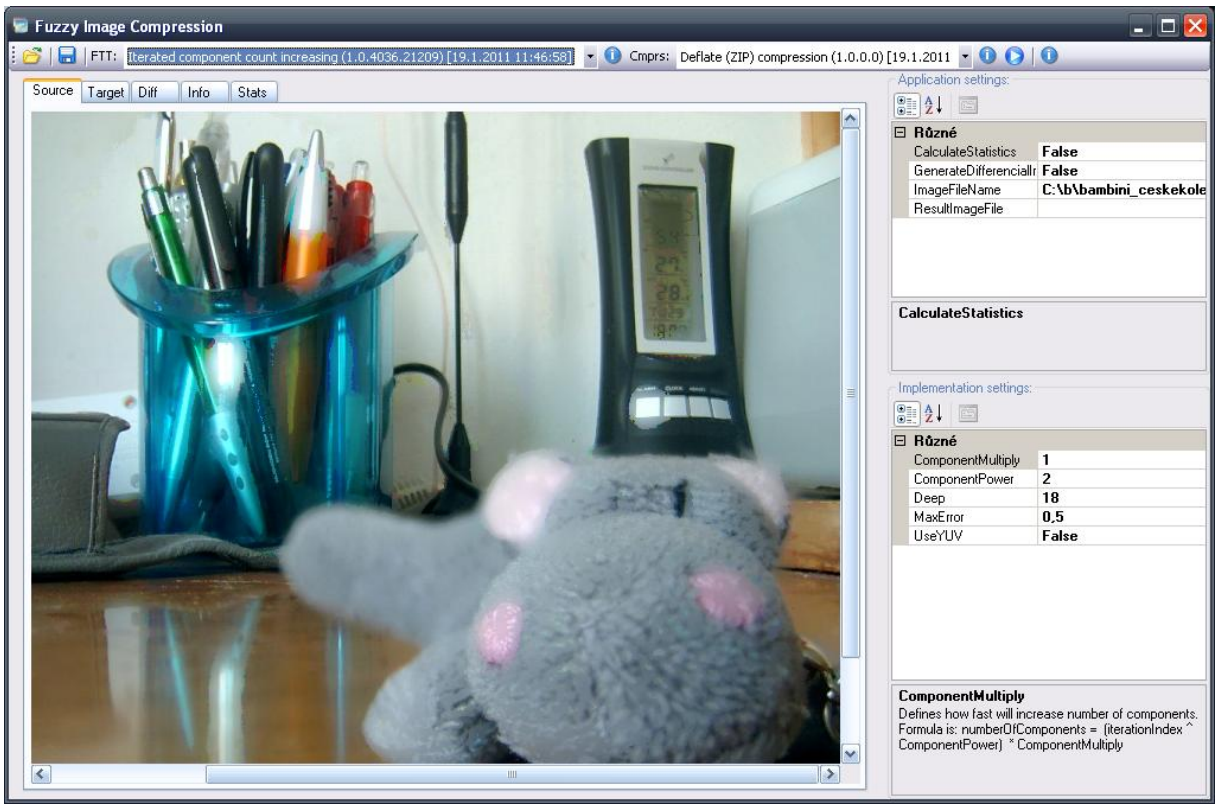
FIC – Fuzzy Image Compression

Ver.: 1. 6

User manual

IRAFM, University of Ostrava, Czech Republic

http://irafm.osu.cz



1 FIC – fuzzy image compression

FIC – fuzzy image compression application uses fuzzy transform described in [http://irafm.osu.cz/en/c94_image-processing/] to compress / decompress the image. It is based on properties of fuzzy transform and its property of capturing information into components. A more detailed explanation about the image compression process and examples can be found on [http://irafm.osu.cz/en/c94_image-processing/ and http://irafm.osu.cz/en/c96_fuzzy-image-compression-tool-fic/] page and its references.

The application takes a set of the selected image and settings as an input and process the forward and inverse fuzzy compression. The algorithm produces the compressed image, backward decompressed image and comparison (if required) between the images with calculated measures of similarity.

2 Installation

The application is a packet with an installer, which will install all required additional prerequisites to run the software. When necessary, internet connection is required to download requirements.

The current version of the application requires:

- Microsoft Windows XP (or higher)
- Microsoft .NET Framework 2.0 (or higher)

You will probably need administrator privileges to execute the installer.

To install the application, execute the archive and execute “setup.exe” file contained in the archive. Then follow the instructions. The Installer will automatically download and install all required missing items.

3 FIC application

FIC application is executed by “FICClient.exe” file located in the target installation folder or start-menu, if this option was selected during installation.

Main application window contain basic parts:

- Topmost task bar.
- Upper and lower property windows on the right side.
- Tab collector with five tab page.

Each part will be explained separately.

3.1.1 Task / menu bar

Task bar contains several icons, buttons, and combo boxes.

Their meaning is:

- **Open image file –** shows a dialog to open a new file from the computer. Supported file formats are BMP, PNG, and JPG.
- **Save image file –** shows a dialog to save the result image file (that is an image after compression and decompression) in a standard bitmap image file format.
- **FTT –** and its corresponding combo box shows all available implemented compression algorithms which were found in libraries located in directory *Implementations* of application folder. The following blue *i* icon will show a dialog with further information about the selected algorithm. The standard install package contains the following implementations (names are followed by version and date of build, all algorithms' properties are explained later in this manual):
  - **1D Basic algorithm via fuzzy transformation –** this implementation does basic one-dimensional fuzzy transformation based on image analysis by rows.
  - **2D Basic algorithm via two-dimensional fuzzy transformation** – this implementation does basic two-dimensional fuzzy transformation based on full image analysis (2D).
  - **Divide & Conquer –** this algorithm divides the image into similar blocks (recursively divided into halves) until criteria is fulfilled; each block is then processed by standard fuzzy transformation.
  - **Similar block compression –** this algorithm processes the image in 1D as long vector, where a similar block is compressed via one-dimensional fuzzy transformation. This implementation seems similar to the previous one, but this one does not divide into halves.
  - **Splitted component range** – this algorithm uses two components at places where a significant image change occurs.
  - **Variable points per base** – this algorithm uses different level of compression on different parts of color models (e.g. in RGB model R component will be less compressed than B component).
- **Cmprs** - and its corresponding combo box show all available standard post-fuzzy-compression methods. The components achieved by fuzzy transformation are compressed by selected method and (if requested) written to the result file. Again, the following blue *i* icon will show dialog with further information about the selected compression method.
- **Run** – blue arrow icon will execute the process of compression/decompression.
- **About** – the last *i* icon will show information about the application.

3.1.2 Application settings & Implementation settings property boxes

Property boxes on the right side of the application window show options and settings for the application and selected implementation. Each option has three parts – name, value and description. Name is in the left side of the table, value is on the right side of the table. Not all properties are writable, some are with fixed value, some can be changed by the different way (e.g. are recalculated from other parameters). A description for the each property is available below each table.

The application properties are the same for all application configurations (that means the properties are the same, but the values can be changed, of course). On the other side, implementation

properties vary and depend on the selected implementation. Each implementation has its own required properties. Most of them are explained further in this manual.

3.1.3 Tab collector & tabs

Tab collector contains five tabs:

- **Source –** this tab shows the currently selected loaded image from the computer which will be compressed.
- **Target –** if available, this tab shows the result image file after compression and decompression.
- **Diff –** if requested (see application options), this tab shows the difference between the source and result image.
- **Info –** this tab shows textual information about the application and progress of compression and decompression.
- **Stats –** if requested (see application options), this tab shows the calculated statistics from the previous algorithm executions.

4 Options

4.1 Application options

Options valid for the whole application are located in the right upper option group box. There are several options:

- **Calculate statistics –** when set to true, whenever the algorithm is executed, the statistics is generated and written in the *Stats* tab.
- **Generate differential image –** when set to true, whenever the algorithm is executed, the differential image between the source and result image is calculated and shown in the *Diff* tab.
- **Image file name –** this property is read only and contains the name of the loaded source file (if any).
- **Result image file –** this property (if entered by user) defines the filename, into which the compressed image file is written. When created, this file contains a compressed image by fuzzy transformation and can be compared with other compression methods

4.2 Individual implementation options

4.2.1 1D and 2D basic algorithm

Those algorithms have following options:

- **Is grayscale –** True if image is grayscale. In that case is processed only once (instead of three times for each color part of R-G-B model).
- **Points per base –** value greater than 2 defining how many pixels will be compressed into one component (for exact definition see web page mentioned at the beginning of the manual).

- **Save result as TXT files –** true if middle components result should be saved into a file in a human-readable text format. The files are automatically generated, named and stored into the current application directory.

4.2.2 Divide and conquer

This algorithm has the following options:

- **Expected number of components –** defines the number of components expected as a result of the detected compressed block.
- **Max range –** defines maximum range of pixel values to add pixel to previous block. If this value exceeds the range, a new block is started and the previous one is sent to compression.
- **Minimum block length –** defines the minimum length of the block.
- **Use Y-U-V –** defines, that the Y-U-V model will be used instead of the R-G-B default model.

4.2.3 Similar block compression

This algorithm uses the default Y-U-V color model and has the following options:

- **Minimal cut value -** defines the minimum range of pixel values to end the previous block and starts a new one. If this value exceeds the range, a new block is started and previous one is sent to compression.
- **Normalizer –** sometimes it could be useful to round the component result value to multiple of some integer; this will lose some information but it may improve compression rate for dictionary compression methods.
- **Points per base divider –** if differs from 1, defines aliquot of component number for U-V (Cr-Cb) components of Y-U-V model. Therefore better compression ratio is gained, but only a bit of quality is lost.

4.2.4 Splitted component range

This algorithm has the following options:

- **Minimum split ratio –** if there is a defined ratio between low and high components on the current component pixels, this value defines the minimum ratio value not to split the interval and use one component. If the ratio exceeds this value, two components are used to describe the interval, one for lower values, one for upper values.
- **Split interval value –** defines where the split edge is. The default value is exactly between 0 – 255 on integer 123 value.

4.2.5 Variable points per base

Settings for this algorithm use marking for the R-G-B or the corresponding Y-U-V model. This algorithm has following options:

- **Bppb** – defines the multiplier for points per base used for B (or V) part of color model.
- **Gppb –** same as the previous, for G (U) part of color model.
- **Rppb –** same as the previous, for R (Y) part of color model.
- **Use YUV –** if true, the YUV model is used instead of the RGB model.

5 Own implementations and extension

A man can create his own implementation of a compression algorithm using existing methods and approaches. The newly created implementation must fulfill these main requirements:

- The final library with implementation must contain data type deriving from type ENG.FIC.Lib.Basic.Classes.Implementation, which is defined in *FICLib* library in file FICLib.dll, with all requested methods and properties. This class is abstract, and defines properties to achieve name, description and author of the implementation, and methods converting the source image into components and vice versa.
- You may use other implemented data types and methods, especially calling the fuzzy processing algorithms located in *FTransformDLLProxy* library and its classes *FTransformByte* and *FTransformDouble*.
- Finally, compiled library have to be placed in the *Implementations* subfolder together with other libraries.

The description of implemented and available methods can be achieved via documentation generated from the XML comments.

6 Uninstallation

The uninstaller of the application will remove all application files. It will not remove installed .NET framework because it can be used by other application. To remove .NET framework, do it manually via “Add or remove programs” dialog in the “Control panel”.

The application stores user settings in the user application folder (typically located at %USER_HOME_FOLDER%\Application Data\FICClient\. This file may be deleted during uninstallation (on request), or it can be deleted manually.

7 Other information

According to continuous research in this area, a newer version of this or similar software may be available. For any further information or comments contact IRAFM staff.

The software has been developed as a part of the project “Data-Algorithms-Decision making” supported by IAA1187301 of the GA AV ČR.

8 Contact

Institute for Research and Applications of FICzy Modeling

University of Ostrava in Ostrava

30. dubna 22

701 03 Ostrava 1

Czech Republic

http://irafm.osu.cz