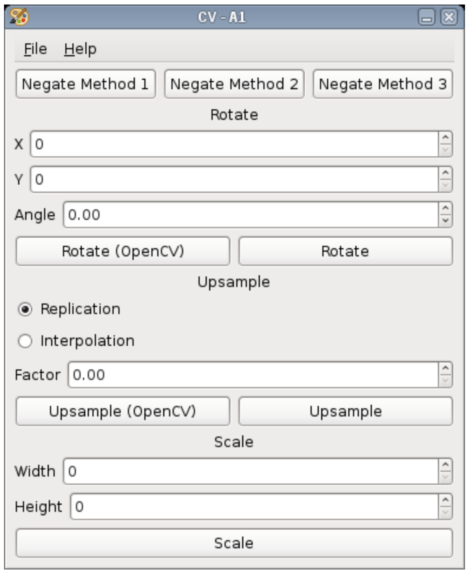
Computer Vision
Assignment 1
German University in Cairo
Frederic-Gerald Morcos
4-1805
E13

## Introduction

The user interface is designed in Glade, converted to GtkBuilder XML using the tool `gtk-builder-convert` and run using the Gtk+ user interface toolkit. The Open Computer Vision library handles image management.



X 0
Y 0
Angle 0.00
● Replication
Factor 0.00
Width 0
Height 0

## Question 1

The application can load image files in **PNG** and **BMP** formats and save to **JPG** and **TIFF** formats. This can be done by accessing the `File > Open` and `File > Save` menu buttons.

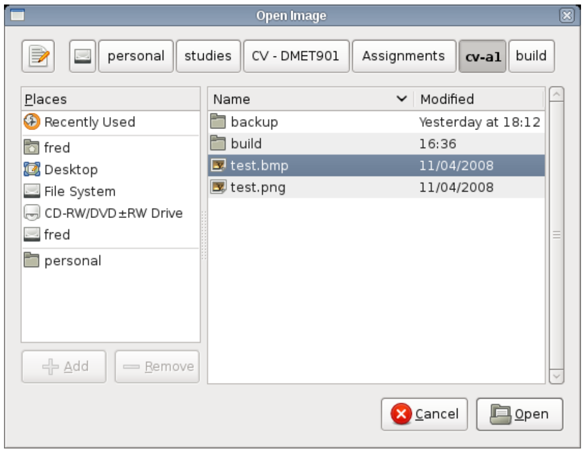
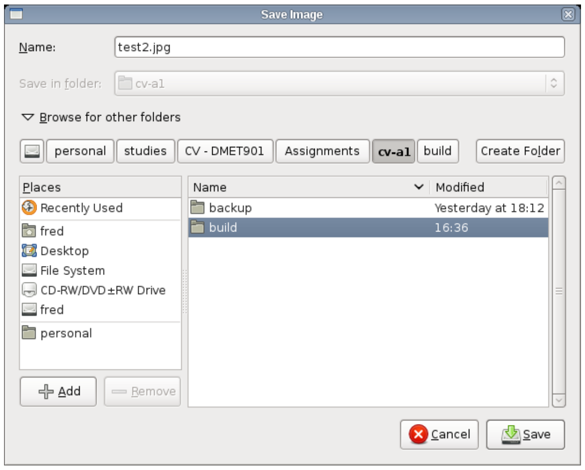


*Illustration 2: Open Dialog*



Name: test2.jpg

*Illustration 1: Save Dialog*

**Question 2**
Negation is implemented in three ways. Negation is done by flipping the bits of a binary number (representing the value of a channel in a pixel).

The first method uses direct access to `IplImage*->imageData` but deals with the data as a two-dimensional matrix of pixel where

```
value(i, j, k) = I * widthStep + j * nChannels + k
```

where I and j are the pixel coordinates and k representing B, G, R and A by 0, 1, 2 and 3 respectively.

The second method uses indirect access by using functions from OpenCV. Namely `cvGet2D()` and `cvSet2D()`, treating the image as a two-dimensional matrix of pixels.

The third method and the fastest treats the image as a one-dimensional array of pixels, iterating over `IplImage*->imageData` and inverting every value. The length of `imageData` is equal to the number of pixels in the image times the number of channels (`width * height * nChannels`).

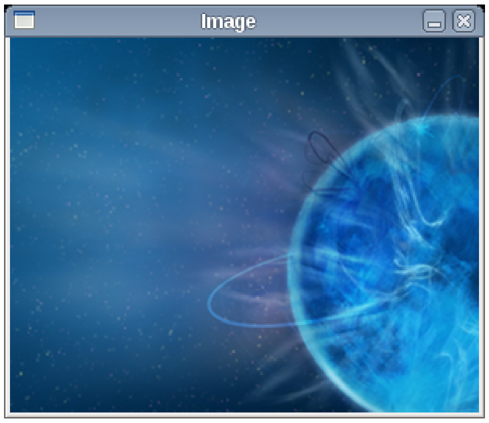


*Illustration 3: Normal Image*

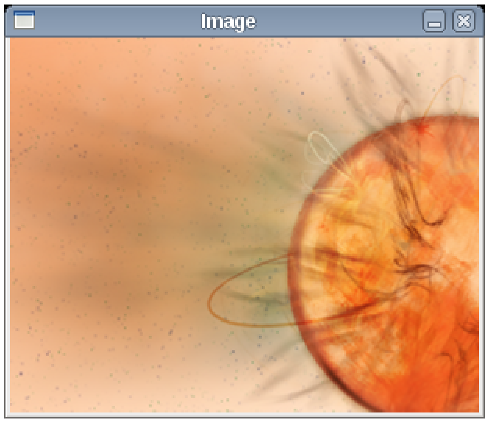


*Illustration 4: Negated Image*

**Question 3**
Rotation around an arbitrary point is done by creating a transformation matrix composed of translation to the origin (whereas the rotation point coincides with the point of origin), rotation by an arbitrary angle and translation back to the point of rotation. If M$_1$, M$_2$ and M$_3$ are the translation to origin, rotation around angle and translation back matrices respectively, then the composed transformation matrix M = M$_1$M$_2$M$_3$.

$$M = \begin{bmatrix} \cos(angle) & -\sin(angle) & ((1-\cos(angle))*x)+(\sin(angle)*y) \\ \sin(angle) & \cos(angle) & (\sin(angle)*-x)+((1-\cos(angle))*y) \\ 0 & 0 & 1 \end{bmatrix}$$

In practice, we have source and destination images. We need to iterate over all the positions in the destination image and get the respective intensities of each channel for each pixel from the source image. To do so, we calculate the inverse of the matrix M and multiply it by every

(I, j) pair, resulting in the original position in the source image (before rotation and translation) from which we can extract the intensity. If the resulting point is outside the source image, we fill the corresponding pixel intensity in the destination image with an arbitrary color (black).

**Interpolation**

A problem arises, if the resulting point is of a floating position (in either x, y or both), we need to calculate the weighted average of the pixel intensities around this point. This is done by using interpolation which multiplies the left and right pixel intensities with the inverted distance between each pixel and the floating x value. The same happens with the y value by interpolating the resulting x intensities.

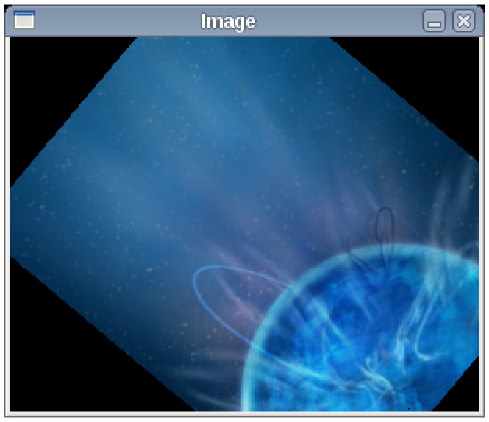


*Illustration 5: Rotated Image*

**Question 4**
Upsampling by replication is done by copying the pixel intensities from a source image several times (aligned) into a destination image, creating a larger replica (by an arbitrary factor affecting the number of replica pixels) of the source image. We loop over the destination image, reading pixel intensities from the corresponding pixels in the source image and writing them to the destination image.

```
intensity(src_x, src_y) =
intensity(floor(dst_x / factor), floor(dst_y / factor))
```

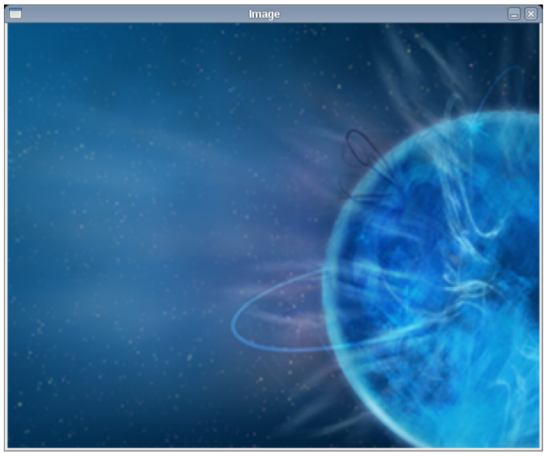


*Illustration 6: Scaled Image*

**The remaining questions have been implemented using OpenCV built-in functions.**

**References**

http://www.gtk.org/
http://glade.gnome.org/
http://opencvlibrary.sourceforge.net/

**Source Code**

**ui.glade**

```
<?xml version="1.0"?>
<glade-interface>
  <!-- interface-requires gtk+ 2.14 -->
  <!-- interface-naming-policy project-wide -->
  <widget class="GtkWindow" id="mainWindow">
    <property name="border_width">5</property>
    <property name="title" translatable="yes">CV - A1</property>
    <property name="resizable">False</property>
    <property name="icon_name">applications-graphics</property>
    <child>
      <widget class="GtkVBox" id="vbox1">
        <property name="visible">True</property>
        <property name="spacing">5</property>
        <child>
          <widget class="GtkMenuBar" id="menubar1">
            <property name="visible">True</property>
            <child>
              <widget class="GtkMenuItem" id="menuitem1">
                <property name="visible">True</property>
                <property name="label" translatable="yes">_File</property>
                <property name="use_underline">True</property>
                <child>
                  <widget class="GtkMenu" id="menu1">
                    <property name="visible">True</property>
                    <child>
                      <widget class="GtkImageMenuItem" id="menuOpen">
                        <property name="label">gtk-open</property>
                        <property name="visible">True</property>
                        <property name="use_underline">True</property>
                        <property name="use_stock">True</property>
                      </widget>
                    </child>
                    <child>
                      <widget class="GtkImageMenuItem" id="menuSave">
                        <property name="label">gtk-save</property>
                        <property name="visible">True</property>
                        <property name="use_underline">True</property>
                        <property name="use_stock">True</property>
                      </widget>
                    </child>
                    <child>
                      <widget class="GtkSeparatorMenuItem"
id="separatormenuitem1">
                        <property name="visible">True</property>
                      </widget>
                    </child>
                    <child>
                      <widget class="GtkImageMenuItem" id="menuQuit">
                        <property name="label">gtk-quit</property>
                        <property name="visible">True</property>
                        <property name="use_underline">True</property>
                        <property name="use_stock">True</property>
                      </widget>
                    </child>
                  </widget>
                </child>
              </widget>
            </child>
            <child>
              <widget class="GtkMenuItem" id="menuitem4">
                <property name="visible">True</property>
                <property name="label" translatable="yes">_Help</property>
                <property name="use_underline">True</property>
                <child>
                  <widget class="GtkMenu" id="menu3">
                    <property name="visible">True</property>
                    <child>
                      <widget class="GtkImageMenuItem" id="menuAbout">
                        <property name="label">gtk-about</property>
                        <property name="visible">True</property>
                        <property name="use_underline">True</property>
                        <property name="use_stock">True</property>
                      </widget>
                    </child>
                  </widget>
                </child>
              </widget>
            </child>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">0</property>
          </packing>
        </child>
        <child>
          <widget class="GtkHBox" id="hbox4">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <property name="homogeneous">True</property>
            <child>
              <widget class="GtkButton" id="negateButton1">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Negate Method
1</property>
              </widget>
              <packing>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkButton" id="negateButton2">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Negate Method 
2</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
            <child>
              <widget class="GtkButton" id="negateButton3">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Negate Method 
3</property>
              </widget>
              <packing>
                <property name="position">2</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="position">1</property>
          </packing>
        </child>
        <child>
          <widget class="GtkLabel" id="label1">
            <property name="visible">True</property>
            <property name="label" translatable="yes">Rotate</property>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">2</property>
          </packing>
        </child>
        <child>
          <widget class="GtkHBox" id="hbox1">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <widget class="GtkLabel" id="label2">
                <property name="visible">True</property>
                <property name="label" translatable="yes">X</property>
              </widget>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkSpinButton" id="rotateXSpin">
```

```
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">0 0 1000 1 10 0</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">3</property>
          </packing>
        </child>
        <child>
          <widget class="GtkHBox" id="hbox2">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <widget class="GtkLabel" id="label3">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Y</property>
              </widget>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkSpinButton" id="rotateYSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">0 0 1000 1 10 0</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">4</property>
          </packing>
        </child>
        <child>
          <widget class="GtkHBox" id="hbox3">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <widget class="GtkLabel" id="label4">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Angle</property>
              </widget>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkSpinButton" id="rotateAngleSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">0 -180 180 1 10 0</property>
                <property name="digits">2</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">5</property>
          </packing>
        </child>
        <child>
          <widget class="GtkHBox" id="hbox8">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <property name="homogeneous">True</property>
            <child>
              <widget class="GtkButton" id="rotateButtonCV">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Rotate
(OpenCV)</property>
              </widget>
              <packing>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkButton" id="rotateButton">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Rotate</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="position">6</property>
          </packing>
        </child>
        <child>
```

```
          <widget class="GtkLabel" id="label5">
            <property name="visible">True</property>
            <property name="label" translatable="yes">Upsample</property>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">7</property>
          </packing>
        </child>
        <child>
          <widget class="GtkRadioButton" id="replicationRadio">
            <property name="visible">True</property>
            <property name="can_focus">True</property>
            <property name="receives_default">False</property>
            <property name="active">True</property>
            <property name="draw_indicator">True</property>
            <property name="label" translatable="yes">Replication</property>
          </widget>
          <packing>
            <property name="position">8</property>
          </packing>
        </child>
        <child>
          <widget class="GtkRadioButton" id="interpolationRadio">
            <property name="visible">True</property>
            <property name="can_focus">True</property>
            <property name="receives_default">False</property>
            <property name="active">True</property>
            <property name="draw_indicator">True</property>
            <property name="group">replicationRadio</property>
            <property name="label" translatable="yes">Interpolation</property>
          </widget>
          <packing>
            <property name="position">9</property>
          </packing>
        </child>
        <child>
          <widget class="GtkHBox" id="hbox5">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <widget class="GtkLabel" id="label7">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Factor</property>
              </widget>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkSpinButton" id="upsampleFactorSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">0 0 100 1 10 0</property>
                <property name="digits">2</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">10</property>
          </packing>
        </child>
        <child>
          <widget class="GtkHBox" id="hbox9">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <property name="homogeneous">True</property>
            <child>
              <widget class="GtkButton" id="upsampleButtonCV">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Upsample
(OpenCV)</property>
              </widget>
              <packing>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkButton" id="upsampleButton">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Upsample</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="position">11</property>
          </packing>
        </child>
        <child>
          <widget class="GtkLabel" id="label8">
            <property name="visible">True</property>
            <property name="label" translatable="yes">Scale</property>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">12</property>
          </packing>
        </child>
```

```xml
        <child>
          <widget class="GtkHBox" id="hbox6">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <widget class="GtkLabel" id="label9">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Width</property>
              </widget>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkSpinButton" id="scaleWidthSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">0 0 1000 1 10 0</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">13</property>
          </packing>
        </child>
        <child>
          <widget class="GtkHBox" id="hbox7">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <widget class="GtkLabel" id="label10">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Height</property>
              </widget>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <widget class="GtkSpinButton" id="scaleHeightSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">0 0 1000 1 10 0</property>
              </widget>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">14</property>
          </packing>
        </child>
        <child>
          <widget class="GtkButton" id="scaleButton">
            <property name="visible">True</property>
            <property name="can_focus">True</property>
            <property name="receives_default">True</property>
            <property name="label" translatable="yes">Scale</property>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">15</property>
          </packing>
        </child>
      </widget>
    </child>
  </widget>
  <widget class="GtkAboutDialog" id="aboutDialog">
    <property name="border_width">5</property>
    <property name="title" translatable="yes">About</property>
    <property name="destroy_with_parent">True</property>
    <property name="type_hint">normal</property>
    <property name="skip_taskbar_hint">True</property>
    <property name="skip_pager_hint">True</property>
    <property name="transient_for">mainWindow</property>
    <property name="program_name">CV - A1</property>
    <property name="version">0.1</property>
    <property name="copyright" translatable="yes">Copyright (C) 2008 Frederic
Morcos</property>
    <property name="comments" translatable="yes">Showcasing OpenCV
Features</property>
    <property name="website">http://fredmorcos.googlecode.com/</property>
    <property name="website_label"
translatable="yes">http://fredmorcos.googlecode.com/</property>
    <property name="license" translatable="yes">Licensed under the
GPLv3.</property>
    <property name="authors">Frederic Morcos
&lt;fred.morcos@gmail.com&gt;</property>
    <property name="documenters">Frederic Morcos
&lt;fred.morcos@gmail.com&gt;</property>
    <property name="artists"></property>
    <child internal-child="vbox">
      <widget class="GtkVBox" id="dialog-vbox1">
        <property name="visible">True</property>
        <property name="spacing">2</property>
        <child>
          <placeholder/>
        </child>
        <child internal-child="action_area">
          <widget class="GtkHButtonBox" id="dialog-action_area1">
            <property name="visible">True</property>
            <property name="layout_style">end</property>
          </widget>
          <packing>
            <property name="expand">False</property>
            <property name="pack_type">end</property>
            <property name="position">0</property>
          </packing>
        </child>
      </widget>
    </child>
  </widget>
</glade-interface>
```

## ui.xml

```
<?xml version="1.0"?>
<interface>
  <object class="GtkAdjustment" id="adjustment1">
    <property name="upper">1000</property>
    <property name="lower">0</property>
    <property name="page_increment">10</property>
    <property name="step_increment">1</property>
    <property name="page_size">0</property>
    <property name="value">0</property>
  </object>
  <object class="GtkAdjustment" id="adjustment2">
    <property name="upper">1000</property>
    <property name="lower">0</property>
    <property name="page_increment">10</property>
    <property name="step_increment">1</property>
    <property name="page_size">0</property>
    <property name="value">0</property>
  </object>
  <object class="GtkAdjustment" id="adjustment3">
    <property name="upper">180</property>
    <property name="lower">-180</property>
    <property name="page_increment">10</property>
    <property name="step_increment">1</property>
    <property name="page_size">0</property>
    <property name="value">0</property>
  </object>
  <object class="GtkAdjustment" id="adjustment4">
    <property name="upper">100</property>
    <property name="lower">0</property>
    <property name="page_increment">10</property>
    <property name="step_increment">1</property>
    <property name="page_size">0</property>
    <property name="value">0</property>
  </object>
  <object class="GtkAdjustment" id="adjustment5">
    <property name="upper">1000</property>
    <property name="lower">0</property>
    <property name="page_increment">10</property>
    <property name="step_increment">1</property>
    <property name="page_size">0</property>
    <property name="value">0</property>
  </object>
  <object class="GtkAdjustment" id="adjustment6">
    <property name="upper">1000</property>
    <property name="lower">0</property>
    <property name="page_increment">10</property>
    <property name="step_increment">1</property>
    <property name="page_size">0</property>
    <property name="value">0</property>
  </object>
  <object class="GtkUIManager" id="uimanager1">
    <child>
      <object class="GtkActionGroup" id="actiongroup1">
        <child>
          <object class="GtkAction" id="menuitem1">
            <property name="name">menuitem1</property>
            <property name="label" translatable="yes">_File</property>
          </object>
        </child>
        <child>
          <object class="GtkAction" id="menuOpen">
            <property name="stock_id">gtk-open</property>
            <property name="name">menuOpen</property>
          </object>
        </child>
        <child>
          <object class="GtkAction" id="menuSave">
            <property name="stock_id">gtk-save</property>
            <property name="name">menuSave</property>
          </object>
        </child>
        <child>
          <object class="GtkAction" id="menuQuit">
            <property name="stock_id">gtk-quit</property>
            <property name="name">menuQuit</property>
          </object>
        </child>
        <child>
          <object class="GtkAction" id="menuitem4">
            <property name="name">menuitem4</property>
            <property name="label" translatable="yes">_Help</property>
          </object>
        </child>
        <child>
          <object class="GtkAction" id="menuAbout">
            <property name="stock_id">gtk-about</property>
            <property name="name">menuAbout</property>
          </object>
        </child>
      </object>
    </child>
    <ui>
      <menubar name="menubar1">
        <menu action="menuitem1">
          <menuitem action="menuOpen"/>
          <menuitem action="menuSave"/>
          <separator/>
          <menuitem action="menuQuit"/>
        </menu>
        <menu action="menuitem4">
          <menuitem action="menuAbout"/>
        </menu>
      </menubar>
    </ui>
  </object>
  <!-- interface-requires gtk+ 2.14 -->
  <!-- interface-naming-policy project-wide -->
  <object class="GtkWindow" id="mainWindow">
    <property name="border_width">5</property>
    <property name="title" translatable="yes">CV - A1</property>
    <property name="resizable">False</property>
    <property name="icon_name">applications-graphics</property>
    <child>
      <object class="GtkVBox" id="vbox1">
        <property name="visible">True</property>
        <property name="spacing">5</property>
        <child>
          <object class="GtkMenuBar" constructor="uimanager1" id="menubar1">
            <property name="visible">True</property>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">0</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox4">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <property name="homogeneous">True</property>
            <child>
              <object class="GtkButton" id="negateButton1">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Negate Method
1</property>
              </object>
              <packing>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkButton" id="negateButton2">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Negate Method
2</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
            <child>
              <object class="GtkButton" id="negateButton3">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Negate Method
3</property>
              </object>
              <packing>
                <property name="position">2</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="position">1</property>
          </packing>
        </child>
        <child>
          <object class="GtkLabel" id="label1">
            <property name="visible">True</property>
            <property name="label" translatable="yes">Rotate</property>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">2</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox1">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <object class="GtkLabel" id="label2">
                <property name="visible">True</property>
                <property name="label" translatable="yes">X</property>
              </object>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkSpinButton" id="rotateXSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">adjustment1</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="expand">False</property>
```

```xml
            <property name="fill">False</property>
            <property name="position">3</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox2">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <object class="GtkLabel" id="label3">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Y</property>
              </object>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkSpinButton" id="rotateYSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">adjustment2</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">4</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox3">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <object class="GtkLabel" id="label4">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Angle</property>
              </object>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkSpinButton" id="rotateAngleSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">adjustment3</property>
                <property name="digits">2</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">5</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox8">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <property name="homogeneous">True</property>
            <child>
              <object class="GtkButton" id="rotateButtonCV">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Rotate
(OpenCV)</property>
              </object>
              <packing>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkButton" id="rotateButton">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Rotate</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="position">6</property>
          </packing>
        </child>
        <child>
          <object class="GtkLabel" id="label5">
            <property name="visible">True</property>
            <property name="label" translatable="yes">Upsample</property>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">7</property>
          </packing>
        </child>
        <child>
```

```
          <object class="GtkRadioButton" id="replicationRadio">
            <property name="visible">True</property>
            <property name="can_focus">True</property>
            <property name="receives_default">False</property>
            <property name="active">True</property>
            <property name="draw_indicator">True</property>
            <property name="label" translatable="yes">Replication</property>
          </object>
          <packing>
            <property name="position">8</property>
          </packing>
        </child>
        <child>
          <object class="GtkRadioButton" id="interpolationRadio">
            <property name="visible">True</property>
            <property name="can_focus">True</property>
            <property name="receives_default">False</property>
            <property name="active">True</property>
            <property name="draw_indicator">True</property>
            <property name="group">replicationRadio</property>
            <property name="label" translatable="yes">Interpolation</property>
          </object>
          <packing>
            <property name="position">9</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox5">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <object class="GtkLabel" id="label7">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Factor</property>
              </object>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkSpinButton" id="upsampleFactorSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">adjustment4</property>
                <property name="digits">2</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">10</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox9">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <property name="homogeneous">True</property>
            <child>
              <object class="GtkButton" id="upsampleButtonCV">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Upsample
(OpenCV)</property>
              </object>
              <packing>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkButton" id="upsampleButton">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
                <property name="label" translatable="yes">Upsample</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="position">11</property>
          </packing>
        </child>
        <child>
          <object class="GtkLabel" id="label8">
            <property name="visible">True</property>
            <property name="label" translatable="yes">Scale</property>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">12</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox6">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <object class="GtkLabel" id="label9">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Width</property>
              </object>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkSpinButton" id="scaleWidthSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">adjustment5</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">13</property>
          </packing>
        </child>
        <child>
          <object class="GtkHBox" id="hbox7">
            <property name="visible">True</property>
            <property name="spacing">5</property>
            <child>
              <object class="GtkLabel" id="label10">
                <property name="visible">True</property>
                <property name="label" translatable="yes">Height</property>
              </object>
              <packing>
                <property name="expand">False</property>
                <property name="fill">False</property>
                <property name="position">0</property>
              </packing>
            </child>
            <child>
              <object class="GtkSpinButton" id="scaleHeightSpin">
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="adjustment">adjustment6</property>
              </object>
              <packing>
                <property name="position">1</property>
              </packing>
            </child>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">14</property>
          </packing>
        </child>
        <child>
          <object class="GtkButton" id="scaleButton">
            <property name="visible">True</property>
            <property name="can_focus">True</property>
            <property name="receives_default">True</property>
            <property name="label" translatable="yes">Scale</property>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="fill">False</property>
            <property name="position">15</property>
          </packing>
        </child>
      </object>
    </child>
  </object>
  <object class="GtkAboutDialog" id="aboutDialog">
    <property name="border_width">5</property>
    <property name="title" translatable="yes">About</property>
    <property name="destroy_with_parent">True</property>
    <property name="type_hint">normal</property>
    <property name="skip_taskbar_hint">True</property>
    <property name="skip_pager_hint">True</property>
    <property name="transient_for">mainWindow</property>
    <property name="program_name">CV - A1</property>
    <property name="version">0.1</property>
    <property name="copyright" translatable="yes">Copyright (C) 2008 Frederic
Morcos</property>
    <property name="comments" translatable="yes">Showcasing OpenCV
Features</property>
    <property name="website">http://fredmorcos.googlecode.com/</property>
    <property name="website_label"
translatable="yes">http://fredmorcos.googlecode.com/</property>
    <property name="license" translatable="yes">Licensed under the
GPLv3.</property>
    <property name="authors">Frederic Morcos
&lt;fred.morcos@gmail.com&gt;</property>
    <property name="documenters">Frederic Morcos
&lt;fred.morcos@gmail.com&gt;</property>
    <property name="artists"/>
    <child internal-child="vbox">
      <object class="GtkVBox" id="dialog-vbox1">
        <property name="visible">True</property>
        <property name="spacing">2</property>
        <child>
          <placeholder/>
        </child>
        <child internal-child="action_area">
          <object class="GtkHButtonBox" id="dialog-action_area1">
            <property name="visible">True</property>
            <property name="layout_style">end</property>
          </object>
          <packing>
            <property name="expand">False</property>
            <property name="pack_type">end</property>
            <property name="position">0</property>
          </packing>
        </child>
      </object>
    </child>
  </object>
```

```
</interface>
```

## main.c

```
/*
        This file is part of cv-a1.

        Copyright (C) 2008      Frederic-Gerald Morcos <fred.morcos@gmail.com>

        cv-a1 is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        cv-a1 is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU General Public License for more details.

        You should have received a copy of the GNU General Public License
        along with cv-a1.  If not, see <http://www.gnu.org/licenses/>.
*/

#include "ui-builder.h"
#include "callbacks.h"
#include "cv-stuff.h"
#include <gtk/gtk.h>

int main (int argc, char *argv[]) {
        GtkWidget       *mainWindow,
                                *negateButton1,
                                *negateButton2,
                                *negateButton3,
                                *rotateButton,
                                *rotateButtonCV,
                                *upsampleButton,
                                *upsampleButtonCV,
                                *scaleButton,
                                *replicationRadio;
        GtkAction       *menuOpen,
                                *menuSave,
                                *menuQuit,
                                *menuAbout;

        gtk_init(&argc, &argv);

        ui_init();
        mainWindow = ui_get_widget("mainWindow");
        negateButton1 = ui_get_widget("negateButton1");
        negateButton2 = ui_get_widget("negateButton2");
        negateButton3 = ui_get_widget("negateButton3");
        rotateButton = ui_get_widget("rotateButton");
        rotateButtonCV = ui_get_widget("rotateButtonCV");
        upsampleButton = ui_get_widget("upsampleButton");
        upsampleButtonCV = ui_get_widget("upsampleButtonCV");
        scaleButton = ui_get_widget("scaleButton");
        replicationRadio = ui_get_widget("replicationRadio");

        menuOpen = ui_get_action("menuOpen");
        menuSave = ui_get_action("menuSave");
        menuQuit = ui_get_action("menuQuit");
        menuAbout = ui_get_action("menuAbout");

        /* connect signals */
        g_signal_connect(G_OBJECT(mainWindow), "delete-event",
                         G_CALLBACK(gtk_main_quit), NULL);
        g_signal_connect(G_OBJECT(menuQuit), "activate",
                         G_CALLBACK(gtk_main_quit), NULL);
        g_signal_connect(G_OBJECT(menuOpen), "activate",
                         G_CALLBACK(menuOpen_activate), NULL);
        g_signal_connect(G_OBJECT(menuSave), "activate",
                         G_CALLBACK(menuSave_activate), NULL);
        g_signal_connect(G_OBJECT(menuAbout), "activate",
                         G_CALLBACK(menuAbout_activate), NULL);
        g_signal_connect(G_OBJECT(negateButton1), "clicked",
                         G_CALLBACK(negateButton1_click), NULL);
        g_signal_connect(G_OBJECT(negateButton2), "clicked",
                         G_CALLBACK(negateButton2_click), NULL);
        g_signal_connect(G_OBJECT(negateButton3), "clicked",
                         G_CALLBACK(negateButton3_click), NULL);
        g_signal_connect(G_OBJECT(rotateButton), "clicked",
                         G_CALLBACK(rotateButton_click), NULL);
        g_signal_connect(G_OBJECT(rotateButtonCV), "clicked",
                         G_CALLBACK(rotateButtonCV_click), NULL);
        g_signal_connect(G_OBJECT(upsampleButton), "clicked",
                         G_CALLBACK(upsampleButton_click), NULL);
        g_signal_connect(G_OBJECT(upsampleButtonCV), "clicked",
                         G_CALLBACK(upsampleButtonCV_click), NULL);
        g_signal_connect(G_OBJECT(scaleButton), "clicked",
                         G_CALLBACK(scaleButton_click), NULL);
        g_signal_connect(G_OBJECT(replicationRadio), "clicked",
                         G_CALLBACK(replicationRadio_toggle), NULL);

        gtk_widget_show_all(mainWindow);
        gtk_main();

        ui_destroy();
        cv_stuff_destroy();

        return 0;
}
```

### callbacks.h

```
/*
        This file is part of cv-a1.

        Copyright (C) 2008      Frederic-Gerald Morcos <fred.morcos@gmail.com>

        cv-a1 is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        cv-a1 is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU General Public License for more details.

        You should have received a copy of the GNU General Public License
        along with cv-a1.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef __CALLBACKS_H__
#define __CALLBACKS_H__

#include <gtk/gtk.h>

void menuAbout_activate(GtkMenuItem *, gpointer);
void menuOpen_activate(GtkMenuItem *, gpointer);
void menuSave_activate(GtkMenuItem *, gpointer);
void negateButton1_click(GtkButton *, gpointer);
void negateButton3_click(GtkButton *, gpointer);
void negateButton2_click(GtkButton *, gpointer);
void rotateButton_click(GtkButton *, gpointer);
void rotateButtonCV_click(GtkButton *, gpointer);
void upsampleButton_click(GtkButton *, gpointer);
void upsampleButtonCV_click(GtkButton *, gpointer);
void scaleButton_click(GtkButton *, gpointer);
void replicationRadio_toggle(GtkToggleButton *, gpointer);

#endif  /* __CALLBACKS_H__ */
```

**callbacks.c**

```
/*
        This file is part of cv-a1.

        Copyright (C) 2008      Frederic-Gerald Morcos <fred.morcos@gmail.com>

        cv-a1 is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        cv-a1 is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU General Public License for more details.

        You should have received a copy of the GNU General Public License
        along with cv-a1.  If not, see <http://www.gnu.org/licenses/>.
*/

#include "callbacks.h"
```

```
#include "ui-builder.h"
#include "cv-stuff.h"
#include <gtk/gtk.h>
#include <string.h>
#include <math.h>

void replicationRadio_toggle(GtkToggleButton *button, gpointer data) {
        GtkWidget *upsampleButton;

        upsampleButton = ui_get_widget("upsampleButton");
        gtk_widget_set_sensitive(upsampleButton,
                        gtk_toggle_button_get_active(button));
}

void scaleButton_click(GtkButton *button, gpointer data) {
        int             height,
                        width;

        height = gtk_spin_button_get_value_as_int(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("scaleHeightSpin")));
        width = gtk_spin_button_get_value_as_int(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("scaleWidthSpin")));

        cv_stuff_scale(width, height);
}

void upsampleButtonCV_click(GtkButton *button, gpointer data) {
        gboolean        replication;
        double          factor;

        replication = gtk_toggle_button_get_active(
                        GTK_TOGGLE_BUTTON(
                                ui_get_widget("replicationRadio")));
        factor = gtk_spin_button_get_value(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("upsampleFactorSpin")));

        cv_stuff_upsample_opencv(replication, factor);
}

void upsampleButton_click(GtkButton *button, gpointer data) {
        gboolean                        replication;
        double                          factor;
        GtkMessageDialog        *dialog;

        replication = gtk_toggle_button_get_active(
                        GTK_TOGGLE_BUTTON(
                                ui_get_widget("replicationRadio")));
        factor = gtk_spin_button_get_value(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("upsampleFactorSpin")));

        if (isFloat(factor)) {
                factor = floor(factor);
                dialog = gtk_message_dialog_new(
                                GTK_WINDOW(ui_get_widget("mainWindow")),
                                GTK_DIALOG_DESTROY_WITH_PARENT,
                                GTK_MESSAGE_INFO,
                                GTK_BUTTONS_OK,
                                "Factor will be rounded to %.1f.",
                                factor);
                gtk_dialog_run(GTK_DIALOG(dialog));
                gtk_widget_destroy(GTK_WIDGET(dialog));
        }

        cv_stuff_upsample(replication, factor);
}

void rotateButton_click(GtkButton *button, gpointer data) {
        int             x,
                        y;
        double  angle;

        x = gtk_spin_button_get_value_as_int(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("rotateXSpin")));
        y = gtk_spin_button_get_value_as_int(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("rotateYSpin")));
        angle = gtk_spin_button_get_value(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("rotateAngleSpin")));

        cv_stuff_rotate(x, y, angle);
}

void rotateButtonCV_click(GtkButton *button, gpointer data) {
        int             x,
                        y;
        double  angle;

        x = gtk_spin_button_get_value_as_int(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("rotateXSpin")));
        y = gtk_spin_button_get_value_as_int(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("rotateYSpin")));
        angle = gtk_spin_button_get_value(
                        GTK_SPIN_BUTTON(
                                ui_get_widget("rotateAngleSpin")));

        cv_stuff_rotate_opencv(x, y, angle);
}

void negateButton1_click(GtkButton *button, gpointer data) {
        cv_stuff_negate1();
}

void negateButton3_click(GtkButton *button, gpointer data) {
        cv_stuff_negate3();
}
```

```
void negateButton2_click(GtkButton *button, gpointer data) {
        cv_stuff_negate2();
}

void menuAbout_activate(GtkMenuItem *item, gpointer data) {
        GtkWidget       *aboutDialog;

        aboutDialog = ui_get_widget("aboutDialog");

        if (gtk_dialog_run(GTK_DIALOG(aboutDialog)) ==
                        GTK_RESPONSE_CANCEL)
                gtk_widget_hide(aboutDialog);
}

void menuOpen_activate(GtkMenuItem *item, gpointer data) {
        GtkWidget               *openDialog;
        GtkFileFilter   *filter;
        char                    *filename;

        filter = gtk_file_filter_new();
        gtk_file_filter_add_pattern(filter, "*.png");
        gtk_file_filter_add_pattern(filter, "*.bmp");

        openDialog = gtk_file_chooser_dialog_new(
                        "Open Image",
                        GTK_WINDOW(ui_get_widget("mainWindow")),
                        GTK_FILE_CHOOSER_ACTION_OPEN,
                        GTK_STOCK_CANCEL, GTK_RESPONSE_CANCEL,
                        GTK_STOCK_OPEN, GTK_RESPONSE_ACCEPT,
                        NULL);
        gtk_file_chooser_set_filter(
                        GTK_FILE_CHOOSER(openDialog), filter);

        if (gtk_dialog_run(GTK_DIALOG(openDialog)) ==
                        GTK_RESPONSE_ACCEPT) {
                filename =
gtk_file_chooser_get_filename(GTK_FILE_CHOOSER(openDialog));
                cv_stuff_window(filename);
        }

        gtk_widget_hide(openDialog);
}

void menuSave_activate(GtkMenuItem *item, gpointer data) {
        GtkWidget               *saveDialog;
        GtkFileFilter   *filter;
        char                    *filename,
                                        *ext;
        GString                 *tmp;

        filter = gtk_file_filter_new();
        gtk_file_filter_add_pattern(filter, "*.jpg");
        gtk_file_filter_add_pattern(filter, "*.tif");

        saveDialog = gtk_file_chooser_dialog_new(
                        "Save Image",
                        GTK_WINDOW(ui_get_widget("mainWindow")),
                        GTK_FILE_CHOOSER_ACTION_SAVE,
                        GTK_STOCK_CANCEL, GTK_RESPONSE_CANCEL,
                        GTK_STOCK_SAVE, GTK_RESPONSE_ACCEPT,
                        NULL);
        gtk_file_chooser_set_filter(
                        GTK_FILE_CHOOSER(saveDialog), filter);
        gtk_file_chooser_set_do_overwrite_confirmation(
                        GTK_FILE_CHOOSER(saveDialog), TRUE);

        if (gtk_dialog_run(GTK_DIALOG(saveDialog)) ==
                        GTK_RESPONSE_ACCEPT) {
                filename =
gtk_file_chooser_get_filename(GTK_FILE_CHOOSER(saveDialog));

                tmp = g_string_new((const gchar *)filename);

                ext = &filename[strlen(filename) - 4];
                if (strcmp(ext, ".jpg") && strcmp(ext, ".tif"))
                        g_string_append(tmp, ".jpg");
                cv_stuff_save_image(tmp->str);
                g_string_free(tmp, FALSE);
        }

        gtk_widget_hide(saveDialog);
}
```

## ui-builder.h

```
/*
        This file is part of cv-a1.

        Copyright (C) 2008      Frederic-Gerald Morcos <fred.morcos@gmail.com>

        cv-a1 is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        cv-a1 is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU General Public License for more details.

        You should have received a copy of the GNU General Public License
        along with cv-a1.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef __UI_BUILDER_H__
#define __UI_BUILDER_H__

#include <gtk/gtk.h>

void                    ui_init();
void                    ui_destroy();
GtkWidget               *ui_get_widget(gchar *);
```

```
GtkAction               *ui_get_action(gchar *);

#endif  /* __UI_BUILDER_H__ */
```

## ui-builder.c

```
/*
        This file is part of cv-a1.

        Copyright (C) 2008      Frederic-Gerald Morcos <fred.morcos@gmail.com>

        cv-a1 is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        cv-a1 is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU General Public License for more details.

        You should have received a copy of the GNU General Public License
        along with cv-a1.  If not, see <http://www.gnu.org/licenses/>.
*/

#include "ui-builder.h"
#include <gtk/gtk.h>

static GtkBuilder       *builder;

void ui_init() {
        builder = gtk_builder_new();
        gtk_builder_add_from_file(builder, "ui.xml", NULL);
}

void ui_destroy() {
        g_object_unref(G_OBJECT(builder));
}

GtkWidget *ui_get_widget(gchar *name) {
        return GTK_WIDGET(gtk_builder_get_object(builder, name));
}

GtkAction *ui_get_action(gchar *name) {
        return GTK_ACTION(gtk_builder_get_object(builder, name));
}
```

## cv-stuff.h

```
/*
        This file is part of cv-a1.

        Copyright (C) 2008      Frederic-Gerald Morcos <fred.morcos@gmail.com>
        cv-a1 is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        cv-a1 is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU General Public License for more details.

        You should have received a copy of the GNU General Public License
        along with cv-a1.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef __CV_STUFF_H__
#define __CV_STUFF_H__

#include <cv.h>
#include <glib.h>

void cv_stuff_window(char *);
void cv_stuff_destroy();
void cv_stuff_reload();
void cv_stuff_check();

void cv_stuff_save_image(char *);

void cv_stuff_negate1();
void cv_stuff_negate2();
void cv_stuff_negate3();
void cv_stuff_rotate(int, int, double);
void cv_stuff_rotate_opencv(int, int, double);
void cv_stuff_upsample(gboolean, double);
void cv_stuff_upsample_opencv(gboolean, double);
void cv_stuff_scale(int, int);

gboolean isFloat (float);
CvScalar interpolate (IplImage *, float, float);

#endif  /* __CV_STUFF_H__ */
```

## cv-stuff.c

```
/*
        This file is part of cv-a1.

        Copyright (C) 2008      Frederic-Gerald Morcos <fred.morcos@gmail.com>

        cv-a1 is free software: you can redistribute it and/or modify
        it under the terms of the GNU General Public License as published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        cv-a1 is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU General Public License for more details.

        You should have received a copy of the GNU General Public License
        along with cv-a1.  If not, see <http://www.gnu.org/licenses/>.
*/

#include "cv-stuff.h"
#include <cv.h>
#include <highgui.h>
#include <glib.h>
#include <math.h>

#define TYPE CV_32FC1

static IplImage *image;         /* currently set image */

/**
 * Creates a window and loads an image from filename
 * into it.
 */
void cv_stuff_window(char *filename) {
        cv_stuff_destroy();
        image = cvLoadImage(filename, -1);
        cvNamedWindow("Image", CV_WINDOW_AUTOSIZE);
        cvShowImage("Image", image);
}

/**
 * Destroys and releases the image memory if there
 * is any.
 */
void cv_stuff_destroy() {
        if (image) cvReleaseImage(&image);
}

/**
 * Reloads the image data into the window.
 */
void cv_stuff_reload() {
        if (!image) return;

        cvShowImage("Image", image);
}

/**
 * Saves image data to a filename.
 */
void cv_stuff_save_image(char *filename) {
        if (image) cvSaveImage(filename, image);
}

/**
 * Checks if the window is still open, if not,
 * then free the image.
 */
```

```c
void cv_stuff_check() {
        if (!image) return;

        void *handle;

        handle = cvGetWindowHandle("Image");

        if (!handle)
                cv_stuff_destroy();
}

/**
 * Inverts an image (gets its negative) by
 * going over all pixels in the image data
 * and doing a val = 255 - val.
 */
void cv_stuff_negate1() {
        cv_stuff_check();
        if (!image) return;

        int             w,              /* width */
                        h,              /* height */
                        s,              /* step */
                        c,              /* channels */
                        i,              /* i counter */
                        j,              /* j counter */
                        k,              /* k counter */
                        p;              /* pos in array */
        uchar   *d;             /* image data */

        w = image->width;
        h = image->height;
        s = image->widthStep;
        c = image->nChannels;
        d = (uchar *)image->imageData;

        /**
         * Invert every pixel in the image. Deals
         * with the image as a 2D matrix with
         * direct access.
         */
        for (i = 0; i < h; i++)
                for (j = 0; j < w; j++)
                        for (k = 0; k < c; k++) {
                                p = i * s + j * c + k;
                                d[p] = 255 - d[p];
                        }

        cv_stuff_reload();
}



/**
 * Inverts an image (gets its negative) by
 * going over all pixels in the image data
 * and doing a val = 255 - val.
 */
void cv_stuff_negate2() {
        cv_stuff_check();
        if (!image) return;

        int             w,              /* width */
                        h,              /* height */
                        s,              /* step */
                        c,              /* channels */
                        i,              /* i counter */
                        j,              /* j counter */
                        k,              /* k counter */
                        p;              /* pos in array */
        uchar   *d;             /* image data */

        w = image->width;
        h = image->height;
        s = image->widthStep;
        c = image->nChannels;
        d = (uchar *)image->imageData;

        /**
         * Another way to get the inverse of an image,
         * though slower. Deals with the image as a
         * 2D matrix with indirect access.
         */
        CvScalar pixel;
        for (i = 0; i < w; i++)
                for (j = 0; j < h; j++) {
                        pixel = cvGet2D(image, j, i);
                        for (k = 0; k < c; k++)
                                pixel.val[k] = 255 - pixel.val[k];
                        cvSet2D(image, j, i, pixel);
                }

        cv_stuff_reload();
}

/**
 * Inverts an image (gets its negative) by
 * going over all pixels in the image data
 * and doing a val = 255 - val.
 */
void cv_stuff_negate3() {
        cv_stuff_check();
        if (!image) return;

        int             w,              /* width */
                        h,              /* height */
                        c,              /* channels */
                        i;              /* counter */
        uchar   *d;             /* image data */

        w = image->width;
        h = image->height;
        c = image->nChannels;
        d = (uchar *)image->imageData;
        /**
         * Invert every pixel in the image. Deals
         * with the image as a 1D matrix.
         */
        for (i = 0; i < h * w * c; i++)
                d[i] = 255 - d[i];

        cv_stuff_reload();
}

/**
 * Receives the position of a pixel in float and interpolates
 * its intensity value (for each channel) using the
 * neighboring pixels.
 */
CvScalar interpolate (IplImage *image, float x, float y) {
        int xf = floorf(x),             /* left */
                xc = ceilf(x),          /* right */
                yf = floorf(y),         /* top */
                yc = ceilf(y);          /* bottom */

        double l = x - xf,              /* left */
                   r = xc - x,          /* right */
                   u = y - yf,          /* up */
                   d = yc - y;          /* down */

        int     k;                                      /* channels counter */

        double v_up,                    /* vertical interpolation, up */
                   v_down;                      /* vertical interpolation, down
*/

        CvScalar res;

        for (k = 0; k < image->nChannels; k++) {
                v_up = (l * cvGet2D(image, xc, yf).val[k]) + (r * cvGet2D(image,
xf, yf).val[k]);
                v_down = (l * cvGet2D(image, xc, yc).val[k]) + (r *
cvGet2D(image, xf, yc).val[k]);
                res.val[k] = (u * v_down) + (d * v_up);
        }

        return res;
}

/**
 * Rotates an image around point (x, y) by angle. Uses
 * OpenCV built-in functions to do so. Unused.
 */
void cv_stuff_rotate_opencv(int x, int y, double angle) {
        cv_stuff_check();
        if (!image) return;

        IplImage                *tmp;
        CvPoint2D32f    rot_point;
        CvMat                   *rot_matrix;
        rot_point = cvPoint2D32f(x, y);
        rot_matrix = cvCreateMat(2, 3, CV_32FC1);
        cv2DRotationMatrix(rot_point, angle, 1.0, rot_matrix);

        tmp = cvCloneImage(image);
        cvWarpAffine(tmp, image, rot_matrix,
                        CV_INTER_LINEAR + CV_WARP_FILL_OUTLIERS, cvScalarAll(0));

        cvReleaseImage(&tmp);
        cv_stuff_reload();
}

/**
 * Rotates an image around point (x, y) by angle
 * manually. First creates a translation matrix to
 * move the data by (-x, -y). Then creates a rotation
 * matrix to rotate by angle then translates back by
 * (x, y). Then, it multiplies the three matrices
 * together and gets the inverse to get the intensity
 * of the original pixel into the destination one.
 * Interpolation is used in the last process to smooth
 * the resulting image.
 */
void cv_stuff_rotate(int y, int x, double angle) {
        cv_stuff_check();
        if (!image) return;

        int i, j;
        float a = angle * M_PI / 180;
        float src_p[3] = {0, 0, 1};
        float dst_p[3] = {0, 0, 1};
        float inv_m[3][3];
        float rot_m[3][3] = {
                {cos(a),                -sin(a),        ((1 - cos(a)) * x) +
(sin(a) * y)},
                {sin(a),                cos(a),         (sin(a) * -x) + ((1 -
cos(a)) * y)},
                {0,                             0,                      1}};

        CvMat rot_mat = cvMat(3, 3, TYPE, rot_m);
        CvMat inv_mat = cvMat(3, 3, TYPE, inv_m);
        CvMat src_pnt = cvMat(3, 1, TYPE, src_p);
        CvMat dst_pnt = cvMat(3, 1, TYPE, dst_p);
        IplImage *tmp = cvCloneImage(image);
        CvScalar pixel;

        cvInvert(&rot_mat, &inv_mat, CV_LU);

        for (i = 0; i < tmp->height; i++)
                for (j = 0; j < tmp->width; j++) {
                        dst_p[0] = i;
                        dst_p[1] = j;

                        cvMatMul(&inv_mat, &dst_pnt, &src_pnt);

                        src_p[0] = cvGetReal2D(&src_pnt, 0, 0);
                        src_p[1] = cvGetReal2D(&src_pnt, 1, 0);

                        if (src_p[0] >= 0 && src_p[0] < tmp->height - 1 &&
                                        src_p[1] >= 0 && src_p[1] < tmp->width -
1) {

                                if (isFloat(src_p[0]) || isFloat(src_p[1]))
                                        pixel = interpolate(tmp, src_p[0],
src_p[1]);
                                else
                                        pixel = cvGet2D(tmp, src_p[0], src_p[1]);
                        }
                        else
                                pixel = CV_RGB(0, 0, 0);
                        cvSet2D(image, i, j, pixel);
                }

        cvReleaseImage(&tmp);
        cv_stuff_reload();
}

/**
 * Returns TRUE if f is a floating point number and FALSE otherwise.
 */
gboolean isFloat (float f) {
        if (f == floorf(f)) return FALSE;
        return TRUE;
}

/**
 * Upsamples the image by replicating pixels * factor.
 */
void cv_stuff_upsample(gboolean replication, double factor) {
        cv_stuff_check();
        if (!image || factor == 0.0) return;

        CvSize          orig_size;
        IplImage        *tmp;
        int                     i,
                                j,
                                x = -1,         /* cached floori */
                                y = -1,         /* cached floorj */
                                floori,         /* new i */
                                floorj;         /* new j */
        CvScalar        pixel;          /* current/cached pixel */

        orig_size = cvGetSize(image);
        tmp = cvCloneImage(image);
        image = cvCreateImage(
                        cvSize(orig_size.width * factor,
                                orig_size.height * factor),
                        tmp->depth, tmp->nChannels);

        if (replication) {
                /**
                 * Loop over the destination image and gets the
                 * corresponding intensity position from the
                 * source image then applies it to the pixels
                 * in the destination image.
                 */
                for(i = 0; i < image->width; i++) {
                        for(j = 0; j < image->height; j++) {
                                floori = floor(i / factor);
                                floorj = floor(j / factor);

                                /**
                                 * If we're going to get the same pixel
                                 * again as the last iteration, don't do
                                 * so, use the already cached pixel.
                                 */
                                if (floori != x || floorj != y) {
                                        pixel = cvGet2D(tmp, floorj, floori);
                                        x = floori;
                                        y = floorj;
                                }

                                cvSet2D(image, j, i, pixel);
                        }
                }
        }

        cvReleaseImage(&tmp);
        cv_stuff_reload();
}

/**
 * Upsamples the image using built-in functions in OpenCV, either
 * by replication or interpolation depending on the boolean value
 * given to replication.
 */
void cv_stuff_upsample_opencv(gboolean replication, double factor) {
        cv_stuff_check();
        if (!image || factor == 0.0) return;

        CvSize          orig_size;
        IplImage        *tmp;

        orig_size = cvGetSize(image);
        tmp = cvCloneImage(image);
        image = cvCreateImage(
                        cvSize(orig_size.width * factor,
                                orig_size.height * factor),
                        tmp->depth, tmp->nChannels);

        if (replication)
                cvResize(tmp, image, CV_INTER_NN);
        else
                cvResize(tmp, image, CV_INTER_LINEAR);

        cvReleaseImage(&tmp);
        cv_stuff_reload();
}

/**
```

```
 * Scales the image to size (width, height) using interpolation
 * by using built-in OpenCV functions.
 */
void cv_stuff_scale(int width, int height) {
        cv_stuff_check();
        if (!image) return;

        IplImage        *tmp;

        tmp = cvCloneImage(image);
        image = cvCreateImage(
                        cvSize(width, height),
                        tmp->depth, tmp->nChannels);
        cvResize(tmp, image, CV_INTER_CUBIC);

        cvReleaseImage(&tmp);
        cv_stuff_reload();
}
```