Graphical Programming in Alida and ImageJ 2.0 with Grappa

Abstract

Solving challenging image analysis problems usually requires to combine several individual analysis steps into comprehensive workflows. To find a suitable combination of algorithms is often quite elaborate and accomplished interactively. The plugin concept of ImageJ and other comparable tools is well-suited to support such an initial design stage by facilitating rapid-prototyping and comparative algorithm evaluation.

Particularly ImageJ's macro language allows for evolving individual plugins into pipelines, however, requires at least a basic understanding of the logic behind computer programs. Thus, as ImageJ is quite popular in the biomedical research community where programming skills are often rudimentary, simplifying workflow design will further substantiate ImageJ's attractiveness not only for developers, but particularly for life-scientists.

One option to overcome the need for programming skills is to provide graphical programming tools that even non-computer scientists can intuitively use. In this paper we will present our graphical program editor Grappa (Graphical Program Editor for Alida). Grappa allows to design workflows from individual image analysis operators and plugins in a graphical manner. It adopts the JGraph library for graph editing and visualization. Operators and plugins are associated with nodes of the graph, and processing pipelines are defined by linking nodes with directed edges.

A graphical editor for ImageJ will only gain broad acceptance if it offers ImageJ's full functionality. At the same time the effort for developers to make all functionality available should be minimal. While the puristic plugin interface in ImageJ renders external access to its functionality difficult, one of the main targets of the ImageJDev project developing ImageJ 2.0 is exactly to unify parameter declarations of plugins to ease external access.

Grappa is built on top of Alida, a library which follows similar aims and design principles as ImageJ 2.0.

It defines operators as basic units for image analysis. As each operator extends a common super class and annotates its parameters, Alida is able to offer generic mechanisms for operator configuration and execution. This is, e.g., the basis for fully automatic generation of GUIs and ImageJ plugins for operators.

Likewise, each Alida operator (and each Image 2.0 plugin) is automatically exported as a node in Grappa.

By linking nodes with edges workflows are defined, where Alida's standardized parameter annotation allows for instantaneous type checking. Parameter configuration and result visualization after execution are both accomplished graphically adopting Alida's generic I/O mechanisms.

In comparison to other tools for graphical programming Grappa is tuned to a minimum of developer effort.

E.g., while ImageFlow requires to represent ImageJ functionality explicitly in terms of XML specifications, Grappa takes advantage of Alida's and ImageJ 2.0's annotations and automatically includes new plugins and operators. KNIME features advanced workflow design, but to account for the specific needs of image analysis is not straightforward, while Alida may easily be extended to support new data types.

Grappa is under active development and available as prototypical plugin for ImageJ and ImageJ 2.0. While the first one supports only Alida operators, the latter one is capable of also handling ImageJ 2.0 plugins natively.

Keywords

Graphical Programming, Workflows, Editor, Ima geJ 2.0, Alida

Short CV

Birgit Moeller received the Diploma in computer science from the University of Bielefeld, Germany, in 2001, and a PhD from the Martin Luther University Halle-Wittenberg, Germany, in 2005. Since 2002 she is working with the Pattern Recognition and Bioinformatics Group at the University Halle-Wittenberg. While earlier research was mainly dedicated to image registration, visual memories, and computer vision for human-machine interaction, currently her work is focused on the automatic analysis and interpretation of various kinds of biomedical image data, and on the development of software tools and libraries for data and image analysis.

Administrative data

Presenting author: Birgit Moeller
Organisation: Institute of Computer Science, Martin Luther University Halle-Wittenberg

co-authors: Steven Kirchner, Stefan Posch