Advanced PixInsight PixelMath Operations

David Ault
Why use PixelMath

• PixelMath is a very powerful tool that gives you access to all sorts of features that otherwise would require javascripts, plug-in development (PCL) or standalone programs

• I use it regularly for:
  • Blending images with various functions (averaging, max, min, etc.)
  • Hot pixel removal
  • Altering or creating masks
  • Testing calibration data
  • Linear gradient based clipping or merging
  • Noise reduction
  • Drawing lines, circles or other geometry on an image
  • Removing unwanted artifacts (star halos, etc.)
Syntax

• **Symbols**
  • These are the equivalent of variables in programming languages
  • For every assignment made in the expression space there must be a corresponding symbol
  • Symbols can also be assigned values but only constants. This is the equivalent of initializing variables.
  • There is a built in symbol, $T$, which is used to reference the active or new instance view (T stands for target)
  • Symbols need to be separated by commas

• **Expressions**
  • This is where all the math is done
  • All sorts of functions are available from, log to sine to image specific functions like biweight midvariance
  • The PixelMath engine can handle parenthetical equations
  • Symbols are available for most functions: addition, subtraction, multiplication, powers.
  • Expressions are separated by semi-colons with the result from the last expression returned as the pixel value
How it works

• PixelMath runs in a big loop over each pixel in the active view

• For example, the expression `0.5 * T` will multiply every pixel in the target view by 0.5 reducing the brightness by half

• There are functions available for determining where you are in the image loop like `x()` & `y()` which can be used for targeting location based variations

• There are also functions that are non loop based like `mean()`, `median()`, `bwmv()`. These operate on an entire image and return a single value result. Some of these functions, like `mean()`, can also operate on a list of values.
The Expression Editor

• You can do a lot of math with the base PixelMath form but if you want to do more complex functions spanning multiple equations the Expression Editor is very useful.

• You get quick access to Images, Symbols, Function, etc. which can be added to your expression by double clicking on them.

• You can also check the syntax of your expressions without running it on the entire image.

• There is also syntax highlighting making it easier to read the expressions.
Examples

• Image Blending
  • Weighted linear blend (also called alpha blend or weighted averaging)
  • Photoshop equivalents
  • Star mask combination
  • Synthetic channel generation

• Rendering
  • Inserting lines and circles
  • Cross-sections

• Hot pixel removal / noise reduction
• Star Halo Removal
• Manual calibration evaluation
Reference
Image Blending

• Alpha Blend
  RGB/K: \( a \times \text{Image1} + (1-a) \times \text{Image2} \)
  or
  RGB/K: \( 0.4 \times R + 0.3 \times G + 0.3 \times B \)
  Symbols: \( a=0.5 \)

• Synthetic Green
  G: \( \text{G} = \begin{cases} 1 - 0.5 & \text{if } B > 0.5, \\ 1 - R \times (1 - (B - 0.5)), \\ R \times (B + 0.5) & \end{cases} \)
  G: \( \text{tg} = 0.1 \times R + 0.9 \times B; \)
  \( a \times \text{tg} + (1-a) \times \min(\text{tg}, (R+B)/2) \)
  Symbols: \( \text{tg}, a=0.5 \)

• Star Mask Combination
  RGB/K: \( \max(\text{star\_mask}, \text{star\_mask1}, \text{star\_mask2}) \)
Photoshop blending modes

The a and a-1 portions of the equations are the alpha blend. This equates to the opacity slider in Photoshop except it has a range of 0 to 1 instead of 0 to 100.

• Normal
  \[ a \cdot \text{top} + (1-a) \cdot \text{bot} \]

• Multiply
  \[ a \cdot \text{top} \cdot \text{bot} + (1-a) \cdot \text{bot} \]

• Screen
  \[ a \cdot (1-(1-\text{top})*(1-\text{bot})) + (1-a) \cdot \text{bot} \]

• Overlay
  \[ a \cdot \text{iiif}(<0.5, 2 \cdot \text{bot} \cdot \text{top}, 1-2*(1-\text{top})*(1-\text{bot})) + (1-a) \cdot \text{bot} \]

• Darken
  \[ a \cdot \text{min} (\text{top, bot}) + (1-a) \cdot \text{bot} \]

• Lighten
  \[ a \cdot \text{max} (\text{top, bot}) + (1-a) \cdot \text{bot} \]

• Addition
  \[ a \cdot (\text{top} + \text{bot}) + (1-a) \cdot \text{bot} \]

• Subtraction
  \[ a \cdot (\text{top} - \text{bot}) + (1-a) \cdot \text{bot} \]

• Division
  \[ a \cdot (\text{top} / \text{bot}) + (1-a) \cdot \text{bot} \]
More Photoshop blending modes

- **Linear Burn**
  \[ a \cdot (\text{top} + \text{bot} - 1) + (1-a) \cdot \text{bot} \]

- **Color Burn**
  \[ a \cdot (1 - (1 - \text{top})/\text{bot}) + (1-a) \cdot \text{bot} \]

- **Color Dodge**
  \[ a \cdot (\text{top}/(1-\text{bot})) + (1-a) \cdot \text{bot} \]

- **Soft Light**
  \[ a \cdot \text{iif}(\text{bot}>0.5, 1-(1-\text{top})*(1-(\text{bot}-0.5)), \text{top}*(\text{bot}+0.5)) + (1-a) \cdot \text{bot} \]

- **Hard Light**
  \[ a \cdot \text{iif}(\text{bot}>0.5, 1-(1-\text{top})*(1-(2*(\text{bot}-0.5))), 2*\text{top} \cdot \text{bot}) + (1-a) \cdot \text{bot} \]

- **Exclusion**
  \[ a \cdot (0.5-2*(\text{top}-0.5)*(\text{bot}-0.5)) + (1-a) \cdot \text{bot} \]
Rendering

• Simple Circle
  RGB/K: \[ r = \sqrt{(x-cx)^2 + (y-cy)^2}; \ iif(|abs(tr-r)|<0.5, 1, $T) \]
  Symbols: \( cx=500, cy=500, tr=400, r \)

• Horizontal Line
  RGB/K: \[ iif(x()==xloc, 1, $T) \]
  Symbols: \( xloc=685 \)

• Aliased Circle
  RGB/K: \[ r = \text{rdist}(cx, cy); a = \text{abs}(tr-r)/(w/2); \ iif(a<1, a*\$T+(1-a), \$T) \]
  Symbols: \( cx=1700, cy=1200, tr=300, w=5, r, a \)

• Aliased Line
  RGB/K: \[ r = \text{d2line}(x1, y1, x2, y2); a = (r/(w/2))^0.5; \ iif(a<1, a*\$T+(1-a), \$T) \]
  Symbols: \( x1=332, y1=788, x2=1472, y2=1112, tr=300, w=5, r, a \)
Rendering

• Green Tick Mark
  
  R: \[ \text{iif((x()>(cx+xo)) \&\& (x()<(cx+xo+xl)) \&\& (y()==cy) | | ((y()>(cy+yo)) \&\& (y()<(cy+yo+yl)) \&\& (x()==cx)), 0, \$T) } \]
  
  G: \[ \text{iif((x()>(cx+xo)) \&\& (x()<(cx+xo+xl)) \&\& (y()==cy) | | ((y()>(cy+yo)) \&\& (y()<(cy+yo+yl)) \&\& (x()==cx)), 1, \$T) } \]
  
  B: \[ \text{iif((x()>(cx+xo)) \&\& (x()<(cx+xo+xl)) \&\& (y()==cy) | | ((y()>(cy+yo)) \&\& (y()<(cy+yo+yl)) \&\& (x()==cx)), 0, \$T) } \]
  
  Symbols: \[ cx=345, cy=322, xo=15, yo=15, xl=30, yl=30 \]

• Line Segment
  
  RGB/K: \[ d = d2seg(llx, lly, urx, ury); a = 1 - d/(lw/2); \text{iif(d<(lw/2), a + (1-a)*\$T, \$T) } \]
  
  Symbols: \[ llx=30, lly=356, urx=965, ury=179, lw=5, d, a \]

• Highlight Box in Yellow
  
  R: \[ \$T[0] \]
  
  G: \[ \$T[1] \]
  
  B: \[ \text{iif(x())>llx \&\& x()<urx \&\& y()>lly \&\& y()<ury, 0, \$T[2]) } \]
  
  Symbols: \[ llx=32, lly=374, urx=723, ury=403 \]
Cross Section analysis

• Cross-section variation (two pass)

  RGB/K: \[\text{pixel}(T, x(), 0.5\times h(T))\]

  RGB/K: \[\text{iif}(((1-T)\times h(T))>y(), 0, T)\]

  or

  RGB/K: \[d = \text{abs}(((1-\text{CIEL}(T))\times h(T))-y()));\]

  \[\text{iif}(d>r, 0, r-d)\text{ where } r=3\]
Hot Pixel Removal

• Symbols
  \( f = 9.0, w, h, x_0, x_1, x_2, y_0, y_1, y_2, p_{00}, p_{01}, p_{02}, p_{10}, p_{11}, p_{12}, p_{20}, p_{21}, p_{22}, \text{value}, \text{sd} \)

• RGB/K
  \[
  \begin{align*}
  w &= \text{width}(T) - 1; \\
  h &= \text{height}(T) - 1; \\
  x_1 &= x(); \\
  y_1 &= y(); \\
  x_0 &= \text{iif}(x_1 < 1, 0, x_1 - 1); \\
  y_0 &= \text{iif}(y_1 < 1, 0, y_1 - 1); \\
  x_2 &= \text{iif}(x_1 > w, w, x_1 + 1); \\
  y_2 &= \text{iif}(y_1 > h, h, y_1 + 1); \\
  p_{00} &= \text{pixel}(T, x_0, y_0); \\
  p_{01} &= \text{pixel}(T, x_0, y_1); \\
  p_{02} &= \text{pixel}(T, x_0, y_2); \\
  p_{10} &= \text{pixel}(T, x_1, y_0); \\
  p_{11} &= \text{pixel}(T, x_1, y_1); \\
  p_{12} &= \text{pixel}(T, x_1, y_2); \\
  p_{20} &= \text{pixel}(T, x_2, y_0); \\
  p_{21} &= \text{pixel}(T, x_2, y_1); \\
  p_{22} &= \text{pixel}(T, x_2, y_2); \\
  \text{value} &= \text{med}(p_{00}, p_{01}, p_{02}, p_{10}, p_{11}, p_{12}, p_{20}, p_{21}, p_{22}); \\
  \text{sd} &= \sqrt{\text{bwmv}(p_{00}, p_{01}, p_{02}, p_{10}, p_{12}, p_{20}, p_{21}, p_{22})}; \\
  \text{iif}(p_{11} > (\text{value} + (f \times \text{sd})), \text{value}, \text{iif}(p_{11} < (\text{value} - (f \times \text{sd})), \text{value}, p_{11}))
  \end{align*}
  \]

• RGB/K continued
  \[
  \begin{align*}
  p_{11} &= \text{pixel}(T, x_1, y_1); \\
  p_{12} &= \text{pixel}(T, x_1, y_2); \\
  p_{20} &= \text{pixel}(T, x_2, y_0); \\
  p_{21} &= \text{pixel}(T, x_2, y_1); \\
  p_{22} &= \text{pixel}(T, x_2, y_2); \\
  \text{value} &= \text{med}(p_{00}, p_{01}, p_{02}, p_{10}, p_{11}, p_{12}, p_{20}, p_{21}, p_{22}); \\
  \text{sd} &= \sqrt{\text{bwmv}(p_{00}, p_{01}, p_{02}, p_{10}, p_{12}, p_{20}, p_{21}, p_{22})}; \\
  \text{iif}(p_{11} > (\text{value} + (f \times \text{sd})), \text{value}, \text{iif}(p_{11} < (\text{value} - (f \times \text{sd})), \text{value}, p_{11}))
  \end{align*}
  \]

• I found biweight midvariance to be more robust for such a small set of pixels compared to standard deviation
Removing Purple Stars

• Magenta Star Reduction
  \[\text{R: } T[0]\]
  \[\text{G: } \text{iiif}(\text{min}(T[0],T[2])>T[1],\text{min}(T[0],T[2]),T[1])\]
  \[\text{B: } T[2]\]

• In order to work on just stars this needs to be combined with a good star mask.
Calibration Math

• Bias and flats only, assuming flats have been calibrated w/ dark flat or bias frames
  \[
  \text{calibrated light} = (\text{light} - \text{bias}) \times \frac{\text{mean(flat)}}{\text{max}(0.00002, \text{flat})}
  \]

• Bias, scaled darks and flats, assuming flats have been calibrated w/ dark flats or bias frames
  \[
  \text{calibrated light} = \frac{(\text{light} - \text{bias}) - k(\text{dark} - \text{bias}) \times \text{mean(flat)}}{\text{max}(0.00002, \text{flat})}
  \]

• Dark and flats only, assuming flats have been calibrated w/ dark flats or bias frames
  \[
  \text{calibrated light} = \frac{(\text{light} - \text{dark}) \times \text{mean(flat)}}{\text{max}(0.00002, \text{flat})}
  \]

• Bias and Flats Only, with uncalibrated flats
  \[
  \text{calibrated light} = \frac{(\text{light} - \text{bias}) \times (\text{mean(flat)} - \text{mean(bias)})}{\text{max}(0.00002, \text{flat-bias})}
  \]

• Bias, scaled darks and flats, with uncalibrated masters
  \[
  \text{calibrated light} = \frac{((\text{light} - \text{bias}) - k(\text{dark} - \text{bias}) \times (\text{mean(flat)} - \text{mean(bias)})}{\text{max}(0.00002, \text{flat-bias})}
  \]

• In most cases bias and dark flats are interchangeable, however if your flat frames are very long and your sensor has high dark current then dark flats will work better
Resources

• http://pixinsight.com/forum/index.php?board=11.0
• http://en.wikipedia.org/wiki/Blend_modes
• http://harrysastroshed.com/pixinsight/pixinsight%20video%20html/Pixinsighthome.html
• http://pixinsight.com/tutorials/master-frames/index.html

• Handbook of CCD Astronomy – Howell
• Lessons from the Masters – Gendler et al.