Post Acquisition Processing

- De-interlacing
- De-mosaicing
- De-noising
- Geometrical rectification
- White balancing

1 Some slides are from: Jana Kosecka, Yung-Yu Chuang, Alexei Efros, Charles B. Owen

Typical Degradation Sources

- Low Illumination
- Optical distortions (geometric, blurring)
- Sensor distortion (quantization, sampling: spatial+temporal+spectral, sensor noise)
- Atmospheric attenuation (haze, turbulence, …)

Reconstruction as an Inverse Problem

\[
y = Hx + n
\]

- Typically:
  - The distortion \( H \) is singular or ill-posed.
  - The noise \( n \) is unknown, only its statistical properties can be learnt.
Key point: Prior knowledge of Natural Images

The Image Prior

Spatial Sampling

Possible Configurations

CCD

CMOS

3 CCD

Bayer pattern

Foveon X3™
Color filter array

- Color filter arrays (CFAs)/color filter mosaics
- Bayer pattern

Inverse Problem: Image Demosaicing

- The CCD sensor in a digital camera acquires a single color component for each pixel.
- **Problem:** How to interpolate the missing components?

Demosaicing –Types of Solutions

- Using spatial coherence:
  - Non adaptive:
    - Nearest Neighbor interpolation
    - Bilinear interpolation
  - Adaptive:
    - Edge sensing interpolation
- Using color coherence:
  - Non Adaptive
  - Adaptive
**Spatial Coherence:** Neighboring pixels within a band are highly correlated.

- **Nearest neighbor interpolation:** Take the value of a missing pixel from its nearest neighbor’s value
  - Assuming piecewise constant function
  - Problematic in gradient areas and near edges

![Graph showing nearest neighbor interpolation](image1)

- **Linear interpolation 1D:**
  \[ G(k) = \frac{G(k+1) + G(k-1)}{2} \]
  - Assuming piecewise linear function
  - Artifacts near edges

![Graph showing linear interpolation](image2)

- **Bilinear interpolation 2D:**

  \[
  G_{44} = \frac{G_{34} + G_{43} + G_{45} + G_{54}}{4} \\
  R_{44} = \frac{R_{33} + R_{35} + R_{53} + R_{55}}{4}
  \]

![Table showing bilinear interpolation](image3)

- **Mosaic Image**
• Adaptive Interpolation 2D (example 1):

\[
\Delta h = \text{abs}\left[ G_{st} - G_{sh} \right]
\]

\[
\Delta v = \text{abs}\left[ G_{sh} - G_{sv} \right]
\]

\[
G_{st} = \begin{cases} 
\frac{G_{sh} + G_{st}}{2} & \text{if } \Delta h \gg \Delta v \\
\frac{G_{sh} + G_{sv}}{2} & \text{if } \Delta v \gg \Delta h \\
\frac{G_{sh} + G_{sv} + G_{st} + G_{st}}{4} & \text{if } \Delta v = \Delta h
\end{cases}
\]
• Adaptive Interpolation 2D (example 2):

\[
G_u = \frac{w_{u1}G_{u1} + w_{u2}G_{u2} + w_{u3}G_{u3} + w_{u4}G_{u4}}{w_{u1} + w_{u2} + w_{u3} + w_{u4}}
\]

\[
\begin{array}{cccccc}
\ h_1 & g_1 & s_1 & h_2 & g_2 & s_2 \\
\ h_3 & g_3 & s_3 & h_4 & g_4 & s_4 \\
\end{array}
\]

• Outcomes:
  – \(R'(x) = G'(x) = B'(x)\)

  – \(R(x) - G(x) \approx C_{rg}\)

• Spatial coherence: \(G(k) = (G(k-1) + G(k+1))/2\)

• Color coherence: \(C_{rg} = \text{average} |R(x) - G(x)|\)

  \(R(k) = G(k) + C_{rb}\)

Problem: Fails near luminance and chrominance edges

Solution: Adaptive interpolation

Color Coherence: Color bands are highly correlated in high frequencies

• Exploiting color coherence for green interpolation:
  – Since \(R'(x) = G'(x)\) we have

\[
\bar{G}_{k+1} = G_{k+1} + \frac{(R_k - R_{k-1})}{2}
\]

\[
\bar{G}_{k-1} = G_{k-1} + \frac{(R_k - R_{k-1})}{2}
\]

• These estimates for \(G_k\)
  can be combined using an adaptive scheme
Edge Sensing Interpolation using Color Coherence:

\[ \bar{G}_{\text{interp}} = G_{ij} \frac{(B_{ij} - B_{hi})}{2} \]

\[ \bar{G}_{\text{interp}} = G_{ij} \frac{(B_{ij} - B_{hi})}{2} \]

\[ \bar{G}_{\text{interp}} = G_{ij} \frac{(B_{ij} - B_{hi})}{2} \]

\[ \bar{G}_{\text{interp}} = G_{ij} \frac{(B_{ij} - B_{hi})}{2} \]

\[ G_{ij} = \frac{w_{ij}G_{\text{interp}} + w_{ih}G_{\text{interp}} + w_{ji}G_{\text{interp}} + w_{ji}G_{\text{interp}}}{w_{ij} + w_{ih} + w_{ji} + w_{ji}} \]

\[ R_{ij} = \frac{w_{ij}R_{\text{interp}} + w_{ij}R_{\text{interp}} + w_{ij}R_{\text{interp}} + w_{ij}R_{\text{interp}}}{w_{ij} + w_{ih} + w_{ji} + w_{ji}} \]

\[ R_{\text{interp}} = G_{ij} + (R_{ij} - G_{ij}) \]
Adaptive + color coherence

Bilinear
Adaptive

Temporal Sampling

Time Sampling (interlacing)

Interlaced scan
Progressive scan

50 interlaced fields per second
25 frames per second

1/50
1/50
1/25

space
time
Interlaced vs. Progressive Scan

• Interlaced scan is what standard analog TVs use today
• Progressive scan is used by plasma, DLP, LCD screens that emulate film
• NTSC (Americas) and PAL (Europe) use interlaced scan

Resolution & Frame Rates

• Video:
  – NTSC: 720 x 480 @ 30 Hz (interlaced)
  – PAL: 720 x 576 @ 25 Hz (interlaced)
• HDTV:
  – 720p: 1280 x 720 @ 60 Hz
  – 1080i: 1920 x 1080 @ 30 Hz (interlaced)
  – 1080p: 1920 x 1080 @ 60 Hz
• Film:
  – 35mm: ~2000 x ~1500 @ 24 Hz
  – 70mm: ~4000 x ~2000 @ 24 Hz
  – IMAX: ~5000 x ~4000 @ 24-48 Hz
• Note: Hz (Hertz) = frames per second (fps)
• Note: Video standards with an i (such as 1080i) are interlaced, while standards with a p (1080p) are progressive scan

Why do we need interlacing?

• 1930s technology…
  – CRTs for TV had to be scanned at AC line frequency in order to prevent interference
  – CRTs could only scan at around 200 lines in 1/50th of a second
• Twice the refresh rate for a given bandwidth
  – Avoid flickering
• Double spatial resolution for a given bandwidth

Why do we need De-interlacing?

• Translating the interlaced video to progressive devices (plasma, DLP, LCD screens)
• PAL ↔ NTSC format conversion
• Produce still images
Odd and even lines are in different places when there is motion.

**Naïve De-interlacing**

<table>
<thead>
<tr>
<th>Odd field</th>
<th>Even field</th>
<th>Odd + Even</th>
</tr>
</thead>
<tbody>
<tr>
<td>No motion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motion</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Odd and even lines are in different places when there is motion.

**Interlaced Frame**

**Approaches for De-interlacing**

- Spatial interpolation
  - Exploiting spatial coherence
- Temporal interpolation
  - Exploiting temporal coherence
- Spatial-Temporal interpolation
- Motion-compensated interpolation
Spatial Interpolation

- BOB
  - Each field is made into a full frame by line doubling (bilinear)

- ELA
  - Each field is made into a full frame by adaptive spatial interpolation

- Downside:
  - Loss of vertical resolution in a frame
  - Flicker when motion
  - Jagged line (especially in BOB)

\[
f(x, y) = \begin{cases} 
    (f(x-1, y-1) + f(x-1, y+1))/2, & \text{if min}\{a,b,c\} = a, \\
    (f(x+1, y-1) + f(x+1, y+1))/2, & \text{if min}\{a,b,c\} = b, \\
    (f(x, y-1) + f(x, y+1))/2, & \text{otherwise,}
\end{cases}
\]
Temporal Interpolation

- 2-frame field merging => Weave
- 3-frame field averaging
  - fill in the missing odd field by averaging odd fields before and after
- Good when no motion
- Downside: Feathering & Combing effect

Feathering – caused by improper handling of motion
Scene Change

Britney Spears' performance at the MTV Video Music Awards 2001

Spatial-Temporal Interpolation

- Effectively combines the spatial & temporal processing.
- Varying information is present in different regions of a video sequence.
  - In stationary regions, a strong correlation between the current field and the adjacent fields exists. Temporal methods are designed to benefit from this fact.
  - Temporal correlation is absent in region with motion, adjacent pixels from the current field contain more information. Spatial methods use this fact.
  - Image segmentation of each image into moving and stationary regions using sophisticated motion detection algorithms.
Spatial-Temporal Interpolation

\[ \text{Current field} \]

\[ m(S) + (1-m)(T) \]

\( m = \text{motion} \)
\( S = \text{spatial interpol.} \)
\( T = \text{temporal interpol.} \)

Non-Adaptive

Adaptive Spatial Temporal

Motion Compensated De-interlacing

- Calculating the motion of a block in the scene
- Interpolate while considering motion vectors
Noise Distortion

Image Denoising

- Additive noise:
  \[ g = f + n \]

- The noise value is not known but its characteristics are known:
  - Parametric type
  - Parameters (mean, variance, ...)

Examples of independent and identically distributed (i.i.d) Noise

- Gaussian white noise (i.i.d.):
  \[ P(g \mid f) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(g-f)^2}{2\sigma^2}} \]

- Uniform noise:
  \[ P(g \mid f) = \begin{cases} \frac{1}{b-a} & a \leq (g-f) \leq b \\ 0 & \text{otherwise} \end{cases} \]

\( \sigma = 20 \)

\( b-a = 20 \)
• Impulse noise (S & P):

\[
P(g | f) = \begin{cases} 
  P_a & \text{for } g = a \\
  P_b & \text{for } g = b \\
  1 - P_a - P_b & \text{for } g = f
\end{cases}
\]

**Note:** this noise is not additive!

---

**Bayesian denoising**

• Assume the noise component is i.i.d. Gaussian:

\[ g = f + n \]

where \( n \) is distributed \( \sim N(0, \sigma) \)

• A MAP estimate for the original \( f \):

\[
\hat{f} = \arg\max_f P(f | g) = \arg\max_f \frac{P(g | f)P(f)}{P(g)}
\]

• Using Bayes rule this leads to:

\[
\hat{f} = \arg\min_f \left\{ (g - f)^2 + \lambda R(f) \right\}
\]

where \( R(f) \) is a penalty for non probable \( f \).

---

**Example 1: Prior Term**

• Similarity of neighboring pixels

\[
R(f_p) = \sum_{q \in N_p} W(p-q) (f_p - f_q)^2
\]

\( W(p-q) \) is a Gaussian profile giving less weight to distant pixels.

\( (f_p - f_q)^2 \)

---

\[
R(f_p) = \sum_{q \in N_p} W(p-q) (f_p - f_q)^2
\]

• This leads to Gaussian smoothing:

\[
\hat{f}_p = (1 - \alpha) g_p + \alpha \sum_{q \in N_p} W(p-q) g_p
\]

\[
\hat{f}_p = (1 - \alpha) g_p + \alpha \sum_{q \in N_p} W(p-q) g_p
\]

• Reduces noise but blurs out edges.
• The parameter \( \alpha \) depends on the noise variance.
Example 2: Prior Term

- Edge sensitive similarity:

\[ R(f_p) = \sum_{q \in N_p} W(p-q) \log(1 + (f_p - f_q)^2) \]

- This leads to edge-preserving smoothing:

\[ \hat{f}_p = (1-\alpha)g_p + \alpha \frac{\sum_{p \in N_p} W(p-q)W_q(g_p - g_q)g_p}{\sum_{p \in N_p} W(p-q)W_q} \]

- \( W_s \) is a monotonically descending spatial weight
- \( W_l \) is a monotonically descending photometric weight
- A.k.a. Bilateral Filtering
• Gaussian Filtering: Non adaptive

• Bilateral Filtering: Adaptive

• Bilateral Filtering: Adaptive

Noisy Image
Geometric Distortion

- Radial distortion of the image
  - Caused by imperfect lenses
  - Deviations are most noticeable for rays that pass through the edge of the lens
Geometric Transformation

- Operations depend on Pixel’s Coordinates.
- Independent of pixel values.

\[ x \rightarrow f_x(x, y) = x' \]
\[ y \rightarrow f_y(x, y) = y' \]

\[ I(x, y) = I'(f_x(x, y), f_y(x, y)) \]

Forward v.s. Inverse Mapping

Forward mapping:

\[ x \rightarrow f_x(x, y) = x' \]
\[ y \rightarrow f_y(x, y) = y' \]

- Two problems: Holes and overlaps

Example: Translation

\[ x' = f_x(x, y) = x + 3 \]
\[ y' = f_y(x, y) = y - 1 \]

\[ I'(x + 3, y - 1) = I(x, y) \]

Inverse mapping:

\[ x' \rightarrow f_x^{-1}(x', y') = x \]
\[ y' \rightarrow f_y^{-1}(x', y') = y \]
Example:

- Forward mapping

\[ x' = 2x \quad ; \quad y' = y \]

- Inverse mapping

\[ x = x'/2 \quad ; \quad y = y' \]

Intensity Interpolation

- What happens when a mapping function calculates a fractional pixel address?

Nearest Neighbor Interpolation

- Advantage: Fast.
- Disadvantage:
  - Jagged results.
  - Discontinues results.
### Bilinear Interpolation

- The assigned value is an intermediate value between the four nearest pixels.

### Linear Interpolation:

\[ v = \alpha v_e + (1 - \alpha) v_w \]

*where we define* \( \alpha = \frac{x - x_w}{x_e - x_w} \in [0,1] \)

### Bilinear Interpolation:

\[ S = SW \cdot (1 - \Delta x) + SE \cdot \Delta x \]
\[ N = NW \cdot (1 - \Delta x) + NE \cdot \Delta x \]
\[ V = S \cdot (1 - \Delta y) + N \cdot \Delta y \]

\( \Delta x, \Delta y \in [0,1] \)
Parameter Estimation

- Let: $x' = f_x(x, y, a_1, a_2, a_3)$ ; $y' = f_y(x, y, b_1, b_2, b_3)$
- If the mappings are linear in \{a_i\} \{b_i\} the parameters can be estimated using linear regression.

Example: Affine Transformation

- Alternative representation

$$
\begin{bmatrix}
    x' \\
    y'
\end{bmatrix} =
\begin{bmatrix}
    a & b \\
    c & d
\end{bmatrix}
\begin{bmatrix}
    x \\
    y
\end{bmatrix} +
\begin{bmatrix}
    e \\
    f
\end{bmatrix}
$$

- Given k points $(P_1, P_2, .., P_k)$ in 2D that have been transformed to $(P'_1, P'_2, .., P'_k)$ by the affine transformation:
  - How many points uniquely define the transformation?
  - How can we find the transformation?
  - What can be done if points coordinates are in inaccurate?

$$
Ha = b \
\hat{a} = \min_a \|Ha - b\| \
\hat{a} = (H^TH)^{-1}H^Tb
$$