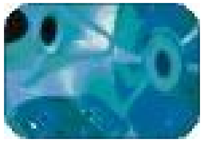
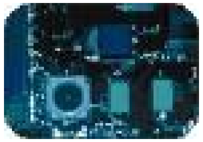
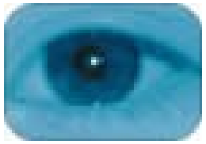




Benchmarking Tools For Fairly Comparing Watermarking Algorithms

Stewart Fraser and Alastair Allen

IASTED SPPRA, Innsbruck, Austria
Feb. 13-15, 2008

Background

- Looking at Digital Image Watermarking
- Many published watermarking algorithms
  - No two algorithms are identical, they can operate in different domains using completely different insertion methods (some with ECC)
- Get algorithms from the web
- Code algorithms myself
- **But which algorithm is the best? How can I <u>measure</u> which algorithm is best?**
- Solution: Benchmarking tools.

Introduction: Watermarking Issues

- Many issues to consider when watermarking
  - Type of host image
  - Length and type of watermark
  - Parameters used (e.g. Embedding strength)
  - Attacks likely to be suffered
- How to compare different watermarking systems?
  - Different parameters (e.g. JPEG quality factor, window size, wavelet levels, embedding strength)
  - For example, embedding strength of 5 may be **strong** in one algorithm and **weak** in another

Summary of the Problem

WM algorithm 1
- spatial domain
- block skip thresh?
- grids (what size?)

WM algorithm 2
- DCT domain
- JPEG value?
- block size?

WM algorithm 3
- wavelet domain
- wavelet levels?
- window size?

How to compare these algorithms fairly?

Blind. Copyright protection. Binary payload. Signal proc. attacks

The Watermarking Algorithms

- Bruyndonckx
  - Spatial domain
  - Non-overlapping 8x8 blocks
  - **Perceptual calculations** performed in blocks to classify pixels into **zones of homogeneous luminance**
  - One watermark bit embedded in the **relationship between mean values** in these zones of homogeneous luminance

- Koch
  - DCT
  - Non-overlapping 8x8 blocks
  - **2 Random DCT coefficients** (mid frequency)
  - Relationship between 2 DCT coeffs altered to embed watermark bit

- Xie
  - Wavelet domain (LL sub-band)
  - Non-overlapping 1x3 window
  - Median of window quantised to embed watermark bit

Adding Error Correcting Codes

UNIVERSITY OF ABERDEEN

Message (no ECC)	Message + ECC → Watermark
Message Generation	Message Generation
	Watermark = ECC + Message (ECC = BCH)
Message Insertion	Watermark Insertion
ATTACK (JPEG)	ATTACK (JPEG)
	Watermark Recovery
Message Recovery	ECC Decode Message
Message Comparison	Message Comparison

The Benchmarking Tools

- **Watson Metric**
  - Uses the Human Visual System (HVS) to rate the quality of a processed digital image compared to the unprocessed original (fair watermark insertion)
- **Normalised Correlation (NC)**
  - Gives a quantitative of measure between the embedded and recovered watermarks (measures watermark similarity)
- **Probability of false alarm calculation (P*fp*)**
  - Computes the probability that an image that was NOT watermarked is flagged as being marked (detector thresholds for different message lengths)
- **Receiver Operating Characteristic (ROC) analysis**
  - Uses a continuously varying threshold value to evaluate the detector performance (measures reliability)

Visual Quality

Diff. Same Diff.

Algorithm	Image	PSNR (dB)	TPE	Embedding strength
Bruyndonckx	Lena	46.75	0.002	7.50
	Fishingboat	46.18	0.002	7.50
	Pentagon	46.83	0.002	7.50
Koch *(JPEG quality setting of 90)*	Lena	43.59	0.002	5.00
	Fishingboat	43.05	0.002	7.50
	Pentagon	42.12	0.002	7.50
Xie *(4 wavelet levels)*	Lena	48.81	0.002	0.10
	Fishingboat	47.29	0.002	0.18
	Pentagon	50.55	0.002	0.25

Table 1: Visual Quality of Images Set Equal Using The Watson Metric

Original

Bruyndonckx

Koch

Xie

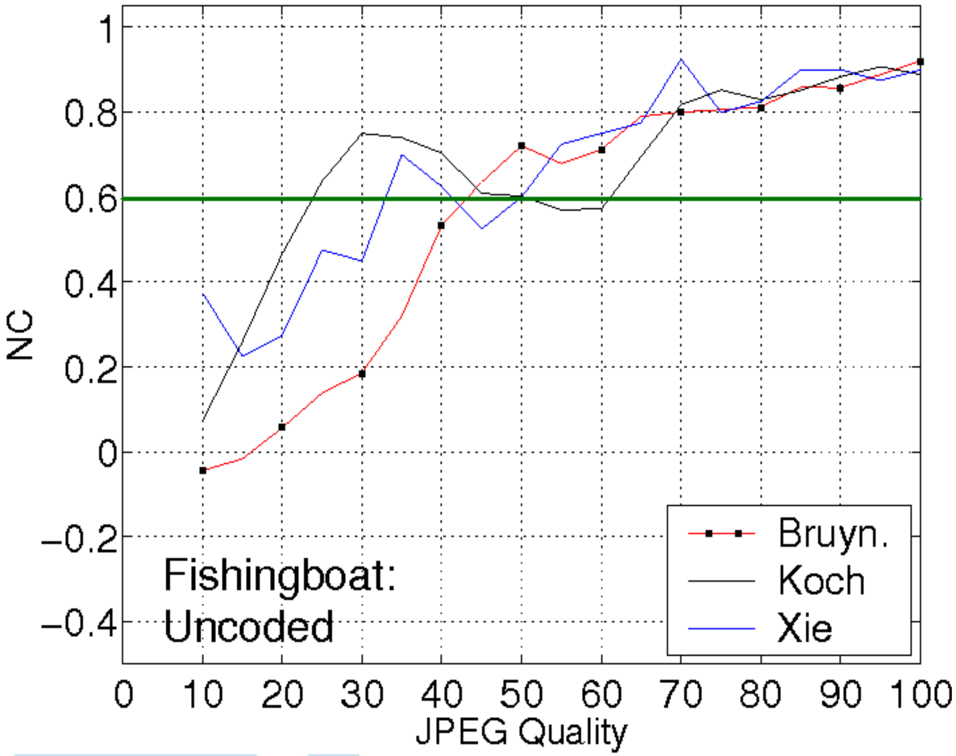
Algorithm	Coding strategy	Detector threshold	P_{fp}
Bruyndonckx	uncoded	0.60	2.5×10^{-8}
	BCH(80,52,9)	0.75	3.5×10^{-8}
	BCH(80,38,13)	0.85	3.3×10^{-8}
	BCH(80,24,19)	1.00	6.0×10^{-8}
Koch	uncoded	0.60	2.5×10^{-8}
	BCH(80,52,9)	0.75	3.5×10^{-8}
	BCH(80,38,13)	0.85	3.3×10^{-8}
	BCH(80,24,19)	1.00	6.0×10^{-8}
Xie	uncoded	0.60	2.5×10^{-8}
	BCH(80,52,9)	0.75	3.5×10^{-8}
	BCH(80,38,13)	0.85	3.3×10^{-8}
	BCH(80,24,19)	1.00	6.0×10^{-8}

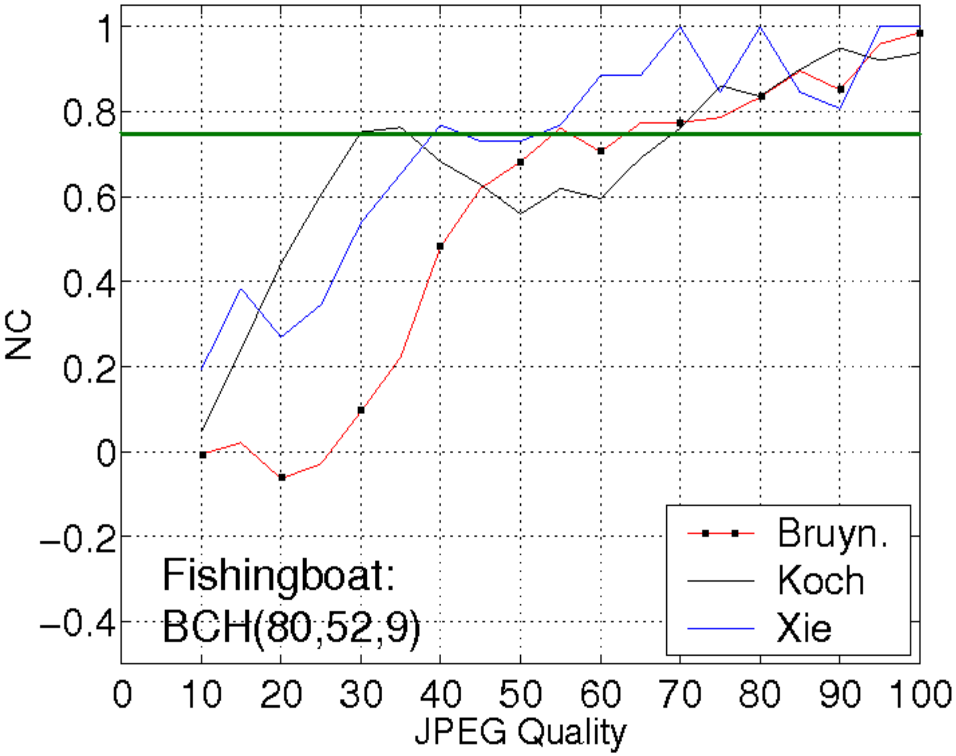
Table 2: Different Detector Thresholds for Different Message Lengths

Different message lengths require different detector thresholds

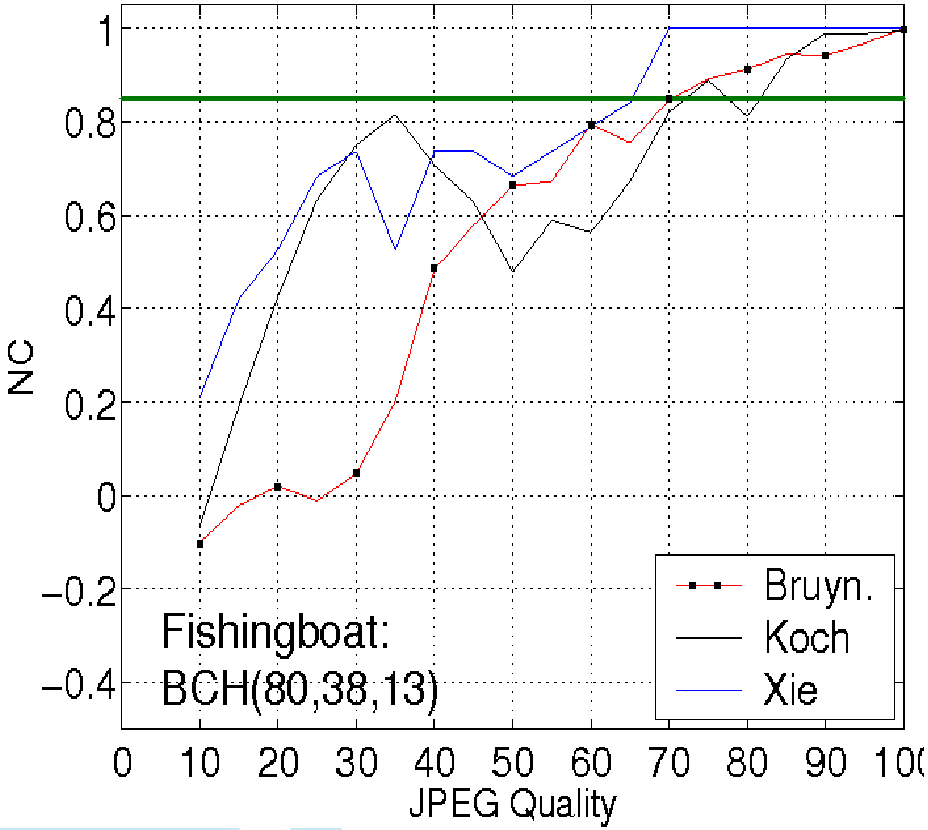
Results

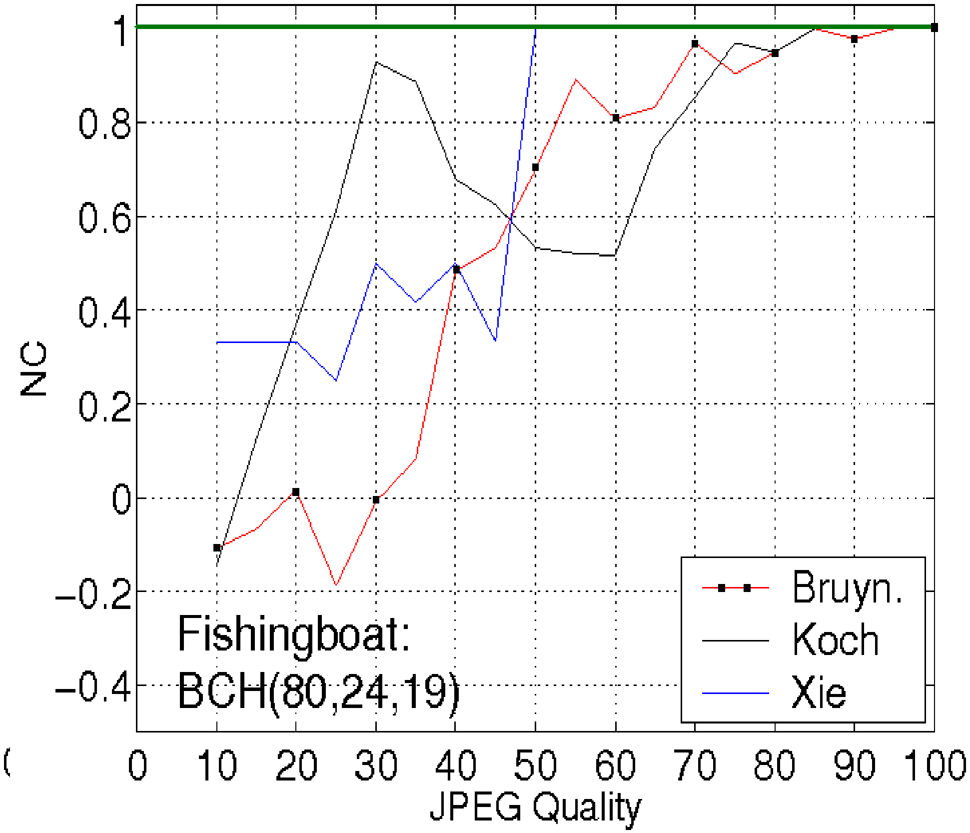
- JPEG attacks from quality 10% to 100% in steps of 5%
- Each JPEG attack run 50 times and averaged
- Different binary watermarks and different seeds each run
- Total of 950 runs for each watermark / ECC combination
- BCH (watermark length, message length, correct errors)





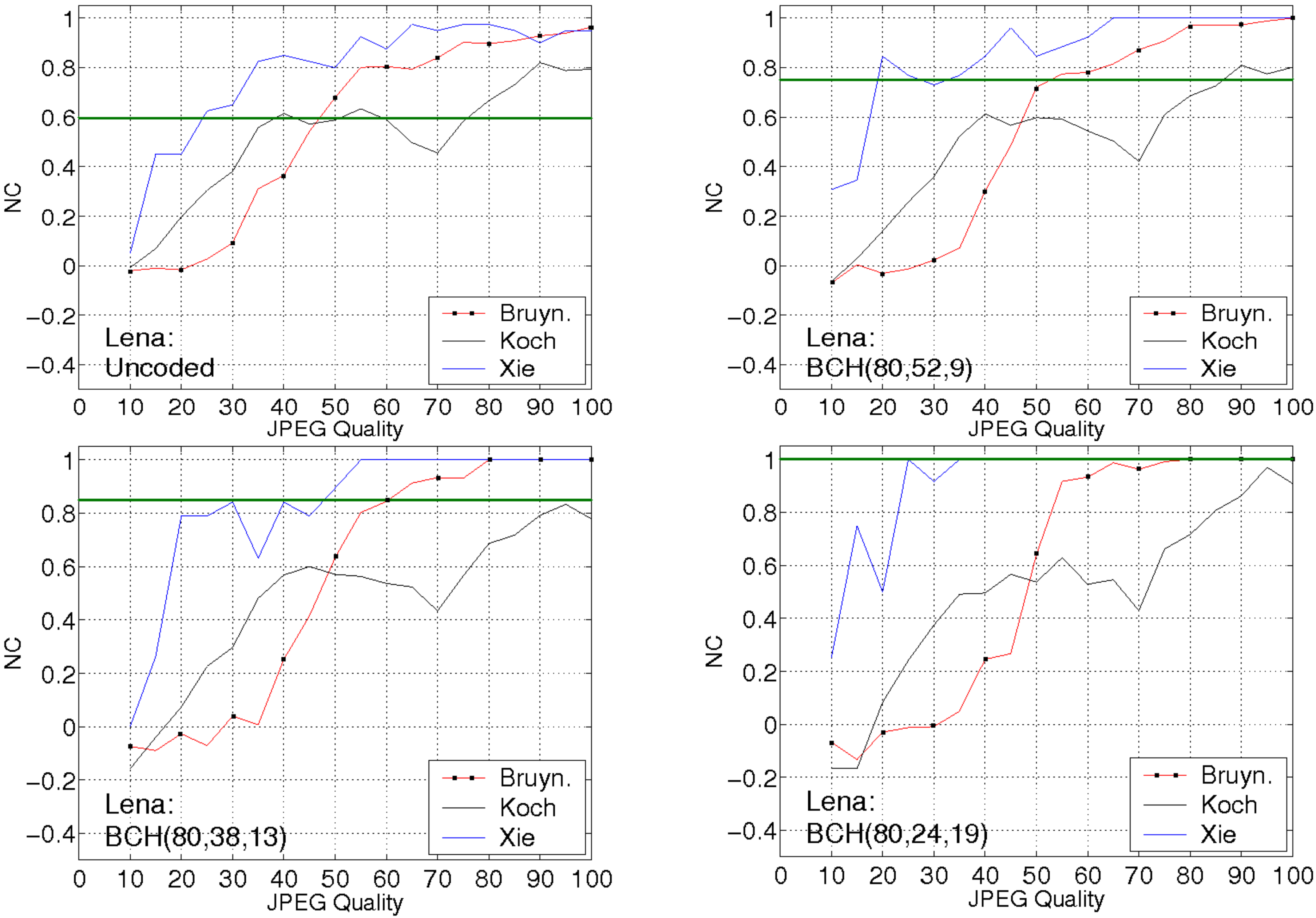
Results



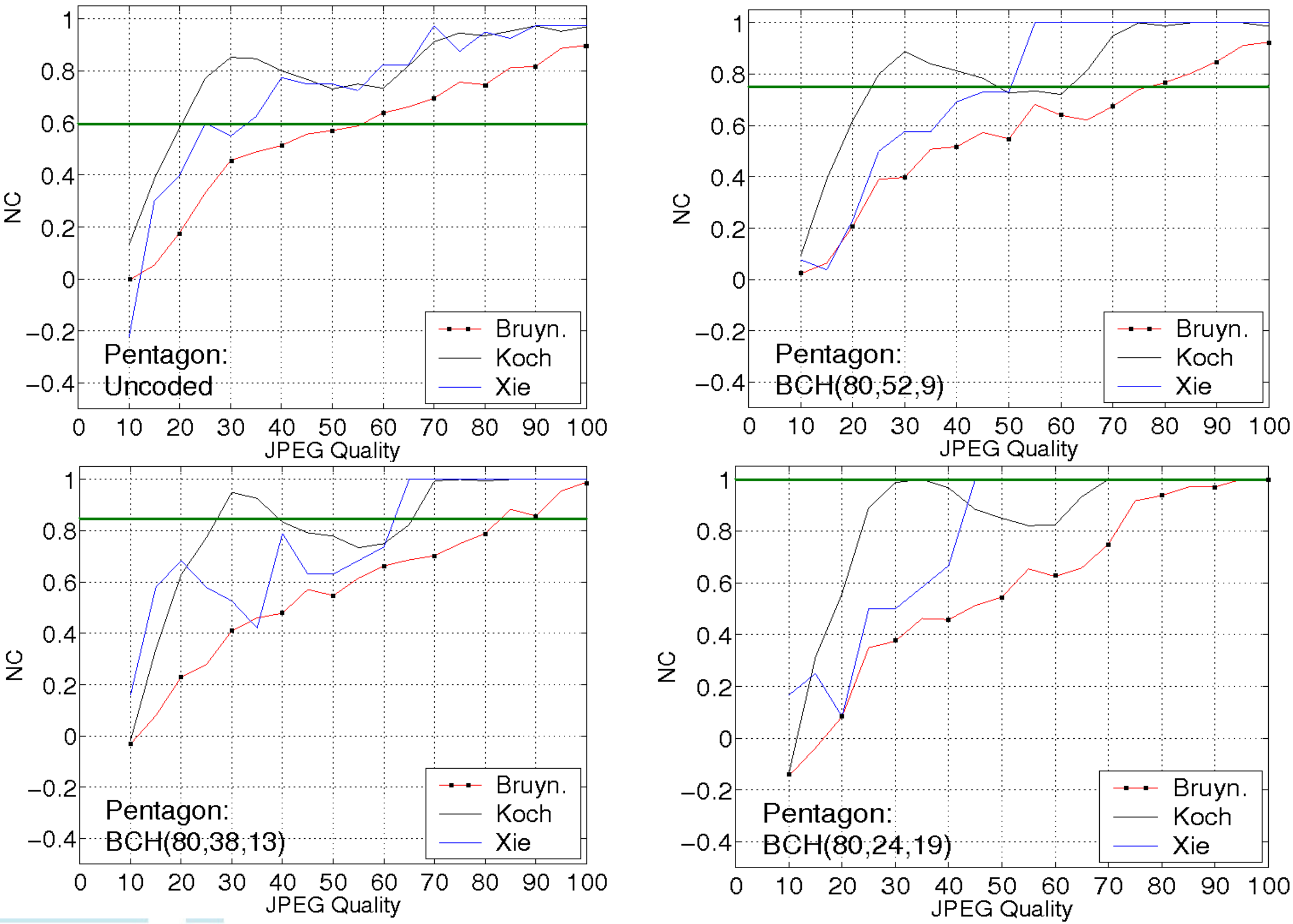


- In general, as more ECC added, **worse robustness**
- Worse robustness, but higher NC values, an **apparent** increase in robustness
- Embedding strength (TPE of 0.002) and image specific!

Results: Lena



Results: Pentagon



ROC curves

Algorithm	Image	ROC Area			
		Uncoded	BCH (80,52,9)	BCH (80,38,13)	BCH (80,24,19)
Bruyn-donckx	Pent.	0.938	0.946	0.923	0.890
	Fish.	0.883	0.845	0.832	0.811
	Lena	0.862	0.832	0.780	0.804
Koch	Pent.	0.995	0.991	0.977	0.962
	Fish.	0.990	0.982	0.973	0.950
	Lena	0.968	0.952	0.931	0.916
Xie	Pent.	0.933	0.913	0.893	0.927
	Fish.	0.949	0.915	0.967	0.929
	Lena	0.908	0.967	0.973	0.958

Table 3: Area Under ROC Curves

All systems reliable as areas under ROC curves closer to 1.0 than to 0.5

Conclusions

- Application of benchmarking tools
  - Watson Metric, NC, P*fp,* ROC
  - Applied to three different images with and without ECCs
  - From graphs, **appeared** to be an increase in robustness
  - But using fair benchmarking tools, it was shown that there was a decrease in robustness
- This work formed part of a bigger project
  - More images, more attacks, more ECCs, bigger messages / watermarks
  - Webpage: www.abdn.ac.uk/~eng565

Caveat

- This work focuses on:
  - **Blind** watermarking for **copyright** protection
    - not tamper proofing nor reversible watermarking
  - **Binary** payloads
    - not 1-bit yes/no watermarks
  - **Signal processing** attacks
    - not geometrical attacks
    - assumes geometrical attacks have been corrected

Visual Quality

Same

Diff

Algorithm	TPE	Embedding strength	Block size	JPEG setting	Wavelet levels
Bruyndonckx	0.006	7	8×8	-	-
Koch	0.006	5	8×8	90	-
Xie	0.006	0.3	1×3	-	3

Table 4: Visual Quality of Lena (320)

WM Length 320

Original

Bruyndonckx

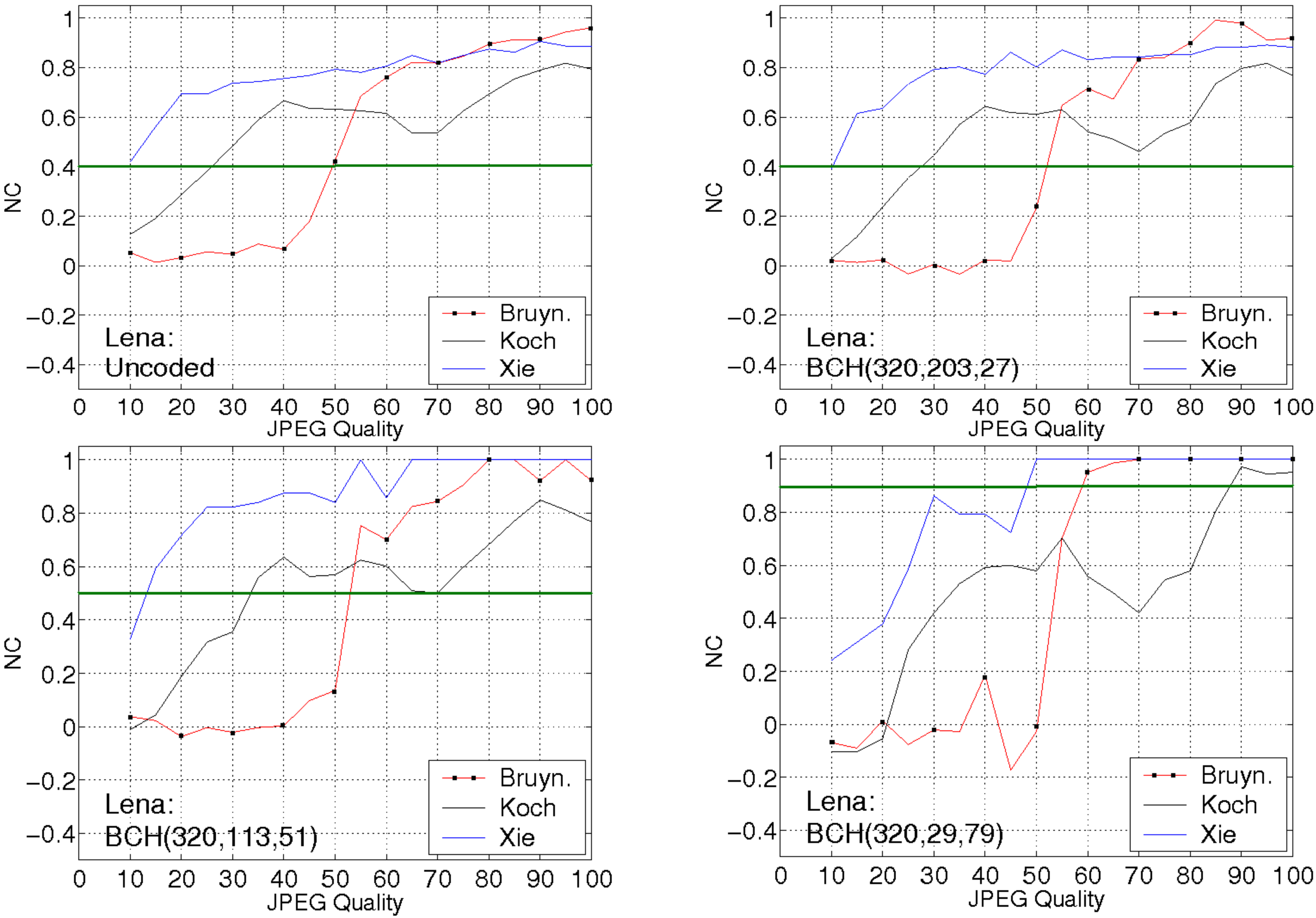
Koch

Xie

Algorithm	Coding strategy	Detector threshold	P_{fp}
Bruyndonckx	uncoded	— 0.40	$< 2.3 \times 10^{-7}$
	BCH(320,203,27)	— 0.40	$< 2.3 \times 10^{-7}$
	BCH(320,113,51)	— 0.50	1.1×10^{-7}
	BCH(320,29,79)	— 0.90	8.1×10^{-7}
Koch	uncoded	— 0.40	$< 2.3 \times 10^{-7}$
	BCH(320,203,27)	— 0.40	$< 2.3 \times 10^{-7}$
	BCH(320,113,51)	— 0.50	1.1×10^{-7}
	BCH(320,29,79)	— 0.90	8.1×10^{-7}
Xie	uncoded	— 0.40	$< 2.3 \times 10^{-7}$
	BCH(320,52,9)	— 0.40	$< 2.3 \times 10^{-7}$
	BCH(320,38,13)	— 0.50	1.1×10^{-7}
	BCH(320,24,79)	— 0.90	8.1×10^{-7}

Table 5: Different Detector Thresholds for Different Message Lengths (320)

Results: Lena (320)



4i2i Communications Ltd. Doig Scott Building, Craibstone Estate, Bucksburn, Aberdeen, AB21 9YA Scotland UK Tel: +44 (0) 1224 712844 Fax: +44 (0) 1224 714413 www.4i2i.com
US Office: 226 Airport Parkway, Suite 470 San Jose, CA 95110 USA Tel: (408) 467 5088 Fax: (408) 436 5524

Coding strategy	Image	Bruyndonckx	Koch	Xie
Uncoded	Lena	0.841	0.971	0.991
BCH(320,203,27)	Lena	0.748	0.925	0.960
BCH(320,113,51)	Lena	0.726	0.897	0.999
BCH(320,29,79)	Lena	0.717	0.864	0.909

Table 6: Area Under ROC curves (320)

Watson Metric

- TPE: Total Perceptual Error.
- HVS: Human Visual System
- Better than pixel based PSNR
- DCT based.
- TPE is a function of:
  - **Contrast sensitivity**
    - Total luminance of display (background + image)
    - Visibility of DCT basis functions as function of luminance
    - Verified via substantial subjective tests
  - **Luminance masking**
    - Visual threshold increases with luminance (increase watermark in bright areas)
  - **Contrast masking**
    - Visibility of one pattern is reduced in the presence of another patter (hide watermark in hetrogenous areas)

Normalised Correlation (NC)

$$\mathrm{NC} = \frac{m^* \cdot m}{||m^*|| \cdot ||m||}$$

m = original watermark
m* = recovered watermark

Convert unipolar vectors $m \in \{0, 1\}$
to bipolar vectors $m \in \{-1, 1\}$

Code snippet

```
corr = 0;
for (i = 0; i < watermarkLength; i++){
    if recoveredWatermark[i] == embeddedWatermark[i]
      corr++;
    else
      corr--;
}
normalisedCorrelation = corr / watermarkLength;
```

Probability of false alarm (P*fp*)

What is the probability of randomly generating a vector that is similar enough to the watermark?

Based on binomial coefficients (Pascal's Triangle):

$$P_{fp} = \sum_{n=\lceil N_w(T+1)/2\rceil}^{N_w} \binom{N_w}{n} 0.5^{N_w} \qquad \binom{N_w}{n} = \frac{N_w!}{n!(N_w-n)!}$$

Nw = message length, T = chosen detector threshold

<u>Code snippet</u>

```
function Pfp = falsePosCalc(T,Nw);
n = floor(Nw*(T+1)/2);
Pfp = 0.0;
for i = n:Nw
    factVal = factorial(Nw) / (factorial(i) * factorial(Nw-i));
    Pfp = Pfp + (factVal * (0.5 ^ Nw));
end; clear i;
```

ROC Curves

- Estimate the influence of threshold selection
- Calculating ROC curves experimentally
  - Feed detector with lots of original images (no watermark). Store results in C0
  - Feed detector with many watermarked images. Store results in C1
  - Chose some threshold values (T) between *min*(C0) and *max*(C1). For each T, count:
    - FPF = C0 > T (False Positive Fraction, P*fp*)
    - TPF = C1 > T (True Positive Fraction, P*p*)
    - Plot TPF (y-axis) against FPF (x-axis)

ROC Curves

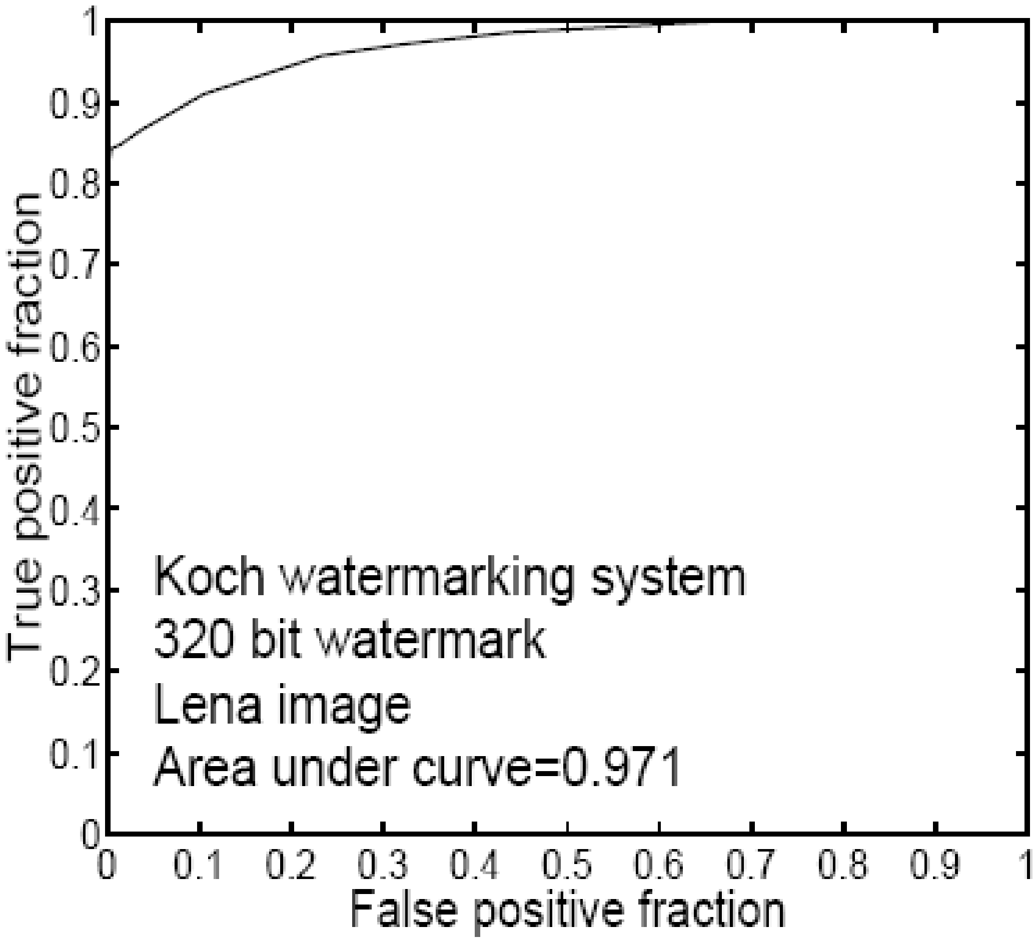
Code snippet

```
TStep=0.025;                 % Threshold step
PfaStore=[]; PpStore=[];     % Initialising empty arrays.
for T=min(C0):TStep:max(C1)  % Choosing threshold values between C0 and C1.
   Pfa=sum(C0>=T)/length(C0);   % FPF
   Pp =sum(C1>=T)/length(C1);   % TPF
   PfaStore=[PfaStore Pfa]; % Storing all Pfa values.
   PpStore =[PpStore Pp];   % Storing all Pp values.
end; clear T;
plot(PfaStore,PpStore);      % Generates ROC graph.
```

ROC Curves

Ideal detector:
Area under curve = 1.0

Random detector:
Area under curve = 0.5



ROC curves

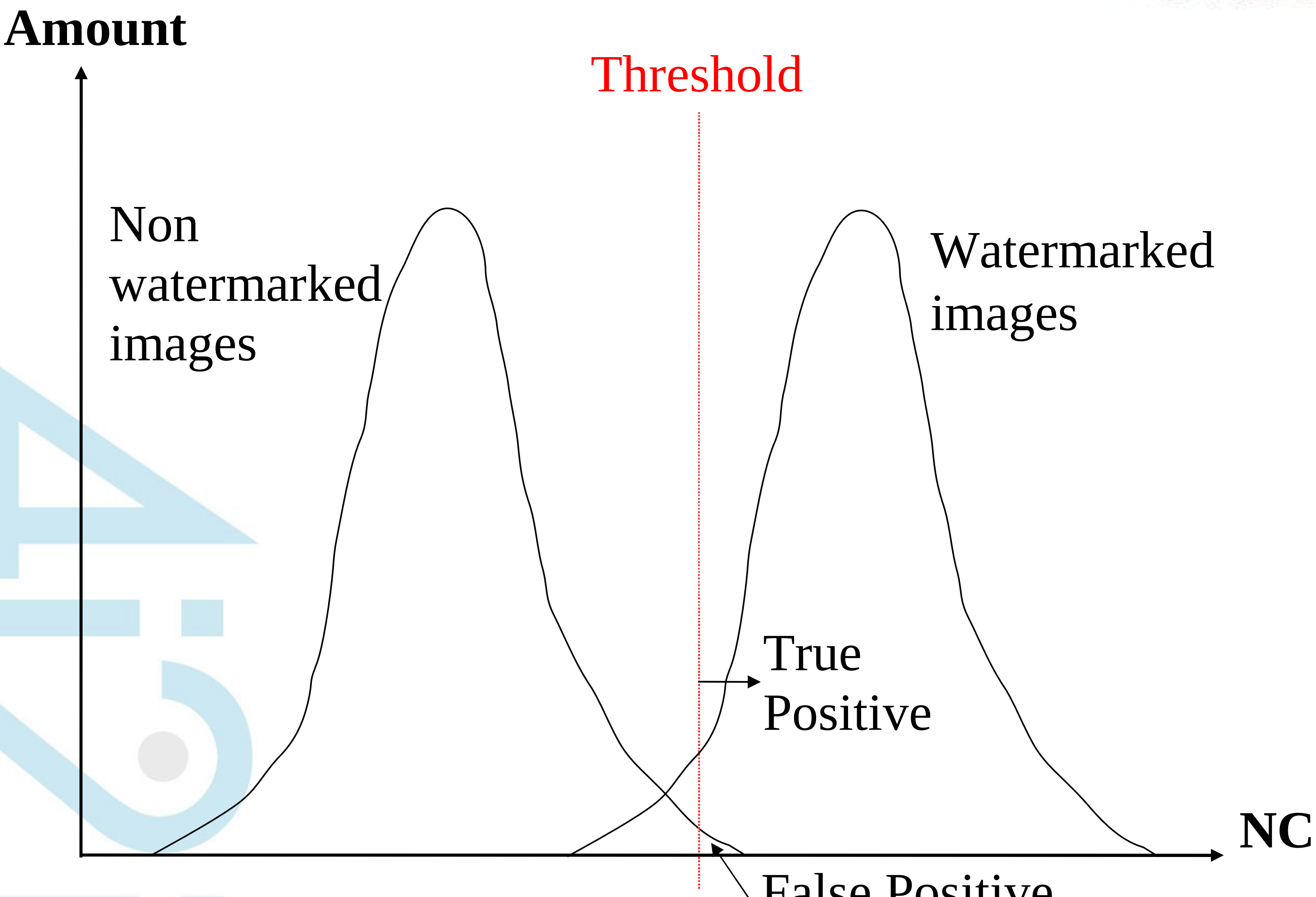

Amount
Threshold
Non
watermarked
images
Watermarked
images
True
Positive
NC
False Positive

Image Specific:
Picture Information Measure (PIM)

$$PIM = \left(\sum_{i=0}^{L-1} h(i)\right) - \max_i[h(i)]$$

L = number of gray levels in a block
h(i) = histogram for grey level i in a block

Image	PIM values	
	Block size 4×4	Block size 8×8
Baboon	27886	31899
Pentagon	23043	28442
Fishingboat	17291	21513
Lena	14365	18848
Peppers	14054	19852

- Measures the complexity of an image
- Non-overlapping 4x4 and 8x8 blocks
- The higher the PIM value, the more heterogeneous an image is

- Bruyndonckx and Xie perform best in smooth images (sub-block mean values and wavelet LL subband)
- Koch performs best in busy images