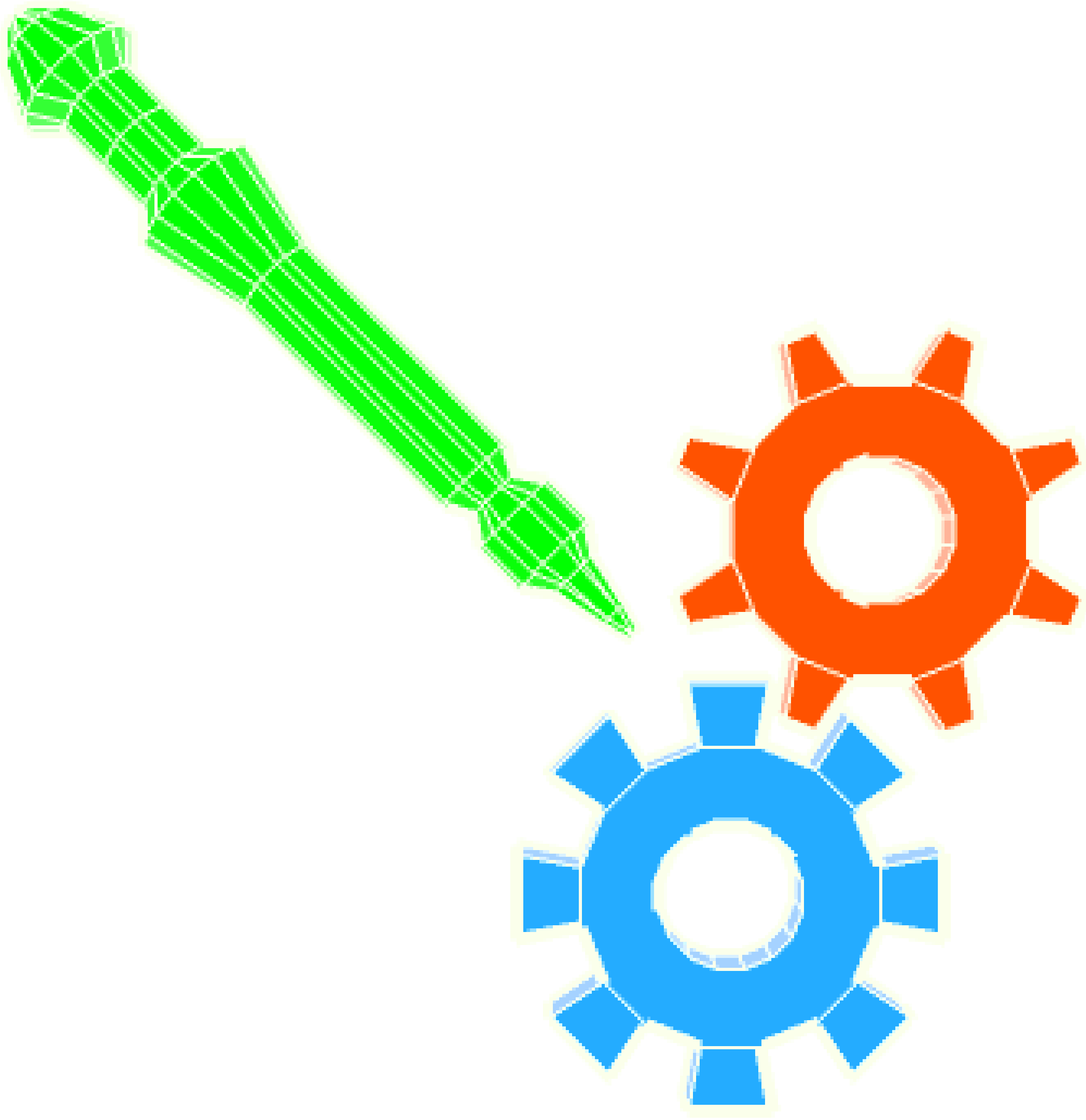
PyImaging Documentation

PyImaging an image treatment programme with severals functionnalities:

- Images file merging algorithms.
- Images filters.
- Gray, red, green, blue scale effects.
- Color matrix multiplications effects.
- Image file informations.



author: Eddie Bruggemann <mrcyberfighter@gmail.com>

Image files

Load images files

Saving images files

Images file informations

Merging images files

Image files merging files formats

Blend images interpolating

Composite images interpolating

Screen images interpolating

Darker images interpolating

Lighter images interpolating

Difference images interpolating

Add images interpolating

Subtract images interpolating

Edition

Undo-Redo

Rotations

Mirroring

Effects

Grayscale

Filters

Adjustments

Colors tools

Redscale editor

Greenscale editor

Bluescale editor

The right tools side-bar

Color inverter matrix

Change intensity

Settings

execution speed

About

About PyImaging

Image files

Load images files

PyImaging support following image files formats (extensions) to load into the programme:

- **bmp** *(Bitmap image file.)*.
- **jpg | jpeg** *(Joint Photographic Experts Group.)*.
- **pcx** *(Pulse Code Modulation.)*.
- **pdf** *(Windows Bitmap.)*.
- **png** *(Portable Network Graphics.)*.
- **ppm** *(Portable pixmap file format.)*.
- **tif | tiff** *(Tag(ged) Image File Format.)*.

In fact the image file is successful loaded into the programme if the program can convert it, if mode conversion needed:

In mode **RGBA** *(Red, Green, Blue, Alpha)*.

Or

In mode **RGB** *(Red, Green, Blue)*.

The programme first check if the image file is in RGBA or RGB mode. If not he try to convert the image file into one of this mode.

This is because the program works, for a part of his functions, on pixels values composed from:

- One Red value.
- One Green value.
- One Blue value.
- An optional Alpha channel value.

  If the image has no alpha channel, transparency capacities, this value is ignored.

You can load an image into **PyImaging** throught:

- The Files menu:

  +-> Files

  +-> Load image
- The button: **Load image** in the toolbar.
- By given the image filepath as argument to the program on the commandline.

```
$ PyImaging Image_filepath
```

Note

By image file loading **PyImaging** create an internal copy of the image for saving the processes effects.

And create if needed an thumbnail internal image if the image size is greater than the image displaying container.

All effect processing operations are apply on this image(s) and not of the original image.

Saving images files

PyImaging support following image files formats (extensions) to save the image as:

- **bmp** *(Bitmap image file.)*.
- **gif** *(Graphics Interchange Format.)*.
- **jpeg** *(Joint Photographic Experts Group.)*.
- **pcx** *(Pulse Code Modulation.)*.
- **pdf** *(Windows Bitmap.)*.
- **png** *(Portable Network Graphics.)*.
- **ppm** *(Portable pixmap file format.)*.
- **tiff** *(Tag(ged) Image File Format.)*.

The programme **PyImaging** can write image files in the differents formats listed above.

Note

Note that the program can write *.gif files but not load it.

And that only the *.jpeg extension is supported (not the *.jpg) for the Joint Photographic Experts Group file format.

This is the same for the *.tiff extension (*.tif is not supported) for the Tag(ged) Image File Format.

You can save an image in **PyImaging** throught:

- The Files menu:

  +-> Files

      +-> Save image
- The button: **Save image** in the toolbar.

Note

You can set the output image file size arbitrary, from the result image file you save as, and choose or not to keep the ratio width / height.

For keeping the ratio after width or height value change, you can press the button **Size ratio**:

the width or height value is preserve in relationship to the selected value and the other value is order to the ratio.

note: Think to press the **Enter** key after setting an value with the keyboard, instead of use of the spinning buttons, for confirm the value so that **PyImaging** can register the value for computing the ratio.

Images file informations

PyImaging can give you some informations and statistics about the loaded image in the current state, it means that the informations and statistics about the image file are relative to the effect process in case of the **pixels stats** and global for the **main informations** and the **miscellaneous** informations.

The image file information window can be access throught:

- The Files menu:

  +-> Files

  +-> Display image informations
- The button **Informations about the file** in the toolbar.

The informations window is divide into 3 parts:

- **Main informations**

  containing:

  - The filename
  - The image format and his description.
  - The image size: width and height.
- **Pixels informations**

  divided into 5 columns (and several rows) :

  - Data: type of information
  - Red: containing the red pixel component statistic digits.
  - Green: containing the green pixel component statistic digits.
  - Blue: containing the blue pixel component statistic digits.
  - Alpha: containing the alpha pixel component statistic digits.
- **Miscellaneous informations**

  containing

  - the hidden informations contains in the image file.

Merging images files

PyImaging has the capability to merge 2 different images files, to an single output file,

which is computed from differents parameters:

- The inputs images files.
- The used image files merging algorithm.
- Additional user settings values, like an alpha, scale, offset value... or even an mask image file.

You can access to the images merging interfaces throught:

- The Files menu:

  +-> Files

  +-> Blend images interpolating

  +-> Composite images interpolating

  +-> Screen images interpolating

  +-> Darker images interpolating

  +-> Lighter images interpolating

  +-> Difference images interpolating

  +-> Multiply images interpolating

  +-> Add images interpolating

  +-> Subtract images interpolating
- The button: **Blend** in the toolbar.
- The button: **Composite** in the toolbar.
- The button: **Screen** in the toolbar.

Note

You must set the 2 inputs image files before setting the output filename, from which you can choose the extension.

You can preview the result from the image files merging by pressing the **Preview** button.

The images are resized per default to the size of the littles input image, this is required that the image to merge have the same size, this is done automatically by **PyImaging**.

But you can set the size, from the result image file you save as, arbitrary, and choose or not to keep the ratio width / height.

For keeping the ratio after width or height value change, you can press the button **Size ratio**:

the width or height value is preserve in relationship to the selected value and the other value is order to the ratio.

note: Think to press the **Enter** key after setting an value with the keyboard, instead of use of the spinning buttons, for confirm the value so that **PyImaging** can register the value for computing the ratio.

Image files merging files formats:

PyImaging support following image files formats (extensions) to load as input files:

- **bmp** *(Bitmap image file.)*.
- **gif** *(Graphics Interchange Format.)*.
- **jpg | jpeg** *(Joint Photographic Experts Group.)*.
- **pcx** *(Pulse Code Modulation.)*.
- **pdf** *(Windows Bitmap.)*.
- **png** *(Portable Network Graphics.)*.
- **ppm** *(Portable pixmap file format.)*.
- **tif | tiff** *(Tag(ged) Image File Format.)*.

Note

It is different for the **Composite** file mergin algorithm mask file which must have transparency support.

So the image file format:

- **gif** *(Graphics Interchange Format.)*.

is exclude from the list.

And the image file format:

- **xbm** (X Bitmap.).

is added to the list.

PyImaging support following image files formats (extensions) to save as output for files merging:

- **bmp** *(Bitmap image file.)*.
- **gif** *(Graphics Interchange Format.)*.
- **jpeg** *(Joint Photographic Experts Group.)*.
- **pcx** *(Pulse Code Modulation.)*.
- **pdf** *(Windows Bitmap.)*.
- **png** *(Portable Network Graphics.)*.
- **ppm** *(Portable pixmap file format.)*.
- **tiff** *(Tag(ged) Image File Format.)*.

The programme **PyImaging** can write image files in the differents formats listed above.

Note

Note that only the *.jpeg extension is supported (not the *.jpg) for the Joint Photographic Experts Group file format.

This is the same for the *.tiff extension (*.tif is not supported) for the Tag(ged) Image File Format.

Blend images interpolating:

For generating an **Blend** output image file you must:

- Set the input images to interpolate.
- Setting the alpha value (which is important).

And the **Blend** images merging algorithm will do the work.

The output file is compute, for every pixel by:

Subtract the result from:

- The multiplication from input Image2 with 1.0 - alpha.

From

- The multiplication from input Image1 with the value of alpha.

$$Result = Image1 * alpha - (1 - alpha) * Image2$$

Note

This file merging algorithm make use of the color blend Principe which is

to generate an color from an source and an destination color:

If the source color mark is 0 and the destination color mark is 1.0:

The alpha value is simply the place between this 2 colors.

Composite images interpolating:

For generating an **Composite** output image file you must:

- Set the input images to interpolate.
- Set an mask image file, which must have an alpha channel (transparency support). If it is not the case **PyImaging** will inform you about and cancel the file selection.

And the **Composite** images merging algorithm will do the work.

The output file is computed by:

Interpolating the 2 input image files with use of the mask image as an transparent mask.

Note

This file merging algorithm make use of an mask image which must have transparency support,

normal for a mask image the others images will be shown throught this mask after being interpolate.

So think about the pixel who are completely transparent, because there will show what behind the mask,

and about who not are completely transparent, they will be shown on the first frame.

Screen images interpolating:

For generating an **Screen** output image file you must:

- Set the input images to interpolate.

And the **Screen** images merging algorithm will do the work.

The output file is compute, for every pixel by:

Subtracting the :

- the maximal value

from

- the multiplication from :
  - The subtract from the maximal value from the Image1 value.

  With

  - The subtract from the maximal value from the Image2 value.

  Divided by the maximal value.

$$Result = MAX - ((MAX - Image1) * (MAX - Image2) / MAX)$$

Note

This file merging algorithm make several use of the maximal pixel value (from the entires pixels values, because otherwise it is not possible).

By subtracting from it the division from the multiplication from the subtraction of it from the Image1 and Image2 values by it.

The maximal value has an great role in this algorithm and the greatest pixel value is the color white.

Think about it wenn you choose your input images !

Darker images interpolating:

For generating an **Darker** output image file you must:

- Set the input images to interpolate.

And the **Darker** images merging algorithm will do the work.

The output file is compute, for every pixel by:

Computing the minimal value comparing Image1 and Image2 pixel value and set it into the result image.

$$Result = min(Image1, Image2)$$

Note

By taking the lower pixel value, by comparing the two images files,

it is sure that the result image will be darker.

Because as low the pixel value is, so nearer from the black it is.

Lighter images interpolating:

For generating an **Lighter** output image file you must:

- Set the input images to interpolate.

And the **Lighter** images merging algorithm will do the work.

The output file is compute, for every pixel by:

Computing the maximal value, by comparing Image1 and Image2 pixel value and set it into the result image.

$$Result = max(Image1, Image2)$$

Note

By taking the highest pixel value, by comparing the two images files,

it is sure that the result image will be lighter.

Because as high as the pixel value is, so nearer from the white it is.

Difference images interpolating:

For generating an **Difference** output image file you must:

- Set the input images to interpolate.

And the **Difference** images merging algorithm will do the work.

The output file is compute, for every pixel by:

Computing the absolute value (unsigned), from:

The subtraction of

- Image1 pixel value.

From

- Image2 pixel value.

$$Result = abs(Image1 - Image2)$$

Note

This algorithm is clever because the subtract from 2 pixels value can give an negative value as result.

And negative values are not accepted by pixel values, and it is an reverting idea to set the absolute (unsigned) value into the result image by subtracting 2 pixel values.

Add images interpolating:

For generating an **Add** output image file you must:

- Set the input images to interpolate.
- Setting an *scale* value.
- Setting an *offset* value.

And the **Add** images merging algorithm will do the work.

The output file is compute, for every pixel by:

The division from

- the adding from Image1 and Image2 pixel value.

By

- The *scale* value.

With adding from the *offset* value to the division result.

$$Result = (Image1 + Image2) / scale + offset$$

Note

How high the *scale* value is, so high can be the *offset* value because this 2 values are in relationship in the **Add** file merging algorithm.

The *scale* value is the divider from the addition of Image1 and Image2 and the *offset* value is added to the result of the division.

So that you can scale the pixels values and add an offset if you want.

Subtract images interpolating:

For generating an **Subtract** output image file you must:

- Set the input images to interpolate.
- Setting an *scale* value.
- Setting an *offset* value.

And the **Subtract** images merging algorithm will do the work.

The output file is compute, for every pixel by:

The division from

- the subtract of Image1 from Image2 pixel value.

By

- The *scale* value.

With adding from the *offset* value to the division result.

$$Result = (Image1 - Image2) / scale + offset$$

Note

How high the *scale* value is, so high can be the *offset* value because this 2 values are in relationship in the **Subtract** file merging algorithm.

The *scale* value is the divider from the subtraction of Image1 from Image2 and the *offset* value is added to the result of the division.

So that you can scale the pixels values and add an offset if you want.

Edition

Undo-Redo

The Undo and Redo functions can be access throught:

- The Edition menu:

  +-> Edition

  +-> Undo

  +-> Redo

- The toolbar **Undo** and **Redo** button.

You can only Undo an action if:

- You have an file loaded and
- have apply some actions on the image file.

Note

The *Undo* function is not limited in numbers of actions to undone.

You can Redo an undone action, **PyImaging** will immediately redo the undone action.

But if you apply an action on the loaded image file after having undone some actions, you cannot re-access (trought the Redo button) to the image after the last undone action.

The images with their effects are lost.

Rotations

The rotating functions (to the left or the right) can be access throught:

- The Edition menu:

  +-> Edition

  +-> Rotate image to the left

  +-> Rotate image to the right

- The toolbar buttons after the label **Rotate** with the icons indicating an rotation to the left or to the right.

You can only rotate the image from 90° to the left and the right in relationship to the current position from the image.

Note

PyImaging automatically resize the displayed image, if needed in case the image is wider than heigher and you rotate it so that the height become the width, but this resizing operation will not appear if you save the image rotated.

This is an internal operation. It is only an image displaying in the image display container convenience.

Mirroring

The mirroring functions can be access throught:

- The Edition menu:

  +-> Edition

      +-> Flip image up

      +-> Flip image down

      +-> Flip image left

      +-> Flip image right

- The toolbar buttons after the label **Mirror** with the arrows icons indicating an flipping from:
  - Down to up.
  - Up to down.
  - Left to right.
  - Right to left.

Flipping is an horizontal or vertical mirroring operation and you can combine the **Up <-> Down** and the **Left <-> Right** flipping.

Note

You cannot flip an image up if the image has been flipped up before, **PyImaging** will ignore your error.

Effects

Grayscale

The grayscale functions can be access throught:

- The Effects menu:

  +-> Effects

  +-> Apply grayscale

  - **+-> From average pixel value**
  - **+-> From minimal pixel value**
  - **+-> From maximal pixel value**
  - **+-> From red pixel value**
  - **+-> From green pixel value**
  - **+-> From blue pixel value**

- The toolbar button **Grayscale** after setting the grayscale base value trought the tear-down list aside the button.

  From which you can choice from the following items the base used for applying grayscale to the image:

  - **Average**.
  - **Minimum**.
  - **Maximum**.
  - **Red**.
  - **Green**.
  - **Blue**.

Note

The differents bases for grayscale computing: you can choice which pixel value will be used as base for grayscaling your image:

Average....: gray pixel value = Average from the red, green, blue values composing an pixel value.

Minimum...: gray pixel value = Minimal value from red, green or blue values value.

Maximum..: gray pixel value = Maximal value from red, green or blue values.

Red...........: gray pixel value = The value from the red component from the pixel value.

Green.......: gray pixel value = The value from the green component from the pixel value.

Blue..........: gray pixel value = The value from the blue component from the pixel value.

The result is that the image appears entires in gray tones in relationship to the selected base.

Filters

The filters functions can be access throught:

- The Effects menu:

  +-> Effects

  +-> Apply filters

  +-> Blur

  +-> Contour

  +-> Detail

  +-> Edge enhance

  +-> Edge enhance more

  +-> Emboss

  +-> Find edge

  +-> Sharpen

  +-> Smooth

  +-> Smooth more

- The toolbar button **Set filter** after setting the filter to apply trought the tear-down list aside the button.

  From which you can choice from the following items as filter to set:

  - **Blur**.
  - **Contour**.
  - **Detail**.
  - **Edge enhance**.
  - **Edge enhance more**.
  - **Emboss**.
  - **Find edge**.
  - **Sharpen**.
  - **Smooth**.
  - **Smooth more**.

Adjustments

The adjustments functions can be access throught:

- The Effects menu:

  +-> Effects

  +-> Apply adjustments

Which will open the adjustments editor from wich you can access to following settings:

- Color adjustments:

  Change the color balance from an image.
- Brightness adjustments:

  Change the brightness of an image.
- Contrast adjustments:

  Change the contrast of an image.
- Sharpeness adjustments:

  Change the value of the sharpness of an image.

Simply choose the adjustment to perform on the image with the wanted value.

And press the **Apply** button to apply the adjustment on the current loaded image.

Note

All adjustments are similar to the controls of an TV set in their effects.

Colors tools

Colors scaling editor

The Colors scaling editor can be access throught:

- The Colors tools menu:

  +-> Colors tools

  +-> Open color scaling matrix editor

- The toolbar button with an icon at extrem right on the second (down) toolbar.

The interface of the color scaling matrix editor is composed of an table representing for each row the value of an color component from an pixel (Red, Green and Blue).

From which you can set the percent of an pixel color component (Red, Green and blue) to set as value for the color component from the row.

The color scaling matrix has following appearance:

R	R	Red value	G	Green value	B	Blue value
G	R	Red value	G	Green value	B	Blue value
B	R	Red value	G	Green value	B	Blue value

Where you can set the percents values.

And in the button bar under the matrix there an field for setting the percent to apply to the alpha channel value.

After setting values and confirm the following computing is performed on every pixel:

```
Red pixel value   = Red pixel value * Red value + Green pixel value * Green value + Blue pixel value * Blue value

Green pixel value = Red pixel value * Red value + Green pixel value * Green value + Blue pixel value * Blue value

Blue pixel value  = Red pixel value * Red value + Green pixel value * Green value + Blue pixel value * Blue value

Alpha pixel value = Alpha pixel value * Alpha value
```

Simply click on the button **Apply multiply matrix** or the **Apply** button in the **Actions color scaling matrix** button box to launch the computing.

You can click the **Reset matrix** button to restore the start settings.

Note

The matrix configured with the start settings does nothing to the image because the values are set to that:

- The red value is set on 100 % and the others on 0 % for the *Red pixel value* component.
- The green value is set on 100 % and the others on 0 % for the *Green pixel value* component.
- The blue value is set on 100 % and the others on 0 % for the *Blue pixel value* component.
- The alpha value is set on 100 %.

What does not affect the image.

Redscale editor

The Redscale editor editor can be access throught:

- The Colors tools menu:

  +-> Colors tools

  +-> Open redscale editor

The interface from the redscale editor is composed of 3 fields:

- Red scale settings:

  From which you can choice from the following items the base used for applying redscale to the image:

  - **Average**.
  - **Minimum**.
  - **Maximum**.
  - **Red**.
  - **Green**.
  - **Blue**.

Note

The differents bases for redscale computing: you can choice which pixel value will be used as base for redscaling your image:

Average....: red pixel value = Average from the red, green, blue values composing an pixel value.

Minimum...: red pixel value = Minimal value from red, green or blue values value.

Maximum..: red pixel value = Maximal value from red, green or blue values.

Red...........: red pixel value = The value from the red component from the pixel value.

Green.......: red pixel value = The value from the green component from the pixel value.

Blue..........: red pixel value = The value from the blue component from the pixel value.

The result is that the image appears entires in red tones in relationship to the selected base.

- Red scale other colors values:

  From which you can set the value of the others pixels component: green, blue and alpha.

  You have the choice between:

  - Set the value of green, blue and alpha as percent from the red value.
  - set an arbitrary value for green, blue and alpha.
  - Set all the values from green, blue on 0 and not modify the alpha value.

- Action

  The action button box with the **close** and **apply** button who launch the computing.

Greenscale editor

The Greenscale editor editor can be access throught:

- The Colors tools menu:

  +-> Colors tools

  +-> Open greenscale editor

The interface from the greenscale editor is composed of 3 fields:

- Green scale settings:

From which you can choice from the following items the base used for applying greenscale to the image:

  - **Average**.
  - **Minimum**.
  - **Maximum**.
  - **Red**.
  - **Green**.
  - **Blue**.

Note

The differents bases for greenscale computing: you can choice which pixel value will be used as base for greenscaling your image:

Average...: green pixel value = Average from the red, green, blue values composing an pixel value.

Minimum..: green pixel value = Minimal value from red, green or blue values value.

Maximum.: green pixel value = Maximal value from red, green or blue values.

Red..........: green pixel value = The value from the red component from the pixel value.

Green......: green pixel value = The value from the green component from the pixel value.

Blue.........: green pixel value = The value from the blue component from the pixel value.

The result is that the image appears entires in green tones in relationship to the selected base.

- Green scale other colors values:

From which you can set the value of the others pixels component: red, blue and alpha.

You have the choice between:

  - Set the value of red, blue and alpha as percent from the green value.
  - set an arbitrary value for red, blue and alpha.
  - Set all the values from red, blue on 0 and not modify the alpha value.

- Action

The action button box with the **close** and **apply** button who launch the computing.

Bluescale editor

The Bluescale editor editor can be access throught:

- The Colors tools menu:

  +-> Colors tools

  +-> Open bluescale editor

The interface from the bluescale editor is composed of 3 fields:

- Blue scale settings:

From which you can choice from the following items the base used for applying bluescale to the image:

  - **Average**.
  - **Minimum**.
  - **Maximum**.
  - **Red**.
  - **Green**.
  - **Blue**.

Note

The differents bases for bluescale computing: you can choice which pixel value will be used as base for bluescaling your image:

Average....: blue pixel value = Average from the red, green, blue values composing an pixel value.

Minimum...: blue pixel value = Minimal value from red, green or blue values value.

Maximum..: blue pixel value = Maximal value from red, green or blue values.

Red...........: blue pixel value = The value from the red component from the pixel value.

Green.......: blue pixel value = The value from the green component from the pixel value.

Blue..........: blue pixel value = The value from the blue component from the pixel value.

The result is that the image appears entires in blue tones in relationship to the selected base.

- Blue scale other colors values:

From which you can set the value of the others pixels component: red, green and alpha.

You have the choice between:

  - Set the value of red, green and alpha as percent from the green value.
  - set an arbitrary value for red, green and alpha.
  - Set all the values from red, green on 0 and not modify the alpha value.

- Action

The action button box with the **close** and **apply** button who launch the computing.

The right tools side-bar

Color inverter matrix

The Color inverter matrix is located:

- At the top frame of right side-bar.

The Color inverter matrix can be used to invert the pixel component values, to create color exchange and tone setting effects.

The color inverter matrix has following appearance:

R	R	1	G	0	B	0
G	R	0	G	1	B	0
B	R	0	G	0	B	1
<	execute					>

Where you can set one value per row:

- on 1 to activate the color component to set the value from the color of the row on the value of the activate color.

  What permit to exchange color values.
- all values on 0 to disable the color from the row.

The buttons with the arrows permit to change the values of the color inverter matrix to predefined matrixes, which description are available by position the mouse curser over the frame, what display an tooltip bubble describing the effect of the current set matrix.

After setting values and pressing the **Compute** button the following computing is performed on every pixel:

```
Red pixel value   = value from the activate pixel color component (R, G or B) or 0 if none is set.

Green pixel value = value from the activate pixel color component (R, G or B) or 0 if none is set.

Blue pixel value  = value from the activate pixel color component (R, G or B) or 0 if none is set.
```

Note

The predefined matrix descriptions tells what the image **should** look at after the treatment but it depending of the content from the image file and it is not ever exact.

The color inverter matrix is slow relative to the other pixels buffered treatment because it is heavy and the execution speed value is arbitrary decrease during the treatment and restored after.

Change intensity

The change intensity frame is located:

- At the bottom from the right side-bar.

The change intensity can be used to change the color intensity from:

- The colors and the alpha channel.
- Change the global color intensity, all colors and the alpha channel are scaled with the same factor.

Note

You can use the arrows from the keyboard for accuracy values settings for the widget having the focus.

And the **Apply intensity change** button launch the computing from the last changed values, even:

- The colors and alpha channel settings.
- The global color intensity.

After setting values and pressing the **Apply intensity change** button the following computing is performed:

An intensity factor is computed for every value depending from your settings and the pixels values are computed following:

```
red pixel value   = red intensity change factor   * red pixel value

green pixel value = green intensity change factor * green pixel value

blue pixel value  = blue intensity change factor  * blue pixel value

alpha pixel value = alpha intensity change factor * alpha pixel value
```

execution speed

The execution speed can be set throught:

- The Settings menu:

  +-> Settings

      +-> execution speed

          +-> 10 %

          +-> 20 %

          +-> 30 %

          +-> 40 %

          +-> 50 %

          +-> 60 %

          +-> 70 %

          +-> 80 %

          +-> 90 %

          +-> 100 %

This value determine how many pixels are threat into an millisecond by buffered pixels computing operations, displaying an progressbar.

And is to set in relationship of the speed from your computer.

The default value is 50.

Note

Everytime you change this value, the value is register into your settings, so that you don't have to change it everytime you start the program.

This value is arbitrary set on 10 by the color inverter matrix which treatment is heavy and restore after the operation is complete. You can pick an editor and change it if you practice the python programmatic.

If your computer don't support the setting speed the side effect are an anormal displaying of the progressbar, but it has not affect the treatment on my machine as i sea that the color inverter matrix treatment is too heavy to support more than 20 % on my computer.

About

About PyImaging

The information window about PyImaging can be access throught:

- The About menu:

  +-> About

  +-> About PyImaging

- The toolbar button with an icon at extrem right on the first (upper) toolbar.

This window display:

- The program name.
- The program version.
- The credits.
- The license under the program is distributed.
- And display the program icon.

Credits

Program:	PyImaging.
Version:	1.0.0
License:	GPLv3
Author:	Eddie Bruggemann <mrcyberfighter@gmail.com>.
Documenter:	Eddie Bruggemann <mrcyberfighter@gmail.com>.
Artist:	Eddie Bruggemann <mrcyberfighter@gmail.com>.
Corrections:	None, Excuse to the English language for the offenses consisting of the errors contains into this document.
Date:	06 July 2014.
Credits:	Thank's to my mother and to the doctors. Keep away from drugs, drugs destroy your brain and your life.