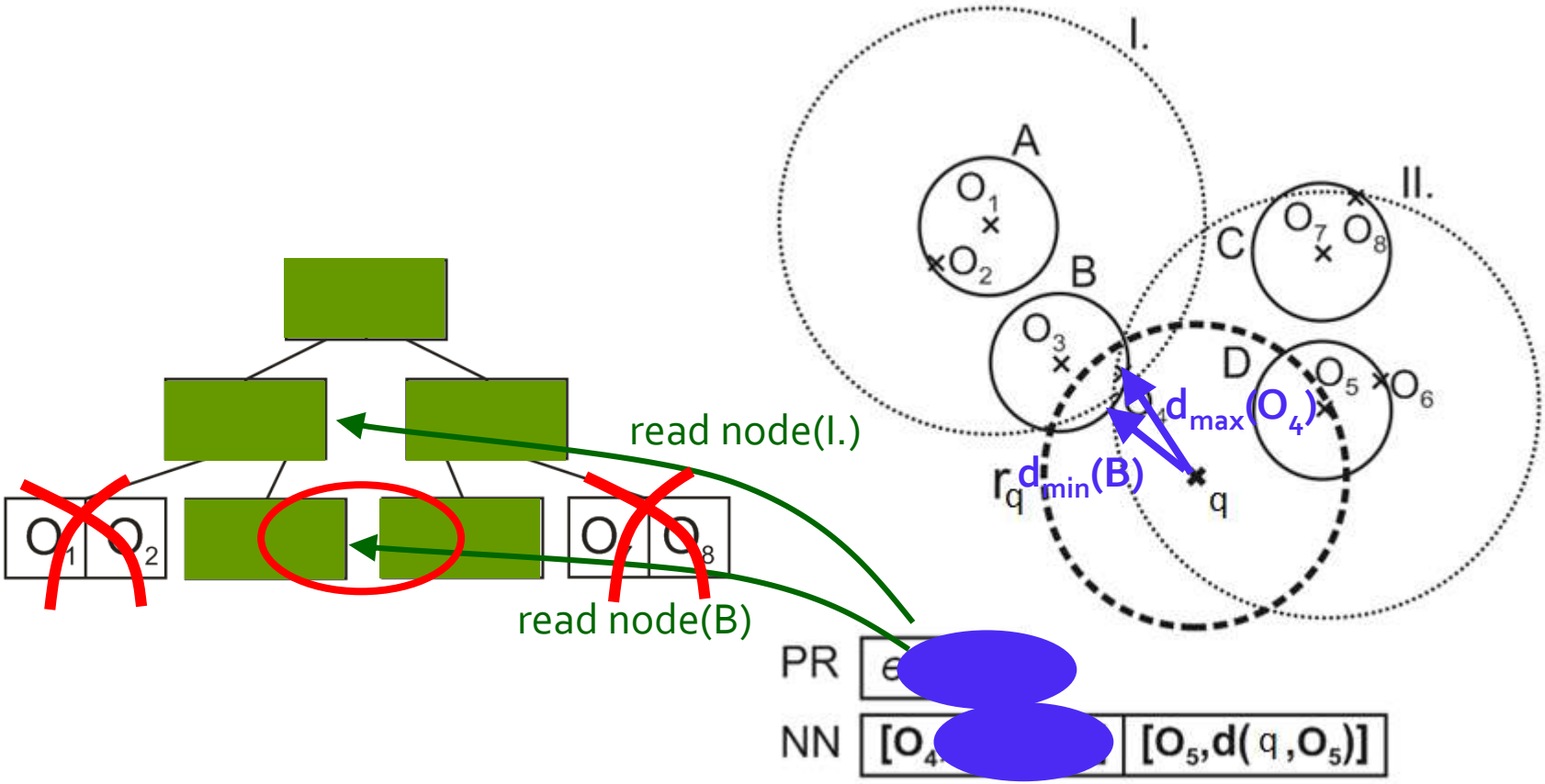
Example: exact 2NN search in M-tree



Example: exact 2NN search in M-tree



Example: exact 2NN search in M-tree

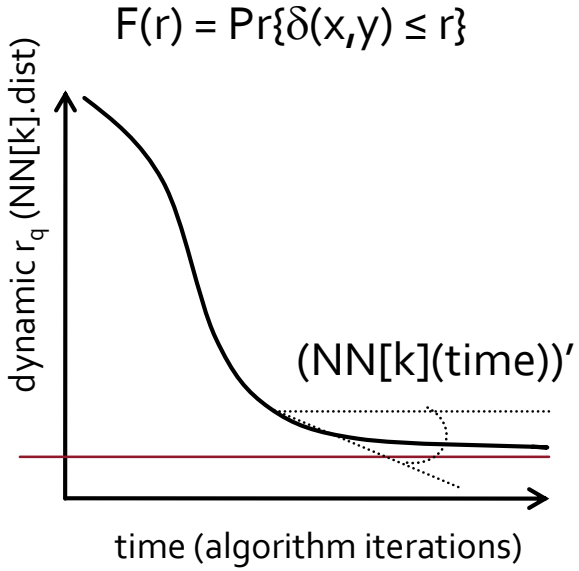


M-tree – approximate kNN search

- the exact kNN algorithm could be adjusted to provide faster approximate search
- we consider four heuristics
  - slowdown of improvements
    - if the NN array changes only slowly, let's terminate
  - approximately correct search
    - returned k nearest neighbors are guaranteed to be not much further than the true k nearest neighbors
  - good fraction approximation
    - if the distance to the candidate k^{th} neighbor is likely to represent small enough portion of the database, let's terminate
  - PAC queries (probably approximately correct queries)

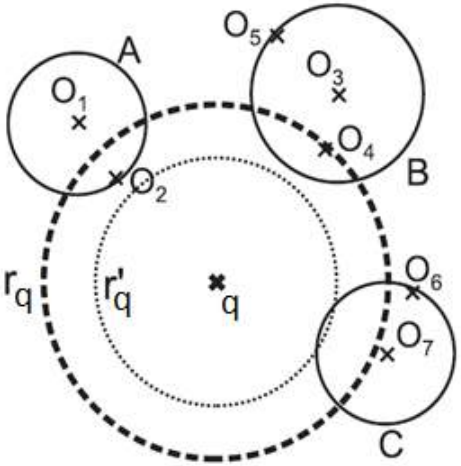
M-tree – approximate kNN search

- slowdown of improvements
  - consider the function NN[k] in time, i.e., how the dynamic query radius improves during the kNN processing
  - given a user-defined parameter κ,
    let's terminate the kNN search
    as soon as the first derivative (absolute value)
    of NN[k](time) falls below κ
    - i.e., the small improvements of the result
      are not worth the effort



M-tree – approximate kNN search

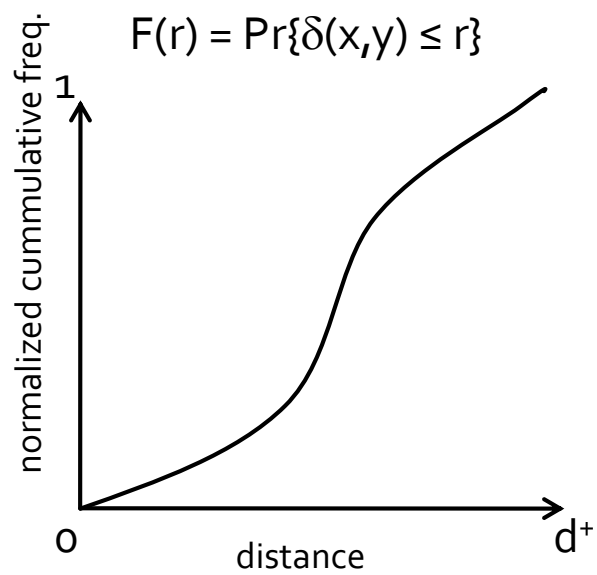
- approximately correct search
  - consider a user-defined error threshold ε, while relaxing the task of kNN search to $(1+\varepsilon)$kNN search
    - i.e., the neighbor returned is allowed to be up to $(1+\varepsilon)$ times further from **q** than the real one
  - algorithm adjustment
    - nodes from PR are filtered more aggressively using $r_q' = r_q/(1+\varepsilon)$ instead of r_q



PR [A,d_{LB}(A)] [C,d_{LB}(C)]
NN [O_4, d(q,O_4)]
↓
PR [C,d_{LB}(C)]
NN [O_4, d(q,O_4)]
↓
PR *empty*
NN [O_4, d(q,O_4)]

M-tree – approximate kNN search

- good fraction approximation
  - given query object **q** and a user-defined fraction **κ** (of the database) that has to be inspected during a (kNN) search
  - for a range query **(q, r)**, let's estimate the fraction of database falling into **(q, r)**
  - then, in the kNN algorithm we test if the database fraction estimated for the actual **(q, NN[k])** is greater than **κ** – if so, terminate
    - the database fraction for (q, r) is computed as $F(r) = Pr\{\delta(x,y) \leq r\}$ using the cummulative distance distribution histogram

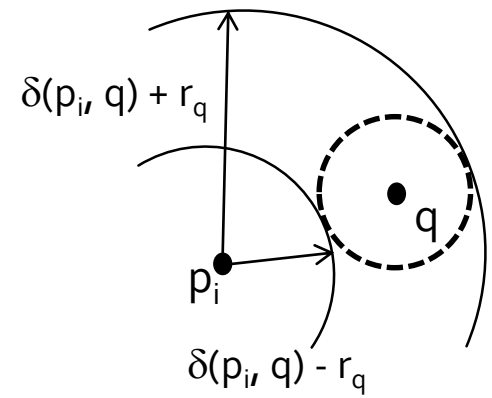


M-tree – approximate kNN search

- PAC queries
  - probably approximately correct queries
  - combination of the probabilistic and approximately correct methods
    - e.g., the two previous methods
  - in M-tree's kNN algorithm
    - nearest neighbors are searched, but guaranteeing only probability κ that the result objects will be at least $(1+\varepsilon)$ nearest neighbors
    - the terminating condition in the kNN algorithm
      - estimate the distance to k^{th} nearest neighbor (query radius $\mathbf{r_q}$)
        - by testing $F(r_q)$, until $F(r_q)$ * size of the database = k
      - nodes from PR are filtered using $r_q' = r_q/(1+\varepsilon)$

Approximate search in pivot tables

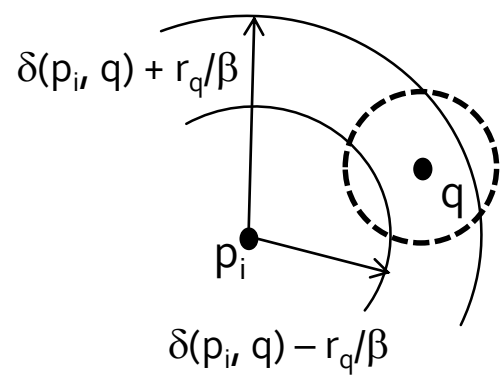
- consider pivot tables and a range query **(q, r_q)**
- using pivots, the exact search algorithm must check all objects that are within all the rings obtained by the pivots and the ranqe query (see on the right)



- in the approximate version, consider a user-defined threshold δ, that guarantees that the search is correct with the probability δ
  - the rings are "thinned" using β, computed as

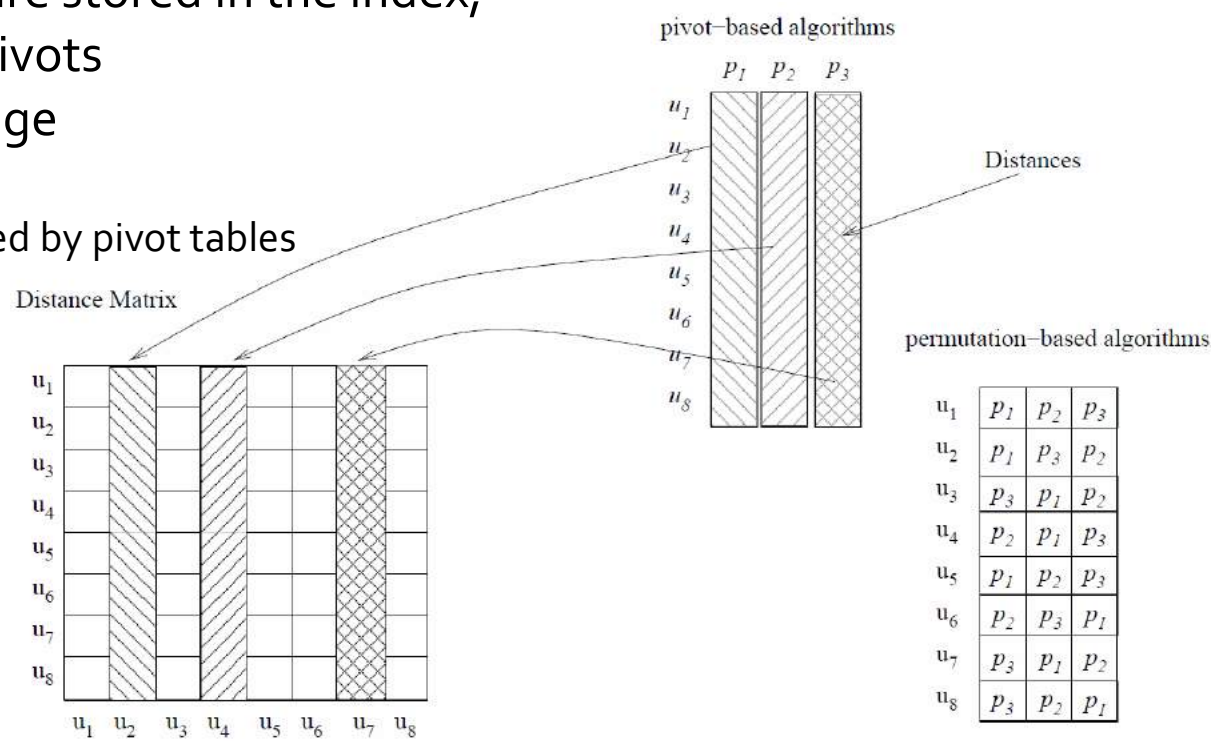
$$\beta \leq \frac{r_Q\sqrt{1-(1-\delta)^{\frac{1}{p}}}}{\sqrt{2}\sigma}$$

where p is number of pivots, and σ is the variance over the distance distribution



Permutation indexes

- consider a set of *k* pivots p_i and a pivot table (distance matrix)
- instead of using the distance matrix, let's create for each database object u a permutation of pivots Π_u, represented as an ordered set of pivots' indices
- hence, no distances are stored in the index, just the ordering of pivots
- allows compact storage
  - $kn \log_2 k$ bits
    instead of $32kn$ bits used by pivot tables



u_1	p_1	p_2	p_3
u_2	p_1	p_3	p_2
u_3	p_3	p_1	p_2
u_4	p_2	p_1	p_3
u_5	p_1	p_2	p_3
u_6	p_2	p_3	p_1
u_7	p_3	p_1	p_2
u_8	p_3	p_2	p_1

image from Chávez et al., Effective Proximity Retrieval by Ordering Permutations, IEEE Tran. on Patt. Anal. and Mach. Int., 30(9), 2008

Permutation indexes

- similarity for permutations
  - Spearman Rho $$S_\rho(\Pi_u, \Pi_q) = \sum_{1 \le i \le k} \left(\Pi_u^{-1}(i) - \Pi_q^{-1}(i)\right)^2$$
  - Spearman Footrule $$F(\Pi_u, \Pi_q) = \sum_{1 \le i \le k} \left|\Pi_u^{-1}(i) - \Pi_q^{-1}(i)\right|$$

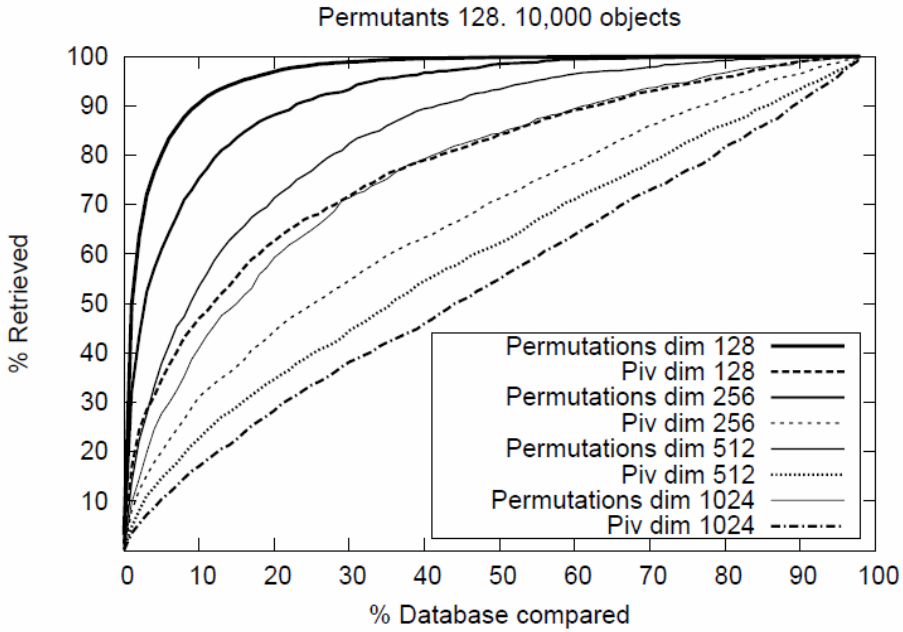
  where $\Pi^{-1}_u(i)$ is the position of pivot p_i in the permutation Π_u

- thus, instead of the **geometric** framework the **combinatorial** framework is used

Permutation indexes

- the approximate queries using permutation indexes are, in fact, similar to the pivot-table ones
- instead of L_∞ on vectors in pivot space, the Spearman Rho/Footrule on pivot permutations is used for sorting the objects in the first step
- the second (refinement) step is the same, using some early termination condition

Permutation indexes



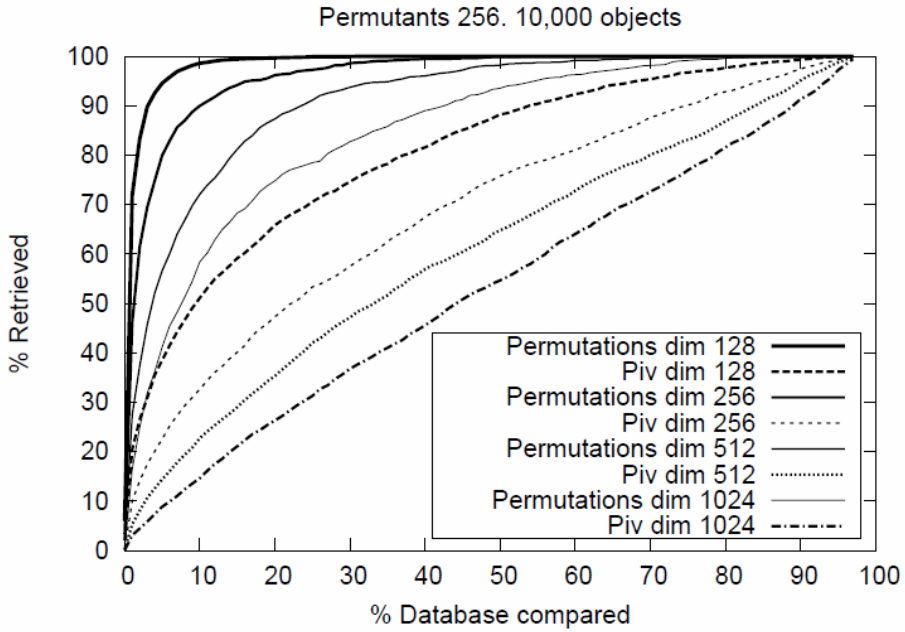


image from Chávez et al., Effective Proximity Retrieval by Ordering Permutations, IEEE Tran. on Patt. Anal. and Mach. Int., 30(9), 2008