Work in LG Electronics

iPAL Group Meeting

September 30, 2011
Where I was


- Research Engineer
- Picture Quality Team
- Advanced Multimedia Processing Group
- Information and Technology Lab.
- Advanced Research Institute
- CTO (Chief Technology Officer) Division
- LG Electronics, Seoul, Korea
What my team did and is doing

Algorithms in Mobile Phone:

- Digital Noise Removal
- High Dynamic Range
- Face Detection
- Face Tracking
- Anti-Shaking
- Auto Scene Detection
- Zero Shutter Lag
- Fast Auto Focus
- Continuous Auto Focus
- Business Card Recognition
- Barrel Distortion Correction
What my team did and is doing

Algorithms in Mobile Phone:

- Digital Noise Removal
- High Dynamic Range
- Face Detection
- Face Tracking
- Anti-Shaking
- Auto Scene Detection
- Zero Shutter Lag
- Fast Auto Focus
- Continuous Auto Focus
- Business Card Recognition
- Barrel Distortion Correction
What my team did is doing(2)

Algorithms in TV and Monitor:

- Superresolution
- Color and Brightness Enhancement
- Monitor Calibration
- Edge Adaptive Image Sharpen
- Deblocking
Representative Solutions

- SmartPic (2007-2008)
- SmartPic2.0 (2008-2009)
- Single-frame Superresolution (2010-2011)
Representative Solutions - SmartPic

- a kind of post image processing
- focused on image enhancement
- application to mobile phone with camera
- algorithms in SmartPic
  - Face Detection
  - Noise Reduction
  - SmartLight
  - Smart Beauty
Representative Solutions - SmartPic2.0

Intelligent One Shot

- Auto Scene Detection + Image Enhancement Algorithm
- providing users with information of current scene and best picture under current condition
- discriminate scene using exposure time, luminance and color information, local histogram, and so on
- Scene Mode: landscape, portrait, backlight, night, indoor (cafe), landscape/portrait, backlight/portrait, night/portrait, indoor/portrait
- algorithms
  - all algorithms in SmartPic
  - auto scene detection
  - Face Tracking
  - Anti Shaking
Single-frame Superresolution

ImageBooster

- Application to monitors and PC

- When watching video clips in Youtube, this algorithm enlarges the size of video you are watching by a factor of 2 for both width and height.

- ImageBooster captures every frame of video the user selected and enhances the frames and displays the result on pop-up window.

- It is software version that is installed in PC.

- This is based on LMS(Least Mean Square) filtering.

- Algorithms
  - Deblocking
  - Image Sharpen
  - Brightness, contrast, and color enhancement
  - LMS Filtering(Interpolation)
ImageBooster : User Scenario

1. Run ImageBooster, then tray icon will be made on the task bar.

2. Open web browser and access the site you can watch streaming video clip like Youtube.

3. Play video clip.

4. Click right button on the tray icon and select run.

5. Move cursor on the video and click left button.

6. Now ImageBooster will enlarge the video and play it on new pop-up window.
Two stages in LMS filtering:

1. Training - Make filter coefficients
2. Filtering - Interpolation
LMS Filtering

Relation between Low-Resolution (LR) and High-Resolution (HR) pixels

\[ \tilde{y}_k = \sum_{i=1}^{9} w_{ki} x_i, \quad k = 1, 2, 3, 4 \]

- we compute 4 green HR pixels with weighted sum of 9 blue LR pixels

- First, we need to calculate weights (coefficients).
LMS Filtering - Training

- make LR image and target HR image
  - degrade original HR image by subsampling, blurring, adding noise, compression, and so on to get LR image.
  - Otherwise, we can make our target HR images from given LR images by useful algorithms with high complexity we already have.

Classify the group of 9 LR pixels.
Assign the class number to each group of 9 LR pixels.
Class number = \[4 \cdot \sum_{i \neq 5} I(x_i) \cdot 2^i + k\]
where \(I(x_i) = 0\) if \(x_i < x_5\), otherwise \(I(x_i) = 1\) and \(k = 0, 1, 2, 3\) which indicates the position of the HR pixel.
In this case, we have \(256 \cdot 4 = 1024\) classes.

Save \(x_i\)'s and \(y_k\)'s in the file or memory assigned to each class respectively. We gathered up to 100,000 data in each class.

Calculate coefficients in each class.
Save Coefficients.
LMS Filtering - Training

1. make LR image and target HR image
   - degrade original HR image by subsampling, blurring, adding noise, compression, and so on to get LR image.
   - Otherwise, we can make our target HR images from given LR images by useful algorithms with high complexity we already have.

2. Classify the group of 9 LR pixels.
   - Assign the class number to each group of 9 LR pixels.
   - Class number = \(4 \cdot \sum_{i \neq 5} I(x_i) \cdot 2^i + k\)
     where \(I(x_i) = 0\) if \(x_i < x_5\), otherwise \(I(x_i) = 1\) and \(k = 0, 1, 2, 3\)
     which indicates the position of the HR pixel.
   - In this case, we have \(256 \cdot 4 = 1024\) classes.
LMS Filtering - Training

1. make LR image and target HR image
   - degrade original HR image by subsampling, blurring, adding noise, compression, and so on to get LR image.
   - Otherwise, we can make our target HR images from given LR images by useful algorithms with high complexity we already have.

2. Classify the group of 9 LR pixels.
   - Assign the class number to each group of 9 LR pixels.
   - Class number = $4 \cdot \sum_{i \neq 5} I(x_i) \cdot 2^i + k$
     where $I(x_i) = 0$ if $x_i < x_5$, otherwise $I(x_i) = 1$ and $k = 0, 1, 2, 3$ which indicates the position of the HR pixel.
   - In this case, we have $256 \cdot 4 = 1024$ classes.

3. Save $x_i'$s and $y_k'$s in the file or memory assigned to each class respectively.
   - We gathered up to 100,000 data in each class.
LMS Filtering - Training

1. make LR image and target HR image
   - degrade original HR image by subsampling, blurring, adding noise, compression, and so on to get LR image.
   - Otherwise, we can make our target HR images from given LR images by useful algorithms with high complexity we already have.

2. Classify the group of 9 LR pixels.
   - Assign the class number to each group of 9 LR pixels.
   - Class number = \( 4 \cdot \sum_{i \neq 5} I(x_i) \cdot 2^i + k \)
     where \( I(x_i) = 0 \) if \( x_i < x_5 \), otherwise \( I(x_i) = 1 \) and \( k = 0, 1, 2, 3 \) which indicates the position of the HR pixel.
   - In this case, we have \( 256 \cdot 4 = 1024 \) classes.

3. Save \( x_i' \)'s and \( y_k' \)'s in the file or memory assigned to each class respectively.
   - We gathered up to 100,000 data in each class.

4. Calculate coefficients in each class.
LMS Filtering - Training

1. make LR image and target HR image
   - degrade original HR image by subsampling, blurring, adding noise, compression, and so on to get LR image.
   - Otherwise, we can make our target HR images from given LR images by useful algorithms with high complexity we already have.

2. Classify the group of 9 LR pixels.
   - Assign the class number to each group of 9 LR pixels.
   - Class number = \( 4 \cdot \sum_{i \neq 5} I(x_i) \cdot 2^i + k \)
     where \( I(x_i) = 0 \) if \( x_i < x_5 \), otherwise \( I(x_i) = 1 \) and \( k = 0, 1, 2, 3 \) which indicates the position of the HR pixel.
   - In this case, we have \( 256 \cdot 4 = 1024 \) classes.

3. Save \( x_i' \)s and \( y_k' \)s in the file or memory assigned to each class respectively.
   - We gathered up to 100,000 data in each class.

4. Calculate coefficients in each class.

5. Save Coefficients.
Calculating coefficient using LMS in each class

\[
X = \begin{bmatrix}
  x_{11} & x_{12} & \cdots & x_{1n} \\
  x_{21} & x_{22} & \cdots & x_{2n} \\
  \vdots & \vdots & \ddots & \vdots \\
  x_{m1} & x_{m2} & \cdots & x_{mn}
\end{bmatrix}, \quad w = \begin{bmatrix}
  w_1 \\
  w_2 \\
  \vdots \\
  w_n
\end{bmatrix}
\]

\[
\tilde{y} = \begin{bmatrix}
  \tilde{y}_1 \\
  \tilde{y}_2 \\
  \vdots \\
  \tilde{y}_m
\end{bmatrix}, \quad y = \begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_m
\end{bmatrix}, \quad e = \begin{bmatrix}
  e_1 \\
  e_2 \\
  \vdots \\
  e_m
\end{bmatrix} = \begin{bmatrix}
  \tilde{y}_1 - y_1 \\
  \tilde{y}_2 - y_2 \\
  \vdots \\
  \tilde{y}_m - y_m
\end{bmatrix}
\]

\[
\tilde{y} = Xw, \quad e = \tilde{y} - y = Xw - y
\]

where \( n = 9 \) and \( m = 100,000 \) in this case.
LMS Filtering - Training

- The coefficients $w_i$ are determined by minimizing the following squared error.
  \[ \sum_{i=1}^{m} e_i^2 \]

- Therefore, $e_1 \frac{\partial e_1}{\partial w_i} + e_2 \frac{\partial e_2}{\partial w_i} + \cdots + e_m \frac{\partial e_m}{\partial w_i} = 0$ for $i = 1, 2, \ldots, n$

- From $e = Xw - y$, we know
  \[ \frac{\partial e_i}{\partial w_1} = x_{i1}, \quad \frac{\partial e_i}{\partial w_2} = x_{i2}, \ldots, \quad \frac{\partial e_i}{\partial w_n} = x_{in} \] for $i = 1, 2, \ldots, m$

- From above two equation,
  \[ \sum_{i=1}^{m} e_i x_{i1} = 0, \sum_{i=1}^{m} e_i x_{i2} = 0, \ldots, \sum_{i=1}^{m} e_i x_{in} = 0 \]
Finally, we obtain the following equations.

\[
\begin{align*}
\left( \sum_{i=1}^{m} x_{i1}x_{i1} \right)w_1 + \left( \sum_{i=1}^{m} x_{i1}x_{i2} \right)w_2 + \cdots + \left( \sum_{i=1}^{m} x_{i1}x_{in} \right)w_n &= \sum_{i=1}^{m} x_{i1}y_i \\
\left( \sum_{i=1}^{m} x_{i2}x_{i1} \right)w_1 + \left( \sum_{i=1}^{m} x_{i2}x_{i2} \right)w_2 + \cdots + \left( \sum_{i=1}^{m} x_{i2}x_{in} \right)w_n &= \sum_{i=1}^{m} x_{i2}y_i \\
\vdots & \quad \vdots \\
\left( \sum_{i=1}^{m} x_{in}x_{i1} \right)w_1 + \left( \sum_{i=1}^{m} x_{in}x_{i2} \right)w_2 + \cdots + \left( \sum_{i=1}^{m} x_{in}x_{in} \right)w_n &= \sum_{i=1}^{m} x_{in}y_i
\end{align*}
\]

We can obtain \( \mathbf{w} \) from these equations.
LMS Filtering - Filtering

LMS Filtering (Interpolation)

- Filtering process is much simpler than training process
  1. Given 9 LR pixels, calculate the class number using the same method as training.
  2. Load the coefficients corresponding to the class number.
  3. Calculate the result HR pixels.

- Since GPU is optimized for parallel computation, we used GPU to implement this filtering in real time.
LMS Filtering

- It is very useful to implement algorithms with high complexity in real time though the result is not same as that of the original algorithm.

- It can be used to any algorithm such as denoising, deblocking, color interpolation, deinterlacing, etc. as well as resolution enhancement.

- Intuitively, the more classes we use, the better performance we can get. For example, if we use $255^9$ classes, it is very near optimal, but it is impossible to implement.

- The performance depends on how well define the class and model the relationship between input and target image under limited condition.