Automatic Contrast Enhancement using Reversible Data Hiding

Presented by
Suah Kim, Rolf Lussi, Xiaochao Qu, Hyoung Joong Kim

Multimedia security lab, Korea University

MULTIMEDIA SECURITY LAB

Introduction

Automatic image enhancement

- When user uploads an image
  - Automatically enhance the image (without user's inputs)
  - With original image recoverability (user is not satisfied with the enhancement)
- "Make them look better without any efforts on user's behalf"
- Two existing solutions
  - Keep the original image
  - Non destructive editing (keep track of the enhancement in an XML format)

Apple

MULTIMEDIA SECURITY LAB

Drawbacks

- Keep the original image
  - **Increase** in **storage requirement** - keeping the original image
- XML based
  - Enhanced image **cannot** be decoded by the standard image decoder
  - Enhancements are not standardized (highly depended on the software and the company)

Goals

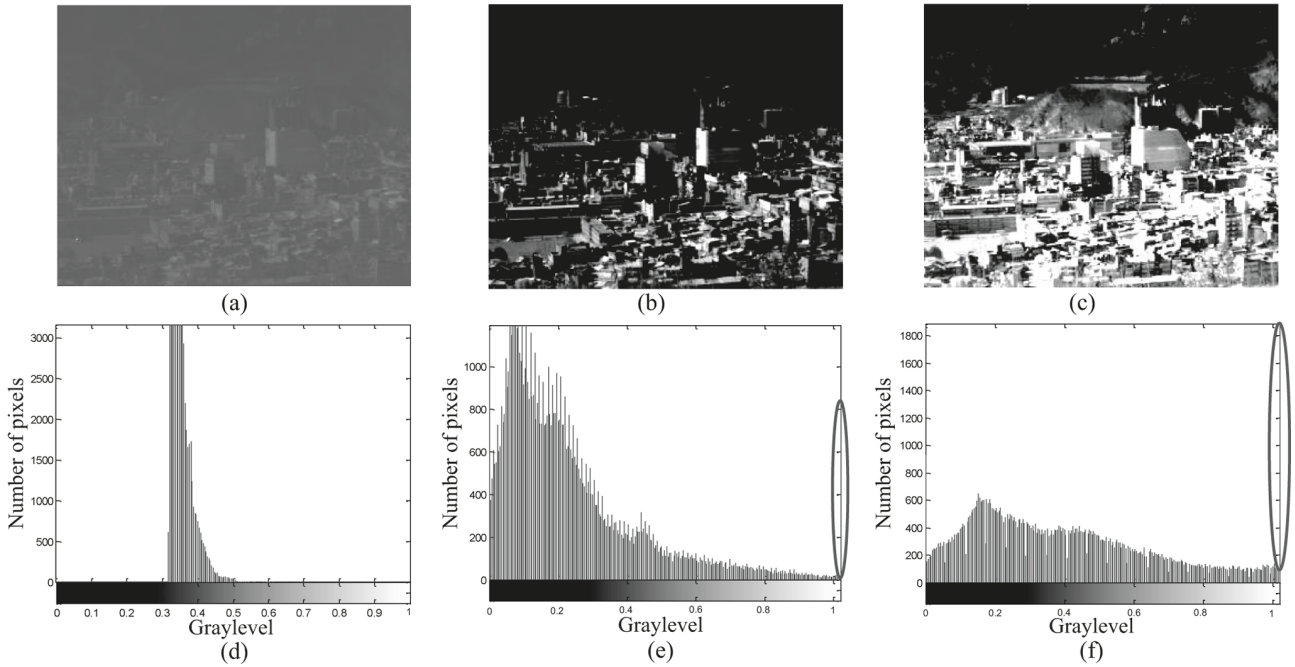
- Find a solution which provides
  - Good contrast enhancement
  - Reduced storage requirements
  - Decodability of the enhanced using the standard decoder (original image recoverability with a special decoder)
  - Data hiding for integrity checking (Embed integrity checking value)

Proposed Method

Contrast enhancement

Contrast enhancement

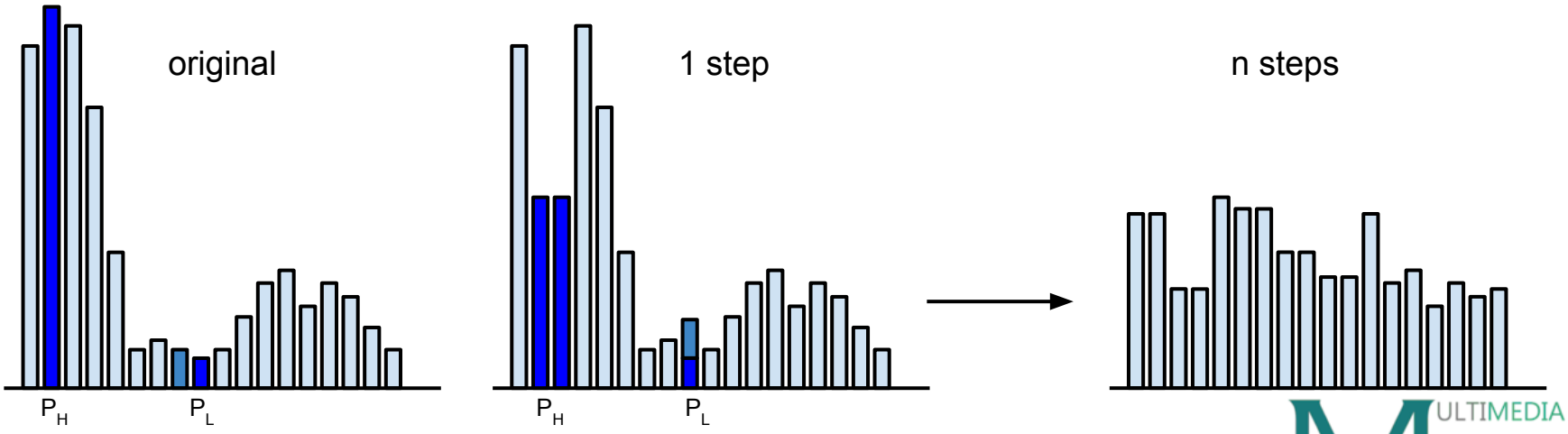
- Useful for enhancing under and over exposed images
- Achieved using histogram equalization



MULTIMEDIA SECURITY LAB

Overview

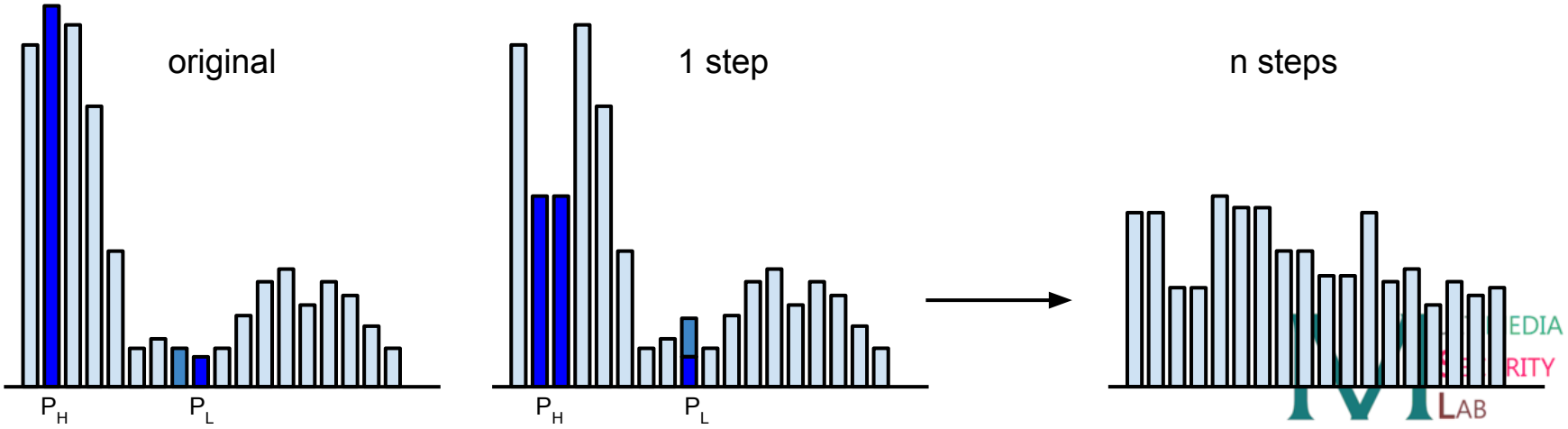
- Histogram Equalization can be achieved using histogram shifting
  - Split the most frequent bin into two bins (iteratively)
- Histogram shifting is a reversible operation (when used with location map)



Unidirectional histogram shifting

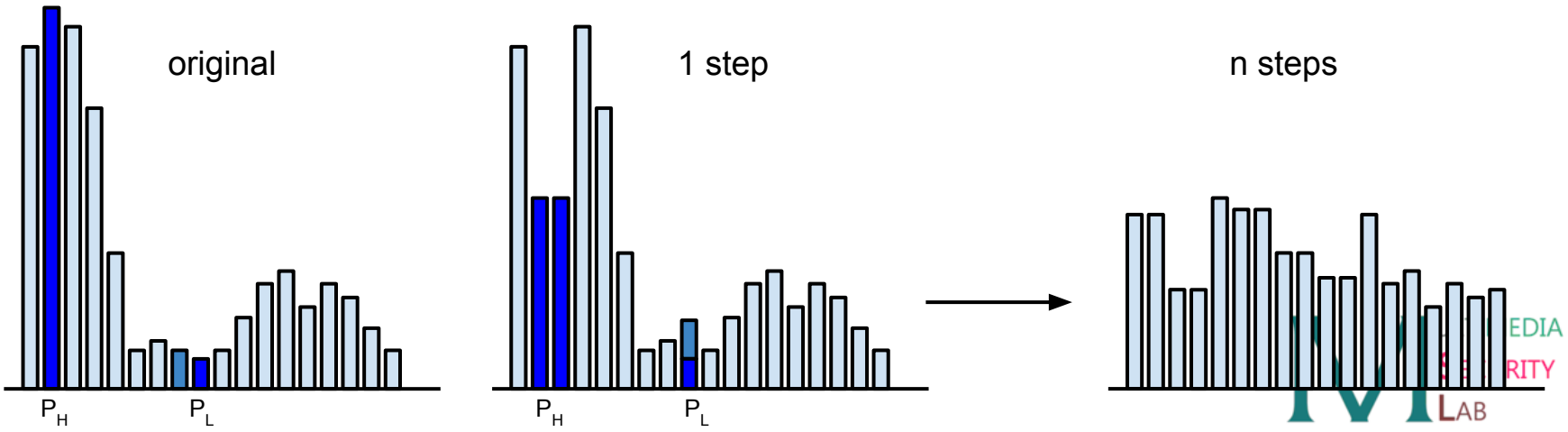
For every iteration of histogram shifting

- Find the most frequent bin P_H
- Find the least frequent bin P_L which is located on the right of P_H
- If $P_H <= P_L$
  - Positive histogram shifting (PHS)
- If $P_H > P_L$
  - Negative histogram shifting (NHS)



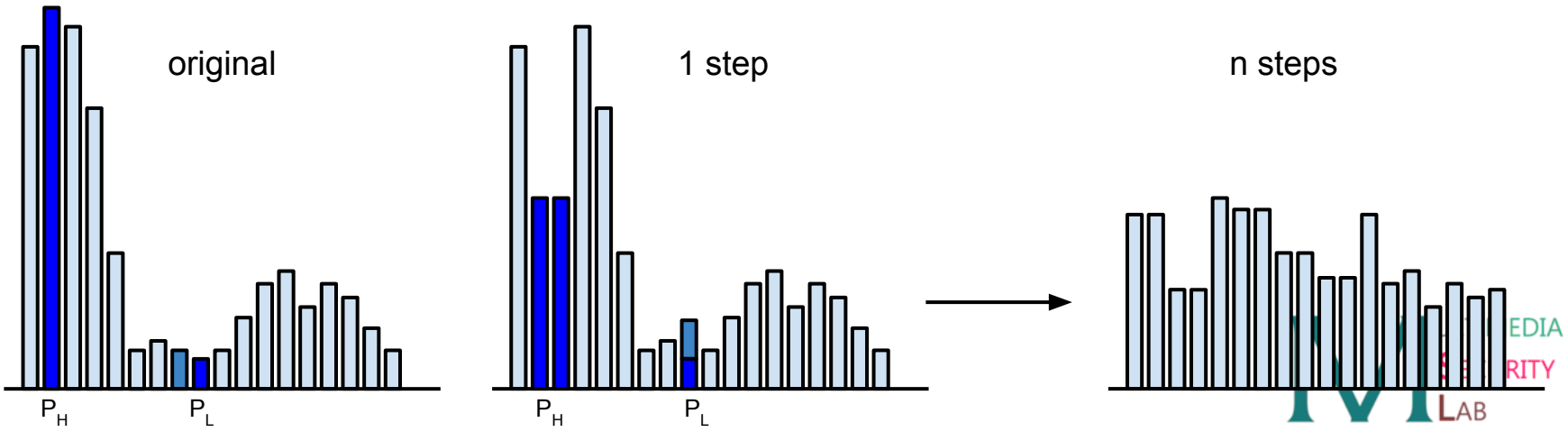
Positive histogram shifting (PHS)

- Combine bin P_L-1 and P_L
- Create an empty bin (all pixels between P_H and P_L is shifted by 1)
- Embed data in bins P_H and P_H+1



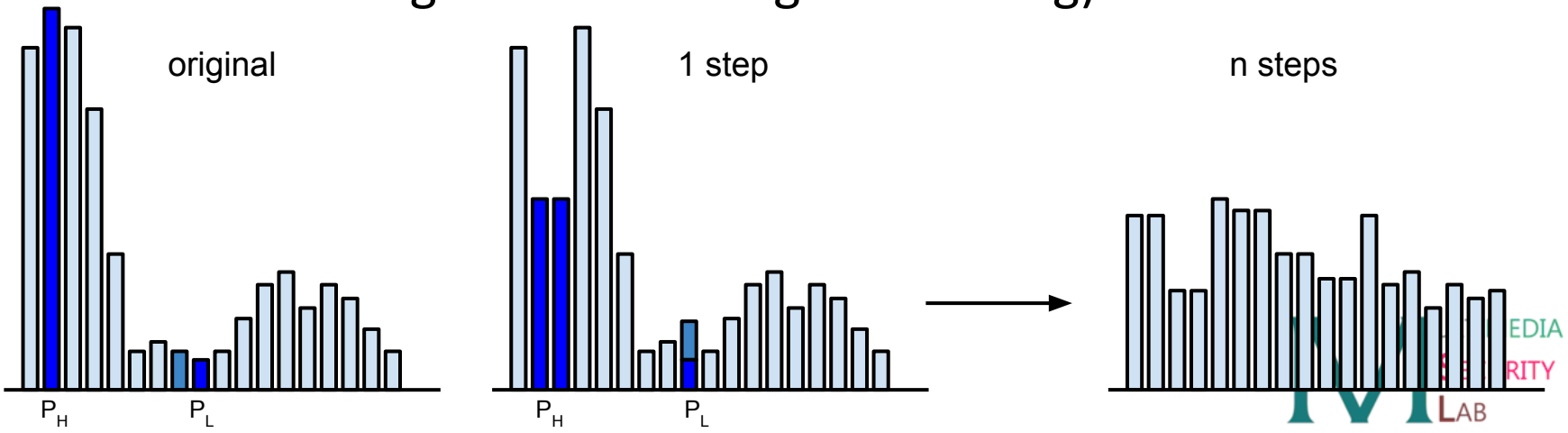
Embedding

- Number of pixels equal to P_H = embedding capacity
- For every pixel values P_H
  - If embedding bit is
    - 0, leave it as P_H
    - 1, modify P_H to P_H+1
- Extraction is trivial
- Negative histogram shifting can be applied in similar fashion (rest of the presentation will explain using PHS)



Reversibility

- What happens when the most frequent bin is split?
  - Pixels are shifted towards the bin P_L
  - Bins P_L and P_L-1 are combined (PHS)
  - Bins P_L and P_L-1 are combined (NHS)
- How to make it reversible?
  - Create a location map
  - Embed side information for reversibility (current P_L and P_L values during the next histogram shifting)



Concurrent location map

- Location map
  - create a separate binary string indicating which P_L pixels are originally P_L-1
- For each combined P_L pixels,
  - If originally P_L then mark 0
  - If originally P_L then mark 1
- Size of location map is equal to the number of P_L and P_L-1 pixels
- (Optionally) compress using arithmetic coding

Stop condition

- Higher number of iterations leads to more equalized histogram => maximize number of iterations
- Stop condition:
  - Location map size > embedding capacity

Side information

- Current P_H and P_L
  - Embedded in the next histogram shifting round
    - In case of the last round, they are recorded using LSB replacement (original pixel values are embedded in the current histogram shifting round)
- Location map
  - Embedded in the current histogram shifting round
- Last flag
  - Indicates whether the current histogram shifting round is the first round
  - Embedded in the current histogram shifting round
- Compression flag

Recoverability of the original image

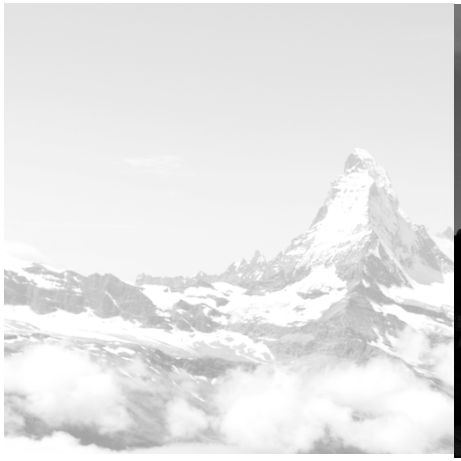
- Read the first 8 LSBs of the image to find P_H and P_L
- Undo histogram shifting
- Repeat histogram shifting until the “last flag” is found

Experimental Results

Test Images

- 4 test images (over and under) and 4 SIPI
- 512 x 512 color images converted into grey scaled image

Visual Evaluation



a) Matterhorn

b) Hanok

c) Creek

d) Plane

- 4 low contrast images are tested
- 2 under-exposed (dark) and 2 over-exposed (light)

MULTIMEDIA SECURITY LAB

Matterhorn

original

16 iterations

MULTIMEDIA SECURITY LAB

Matterhorn

original

32 iterations

Matterhorn

original

48 iterations

MULTIMEDIA SECURITY LAB

Matterhorn

original

64 iterations

Matterhorn

original

80 iterations

Matterhorn

original

96 iterations

Matterhorn

original

112 iterations

Matterhorn

original

120 iterations
(Max)

MULTIMEDIA SECURITY LAB

Plane

a) Original

b) Proposed

c) Histeq

MULTIMEDIA SECURITY LAB

Hanok

a) Original

b) Proposed

c) Histeq

MULTIMEDIA SECURITY LAB

Creek

a) Original

b) Proposed

c) Histeq

MULTIMEDIA SECURITY LAB

Color Extension (sign)

MULTIMEDIA SECURITY LAB

Embedding Capacity

- Lena
  - 130,000 bits
- Boat
  - 180,000 bits
- Barbara
  - 70,000 bits
- Airplane
  - 310,000 bits

Related work

- Wu et al.
  - Limitations:
    - Not automatic (preset number of iterations)
    - Enhancement is very bad for large number of iterations (due to bad implementation of location map)
    - Hard to predict the enhancement effect
    - Unsuitable for automatic contrast enhancement
  - More details can be found in our paper

Applications

- Automatic image enhancement
  - Enhanced image has equalized histogram
  - Original image doesn't have to kept
  - Interoperable with the existing image standard
  - Image integrity information can be embedded within the image

Further works

- Over enhancement can be a problem
- Extending the work to JPEG file format
- Extending the work for other image enhancement techniques

Questions?