Demosaicing

October 1, 2015

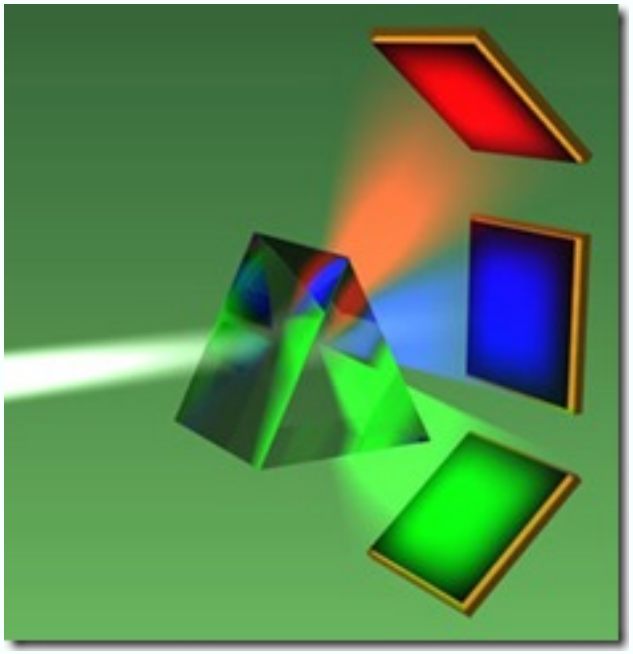
Guillaume Lemaitre

Université de Bourgogne

1. Color imaging

2. Demosaicing

Tri-CCD



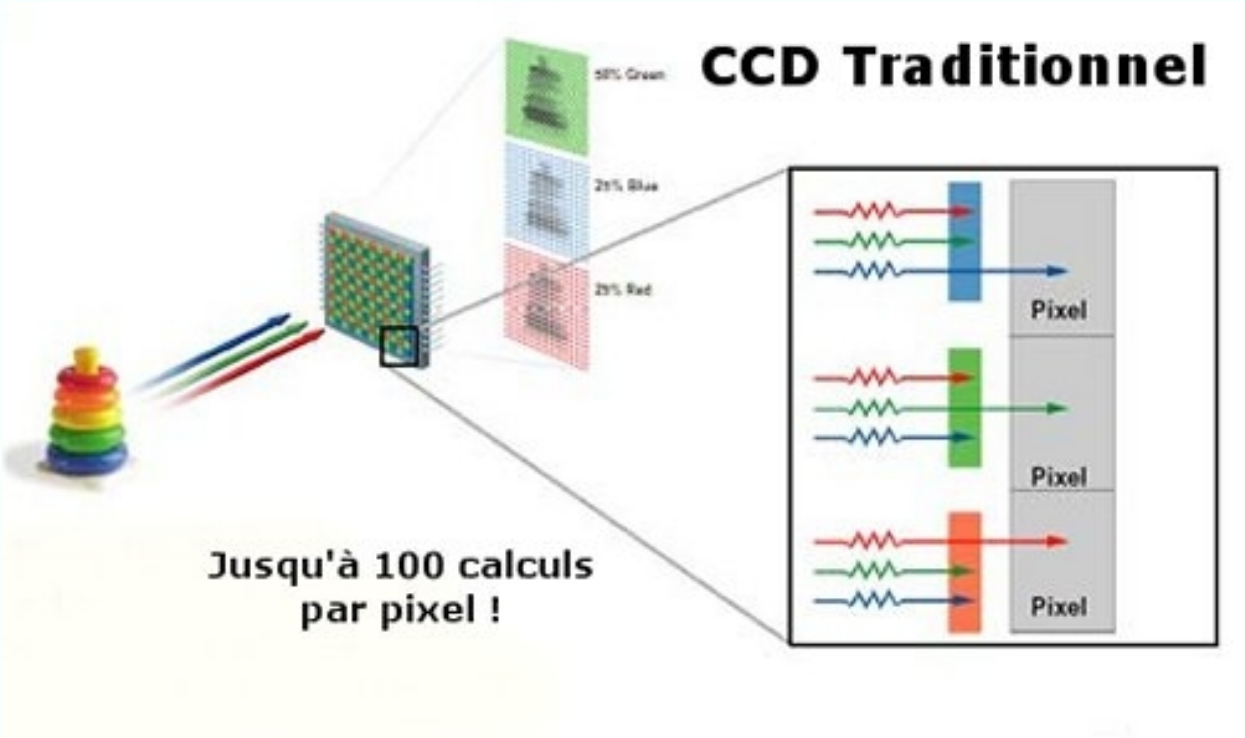
- Good spatial resolution
- Good color separation
- Expensive
- Problem of channel alignment

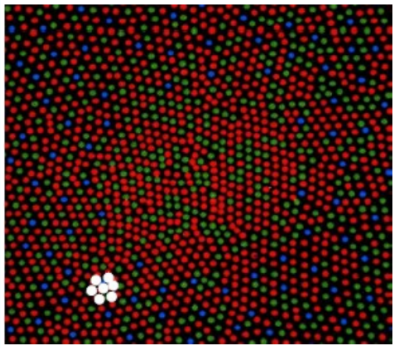
Tri-CCD

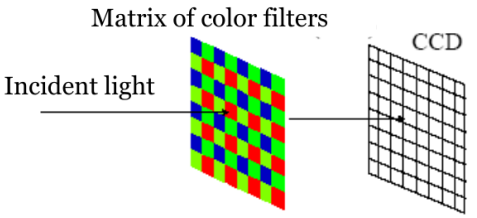
Tri-CCD Image

=

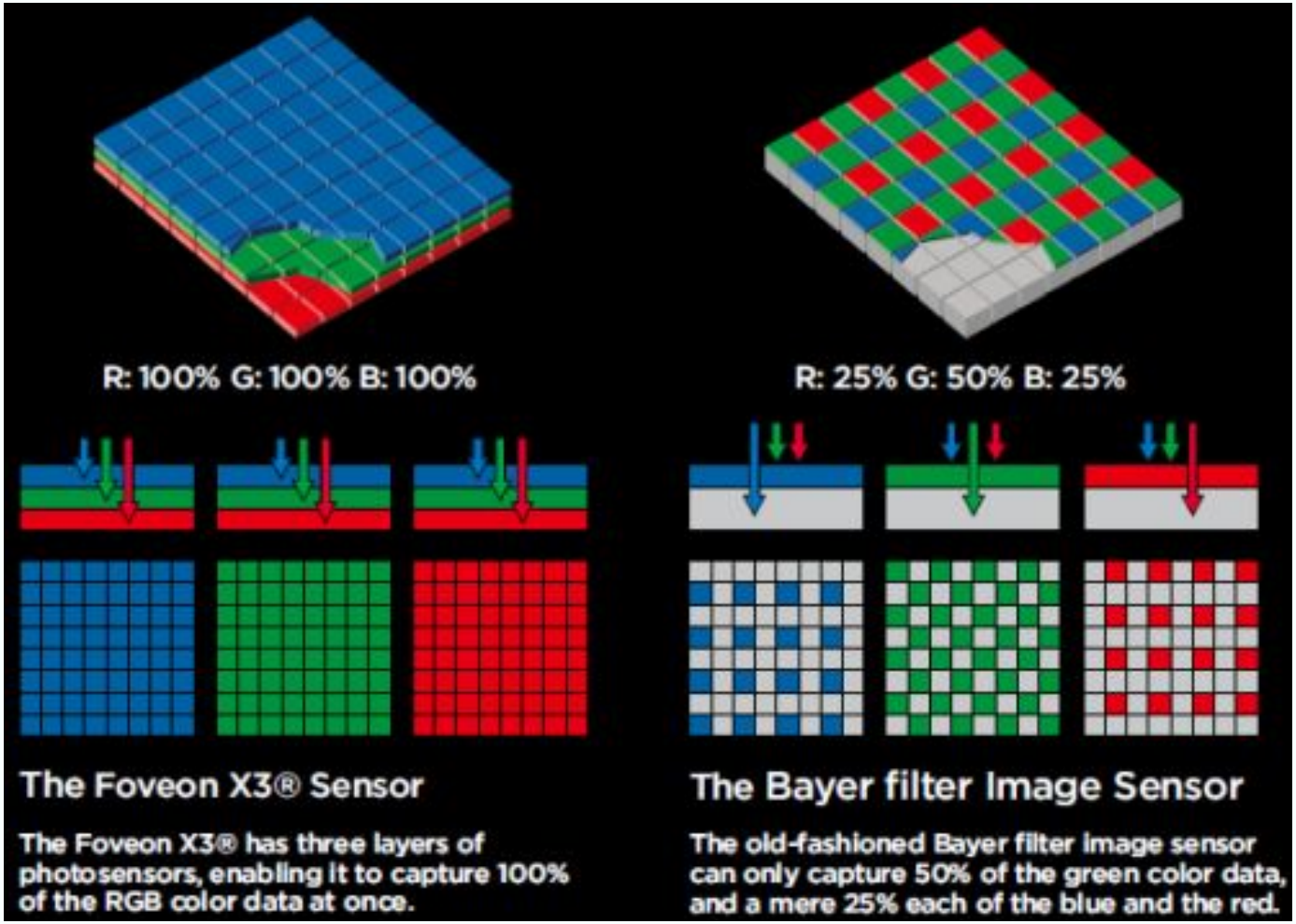
Mono CCD: CFA


CCD Traditionnel
Pixel
Pixel
Pixel
Jusqu'à 100 calculs
par pixel !



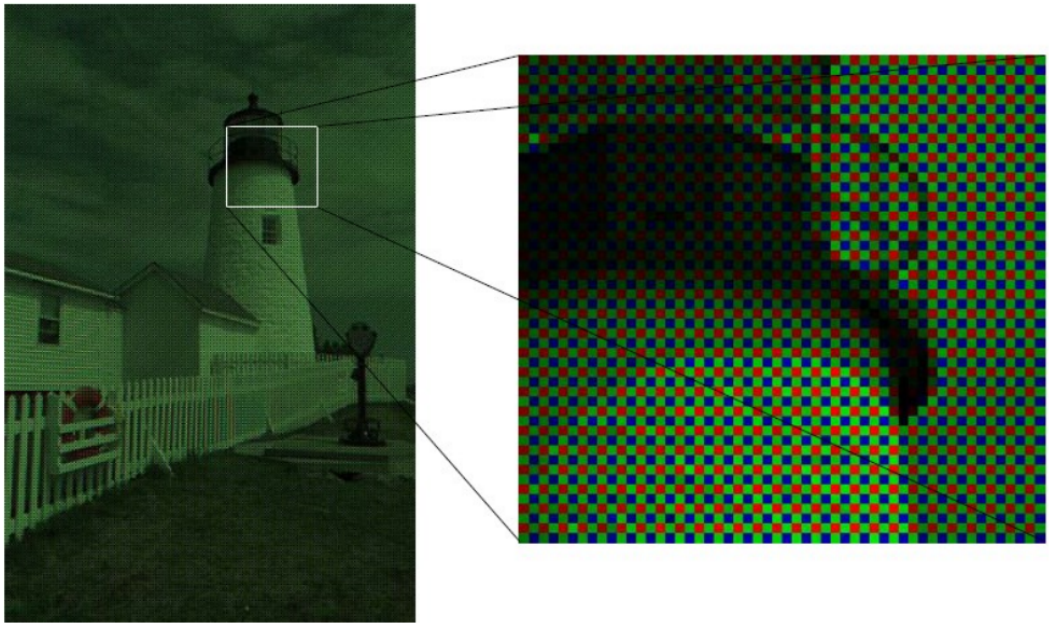

Matrix of color filters
CCD
Incident light

Mono CCD
Bayer pattern and Foveon X3



Mono CCD
Bayer pattern and Foveon X3

CFA-Bayer Foveon X_3

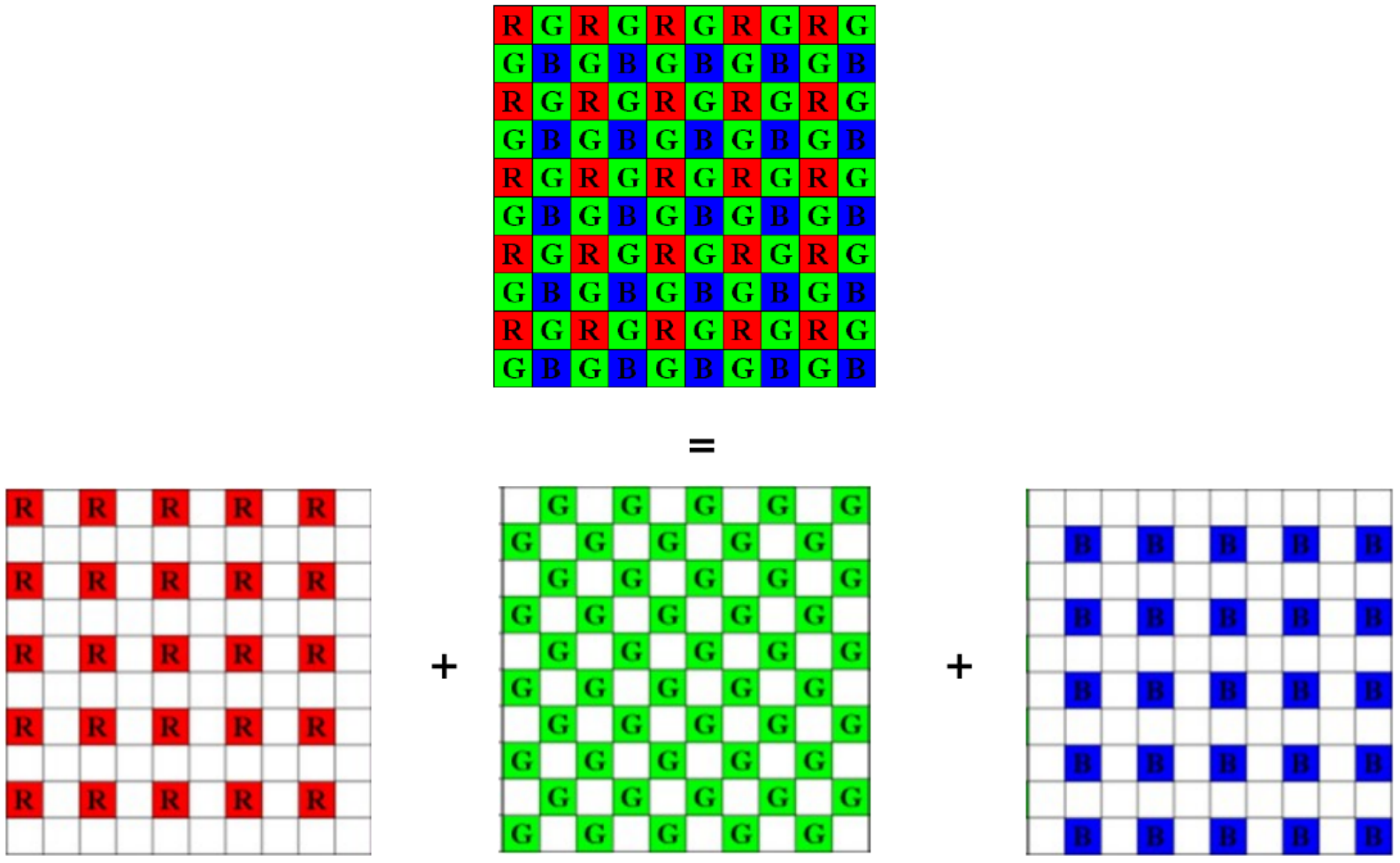


Bayer-CFA Image

Demosaicing
Bayer Color filter Array (CFA) Image

ULR: Original, Red, BLR: Green and blue

Demosaicing
Bayer Color filter Array (CFA) Image



Demosaicing

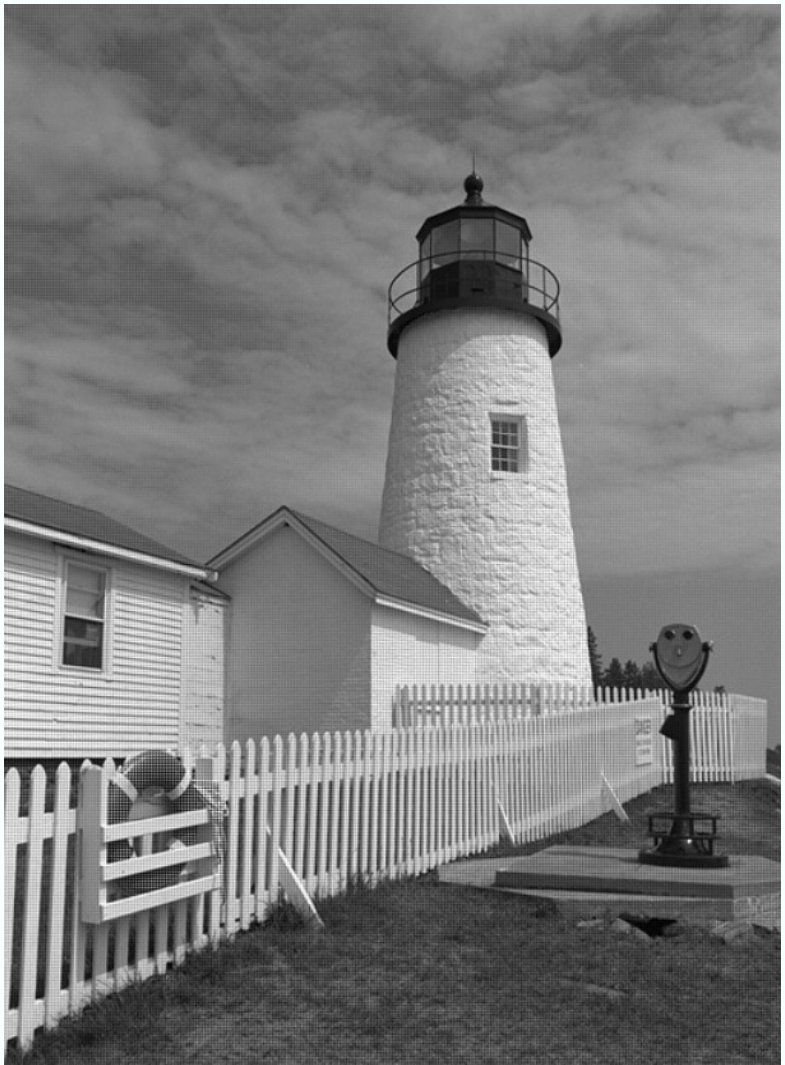
Original and Red

Bayer Color filter Array (CFA) Image

Green and Blue

Bayer Color filter Array (CFA) Image

Bayer CFA image

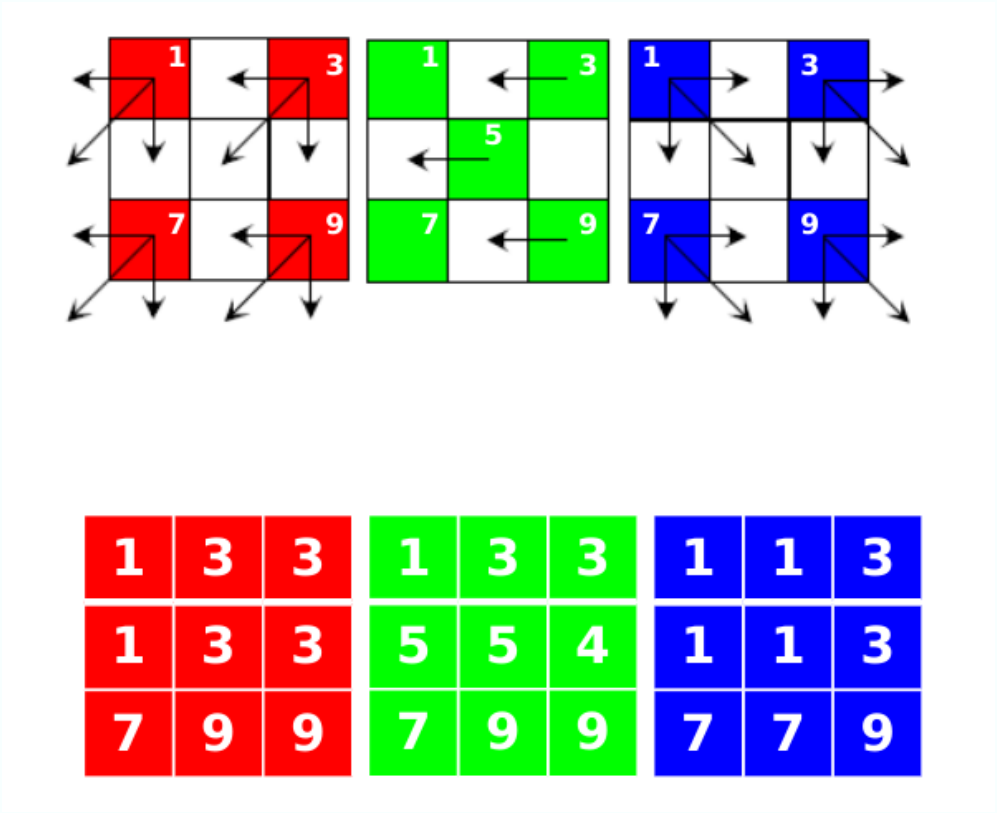


Demosaicing

Techniques

- Nearest neighbor replication
- Bilinear interpolation
- Smooth Hue transition interpolation
- Edge preserving interpolation
- Pattern recognition interpolation

Demosaicing
Nearest Neighbor Replication

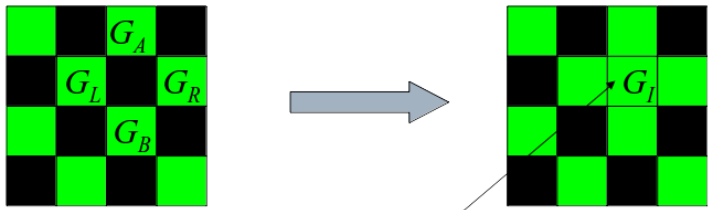


Demosaicing
Nearest Neighbor Replication

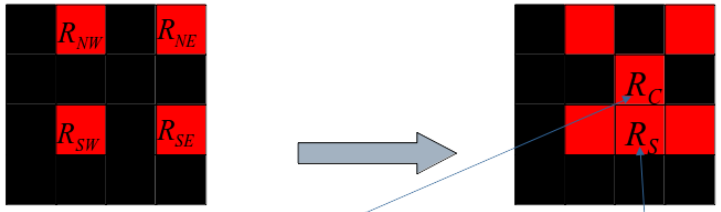
Result



Demosaicing
Bilinear Interpolation



$$G_I = \frac{1}{4}(G_L + G_R + G_B + G_A)$$



$$R_C = \frac{1}{4}(R_{NW} + R_{NE} + R_{SW} + R_{SE})$$

$$R_S = \frac{1}{2}(R_{SW} + R_{SE})$$

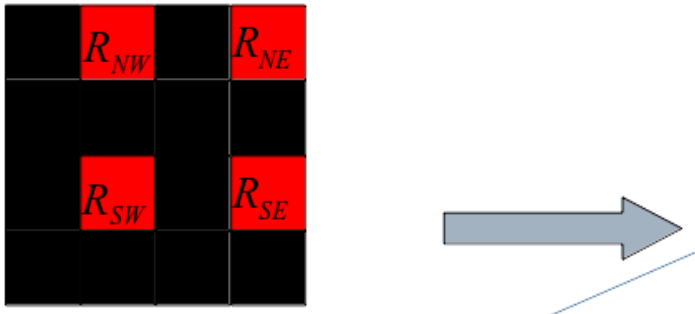
Demosaicing
Bilinear Interpolation

Result




Demosaicing
Smooth Hue Transition Interpolation

- Impose smooth transition in hue from pixel to pixel → Red, hue values = R/G, and Blue, hue values = B/G
- Interpolation of Green pixels same as Bilinear interpolation



$$R_C = \frac{G_I}{4}\left(\frac{R_{NW}}{G_{NW}} + \frac{R_{NE}}{G_{NE}} + \frac{R_{SW}}{G_{SW}} + \frac{R_{SE}}{G_{SE}}\right)$$

$$R_S = \frac{G_S}{2}\left(\frac{R_{SW}}{G_{SW}} + \frac{R_{SE}}{G_{SE}}\right)$$

Demosaicing
Smooth Hue Transition Interpolation

Result




Demosaicing
Edge Preserving Interpolation

- Interpolation of Red / Blue pixels is the same as Smooth Hue Transition
- Working with two gradient for the Green pattern, one in horizontal direction and one in vertical direction
- $\Delta H = |G_L - G_R|$ and $\Delta V = |G_A - G_B|$

If $\Delta H < \Delta V$

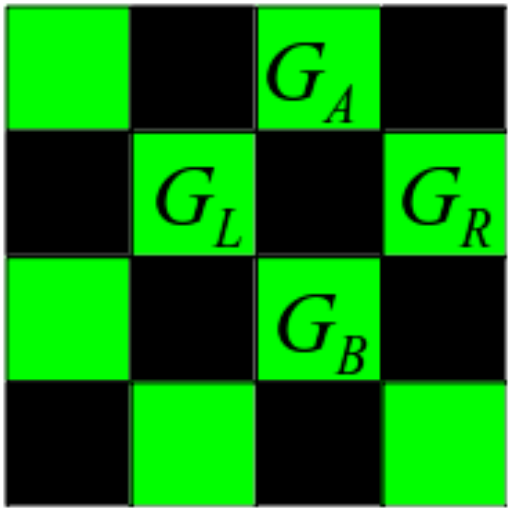
$$G_I = (G_L + G_R)/2$$

Else if $\Delta H > \Delta V$

$$G_I = (G_A + G_B)/2$$

Else

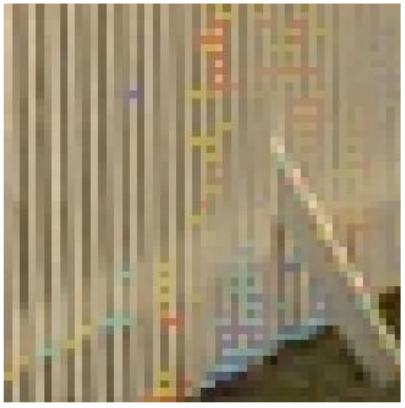
$$G_I = (G_L + G_R + G_A + G_B)/4$$



Demosaicing
Edge Preserving Interpolation

Result



Demosaicing
Edge Preserving Interpolation (larger neighborhood)

- Interpolation of Red / Blue pixels is the same as Smooth Hue Transition
- Horizontal gradient, $\Delta H = |G_4 - G_6| + |R_5 - R_3 + R_5 - R_7|$
- Vertical gradient, $\Delta V = |G_2 - G_8| + |R_5 - R_1 + R_5 - R_9|$

If $\Delta H < \Delta V$

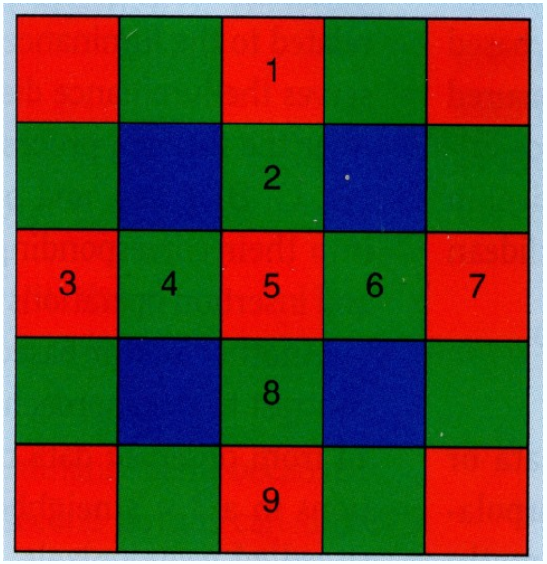
$$G_5 = (G_4 + G_6)/2 + (R_5 - R_3 + R_5 - R_7)/4$$

Else if $\Delta H > \Delta V$

$$G_5 = (G_2 + G_8)/2 + (R_5 - R_1 + R_5 - R_9)/4$$

Else

$$G_I = (G_L + G_R + G_A + G_B)/4 + (R_5 - R_1 + R_5 - R_9 + R_5 - R_3 + R_5 - R_7)/8$$

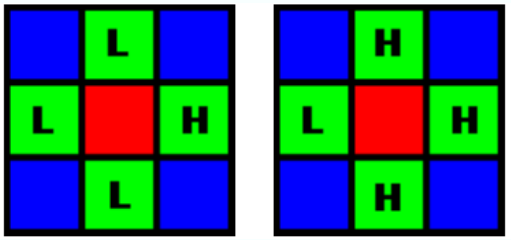


Demosaicing
Edge Preserving Interpolation (Larger neighborhood)

Result

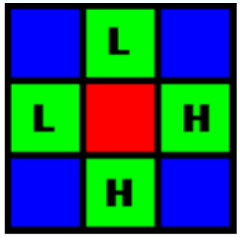
Demosaicing
Pattern Recognition Interpolation

- Interpolating the green color plane considering three different edge type
- Once the green plane is interpolated, the red and blue planes are interpolated using the Smooth Hue Transition
- First: Classifying the four neighborhood and low L or high H in comparison to their average
- Edge: if three neighbor pixel share the same classification (Low and High edge, respectively)

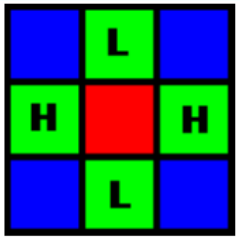


Demosaicing
Edge Preserving Interpolation (Larger neighborhood)

- Corner: if two adjacent neighbor pixels have the same classification



- Stripe: if two opposite pixels have the same classification



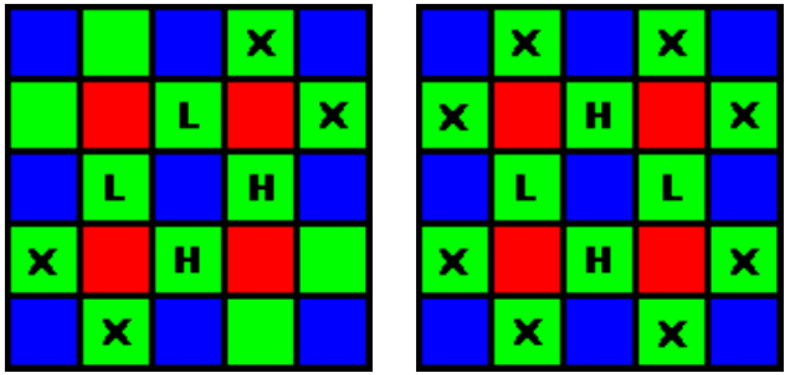
Demosaicing
Edge Preserving Interpolation (Larger neighborhood)

- Edge pixel:
  - $G_I = \text{median}\{\ G_R, G_L, G_A, G_B\ \}$
  - If we sort the neighbors $\{\ G_R, G_L, G_A, G_B\ \}$ as $A > B > C > D$
  - $G_I = (B + C)/2$
- Clip function for calculating the corner and stripe values

$$clip_C^B(x) = \begin{cases} B & x > B \\ x & C \leq x \leq B \\ C & x < C \end{cases}$$

Demosaicing
Edge Preserving Interpolation (Larger neighborhood)

- Corner and Stripe pixel:
  - $G_I = clip_C^B(2M - S)$
  - M median of H and L pixels
  - S average of X pixels in the neighborhood



Demosaicing
Edge Preserving Interpolation (Larger neighborhood)

Result