www.4i2i.com

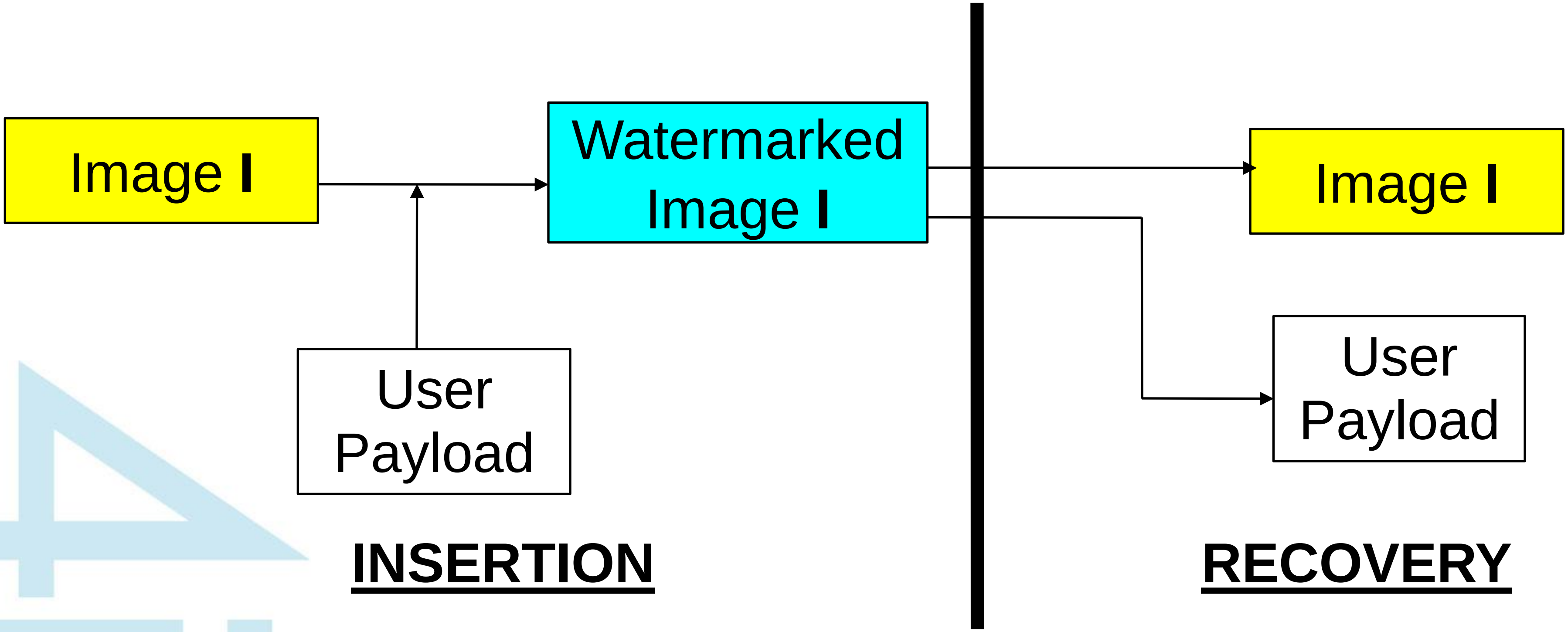
www.abdn.ac.uk

A High Capacity Reversible Watermarking Technique Based On Difference Expansion

Stewart Fraser and Alastair Allen

IASTED SIP, Kailua-Kona, Hawaii, USA
August 18-20, 2008

Reversible Watermarking



- The original image **I** can be recovered EXACTLY from **I'**
- Watermark recovery is a BLIND process
- Military and medical applications

Reversible Watermarking Methods

UNIVERSITY OF ABERDEEN

- Data compression schemes
- Histogram bin exchanging
- Difference expansion

- Novel method here based upon Tian's difference expansion method (with some important adaptations)
- Will overview Tian's method and then describe the adaptations and how they provide an improvement

Tian Method

- Blind watermaking scheme using difference expansion
- Watermark consists of image information and payload

W = | Image Info | User Payload |

- Integer (Haar) transform applied to host image, [1x2] or [2x1] non-overlapping
- Fundamental property:
  - Pixels pairs either changed or expanded
  - Changed pairs provide no extra storage
  - Expanded pairs give one free bit of storage

for a neighboring pair of pixels (x,y) in a grayscale image ...

Forward Integer Transform

$$\text{average } (a) = \left\lfloor \frac{x + y}{2} \right\rfloor$$

$$\text{difference } (h) = x - y$$

Inverse Integer Transform

$$x = a + \left\lfloor \frac{h + 1}{2} \right\rfloor$$

$$y = a - \left\lfloor \frac{h}{2} \right\rfloor$$

... pixel pair averages and **differences** obtained

Watermark Insertion

- A pixel pair difference (**h**) can be either changed, expanded or unaltered

- An altered **h** must still result in an image in range [0,255], this can be determined mathematically: $|h| \le \min(2(255-a), 2a+1)$

Rearrangement of integer transform

- Alterable h values:
  - Preferably, **h** values are expanded
  - If **h** value cannot be expanded, they are changed
- **h** values are left unaltered if they cannot stay in [0,255]

Altering "h" value via Changing

- A pixel pair are changeable if:

$$\left| 2 \left\lfloor \frac{h}{2} \right\rfloor + b \right| \leq \min(2(255-a), 2a+1)$$

- New difference value (h_new) for a changed pixel pair:

$$h\_new = 2 \left\lfloor \frac{h}{2} \right\rfloor + b$$

- Remove LSB and replace with the watermark bit (b)
- Removed LSB must be stored (in vector C) in order to restore original image
- This DOES NOT provide extra storage space

Altering "h" value via Expanding

- A pixel pair are expandable if:

$$|2h + b| \leq \min(2(255-a), 2a+1)$$

- New difference value (h_new) for a expanded pixel pair:

$$h\_new = 2h + b$$

- Left shift all values by one
- Insert watermark bit (b) into newly freed LSB
- This DOES provide one extra bit of storage space
- Note that if a pixel pair are expandable, they are by definition changeable too (used for watermark recovery)

Visual degradation

- In order control payload capacity .v. image quality, user defined threshold (th) introduced to restrict expandable pairs
- Expandable pair:
    - if (h_new ≥ th)
        - pixel pair are changed rather than expanded
- **Detection problem created!**
    - Example: expandable pair where th=10, h=6, b=1
    - h_new = 2h+b = 2(6)+1 = 13
    - At detection, cannot tell if this pixel has been expanded (h=6) or changed (because 13 > th)
    - Solution: Map showing location of all expanded pairs

Location Map

- Shows location of all expanded pairs
- Binary format
- Losslessly compressed JBIG compression

EOS

Watermark Construction

WM = | Comp Loc Map | C bits | Payload |

Stored LSBs from changed **h** values

Binary Location Map of Lena

Watermark Recovery

- Apply integer transform to a watermarked image (get **h'**)
- Find all changeable pairs (expandable are changeable)
  - Collect all LSB from the changeable/expandable pairs

WM = | Comp Loc Map | C bits | Payload |

- Identify the EOS symbol for JBIG compressed location map
  - Decompress the location map
  - Locations of expanded and changed pairs found
- Find the original differences (**h_orig**)
  - Expandable: $h\_orig = \lfloor h'/2 \rfloor$
  - Changeable: $h\_orig = 2\lfloor h'/2 \rfloor + c$

Apply inverse integer transform using **h_orig** to restore original image

Tian Summary

- Integer transform
- Difference values changed (no gain) or expanded (1 extra bit)
- Visual degradation, *threshold* introduced
- *Threshold* causes some expanded pairs to be changed instead
    - Better visual quality, less capacity
- *Threshold* requires position of expanded pairs to be marked (location map)
- User payload inserted and recovery of original image possible

Novel scheme

- Integer transform
- For unaltered and changed pixel pairs, no difference
- Drastic difference for expanded pixel pairs
- Basic premise: companding technique used to increase the amount of pixel pairs that are expanded (rather than changed)
  - More expanded but **no extra** capacity (companding errors)
  - However ...
    - Leads to very sparse location maps
    - Easy to JBIG compress these location maps
    - More space for user payload (higher capacity)
- High capacity user payload inserted and recovery of original image possible

Companding technique

- Companding = Portmanteau of c<u>omp</u>ression followed by exp<u>anding</u>
- Compress (C) and then decompress (D) a signal (x) magnitude $\geq$ threshold (*th*)
  - $C(x) = \mathrm{sgn}(x)\left[\left\lfloor (|x|-th)/2 \right\rfloor + th\right]$
  - $D(x) = \mathrm{sgn}(x)\left[2|x|-th\right]$
- C(x) has the effect of shrinking magnitudes towards *th*
- Errors (*r*) occur when $|x| \geq th$
  - $r = |x| - |D(C(x))|$ where $r \in \{0,1\}$

Companding application

- If difference value (h) between a pixel pair $\geq th$
  - Shrink h towards th via companding compress
  - Calculate and store the companding error (r) in vector R

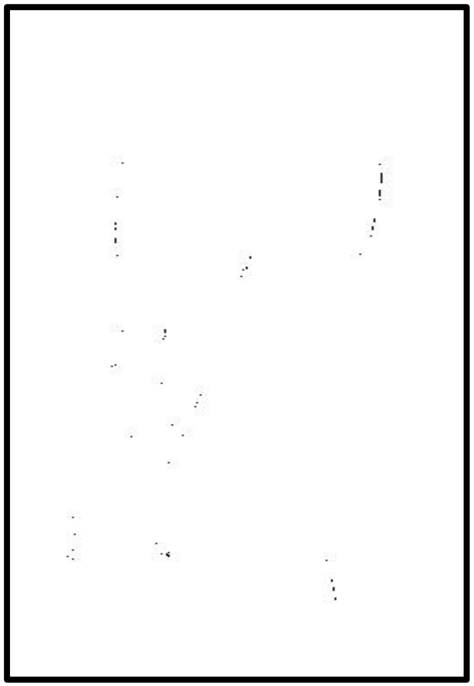
WM =	Comp Loc Map	C bits	R bits	Payload

- h can now be expanded
- MAJOR DIFFERENCE:
  - Tian: if $h \geq th; h$ value changed (store LSB of changed value)
  - Novel: if $h \geq th; h$ value expanded (store companding error)
- Expanding after companding compress does NOT provide higher payload capacity ...

Novel scheme improvement

• ... however, companding allows more difference values to be expanded rather than changed ... **sparse location maps**

Tian



Novel

original compressed

	PBM (bytes)	JBIG (bytes)
Tian	16396	3785
Novel	16396	101

Results for Lena
512 x 512

• Sparse location map easier to compress; more space for payload

Watermark Recovery

- Apply integer transform to a watermarked image (get **h'**)
- Find all changeable pairs (expandable are changeable)
  - Collect all LSB from the changeable/expandable pairs

WM =

Comp Loc Map	C bits	R bits	Payload

- Identify the EOS symbol for JBIG compressed location map
  - Decompress the location map
  - Locations of expanded and changed pairs found
- Find the original differences (**h_orig**)
  - Changeable: $h\_orig = 2\lfloor h'/2 \rfloor + c$
  - Expandable: $h\_orig = \lfloor h'/2 \rfloor$
    - if $h\_orig \geq th$; $h\_orig = \mathrm{sgn}(h\_orig)\,((2|h\_orig|-th+r)$ [Decompress]

Apply inverse integer transform using **h_orig** to restore original image

Results

- Lena test image
- Scanning Laser Ophthalmoscope (SLO) Eye image
- Synthetic Aperture Radar (SAR) image
- Payload capacity measured in Bits Per Pixel (bpp)
  - (**just payload**, not L, C, or R included)
- Visual degradation measured in PSNR (dB)
- Multiple embedding (vertical then horizontal) > 0.5bpp
- Results show that for approximately equal payload bitrates, the visual quality of the novel scheme **outperforms** the Tian scheme consistently across all images
  - Especially so at lower payload bitrates

Results: Test, Medical and Military

Tian: payload bitrate (bpp)	0.15	0.24	0.32	0.39	0.46	0.54	0.67	0.74	0.85
Tian: PSNR (dB)	44.20	42.86	41.55	40.06	37.66	36.15	34.80	33.05	32.54
novel: payload bitrate (bpp)	0.16	0.24	0.37	0.41	0.48	0.57	0.70	0.78	0.87
novel: PSNR (dB)	47.09	45.42	42.45	41.22	40.35	38.84	36.78	35.31	33.68

Results for the Lena test image; grayscale 512×512

Tian: payload bitrate (bpp)	0.23	0.29	0.34	0.47	0.53	0.66	0.79	0.86	0.94
Tian: PSNR (dB)	42.53	41.62	40.94	37.87	35.97	34.89	33.53	32.73	31.65
novel: payload bitrate (bpp)	0.24	0.29	0.36	0.43	0.60	0.66	0.81	0.89	0.94
novel: PSNR (dB)	47.09	45.27	42.93	40.54	39.61	38.48	35.71	33.77	32.06

Results for the Eye medical image; grayscale 256×256

Tian: payload bitrate (bpp)	0.18	0.25	0.33	0.47	0.52	0.64	0.76	0.85	0.90
Tian: PSNR (dB)	36.30	35.29	33.94	29.88	29.50	28.54	27.46	26.58	26.02
novel: payload bitrate (bpp)	0.19	0.25	0.33	0.45	0.54	0.66	0.77	0.85	0.90
novel: PSNR (dB)	40.71	38.74	36.32	34.40	32.87	30.81	28.99	27.41	26.32

Results for the SAR military image; grayscale 256×256

Conclusions

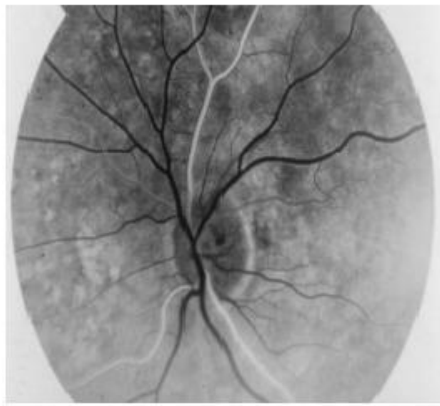
- Novel scheme applies companding technique to Tian scheme
    - results in sparser location maps → more capacity for payload
- Results show that the novel scheme outperforms the Tian scheme
    - for similar PSNRs, the novel scheme returns much higher payload capacities

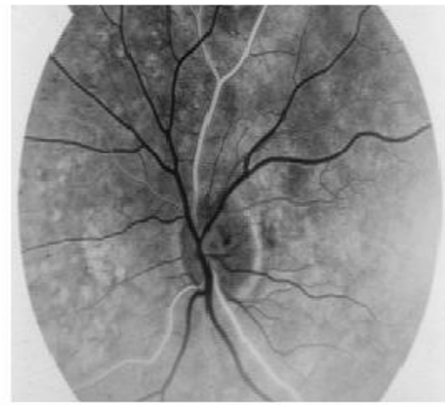
Sample Images

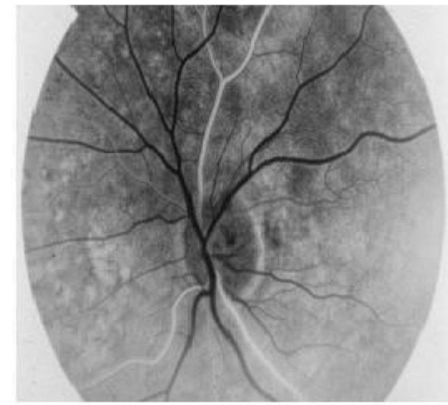

Original


Tian, 0.38bpp, 40.06dB


Novel, 0.48bpp, 40.35dB


Original


Tian, 0.47bpp, 37.87dB


Novel, 0.66bpp, 38.48dB


Original


Tian, 0.18bpp, 36.30dB


Novel, 0.33bpp, 36.32dB