Pit Pattern Classification of Zoom-Endoscopic Colon Images using DCT and FFT

Leonhard Brunauer Hannes Payer Robert Resch

Department of Computer Science
University of Salzburg

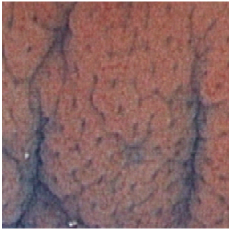
February 1, 2007

Outline

Goals

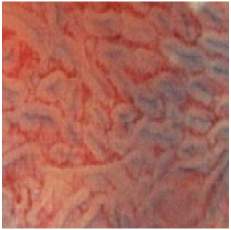
- Classify images retreived from colonoscope.
- 6 class problem
  - Classify according to Pit Pattern classes.
  - Six cancer classes (`I`, `II`, `III L`, `III S`, `IV`, `V`)
- 2 class problem
  - Classify as either "operation required" or "operation not required".
  - Type `I` and `II` need not be removed.
  - Type `V` cannot be removed.
  - Type `III` and `IV` should be removed.

Pit patterns



Type I

Type II



Type IV

Goals (cont'd)

- Previous work done at this university.
  - Wavelet-based approach.
  - Histogram-based approach.
- Work done in this project.
  - Discrete Cosine Transform (DCT) based approach.
  - Fast Fourier Transform (FFT) based approach.

Processing steps

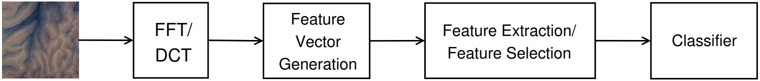
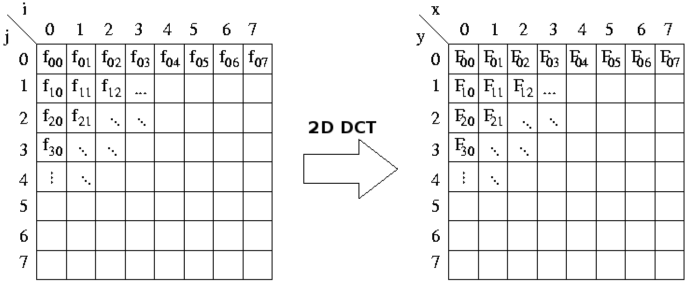


Figure: Processing pipeline

DCT Overview

- decompose image in blocks of size $n * n$
- compute 2D-FDCT on every single block
- $F_{x,y} = \frac{2 \cdot C(x) \cdot C(y)}{N} \cdot \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f_{i,j} \cdot cos(\frac{(2i+1) \cdot x \cdot \pi}{2 * N}) \cdot cos(\frac{(2j+1) \cdot y \cdot \pi}{2 \cdot N})$
- $f_{i,j}$:= pixel i,j of the $n \times n$ input block
- $F_{x,y}$:= the x,y DCT coefficient of the $n \times n$ DCT coefficient matrix
- $C(x)$ and $C(y)$ are constants
- $C(n) \begin{cases} \frac{1}{\sqrt{2}} & \text{if } n = 0 \\ 1 & \text{if } n \neq 0 \end{cases}$

DCT Overview



Bildpunkte

DCT–Koeffizienten

- F_{00} lowest frequency
- F_{nn} highest frequency

DCT Overview

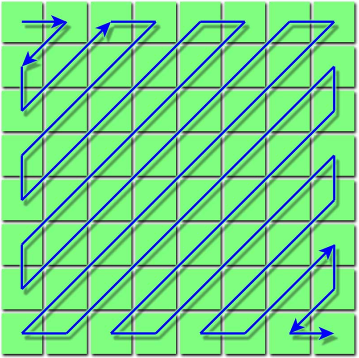


Figure: lowest frequency to highest frequency

DCT and Pitpat

- image size $256 * 256$ pixels
- blocksize $8 * 8$ pixels
- $\Rightarrow 32 * 32$ blocks
- $\Rightarrow$ 64 DCT coefficients for a block
- $\Rightarrow$ 65536 DCT coefficients altogether

calculate global information:

- calculate arithmetic mean over DCT matrices
- $\Rightarrow$ 64 DCT coefficients

DCT Experiments

- different blocksizes ($2^n * 2^n, n = 1, 2, \ldots$)
- other statistic tools (standard deviation, variance, . . .)
- color spaces:
  - YUV luminance channel
  - RGB red, green, blue channel
  - RGB all channels (3 result matrices)

Discrete Fourier Transformation (DFT)

Continuous case:

- $\hat{f}(u) = \frac{1}{2\pi} \int\limits_{-\infty}^{\infty} f(x) e^{-2\pi iux} dx$
- $f(x) = \int\limits_{-\infty}^{\infty} \hat{f}(u) e^{2\pi iux} du$

Discretization:

- $\hat{f}(u) = \frac{1}{N} \sum\limits_{x=0}^{N-1} f(x) e^{-2\pi iux/N}$
- $f(x) = \sum\limits_{x=0}^{N-1} \hat{f}(u) e^{2\pi iux/N}$

Discrete Fourier Transformation (DFT)

2D case:

- $\hat{f}(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-2\pi i(ux/M+vy/N)}$
- $f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \hat{f}(u,v) e^{2\pi i(ux/M+vy/N)}$

Separability:

- $\hat{f}(u,v) = \frac{1}{M} \sum_{x=0}^{M-1} \left(\frac{1}{N} \sum_{y=0}^{N-1} f(x,y) e^{-2\pi ivy/N} \right) e^{-2\pi iux/M}$

Problem:

- High complexity $O(N^2)$
- $\Rightarrow$ FFT

Fast Fourier Transformation (FFT)

- $O(N \log N)$
- Cooley-Tukey (1965)
- Divide and conquer algorithm (Radix - 2)
- Divide the transform into two pieces of size $N/2$ at each step
- $\hat{f}(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)e^{-2\pi iux/N} =$
  $\frac{1}{N}\sum_{x=0}^{N/2-1}\left(f(2x)e^{-2\pi iu\cdot 2x/N} + f(2x+1)e^{-2\pi iu\cdot(2x+1)/N}\right)$
- Implementation: Technische Universität München, Fakultät für Informatik

Sample Images of 2D-FFT

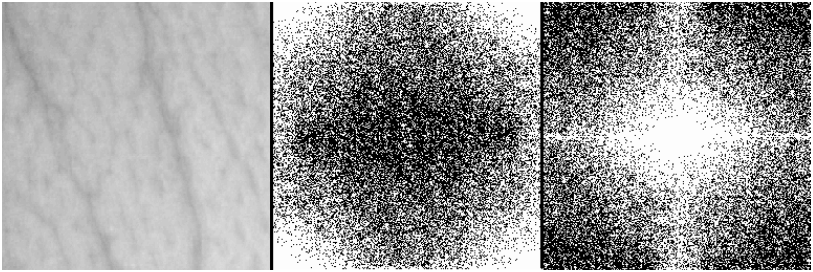


Figure: Sample of Class I with the Fourier-Transformed

Feature Generation

Topologies

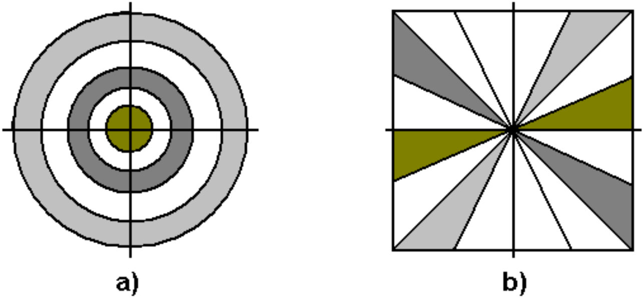


Figure: Partitioning of Fourier Spectrum (a) ring filter; (b) wedge filter

FFT Experiments

- Variable number of rings
- Variable width of rings
- Statistic tools (mean, standard deviation, . . .)
- Color spaces: YUV, RGB, . . .
  - YUV: luminance channel
  - RGB red, green, blue channel
  - RGB all channels

Pattern Classification

- A wide variety of classification approaches exists.
- Statistical pattern classification has been used in this project.
- Classifier must be trained before being ready to use.
- The classifier's input is a feature vector extracted by FFT/DCT.
- Feature vectors are assigned to one of the classes provided during the training phase.

Statistical Pattern Classification

- For each class, use some probability density function.
- Assign a pattern to the class for which it yields the maximal density.

Statistical Pattern Classification (cont'd)

- Parametric approach
  - Select a statistical distribution (e.g. Gaussian).
  - Use the training set to adjust the distribution's parameters (e.g. mean, covariance matrix).
- Non-Parametric approach
  - Use the training set to estimate a class' density function.
- A parametric (Gaussian) approach has been used in this project.

Parameter Estimation - Maximum Likelihood Estimation

- Separate training set into corresponding classes.
- For every class C calculate mean μ_C and covariance matrix Σ_C.

$$
\begin{aligned}
\mu_C &= \frac{1}{|C|} \sum_{x \in C} x \\
\Sigma_C &= \frac{1}{|C|} \sum_{x \in C} (x - \mu_C)(x - \mu_C)^T
\end{aligned}
$$

- Assume C's distribution is $N(\mu_C, \Sigma_C)$

Bayes Normal Classifier

- Assume that class affiliation is a Gaussian distribution.
- Properties
  - Good results for small training sets.
  - Simple and fast classifier.
- Linear decision boundaries
  - Use the whole training set to estimate a single covariance matrix.
  - This covariance matrix is used as a parameter for every class' probability density function.
- Quadratic decision boundaries
  - For every class a separate covariance matrix is generated.

Statistical Pattern Classification

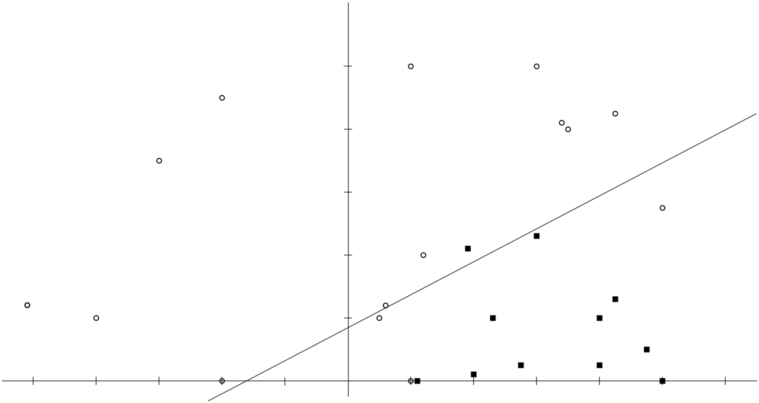


Figure: Linear decision boundary

Statistical Pattern Classification

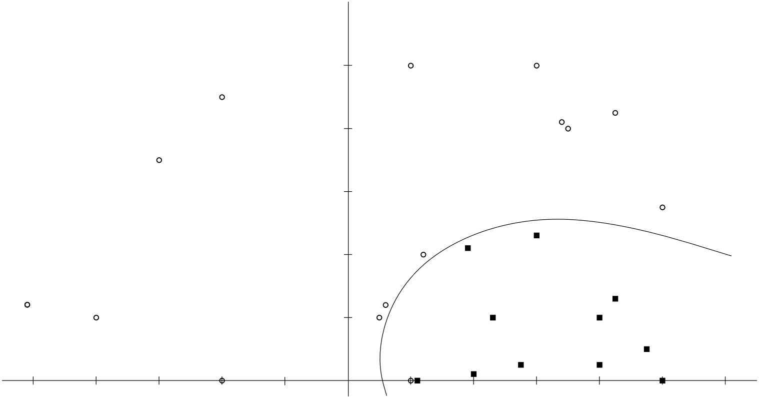


Figure: Quadratic decision boundary

Feature Extraction

- Large feature vectors cause severe performance penalties.
- A lot of data is irrelevant for classification.
- Larger feature vectors may degrade the classifier's performance.
    - "Peaking Phenomenon"

Peaking Phenomenon

- Intuitively, increasing the number of features should increase the classifier's performance.
- Increasing the number of features often degrades the performance of parametric classifiers in practice.
- Parametric classifiers rely on accurate estimates of a class' mean and covariance matrix.
- Increasing the size of feature vectors decreases the quality of these estimates.
- Number of training samples per class n should be at least ten times the size of a feature vector d.

$$\frac{n}{d} > 10$$

Feature Selection (cont'd)

- Create a subset of relevant features.
- Do not transform feature space, but use original features.
- Optimize according to Fisher's Criterion
  - Keep scatter within each class small.
  - Let scatter between different classes be high.
- For C classes, the dimensionality must not be reduced below $C - 1$.

Feature Extraction (cont'd)

- Branch-and-Bound search.
- Guaranteed to find optimal solution.
- Reasonable performance for dropping just a few features.
- Bad performance for selecting very small subsets.
- Exponential blowup in worst case.

Classifier Performance

- Separate pattern set into a *training set* and a *test set*.
- Use the training set to adjust the classification algorithm's parameter.
- Use the test set to analyze the classifier's performance (i.e. get the rate of patterns that have been classified correctly).
- Leave-one-out method has been used in this project.
  - For every pattern x in the pattern set P, use $\{x\}$ as the test set and all other patterns $P \setminus \{x\}$ as the training set.

Results - 2 classes

Channel	Blocksize	Vectorsize	Correctly classified
Y	4	10	64,9%
R	4		65,3%
G	4		64,5%
B	4		60,6%
R	4	8	67,0%
RGB	2		70,4%

Results - 2 classes

Channel	Bands	Band widths	Correctly classified
Y	45		83,8%
Y	45 - 7		85,1%
R	40		83,4%
G	43		83,0%
B	41		84,2%
RGB	3 x 14	[1,1,1,2,2,2,8,11, 11,9,6,6,9,6]	95,9%

Results - 6 classes

Channel	Bands	Band widths	Correctly classified
RGB	3 x 5		58,5%
RGB	3 x 6		60,7%
RGB	3 x 7		14,9%
RGB	3 x 6	[1,1,7,10,5,1]	68,4%
RGB	3 x 6	[1,1,5,10,9,2]	80,4%

Change the Color Model

already used:

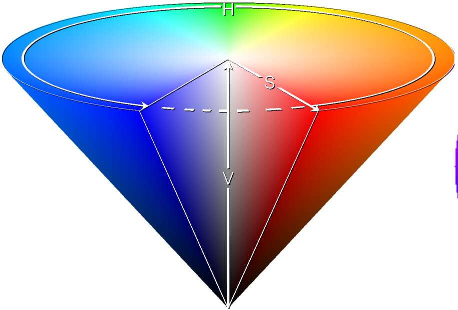
- YUV
- RGB

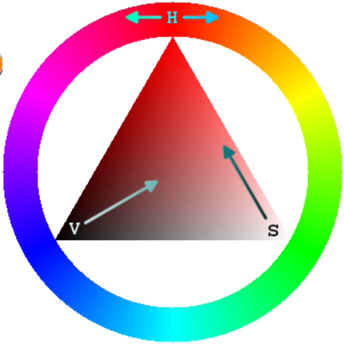
used for experiments:

- HSV
- HLS

HSV Color Model

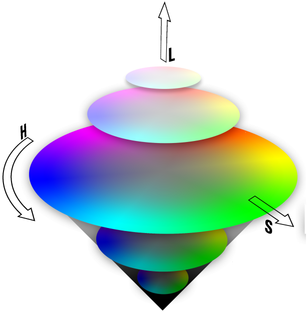
- H - Hue $\varepsilon(0, 360)$
- S - Saturation $\varepsilon(0, 1)$
- V - Value $\varepsilon(0, 1)$ (brightness of the color)





HSL Color Model

- H - Hue $\varepsilon(0, 360)$
- S - Saturation $\varepsilon(0, 1)$
- L - Lightness $\varepsilon(0, 1)$



FFT Result to optimize

- dynamic amount of bands
- dynamic amount of coefficients in a band
- optimize amount of correct classified images (maximization problem)

$\Rightarrow$ use a genetic algorithm

Genetic Algorithm Design

- fitness function: amount of correct classified images
- chromosome encoding:
  - bit chromosome
  - fixed length (first bits determine amount of used bands, the following bits the amout of used coefficients in a band)
  - max 63 bands
  - max 63 coefficients in a band
  - $\Rightarrow$ 6 bits header + $6 * 63$ bits for coefficients in a band
  - $\Rightarrow$ 384 bits altogether
- use tournament selection and 2-point crossover to evolve bit chromosome
- mutation rate: $\frac{k}{chromosomelength}$ $k = 1, 2, 3 \ldots$

Memory Problem

hold FFT coefficients in memory to speed up evolution process

- to calculate for every setting the FFT coefficients is very expensive
- hold the data in main memory is not possible - lack of memory :(
- use a database to handle information - too slow
- use distributed computing technology to distribute data