Fast Rehalftoning and Interpolated Halftoning with Flat Low-Frequency Response

Thomas D. Kite

Audio Precision, Beaverton, OR

Brian L. Evans and Alan C. Bovik

Laboratory for Image and Video Engineering

The University of Texas at Austin

http://signal.ece.utexas.edu

Outline

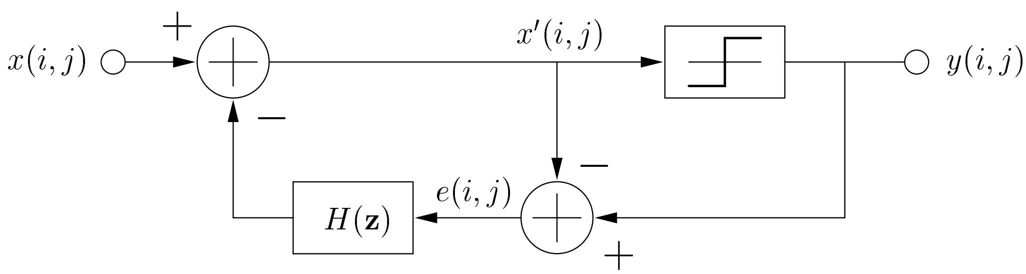
- Introduction to halftoning
- Halftoning by error diffusion
- Rehalftoning
  - Algorithm
  - Results
- Interpolated halftoning
  - Algorithm
  - Results
- Computational requirements
- Conclusion

Introduction to Halftoning

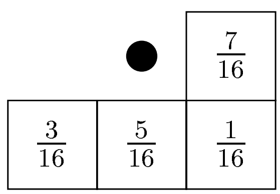
- Word length reduction for images
  - 8-bit to 1-bit for grayscale
  - 24-bit RGB to 8-bit for color displays
  - 24-bit RGB to CMYK for color printers
- Applications
  - Printers
  - Digital copiers
  - Liquid crystal displays
  - Video cards
- Halftoning methods
  - Screening
  - Error diffusion
  - Direct binary search
  - Hybrid schemes
- Consider grayscale error diffusion

Halftoning by Error Diffusion

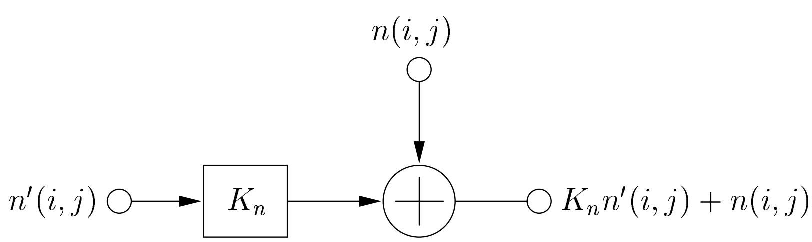
- 2-D delta-sigma modulator
- Noise shaping feedback coder



- Error filter



- Model by linearizing quantizer

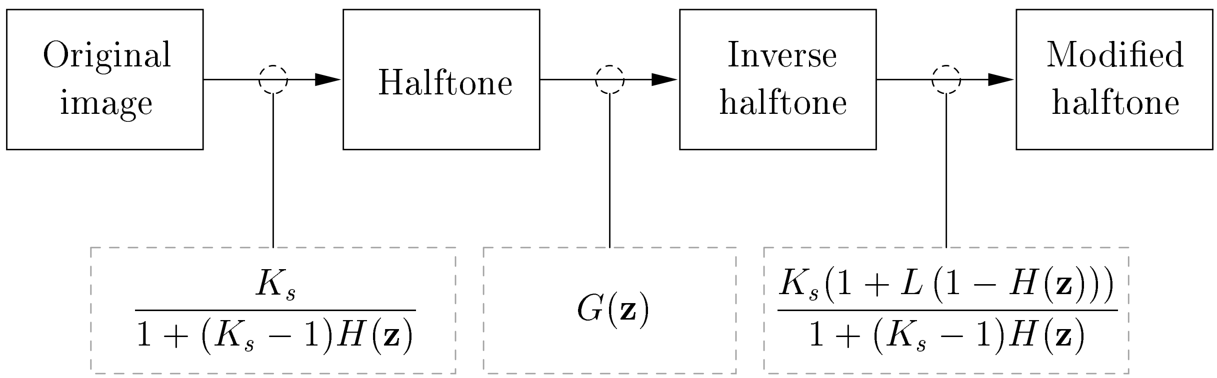


- $K_n = 1$ for error diffusion
- $K_s \approx 2$ for Floyd-Steinberg filter

Rehalftoning

- Different halftoning methods in use
  - Laser printer: ordered screening
  - Inkjet printer: error diffusion
- Optimal rendering of halftone for a particular device
  - Inverse halftone to grayscale image
  - Rehalftone grayscale image
- Inverse halftoning is expensive
  - Use simple inverse halftoning method
  - Artifacts concealed
  - Response errors corrected in halftoning
- Use modified error diffusion [Eschbach and Knox, 1991]
  - Sharpness control parameter L
- Assume input and output are error diffused halftones

Rehalftoning Algorithm



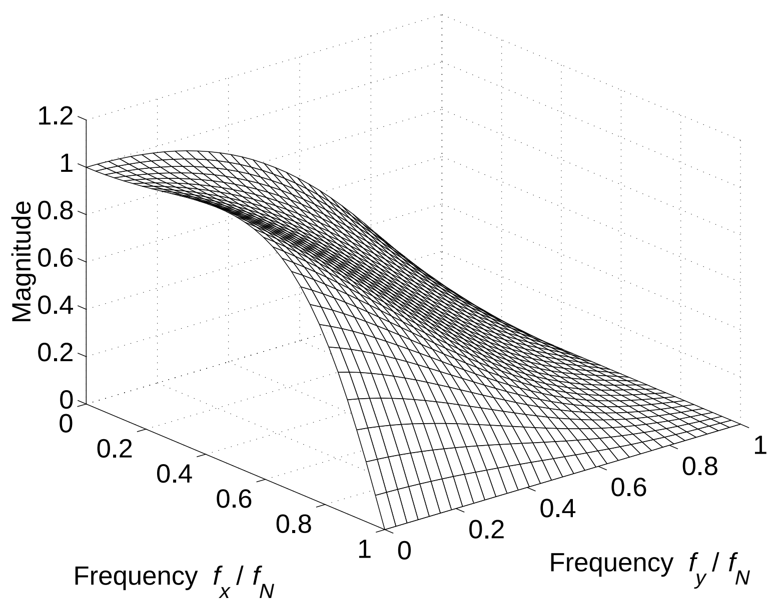
Block diagram of rehalftoning system

- Convert one error diffused halftone into another error diffused halftone
- Inverse halftone with 4×4 filter
  - Symmetric FIR filter
  - Zeros at band edges
  - 6-bit output resolution
- Set sharpness control parameter L to flatten system response at low frequencies
  - $L \approx 0.188$ for Floyd-Steinberg filter

Rehalftoning Results

Original image

Rehalftone



End-to-end signal transfer function

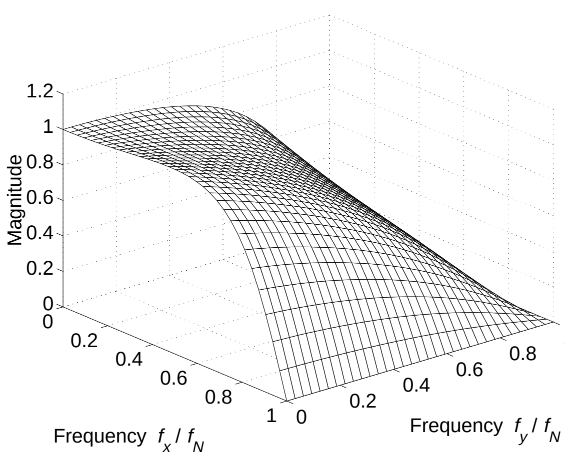
Interpolated Halftoning

- Image resizing
- Methods in increasing complexity
  - Nearest neighbor
  - Bilinear
  - Bicubic, cubic splines, lowpass filtering
- Nearest neighbor, bilinear methods
  - Low computational cost
  - Artifacts masked by quantization noise in halftone
  - Blurring correctable by modified error diffusion
- Examine $\times 2$ interpolation
  - Method applies to any scaling factor
  - Design L for flat transfer function using linear gain model of the quantizer
  - L constant for an interpolation method

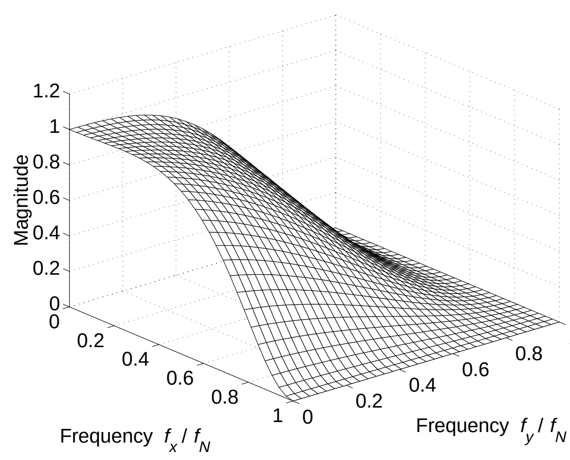
Interpolation Results

Nearest neighbor ×2

Bilinear ×2



Transfer function
$L = -0.0105$



Transfer function
$L = 0.340$

Computational Requirements

- Rehalftoning per pixel
  - 34 increments (++)
  - 12–28 integer additions
  - 4 integer multiplications
  - 2 bit shifts
- Rehalftoning a 512×512 image
  - Computation: 16 million operations
  - Memory usage: 2060 bytes
- $\times 2$ interpolation per pixel
  - 2 increments (++)
  - 9.67 integer additions
  - 4 integer multiplications
  - 3 bit shifts
- $\times 2$ interpolation on 512×512 image
  - Computation: 5 million operations
  - Memory usage: 1024 bytes

Conclusion

- Rehalftoning needed for scanning, processing, and reprinting
- Interpolation needed for resizing in printing and copying
- Developed algorithms for error diffused halftones
  - Rehalftoning & interpolated halftoning
  - Flat low-frequency end-to-end response
- Efficient implementation
  - Local memory
  - Integer arithmetic
  - Suitable for embedded implementation
- Web site for software and papers

http://www.ece.utexas.edu/~bevans/projects/inverseHalftoning.html