# Fast Coding Unit Selection Based on Local Texture Characteristics for HEVC Intra Frame

Milos Radosavljević, Georgios Georgakarakos, Sebastien Lafond and Dejan Vukobratović

Åbo Akademi University

IEEE GlobalSIP
3rd IEEE Global Conference on Signal & Information Processing
Orlando, Florida, USA December 14-16 2015

# Outline

- HEVC Introduction
- HEVC Intra Coding
- HEVC Complexity and Performance - Motivation for our research
- Local Binary Patterns - LBP
- Proposed Solution
- Simulation Results
- Q/A

# HEVC Introduction

- Novel video compression standard
- High importance to the industrial and academic circles due to
  - New devices with wide range of different resolution
  - New services – live sports, live music, other events, home theater, mobile streaming (for e.g. new Twitter video)
  - Increasing need for video transmission: most IP traffic is video
  - Increasing use of parallel processing architectures
- Goal was to achieves 2x higher compression compared to H.264/AVC and to improve throughput

# HEVC Introduction

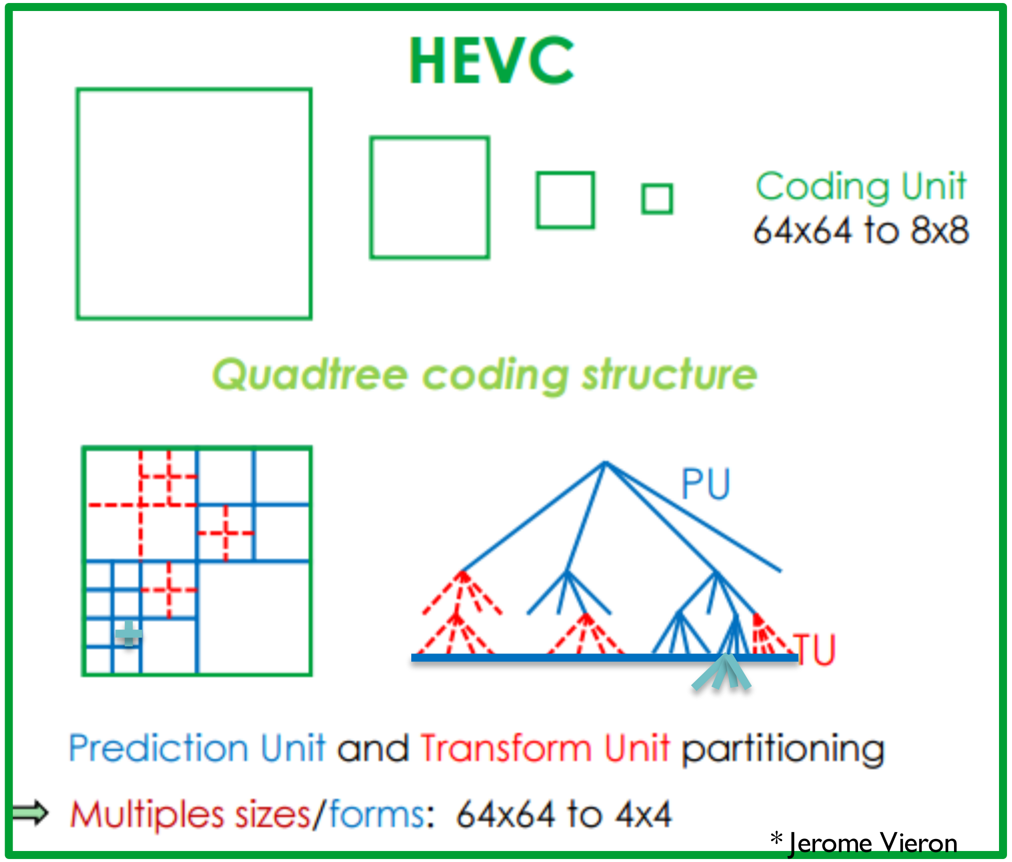
- Block based compression approach
  - Each frame is segmented into blocks
- Larger and flexible block size
- More intra prediction modes
- Larger DCT transform sizes
- High throughput CABAC
- New Advanced Motion Vector Precision with ¼ pixel accuracy
- Etc.

# HEVC Intra Coding – block structure

- Support more flexible block size – quadtree structure
- Basic processing unit is Coding Unit - CU
- CU size ranges from 8x8 up to 64x64 pixels
- Each CU can incorporate within its structure PUs and TUs with different sizes
  - from 4x4 to 64x64 for PU
  - from 4x4 to 32x32 for TU

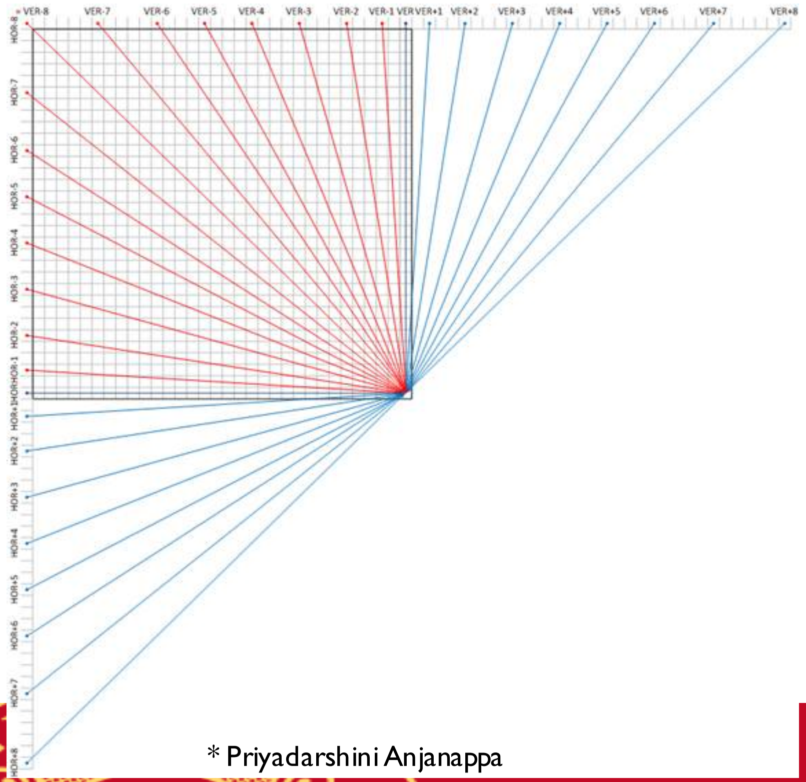
# HEVC Intra Coding – block structure

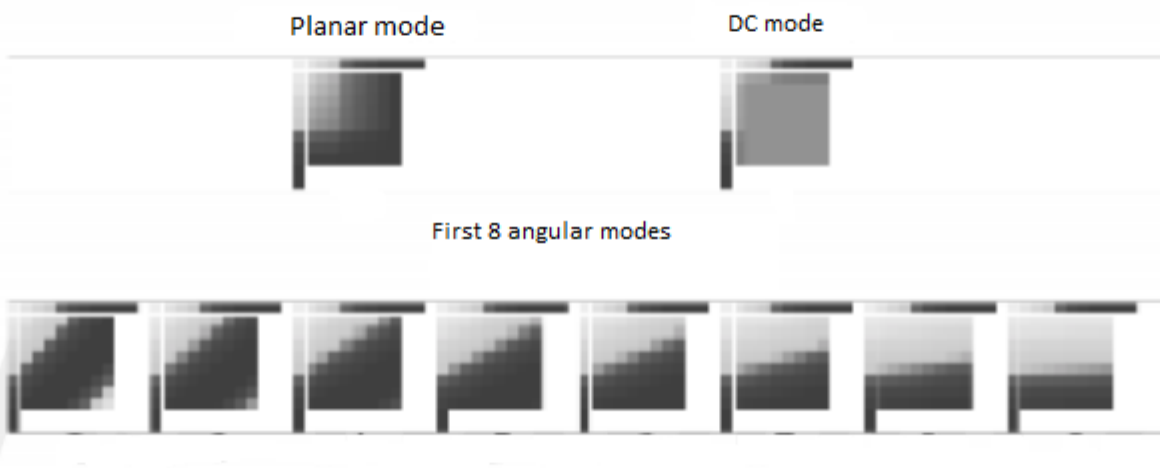
- Quadtree structure example
- Green – CU (64x64-8x8)
- Blue – PU (64x64-4x4)
  - For Intra: 2Nx2N mode or NxN mode
- Red – TU(32x32-4x4)
  - DCT/DST transform




# HEVC Intra Coding – prediction modes

- Number of intra prediction modes grows up to 35, compared with 9 in H.264/AVC
- HEVC uses DC, Planar, and 33 angular modes






* Priyadarshini Anjanappa

# HEVC Complexity/Motivation for our work

- Decisions are based on cost minimization
  - $J = D + \lambda * R$
  - Rate-Distortion Optimization based on Lagrangian method
- In order to get D and R in brute force approach
  - Full forward (encoding) process has to be performed
  - Full inverse (decoding) process has to be performed
- Prediction, DCT, quantization, entropy coding, get R, entropy decoding, inverse quantization, IDCT, get D

# HEVC Complexity/Motivation for our work

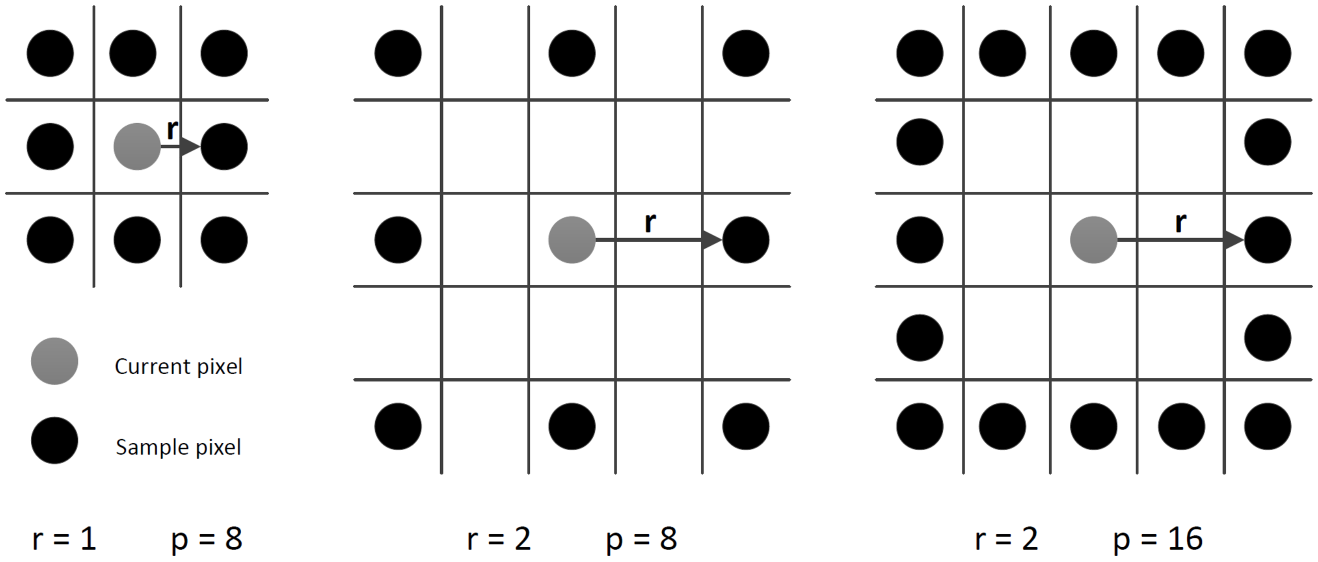
- To decide an optimal quadtree structure and prediction mode
  - Go trough all possible outcomes in terms of different block sizes and different prediction modes
  - Minimize the cost J
- Time consuming process
  - Impossible to do it in real time
- HEVC reference software has implemented fast approaches
  - Still space (and need) for more speed-up and simplification
  - For the price of some performance degradation

# HEVC Complexity/Motivation for our work

- Correlation between the CU size and the texture within the CU
- RDO tend to use smaller CUs
  - Rich texture regions
  - On the edge between objects
  - Between objects and background
  - Transition between different textures
- Use texture descriptors to decide
  - If CU contains different textures
  - Flat regions
  - Texture rich/complex regions

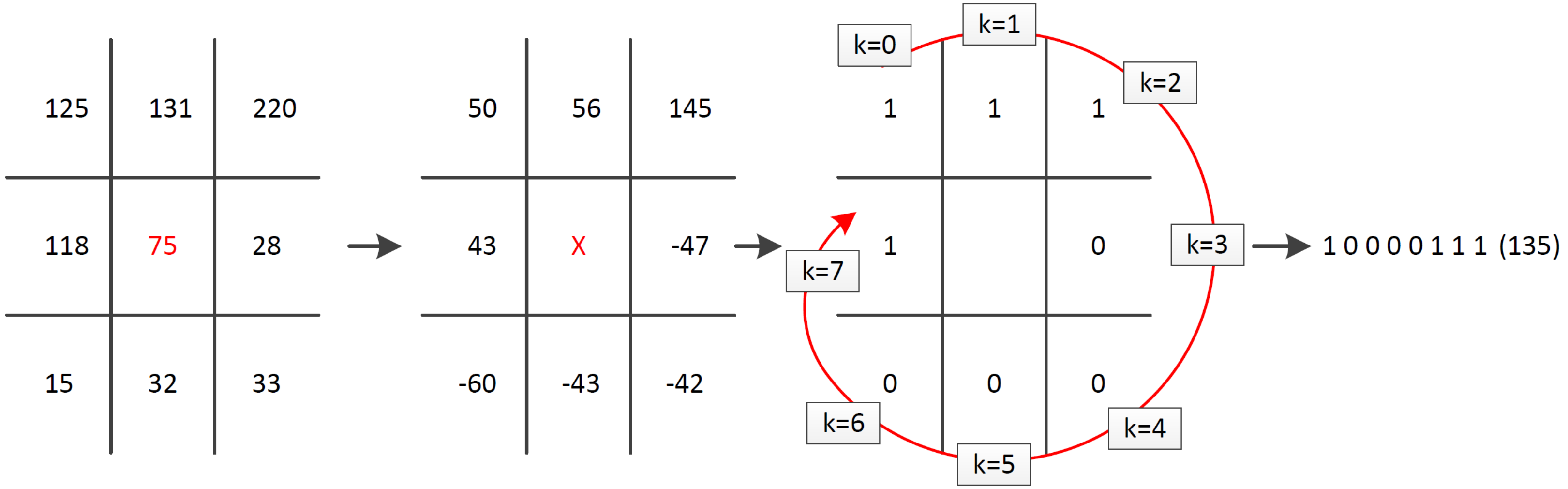
# Local Binary Patterns - LBP

- LBP as texture descriptor
  - Histograms of LBPs
- Calculated on pixel level
  - assigning to each pixel a corresponding LBP value (LBP code)



# Local Binary Patterns - LBP

$$LBP_{p,r}^{u} = \sum_{k=0}^{p-1} I(sp_k - cp)2^k \qquad I(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$




Original pixel values

Difference with the sample pixels

Threshold values and sampling order

Binary representation of the LBP code and corresponding decimal value

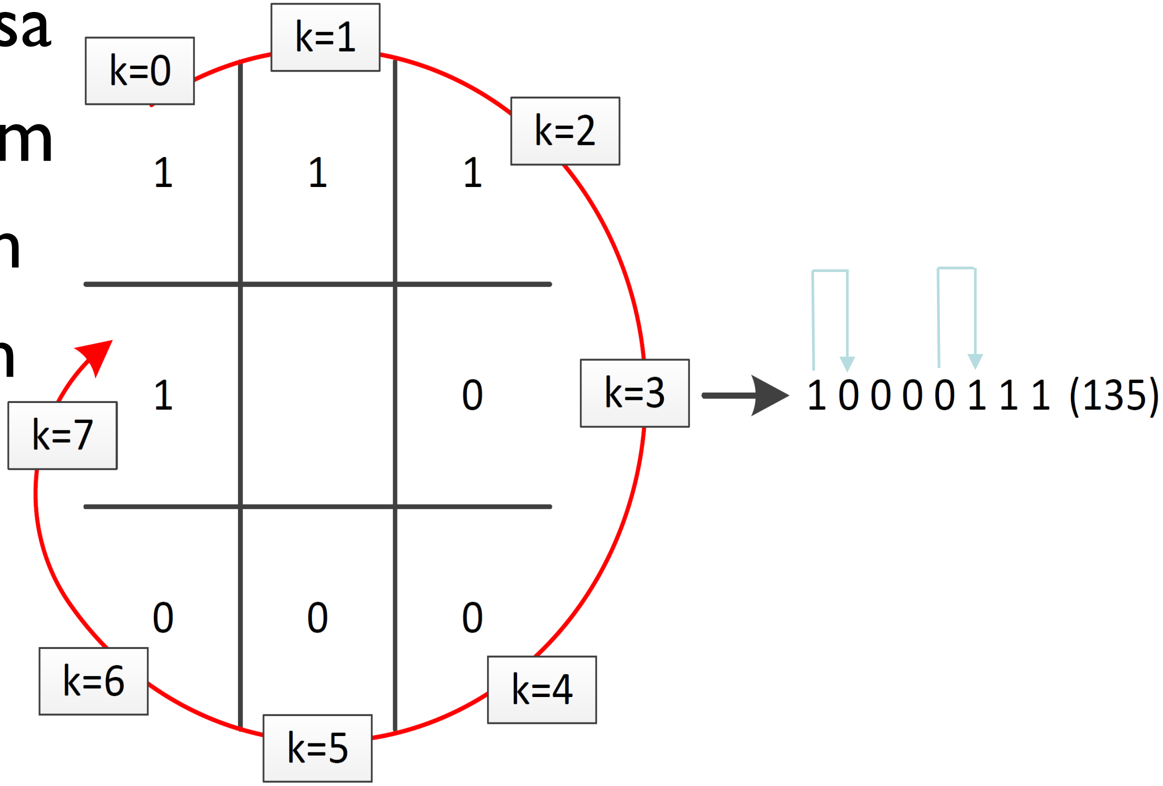
# Local Binary Patterns - LBP

- Fast implementation using only shifts and logical operations

$$LBP_{code}| = (sp(i-1, j-1) > sp(i, j)) << 0;$$
$$LBP_{code}| = (sp(i-1, j) > sp(i, j)) << 1;$$
$$LBP_{code}| = (sp(i-1, j+1) > sp(i, j)) << 2;$$
$$LBP_{code}| = (sp(i, j+1) > sp(i, j)) << 3;$$
$$LBP_{code}| = (sp(i+1, j+1) > sp(i, j)) << 4;$$
$$LBP_{code}| = (sp(i+1, j) > sp(i, j)) << 5;$$
$$LBP_{code}| = (sp(i+1, j-1) > sp(i, j)) << 6;$$
$$LBP_{code}| = (sp(i, j-1) > sp(i, j)) << 7;$$

# Local Binary Patterns - LBP

- Uniform patterns
  - u=2 in our research
  - 2 transitions from 0 to 1 and vice versa
- Group uniform patterns into histogram
  - Each uniform allocated to separate bin
  - All non-uniform voted to only one bin
- For u=2 ⇒ 58 uniform codes
  - 58 + 1 dimensional histogram

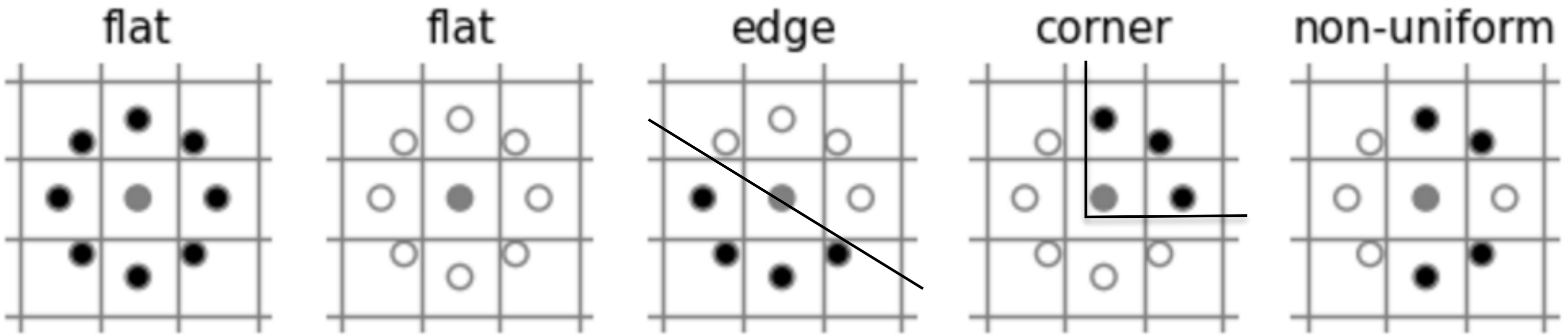


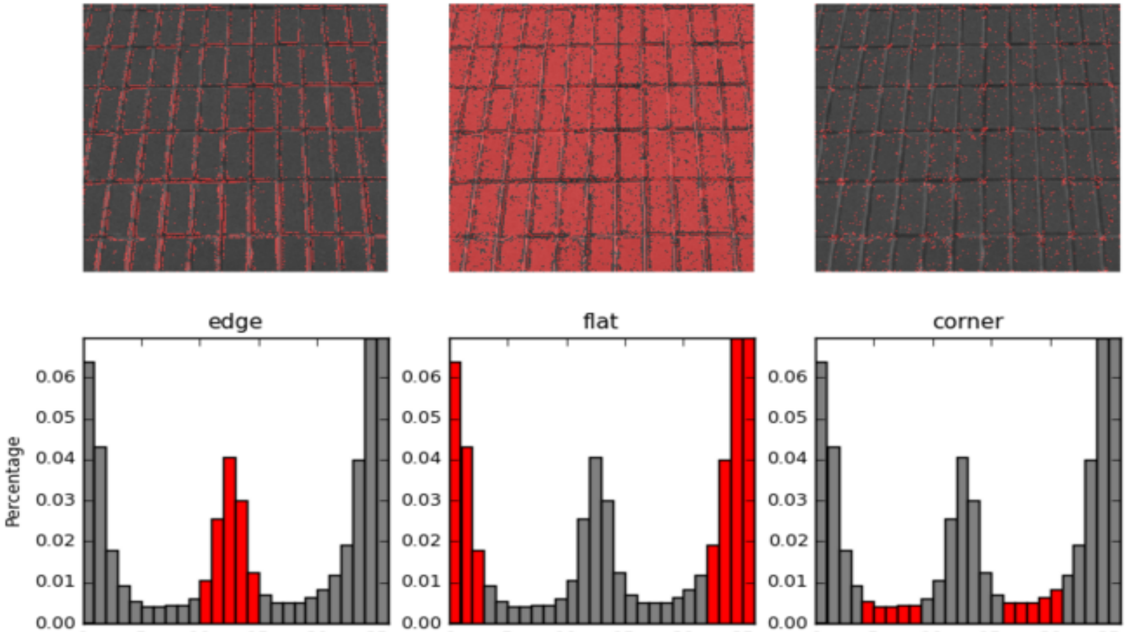
# Local Binary Patterns - LBP

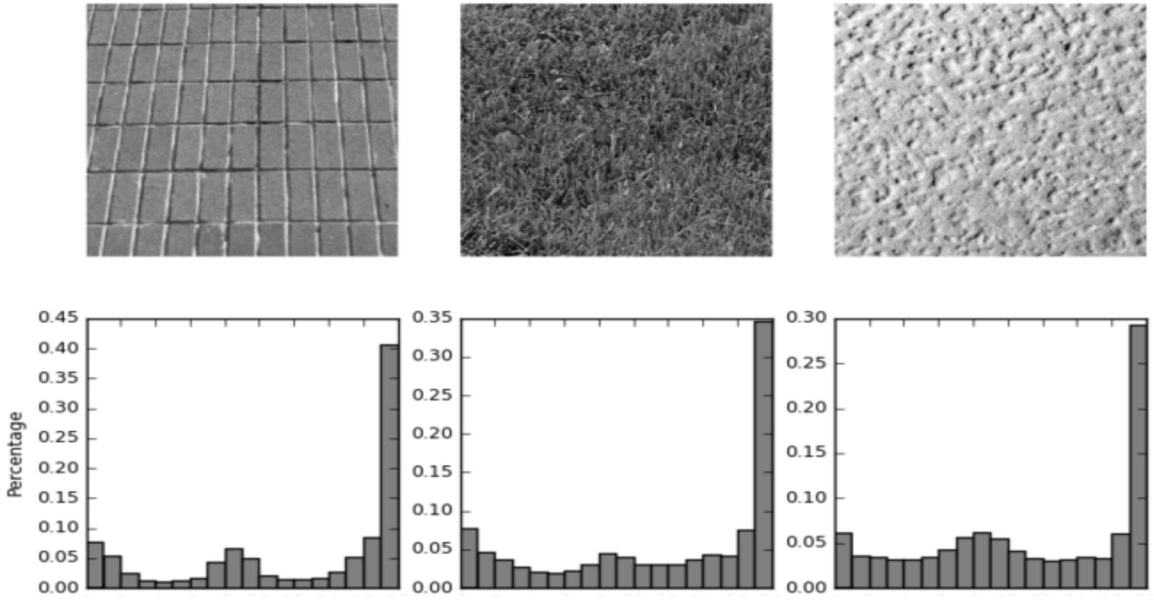
| No. | value | | No. | value | | No. | value | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 00000000 | 21 | 62 | 00111110 | 41 | 207 | 11001111 |
| 2 | 1 | 00000001 | 22 | 63 | 00111111 | 42 | 223 | 11011111 |
| 3 | 2 | 00000010 | 23 | 64 | 01000000 | 43 | 224 | 11100000 |
| 4 | 3 | 00000011 | 24 | 96 | 01100000 | 44 | 225 | 11100001 |
| 5 | 4 | 00000100 | 25 | 112 | 01110000 | 45 | 227 | 11100011 |
| 6 | 6 | 00000110 | 26 | 120 | 01111000 | 46 | 231 | 11100111 |
| 7 | 7 | 00000111 | 27 | 124 | 01111100 | 47 | 239 | 11101111 |
| 8 | 8 | 00001000 | 28 | 126 | 01111110 | 48 | 240 | 11110000 |
| 9 | 12 | 00001100 | 29 | 127 | 01111111 | 49 | 241 | 11110001 |
| 10 | 14 | 00001110 | 30 | 128 | 10000000 | 50 | 243 | 11110011 |
| 11 | 15 | 00001111 | 31 | 129 | 10000001 | 51 | 247 | 11110111 |
| 12 | 16 | 00010000 | 32 | 131 | 10000011 | 52 | 248 | 11111000 |
| 13 | 24 | 00011000 | 33 | 135 | 10000111 | 53 | 249 | 11111001 |
| 14 | 28 | 00011100 | 34 | 143 | 10001111 | 54 | 251 | 11111011 |
| 15 | 30 | 00011110 | 35 | 159 | 10011111 | 55 | 252 | 11111100 |
| 16 | 31 | 00011111 | 36 | 191 | 10111111 | 56 | 253 | 11111101 |
| 17 | 32 | 00100000 | 37 | 192 | 11000000 | 57 | 254 | 11111110 |
| 18 | 48 | 00110000 | 38 | 193 | 11000001 | 58 | 255 | 11111111 |
| 19 | 56 | 00111000 | 39 | 195 | 11000011 | 59 | all other | |
| 20 | 60 | 00111100 | 40 | 199 | 11000111 | | | |

# Local Binary Patterns - LBP

- Intuitior



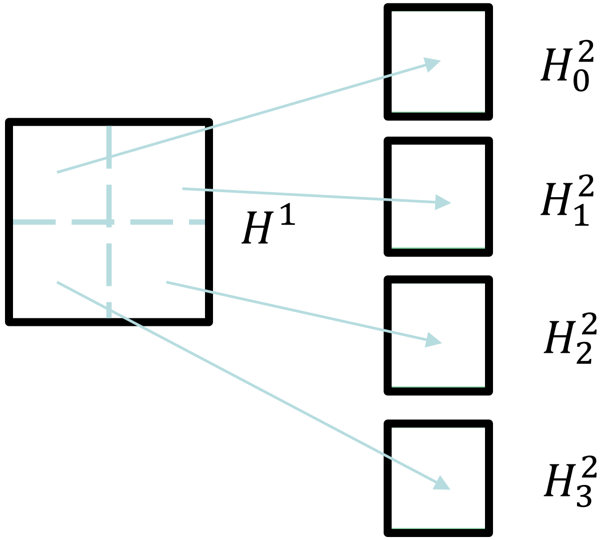




*skit-image database

# Fast CU size selection based on local texture descriptors

- Based on the LBP histogram matching
- Occurrence histogram $H^d$ is formed on the CU level
- Sub-histograms are calculated for sub-CUs on the higher depth $H_i^{d+1}$
- To make decisions use

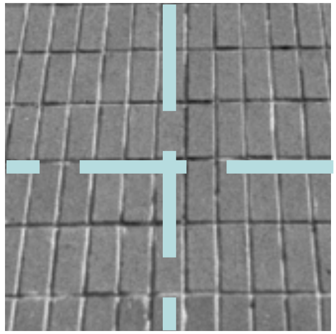
  $S(H^d; H_i^{d+1})$ similarity measure

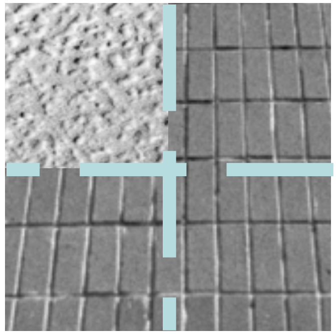


# Fast CU size selection based on local texture descriptors

– $S(H^d; H_i^{d+1})$ similarity measure function

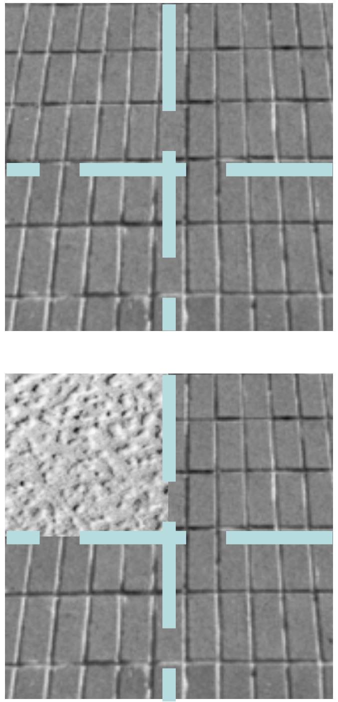
- Sufficiently high similarity $(\forall_i)$ $S(H^d; H_i^{d+1})$
  – CU consists only of one texture region
- Otherwise if $(\exists_i)$ $S(H^d; H_i^{d+1})$ is low
  – High possibility that the CU is composed of different textures
  – More likely that CU is going to be split





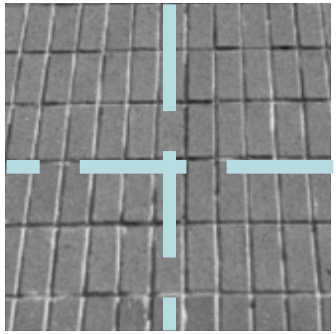
# Fast CU size selection based on local texture descriptors

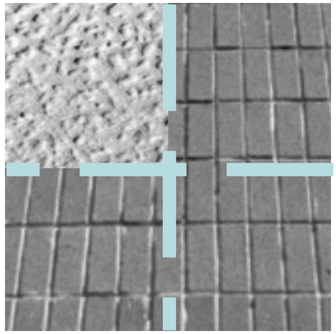
- CU split decision is given by
  - S($H^d$; $H_i^{d+1}$)$\leq thr(Qp)$
  - For at least one i
- If S($H^d$; $H_i^{d+1}$)$\leq thr(Qp)$ is not satisfied
  - 4 different approaches based on performance-complexity tradeoff



# Fast CU size selection based on local texture descriptors

- Criterion1
  - all further calculation is omitted and current CU depth is selected
- Criterion2
  - one additional depth with RDO is performed
- Criterion3
  - two additional depths are calculated with RDO
- Criterion4
  - all additional depths are calculated with RDO





# Performance

BD-PSNR AND TIME REDUCTION FOR THE PROPOSED ALGORITHM

| Sequence | Criterion1 | | Criterion2 | | Criterion3 | | Criterion4 | |
|---|---|---|---|---|---|---|---|---|
| | BD-PSNR [dB] | ΔT [%] | BD-PSNR [dB] | ΔT [%] | BD-PSNR [dB] | ΔT [%] | BD-PSNR [dB] | ΔT [%] |
| PeopleOnStreet(ClassA) | -0.9826 | 76.6890 | -0.5689 | 62.6812 | -0.1561 | 21.9119 | -0.0163 | 4.0559 |
| Cactus(ClassB) | -0.6277 | 80.2024 | -0.3486 | 57.2062 | -0.1249 | 25.1177 | -0.0062 | 3.2532 |
| RaceHorses1(ClassC) | -0.9997 | 81.4038 | -0.5296 | 58.8562 | -0.1348 | 19.0552 | -0.0068 | 6.9282 |
| BlowingBubbles(ClassD) | -1.0342 | 80.7887 | -0.4416 | 45.0893 | -0.0955 | 14.3006 | -0.0011 | 6.9940 |
| Johnny(ClassE) | -1.0230 | 79.0388 | -0.5330 | 58.8651 | -0.1435 | 24.7559 | -0.0502 | 2.6000 |
| Lena_Image | -0.5871 | 83.0630 | -0.2895 | 58.2412 | -0.0682 | 25.8615 | -0.0104 | 8.5753 |
| Average | -0.8757 | 80.1976 | -0.4519 | 56.8232 | -0.1205 | 21.8338 | -0.0152 | 5.4011 |

# Questions ?

# Backup slide

Similarity measure-histogram correlation

$$S(H_1, H_2) = \frac{\sum_I (H_1(I) - \overline{H}_1)(H_2(I) - \overline{H}_2)}{\sqrt{\sum_I (H_1(I) - \overline{H}_1)^2 \sum_I (H_1(I) - \overline{H}_1)^2}}$$

where,

$$\overline{H}_k = \frac{1}{N} \sum_J H_k(J)$$

Threshold values

| Qp | thr | Qp | thr | Qp | thr | Qp | thr | Qp | thr |
|---|---|---|---|---|---|---|---|---|---|
| 0-7 | 0.99 | 16 | 0.75 | 25 | 0.66 | 34 | 0.57 | 43 | 0.44 |
| 8 | 0.98 | 17 | 0.74 | 26 | 0.65 | 35 | 0.56 | 44 | 0.41 |
| 9 | 0.95 | 18 | 0.73 | 27 | 0.64 | 36 | 0.55 | 45 | 0.38 |
| 10 | 0.93 | 19 | 0.72 | 28 | 0.63 | 37 | 0.54 | 46 | 0.35 |
| 11 | 0.90 | 20 | 0.71 | 29 | 0.62 | 38 | 0.53 | 47 | 0.32 |
| 12 | 0.88 | 21 | 0.70 | 30 | 0.61 | 39 | 0.52 | 48 | 0.29 |
| 13 | 0.83 | 22 | 0.69 | 31 | 0.60 | 40 | 0.51 | 49 | 0.26 |
| 14 | 0.79 | 23 | 0.68 | 32 | 0.59 | 41 | 0.50 | 50 | 0.23 |
| 15 | 0.76 | 24 | 0.67 | 33 | 0.58 | 42 | 0.47 | 51 | 0.20 |

Other similarity metrics can be in use but it requires thr recalculation