ToolIP
Tool for Image Processing

Version 2022

User Manual

Fraunhofer ITWM

www.itwm.fraunhofer.de/toolip
toolip@itwm.fraunhofer.de
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.17 Timing On/Off</td>
<td>29</td>
</tr>
<tr>
<td>4.1.18 Safe Mode</td>
<td>29</td>
</tr>
<tr>
<td>4.1.19 Reload Library</td>
<td>30</td>
</tr>
<tr>
<td>4.1.20 RAGBI - Run A Graph on a Batch of Images or Items</td>
<td>30</td>
</tr>
<tr>
<td>4.2 RAGBI</td>
<td>30</td>
</tr>
<tr>
<td>4.2.1 Settings</td>
<td>32</td>
</tr>
<tr>
<td>4.2.2 About</td>
<td>32</td>
</tr>
<tr>
<td>4.3 Plugin Menu</td>
<td>34</td>
</tr>
<tr>
<td>4.4 Search for Plugins</td>
<td>34</td>
</tr>
<tr>
<td>4.5 Workspace</td>
<td>35</td>
</tr>
<tr>
<td>4.5.1 Graph Parameters</td>
<td>35</td>
</tr>
<tr>
<td>4.6 Log Window</td>
<td>35</td>
</tr>
<tr>
<td>5 Nodes and Specifications</td>
<td>37</td>
</tr>
<tr>
<td>5.1 Plugins</td>
<td>37</td>
</tr>
<tr>
<td>5.1.1 Status field</td>
<td>37</td>
</tr>
<tr>
<td>5.1.2 Parameter Button</td>
<td>38</td>
</tr>
<tr>
<td>5.1.3 In- and Output Pins</td>
<td>41</td>
</tr>
<tr>
<td>5.1.4 Parameter In- and Output</td>
<td>41</td>
</tr>
<tr>
<td>5.1.5 Node Label</td>
<td>42</td>
</tr>
<tr>
<td>5.1.6 Error Handling</td>
<td>43</td>
</tr>
<tr>
<td>5.2 Linking Ports</td>
<td>43</td>
</tr>
<tr>
<td>5.3 Data Flow Control</td>
<td>44</td>
</tr>
<tr>
<td>5.3.1 Repeat Node</td>
<td>44</td>
</tr>
<tr>
<td>5.3.2 Branch Node</td>
<td>44</td>
</tr>
<tr>
<td>5.3.3 Merge Node</td>
<td>46</td>
</tr>
<tr>
<td>5.3.4 Switch Node</td>
<td>47</td>
</tr>
<tr>
<td>6 Further Examples</td>
<td>49</td>
</tr>
<tr>
<td>6.1 Interactive choice of parameters</td>
<td>49</td>
</tr>
<tr>
<td>6.2 Characterizing objects and textures</td>
<td>49</td>
</tr>
<tr>
<td>7 Generation of Runtime Licenses</td>
<td>53</td>
</tr>
<tr>
<td>8 Algorithm Documentation</td>
<td>55</td>
</tr>
<tr>
<td>8.1 Introduction</td>
<td>55</td>
</tr>
<tr>
<td>8.2 Toolbox Introductions</td>
<td>55</td>
</tr>
<tr>
<td>8.2.1 Arithmetic</td>
<td>55</td>
</tr>
<tr>
<td>8.2.2 Classification</td>
<td>55</td>
</tr>
<tr>
<td>8.2.3 Color</td>
<td>56</td>
</tr>
<tr>
<td>8.2.4 Data</td>
<td>56</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>8.2.5</td>
<td>Display</td>
</tr>
<tr>
<td>8.2.6</td>
<td>Features</td>
</tr>
<tr>
<td>8.2.7</td>
<td>Filter</td>
</tr>
<tr>
<td>8.2.8</td>
<td>LabelImage</td>
</tr>
<tr>
<td>8.2.9</td>
<td>Manipulation</td>
</tr>
<tr>
<td>8.2.10</td>
<td>Matching</td>
</tr>
<tr>
<td>8.2.11</td>
<td>Matrix</td>
</tr>
<tr>
<td>8.2.12</td>
<td>Morphology</td>
</tr>
<tr>
<td>8.2.13</td>
<td>Segmentation</td>
</tr>
<tr>
<td>8.2.14</td>
<td>Transformation</td>
</tr>
<tr>
<td>8.2.15</td>
<td>Utility</td>
</tr>
<tr>
<td>8.3</td>
<td>Arithmetic Plugins</td>
</tr>
<tr>
<td>8.3.1</td>
<td>Plugin Abs</td>
</tr>
<tr>
<td>8.3.2</td>
<td>Plugin AbsDiff</td>
</tr>
<tr>
<td>8.3.3</td>
<td>Plugin Add</td>
</tr>
<tr>
<td>8.3.4</td>
<td>Plugin And</td>
</tr>
<tr>
<td>8.3.5</td>
<td>Plugin AssertEq</td>
</tr>
<tr>
<td>8.3.6</td>
<td>Plugin Ceil</td>
</tr>
<tr>
<td>8.3.7</td>
<td>Plugin Cos</td>
</tr>
<tr>
<td>8.3.8</td>
<td>Plugin Divide</td>
</tr>
<tr>
<td>8.3.9</td>
<td>Plugin Equal</td>
</tr>
<tr>
<td>8.3.10</td>
<td>Plugin Exp</td>
</tr>
<tr>
<td>8.3.11</td>
<td>Plugin Floor</td>
</tr>
<tr>
<td>8.3.12</td>
<td>Plugin Log</td>
</tr>
<tr>
<td>8.3.13</td>
<td>Plugin Math</td>
</tr>
<tr>
<td>8.3.14</td>
<td>Plugin MathConstant</td>
</tr>
<tr>
<td>8.3.15</td>
<td>Plugin Maximum</td>
</tr>
<tr>
<td>8.3.16</td>
<td>Plugin Minimum</td>
</tr>
<tr>
<td>8.3.17</td>
<td>Plugin Modulus</td>
</tr>
<tr>
<td>8.3.18</td>
<td>Plugin Multiply</td>
</tr>
<tr>
<td>8.3.19</td>
<td>Plugin Negative</td>
</tr>
<tr>
<td>8.3.20</td>
<td>Plugin Or</td>
</tr>
<tr>
<td>8.3.21</td>
<td>Plugin Pow</td>
</tr>
<tr>
<td>8.3.22</td>
<td>Plugin RPNC</td>
</tr>
<tr>
<td>8.3.23</td>
<td>Plugin Sin</td>
</tr>
<tr>
<td>8.3.24</td>
<td>Plugin SquareRoot</td>
</tr>
<tr>
<td>8.3.25</td>
<td>Plugin Subtract</td>
</tr>
<tr>
<td>8.3.26</td>
<td>Plugin Tan</td>
</tr>
<tr>
<td>8.4</td>
<td>Barcode Plugins</td>
</tr>
<tr>
<td>8.4.1</td>
<td>Plugin BarcodeReader</td>
</tr>
</tbody>
</table>
8.5 Color Plugins .......................................................... 61
  8.5.1 Plugin ColorMap .................................................. 61
  8.5.2 Plugin ColorTransform .......................................... 61
  8.5.3 Plugin Combine .................................................. 61
  8.5.4 Plugin Separate .................................................. 62
8.6 Calculator Plugins ................................................. 62
  8.6.1 Plugin ValueCalc ............................................... 62
8.7 Data Plugins .......................................................... 62
  8.7.1 Plugin Ball ....................................................... 62
  8.7.2 Plugin Box ....................................................... 62
  8.7.3 Plugin Circle .................................................... 62
  8.7.4 Plugin Constant ................................................ 62
  8.7.5 Plugin HalfPlane ............................................... 62
  8.7.6 Plugin Line ...................................................... 62
  8.7.7 Plugin Noise ..................................................... 62
  8.7.8 Plugin PixelValue .............................................. 62
  8.7.9 Plugin Point ..................................................... 62
  8.7.10 Plugin PositionToValue ...................................... 63
  8.7.11 Plugin Spline .................................................. 63
  8.7.12 Plugin Value ................................................... 63
8.8 Features Plugins .................................................... 63
  8.8.1 Plugin Area ....................................................... 63
  8.8.2 Plugin AveragePhi .............................................. 63
  8.8.3 Plugin AveragePolar ............................................ 63
  8.8.4 Plugin BinNbhHist .............................................. 63
  8.8.5 Plugin BinNbhHistPlusXYEntropy ............................... 63
  8.8.6 Plugin Border ................................................... 63
  8.8.7 Plugin Covbin ................................................... 63
  8.8.8 Plugin Entropy .................................................. 63
  8.8.9 Plugin FromGraph .............................................. 64
  8.8.10 Plugin Geometry ............................................... 64
  8.8.11 Plugin Haralick ................................................ 64
  8.8.12 Plugin HoughEntropy ......................................... 64
  8.8.13 Plugin LBPHist ................................................ 64
  8.8.14 Plugin Median .................................................. 64
  8.8.15 Plugin NbhSpHist .............................................. 64
  8.8.16 Plugin RelnbhHist ............................................. 64
  8.8.17 Plugin RotHistVar ............................................. 64
  8.8.18 Plugin Rough ................................................... 64
  8.8.19 Plugin SDistMedianqDiff .................................... 64
8.8.20 Plugin SdistAveVar ............................................................. 64
8.8.21 Plugin Shadow ................................................................. 65
8.8.22 Plugin ShiftHistVar ...................................................... 65
8.8.23 Plugin Statistics ............................................................. 65
8.8.24 Plugin ThreePartsRelNbhHist ...................................... 65
8.8.25 Plugin TotalVariation ................................................... 65
8.8.26 Plugin VSlope ............................................................... 65
8.8.27 Plugin XYEntropy ......................................................... 65
8.8.28 Plugin XYInteractionHist ........................................... 65

8.9 Fast Fourier Transform Plugins ........................................ 65
8.9.1 Plugin DCT ................................................................. 65
8.9.2 Plugin FFTW ................................................................. 65

8.10 Filter Plugins ............................................................... 65
8.10.1 Plugin Average1d ......................................................... 65
8.10.2 Plugin Average3d ........................................................ 66
8.10.3 Plugin Average3dMasked .......................................... 66
8.10.4 Plugin BinNbh ............................................................... 66
8.10.5 Plugin BoundaryRegion ................................................ 66
8.10.6 Plugin CEShock ............................................................ 66
8.10.7 Plugin Canny ............................................................... 66
8.10.8 Plugin Convolve .......................................................... 66
8.10.9 Plugin Covbin .............................................................. 66
8.10.10 Plugin Covdet ............................................................. 66
8.10.11 Plugin Diffusion ......................................................... 66
8.10.12 Plugin Distance .......................................................... 66
8.10.13 Plugin FourConnected .............................................. 66
8.10.14 Plugin Gauss .............................................................. 66
8.10.15 Plugin GaussDerivative ............................................. 67
8.10.16 Plugin IsoNonlinDiffusion ....................................... 67
8.10.17 Plugin LBP ................................................................. 67
8.10.18 Plugin Laplace ........................................................... 67
8.10.19 Plugin LocalBinaryPatterns ..................................... 67
8.10.20 Plugin LowCount ....................................................... 67
8.10.21 Plugin Median ............................................................ 67
8.10.22 Plugin Ranking ........................................................... 67
8.10.23 Plugin RidgeDetector ............................................... 67
8.10.24 Plugin RobertsCross ................................................. 67
8.10.25 Plugin Sobel ............................................................... 67
8.10.26 Plugin StructureTensor2D ........................................ 68
8.10.27 Plugin StructureTensorEVD2D .................................. 68
### 8.10.28 Plugin StructureTensorEVD3D

---

### 8.10.29 Plugin Variance

---

### 8.10.30 Plugin ceShock_IsoNonlinDiff

---

### 8.11 Handling Plugins

---

#### 8.11.1 Plugin GraphFromFile

---

#### 8.11.2 Plugin GraphOnLabel

---

#### 8.11.3 Plugin GraphOnSlice

---

#### 8.11.4 Plugin GraphOnZone

---

#### 8.11.5 Plugin Loop

---

#### 8.11.6 Plugin LoopAndTile

---

#### 8.11.7 Plugin ParameterLoop

---

#### 8.11.8 Plugin Paste

---

#### 8.11.9 Plugin Tile

---

#### 8.11.10 Plugin TileOneInTwoOut

---

### 8.12 Manipulation Plugins

---

#### 8.12.1 Plugin Append

---

#### 8.12.2 Plugin Autocrop

---

#### 8.12.3 Plugin CartesianToPolar

---

#### 8.12.4 Plugin ComposeImage

---

#### 8.12.5 Plugin DeleteDuplicate

---

#### 8.12.6 Plugin Deserialize

---

#### 8.12.7 Plugin Expand

---

#### 8.12.8 Plugin Extract

---

#### 8.12.9 Plugin Flip

---

#### 8.12.10 Plugin LabelToColor

---

#### 8.12.11 Plugin LabelToFeature

---

#### 8.12.12 Plugin LabelToSize

---

#### 8.12.13 Plugin LabelToValue

---

#### 8.12.14 Plugin Pad

---

#### 8.12.15 Plugin PlaneToVolume

---

#### 8.12.16 Plugin PolarToCartesian

---

#### 8.12.17 Plugin Reorient

---

#### 8.12.18 Plugin Replicate

---

#### 8.12.19 Plugin Resample

---

#### 8.12.20 Plugin Resize

---

#### 8.12.21 Plugin Rotate

---

#### 8.12.22 Plugin Scale

---

#### 8.12.23 Plugin SelectRays

---

#### 8.12.24 Plugin Serialize

---

#### 8.12.25 Plugin Shrink

---
8.12.26 Plugin Split ............................................. 71
8.12.27 Plugin SwitchAxes ................................. 71
8.12.28 Plugin Translate ................................. 71
8.12.29 Plugin VolumeToPlane ............................ 71

8.13 Matching Plugins ................................. 71
8.13.1 Plugin FindShape .................................... 71
8.13.2 Plugin Fsdh ........................................ 72
8.13.3 Plugin Match ....................................... 72
8.13.4 Plugin TranslateAdd ............................... 72

8.14 Matrix Plugins ................................. 72
8.14.1 Plugin BinNbhThFlow .............................. 72
8.14.2 Plugin CovarianceMatrix ....................... 72
8.14.3 Plugin Determinant .............................. 72
8.14.4 Plugin EVD ......................................... 72
8.14.5 Plugin FeatureImage .............................. 72
8.14.6 Plugin GaussianMixture ......................... 72
8.14.7 Plugin ICA ......................................... 72
8.14.8 Plugin Invert ....................................... 72
8.14.9 Plugin KMeans .................................... 73
8.14.10 Plugin Multiply .................................. 73
8.14.11 Plugin PCA ........................................ 73
8.14.12 Plugin QR .......................................... 73
8.14.13 Plugin RANSAC ................................. 73
8.14.14 Plugin RANSACWithModellOut .................. 73
8.14.15 Plugin ReplicateColumn ....................... 73
8.14.16 Plugin ReplicateRow ............................ 73
8.14.17 Plugin SVD ....................................... 73
8.14.18 Plugin ShuffleRows ............................. 73
8.14.19 Plugin Transpose ............................... 73

8.15 Morphology Plugins ......................... 73
8.15.1 Plugin BeucherGradient ......................... 74
8.15.2 Plugin BlackTopHat .............................. 74
8.15.3 Plugin Closing .................................... 74
8.15.4 Plugin CutHill .................................... 74
8.15.5 Plugin Dilation .................................... 74
8.15.6 Plugin Erosion .................................... 74
8.15.7 Plugin ExternalGradient ....................... 74
8.15.8 Plugin FillHole .................................... 74
8.15.9 Plugin InternalGradient ....................... 74
8.15.10 Plugin MidrangeFilter ......................... 74
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.17.24 Plugin RayMaximum</td>
<td>78</td>
</tr>
<tr>
<td>8.17.25 Plugin RayMinimum</td>
<td>78</td>
</tr>
<tr>
<td>8.17.26 Plugin ReadASCII</td>
<td>78</td>
</tr>
<tr>
<td>8.17.27 Plugin ReadImage</td>
<td>78</td>
</tr>
<tr>
<td>8.17.28 Plugin ReadMap</td>
<td>78</td>
</tr>
<tr>
<td>8.17.29 Plugin ReplaceNonFinite</td>
<td>78</td>
</tr>
<tr>
<td>8.17.30 Plugin ReplaceValue</td>
<td>78</td>
</tr>
<tr>
<td>8.17.31 Plugin RingBuffer</td>
<td>78</td>
</tr>
<tr>
<td>8.17.32 Plugin SampleLine</td>
<td>78</td>
</tr>
<tr>
<td>8.17.33 Plugin SamplePoints</td>
<td>78</td>
</tr>
<tr>
<td>8.17.34 Plugin SaveASCII</td>
<td>79</td>
</tr>
<tr>
<td>8.17.35 Plugin SaveImage</td>
<td>79</td>
</tr>
<tr>
<td>8.17.36 Plugin Script</td>
<td>79</td>
</tr>
<tr>
<td>8.17.37 Plugin SelectImage</td>
<td>79</td>
</tr>
<tr>
<td>8.17.38 Plugin SelectPixel</td>
<td>79</td>
</tr>
<tr>
<td>8.17.39 Plugin SetImageAttributes</td>
<td>79</td>
</tr>
<tr>
<td>8.17.40 Plugin Sleep</td>
<td>79</td>
</tr>
<tr>
<td>8.17.41 Plugin Sort</td>
<td>79</td>
</tr>
<tr>
<td>8.17.42 Plugin StaticImage</td>
<td>79</td>
</tr>
<tr>
<td>8.17.43 Plugin StaticStorage</td>
<td>79</td>
</tr>
<tr>
<td>8.17.44 Plugin Statistics</td>
<td>79</td>
</tr>
<tr>
<td>8.17.45 Plugin StringManipulation</td>
<td>79</td>
</tr>
<tr>
<td>8.17.46 Plugin StringToImage</td>
<td>80</td>
</tr>
<tr>
<td>8.17.47 Plugin TJHoughCircle</td>
<td>80</td>
</tr>
<tr>
<td>8.17.48 Plugin TextSerializer</td>
<td>80</td>
</tr>
<tr>
<td>8.17.49 Plugin TimerStart</td>
<td>80</td>
</tr>
<tr>
<td>8.17.50 Plugin TimerStop</td>
<td>80</td>
</tr>
<tr>
<td>8.17.51 Plugin ZoneList</td>
<td>80</td>
</tr>
<tr>
<td>8.18 Sysutils Plugins</td>
<td>80</td>
</tr>
<tr>
<td>8.18.1 Plugin Console</td>
<td>80</td>
</tr>
<tr>
<td>8.18.2 Plugin EnvVar</td>
<td>80</td>
</tr>
<tr>
<td>8.18.3 Plugin FileExists</td>
<td>80</td>
</tr>
<tr>
<td>8.18.4 Plugin Platform</td>
<td>80</td>
</tr>
<tr>
<td>8.18.5 Plugin ReadTextFile</td>
<td>80</td>
</tr>
<tr>
<td>8.18.6 Plugin SaveText</td>
<td>81</td>
</tr>
<tr>
<td>8.19 Wavelets Plugins</td>
<td>81</td>
</tr>
<tr>
<td>8.19.1 Plugin ModulusMaxima</td>
<td>81</td>
</tr>
<tr>
<td>8.19.2 Plugin MultiscaleWavelet</td>
<td>81</td>
</tr>
<tr>
<td>8.19.3 Plugin OrthoFWT</td>
<td>81</td>
</tr>
<tr>
<td>8.19.4 Plugin OrthoIWT</td>
<td>81</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This manual gives an introduction to the software ToolIP (Tool for Image Processing). ToolIP has been developed by the image processing department at Fraunhofer ITWM in Kaiserslautern, Germany. As graphical development user interface, it comes with an integrated image processing library developed at ITWM. This library contains a large variety of image processing and analysis algorithms that can be arranged in data flow graphs and thus combined to the solutions of complex inspection problems. Together with its library, ToolIP allows for solving complex image processing and analysis problems by visual programming. This approach has the advantage of being very intuitive, and so the user can start working productively after a very short training period. The descriptions of such data flow graphs can be saved and then immediately be used with a command line interpreter called MAOICmd. It is also possible to call a graph from inside a C or C++ application. This makes it easy to integrate a graph developed in ToolIP in an inspection system or a solution directly in the production line. ToolIP and its library are serving as the basis for many image analysis and quality assurance projects in the academic and industrial field.

Are you interested in trying ToolIP for free? An installer of the demo version can be found at the homepage www.itwm.fraunhofer.de/toolip.

The demo version underlies some restrictions, which are explained in section 2.3.2. If you are interested in the licensing conditions or if you want to purchase a run-time or developer version of the underlying algorithms, please do not hesitate to contact Fraunhofer ITWM via toolip@itwm.fraunhofer.de.

It is also possible to use the algorithms from the software MAVI (Modular Algorithms for Volume Images) for volume image processing and analysis within ToolIP. This toolbox is called MAVIkit and offers a comprehensive bandwidth of processing and analysis tools especially designed and useful for volume image data together with a geometrical characterization. Its major field of application is data from materials science.

For obtaining more information concerning or for using ToolIP with MAVIkit please visit www.itwm.fraunhofer.de/mavi.

This manual is organized as follows:
The installation of ToolIP is explained in chapter 2. A first overview over the GUI of ToolIP is given in chapter 3. This chapter describes the first steps for building an image analysis graph. The menus and functions of the software are explained in detail in chapter 4. Of special importance are the nodes of a graph since they model the basic algorithms. How these nodes are represented, how parameters are modified and which states the nodes can be in is shown in chapter 5. Some further examples in chapter 6 point out features of the tool useful in practice. In chapter 7 the generation of runtime licenses is described. The documentation of the algorithm nodes can be found in chapter 8. The manual is concluded with a short summary in chapter 9.

Appendix A provides a list of useful shortcuts to make every-day work more efficient. Last but not least, credits to the utilized third party libraries can be found in appendix B.
Chapter 2

Installation

In this chapter, we describe the installation procedure of the software ToolIP.

2.1 Windows

During the first installation of ToolIP administrator rights are required since some environment variables need to be set. If you start the installer suitable for your operating system you find the menu as shown in figure 2.1.

![Figure 2.1: ToolIP installer menu.](image)

In the drop-down menu, you can choose the option Basic Install, which will only install the ToolIP package. This might be sufficient if Microsoft Visual Studio 2010 is already installed on your computer. Nevertheless, if the version (including service pack number) of the compiler does not fit the binaries, there will be a problem during execution of the program. In case you are not sure, please tick the box Vc Redistributable. Furthermore, a plugin Software Development Kit (SDK) can be installed by ticking the corresponding box. The plugin SDK enables the user to write his own algorithms in C++ and integrate and use them in ToolIP.

After clicking Next a window appears. This window is shown in figure 2.2. You are asked to enter your license file name or the directory it is stored in. By default this path is set to:

```
toolip.lic
```

To choose the right location click onto the three dots next to the path label. After you did click on Next, you are asked to give a path for the installation directory. This is shown in figure 2.3. By default this path is set to:

```
C:\Program Files\ToolIP
```
2.2 Linux Installation

Figure 2.2: Choose the license directory.

This path is added to a new windows environment variable called ITWMDIR. After specifying the directory the actual installation procedure starts. As a first step during the installation procedure, the driver for license checks with a USB dongle is installed. The driver installer opens an extra window similar to figure 2.4. Please press install to continue the installation. If you have chosen to install the Microsoft Visual C Redistributable, the setup wizard window for this software, as shown in figure 2.5 is being opened.

Please follow the instructions, accept the license terms and start the installation process. As soon as this setup for the Microsoft Visual C Redistributable is completed click on finish. Now, the ToolIP installer should show the window in figure 2.6. By clicking the close button the installation process is completed and you can start ToolIP from the Windows start menu.

2.2 Linux

ToolIP can also be used under some Linux distributions. So far, there is no installer for those platforms available in the download section. If you are interested in using ToolIP with Linux, please contact us at toolip@itwm.fraunhofer.de for further information.
2.3 Starting ToolIP

After a successful installation you are now able to start ToolIP. Depending on whether you use the full or the demo version, the functionality of your software will be different.

2.3.1 Full Version

The full version of ToolIP is delivered with a USB dongle and a license file. The license file must be in the following location for ToolIP to find it:

```
%ITWMDIR%\share\config
```

After starting the software with dongle in USB slot the application main window will appear on the desktop. This will look like the screenshot in figure 2.7. You are now able to start using ToolIP.

If the dongle is not found, that is ToolIP starts in demo mode even with dongle in USB slot, please re-install the dongle driver. You find the installer in

```
%ITWMDIR%\support\matrix\inf_inst.exe
```
2.3 Starting ToolIP

2.3.2 Demo Version

If ToolIP is started without dongle or license file, the software will work in demo mode. This is indicated with the message shown in figure 2.8.

Note: The demo mode comes with the following restrictions:

- All input images larger than 256x256x8 pixels are cropped to those respective size in each coordinate direction. This is also the case for ToolIP’s command line interpreter MAOImd.
- The save option for images is not available.
- The safe mode does not work in the demo version. It is mainly useful for users interested in writing their own plugins for ToolIP, since in this mode, each plugin is started in a separate process. If a plugin crashes, the application will continue working. This makes debugging of new plugins much easier.

If you want to purchase a full version, please contact the ToolIP mailing address
Figure 2.8: **ToolIP** will start in demo mode.

toolip@itwm.fraunhofer.de
for further information.
Chapter 3

Overview

After having finished the installation, we give a short overview over ToolIP with very simple image processing examples. A more detailed description of all the functions provided by the program can be found in the following chapters.

All the graphs shown here come with the installation package and can be found in the installation directory, which will be referred to as INST_DIR. Please replace INST_DIR with your installation directory. To relocate your installation directory, please remember that the environment variable ITWMDIR points to it. In the installation directory, the graphs are contained within the subfolder

INST_DIR/examples/graphs.

You can open the graphs and follow the descriptions given here step by step.

Figure 3.1: ToolIP with a minimalistic example graph.
3.1 First Example – Image Simplification

Let us start with a first example: Figure 3.1 shows ToolIP with a minimalistic example graph performing nonlinear diffusion filtering on a gray value image of a bridge. The graph for this example can be found here:

INST_DIR/examples/graphs/Diffusion.tlp

Please replace INST_DIR with your installation directory. During installation, the environment variable %ITWMDIR% has been set to name of the installation directory, too.

You can start the graph in two ways: By clicking on the Start Graph-button in the menu bar, all the nodes of the graph are executed, and you can see the results immediately. The nodes can be executed step by step by clicking on the green status field of each node.

In the following, we describe this example in detail: Each of the five boxes in the ToolIP workspace – the nodes of the graph – represents an image processing algorithm while the edges between these nodes visualize the data flow.

The graph is shown in figure 3.2. The leftmost node (ReadImage) is the source of the data and reads an image from a file. In our example, the image is an 8-bit integer gray value image of size 512x512 pixels. If you are running ToolIP in demo mode (see 2.3.2), only the upper left corner with size 256x256 is read. The second node (ConvertType) converts this image to a floating point data type. This is necessary since the third node (IsoNonlinDiffusion) works with floating point precision. This node implements the functionality of an adaptive simplification and denoising of the image data. The remaining two nodes (Display) provide the displays of the converted image in floating point precision and the resulting simplified image.

After showing how to execute a graph, we are going to take a look at how to build a new graph in the next section.

3.2 Building Your Own Graph

After starting ToolIP, the graph editor starts with an empty workspace (as already seen in figure 2.7). You can select nodes from the plugin menu on the left-hand side and drag-and-drop them into the workspace with the mouse pointer. Since the number of nodes can make it complicated to find the desired one fast, the search widget can help to make it easier.

Let us now start with building a very simple example graph with only four nodes. First, we need an input for reading data from file. We propose to use the node ReadImage which can be found in the menu under In-/Output → File → ReadImage. The node can also be found with the help of the search bar as shown in figure 3.3. Clicking on the plugin name either in the search results or in the plugin menu, the plugin can be drag-and-dropped into the workspace. By clicking on the blue box inside the plugin, the parameter dialog opens. You can now enter the name of the image file to open. Any gray value image can be used for this simple graph, for example.
3.2 Building Your Own Graph

Figure 3.3: Building your own graph. Left: Find the plugin ReadImage. Right: Entering parameters to the ReadImage plugin.

%ITWMDIR%/examples/graphs/bridge.pgm

Note that the variable name %ITWMDIR% is resolved by the application. After typing the parameters and clicking the OK-Button, one can run the plugin by clicking on the green box inside the plugin icon. Data available at the output pin of the node is indicated by cyan color. Now you can drag-and-drop the second node for our graph. For this example, we have chosen Image → Morphology → Opening to perform a morphological opening with a rectangular structuring element. To create an edge between the two nodes, first click on the output pin of the ReadImage node which then turns to yellow. This can be seen in figure 3.4 on the left. Clicking on the input pin of the Opening node creates an edge, i.e. a data connection between these two nodes. We can now add two Display plugins to finish the simple graph. The resulting graph is shown in figure 3.4 in the middle. The graph for this example can also be found in the file

INST_DIR/examples/graphs/opening.tlp.

If you want to execute the graph node by node, it is helpful to see the progress graphically. To this end, green edges indicate that there is data available on this edge, that means the node before this edge has already produced its output. Black edges indicate that no data is available, and the node after this edge will not be able to run. You can see this in figure 3.4 in the middle and on the right.

The effect of the morphological opening on an example image is displayed in figure 3.5. The rectangular structuring element is clearly visible. Analogously to setting the file name in the ReadImage plugin, one can use the parameter dialog of the Opening plugin to use a larger structuring element and make the effect of the opening more apparent.

Now you have built your first graph in ToolIP. You can save the result by clicking on the floppy disc symbol in the menu bar.

We hope that this chapter could give a first impression of the typical graph development process with ToolIP. This first impression will be substantiated by the detailed descriptions of the graphical user interface and the nodes in the following chapters.
### 3.3 Image data types

With ToolIP, most of the data passing between nodes will be represented in the form of images. Depending on the situation, however, images may contain information of different types, e.g. integer-valued or color information. Table 3.1 gives an overview of the different existing image types and their interpretation.

<table>
<thead>
<tr>
<th>image type</th>
<th>description</th>
<th>range</th>
</tr>
</thead>
<tbody>
<tr>
<td>MONO</td>
<td>just 2 components – foreground and background</td>
<td>0 (background), 1 (foreground)</td>
</tr>
<tr>
<td>GRAY8</td>
<td>8 bit integer gray values</td>
<td>[0, 255]</td>
</tr>
<tr>
<td>GRAY16</td>
<td>16 bit integer gray values</td>
<td>[0, 65,535]</td>
</tr>
<tr>
<td>GRAY32</td>
<td>32 bit integer gray values</td>
<td>[0, 4,294,967,295]</td>
</tr>
<tr>
<td>GRAYF</td>
<td>floating point gray values (single IEEE754 precision)</td>
<td>approx. $[1.17e^{-38}, 3.40e^{38}]$</td>
</tr>
<tr>
<td>COMPLEX</td>
<td>complex floating point gray values (single IEEE754 precision)</td>
<td>approx. $[1.17e^{-38}, 3.40e^{38}]^2$ for both real and imaginary components</td>
</tr>
<tr>
<td>RGB8</td>
<td>Color image, 3 channels, non-interleaved</td>
<td>[0,255] in every channel</td>
</tr>
<tr>
<td>RGB8I</td>
<td>Color image, 3 channels, interleaved</td>
<td>[0,255] in every channel</td>
</tr>
<tr>
<td>RGB16</td>
<td>Color image, 3 channels, non-interleaved</td>
<td>[0, 65,535] in every channel</td>
</tr>
</tbody>
</table>

Table 3.1: ToolIP’s image types

The most frequently encountered image type within ToolIP is GRAYF, i.e., the information stored in every pixel is represented by a floating point number.
Chapter 4

Graphical User Interface

If ToolIP is started, the user finds himself in the main window as shown in figure 4.1. Herein included are the menu bar, the workspace, the plugin menu, the search plugins and the log window. In the following, these individual elements of the graphical interface and its operation are presented in detail.

![Main window of ToolIP](image)

Figure 4.1: Main window of ToolIP

4.1 Menu Bar

At the top of the main window is the menu bar (see figure 4.2). On the menu bar almost all important functions of ToolIP are available, which are described below.
4.1 Menu Bar

4.1.1 New

By clicking on the symbol **New** a new tab for displaying graphs opens. A tab is used to create a new graph. If there is an asterisk next to the tab name visible, the file was changed or not yet saved.

Shortcut: Strg+N

4.1.2 Open File

Loading an existing graph. By clicking on the icon **Open** the Open file... window will be opened for selecting the file to be loaded. The graph will be opened in the current tab in case it is still empty. Otherwise a new tab will be added containing the selected graph.

Shortcut: Strg+O

4.1.3 Save

The generated graph will be saved. The first time you save the graph, the window **Save file as...** opens. Here it is possible to specify the saving location. If the graph was already saved before, the current file will be overwritten while saving.

Shortcut: Strg+S

4.1.4 Save As

By clicking on the symbol **Save as** the created graph can be saved in a user-specified location.

Shortcut: Strg+W

4.1.5 Close Tab

The current tab will be closed. If the graph has been changed, a window appears prompting to save the current graph. Non-saved graphs are indicated by an asterisk symbol in the corresponding tab.

Shortcut: Strg+Q

4.1.6 Reset Graph

 Resets the current graph to the non-processed state. All outputs of the nodes will be removed.
4.1.7 Start the Graph

Processes all plugins in the graph automatically from beginning to the end. Which node is currently running can be recognised by the status button of the node. Already executed nodes have a green connecting line, those which have not yet been processed, a black one. When you open the drop-down box of the Start Graph button you can select Infinite execution, which will perform an infinite execution of the graph. A reset is performed automatically after your graph finished execution.

4.1.8 Stop the Graph

During execution the button turns red. You can always stop the execution process by pressing the Stop Graph button.

4.1.9 Execute Consecutively

Enable this feature for single nodes by right-clicking on the green field. This menu bar icon enables the automatic execution. I.e., if an algorithm is started, all consecutive nodes marked with this flash, will be executed automatically, as soon as the input data is available. This mode can be enabled and disabled at any time.

4.1.10 Undo

The symbol Undo undoes the last action. (see figure 4.3).

Shortcut: Strg+Z

Figure 4.3: Resetting the action connecting nodes

4.1.11 Redo

Redo revokes the treatment Undo. Has an action been undone, the user can get this action back with Redo.

Shortcut: Strg+R

Figure 4.4: Previously undo action withdrawn
4.1 Menu Bar

Graphical User Interface

Note: The undo button is only available for the following actions:

• adding/deleting a node/an edge
• changing node position (e.g. by dragging)
• port linking/Port reordering
• node parameter change (only if this happens as a result of closing the parameter dialog, changing the parameters during running is not saved)
• graph comment change
• setting flash/stop inside a node

4.1.12 Zoom

The user has the opportunity to set the zoom factor in his workspace. The preset value will be transmitted automatically to all tabs opened. For large graphs it can be useful to choose a lower scale in order to see the complete graph. With the adjustment fit graph the zoom settings will be fitted automatically to the current window size. The setting of fit selection ensures that all the selected elements are visible.

4.1.13 Search

Search the current graph for nodes. The search can be performed by label, by name or both, it's also possible to search explicitly for defect nodes.

4.1.14 Auto Layout

Arranges all nodes in the current graph automatically, thereby creating an optimal structure for the graph.

4.1.15 Create Subgraph

Moves a selected part of the graph (see Figure 4.5(a)) into a subgraph (see figure 4.5(b)). A marked node can be recognized by a yellow border. By Ctrl + click or dragging with the mouse multiple node can be selected simultaneously. By clicking on the small turquoise triangle of the subgraph node the subgraph opens in a new window (see figure 4.5(d)).

By clicking on the small green triangle at the left top corner of the open subgraph the window is being attached to the tab bar (see figure 4.5(c)). Subgraphs can be distinguished from main graphs with the help of the corresponding symbol in the tab bar. A main graph is marked with a gray rectangle, subgraphs additionally contain a small dark rectangle in its center. If a subgraph already open, it will be displayed by a yellow triangle in the subgraph node.

When you open the drop-down menu of the Create Subgraph button you can choose Unfold Subgraph which will take the contents of the currently selected subgraph and paste it back into the main graph. The connections in the main graph will be updated accordingly. Afterwards, the subgraph will be deleted.
4.1.16 Memory Information

The bar chart provides information about the current memory usage of the graph. A limit can be specified, at what load an automatic deletion of unnecessary data will be performed.

4.1.17 Timing On/Off

Here, a run-time measurement of all nodes in the graph can be activated. By default it is disabled. In active mode, single nodes can be deactivated by selection of the relevant nodes and the use of the access key Strg+A. The run-time measurement for each node is shown below the parameter field in milliseconds.

Shortcut: Strg+T

4.1.18 Safe Mode

In Safe Mode each single node in the current graph is being executed in a separate process.

Note: The Safe Mode is not available when using the demo version.
4.1.19 Reload Library

**For Windows:** Pressing the button will open the *Reload Library Dialog*. First, unload all the libraries you want to recompile. After unloading the libraries, you can recompile and install them. Before unloading you cannot install them, since the files are in use by Windows (a file which is currently opened cannot be replaced). After recompiling and installing the library you can close the dialog box and all the libraries which were unloaded will be reloaded.

**For Linux:** This function reloads all the libraries of the currently selected nodes. If no nodes are selected, all nodes in the current workspace are used. The 'Reload library' button has a drop-down menu where you can reload all available libraries.

4.1.20 RAGBI - Run A Graph on a Batch of Images or Items

RAGBI is a tool to run the current graph successively on a set of images. A click on the icon opens RAGBI in a new window. It also can be found in the plugin menu at *Generic–>Utility–>RAGBI*. RAGBI can be added into the current workspace via drag & drop. More details in the next section.

4.2 RAGBI

Figure 4.6: The RunGraph window

Figure 4.6 shows the RAGBI window. Various things can be customized, most importantly the graph that is used, as well as input and output images. How this is done is now explained in detail:

1. Selection menu to navigate to the graph that shall be executed, default is the graph currently active in ToolIP
2. Button to reset the graph that shall be executed to the default, i.e. the currently active ToolIP graph (“CURRENT GRAPH”)
3. Input image area: Number of input tabs corresponds to the number of input ports of the graph
(a) Cursor to move the selected image in the list up/down
(b) Selection menu to choose input images. This button is inactive, if your graph has no input port
(c) Delete selected images from list
(d) List of pictures

4. Number of threads used to run graph. Default is to use all available threads (value "0")

5. Path to output directory, i.e. folder where images/outputs are saved

6. Naming convention for the saving the output images. %i[n] stands for input filename of the nth file, i.e. %i1 is replaced by the first input filename, %i2 by the second input filename and so on. %d is replaced by an output index (starting from 1)


8. If %d appears in the output and this box is unchecked, no images (from a previous run) will be overwritten

9. Start/Stop graph execution

Note: If your output image is not IMAGE_GREY_8 RABGI tries to save the result image in tif-format, i.e. your specified output format (7) will be ignored.
4.2 RAGBI Graphical User Interface

(a) RunGraph for one input tab

(b) RunGraph output for one input tab

For further clarification of the output image naming convention: An example, how the output filename will look if there are more than one input ports in the current graph. For the first input the number “1” will be replaced by the first output filename, see figure 4.2(b), if %d appears in the output and the box on (8) is unchecked, no images (from a previous run) will be overwritten. The same will be done for the second input, number “2” will be replaced in the output filename, see figure 4.2(d). This applies for all inputs.

(c) RABGI for two input tabs

(d) RAGBI output for two input tabs

4.2.1 Settings

If you click on the gear button, a dialog box will pop-up allowing you to change the global ToolIP settings. Here you can set for instance how ToolIP saves plugin paths and how the node dialog box is arranged. The settings window is shown in figure 4.7.

4.2.2 About

Opens an extra window showing information about the running ToolIP version.
Figure 4.7: The settings window
4.3 Plugin Menu

![Plugin menu](image)

(a) Plugin menu

(b) Load plugins

Figure 4.8: How to load Plugins in ToolIP

After the first start of ToolIP a standard set of plugins is placed in the plugin menu (see figure 4.8(a)). By right-clicking in the plugin menu, there is a context menu option Load Library/Graphs allowing to load additional libraries (see figure 4.8(b)) or graphs. In the opening window all plugin libraries of interest can be selected. As soon as all the additional plugins are loaded they are also displayed in the plugin menu. The currently loaded set of libraries and graphs can be saved by right clicking and choosing Save Configuration. Also graphs can be added to the plugin menu. This is especially helpful for the design of re-usable modules. They will also be available in the plugin menu and shown each time ToolIP is started. Removing a plugin is possible by right-clicking on the appropriate plugin and selecting Remove Item. It is possible to delete both (sub-)slider and individual plugins. Reloading of libraries is done by right-clicking on a plugin and the selection Reload Library.

The user can see and change the properties of a plugin by right-clicking and selection of Properties. Another right-click function of the plugin is to reset the parameters of the plugin. If the cursor lingers over a plugin, the tooltip will be displayed. Some details about the plugin, the corresponding group, the library name, data input and output as well as parameters are shown.

The plugin menu can be solved, repositioned or closed by the corresponding symbols (at the top right corner) of the ToolIP user interface. The plugin libraries are sorted by tabs. By selecting the relevant library, the plugin of interest can be moved into the workspace by Drag & Drop. Now the plugin can be attached to a graph.

4.4 Search for Plugins

There is a search function for plugins at the top of the plugin menu. The search is equipped with a auto-completion feature. As soon as you type more than two characters in the search box a list of matches for your search query will pop-up. You can Drag & Drop the entries from the auto-completion list directly into the workspace. Also adding the selected entry by hitting Enter.
4.5 Workspace

In the workspace the user can include the required nodes and arrange them into graphs (see figure 4.9). At the top of the workspace window, the respective tabs are listed. Each tab contains a graph or a subgraph. The green triangle in the upper left corner allows the user to change the tab into a floating window. By clicking on the triangle the graph will be inserted in the tab-bar again. The save state of the open graph is indicated by a sign in its tab. If an asterisk icon is present, the graph was changed and should be saved before closing the tab. Whether the tab contains a main graph or a subgraph, can be recognized by the corresponding tab icon.

4.5.1 Graph Parameters

By clicking on the blue rectangle in the top right corner the menu box Graph Parameters is opened as shown in figure 4.10. Here you are able to define new parameters for your graph and add comments. If you are working on a subgraph, the menu box will be called Subgraph Parameters and you can define parameters for your subgraph.

Parameters of nodes inside a subgraph can be defined as parameters of the subgraph. This option is described in detail in section 5.1.2.

4.6 Log Window

In the log window, the entire course of the program is recorded. In here error messages, warnings and possible status output of the nodes are displayed. You can clear the messages from the log window by pressing the brush button at the bottom right of the log window.

Additionally to the log window the session is logged to a file in the user’s appdata directory.

On Windows, the file can be found in C:\Users\YOUR_USERNAME\AppData\Roaming\ITWM\ToolIP.log.

On Linux, the file can be found in $HOME/.ITWM/ToolIP.log.
4.6 Log Window

Graphical User Interface

Figure 4.10: Graph Parameters

Figure 4.11: Log window
Chapter 5

Nodes and Specifications

In the following chapter the different types of nodes and their functionalities are being introduced. In the first section the handling of the basic plugins and the corresponding parameter settings are explained. The latter two sections describe special nodes available for input and output handling of graphs as well as for data flow control.

5.1 Plugins

A plugin node consists of the elements Plugin name, Status field, Parameter field, Node label and data in- and output pin. These elements are described in more detail in the following subsections.

5.1.1 Status field

By clicking on the green status field the selected plugin node will be processed. If the node is currently running, the status field turns red and the node label changes into “running!”. Since most plugins process very fast this change of label is usually not visible. Processing errors are displayed by a warning. Further details on errors can be found in subsection 5.1.6.

With a single right click on the green status box a bolt appears inside the status field. This status corresponds to a consecutive processing of the plugin. As soon as the running of the previous plugin is finished, the following nodes marked with the lightning symbol are executed automatically. By another two-time right-click the automatic processing function is disabled again. This functionality is also available as the tool bar action Execute consecutively and will in this case act as a general circuit for the whole graph.

By two-time right-click on the green status box an executing ban is activated, i.e., further processing of the following branch is prevented. This functionality is useful when during execution certain branches shouldn’t be processed. It can be used in combination with the tool bar actions Start/Stop the graph and Execute consecutively (lightning bolt).
5.1 Plugins

5.1.1 Status Field

After execution the color of the status field of the nodes Display, View and Plotview will change to orange. By clicking on it, the display window will disappear.

5.1.2 Parameter Button

By clicking on the blue parameter button the parameter window (see figure 5.2) will open. In this window all parameters for the plugin can be set. Under the tab Description is a brief description of the node.

![Figure 5.2: Parameter window of a plugin node](image)

Setting parameters is most comfortable using the tab Parameters. Behind each parameter the expected type is shown in parentheses. It is also possible to edit the parameters directly in XML-notation by using the tab XML. The corresponding XML-file of the node will be opened and can be changed. The tab comments can be used to add notes to the current plugin. This might for instance be information about specific parameter settings or a brief description of a subgraph.

If the parameter field is opened and you execute the plugin, the plugin is executed with the parameters set in the Node Parameter Dialog. If you click on Cancel inside the dialog box, the parameters are set back to the initial state.

You are able to change a parameter by using a slider. In order to change the display to sliders, you need to right-click inside the setting-field of a parameter and choose the option Create/Change GUI Controls. The window Create/Change GUI Controls, shown in figure 5.3, will be opened, where you can set the range of the parameter’s slider.

![Figure 5.3: Create/Change GUI Controls](image)

After closing this window, the parameter window will show sliders to enable you to set parameters (see figure 5.4). It is furthermore possible to add the parameters of a node in a subgraph as a parameter of the subgraph. This can be done in the parameter window of the node at the tab Mappings. Here, you need to click on the button
Add Mapping. In the now opening menu choose the parameter you want to add to your subgraph. The chosen parameter will now appear in the tab **Mapping** next to an input field (see figure 5.5).

Please define a name for the variable by using this input field. This name will be displayed in the parameter window of the subgraph (see figure 5.6).

You can un-map a node parameter as a subgraph parameter by clicking on the red X next to the respective mapped parameter in the parameter window of the plugin. Graph parameters only make sense if there is at least one plugin inside the graph that actually maps its value to the subgraph. Gray background line edits at a parameter of a subgraph indicate that there is no plugin inside the graph that maps to this parameter. This is a visual feedback to the user, i.e., this parameter of the graph is no longer necessary or not mapped. At the moment you can only...
delete this mapping as follows: Choose the tab **Xml**. You have to manually delete the line containing your chosen parameter name until the window looks like in figure 5.7.

**Figure 5.6:** The node parameter is now a subgraph parameter.

**Figure 5.7:** This is how the Xml tab should look like after the deletion of all mapped parameters.

If you execute the plugin, while the parameter window is opened, the plugin is executed with the parameters set in the Node Parameter Dialog. If you click on Cancel inside the dialog box, the parameters are set back to the initial state.
Note: Some plugins (matrix::FeatureImage, classification::SVM, manipulation::LabelToFeature,...) use feature plugins as parameters. The current implementation does not work properly when using drag-and-drop from the plugin window. The trick is to specify the feature plugin parameter in advance. This can be done by right clicking on the plugin in the plugin window and selecting the Properties item. Here the feature plugin parameter can be set directly.

In the future, feature plugins will be available as standard plugins and have an online documentation.

Parameter changes can eventually be tested over the button Run. Are for all subsequent nodes up to a node of the category display->image the status fields set to Execute consecutively, the changes can be visualized directly. With OK, the entries are confirmed. Cancel reverses all the changes.

5.1.3 In- and Output Pins

![MultiTh](image)

Figure 5.8: Node in- and output pins

Depending on the functionality each plugin contains one or more in- and output pins: The input pins are on the left, the output pins on the right side of the node. Each output pin can be connected to one or more than one input pins. To each input pin only one output pin can be connected. The connections can be created by selecting and holding a pin and pulling the link to a second pin. By releasing the mouse button the connection is established. Links can also be generated with a simple click on the two relevant pins. Non-connected pins are shown in gray. If a pin is selected, it will be displayed in yellow. As soon as two pins are connected they will become red. In this state the color of the connector is black, indicating that no processing has been performed, yet. After execution of the first plugin the color of the output pins and the related connectors and input pins become green.

5.1.4 Parameter In- and Output

![Divide](image)

Figure 5.9: Parameter In- and Output

It is possible to define the value of a parameter as the result of another node. When lingering the cursor over the bottom of a node a small gray triangle appears. After clicking on this triangle, the parameter input menu pops out as shown in figure 5.9. Here an output and an input pin is defined for every parameter. It is therefore possible to connect the output pin of another node with the input pin of a parameter. The parameter will then be set to the resulting value of the previous node. An example for this use of the parameter pins is shown in figure 5.10.
Parameters of subgraphs can also receive input or return output. By clicking on the gray triangle on the bottom of the node of a subgraph, a parameter input menu pops out, which is similar to the one of plugins. Since the parameters of nodes inside a plugin can be added as subgraph parameters as shown in section 5.1.2, the parameter in- and output of subgraphs allows you to define parameters inside a subgraph to be dependent on a node outside the subgraph. Figure 5.11 shows an example, in which the value of a parameter of a Node inside the subgraph is the output of a node in the main graph.

You can even use the parameter output of another subgraph to be the parameter input of your subgraph (see figure 5.12).

5.1.5 Node Label

By double clicking on the node label (below the status field) a window opens to set a new label for this node. The text before the double lines indicates the label of the node. In addition, a text can be written behind these double lines.
5.2 Linking Ports

Nodes and Specifications

Figure 5.12: Output parameter upcast of subgraph 1 is used as input for parameter upcast of subgraph 2

Figure 5.13: Nodes: Changing name and tooltip

lines. This text will occur as tooltip by longer linger with the mouse over the node.

5.1.5.1 Automatic Node Labeling

Nodes will always be labeled automatically by the scheme "NAME NUMBER", where NAME corresponds to the plugin name, and NUMBER is a continuous number, which starts at 1 for each ToolIP session. If you reload a saved graph, it is not guaranteed, that all nodes will get the same label again, since the labeling is dynamical. But if you give the node a new label, the automatic labeling is over-ruled, and the node will get the same label again.

Note: Plugins like Load or ReadImage inform the user, which image is currently loaded. Therefore this information is directly being displayed by the automatic node labeling. This automatic node labeling is active as long as no user-defined label is defined.

5.1.6 Error Handling

If a node could not be executed, a warning appears below the node. It is a yellow triangle containing an exclamation mark (see figure 5.14). This might for instance be the case when there is a type mismatch between the data at the input pin and the input data settings of the plugin. By clicking on the warning the error message will be displayed, and the warning will disappear.

5.2 Linking Ports

In ToolIP two nodes are available which have a special task in transporting data into and out of a (sub-)graph and have no other functionality. These two nodes, called Input Port and Output Port can be found under Generic—
5.3 Data Flow Control

Port. An example is shown in figure 5.15.

Figure 5.15: In- and output ports

By drag & drop compatible plugins can be attached on top of (be merged into) a port. This way the port acquires the functionality of the plugin. Compatible plugins for input ports are Load, ReadImage and ReadASCII, whereas for output ports the compatible plugins are Display, View, SaveImage, SaveASCII and ScatterPlot. Plugins can only be attached to the ports on the top level graphs, for subgraphs the attachment functionality is always disabled. Ports do not have a label by default, but they will get one as soon as a compatible node has been added.

Note: In the demo mode the plugins SaveImage and SaveASCII are not available and can therefore not be attached to an output port.

5.3 Data Flow Control

In ToolIP four nodes are implemented which allow you to control data flow within a graph: Repeat, Branch, Merge and Switch. These flow control nodes can be found under Generic–>Flow Control.

5.3.1 Repeat Node

The Repeat Node enables the user to include a loop in an algorithm subgraph that is the equivalent to a for loop in programming languages. It has one input. The complete subgraph containing the repeat node will be looped.

As with any loop in programming, the user has to specify an increment in the subgraph and provide a loop condition when the repeat node should stop iterating. At the parameter field of the repeat node, under the tab Port Mappings, the user has to specify which output should be mapped onto which input for the next iteration.

To understand how the Repeat Node is used it is recommended to open the example graph

INST_DIR/examples/graphs/RepeatRotateLena.tlp,

the usage is also shown with additional hints at the key points in figure 5.17.

5.3.2 Branch Node

The branch node allows you to branch out your data flow given some boolean condition. It has two inputs, one for the incoming data (i.e. the image) and one for the branch condition. Depending on the condition it will either
Figure 5.16: Data flow: Repeat

Figure 5.17: Subgraph containing the Repeat node.
place the incoming data onto the first or the second output pin. Thus, the branch node can be thought of as an if/else-statement.

![Figure 5.18: Data flow: Branch](image)

The bottom left input is the data input whereas the top left input represents the conditional input. The conditional input has to be of type `CValueBool`. If the condition evaluates to true, the input data will be placed on the first output, otherwise the data will be placed on the second output. If the conditional input pin is not connected to any other node, the data will be placed on the first output.

![Figure 5.19: Graph using the data flow controls Branch and Merge](image)

Figure 5.19 shows how the branch node can be used in practice. An input image is being read using ReadImage. If the value of the input parameter is not equal to zero, a gaussian smoothing is being performed on the image, otherwise the image remains unchanged.

### 5.3.3 Merge Node

The merge node is the counterpart to the branch node. It allows you to migrate two data streams into one. The merge node forwards its input data to its single output immediately when it arrives. That is, if both inputs of the merge node are connected and if the first input receives data first, it will immediately place this data on its output triggering the next node. Once the second input receives its data it will again trigger the next node with the newly arrived data. The merge node does not wait until all input data is available before it executes itself. If being used in combination with the branch node either the first or the second output of the branch node will be further processed.

![Figure 5.20: Data flow: Merge](image)

Another example showing the combined use of branch and merge can be seen in figure 5.21. For the input image is being checked whether the image width is larger than its height. If this is the case, the image is being transposed, otherwise the image remains unchanged. The merge node combines the two possible outputs and creates a single output stream.
5.3.4 Switch Node

The switch node is somewhat similar to the branch node. Given some conditional input, it will either forward all of its incoming data to corresponding output or block all execution of following nodes completely. Thus it can be considered as a switch allowing you to turn data flow on or off. The conditional input is the input at the top of the switch node. Again, only CValueBool type conditions are allowed. In default mode, every input is forwarded to its corresponding output if the condition is true. However, if you click the true/false button once (so that it displays false) it will forward the input data if the condition is false. If no node is connected to the condition pin it will be considered to be true, thus forwarding data if the true/false button is set to true, and blocking further execution if it is set to false. By default the switch node comes with only one pair of input/output pins. If additional input/output pairs are needed, they can be added by pressing the + button. If you press this button while holding the Shift key, you can remove a pair of input/output pins again.

An alternative way to construct a data flow as shown in 5.19 is by using two switch nodes as shown in 5.23. The top flow puts the input image on the merge node in case the boolean is false. Otherwise, the data stream at the bottom applies a gaussian smoothing on the input image. Only one of the streams is being transferred to the display plugin.
Chapter 6

Further Examples

In this chapter, we take a look at some further examples to point out useful tips and tricks for a better working experience with ToolIP.

6.1 Interactive choice of parameters

The first graph we take a look at can be found in the examples directory under the name threshold.tlp and is displayed in figure 6.1. Concerning the functionality it is very simple: It only performs a threshold on a gray value image. The interesting point with this example is that the Threshold node allows to choose the threshold parameter in a live mode. That means there is a slider in the parameter dialog of the threshold plugin that can be dragged with the mouse (see figure 6.2). Since the following display plugin has a bolt symbol (see Chapter 5) this plugin is executed every time the slider is moved. This makes an interactive choice of the parameter possible. If the path of the graph after the plugin with the slider is longer, it might be that the running time is longer and the update of the display slightly slower. Nevertheless, this can still be helpful for efficiently choosing parameters.

6.2 Characterizing objects and textures

It is very common in practical applications that a segmentation yields certain objects that should be characterized geometrically. Now we turn our attention to an example showing how label images can help to do this (see figure
6.2 Characterizing objects and textures

Further Examples

Figure 6.2: Parameter dialog of the **Threshold** plugin in live mode.

6.3 and the corresponding graph file `labels.tlp`.

Figure 6.3: Characterizing objects and textures: Example graph.

For simplicity, the segmentation part of the graph is represented by the well-known threshold example from the previous section 6.1. This segmentation yields a binary image where the foreground is marked with 1 and the background with 0. After segmenting the objects, they are now labeled in this example. That means we count the connected components in the foreground, and each pixel inside a connected component is mapped to the number of its component. We now show four ways to extract information out of this labeling:

1. The topmost path extracts the size of the objects with the **LabelToSize** plugin.
2. The second path creates a **ZoneList** of the objects and displays it as list.
3. The third path calculates geometric features of the objects with the **LabelToFeature** plugin.
4. The fourth path uses a subgraph to draw the bounding boxes of all detected objects in red into the input image.
Further Examples 6.2 Characterizing objects and textures

Figure 6.4 shows the results of this graph.

Figure 6.4: Results of running the example graph (see Figure 6.3).

With the fourth path we can see how useful it can be to work with subgraphs: The task for drawing bounding boxes in red might appear more often, and so it is possible to save this subgraph in a separate .tlp file (see figure 6.5). We see that ToolIP uses in- and output ports to model the inputs and outputs of the subgraph. It is now possible to load such a subgraph into the plugin menu by right-click onto the menu and choosing **Load Library/Graphs**. Choosing the .tlp file creates a menu entry as shown in figure 6.6. All subgraphs are sorted under the menu point **Generic → XML-Graphs**. This makes it possible to extend the plugin menu by designing graphs for frequently used complex algorithms. This way solutions can be added to ToolIP’s algorithms even without writing C++ code.

Figure 6.5: Subgraph for drawing bounding boxes in red.
6.2 Characterizing objects and textures

Further Examples

Figure 6.6: Menu entry for the subgraph DrawBoxesRed.
Chapter 7

Generation of Runtime Licenses

In this chapter, we describe the procedure to generate a runtime license file. After doing so, you can run your own ToolIP graphs without the dongle and on another PC or notebook. Therefore you could provide others with an image processing application or use your graph solutions by integrating them into your own application framework.

Please note that this is only possible if you own a system builder license for ToolIP. Please note that in all cases this procedure modifies the license file that you are using. So be sure that you did store the original license file safely at another location.

Use the PC or notebook on which you want to be able to use the ToolIP graphs. You need to install ToolIP first, to do so follow the instructions in section 2 and use your own installer. Use your own license file as well (remember: be sure it is stored safely in another spot!).

After installation put your ToolIP Dongle into the USB slot. Wait until it is recognized by the system, then open a command shell as an administrator. You can do this by pressing the windows key and typing “cmd”. Then you have to right click on cmd at the top of the opening menu as shown in figure 7.1. Choose the option ”Run as administrator”.

![Figure 7.1: Run a command shell as administrator.](image)

In the opened command shell you first have to set the path to your license file. Type
set ITWM_BV_LICENSE_FILE=C:\Program Files\ToolIP\share\config\tooltip.lic

as you can see in figure 7.2 in the first prompt (license file is by default stored at that location). Please beware that there must not be any quotation marks in the file path! Then hit enter.

![Figure 7.2: Set the environment variable and unlock license file from dongle.](image)

Second you have to type

```
unlocklicense.exe
```

and again hit enter. If there is no error message but the output as in figure 7.2, you are done!

Please note: The license file now is unlocked from your dongle and bound to the current PC or notebook in use. That is, you can run ToolIP graphs without the dongle, with that license file and the installed version of ToolIP on the current PC or notebook only.
8.1 Introduction

Image processing is becoming a crucial technology in a variety of scientific and applied fields. Computer vision, automated quality assurance, biological and medical research, materials science and many others benefit from advanced image processing algorithms.

ToolIP's integrated library provides a large variety of image processing algorithms in a convenient and flexible framework. It has been designed for ease of use and speed. Available algorithms range from simple pixel operations and image filters to advanced data analysis and machine learning methods. State of the art mathematical morphology and linear algebra are also included.

The algorithms are implemented to handle 2D and 3D image data, both gray scale and color types are supported. This manual is intended as a comprehensive description of the image processing functions provided. Related algorithms are grouped into separate modules (realized technically as different libraries). This structure is reflected in the sections of this manual.

8.2 Toolbox Introductions

8.2.1 Arithmetic

This group of algorithms provides basic arithmetic functions that can be applied either on singles images, or to combine the pixel values from two images:

- Operations on single images: These can be used e.g. to add or subtract constants to or from each pixel.
- Operations between two images, e.g., add two images pixel wise to form the result.

A common application example for such pixel wise operations is the correction of shading artifacts: One can apply a large Gaussian filter kernel and subtract the result from the original image. This may help in situations with varying illumination conditions in images. The basic arithmetic operations are supplemented by more advanced functions such as the RPNC node (reverse polish notation calculator). This allows to easily apply more complex mathematical expressions to pixel values in one step. In order to invert an image, one may use the Negative node, which multiplies every pixel by $-1$. Furthermore, basic binary valued arithmetic operations are available with the nodes And and Or.

8.2.2 Classification

Tools and algorithms related to classification can be found in the classification section. By using these nodes, tasks such as the following ones are tackled:
8.2 Toolbox Introductions

- semantic grouping of segments,
- assigning features vectors to predefined groups,
- detection of statistical outliers.

Next to typical classifiers like KMeans, functions such as PCA or ICA for preprocessing of feature spaces can be found.

Note: The toolbox Classification does not have its own documentation section. You find the node ConvexHull in Chapter Utility Plugins, and the nodes GaussianMixture, ICA, KMeans, PCA, and RANSAC in Chapter Matrix Plugins.

8.2.3 Color

Most algorithms available in ToolIP deal with gray scale images. Yet, for applications involving color images, the nodes in this section provide basic tools to transform color images from and to gray scale data.

- **Separate:** This node allows to extract single color channels (R, G, B) from a color image and outputs them as gray scale image either in 8-bit or float format.

- **Combine:** This node allows to write three gray scale images into the RGB-color channels of a color image.

- **ColorTransform:** Converts an RGB-color image into various other color formats such as HSV or YUV.

Other than the transformations themselves, especially the node ColorTransform can be useful to identify a color representation which sometimes allows e.g. for more accurate segmentation.

8.2.4 Data

The data module provides a set tools for parametrized image creation. This can range from simple geometrical objects like balls, boxes and lines to more sohisticated ones like splines. Random numbers according to different distributions can be generated too.

By combining different data plugins parametrized sequences of test images can be easily created. Another important use is the creation of adaptive masks to facilitate segmentation or the visual presentation of final result of for example a detection algorithm. Still another use is the generation of realization of stochastic geometry models like for example the boolean model.

8.2.5 Display

Display groups together all nodes available for data and image visualization. Thus, the nodes found here are all terminals. The following display nodes are currently available:

- **View** provides fast and simple viewing of 2D and 3D images.

- **Display** is a highly configurable node that allows to display images and data contained in value formats such as ValueMap or ValueDouble.

- **PlotView** is a tool for displaying 1D functions and histograms.
8.2.6 Features

The features module provides tools to characterize image regions. Regions are defined by assigning the same label value to a set of pixels, i.e. pixels with the same label belong to one region. The most common case is that regions correspond to connected components in a binary image, then the label assignment is performed by the node Labeling producing a label image.

Image regions can be characterized by their geometry (shape of the region) or by their content (values of the pixels they contain). Features like form factor, area, boundary length, orientation deal with the geometry aspect. Features like average, variance, entropy or the Haralick features deal with gray values, their distribution and spacial arrangement.

Once a characterization of the regions has been obtained, further algorithms like clustering or classification can be applied to detect common features or classify the regions according to specific criteria.

8.2.7 Filter

The idea of filtering is to replace the value of a pixel by combining the values in its neighborhood. How the neighborhood is defined depends on the specific filter. The filter function can be a linear combination of input values (linear filter) or a more general nonlinear function.

Filtering is typically used in the initial steps of the processing chain. It is a powerful technique for image corrections, image restoration and denoising. Moreover, special filters can be used to detect and characterize relevant structures, as for instance edges, corners or local extrema. Applying some filtering can be also helpful to facilitate segmentation or analysis at later stages.

The library provides several standard linear (e.g. Average, Gauss) and nonlinear (e.g. Median) filters as well as filters based on partial differential equations (PDEs).

8.2.8 LabelImage

In various applications, an image contains more than one region of interest. It is often useful to consider these regions separately. This is possible with the LabelImage plugin group. One of the most important nodes is Labeling, where labels are created from a binary image by incrementally numbering all connected foreground components. Further plugins from this group can be used to assign color, size, or specific values to these regions.

Another frequently used node from the current section is FindMax, which can be used to extract the parameters of lines and circles when combined with Hough and HoughCircle.

Note: The toolbox LabelImage does not have its own documentation section. You find the nodes BoundingBox, FindMax, FindBalancePoint, Labeling, ZoneList in Chapter Utility Plugins, the node Hist in Chapter Features Plugins and the nodes LabelToColour, LabelToFeature, LabelToSize, LabelToValue in Chapter Manipulation Plugins.

8.2.9 Manipulation

The manipulation group of nodes contains various tools to manipulate either the coordinate system or the observation window of an image. The range of functionality of these nodes spans from simple concepts such as translation or rotation up to non-linear coordinate transformations such as CartesianToPolar.

The manipulation nodes find their applications as subroutines in almost all image processing graphs. They are very versatile, which renders them suitable to create highly adaptive and robust graphs.

8.2.10 Matching

Matching or registration is the process of transforming the given data into one common coordinate system. This is usually necessary to allow comparison of image data acquired in different coordinate systems at different times,
or using different imaging techniques. It can also be used to detect known objects in an image or for motion estimation.

### 8.2.11 Matrix

Images can be interpreted as matrices, where each pixel value corresponds to a matrix entry value in a natural way. Image characteristics can be extracted in form of features and stored as an image. This allows to apply linear algebra methods to process and analyse features extracted from an image without the need of an additional external library.

The matrix module provides many standard linear algebra operations (e.g. Multiply, SVD, Eigenvalues) that can be combined to realize more complex algorithms (e.g. Spectral Clustering, Kernel PCA).

### 8.2.12 Morphology

Mathematical morphology encompasses a range of non-linear filter operations with respect to a filter mask, which is called “structuring element” within this context. In morphology, the maximum and minimum filters correspond to the so-called Dilation and Erosion operators, respectively. The resulting functions can be used to correct image artifacts, to reconstruct image structures lost during image acquisition, to detect edges and other predefined structures, or to prepare images for segmentation.

Starting from those basic building blocks, various image processing operations are available, e.g.:

- **Opening** and **Closing** to separate and to join image regions or structures, respectively.
- **FillHoles** and **CutHills** to remove isolated gray value “valleys” and “peaks”.
- **BlackTopHat** and **WhiteTopHat** to emphasize dark and bright regions at least as large as the given structuring element.

The shape of the structuring elements is restricted to rectangles (2D) and cuboids (3D). In all three coordinate directions, its size must be specified by the half lengths.

| Note: By definition, the resulting structuring element will always be of size $2n + 1$ for a given parameter value $n > 0$. |

### 8.2.13 Segmentation

Segmentation is the process of partitioning the image into regions. How the regions are defined depends on the image data or on the intended application. Result of the segmentation of a grey value image can be a binary image, i.e. where the regions consist of background and foreground pixels, foreground usually corresponding to the object of interest. A more complex example would be to identify every single object in a collection resulting in a label image.

ToolIP provides threshold based segmentation procedures (e.g. Otsu) as well as more sophisticated algorithms (e.g. ChanVese).

### 8.2.14 Transformation

This section describes mathematical, reversible transformations between different bases, special transformations suitable for feature extraction as well as a few utilities that enable the handling of the resulting data. Application examples are:

- **Hough** and **HoughCircle** can be used to locate lines and circles in images, respectively.
• Wavelet transformations (OrthoFWT and OrthoIWT) are frequently applied in multi-scale image processing.

• Fourier transformations (FFT) are useful in spectral image analysis, e.g. for texture analysis.

In contrast to most other modules in ToolIP, these transformations operate not in the usual cartesian space.

Note: The toolbox Transformation does not have its own documentation section. You can find the node Distance in Chapter Filter Plugins, the node FFT in Chapter FFT Plugins, the nodes Hough, HoughCircle, Radon in Chapter Utility Plugins, the nodes ModulusMaxima, MultiscaleWavelet, OrthoFWT, OrthoIWT in Chapter Wavelets Plugins.

8.2.15 Utility

The methods in this group can be described as auxiliary functions. One of the most frequently used is ConvertType. This plugin converts the input image into another format that can be used by the next plugin - this is necessary as most of them require a specific image type input (IMAGE_GREY_F is the most common one). Some of them are characteristics that are calculated over the whole image (in contrast to the feature plugin group), such as Statistics, Histogram (1 and 2D), MutualInformation, and more. Others introduce or change the content of the image, such as Normalize, Projection, PaintMarks, and so on.

8.3 Arithmetic Plugins

8.3.1 Plugin Abs
No description available...

8.3.2 Plugin AbsDiff
No description available...

8.3.3 Plugin Add
No description available...

8.3.4 Plugin And
No description available...

8.3.5 Plugin AssertEq
No description available...

8.3.6 Plugin Ceil
No description available...

8.3.7 Plugin Cos
No description available...
8.3 Arithmetic Plugins

8.3.8 Plugin Divide
No description available...

8.3.9 Plugin Equal
No description available...

8.3.10 Plugin Exp
No description available...

8.3.11 Plugin Floor
No description available...

8.3.12 Plugin Log
No description available...

8.3.13 Plugin Math
No description available...

8.3.14 Plugin MathConstant
No description available...

8.3.15 Plugin Maximum
No description available...

8.3.16 Plugin Minimum
No description available...

8.3.17 Plugin Modulus
No description available...

8.3.18 Plugin Multiply
No description available...

8.3.19 Plugin Negative
No description available...
8.3.20 Plugin Or
No description available...

8.3.21 Plugin Pow
No description available...

8.3.22 Plugin RPNC
No description available...

8.3.23 Plugin Sin
No description available...

8.3.24 Plugin SquareRoot
No description available...

8.3.25 Plugin Subtract
No description available...

8.3.26 Plugin Tan
No description available...

8.4 Barcode Plugins

8.4.1 Plugin BarcodeReader
No description available...

8.5 Color Plugins

8.5.1 Plugin ColorMap
No description available...

8.5.2 Plugin ColorTransform
No description available...

8.5.3 Plugin Combine
No description available...
8.5.4  Plugin Separate
No description available...

8.6  Calculator Plugins
8.6.1  Plugin ValueCalc
No description available...

8.7  Data Plugins
8.7.1  Plugin Ball
No description available...
8.7.2  Plugin Box
No description available...
8.7.3  Plugin Circle
No description available...
8.7.4  Plugin Constant
No description available...
8.7.5  Plugin HalfPlane
No description available...
8.7.6  Plugin Line
No description available...
8.7.7  Plugin Noise
No description available...
8.7.8  Plugin PixelValue
No description available...
8.7.9  Plugin Point
No description available...
8.7.10 Plugin PositionToValue

No description available...

8.7.11 Plugin Spline

No description available...

8.7.12 Plugin Value

No description available...

8.8 Features Plugins

8.8.1 Plugin Area

No description available...

8.8.2 Plugin AveragePhi

No description available...

8.8.3 Plugin AveragePolar

No description available...

8.8.4 Plugin BinNbhHist

No description available...

8.8.5 Plugin BinNbhHistPlusXYEntropy

No description available...

8.8.6 Plugin Border

No description available...

8.8.7 Plugin Covbin

No description available...

8.8.8 Plugin Entropy

No description available...
8.8.9  Plugin FromGraph
No description available...

8.8.10 Plugin Geometry
No description available...

8.8.11 Plugin Haralick
No description available...

8.8.12 Plugin HoughEntropy
No description available...

8.8.13 Plugin LBPHist
No description available...

8.8.14 Plugin Median
No description available...

8.8.15 Plugin NbhSpHist
No description available...

8.8.16 Plugin RelnbhHist
No description available...

8.8.17 Plugin RotHistVar
No description available...

8.8.18 Plugin Rough
No description available...

8.8.19 Plugin SDistMedianqDiff
No description available...

8.8.20 Plugin SdistAveVar
No description available...
8.8.21 Plugin Shadow
No description available...

8.8.22 Plugin ShiftHistVar
No description available...

8.8.23 Plugin Statistics
No description available...

8.8.24 Plugin ThreePartsRelNbhHist
No description available...

8.8.25 Plugin TotalVariation
No description available...

8.8.26 Plugin VSlope
No description available...

8.8.27 Plugin XYEntropy
No description available...

8.8.28 Plugin XYInteractionHist
No description available...

8.9 Fast Fourier Transform Plugins

8.9.1 Plugin DCT
No description available...

8.9.2 Plugin FFTW
No description available...

8.10 Filter Plugins

8.10.1 Plugin Average1d
No description available...
8.10 Filter Plugins

8.10.2 Plugin Average3d
No description available...

8.10.3 Plugin Average3dMasked
No description available...

8.10.4 Plugin BinNbh
No description available...

8.10.5 Plugin BoundaryRegion
No description available...

8.10.6 Plugin CEShock
No description available...

8.10.7 Plugin Canny
No description available...

8.10.8 Plugin Convolve
No description available...

8.10.9 Plugin Covbin
No description available...

8.10.10 Plugin Covdet
No description available...

8.10.11 Plugin Diffusion
No description available...

8.10.12 Plugin Distance
No description available...

8.10.13 Plugin FourConnected
No description available...
8.10.14 Plugin Gauss
No description available...

8.10.15 Plugin GaussDerivative
No description available...

8.10.16 Plugin IsoNonlinDiffusion
No description available...

8.10.17 Plugin LBP
No description available...

8.10.18 Plugin Laplace
No description available...

8.10.19 Plugin LocalBinaryPatterns
No description available...

8.10.20 Plugin LowCount
No description available...

8.10.21 Plugin Median
No description available...

8.10.22 Plugin Ranking
No description available...

8.10.23 Plugin RidgeDetector
No description available...

8.10.24 Plugin RobertsCross
No description available...

8.10.25 Plugin Sobel
No description available...
8.10.26 Plugin StructureTensor2D
No description available...

8.10.27 Plugin StructureTensorEVD2D
No description available...

8.10.28 Plugin StructureTensorEVD3D
No description available...

8.10.29 Plugin Variance
No description available...

8.10.30 Plugin ceShock_IsoNonlinDiff
No description available...

8.11 Handling Plugins

8.11.1 Plugin GraphFromFile
No description available...

8.11.2 Plugin GraphOnLabel
No description available...

8.11.3 Plugin GraphOnSlice
No description available...

8.11.4 Plugin GraphOnZone
No description available...

8.11.5 Plugin Loop
No description available...

8.11.6 Plugin LoopAndTile
No description available...
8.11.7 Plugin ParameterLoop
No description available...

8.11.8 Plugin Paste
No description available...

8.11.9 Plugin Tile
No description available...

8.11.10 Plugin TileOneInTwoOut
No description available...

8.12 Manipulation Plugins

8.12.1 Plugin Append
No description available...

8.12.2 Plugin Autocrop
No description available...

8.12.3 Plugin CartesianToPolar
No description available...

8.12.4 Plugin ComposeImage
No description available...

8.12.5 Plugin DeleteDuplicate
No description available...

8.12.6 Plugin Deserialize
No description available...

8.12.7 Plugin Expand
No description available...
8.12 Manipulation Plugins

8.12.8 Plugin Extract
No description available...

8.12.9 Plugin Flip
No description available...

8.12.10 Plugin LabelToColor
No description available...

8.12.11 Plugin LabelToFeature
No description available...

8.12.12 Plugin LabelToSize
No description available...

8.12.13 Plugin LabelToValue
No description available...

8.12.14 Plugin Pad
No description available...

8.12.15 Plugin PlaneToVolume
No description available...

8.12.16 Plugin PolarToCartesian
No description available...

8.12.17 Plugin Reorient
No description available...

8.12.18 Plugin Replicate
No description available...

8.12.19 Plugin Resample
No description available...
8.12.20 Plugin Resize
No description available...

8.12.21 Plugin Rotate
No description available...

8.12.22 Plugin Scale
No description available...

8.12.23 Plugin SelectRays
No description available...

8.12.24 Plugin Serialize
No description available...

8.12.25 Plugin Shrink
No description available...

8.12.26 Plugin Split
No description available...

8.12.27 Plugin SwitchAxes
No description available...

8.12.28 Plugin Translate
No description available...

8.12.29 Plugin VolumeToPlane
No description available...

8.13 Matching Plugins

8.13.1 Plugin FindShape
No description available...
8.14 Matrix Plugins

8.14.1 Plugin BinNbhThFlow
No description available...

8.14.2 Plugin CovarianceMatrix
No description available...

8.14.3 Plugin Determinant
No description available...

8.14.4 Plugin EVD
No description available...

8.14.5 Plugin FeatureImage
No description available...

8.14.6 Plugin GaussianMixture
No description available...

8.14.7 Plugin ICA
No description available...

8.14.8 Plugin Invert
No description available...
8.14.9  Plugin KMeans
No description available...

8.14.10 Plugin Multiply
No description available...

8.14.11 Plugin PCA
No description available...

8.14.12 Plugin QR
No description available...

8.14.13 Plugin RANSAC
No description available...

8.14.14 Plugin RANSACWithModellOut
No description available...

8.14.15 Plugin ReplicateColumn
No description available...

8.14.16 Plugin ReplicateRow
No description available...

8.14.17 Plugin SVD
No description available...

8.14.18 Plugin ShuffleRows
No description available...

8.14.19 Plugin Transpose
No description available...

8.15  Morphology Plugins
8.15.1 Plugin BeucherGradient
No description available...

8.15.2 Plugin BlackTopHat
No description available...

8.15.3 Plugin Closing
No description available...

8.15.4 Plugin CutHill
No description available...

8.15.5 Plugin Dilation
No description available...

8.15.6 Plugin Erosion
No description available...

8.15.7 Plugin ExternalGradient
No description available...

8.15.8 Plugin FillHole
No description available...

8.15.9 Plugin InternalGradient
No description available...

8.15.10 Plugin MidrangeFilter
No description available...

8.15.11 Plugin Minimum
No description available...

8.15.12 Plugin MorphologicalLaplacian
No description available...
8.15.13 Plugin MorphologicalShockFilter
No description available...

8.15.14 Plugin Opening
No description available...

8.15.15 Plugin Reconstruction
No description available...

8.15.16 Plugin SelfDualTopHat
No description available...

8.15.17 Plugin Thinning
No description available...

8.15.18 Plugin WhiteTopHat
No description available...

8.16 Segmentation Plugins

8.16.1 Plugin FastChanVese
No description available...

8.16.2 Plugin FeatureSegmentation
No description available...

8.16.3 Plugin MultiTh
No description available...

8.16.4 Plugin Otsu
No description available...

8.16.5 Plugin RandomChain
No description available...
8.17 Utility Plugins

8.17.1 Plugin AddMetaData
No description available...

8.17.2 Plugin BoundingBox
No description available...

8.17.3 Plugin ConvertType
No description available...

8.17.4 Plugin ConvexHull
No description available...

8.17.5 Plugin CopyImage
No description available...

8.17.6 Plugin CurveIndex
No description available...

8.17.7 Plugin DrawString
No description available...

8.17.8 Plugin FindBalancePoint
No description available...

8.17.9 Plugin FindMax
No description available...
8.17.10 Plugin GetCoordinates
No description available...

8.17.11 Plugin GetImageAttributes
No description available...

8.17.12 Plugin Histogram
No description available...

8.17.13 Plugin Histogram2d
No description available...

8.17.14 Plugin Hough
No description available...

8.17.15 Plugin HoughCircle
No description available...

8.17.16 Plugin ImageInfo
No description available...

8.17.17 Plugin ImageToString
No description available...

8.17.18 Plugin Labeling
No description available...

8.17.19 Plugin MutualInformation
No description available...

8.17.20 Plugin Normalize
No description available...

8.17.21 Plugin PaintMarks
No description available...
8.17.22 Plugin PixelLabel
No description available...

8.17.23 Plugin Radon
No description available...

8.17.24 Plugin RayMaximum
No description available...

8.17.25 Plugin RayMinimum
No description available...

8.17.26 Plugin ReadASCII
No description available...

8.17.27 Plugin ReadImage
No description available...

8.17.28 Plugin ReadMap
No description available...

8.17.29 Plugin ReplaceNonFinite
No description available...

8.17.30 Plugin ReplaceValue
No description available...

8.17.31 Plugin RingBuffer
No description available...

8.17.32 Plugin SampleLine
No description available...

8.17.33 Plugin SamplePoints
No description available...
8.17.34 Plugin SaveASCII
No description available...

8.17.35 Plugin SaveImage
No description available...

8.17.36 Plugin Script
No description available...

8.17.37 Plugin SelectImage
No description available...

8.17.38 Plugin SelectPixel
No description available...

8.17.39 Plugin SetImageAttributes
No description available...

8.17.40 Plugin Sleep
No description available...

8.17.41 Plugin Sort
No description available...

8.17.42 Plugin StaticImage
No description available...

8.17.43 Plugin StaticStorage
No description available...

8.17.44 Plugin Statistics
No description available...

8.17.45 Plugin StringManipulation
No description available...
8.17.46 Plugin StringToImage
No description available...

8.17.47 Plugin TJHoughCircle
No description available...

8.17.48 Plugin TextSerializer
No description available...

8.17.49 Plugin TimerStart
No description available...

8.17.50 Plugin TimerStop
No description available...

8.17.51 Plugin ZoneList
No description available...

8.18 Sysutils Plugins

8.18.1 Plugin Console
No description available...

8.18.2 Plugin EnvVar
No description available...

8.18.3 Plugin FileExists
No description available...

8.18.4 Plugin Platform
No description available...

8.18.5 Plugin ReadTextFile
No description available...
8.18.6 Plugin SaveText

No description available...

8.19 Wavelets Plugins

8.19.1 Plugin ModulusMaxima

No description available...

8.19.2 Plugin MultiscaleWavelet

No description available...

8.19.3 Plugin OrthoFWT

No description available...

8.19.4 Plugin OrthoIWT

No description available...

8.20 Display Plugins

8.20.1 Plugin Display

No description available...

8.20.2 Plugin Load

No description available...

8.20.3 Plugin View

No description available...

8.21 Plot Plugins

8.21.1 Plugin Plotview

No description available...

8.22 Toolima Plugins

8.22.1 Plugin SelectionTool

No description available...
Chapter 9

Summary

In this manual, we introduced the software ToolIP and showed many ways of working with its integrated image processing library. This first steps with ToolIP show its potential and its flexibility - and thus help you by designing solutions for image analysis problems.

In case of any questions concerning or problems with ToolIP and this manual, please do not hesitate to contact us:

toolip@itwm.fraunhofer.de

We appreciate to receive feedback and suggestions!
## Appendix A

## Shortcuts

In the following table the keyboard and mouse shortcuts of ToolIP are summarized:

<table>
<thead>
<tr>
<th>Shortcuts</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl + N</td>
<td>open a new workspace</td>
</tr>
<tr>
<td>Ctrl + O</td>
<td>open graph file</td>
</tr>
<tr>
<td>Ctrl + S</td>
<td>save graph</td>
</tr>
<tr>
<td>Ctrl + Shift + S</td>
<td>save graph as</td>
</tr>
<tr>
<td>Ctrl + W</td>
<td>close current graph</td>
</tr>
<tr>
<td>Ctrl + Z</td>
<td>undo last action</td>
</tr>
<tr>
<td>Ctrl + Y</td>
<td>redo last undone action</td>
</tr>
<tr>
<td>Ctrl + A</td>
<td>select all nodes in the current graph</td>
</tr>
<tr>
<td>Ctrl + C</td>
<td>copy selected items into clipboard</td>
</tr>
<tr>
<td>Ctrl + X</td>
<td>cut selected items (remove items and copy them into clipboard)</td>
</tr>
<tr>
<td>Ctrl + V</td>
<td>paste items from clipboard into current graph</td>
</tr>
<tr>
<td>Ctrl + D</td>
<td>duplicate selected items into current graph (copy&amp;paste)</td>
</tr>
<tr>
<td>Ctrl + left-click</td>
<td>on output pin: open visualization on data (see Settings, first visualization plugin)</td>
</tr>
<tr>
<td>Ctrl + right-click</td>
<td>on output pin: open visualization on data (see Settings, second visualization plugin)</td>
</tr>
<tr>
<td>left click + mouse moving</td>
<td>rectangular selection</td>
</tr>
<tr>
<td>Delete</td>
<td>delete selected items from the current graph</td>
</tr>
<tr>
<td>Insert</td>
<td>open fast insertion box to look for a node (updates selection while typing, navigation by Up and Down key, add selected plugin into current graph via drag&amp;drop or by Enter)</td>
</tr>
<tr>
<td>Ctrl + F</td>
<td>open plugin search window for finding a node by attributes in the workspace</td>
</tr>
<tr>
<td>Ctrl + T</td>
<td>activate/deactivate time measurement</td>
</tr>
<tr>
<td>F2</td>
<td>rename selected node and set tooltip</td>
</tr>
<tr>
<td>right click</td>
<td>open context menu</td>
</tr>
<tr>
<td>left double click</td>
<td>on item in workspace: rename item and set tooltip</td>
</tr>
</tbody>
</table>
Appendix B

Credits for Third Party Libraries

ToolIP uses a number of third party libraries, each released under its individual license terms, see below. The
copyright of these libraries remains with their original authors, see below. Note, however, that this does not affect
the license terms of ToolIP itself in any way. If you would like to obtain the source code of any of the open
source libraries used within this software, please contact Fraunhofer ITWM either by mail or e-mail. Please state
specifically for which one of the open source third party libraries you require the source code and where it should
be sent to.

Fraunhofer ITWM
Markus Rauhut
Fraunhofer Platz 1
67663 Kaiserslautern
toolip@itwm.fraunhofer.de

The following open source third party libraries are used, and their respective authors are acknowledged for their
work:

- Boost under the Boost software license, http://www.boost.org/.
- Qt under the GNU Lesser General Public License v2, http://qt-project.org/.
- zlib under the zlib license, http://zlib.net/.
- Qwt under the Qwt license v1, http://qwt.sourceforge.net/.
- Tesseract under the Apache license v2, https://code.google.com/p/tesseract-ocr/.
- FreeType under The FreeType Project License, http://git.savannah.gnu.org/cgit/freetype/freetype2.git/tree/docs/FTL.TXT.

For the original terms and conditions of each of these libraries, please read the contents of the file

%ITWMDIR%\share\license-thirdparty.txt
Index

About, 32
Action
   Redo, 27
   Undo, 27
Arithmetic Plugins, 59
Auto Layout, 28
Barcode Plugins, 61
Calculator Plugins, 62
Color Plugins, 61
Custom Ui Plugins, 30
Data Plugins, 62
Display Plugins, 81
Fast Fourier Transform Plugins, 65
Features Plugins, 63
Filter Plugins, 65
Graph
   Create Subgraph, 28
   Execute consecutively, 27
   Reset, 26
   run once, 27
   Save, 26
   Save as, 26
   Start, 27
   Stop, 27
Handling Plugins, 68
image
   COMPLEX, 24
   GRAY16, 24
   GRAY32, 24
   GRAY8, 24
   MONO, 24
Log window, 35
Manipulation Plugins, 69
Matching Plugins, 71
Matrix Plugins, 72
Memory information, 29
Menu bar, 25
Morphology Plugins, 73
Nodes, 37, 38
   comments, 38
   parameter, 38

window, 38
Run, 41
XML, 38
ban, 37
bolt, 37
Description, 38
Parameter
   -configuration, 38
   -field, 38
   Status field, 37
Open
   File, 26
   Tab, 26
Plot Plugins, 81
Plugin
   menu, 34
   Search, 34
Reload Library, 30
Running a graph on multiple images, 30
Safe Mode, 29
Search, 28
Segmentation Plugins, 75
Settings, 32
shortcuts, 85
Sysutils Plugins, 80
Tab
   Close, 26
   New, 26
Timing, 29
Toolima Plugins, 81
Utility Plugins, 76
Wavelets Plugins, 81
Workspace, 35
Zoom, 28