OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG | INF | FAKULTÄT FÜR INFORMATIK

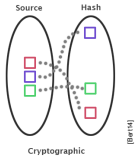
5. Perceptual Hashing of Images

Byron Worms

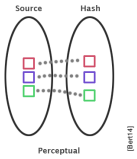
Contents

Perceptual hashing

- Cryptographic hash functions
  - Map data to hash values with high dispersion
  - Minor changes yield in a completely different hash
  - *Measurement of equality*



- Perceptual hash functions
  - Map data to fixed sized hash values
  - Maintain the data's correlation
  - *Measurement of similarity*



Task specifications

- Implementation of a front-end for perceptual hashing of images
  - Back-end C++ library is *phash*
  - Operational modes:
    - RADISH
    - DCT
    - Marr/mexican hat wavelet
    - Extended by: BMB
- Analyses of
  - Detection rates for similiar images
  - Detection rates for dissimilar images
  - Speed improvements due to the use of perceptual hashing

Algorithms

Algorithm	Keyword	Comparison	Hash length
RADISH[1)]	Radon projection	Cross correlation	320 bit
DCT[2)]	Fourier-related transformation	Hamming distance	64 bit
Wavelet[3)]	Edge-detection	Hamming distance	576 bit
BMB[4)]	Uniform grid	Hamming distance	variable

[Zau10]

- [1)] Radial hash projections
- [2)] Discrete cosine transformation
- [3)] Marr/mexican hat wavelet
- [4)] Block mean value based

Test data

- 1. Set: Plain colored images
  - 9 different colors
  - Specifications: JPEG encoded, 600x600px
- 2. Set: Images with elementary brushes
  - 3 groups of structural images (cycles, rectangles, various shapes)
  - Besides the original images, different modifications have been applied
    - Resizing (down, up), rotation/flipping, gamma corrections, noise
  - 54 images in each group → 162 images in total
  - Specifications: JPEG encoded, 600x600px
- 3. Set: Complex images
  - General structure as above
  - Contains similar and dissimilar images
  - 21 images in each group → 126 images in total
  - Specifications: JPEG encoded, 1024x768px

Test data /Examples

- 2. Image set:

- 3. Image set:

[CMR]

Experiments

- Different experiments were defined on the image sets
  - Basic functionality tests
  - More complex analyses
  - Speed investigations
- Configuration for the analyses:
  - Standard values for all algorithms
  - Threshold constant at 90%
  - Custom developed tool (see live demo!)
  - Test system with following characteristics:

System information	
Operation system	Windows 8.1 x64
Processor	AMD x6 1055t (~3.3ghz)
Memory	8GB DDR3
Hard disk	Samsung SSD 850 EVO 512GB

(own research)

Basic functionality tests

1. Analysis of plain colored images
   - 9 images → 45 comparison pairs
   - Results depend on the algorithms' nature
     - $FRR = 1, FAR = 1$
2. Analysis of simple comparison pairs (self-tests)
   - 288 images from image sets 2 & 3
     - $FRR = 0$

[Pic15]

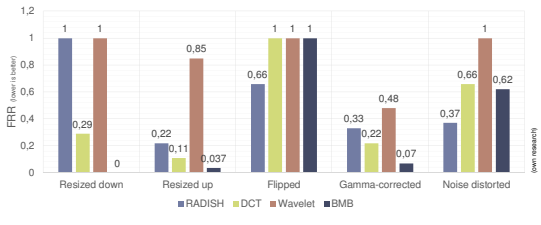
Original versus modified images

- Analyses are done as follows:
  - Original vs. resized down
  - Original vs. resized up
  - Original vs. flipped
  - Original vs. gamma-corrected
  - Original vs. noise distorted
- Performed for image sets:
  - 2. Set: 135 comparison pairs in total
  - 3. Set: 105 comparison pairs in total
  - No set intersections!
- *FRR* as measurement

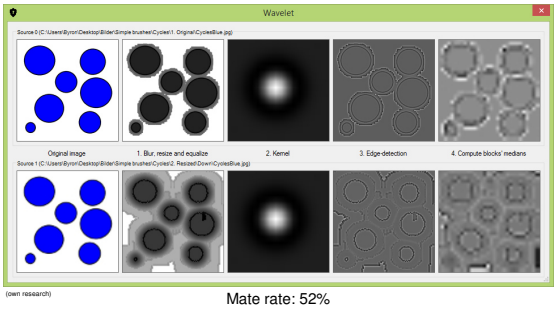
[Pic15]

Original vs. modified images /Brushes

- Simplified image structure
  - Vulnerable against basic image modifications
  - Modifications heavily influence the structural content
    - Serious textural alternations



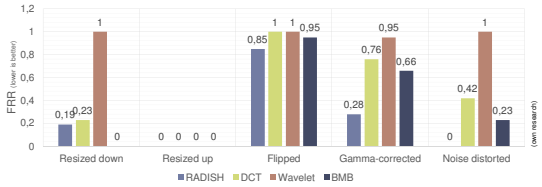
Original vs. modified images /Brushes



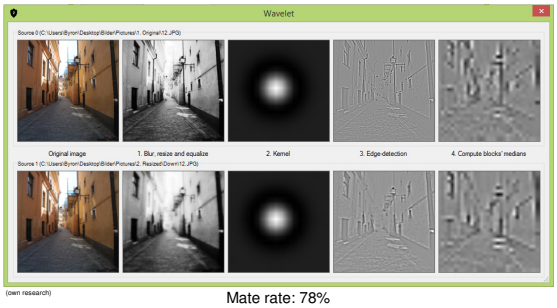
Mate rate: 52%

Original vs. modified images /Images

- Complex structural image content
  - Modifications don't affect the content seriously
- Faults increased for *gamma-correction*
  - Parameterization to blame



Original vs. modified images /Images



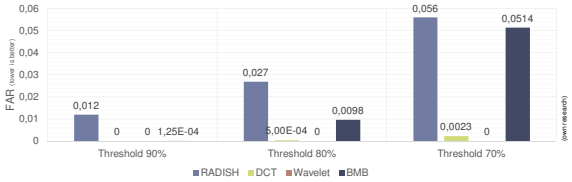
Mate rate: 78%

Cross-comparison

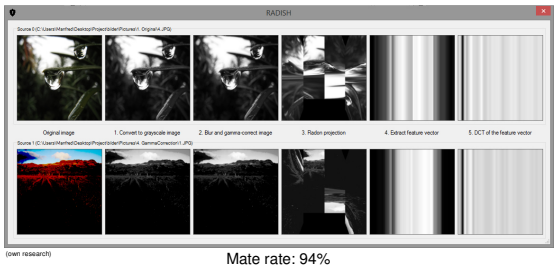
- Performed for 3. image set only
  - In total 8001 possible comparison pairs
  - 2. Set contains only simplified images
- Analystical passes:
  - 1. Threshold set to 90%
  - 2. Threshold set to 80%
  - 3. Threshold set to 70%
- *FAR* as measurement
  (in respect of the set threshold)

Cross-comparison /2

- RADISH: Similiar light distributions yield in related radon projections
  - Feature vectors are alike
- DCT (70%): Just similiar images introduce faults
- BMB: Similar structural image content with regard to the accepted error



Cross-comparison /RADISH



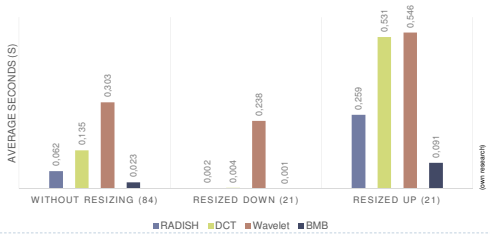
Mate rate: 94%

Speed Investigation

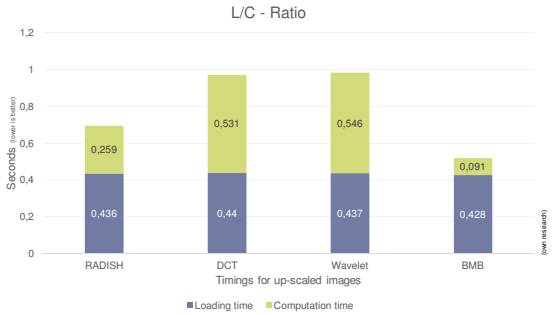
- Depending on a variety of conditions
  - Implementation of the algorithms and parametrizations
  - Processor type (with respect to speed and concurrency), RAM
  - Communication between different infrastructures
    (e.g. hosting hard disk, download speed)
  - (Operation system and platform)
- Test case:
  - Distinction between original and resized images
  - *Computation time* as measurement
  - Analysis just for complex images
  - Magnification factors: $\frac{1}{6}x, 2x$

Speed Investigation /Complex images

- Best: BMB and RADISH
  - Simple pixel operations to extract features
- Slowest: Wavelet and DCT
  - Complex pixel operations to obtain features



Speed Investigation /Ratio



Speed Investigation /Optimization

- Both images are loaded and computed
  - Total time: 2x loading time + 2x computation time
- Store template image's hash to file
  - E.g. 80 bytes for hex. RADISH hash
  - Loading time: ~0,000246s
  - Total time: (1x loading time + 1x computation time) + 1x loading time (template)
- Example (RADISH, up-scaled):
  - Load/compute both: ~1,39s in total
  - Store template's hash: ~0,695246s in total
  - ~49% saving of time

References

- [Bert14] http://bertolami.com/index.php?engine=blog&content=posts&detail=perceptual-hashing (visited: 10.06.15)
- [Pic15] http://www.flaticon.com/ (visited: 10.06.2015)
- [CMR] *Course media repositry (CMR)*
- [Zau10] *Implementation and Benchmarking of Perceptual Image Hash Functions*, Christop Zauner, 2010

Live demo

Thank you for your attention

Speed Investigation /Brushes

- Magnification factors: $\frac{1}{6}x, 6x$

- Best: BMB and RADISH
- Slowest: Wavelet and DCT