Scaling input value

- According to basic grayworld algorithm

$$L_i \approx f a_i = \frac{f}{n} \sum_{x,y} c_i(x,y)$$ (11.1)

- Eq 11.1 simplifying

$$L_i(x,y) \approx f a_i(x,y)$$ (11.2)

- For, nonuniform illumination, using geometry factor,

$$c_i(x,y) = G(x,y) R_i(x,y) L_i(x,y)$$ (11.3)
Eq 11.3 become

\[ G(x, y)R_i(x, y) = \frac{c_i(x, y)}{L_i(x, y)} \approx \frac{c_i(x, y)}{f\alpha_i(x, y)} \]  

\[ (11.4) \]

Implementation of color constancy algorithm

**Fig. 11.1.** Implementation of color constancy algorithm that uses local space average color to scale the input value. Data from surrounding pixels is iteratively averaged. Input value are divided by twice the local space average.
Five assumption of local space illumination estimation
- Geometry factor not depend on viewers position
- Sensors are delta function,
- Sensors’s response is linear with no bias
- On average, the world is gray
- Illumination is locally constant

Output result

Fig. 11.2. Output images that were generated by dividing the input values by twice the local space average color.
Color shift
- Cutting unit RGB cube,

Fig. 11.3. Cuts through the red, green, and blue (RGB) cube at a distance of (a) 10%, (b) 30%, (c) 50%, (d) 70% and (e) 90% along the line from black to white.
- Calculating distance between gray vector and local space average color

**Fig. 11.4.** The Gray vector pass from [0,0,0] to [1,1,1] directly through the middle of the color cube. If local space average color is located away from the gray vector, we can use a shift perpendicular to the gray vector to move the color back to the center.
Component of local space average color calculated as

\[ a_\perp = a - (a^T w)w \]  \hspace{1cm} (11.5)

Moving local space average color back to the gray vector

\[ o = c - a_\perp \]  \hspace{1cm} (11.6)

Individual component of Eq. 11.6

\[ o_i = c_i - a_i + \frac{1}{3}(a_r + a_g + a_b) \]  \hspace{1cm} (11.7)

Eq. 11.7 simplifying

\[ o_i = c_i - a_i + \bar{a} \]  \hspace{1cm} (11.8)
– Output result of subtracting local space average color

Fig. 11.5. For these image, output pixels were calculated by subtracting the component of local space average color, which is perpendicular to gray vector, from the current pixel color.
– Visualized calculation of local space average moving

**Fig. 11.6.** First we calculate the component $\mathbf{a} \perp$ that is perpendicular to the gray vector $\mathbf{w}$ and points to local space average color. If we subtract this vector from the color of the input pixel $\mathbf{c}$, we essentially move the local space average color onto the gray vector.
Data flow of computation of local space average color

Fig. 11.6. The first layer is the input layer. The second layer computes local space average color. The output is computed in the third layer. The measured red, green, and blue components are stored in cells $c_i$ with $i \in \{r, g, b\}$. Space average color is stored in cells $a_i$, the average of these three components is stored in $a$. Output cells are denoted by $o_i$ (from Ebner 2003c).
Normalized color shift

- Because of current pixel color and local space average color located different level
- Using normalized color vector

\[
\hat{c} = \frac{1}{c_r + c_g + c_b} [c_r, c_g, c_b]^T
\]  

\[
\hat{a} = \frac{1}{a_r + a_g + a_b} [a_r, a_g, a_b]^T
\]

- Local space average vector calculated as

\[
a_{\perp} = a - (a^T w)w
\]
- Moving local average to gray vector

\[ \mathbf{o} = \mathbf{c} - \mathbf{a}_\perp \]  

(11.12)

- Individual component as

\[ \hat{o}_i = \hat{c}_i - \hat{a}_i + \frac{1}{3} \]  

(11.13)

- Scaling output pixel back to the original intensity

\[
o_i = (c_r + c_g + c_b)\hat{o}_i \\
= c_i - (c_r + c_g + c_b)(\hat{a}_i - \frac{1}{3}) \\
= c_i - \frac{c_r + c_g + c_b}{a_r + a_g + a_b}(a_i - \frac{1}{3}(a_r + a_g + a_b)) \\
= c_i - \frac{\overline{c}}{\overline{a}}(a_i - \overline{a}) \]  

(11.16)
Visual operation

Fig. 11.8. The color of the current pixel $c$, and local space average color $a$ are projected onto the plane $r + g + b = 1$. Let $\hat{c}$ and $\hat{a}$ be the normalized points. Now, normalized local space average color is projected onto the gray vector $w$. The projection is subtracted from $\hat{a}$, which gives us $\hat{a}_\perp$. The component $\hat{a}_\perp$ is orthogonal to the gray vector $w$. This component is subtracted from the color of the current pixel that gives us the normalized output color $\hat{o}$. Finally, the output color is scaled back to the intensity of the input pixel.
- Output image

Fig. 11.9. For these images, output pixels were calculated by subtracting the component of local space average color, which is perpendicular to the gray vector, from the current pixel color.
- Data flow

**Fig. 11.10.** Computation of color constant descriptors using normalized color shifts. The first layer is the input layer. The second layer computes local space average color. The output is computed in the third layer. The measured red, green, and blue components are stored in cells $c_i$ with $i \in \{r, g, b\}$. The average of these three components is stored in cell $.c$. Space average color is stored in cells $.a$, the average of these three components is stored in $.a$. Output cells are denoted by $o_i$ (from Ebner 2003c).
Color gamut of different local space average

Fig. 11.11. Color gamut for three different values of local space average color. If we assume that the space of possible colors is shifted toward local space average color $a$, this also implies a smaller color gamut.