Ing. David Motl

C-Munipack 1.1

User’s manual

Introduction

At the begining of the 21st century, astronomical photometry has changed a lot. Instead of visual and photoelectric observation, the observers began to use CCD cameras. The CCD cameras became cheaper and therefore more affordable even for amateur astronomers. This gave rise to the demand for software tool for processing the CCD-data, which would be available for the amateur astronomer, user-friendly, but also using reliable algorithms.

In April 2003 the C-Munipack Project was started; its goal was to supply astronomers with powerful but also user-friendly tool for processing their CCD-data. At the time, two similar tools existed: MuniDOS, a text-mode program, which is somewhat limited by the user interface. Additionaly, according to the authors, the program is not using modern algorithms, and no future development is planned. The other available was Munipack, written partly in Fortran and partly in C. Munipack is modern and still maintained, but there were several disadvantages: the users of the Windows operating system would have severe problems with its compilation, and the graphical interface, that program offers, is available only for unix-based operating systems. Of course, there are some commercial products avaiable on the market, some of them are really good, but these products are not affordable for amateur astronomers, since they have to pay for everything by themselves.

I would like to thank to all my colleagues, who contributed with their ideas, advice, and suggestions, to this program. Namely to Filip Hroch for sharing the Munipack source code, and for helping with porting it from Fortran to C, to Lukáš Král for sharing his Varfind source code, to Miloslav Zejda, Ondřej Pejcha and Petr Svoboda for helping with testing and valuable suggestions to the user interface and to Jitka Kudrnáčová and Petr Lut'cha for help with english version of the documentation.

Brno, May 2006 Author

Contents

1 Basic informations

The *C-Munipack* software package, described in this manual, is the huge software package, which offers the complete solution for reduction of images carried out by CCD camera, oriented at an observation of variable stars. The specific programs of reduction process can be called from command line or via intuitive graphical user interface. For developers a dynamic-link library with powerfull interface is available.

The project is based on the previous Munipack package, the programs have the same name, but contrary to Munipack, it is coded in C language and it contains a several new functions and tools in addition. Assignment of the command-line parameters and configuration files is similar in both projects, exceptions are described in the project documentation. Graphical user interface should be familiar for Munidos users, but it takes full advantage of graphical environment, and therefore the interface is more comfortable and enables user an improved control over reduction process in comparison with original interface of Munidos.

Project's WWW pages and FTP archive

The latest version of the C-Munipack's binaries, source codes and documentations are available on the internet web pages and also on the anonymous FTP server:

`http://integral.sci.muni.cz/cmunipack/`

`ftp://integral.sci.muni.cz/pub/cmunipack/`

On authors

The project manager David Motl is also the author of the most part of the source codes. Some small pieces of the sources originate from Munipack package, coded by Filip Hroch. Algorithms for aperture photometry originate from Daophot software by P. B. Stetson. Munifind algorithm originates from Varfind tool coded by Lukáš Král.

The package uses the FITSIO library (Dr. William Pence, NASA), the FreeImage library maintained by Hervé Drolon and the James Clark's EXPAT library. See *References* section for further details.

Software license

Documentation license

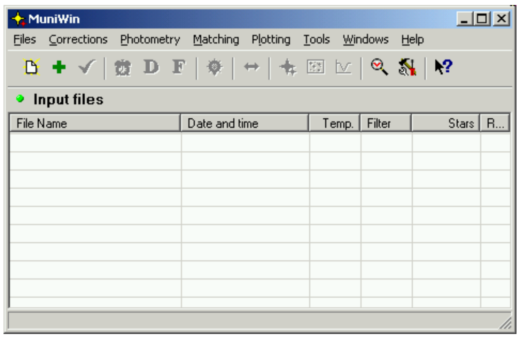
2 The Muniwin program

The Muniwin program is the graphical user interface, which presents system for CCD frames reduction, starts with source files conversion into the image format with computation of light-curve of the selected stars.

2.1 Basic processing of the observation

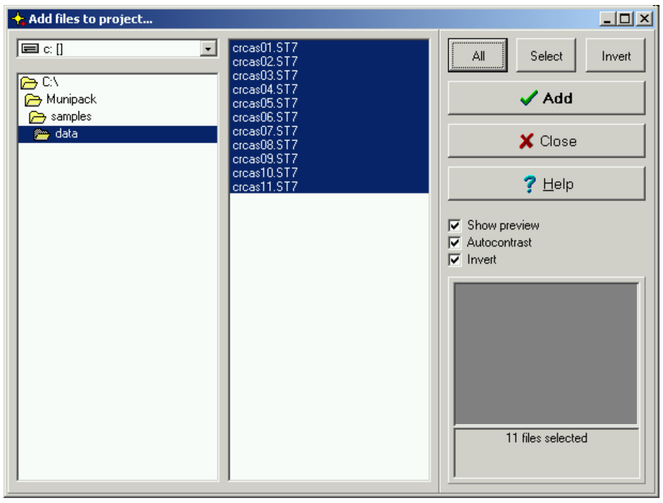
The following procedure is the basic procedure for processing the observation of the short-periodic variable stars. From the internet pages of this project, the package with demo data can be dowloaded; this set of data can be used for testing this procedure. Unpack the dowloaded archive to any empty folder at your hard disk.

1. Launch the *Muniwin* program – click on the *Start* button, select *Programs* folder and in the *Muniwin* folder click on the *Muniwin* icon.


MuniWin
Files Corrections Photometry Matching Plotting Tools Windows Help
Input files
File Name
Date and time
Temp.
Filter
Stars
R...

2. From the *Files* menu choose *Clear files*. You will be asked for confirmation, press the *Yes* button. The table of files in the window *Input files* should be now clear.

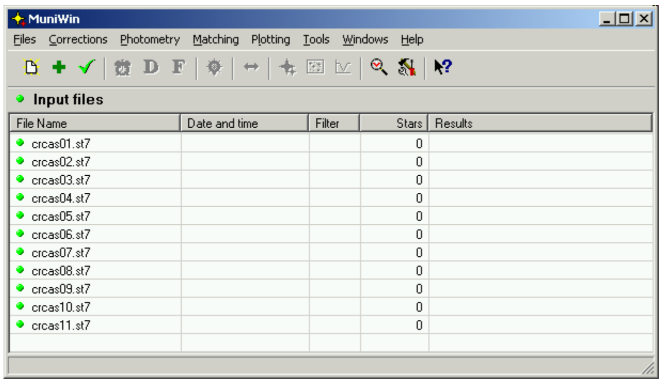
3. From the *Files* menu choose option *Add files*. With this dialog the images are inserted into the table of the input files. In the left window find the folder containing the data (the demo data are stored in the `data` subfolder). If you want to add all the files from the folder, push the *All* button. If you don't want to, mark the desired files in the table using a mouse. You can also use the dialog for choosing the files based on their properties; this dialog is opened by clicking at the *Select* button.



crcas01.ST7
crcas02.ST7
crcas03.ST7
crcas04.ST7
crcas05.ST7
crcas06.ST7
crcas07.ST7
crcas08.ST7
crcas09.ST7
crcas10.ST7
crcas11.ST7

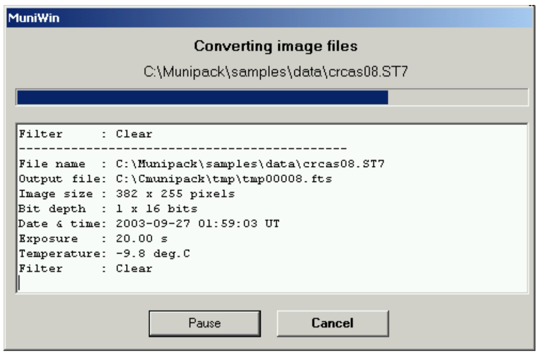
11 files selected

4. Push the *Add* button. If you want to process data taken in several nights at once, use the same procedure to add those files to the table. After insering all the files you want to process, close the dialog by pushing the *Close* button.


MuniWin
Files Corrections Photometry Matching Plotting Tools Windows Help
D
F
Input files

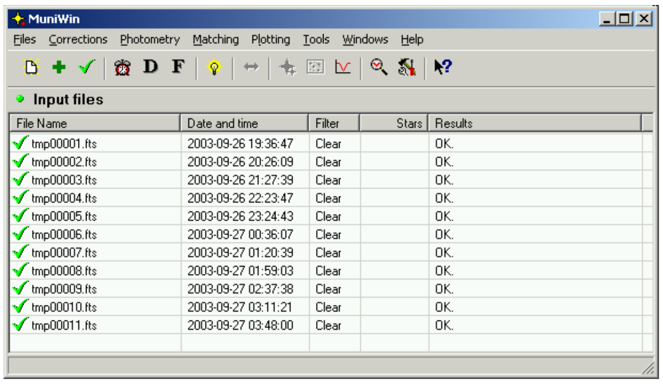
File Name	Date and time	Filter	Stars	Results
crcas01.st7			0	
crcas02.st7			0	
crcas03.st7			0	
crcas04.st7			0	
crcas05.st7			0	
crcas06.st7			0	
crcas07.st7			0	
crcas08.st7			0	
crcas09.st7			0	
crcas10.st7			0	
crcas11.st7			0	

5. Now do the conversion of the input data into the FITS file format and save the data to the temporary files folder. This step must be done, even when the input data are stored in the FITS file format (the program will just copy the files into the temporary folder). From the *Files* menu choose the *Fetch/convert files* option and in the dialog press the *OK* button.



```
Filter      : Clear
--------------------------------------------
File name   : C:\Munipack\samples\data\crcas08.ST7
Output file: C:\Cmunipack\tmp\tmp00008.fts
Image size : 382 x 255 pixels
Bit depth   : 1 x 16 bits
Date & time: 2003-09-27 01:59:03 UT
Exposure    : 20.00 s
Temperature: -9.8 deg.C
Filter      : Clear
```

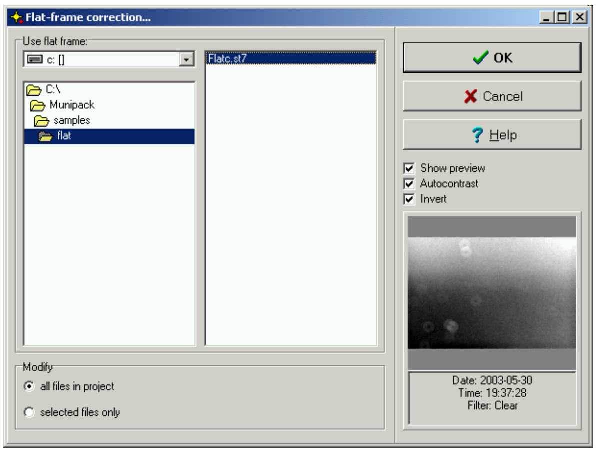
6. During the conversion a new window will be opened, in which the state of the process is displayed; all the information is also presented there. This window will be automatically closed after finishing the process. Wait for the conversion of all the files to finish. After finishing, the icon in the file table will be changed; the information about the time of observation, the length of the exposition and the used filter will be added. In case, some of the files could not be converted, the file will be marked with a special icon and in the *Result* column the error message will be reported.


MuniWin
Files Corrections Photometry Matching Plotting Tools Windows Help
D
F
Input files

File Name	Date and time	Filter	Stars	Results
✓ tmp00001.fts	2003-09-26 19:36:47	Clear		OK.
✓ tmp00002.fts	2003-09-26 20:26:09	Clear		OK.
✓ tmp00003.fts	2003-09-26 21:27:39	Clear		OK.
✓ tmp00004.fts	2003-09-26 22:23:47	Clear		OK.
✓ tmp00005.fts	2003-09-26 23:24:43	Clear		OK.
✓ tmp00006.fts	2003-09-27 00:36:07	Clear		OK.
✓ tmp00007.fts	2003-09-27 01:20:39	Clear		OK.
✓ tmp00008.fts	2003-09-27 01:59:03	Clear		OK.
✓ tmp00009.fts	2003-09-27 02:37:38	Clear		OK.
✓ tmp00010.fts	2003-09-27 03:11:21	Clear		OK.
✓ tmp00011.fts	2003-09-27 03:48:00	Clear		OK.

7. In the next step make all the necessary corrections of the images. The demo data needs only flat field correction, because the rest of the corrections was done automatically during the

observation. From the *Corrections* menu choose the *Flat field* option. Search and mark the file containing a flat field correction image (in the demo data use the `flat20v.st7` file from the subfolder `flat`) and close the dialog by pressing the *OK* button.



Use flat frame: c: []

Flatc.st7

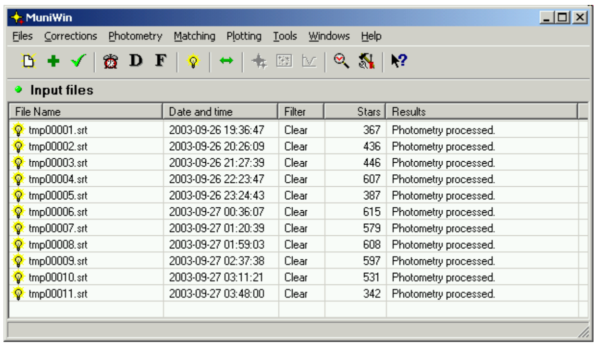
C:\
Munipack
samples
flat

[✓] Show preview
[✓] Autocontrast
[✓] Invert

Date: 2003-05-30
Time: 19:37:28
Filter: Clear

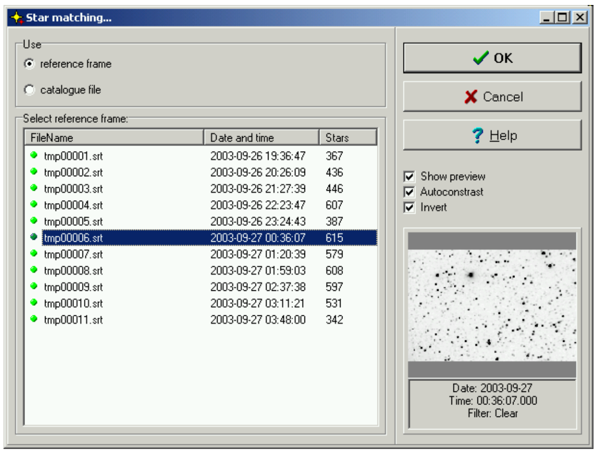
Modify
[✓ all files in project]
selected files only

8. After all necessary corrections comes the photometry of the images, e.g. finding the stars and determining their brightness. From the *Photometry* menu choose the *Photometry* option and in the dialog press the *OK* button. After finishing the processing of the images, the number of the stars found on each of the images will be written to the file table.


MuniWin
Files Corrections Photometry Matching Plotting Tools Windows Help
Input files

File Name	Date and time	Filter	Stars	Results
tmp00001.srt	2003-09-26 19:36:47	Clear	367	Photometry processed.
tmp00002.srt	2003-09-26 20:26:09	Clear	436	Photometry processed.
tmp00003.srt	2003-09-26 21:27:39	Clear	446	Photometry processed.
tmp00004.srt	2003-09-26 22:23:47	Clear	607	Photometry processed.
tmp00005.srt	2003-09-26 23:24:43	Clear	387	Photometry processed.
tmp00006.srt	2003-09-27 00:36:07	Clear	615	Photometry processed.
tmp00007.srt	2003-09-27 01:20:39	Clear	579	Photometry processed.
tmp00008.srt	2003-09-27 01:59:03	Clear	608	Photometry processed.
tmp00009.srt	2003-09-27 02:37:38	Clear	597	Photometry processed.
tmp00010.srt	2003-09-27 03:11:21	Clear	531	Photometry processed.
tmp00011.srt	2003-09-27 03:48:00	Clear	342	Photometry processed.

9. Now shall be done the composition of the photometric files made in the previous step, e.g. finding the corresponding stars on all the files using one chosen image. From the *Matching* menu choose the *Match stars* option. In the dialog click on the option *Reference frame* and mark the image you want to use as the reference one and confirm your choice by pushing the *OK* button.



Use
[✓ reference frame]
catalogue file

Select reference frame:

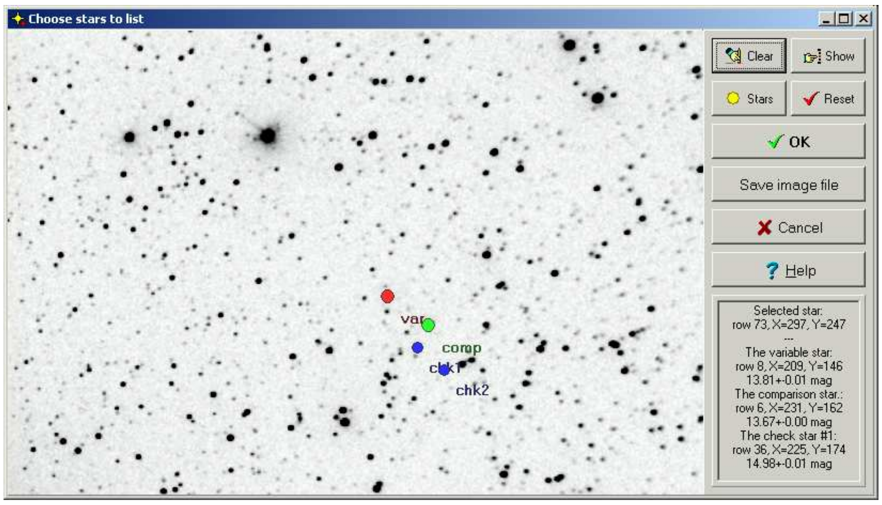
FileName	Date and time	Stars
tmp00001.srt	2003-09-26 19:36:47	367
tmp00002.srt	2003-09-26 20:26:09	436
tmp00003.srt	2003-09-26 21:27:39	446
tmp00004.srt	2003-09-26 22:23:47	607
tmp00005.srt	2003-09-26 23:24:43	387
tmp00006.srt	2003-09-27 00:36:07	615
tmp00007.srt	2003-09-27 01:20:39	579
tmp00008.srt	2003-09-27 01:59:03	608
tmp00009.srt	2003-09-27 02:37:38	597
tmp00010.srt	2003-09-27 03:11:21	531
tmp00011.srt	2003-09-27 03:48:00	342

[✓] Show preview
[✓] Autocontrast
[✓] Invert

Date: 2003-09-27
Time: 00:36:07.000
Filter: Clear

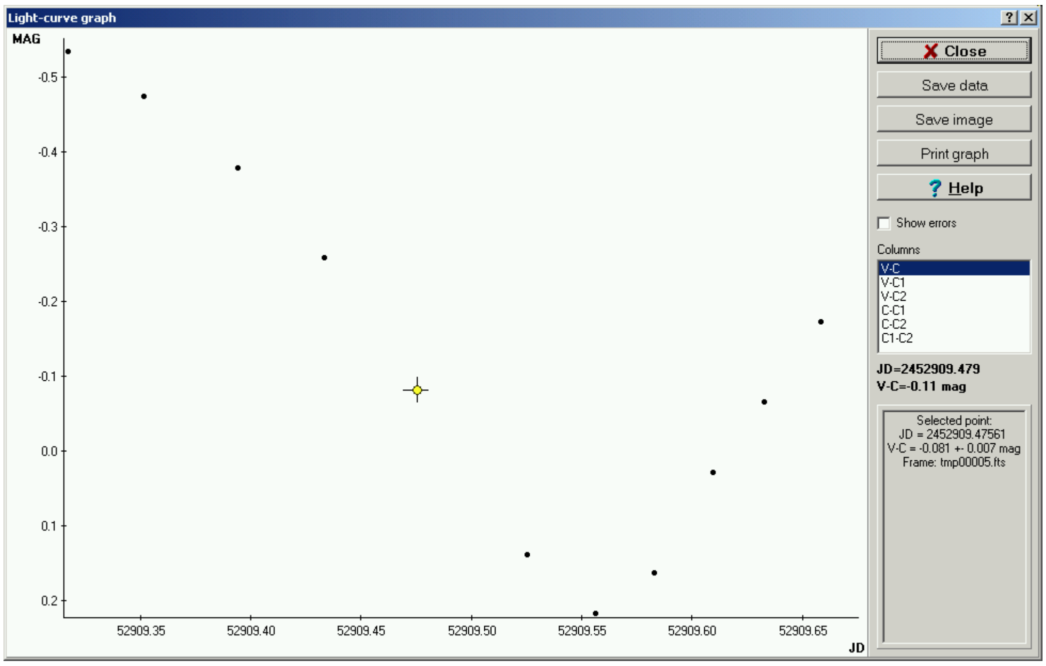
If using the demo data, choose the `tmp00006.srt` image. The best image is usually the one with the most stars on it and in the preview, displayed in the right-bottom corner of the dialog, this image should not be visibly damaged.

10. The last part of the reduction is matching of the light curve – in the *Plotting* menu click on *Choose stars*. In the dialog window the reference image will be displayed. The star nearest to the cursor of the mouse is marked with a cross. On the image click on the measured (variable) star and a comparison star respectively, you can also add several control stars. If using the demo data, use the stars according to the following picture:



```
Selected star:
row 73, X=297, Y=247
---
The variable star:
row 8, X=209, Y=146
13.81+-0.01 mag
The comparison star.:
row 6, X=231, Y=162
13.67+-0.00 mag
The check star #1:
row 36, X=225, Y=174
14.98+-0.01 mag
```

11. Confirm your choice by clicking on the *OK* button. The program will compute the brightness of the chosen stars and displays the light curve – a graph of the relative brightness of the variable star according to the comparison star in depence of the time measured in the shortened Julian date (JD-2400000.0)



```
JD=2452909.479
V-C=-0.11 mag

Selected point:
JD = 2452909.47561
V-C = -0.081 +- 0.007 mag
Frame: tmp00005.fts
```

12. Press the *Save data* button. Locate the folder where you want to save the data file with the results and set the name of the output file. Close the dialog by clicking on the *Close* button. The output file contains on the first two lines the heading, then the table containing data. Each image corresponds to one row in the table, in the first column is full Julian date, the second one contains the relative brightness of the variable star in comparison to the comparison star and the third one contains the error of the brightness. In the following columns are saved the brightness and the errors of the comparison star and the control stars (for more detailed description see the documentation of the Munilist program).

```
JD V-C s1 V-C1 s2 V-C2 s3 C-C1 s4 C-C2 s5 C1-C2 s6
Aperture: 1, Filter: Clear, JD: geocentric
2452909.31733 -0.528 0.007 -1.822 0.014 -2.024 0.016 -1.294 0.014 -1.496 0.016 -0.202 0.020
2452909.35161 -0.465 0.007 -1.721 0.015 -1.977 0.016 -1.256 0.015 -1.512 0.017 -0.256 0.021
2452909.39432 -0.371 0.007 -1.644 0.014 -1.871 0.016 -1.273 0.014 -1.500 0.016 -0.227 0.020
2452909.43330 -0.253 0.006 -1.549 0.011 -1.758 0.012 -1.296 0.011 -1.505 0.012 -0.209 0.015
2452909.47561 -0.084 0.007 -1.364 0.013 -1.588 0.015 -1.280 0.013 -1.504 0.015 -0.224 0.018
2452909.52520 0.140  0.006 -1.167 0.010 -1.364 0.012 -1.307 0.010 -1.504 0.012 -0.197 0.014
2452909.55612 0.226  0.006 -1.062 0.011 -1.292 0.012 -1.288 0.011 -1.518 0.012 -0.230 0.015
2452909.58279 0.165  0.006 -1.136 0.011 -1.334 0.012 -1.301 0.011 -1.499 0.012 -0.198 0.015
2452909.60958 0.034  0.006 -1.244 0.011 -1.451 0.012 -1.278 0.011 -1.485 0.012 -0.207 0.015
2452909.63300 -0.059 0.007 -1.356 0.012 -1.566 0.014 -1.297 0.012 -1.507 0.014 -0.210 0.017
2452909.65845 -0.167 0.011 -1.445 0.023 -1.665 0.027 -1.278 0.023 -1.498 0.027 -0.220 0.034
```

2.2 The application's main window

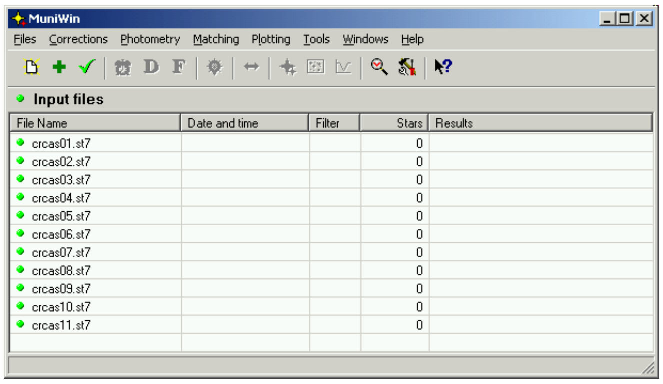
After the initialisation of the program, the application's main window opens. The main window consists of the menu bar, the tool bar for quick access to frequently used functions and table of input files. The program saves its present state to the project file located in the temporary folder, so the last state is automatically restored after initialisation phase.

2.2.1 The table of input files

The table of input files is displayed in the program's main window. In the first step of the reduction process, the source image files are inserted to this table. Those files can be located in different directories – so you can process the images observed in different nights without necessity of copying them into one folder. Each row in the table contains many attributes, except source file name, e.g. current state indicator, temporary file name, photometry file name, result of last operation, etc. This attributes are actualized after each step during reduction process.

Context menu, which is open by click on right mouse button within the table, consists of many useful features, which allows to modify the appearance of the table. The list can be set to one of three modes:

The Table mode


MuniWin
Files Corrections Photometry Matching Plotting Tools Windows Help
Input files

File Name	Date and time	Filter	Stars	Results
crcas01.st7			0	
crcas02.st7			0	
crcas03.st7			0	
crcas04.st7			0	
crcas05.st7			0	
crcas06.st7			0	
crcas07.st7			0	
crcas08.st7			0	
crcas09.st7			0	
crcas10.st7			0	
crcas11.st7			0	

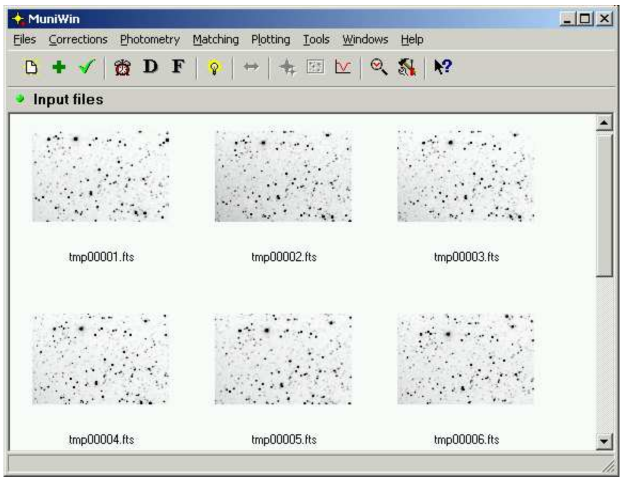
In this mode, the content of the table is presented in simple table form. The small icon in the first column indicates the process state or the error generated in the last procedure respectively. More further informations about a frame is displayed in the *Properties* dialog – the right mouse button click on any row, select the *Properties* in the pop-up menu. By selecting the other items in this menu, you can open the selected frame as an image or an chart in full resolution. At the bottom of the window the status bar is placed showing the full path and name of the source file corresponding to the selected frame.

To display nonstandard columns (i.e. date and time of observation in julian date mode, temperate etc.), click on the right mouse button, open the *Data shown* item. The dialog window opens. Check on / off the columns you want to display / hide and confirm the dialog by the *OK* button.

The Previews mode

In this mode, the small input file previews are shown in the window. After one step of the reduction process is finished, the previews are automatically actualised showing the result of the last step. Similarly as in many other dialogs, you can modify the previews properties:

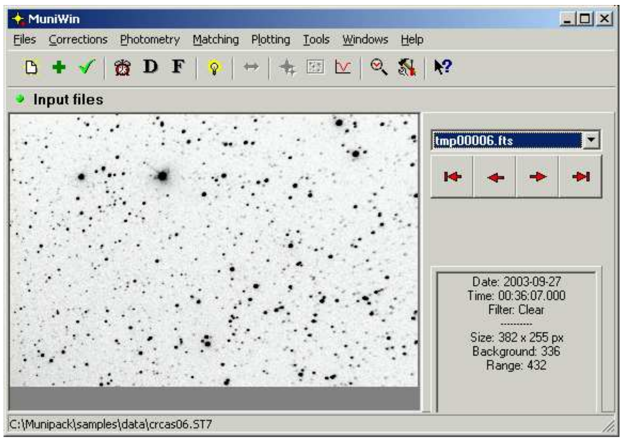
- *Autocontrast* – check on if you want the program to compute automatically the brightness and contrast of displayed image. Uncheck the box if you want to use the values stored in the file header.
- *Invert* – switches positive and negative display rendering.



The Slideshow mode

In this mode, the image or photometry file (chart) in full resolution is rendered on the window. After one step of the reduction process is finished, the image is automatically actualised showing the result of the last step. Use the arrows in the window to move to the first, previous, next or last frame in the table respectively. You can also move to any frame by selecting it in the selection filed (up right). Similarly as in many other dialogs, you can modify the properties of the preview:

- *Autocontrast* – check on if you want the program to compute automatically the brightness and contrast of displayed image. Uncheck the box if you want to use the values stored in the file header.
- *Invert* – switches positive and negative display rendering.


MuniWin
Files Corrections Photometry Matching Plotting Tools Windows Help
Input files
tmp00006.fts

Date: 2003-09-27
Time: 00:36:07.000
Filter: Clear

Size: 382 x 255 px
Background: 336
Range: 432

C:\Munipack\samples\data\crcas06.ST7

2.2.2 Preview window

The preview window allows user to further inspect the content of a CCD frame or a result of reduction process. The dialog window is open by double-click on one of the files in the table of the input files. On the basis on the current state of the reduction process, it is automatically decided, whether an original frame, a temporary frame or a photometry file shall be displayed. The preview window can be open also from the context menu (*Open* item) – in this case, it allows user to choose, which style of preview shall be displayed, by selecting corresponding item in the submenu.

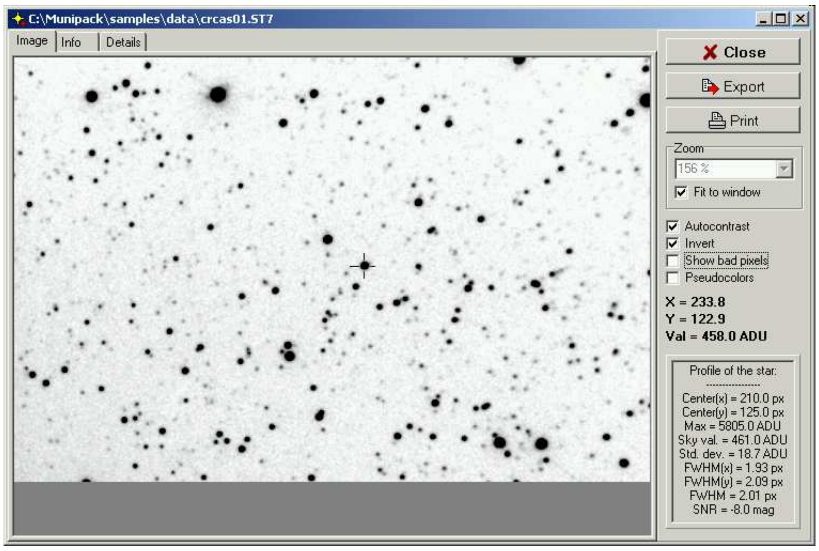
Click the *Close* button to close the dialog. It is possible to export a frame to a file in the JPEG, PNG, TIFF or BMP format (the *Export* button). The *Print* feature is also supplied.

Displaying the CCD frame

The information is arranged in the three pages, which are switched by tabs on the upper part of the dialog. The CCD image is presented on the first page. The second page consists of brief information about the frame. Complete listing of all items from the header is displayed on the third page.

The controls located on the right panel of the dialog are intended for setting the preview options:

- *Zoom* – this section sets the magnification of the image. You can either select one item in the combo box or check the *Fit to window* field. In the latter case, the zoom is automatically adjusted to fit the window size.
- *Autocontrast* – check on if you want the program to compute automatically the brightness and contrast of displayed image. Uncheck the box if you want to use the values stored in the file header.
- *Invert* – switches positive and negative display rendering.
- *Show bad pixels* – check this option on to highlite overexposed or bad pixels on the frame. The pixels, whose value is equal to maximum (65535 by default), equal to zero or even negative, are dyed in red color.
- *Pseudocolors* – switches grayscale and pseudocolor display rendering. This function is useful for checking uniformity of the sky level after flat-frame correction.
- *Show scale* – check on to display the scale of the pixel values. The lables shows values in ADU corresponding to intensity or color hue.
- *Show apertures* – check this option to display the default aperture and the sky annulus. The default aperture is displayed in form of a circle drawn by a solid pen, two dashed circles shows the annulus used for computing a value of local sky brightness.



156 %
[✓] Fit to window
[✓] Autocontrast
[✓] Invert
[ ] Show bad pixels
[ ] Pseudocolors

```
X = 233.8
Y = 122.9
Val = 458.0 ADU

Profile of the star:
----------------
Center(x) = 210.0 px
Center(y) = 125.0 px
Max = 5805.0 ADU
Sky val. = 461.0 ADU
Std. dev. = 18.7 ADU
FWHM(x) = 1.93 px
FWHM(y) = 2.09 px
FWHM = 2.01 px
SNR = -8.0 mag
```

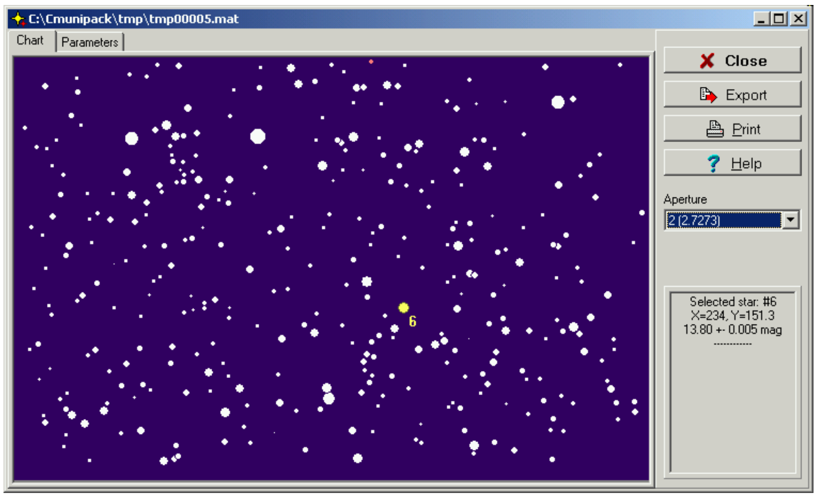
The preview dialog allows to find out the basic photometric attributes of a star on the CCD frame before the photometry reduction phase is executed. This feature is intended for checking whether a star is not overexposed and it allows to determine the approximate value of the FWHM value (Full Width at Half Maximum). The computation is initiated by single click on left mouse button within the image. The highest peak is found in the vicinity to position, where the user has clicked on. Then, the simplified photometry calculation is performed. The results are print out to the information panel in the bottom right part of the dialog.

- *Center(x), Center(y)* – centroid's position (center of the star) in pixels with respect to upper left corner of the image.

- *Max* – pixel value in the local maxima in ADU
- *Sky val.* – local mean value of sky level in ADU, computed by means of median algorithm.
- *Std. dev.* – standard deviation of sky level in ADU.
- *FWHM(x), FWHM(y), FWHM* – half width of the profile in horizontal direction, in vertical direction and average of the previous two values respectively.
- *SNR* – ratio of the integral signal of the star to std. deviation of the sky level, expressed in magnitudes (the SNR value equal to -2.5 mag. means, that the signal of the star is 100 times greater then noise of its background).

Displaying the photometry file

The information is arranged in two pages, which are switched by the tabs on the upper part of the dialog. The chart is displayed on the first page. The diameters of circles correspond to the magnitudes of stars. Stars, which could not be measured, are painted in red color. The second page consists of complete listing of all items from the header.



Aperture 2 (2.7273)

Selected star: #6
X=234, Y=151.3
13.80 +- 0.005 mag

By single click on a star, the further data is displayed in the information panel, which is located in the bottom right part of the dialog window. The selected star is highlited by yellow color and labeled by its ordinal number.

You can also change the index of aperture used by means of the *Aperture* control.

2.3 The Files menu

2.3.1 Clear files

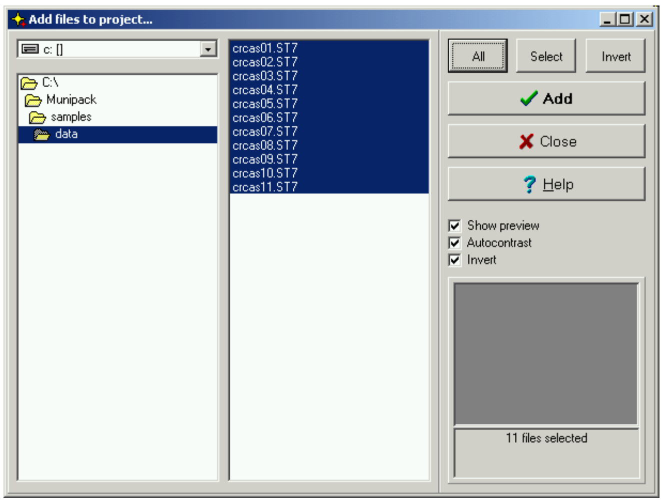
This function deletes all items stored in the *Input files* table and cleans the directory with temporary files (Files menu). It does not remove the source files from the disk. If you want to remove any selected files from the table, use the *Remove files* function instead.

2.3.2 Add files

Opens a dialog for files insertion into the table of input files. It is always used at the beginning of reduction process or prior the start of *Process new files* tool. This dialog is a standard way how to add files into the project. The *Drag & Drop* function and/or *Copy & Paste* can be used alternatively.

The Add files dialog

Choose the *Add Files* item from the *Files* menu. The dialog window appears. Find the directory and select the files you want to add to the project. Press the *Add* button. The dialog remains open, so you can include the files from another folder, simply by changing the directory and repeat the addition. Duplicities are automatically ignored. The *Close* button closes the dialog.



For selection of large number of files, use the standard way with `Ctrl` or `Shift` keys or click on function buttons in the right panel:

- *All* – select all files in the current folder
- *Invert* – inverts the selection of files
- *Select* – allows selection of the files based on their properties (date and time of observation, filter and file name)

Selecting file (containing the CCD image in supported format) from the list, an image preview with a brief information is shown at the left bottom of the dialog window. Double-click on preview opens a dialog with full resolution image. The preview features can be modified by selecting the check boxes:

- *Preview* – turns the preview display on and off. It's a good idea to turn off the preview on slow computers.

- *Autocontrast* – check on if you want the program to compute automatically the brightness and contrast of displayed image. Uncheck the box if you want to use the values stored in the file header.
- *Invert* – switches positive and negative display rendering.

The Drag & Drop function

From the *Windows* menu, open the *Show Input files* window. Using Windows explorer (This computer) find the folder, where the source files are located. Mark the files you want to process, press and hold the left mouse button. Drag the files onto *Muniwin* icon on system bar. Wait till the program switches to foreground, then move the mouse to the *Input files* window and release the mouse button.

By the similar way, you can drag and drop files from any other file manager.

The Copy & Paste function

From the *Windows* menu, open the *Show Input files* window. Using Windows explorer (This computer) find the folder, where the source files are located. Mark the files, you want to reduce, and press the right mouse button. In the context menu, select the *Copy* item. Switch back to Muniwin program, and in the *Input files* window press the right button again. In the context menu, select the *Paste from clipboard* item.

By the similar way, you can drag and drop files from any other file manager.

2.3.3 Remove files

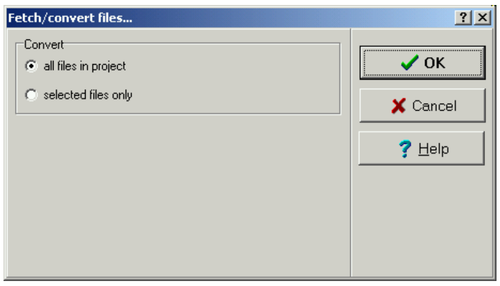
This function removes selected rows from the table of the source files (Input Files window). If the window is switched to *Slideshow* mode, it will remove currently displayed file only. The function removes the links from the file table only, it does not removes the source files physically. If you want to remove all files in the table, use rather *Clear files* function instead.

2.3.4 Fetch/Convert files

This function starts the reduction of CCD observation. It is always executed after the source files are inserted into the Input files table. The program read and check the source files, perform conversion to a working format (if needed) and store them into the temporary file folder. All the following functions and tools always process those temporary files.

Click on the *Fetch/convert files* in the *Files* menu. Simple dialog window opens with one option only – whether you want to fetch all files in the table or the selected ones only. Click on *OK* button to start the conversion process.

Note, that you also have to use this function at the beginning of conversion process, if the source files are already stored in FITS format.



2.3.5 Open file

The function is not intended for reduction process purpose. This additional function only assists to displaying the content of any file. The file must contain CCD frame, photometry file, catalogue file or text table in one of supported formats (see the section 5).

Select the *Open file* in the *Files* menu to open dialog window. Specify the directory and select the file, which you want to open. Click on *Open* to confirm the selection. The program automatically detects the file format and opens appropriate preview window. Use the *Close* button to close the preview window.

2.3.6 Exit

Click on *Exit* in *Files* menu to close the program.

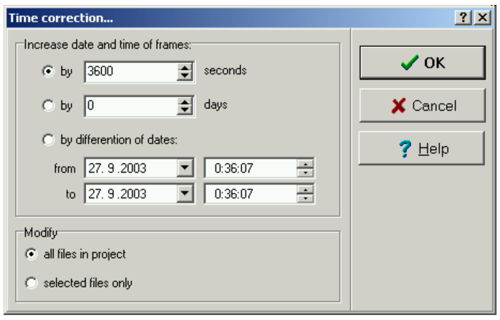
2.4 The Corrections menu

2.4.1 Time correction

The time correction modifies the date and time of observations by adding or subtracting a specified time interval. The correction can be used repeatedly on one file, since the function is cumulative, so the intervals are sumarised. For example, if you add 60 seconds and then add 40 seconds, the resulting observation time will be shifted by 100 seconds.

Click on the *Time correction* in the *Corrections* menu to open dialog window. Select the options to choose the way how to enter the time interval. You can enter either the exact value in seconds or days or enter interval by specifying two dates. In this case the program will compute the difference between the dates and modifies the observations times by appropriate number of seconds.

If you want to modify several source files only, select the files in the *Input files* table, and in the dialog check *Modify – selected files only* option. You can repeat this procedure to set proper times of observations for all source files.


Time correction...

Increase date and time of frames:
[✓ by] 3600 seconds
by 0 days
by differention of dates:
from 27. 9 .2003 0:36:07
to 27. 9 .2003 0:36:07

Modify
[✓ all files in project]
selected files only

2.4.2 Dark correction

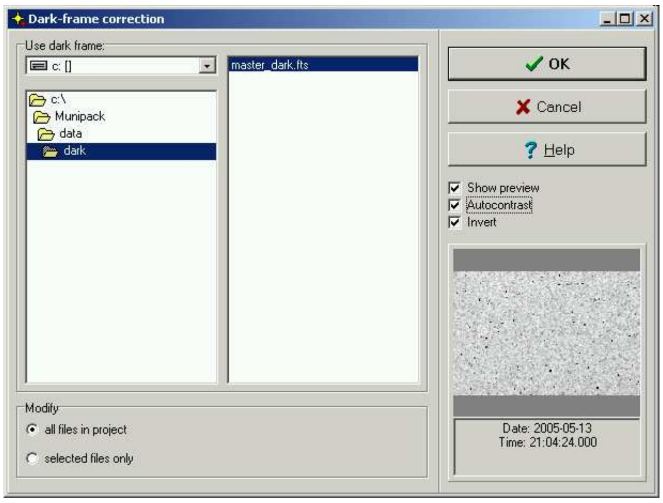
This function subtract the dark frame from source files. The dark frame is carried out by camera when the aperture is closed. Instead of using the automatic autodark function of your CCD camera, it is better to provide five to ten dark frames at the end of observation when the camera's chip is uniformly cooled. Then, use the *Master-dark* function to make one composed frame to be used for correction. This substantially supresses the noise. The correction frame shall be saved in one of supported image formats, the program automatically ensures the conversion if neccessary.

Click on the *Dark correction* in the *Correction* menu to open a dialog window. Select the path where the dark frame file is located. Press the *OK* button.

You always should use an individual dark frame for each observation series taken from several nights. Perform the correction step by step. In the table of input files, select the frames taken in one night, open correction dialog, check the *Modify – selected files only* option, select the corresponding dark frame and start the correction. Repeat the steps for each frame series.

Selecting file (containing the CCD image in supported format) from the list, an image preview with a brief information is shown at the left bottom of the dialog window. The preview features can be modified by selecting the check boxes:

- *Preview* – turns the preview display on and off. It's a good idea to turn off the preview on slow computers.
- *Autocontrast* – check on if you want the program to compute automatically the brightness and contrast of displayed image. Uncheck the box if you want to use the values stored in the file header.
- *Invert* – switches positive and negative display rendering.



Date: 2005-05-13
Time: 21:04:24.000

2.4.3 Flat correction

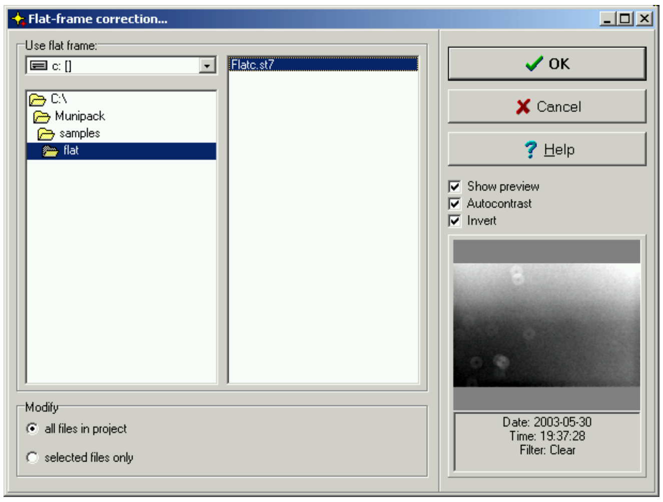
This correction always follows the dark frame correction. The flat frame is carried out when the camera is pointed to uniformly illuminated background. It eliminates variable gain of camera's chip CCD elements and also the optical system unbalanced sensitivity. To reduce the noise and increase the quality of photometry, you can combine a sequence of flat frames to obtain a *Master-flat* frame. The separate correction frame has to be used for each filter. Avoid eventual camera rotation on its mount. The correction frame must be saved in one of supported image formats, the program automatically ensures the conversion if neccessary.

Click on the *Flat correction* in the *Corrections* menu. The dialog windows appears. Find the directory and select the file with the flat frame. Press the *OK* button.

If you are procesing data carried out from several observations, then you will need an individual correction frame for each observation. In this case, perfomm the correction step by step. In the table of input files, select the frames, open correction dialog, check the *Modify – selected files only* only option, select the corresponding flat frame and start the correction. Repeat the steps for each series of frames.

Selecting file (containing the CCD image in supported format) from the list, an image preview with a brief information is shown at the left bottom of the dialog window. The preview features can be modified by selecting the check boxes:

- *Preview* – turns the preview display on and off. It's a good idea to turn off the preview on slow computers.
- *Autocontrast* – check on if you want the program to compute automatically the brightness and contrast of displayed image. Uncheck the box if you want to use the values stored in the file header.
- *Invert* – switches positive and negative display rendering.



Use flat frame:
c: []
C:\
Munipack
samples
flat
Flatc.st7

[✓] Show preview
[✓] Autocontrast
[✓] Invert

Date: 2003-05-30
Time: 19:37:28
Filter: Clear

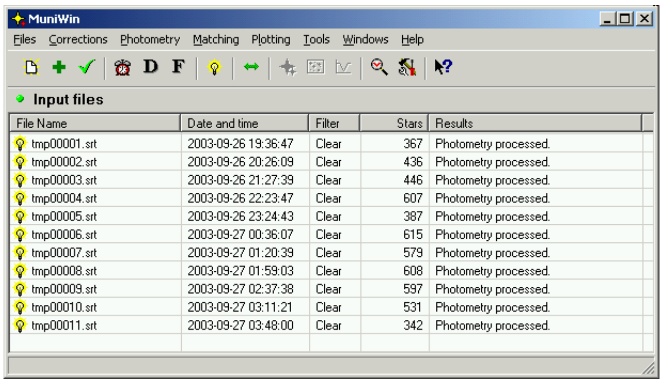
Modify
[✓ all files in project]
selected files only

2.5 The Photometry menu

2.5.1 Photometry

The function performs the aperture photometry of the input frames. All neccessary corrections have to be done prior the start. The results of photometry – photometry files – are stored in the folder for temporary files with `srt` extension. The file contains the data of position and brightness of each star on the source frame.

Select *Photometry* in the *Photometry* menu to open dialog window containing the option – whether to process all files in the table or the selected ones only. Click on the *OK* button to start the process. During processing, information messages are shown in the special dialog window. Finally the program actualizes the table of input files and inserts data to column *Stars* – the number of stars found on each input frame.


MuniWin
Files Corrections Photometry Matching Plotting Tools Windows Help
D F
Input files

File Name	Date and time	Filter	Stars	Results
tmp00001.srt	2003-09-26 19:36:47	Clear	367	Photometry processed.
tmp00002.srt	2003-09-26 20:26:09	Clear	436	Photometry processed.
tmp00003.srt	2003-09-26 21:27:39	Clear	446	Photometry processed.
tmp00004.srt	2003-09-26 22:23:47	Clear	607	Photometry processed.
tmp00005.srt	2003-09-26 23:24:43	Clear	387	Photometry processed.
tmp00006.srt	2003-09-27 00:36:07	Clear	615	Photometry processed.
tmp00007.srt	2003-09-27 01:20:39	Clear	579	Photometry processed.
tmp00008.srt	2003-09-27 01:59:03	Clear	608	Photometry processed.
tmp00009.srt	2003-09-27 02:37:38	Clear	597	Photometry processed.
tmp00010.srt	2003-09-27 03:11:21	Clear	531	Photometry processed.
tmp00011.srt	2003-09-27 03:48:00	Clear	342	Photometry processed.

If you want to apply the photometry only to a subset of input files (for example after the changing of photometry parameters), select the files in the *Input files* table, open the photometry dialog, check the *Modify – selected files only* option. Confirm by the *OK* button.

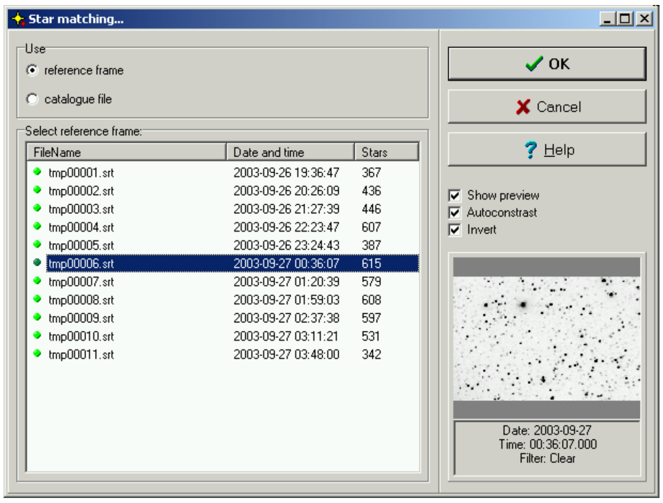
The photometry of the images, saved in the temporary folder, (they are not changed by photometry) can be reprocessed to achieve better results without any necessity to perform the conversion and correction of the frames again.

The program readilly allows to show the results of photometry: Select the file in the table and press right mouse button. In the pop-up menu choose *Open photometry* file. The new window with an image is displayed. White circles on the blue background represent stars in the photometry file. The diameter of circles matches the magnitude of found stars. The red-color circles correspond to stars which have been incorrectly detected (the relevant value of magnitude is greater than 99.0 mag).

2.6 The Matching menu

2.6.1 Match stars

The matching of stars finds the cross-references between the stars in the source file and the reference file. The result of the function is also photometry file, which contains the stars from the source file, but they are sorted in the order of the stars in the reference file. The reference file is in most cases one frame chosen from the serie of the input frames. The catalogue file can be also used instead – it is special photometry file, which contains the position of the stars and also the selection of variable and comparison stars.



Use: [✓ reference frame] catalogue file

Select reference frame:

FileName	Date and time	Stars
tmp00001.srt	2003-09-26 19:36:47	367
tmp00002.srt	2003-09-26 20:26:09	436
tmp00003.srt	2003-09-26 21:27:39	446
tmp00004.srt	2003-09-26 22:23:47	607
tmp00005.srt	2003-09-26 23:24:43	387
tmp00006.srt	2003-09-27 00:36:07	615
tmp00007.srt	2003-09-27 01:20:39	579
tmp00008.srt	2003-09-27 01:59:03	608
tmp00009.srt	2003-09-27 02:37:38	597
tmp00010.srt	2003-09-27 03:11:21	531
tmp00011.srt	2003-09-27 03:48:00	342

[✓] Show preview
[✓] Autoconstrast
[✓] Invert

Date: 2003-09-27
Time: 00:36:07.000
Filter: Clear

Open the *Matching* menu and select the *Match stars* item. In the dialog window, choose the reference file (select *reference file* option) or catalogue file (select *catalogue file* option). Confirm the dialog by the *OK* button. In most cases, the best reference file has greatest number of stars detected and in the preview and there are no defects visible. As well as in other dialogs, the preview of the selected file is displayed in the right bottom.

As soon as the process finishes the informations in the *Input files* window are updated. The data in the *Stars* column is changed, and they show the fraction of *x*/*y*, where *x* is the number of successfully matched stars and the letter *y* stands for the total number of stars in the source photometry file.

For the matching procedure does not affect the source photometry files, it can be repeated more times to get best results without need to repeat the photometry.

2.7 The Plotting menu

2.7.1 Choose stars

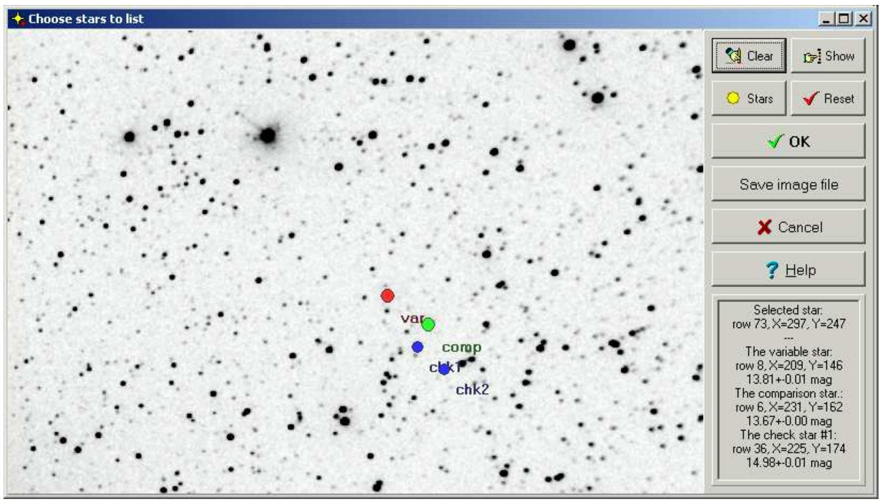
This function performs the final step of reduction process. It finalizes the selected variable star light curve related to the comparison and check stars. In the dialog, the user selects the variable, comparison and/or check stars on the reference frame.

Open the *Plotting* menu and select the *Choose stars* item. The maximized dialog window appears showing the reference frame or the chart from catalogue files respectively. Sequentially, click on the variable star (small red circle mark with the label *var*), then click on comparison star (green color circle with *comp* label) and optionally you can select several check stars (*chk1*, *chk2*, etc.). If the catalogue file with the matching stars is used, the stars will be automatically selected – user checks the selection only. If the output format is set to *instrumental magnitudes*, the selection is similar, but the stars are labeled *s1*, *s2*, ...

If you want to enter new selection, click on the *Clear* button and select the stars again. The *Reset* button resets back the selection to the state stored in the catalogue file. If you want to find a star by its number (position in the file), press the *Show* button and enter the index of the star. Press and hold the right mouse button to hide the labels on the chart.

Confirm the selection by *OK* button. The table of brightness is computed and the light curve graph is displayed in the separate dialog window (see *Plot Curve* section).

The *Stars* button turns on and off the small circles, which mark up the positions of all stars from the reference file. The yellow and red cirles indicates the good and the invalid values of magnitude respectively.



Selected star:
row 73, X=297, Y=247

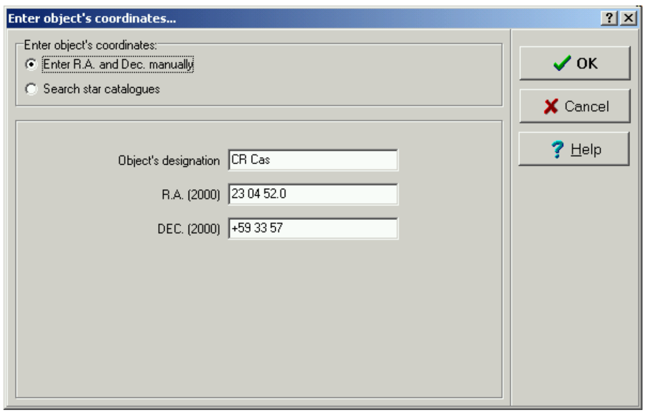
The variable star:
row 8, X=209, Y=146
13.81+-0.01 mag
The comparison star.:
row 6, X=231, Y=162
13.67+-0.00 mag
The check star #1:
row 36, X=225, Y=174
14.98+-0.01 mag

2.7.2 Enter object's coordinates

This item is shown in the menu only if the object's coordinates are needed in the reduction process (computation of heliocentric correction or air-mass coefficient). The position can be entered by two ways: manually or selecting from the catalogue. The first method is applicable, when you know the equatorial coordinates of the observed object. The second method is helpful to variable stars observers, because the program can seek through a number of popular catalogues of variable stars and find records by star's designation or its part. Because the catalogues are not included in the binary installation, it is neccessary to set up the path to the files in the *Preferencies* dialog (*Catalogues* page).

Manual entering of coordinates

In the *Plotting* menu, click on the *Enter object's coordinates* item. New dialog window is open. In its upper section, select *Enter R.A. and Dec. manually* option and enter the right ascension and the declination to appripriate edit fields. Examples: *23 57 23.5, +43 49 15; 235723.5, +43 49 15* or *23.95653, 43.8208*. You can also fill the object's designation, this value will be used on printed charts and graphs. Close the dialog by *OK* button.

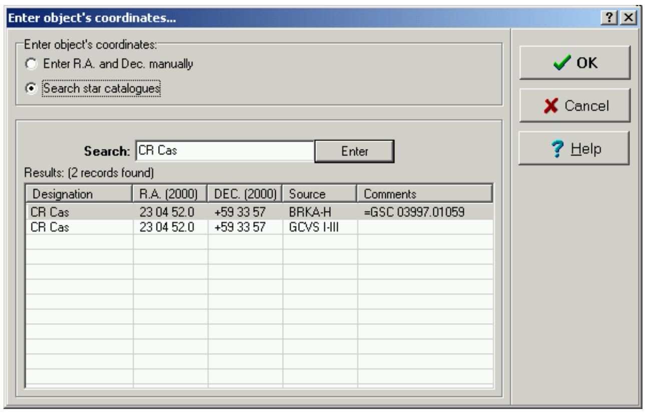


Enter object's coordinates:
[✓ Enter R.A. and Dec. manually] Search star catalogues

Object's designation	CR Cas
R.A. (2000)	23 04 52.0
DEC. (2000)	+59 33 57

Reading coordinates from the catalogues

In the *Plotting* menu, click on the *Enter object's coordinates* item. New dialog window is open. Unlike the previous case, select *Search star catalogues* option and fill the variable star's designation or its part to the *Search* field. You had to enter at least three characters, upper and lower case is ignored as well as extra or missing spaces and other common delimiters. Press *Enter* button to start the searching.



Enter object's coordinates:
Enter R.A. and Dec. manually [✓ Search star catalogues]

Search: CR Cas

Results: (2 records found)

Designation	R.A. (2000)	DEC. (2000)	Source	Comments
CR Cas	23 04 52.0	+59 33 57	BRKA-H	=GSC 03997.01059
CR Cas	23 04 52.0	+59 33 57	GCVS I-III	

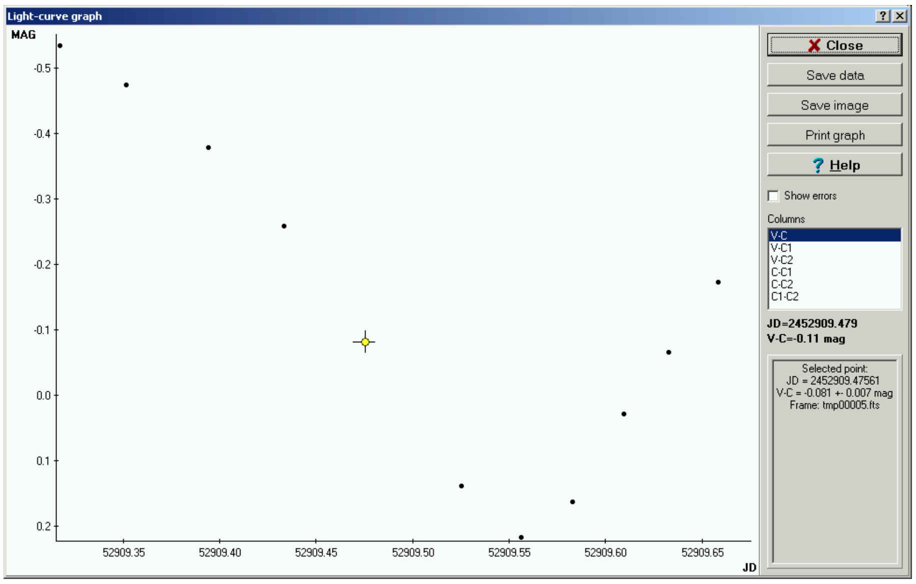
The results are presented in the table, which contains all matched records. In the first column, the precise designation is displayed and followed by the right ascension and declination, name of catalogue. Additional information are shown in the last column, in most cases it contains other possible designations of the variable star.

Go through the table and click on the row which you regard as best. It is possible also to change the search string and repeat the searching. Confirm the selection by *OK* button.

2.7.3 Plot curve

The data file contains the reduction process output. Select the *Plot Curve* item in the *Plotting* menu in the to open a dialog window, where a graph of the last generated light curve is displayed. Using the control buttons on the panel, the data can be saved in text or image format or printed on the printer.

The ouput data file contains not only the brightness of the variable star with respect to the comparison star and all check stars, but also the relative brightness of the comparison star with respect to check stars and all other combinations. These values are used to check, whether the comparison star is not variable, too. A table with a list of available data in file occurrs in the panel bottom part, the first column is displayed by default. To view another light curve select the appropriate row in the table. The *Show errors* option allows to switch on / off displaying the error lines.



```
JD=2452909.479
V-C=-0.11 mag

Selected point:
JD = 2452909.47561
V-C = -0.081 +- 0.007 mag
Frame: tmp00005.fts
```

The final processed files containing data of light curve shall be saved properly either to source files folder or to user specified one. In the dialog window, press *Save data* button and specify a directory and filename, confirm by the *Save* button. The output files are saved into the results subdirectory in the application directory, by default. The path can be changed in the preferencies menu. This folder is accessible directly from the user interface (see the section 2.9.2).

In the light curve graph it is possible to display further information – click on the curve point by left mouse button. The selected point is marked by yellow circle and the detailed information are displayed at the panel right bottom part.

To print the graph on the printer, click on the *Print graph* to open a preview window. The *Print* button causes to open a standard print dialog. Confirm by the *OK* button.

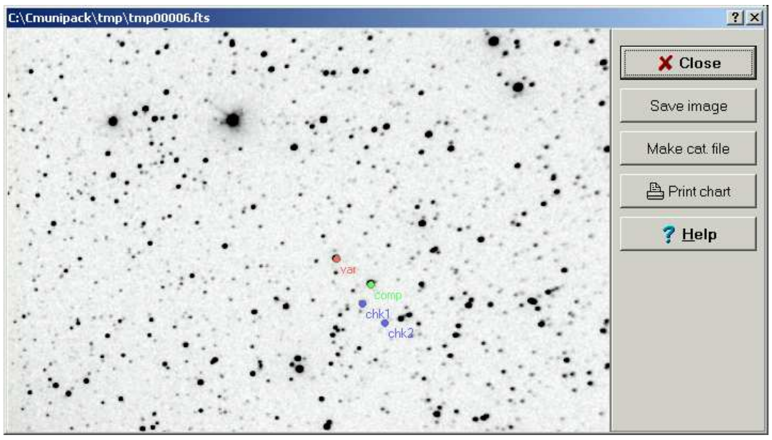
You can also save the graph as JPEG, PNG, TIFF or BMP image to the file by selecting the *Save image* button.

2.7.4 View chart

This dialog window shows the chart of last used selection of the stars. On the window background, the reference frame or the catalogue file is displayed, and the stars are coloured and labeled by the same way as in the *Choose stars* dialog. You can save the chart to JPEG, PNG, TIFF or BMP image or print on the printer. Also the new catalogue file can be created from this dialog.

In the *Plotting* menu select the *View chart* to open dialog window. If you want to save the chart into the image file, click on the *Save image* button and in the standardised save dialog specify a directory and enter the file name. Confirm by the *Save* button. The charts are saved in the same directory as ouput data files (see the section 2.9.2), by default. This files are accessible in the *Results* dialog.

If you want to print the chart on the printer, press the *Print chart* button. The preview window appears. In the window, press *Print* button, which opens a standard print dialog. Confirm by the *OK* button.



2.8 The Tools menu

2.8.1 Master-dark

The *Master-dark* tool compose a set of dark frames and performs one dark frame called *master-dark*. Applying this function, you can achieve a high quality correction frame to reduce the noise. It is recommened to provide dark frames on the end of observation, when the camera is enough uniformly cooled. The source frames must be saved in one of supported image formats, the program automatically ensures the conversion if neccessary. The output file is always in FITS format.

Add the source dark frames to the table of input files and proceed their conversion. Open the *Tools* menu and click on the *Master-dark* item. In the dialog window specify whether all files shall be processed or only the selected ones. Confirm by the *OK* button. In the following dialog select a directory and enter the name of output file to be saved. Press the *Save* button.

2.8.2 Master-flat

The *Master-flat* tool reads a set of flat fields, normalizes them to the set up level of brightness (10000 by default). Then, applying the robust median algorith to corresponding pixels on source frames, the one flat field called *master-flat* is composed. With help of this function, you can achieve high quality correction frame to reduce the noise. The separate correction frame has to be used for each filter, if the color fliters were used. Avoid eventual camera rotation on its mount. The source frames must be saved in one of supported image formats, the program automatically ensures the conversion if neccessary. The output file is always in FITS format.

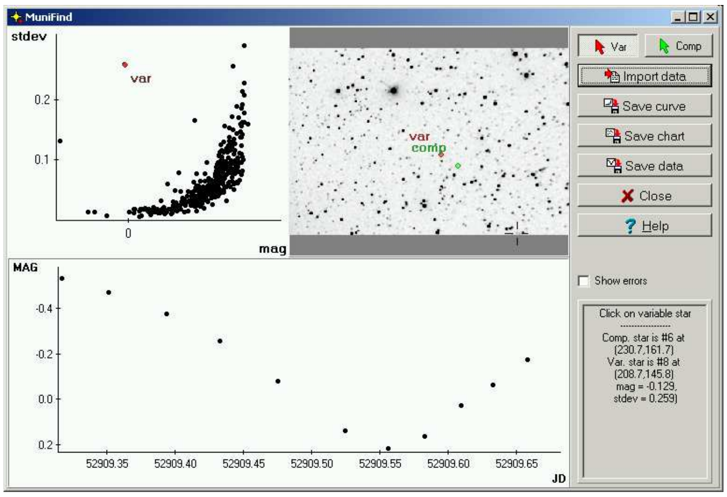
Add the source flat frames to the table of input files and proceed their conversion, apply the *Dark correction* also. Open the *Tools* menu and click on the *Master-flat*. In the dialog window specify whether all files shall be processed or only the selected ones. Confirm by the *OK* button. In the following dialog select a directory and enter the name of output file to be saved. Press the *Save* button.

2.8.3 Find variables

The CCD observation allows to measure brightness of all stars occuring in the field of the telescope. It may happen, that on a series of frames that appart from observed star there occures also another variable star. By the following steps, you can perhaps discover a new variable star.

The *Find variables* dialog is the useful tool that provides semi-automatic scaning of variable stars in the series of source files. It is based on the relation between the standard deviations of the brightness of the stars and their mean brightness. The program reads all photometry files and compute the mean brightness and standard deviations of brightness of all stars. The algorithm automatically removes stars that are missing on majority of source frames. For the star of lower magnitude the deviation of brightness exhibits higher value than deviation for stars with a higher brightness.

First follow all steps of observed frames reduction through processing the photometry files. In the *Tools* menu select the *Find variables*. The program computes all stars brightness data at all frames and selects one comparison star. The dialog window opens. At top left area a graph of mag vs. stdev is shown. On the right part the identification chart with variable star marked up is presented. Also the selected star light-curve graph is shown.



```
Click on variable star
----------------
Comp. star is #6 at
(230.7,161.7)
Var. star is #8 at
(208.7,145.8)
mag = -0.129,
stdev = 0.259)
```

Click at any point of the graph left. The program marks the point with red circle with the *var* label. The same way, the corresponding star is marked up on the identification chart. In the bottom part of the dialog window, the light-curve graph of the star is plotted. Sequentially indicate all stars, you suspect to be a variable star.

The comparison star is marked with green circle with the *comp* label. Although the comparison star is automatically selected, the selection does not need be ideal. To change the comparison star manually, click on *Comp* button on the control panel and select the star on the identification chart. When done, click on the *Var* button to switch back to variable selection mode.

By the help of the buttons on the panel, the graph, chart or the light-curve can be saved as data or image file or can be sent to the printer.

2.8.4 Plot track list

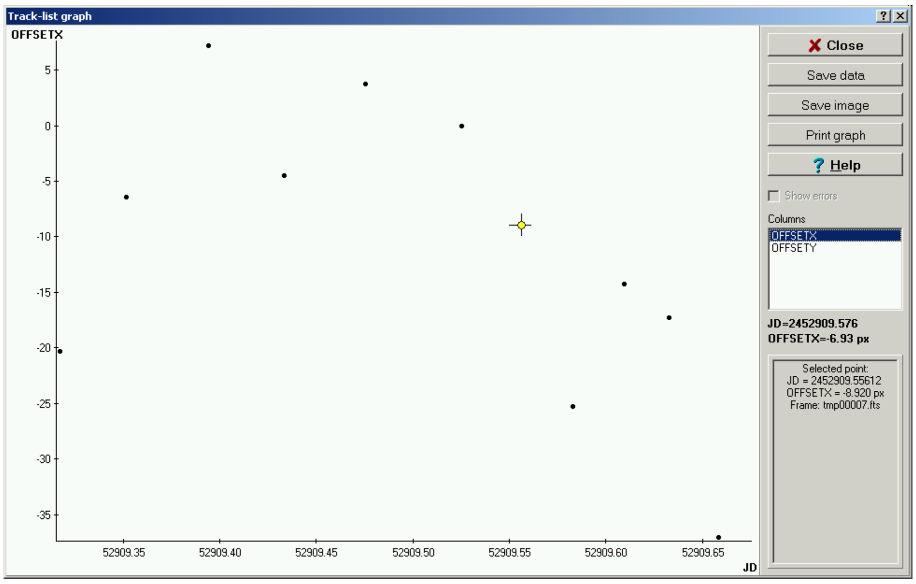
The track-list graph shows the relation between the stars shift on the frames against their positions on the reference file. It may be used for computing the telescope mounting periodic error.

In the *Tools* menu, click on *Plot track list* to open dialog window. The julian date is on the horizontal axis of the graph and the offsets in pixels are on the vertical axis. The list of available data in a file is dispalyed in the panel middle part. The data file consists two contextual columns:

- OFFSETX – offset in horizontal direction, positive values mean the shift to the right, negative to the left.
- OFFSETY – offset in vertical direction, positive values mean the shift to the bottom, negative to the top.

In the dialog, press the *Save data* and specify a directory, where the file is to be saved. Confirm by the *Save* button. The output files are saved into the `results` subdirectory in the application directory by default, but the path can be changed in the preferencies. This folder is directly accessible in the user interface (see the section 2.9.2).

In the light curve graph there is possible to display further information – click on the curve point by left mouse button. The selected point is marked by yellow circle and the detailed information is shown at the panel right bottom part.



JD=2452909.576
OFFSETX=-6.93 px

Selected point:
JD = 2452909.55612
OFFSETX = -8.920 px
Frame: tmp00007.fts

To print the graph on the printer, click on the *Print graph* to open a preview window. Print button causes to open standard print dialog. Confirm by the *OK* button.

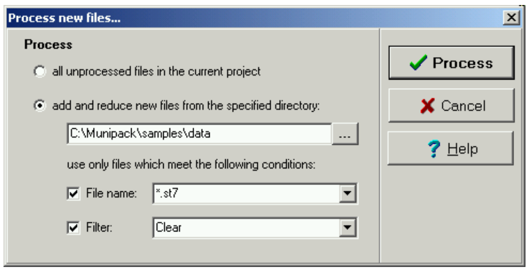
You can also save the graph as JPEG, PNG, TIFF or BMP image to the file selecting the *Save image* button.

2.8.5 Process new files

The tool is designated for short-period variables observation. Using this tool, it is possible to trace the variable behavior during the observation and eventually decide, assisted by the continous light curve, whether to resume or stop the variable scaning.

During the first run, process the CCD frames by the standard procedure up to the selection of the stars and produce the light curve. At this moment you can exit the Muniwin program (it always saves the last state). After acquiring a several further frames, start the Muniwin program again, open the *Tools* menu and select the *Process new files* item. New dialog window is open. In the dialog, select the second option, and enter the path where the new frames are stored during the observation. In case you are observing a number of variables or you are using several filters, enter the file name mask and/or filter name to the corresponding fields.

Confirm the dialog by the *Process* button. The program detects the new files which meets specified conditions, inserts them to the table of input files and performs the necessary conversion, corrections, photometry, matching and finally plots actualized light curve graph. Similarly you can continue up to the end of observation.



all unprocessed files in the current project

[✓ add and reduce new files from the specified directory:]

C:\Munipack\samples\data

use only files which meet the following conditions:

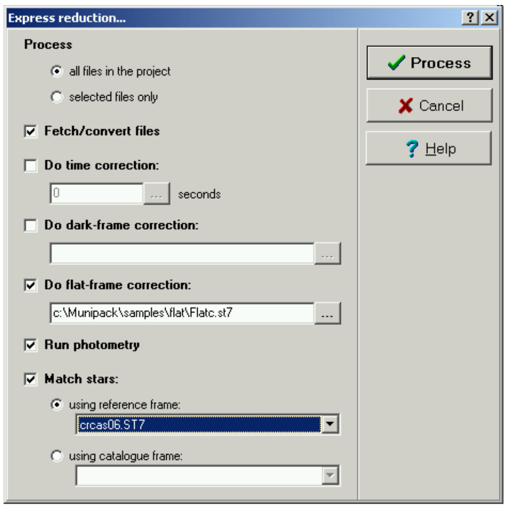
[✓ File name:] *.st7

[✓ Filter:] Clear

2.8.6 Express reduction

This tool offers the simple way how to reduce CCD observation in the batch mode. Using the standard way, the user is to procede step by step manually. This dialog allows you to enter all necessary data in one dialog and then the program performs all required steps in succession.

First, enter the source files into the input table by standard way using *Add files* dialog. Then, open the *Tools* menu and select the *Express reduction* item. In the dialog window, check the options to specify whether you want to process all files or the selected ones. If you wish to convert files to temporary folder, check *Fetch/convert files*. In order to perform the frames time correction, check *Do time correction* and enter the time period (in seconds) to the editing field. By similar way specify all the steps of reduction process you want to proceed along. Click on the *Process* button to start the reduction process.



Process
- (●) all files in the project
- ( ) selected files only

[✓] Fetch/convert files

[ ] Do time correction: 0 seconds

[ ] Do dark-frame correction:

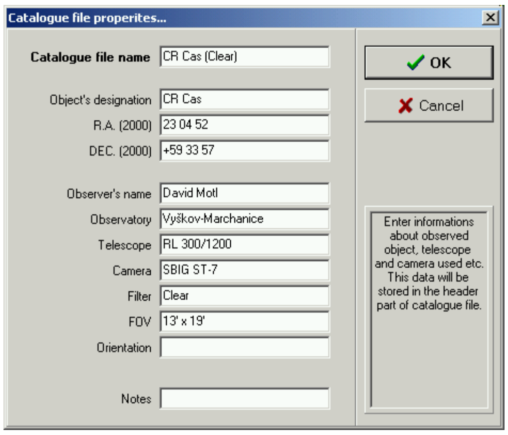
[✓] Do flat-frame correction: c:\Munipack\samples\flat\Flatc.st7

[✓] Run photometry

[✓] Match stars:
- (●) using reference frame: crcas06.ST7
- ( ) using catalogue frame:

2.8.7 Make catalogue file

In the *Tools* menu, select the *Make catalogue file* item. New dialog is open Enter the basic data of the observed object to editing fields. If you share your catalogues with other observers, complete the information about observer and instrument you used. It is only required to enter the star name into the field. Confirm by the *OK* button. The catalogue file and the reference frame is saved into catalogue folder. It is the `cat` subdirectory in the application directory, by default, but the path can be changed in the preferences.



Catalogue file name	CR Cas (Clear)
Object's designation	CR Cas
R.A. (2000)	23 04 52
DEC. (2000)	+59 33 57
Observer's name	David Motl
Observatory	Vyškov-Marchanice
Telescope	RL 300/1200
Camera	SBIG ST-7
Filter	Clear
FOV	13' x 19'
Orientation	
Notes	

Enter informations about observed object, telescope and camera used etc. This data will be stored in the header part of catalogue file.

2.8.8 Preferences

The *Preferences* item in the *Tools* menu opens a configuration dialog used to modify the properties of the user interface related to parameters of the reduction process functions. The options integrated in one functional block are assigned to an individual tab. On each tab, the *Set defaults* button is present to set the options in the current tab to default values. In the following text, the default values are printed out in the [square brackets].

Backuping the configuration

When you are going to do some experiments especially with photometry parameters, it is advisable to back up your settings before. You can achieve this simply by making copy of all files from the application directory with `ini` extension, though much simpler way is offered directly in the *Preferencies* dialog.

To backup your configuration, click on the *Backup* button. The standard dialog for saving files are open. Move to the directory, where the backup copy shall be placed to and close the dialog by *Save* button.

Similarly you can restore the configuration back. In the *Preferencies* dialog, press the *Restore* button. The standard dialog for opening files are open. Move to the directory, where the backup files have been placed and close the dialog by the *Open* button.

General options

Allows to modify the Muniwin application configuration and the user interface setting (*Tools / Preferences*).

- *Save process logs to file* – switches on and off the message logging stored during the computation process. The files have the log extension and are located in temporary folder (see below). [off]
- *Debug mode (verbose process logs)* – this option switches on and off debug mode of processing outputs. In this mode, which is intended mainly for debugging purposes, all print-outs are very extensive. [off]
- *Do not save image files to catalogue folder* – check this option to supress saving of the image files into the catalogue folder. [off]
- *Directory with temp. files* – the path to the folder, where the temporary files are stored. This is the `tmp` subdirectory in the application directory by default.
- *Directory with cat. files* – the path to the folder, where the catalogue files are stored. This is *cat* subdirectory in the application directory by default.
- *Directory with res. files* – the path to the folder for the output files. This is `results` subdirectory in the application directory by default.
- *Electronic help language* – to select language of the help.

Photometry options

The parameters of the photometry process can be set in *Tools / Preferences / Photometry options*. Among many parameters of the aperture photometry procedure a special attention should be payed to the following: *FWHM* and the *Threshold*. These parameters control mainly the detection of stars on the CCD frames. *FWHM* is the abbreviation of the term "Full Width at Half Maximum" – inspect the stars on the frame to determine the proper value. By decreasing the *FWHM* value the fainter stars will be found. It is to be pointed out, that too small value may take the artefacts on the background as regular stars. The *Threshold* value sets the distance between the finest stars detected and the background sky noise. The value is entered in "sigma" units.

- *Read noise* – readout noise in electrons, you should enter the proper value stated in the camera's documentation.
- *Gain* – number of electrons per ADU, you should enter the proper value stated in the camera's documentation.
- *Low good datum* – lowest good datum of pixel in sigmas [7.00]
- *High good datum* – highest goot datum of pixel in ADU [65535.00 for 16bit cameras]
- *FWHM of object* – expected full width at half maximum of the stars, for first tries use the default value [3.00]
- *Threshold* – the finest stars detection threshold eliminating out the artefacts, in sigmas, for first tries use the default value [4.00]
- *Low sharpness cutoff* – low sharpness cutoff [0.20]
- *High sharpness cutoff* – high sharpness cutoff [1.00]
- *Low roundness cutoff* – low roundness cutoff [-1.00]
- *High roundness cutoff* – high roudness cutoff [1.00]

Aperture options

On this tab (Tools/Preferences) the configuration window enables setting the apertures used in aperture photometry. The aperture index used in output data computation is defined in the image dots (pixels).

- *Aperture #1..#12* – aperture radii #1 to #12
- *Inner sky radius* – inner radius of sky aperture [20.0]
- *Outer sky radius* – outer radius of sky aperture [30.0]

Matching options

The *Tools/Preferences/Match stars* enables the parameters configuration used in frames matching. In most cases, it is not necessary to change the default values.

- *No. of stars used in matching* – number of stars used from the source photometry files. Input data for the transformation matrix evaluation. [10]
- *No. of identification stars* – number of the polygons apexes used in the matching. [5]
- *Sigma clipping factor* – the inaccuracy threshold of polygons angles. [2.5]

Plotting options

In this section the functions for light curve plotting and setting the output data can be configured.

- *Use aperture no.* – index of the aperture used (see the section 2.8.8) [1]
- *Output format* – format of output data files [differential magnitudes]
- *Date & time format* – format of date and time of observation [geocentric julian date]
- *Include heliocentric correction* – check this option to include the value of heliocentric correction to output files. [unchecked]

- *Include air-mass coefficient* – check this option to include the value of air-mass coefficient to output files. [unchecked]
- *Observatory* – name of the observatory (optional). [empty]
- *Longitude* – longitude of the observatory. Examples: *E 16 02 45*, *+160245* or *+16.0458*. Positive values correspond to locations to the east and negative values are regarded as locations to the west of zero meridian. [empty]
- *Latitude* – latitude of the observatory. Examples: *N 49 13 00*, *+491300* or *+49.2167*. Positive values correspond to locations to the north and negative values are regarded as locations to the south of equator. [empty]

Note 1: Instrumental magnitudes are intended for further processing only and allows experienced users perform their own post-processing algorithms, they are not absolute magnitudes and so they do not correspond to the values listed in stellar catalogues.

Note 2: Observatory name is always optional. The coordinates are required only if the air-mass coefficient shall be computed. This feature is disabled by default.

Tools

This section is intended for configuration of auxiliary utilities, which are accessible in the *Tools* menu. The meaning of the parameters are discused in the description of the tools (see chapter 2.8).

Catalogues

This section of the *Preferencies* dialog is intended for observers of variable stars. It is neccessary to fill the data only if you want to search the object's coordinates during the reduction process in the catalogues of variable stars.

The catalogues are not included in the installation package, it is possible to download them from the internet (see the *References* chapter). The program needs to know, where the appropriate files are located.

2.9 The Windows menu

2.9.1 Show message log

To view the messages log performed by last reduction step select the *Show message log* in the *Windows* menu. In the window context menu it is possible to copy the text to clipboard.

If you want to use the log printout for further processing, you can set automatic saving into the files in the temporary file folder. Files have `log` extension. In the *Tools* menu, open the *Preferences* dialog, select the *General* tab and check the *Save process logs to file* option.

2.9.2 Show results

This window shows the content of output files directory (e.g. data files, charts). They are automatically saved into the `results` subdirectory in the application directory by default, but the path can changed in the preferencies. In the table, double-click to open a file prieview window. Use right mouse button to open the context functions menu.

When the reduction process is finished, it is recommended to move the files from the `Results` folder to other location so to avoid overwritting them by the next process. The program provides three different ways: copy the selected files into a directory using the *Save files* dialog, move the selected ones using *Drag-and-drop* function or copy files via *clipboard*.

The Save files dialog

In the output files table, select the files you want to save. Then press right mouse button to open the context menu. Select *Save files*. The standard save dialog appears. Choose a directory, where you want to copy the files, and confirm by the *OK* button.

The Drag & Drop function

Open Windows explorer (This computer) and choose the folder, where the files shall be saved into. In the Muniwin program, open *Results* window and select the files. Press and hold left mouse button and drag the files into the folder icon on system bar. Wait till the window switches to foreground, drag the files to the window area and then release the mouse button.

By similar way the drag and drop method can be used in other file managers.

The Copy & Paste function

In the Muniwin program open the *Results* window. Select the files to copy and press the right mouse button. In the context menu, select *Copy to clipboard*. Open Windows explorer (This computer) and choose the folder, where the files shall be saved into. Press the right mouse button again and select the *Paste* in the context menu.

By similar way the copy and paste method can be used in other file managers.

2.9.3 Show catalogue files

This window displays the content of catalogue folder (*Windows* menu). This is the `cat` subdirectory in the application directory by default, but you can change the path in the preferences (*Directory with cat. files* on the *General* tab).

Double-click on an item to open a chart in full resolution. The preview shows the chart or the image (if present), the variable and the comparison stars are marked and labeled.

2.10 The Help menu

2.10.1 Open help

Click on *Help/Open help* to open the electronic help. By pressing *F1* button everywhere in the program the context help can be opened, too. The language of the help can be selected in the *Preferences* dialog.

2.10.2 About Muniwin

Opens the dialog with the information about the program version and origin of the program.

3 Command-line interface

This section decribes the using of the programs of the *C-Munipack* package, which are invocated from a command-line interface. There allows to execute the reduction process from the command line or shell, user application, batch or script.

In the text, all keywords are printed in `teletype style`, the phrases, which shall be replaced by a parameter value (number, file name, etc.), are marked up by *cursive font*. Default values of parameters are places in the [square brackets].

3.1 Generic parameters and syntax rules

The `mask` parameters allows to set the naming of output files. The mask consists of letters, numbers and other characters allowed in the file name by the operating system. The special meaning has the question mark or the sequence of it. They will be replaced by the ordinal number of the output file, the number is always indented by leading zeros to the same number of decimal places as the number of the question marks. For example, when the mask `dout????.fts` is given, the output files named `dout0001.fts, dout0002.fts, dout0003.fts` will be generated.

All described programs supports this generic parameters:

`-h, --help`	prints short help
`-l, --license`	prints the license
`-v, --version`	prints the program name and its version
`-q, --quiet`	quiet mode – inhibits the screen printouts
`--debug`	debug mode – extends the screen printouts

The list of input files is usualy entered directly on the command line. If you are going to process a large set of files, you can use the question marks (for a single chars) or the asterisk (for a strings of chars).

The alternate way, how to process a large set of files, is the batch mode. In this mode, the list of input files is read from a given file or from the standard input. The directory file, for example named `dirfile.txt`, must have the text format and the names of the input files are written on the separate lines. If the files are placed not in the current working directory, you have to specify the proper path in full or shortened form. Such file is created by the following command:
`dir /b *.st7 > dirfile.txt`.

3.2 Reduction pipeline

3.2.1 Konve (CCD image conversion)

The Konve program converts CCD frames from the format used by camera controling software to FITS format. The correction of time of observation or image flipping can be applied also.

The program can be run in several modes, the mode is selected by command-line parameters. Besides the usual conversion mode, it allows printing header informations of source files in short or detailed form. You can convert header only while not copying the image. Additional informations, which shall be written to the output files can be given in parameter file.

Current version of program supports reading SBIG (ST-xx) compressed and uncompressed files and the FITS files also. Note, that you can use FITS to FITS conversion for gathering files from many locations into one directory. It will solve the file name collisions.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`par=`*file*	optional parameter file [none]
`mask=`*mask*	mask of output file(s) [kout????.fts]
`tcor=`*sec*	time correction in seconds [0]
`flip=`*xy*	image flipping (values: 'x', 'y', 'xy')
image1 ...	names of source files (no wildcards *,?)
`-p`*file*	optional parameter file (equivalent with 'par=')
`-o`*mask*	mask of output file(s) (equivalent with 'out=')
`-c`*x*	initial counter value [1]
`-flip`*xy*	image flipping (equivalent with 'flip=')
`-header`	converts headers only, doesn't convert image data
`-info`	prints short header informations only
`-details`	prints full header informations only

Configuration files: None

Example: The following command converts files 'crcas01.st7', 'crcas02.st7' and 'crcas03.st7' and writes the output to files 'kout0001.fts', 'kout0002.fts' and 'kout0003.fts'.

```
C:\cmunipack\bin>konve crcas01.st7 crcas02.st7 crcas03.st7
KONVE Version 1.0.10

--------------------------------------------
(crcas01.st7) -> (kout0001.fts)
File format: ST-7 Compressed Image
Image size : 382 x 255 pixels
Date & time: 2003-09-26 19:36:47 UT
Exposure   : 20.00 s
Filter     : Clear
--------------------------------------------
(crcas02.st7) -> (kout0002.fts)
File format: ST-7 Compressed Image
Image size : 382 x 255 pixels
Date & time: 2003-09-26 20:00:24 UT
Exposure   : 20.00 s
Filter     : Clear
--------------------------------------------
(crcas03.st7) -> (kout0003.fts)
File format: ST-7 Compressed Image
Image size : 382 x 255 pixels
Date & time: 2003-09-26 20:26:09 UT
Exposure   : 20.00 s
Filter     : Clear

C:\Munipack\bin>_
```

3.2.2 Timebat (Time correction)

The Timebat program applies time correction to the source frames. It means, that it adds given amount of time to the times of observation. The program does not touch neither the image data nor other parameters in header.

The source frames and also dark frame must be in FITS format. The output file is in FITS format too.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`tcor=`*sec*	time correction in seconds
`mask=`*mask*	mask of output file(s) [tout????.fts]

image1 ...	names of source files (no wildcards *,?)
`-t`*sec*	time correction in seconds (equivalent with 'tcor=')
`-o`*mask*	mask of output file(s) (equivalent with 'mask=')
`-c`*x*	counter initial value [1]

Configuration files: None

Example: The following command shifts time of observation of 'kout0001.fts' by 1 hour to the future. The output shall be written to 'tout0001.fts' file.

```
C:\cmunipack\bin>timebat tcor=3600 kout0001.fts
Time correction: 3600 s
(kout0001.fts) => (tout0001.fits)

C:\Munipack\bin>_
```

3.2.3 Darkbat (Dark-frame correction)

The Darkbat program applies dark-frame correction to a set of source frames. It means, that it subtracts the dark frame from source files and the results writes to the output file.

The source frames and also dark frame must be in FITS format and of same dimensions. Frames of the same exposition duration should be used. The output file is in FITS format too.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`dark=`*file*	name of dark-frame file
`mask=`*mask*	mask of output file(s) [dark????.fts]
image1 ...	names of source files (no wildcards *,?)
`-d`*file*	name of dark-frame file (equivalent with 'dark=')
`-o`*mask*	mask of output file(s) (equivalent with 'mask=')
`-c`*x*	counter initial value [1]

Configuration files: None

Example: The following command applies dark-frame correction of 'kout0001.fts' using 'dark.fts' as dark frame. The output is written to 'dout0001.fts'.

```
C:\Munipack\bin>darkbat dark=dark.fts dout0001.fts
Using dark frame: dark.fts
(kout0001.fts) => (dout0001.fts)

C:\Munipack\bin>_
```

3.2.4 Flatbat (flat-frame correction)

The Flatbat program applies flat-field correction to a set of source frames. It means, that it divides the source frames by the flat frame pixel-by-pixel and the result is multiplied by median value of the flat frame. The resulting image is written to output file.

The source frames and also flat frame must be in FITS format and of same dimensions. Frames of the same exposition duration and color filter should be used. The output file is in FITS format too.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`flat=`*file*	name of flat-frame file
`mask=`*mask*	mask of output file(s) [flat????.fts]
image1 ...	names of source files (no wildcards *,?)
`-f`*file*	name of flat-frame file (equivalent with 'flat=')
`-o`*mask*	mask of output file(s) (equivalent with 'mask=')
`-c`*x*	counter initial value [1]

Configuration files: None

Example: The following command applies flat-frame correction to 'dout0001.fts' using 'flat.fts' as flat frame. The output is writted to 'fout0001.fts'.

```
C:\cmunipack\bin>flatbat flat=flat.fts fout0001.fts
Using flat frame: flat.fts
(dout0001.fts) => (fout0001.fts)

C:\Munipack\bin>_
```

3.2.5 Muniphot (Photometry)

The Muniphot program applies photometry to a single frame or a set of frames. The output is written in text format to a file. (Such files are refered as photometry files in this paper.) Current version version uses aperture photometry algorithm only.

The source file must be in FITS format. The output files are in Daophot-compatible text format. It does not generate `.coo` and `.ap` files in contrast to Muniphot of original Munipack. The output files have the same name as the corresponding source files, but extension is changed to `srt`.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`par=`*file*	name of parameter file [muniphot.ini]
`ap=`*file*	name of aperture file [apertures.ini]
image1 ...	names of source files (no wildcards *,?)

Configuration files: `Muniphot.ini`, `Apertures.ini`

Example: The following command computes photometry of 'fout0001.fts' and stores the output to 'fout0001.srt' file.

```
C:\cmunipack\bin>muniphot tmp0001.fts
------------------------------------------
Image file   : tmp0001.fts
Date & time  : 2003-09-26 19:36:47 UT
Picture size : 382 x 255 pixels
Sky value    : 310.7 +- 25.7 ADU
Mean & median: 310.7, 310.7 ADU
Rel. error   : 1.07
Stars found  : 354
Mag. limit   : 18.1 +- 0.2 per star in aperture #1.

C:\cmunipack\bin>type tmp0001.srt
 NL   NX   NY  LOWBAD HIGHBAD  THRESH     AP1  PH/ADU  RNOISE...
  2  382  255   314.7 65000.0  110.15    2.00    2.30   15.00...
```

```
     0  380.349   26.646   11.673   99.999   99.999   99.999...
      466.192 22.23  0.00   0.002    9.999    9.999    9.999...

     1   47.193   24.184   12.447   12.299   12.229   12.199...
      462.042 17.61  0.00   0.002    0.002    0.002    0.002...

     2  317.126  232.323   12.588   12.408   12.317   12.279...
      459.838 20.55  0.00   0.003    0.002    0.002    0.003...

C:\Munipack\bin>_
```

3.2.6 Munimatch (Matching files)

The Munimatch finds corresponding stars in two photometry files. One file is refered as reference file, the second one is called source file. The output of the matching process is the photometry file, which the stars from source file is written in, but their order is changed, so corresponding stars are on the same indices in output and reference files. Instead of a reference file, which is usually one frame from a sequence being processed, a catalogue file in XML format can be used.

The source and output files have to be in photometry file format. The reference file should be in photometry or catalogue file format, optionally. If a set of files is processed, then the reference file is common for all sources.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`ref=`*file*	name of reference or catalogue file
`par=`*file*	name of parameter file [munimatch.ini]
image1 ...	names of source files (no wildcards *,?)
`-p`*file*	optional parameter file (equivalent with 'par=')

Configuration files: `Munimatch.ini`

Example: The following command matches 'tmp0001.srt' as source file and 'ref.srt' as reference file and the output writes to 'tmp0001.mat'.

```
C:\cmunipack\bin>munimatch ref=ref.srt tmp0001.srt
Reference file: ref.srt
-------------------------------
(tmp0001.srt) -> (tmp0001.mat)
Number of matched stars          : 268 from 354 (76\%)
Sum of square in the better case : 0.1979
Tolerance                        : 0.50
Transformation matrix :
            1.000          0.001          15.407
           -0.001          1.000         -11.320
            0.000          0.000           1.000

C:\Munipack\bin>_
```

3.2.7 Munilist (Making light-curve)

The Munilist program reads photometry files and creates the table of magnitudes of selected stars in the dependence on a time. The table is written to a output file in text format. The format of the table depends on given parameters and on the number of selected stars. The Munilist is usually the last step of reduction process.

The list of stars is given on command line, the stars are identified by index number according to their order in the file (not by the indices in the first column in the photometry file!), first star in the file has index 1, empty lines are included.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`src=`*file*	the program will read the source data from a single file
var ...	indices of variable star, comparison stars and check stars
`par=`*file*	name of parameter file [munilist.ini]
`out=`*file*	name of output file [data.dat]
`ap=`*n*	aperture index [1]
image1 ...	names of source files (no wildcards *,?)
`-os`	use the 'differential magnitudes' output format (default)
`-of`	use the 'instrumental magnitudes' output format
`-oa`	use the 'read-all' output format
`-ot`	use the 'track-list' output format
`-p`*file*	optional parameter file (equivalent with 'par=')

Configuration files: `munilist.ini`

Example: The following command creates table of brightness of the star #2 (stored on second position in photometry files) relative to the star #3. The output in short format is required; the magnitudes will be differential instrumental magnitues (V-C).

```
C:\cmunipack\bin>munilist 2 3 test1.mat test2.mat test3.mat
test1.mat
test2.mat
test3.mat

C:\cmunipack\bin>type data.dat
JD V-C s1
Aperture: 2.00, Filter: Clear
2452909.31733 -0.134 0.004
2452909.33373 -0.147 0.004
2452909.35161 -0.164 0.004

C:\Munipack\bin>_
```

3.3 Other programs

3.3.1 Autoflat (Flat-frame composition)

The Autoflat program compose a set of flat-fields and performs one flat-field called master-flat. The source frames are normalized to brigtness defined level (10000 by default). Then, it computes the robust mean algorithm to the corresponding pixels to get the resulting value. Applying this function, you can achieve high quality correction frame to reduce the noise. The separate correction frame has to be used for each filter, if the color fliters were used. Avoid eventual camera rotation on its mount.

All source frames must be in FITS format and of same dimensions. Frames of the same color filter and exposition duration should be used. The output file is in FITS format too.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`par=`*file*	name of parameter file [autoflat.ini]
`out=`*file*	name of output file [autoflat.fts]
image1 ...	names of source files (no wildcards *,?)
`-o`*file*	name of output file (equivalent with 'out=')
`-p`*file*	optional parameter file (equivalent with 'par=')

Configuration files: autoflat.ini

Example: The following command computes median of frames: 'test1.fts', 'test2.fts' a 'test3.fts'; than the resulting frame normalizes to level of 10000 and stores the output frame fo 'autoflat.fts'.

```
C:\Munipack\bin>autoflat test1.fts test2.fts test3.fts
Image file   : test1.fts
Median       : 459.0565
Skysig       : 18.4670
-------------------------------
Image file   : test2.fts
Median       : 475.4877
Skysig       : 18.6898
-------------------------------
Image file   : test3.fts
Median       : 496.4252
Skysig       : 18.0923
-------------------------------
Output file  : autoflat.fts
Final median : 10350.3
Final skysig : 9.8

C:\Munipack\bin>_
```

3.3.2 Meandark (Dark-frame composition)

The Meandark program compose a set of dark-fields and performs one dark-field called master-dark. It applies the robust mean algorithm to the corresponding pixels to get the resulting value. Applying this function, you can achieve high quality correction frame to reduce the noise.

All source frames must be in FITS format and of same dimensions. Frames of the same exposition duration should be used. The output file is in FITS format too.

Parameters:

@*dirfile*	name of text file with list of source files
@	the program will read the list of source files from stdin
out=*file*	name of output file, [meandark.fts]
image1 ...	names of source files (no wildcards *,?)
-o*file*	name of output file (equivalent with 'out=')

Configuration files: None

Example: The following command computes mean of frames: 'test1.fts', 'test2.fts' a 'test3.fts' and stores the output frame fo 'meandark.fts'.

```
C:\Munipack\bin>meandark test1.fts test2.fts test3.fts
MEANDARK Version 1.0.10

-------------------------------
Image file   : test1.fts
-------------------------------
Image file   : test2.fts
-------------------------------
Image file   : test3.fts
-------------------------------
Output file  : meandark.fts

C:\Munipack\bin>_
```

3.3.3 Helcor (Heliocentric correction)

The *Helcor* program converts geocentric julian date (jdgeo) to heliocentric date (jdhel) or back. It works either in batch mode, where the table stored in file is converted and saved to another file, or in terminal mode, where the user enters simple commands and queries by keyboard.

When running the batch mode, the program reads the source file line by line, and expects the JD value in the first column, which must be divided at least one of common used dividers (semicolon, comma, space, tab char, ...). The JD value can be in full (2453xxx.x) or short (53xxx.x) form. Decimal places must be separated by point, not comma. The lines, which doesn't consist of valid julian date, are copied into output without any change. In other cases the heliocentric correction is computed, its value is added to or subtracted from read julian date and the old JD value is replaced by the new one in the same form (full or short form, number of decimal places). The rest of the line remains unchanged.

In terminal mode, the opening informations is displayed and the program prompts the user to enter object's coordinates. First, specify right ascension (R.A.) a confirm by the `Enter` key. By the same way, the declination (DEC.) is set up. The value is printed out as response to the command. After setting up coordinates, entering of a julian date is expected (`JDgeo>` or `JDhel>`). The user enters the julian date (confirms by the `Enter` key) and the program computes the value of heliocentric correction and prints out to console three values: the geocentric and the heliocentric julian date and value of h.c. in days. Other commands are descibed below. Entering the 'q' command exits the program.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`mask=`*mask*	mask of output file(s) [hkor????.dat]
file1 ...	names of source files (no wildcards *,?)
`ra=`*hhmmss*	object's right ascension (hhmmss or hhmm)
`dec=`*+ddmmss*	object's declination (ddmmss, ddmm, -ddmmss or -ddmm)
`-o`*mask*	name of output file(s) (equivalent with 'mask=')
`-c`*x*	counter initial value [1]
`-r`	reverse mode (jdhel → jdgeo)

Configuration files: None

Terminal mode commands:

`r`	Sets reverse mode (jdhel → jdgeo)
`n`	Sets normal mode (jdgeo → jdhel)
`ra` *hhmmss*	Sets right ascension
`dec` *+ddmmss*	Sets declination
`h` nebo `?`	Prints short help
`empty line`	Prints current coordinates
q	Exits program

Example 1: The program converts geocentric julian dates in data.dat file to heliocentric dates and the resulting table stores in helioc.dat file. Object's coordinates are R.A. = $22^{h}00^{m}$, DEC = $+58^\circ 10'$.

```
C:\cmunipack\bin>helcor mask=helioc.dat ra=2200 dec=5810 data.dat
HELCOR Version 1.1.7

data.dat -> helioc.dat

C:\cmunipack\bin>type data.dat
JD V-C s1 V-C1 s2 V-C2 s3 C-C1 s4 C-C2 s5 C1-C2 s6
```

```
Aperture: 2.00, Filter: Clear
2452909.31733 -0.134 0.004 -0.212 0.004 ...
2452909.33373 -0.147 0.004 -0.220 0.004 ...
2452909.35161 -0.164 0.004 -0.236 0.004 ...

C:\cmunipack\bin>type helioc.dat
JD V-C s1 V-C1 s2 V-C2 s3 C-C1 s4 C-C2 s5 C1-C2 s6
Aperture: 2.00, Filter: Clear
2452909.31478 -0.134 0.004 -0.212 0.004 ...
2452909.33118 -0.147 0.004 -0.220 0.004 ...
2452909.34906 -0.164 0.004 -0.236 0.004 ...

C:\Munipack\bin>_
```

Example 2: Convert GJD = 2452909.31733 to heliocentric date and back using terminal mode. Object's coordinates are R.A. = $22^h00^m$and DEC. = $+58°10'$.

```
C:\cmunipack\bin>helcor
helcor (C-Munipack) 1.1.8

This is terminal mode. Enter 'h' command for help or 'q' for exit.

Object's coordinates have not been set yet.
R.A.:2200[Enter]
R.A. = 22 00 00
DEC.:+5810[Enter]
DEC. = +58 10 00
JDgeo:2452909.31733[Enter]
-----------------------------------------------
JD (geoc.)  : 2452909.31733
Date & time : 2003-09-26 19:36:57.312 UT
Hel. corr.  : 0.00267 d = 3 min 50.909 s
JD (hel.)   : 2452909.32000
Date & time : 2003-09-26 19:40:48.221 UT
-----------------------------------------------
JDhel:q[Enter]

Exiting terminal mode.

C:\Munipack\bin>_
```

3.3.4 Munifind (Finding variable stars)

The *Munifind* program is the useful tool used in semi-automatic scaning of variable stars in the series of source files. It is based on the relation between the standard deviations of the brightness of the stars and their mean brightness. The program reads all photometry files and compute the mean brightness and standard deviations of brightness of all stars. The algorithm automatically removes stars that are missing on majority of source frames (threshold parameter). The comparison star is automaticaly selected or given by a optional parameter.

The output file is saved in text format and contains the table of mean magnitudes and standard deviations.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`src=`*file*	the program will read the source data from a single file
`ref=`*n*	index of reference file [autodetection]
`par=`*file*	optional parameter file [munifind.ini]
`out=`*file*	name of output file [munifind.dat]
`ap=`*file*	aperture index [1]
`thr=`*x*	threshold in precents [60]
file1 ...	names of source files (no wildcards *,?)
`-o`*file*	name of output file (equivalent with 'out=')
`-a`*file*	name of source file (equivalent with 'src=')
`-p`*file*	optional parameter file (equivalent with 'par=')

Configuration files: `munifind.ini`

Example: The following command demonstrates, how to create a file of magnitudes and the deviations from the source files `test1.mat`, `test2.mat` and `test3.mat`.

```
C:\Munipack\bin>munifind test1.mat test2.mat test3.mat
test1.mat
test2.mat
test3.mat
---------------------------------------------
Number of analyzed stars:           588 (+ reference star #1)
Number of source files:            3
Total number of data points:        1013
Reference star used:                1
Reference star errors:              0 (0%)
Number of good points required:     2
Number of stars written to output:  347 (59%)
Number of points ruled out:         0 (0%)
---------------------------------------------
Output file:                       munifind.dat

C:\Munipack\bin>_
```

3.3.5 Airmass (Computing airmass coefficient)

The *Airmass* program computes value of the airmass coefficient from given julian date, object's coordinates and observer's coordinates. It works either in batch mode, where the table stored in file is converted and saved to another file, or in terminal mode, where the user enters simple commands and queries by keyboard.

When running the batch mode, the program reads the source file line by line, and expects the JD value in the first column, which must be divided at least one of common used dividers (semicolon, comma, space, tab char, ...). The JD value can be in full (2453xxx.x) or in short (53xxx.x) form. Decimal places must be separated by points, not by commas. The lines, which doesn't correspond to a table header or doesn't consist of valid julian date, are copied into output without any change. In other cases the airmass coefficient is computed, its value is added to the end of the line. The AIRMASS field is added to the header line of the table.

In terminal mode, the opening informations is displayed and the program prompts the user to enter the right ascension of the object (R.A.) a excepts confirmation by the `Enter` key. By the same way, the declination (DEC.) is set up. The value is printed out as response to the command. Then, by the same way again, the observer's geographical longitude (LON.) and latitude (LAT.) is entered. Use positive values of latitude for locations to the north of equator. Positive values of longitude means, that the observation place is located to the east of zero meridian. After setting up all coordinates, entering of a julian date is expected (JD). The user enters the julian date and the program computes and prints out the values of azimuth, altitude and value of airmass coefficient. All commands are descibed below. Entering the *q* command exits the program.

Parameters:

`@`*dirfile*	name of text file with list of source files
`@`	the program will read the list of source files from stdin
`mask=`*mask*	mask of output file(s) [amass????.dat]
file1 ...	names of source files (no wildcards *,?)
`ra=`*hhmmss*	object's right ascension
`dec=`+*ddmmss*	object's declination
`lon=`+*dddmmss*	observer's longitude
`lat=`+*dddmmss*	observer's latitude
`-o`*mask*	name of output file(s) (equivalent with 'mask=')
`-c`*x*	counter initial value [1]
`-r`	reverse mode (jdhel → jdgeo)

Configuration files: None

Terminal mode commands:

`ra` *hhmmss*	Sets right ascension
`dec` +*ddmmss*	Sets declination
`lon` +*dddmmss*	observer's longitude
`lat` +*dddmmss*	observer's latitude
`h` nebo `?`	Prints short help
`empty line`	Prints current coordinates
q	Exits program

Example 1: The program adds values of airmass coefficient to the table stored in *data.dat* file and the resulting table stores in *helioc.dat* file. The object's coordinates are α = $18^h29^m32^s$, δ = $+22°34'24''$, the observer's coordinates are λ = 16°40′= 16.6667° to the east of the zero meridian and ϕ = 49°13′= 49.2167° to the north of the equator.

```
C:\cmunipack\bin>airmass lon=16.6667 lat=49.2167 ra=182932 dec=223424 data.dat
airmass (C-Munipack) 1.1.7

data.dat -> amass0001.dat

C:\cmunipack\bin>type data.dat
JD V-C s1 V-C1 s2 V-C2 s3 C-C1 s4 C-C2 s5 C1-C2 s6
Aperture: 1, Filter: I, JD: geocentric
2453868.39368 -1.164 0.017 -1.750 0.025 ...
2453868.39484 -1.191 0.018 -1.706 0.025 ...
2453868.39598 -1.138 0.017 -1.736 0.026 ...

C:\cmunipack\bin>type amass0001.dat
JD V-C s1 V-C1 s2 V-C2 s3 C-C1 s4 C-C2 s5 C1-C2 s6 AIRMASS
Aperture: 1, Filter: I, JD: geocentric
2453868.39368 -1.164 0.017 -1.750 0.025 ... 1.949
2453868.39484 -1.191 0.018 -1.706 0.025 ... 1.933
2453868.39598 -1.138 0.017 -1.736 0.026 ... 1.918

C:\Munipack\bin>_
```

Example 2: Compute value of airmass coefficient for V1011 Her (α = $18^h29^m32^s$, δ = $+22°34'24''$). The observation was made in Brno, Czech republic (λ = 16°40′= 16.6667° to the east of the zero meridian and ϕ = 49°13′= 49.2167° to the north of the equator). Julian date of observation is JD = 2453868.39368 UT.

```
C:\cmunipack\bin>airmass
airmass (C-Munipack) 1.1.7

This is terminal mode. Enter 'h' command for help or 'q' for exit.

Object's coordinates have not been set yet.
R.A.:182932[Enter]
R.A. = 18 29 32
DEC.:223424[Enter]
DEC. = +22 34 24
LON.:16.6667[Enter]
LON. = E 16 40 00
LAT.:49.2167[Enter]
LAT. = N 49 13 00
JD:2453868.39368[Enter]
-----------------------------------------------
Julian date: 2453868.3937
Date & time: 2006-05-12 21:26:53.952 UT
Az. & alt. : 270 22 45 (E), 30 47 18
Airmass (X): 1.949
-----------------------------------------------
JD:q[Enter]

Exiting terminal mode.

C:\Munipack\bin>_
```

4 C-Munipack library

The *C-Munipack* library provides high-level interface for operating on CCD data. It allows to use all functions from the C-Munipack project in user's application. It is used by the Muniwin program and it is also intended for developers, who are going to start their own CCD data processing application.

On Windows operating systems, the binary code is linked to the *cmunipack.dll* file. This file must be present in the same directory as the application's executable or at least on the place, where the system can find it. The CFitsio library is dynamically linked to this library, so the *cfitsio.dll* file is also required. The Expat library is linked statically by default and thus no binaries is needed.

On Linux-type operating system, the binary code is linked to *libcmunipack.<version>.so* automatically during execution of the *make* command and all necessary files are copied to proper folders by the *make install* command. The CFitsio and Expat libraries are also required.

Definitions of constants, data types and public functions are located in the public header file named *cmunipack.h* included in the source code package.

4.1 Auxiliary functions

`void` **cmpack_getversionid**`(char *buf, int buflen);`

The function stores the version identifier to the specified memory buffer. The *buf* parameter is the pointer to the allocated memory, the *buflen* parameter is the size of the buffer in bytes.

`void` **cmpack_formaterror**`(char *buf, int buflen, int code);`

The function translates the error code to its string representation. The *buf* parameter is pointer to allocated memory, the *buflen* parameter is the size of the buffer in bytes. The *code* parameter specifies the error code. This function supports only C-Munipack's error codes (greater than 1000). For translating FITSIO codes (less than 1000), the CFitsio library provides its own translating function.

`int` **cmpack_setoutput**`(cmpack_cbtype *cbproc, int level);`

The functions sets the message output procedure and verbosity level. All messages are passed out to the callback function, which must have one parameter of *const char ** type and returns nothing. The function MUST NOT free the buffer.

The *cbproc* parameter specifies the pointer to the callback function, if it is set to NULL, all messages will be discarded. This is default behaviour. The second parameter, *level*, modifies the amount of messages produced, see *MPK_LEVEL_xxx* constants.

The function returns previous value of the level.

4.1.1 Coordinate transformations

`int` **cmpack_strtora**`(const char *buf, double *ra);`

The function translates a string representation of right ascension to a real number in angle hours. The *buf* parameter is pointer to the null-terminated input string, the second parameter points to the variable, where the output value shall be stored to. If the transformation has been successful, it returns zero. Otherwise, the *MPK_INVALID_PAR* code is returned.

Supported formats: hh mm ss.s, hhmmss.s or hh.hhhhh

`int` **cmpack_strtodec**`(const char *buf, double *dec);`

The function translates a string representation of declination to a real number in degrees. The *buf* parameter is pointer to the null-terminated input string, the second parameter points to the variable, where the output value shall be stored to. If the transformation has been successful, it returns zero. Otherwise, the *MPK_INVALID_PAR* code is returned.

Supported formats: -ddd mm ss.s, -dddmmss.s or -dd.dddd

`void` **cmpack_ratostr**`(double ra, char *buf);`

The function translates right ascension in hours to its string representation. The first parameter specifies the input value. The *buf* parameter is pointer to the allocated memory (at least 9 bytes), where the null-terminated output string shall be stored to. The function has no return value.

Supported format: hh mm ss

`void` **cmpack_dectostr**`(double dec, char *buf);`

The function translates declination in degrees to its string representation. The first parameter specifies the input value. The *buf* parameter is pointer to the allocated memory (at least 11 bytes), where the null-terminated output string shall be stored to. The function has no return value.

Supported format: +ddd mm ss

`int` **cmpack_strtolat**`(const char *buf, double *lat);`

The function translates a string representation of latitude to a real number in degrees. The *buf* parameter is pointer to the null-terminated input string, the second parameter points to the variable, where the output value shall be stored to. If the transformation has been successful, it returns zero. Otherwise, the *MPK_INVALID_PAR* code is returned.

Supported formats: N dd mm ss, Nddmmss, +dd mm ss, +ddmmss, +dd.dddd, S dd mm ss, Sddmmss, -dd mm ss, -ddmmss or -dd.dddd (Positive values and 'N' prefix indicate locations on the north hemisphere, negative values and 'S' prefix indicate locations on the south hemisphere.)

`int` **cmpack_strtolon**`(const char *buf, double *lon);`

The function translates a string representation of longitude to a real number in degrees. The *buf* parameter is pointer to the null-terminated input string, the second parameter points to the variable, where the output value shall be stored to. If the transformation has been successful, it returns zero. Otherwise, the *MPK_INVALID_PAR* code is returned.

Supported formats: E ddd mm ss, Edddmmss, +ddd mm ss, +dddmmss, +ddd.dddd, W ddd mm ss, Wdddmmss, -ddd mm ss, -dddmmss or -ddd.dddd (Positive values and 'E' prefix indicate locations on the eastern hemisphere, negative values and 'W' prefix indicate locations on the western hemisphere.)

`void` **cmpack_lattostr**`(double lat, char *buf);`

The function translates latitude in degrees to its string representation. The first parameter specifies the input value. The *buf* parameter is pointer to the allocated memory (at least 11 bytes), where the null-terminated output string shall be stored to. The function has no return value.

Supported format: D dd mm ss (where D is the 'N' or 'S' prefix)

`void` **cmpack_lontostr**`(double lon, char *buf);`

The function translates longitude in degrees to its string representation. The first parameter specifies the input value. The *buf* parameter is pointer to the allocated memory (at least 12 bytes), where the null-terminated output string shall be stored to. The function has no return value.

Supported format: D ddd mm ss (where D is the 'E' or 'W' prefix)

`void` **cmpack_aztostr**`(double az, char *buf);`

The function translates azimuth in degrees to its string representation. The first parameter specifies the input value. The *buf* parameter is pointer to the allocated memory (at least 16 bytes), where the null-terminated output string shall be stored to. The function has no return value.

Supported format: ddd mm ss (XXX) (where XXX is replaced by direction code: 'S', 'SSW', 'SW', 'WSW', 'W', ... 'N') `void` **cmpack_alttostr**`(double alt, char *buf);`

The function translates altitude in degrees to its string representation. The first parameter specifies the input value. The *buf* parameter is pointer to the allocated memory (at least 10 bytes), where the null-terminated output string shall be stored to. The function has no return value.

Supported format: -dd mm ss

4.1.2 Date and time transformations

`void` **cmpack_strtodate**`(const char *datestr, int *year, int *month, int *day);`

The function parses a string representation of date and returns a date in form of three integer numbers representing year, month and day. The *buf* parameter is pointer to the null-terminated input string, the next three parameters are pointers to the variables, where the output values shall be stored to. If the transformation is successful, it returns zero. Otherwise, the *MPK_INVALID_PAR* code is returned.

Supported format: yyyy-mm-dd

`void` **cmpack_strtotime**`(const char *timestr, int *hour, int *minute, int *second, int *milisecond);`

The function parses a string representation of time and returns a time in form of four integer numbers representing hours, minutes, seconds and miliseconds. The *buf* parameter is pointer to the null-terminated input string, the next four parameters are pointers to the variables, where the output values shall be stored to. If the transformation is successful, it returns zero. Otherwise, the *MPK_INVALID_PAR* code is returned.

Supported formats: hh:mm:ss, hh:mm:ss.ssss

`double` **cmpack_strtojd**`(const char *datestr, const char *timestr);`

The function parses a string representations of date and time and returns a full Julian date in form of double precision real number. The *datestr* and the *timestr* parameters are pointers to the null-terminated input strings with date and time respectively. If the transformation is successful, it returns Julian date. Otherwise, the zero value is returned.

Supported formats: see *cmpack_strtotime* and *cmpack_strtodate*.

`double` **cmpack_encodejd**`(int year, int month, int day, int hour, int minute, int second, int milisecond);`

The function converts a numeric representation of date and time to a Julian date. The parameters specify the input values. If the transformation is successful, it returns Julian date. Otherwise, the zero value is returned.

`void` **cmpack_datetostr**`(int year, int month, int day, char *datestr);`

The function converts a numeric representation of date to its string representation. The *datestr* parameter is pointer to the allocated memory (at least 11 bytes), where the null-terminated output string shall be stored to. The function returns no value.

Supported format: yyyy-mm-dd

`void` **cmpack_timetostr**`(int hour, int minute, int second, int msec, char *timestr);`

The function converts a numeric representation of time to its string representation. The *timestr* parameter is pointer to the allocated memory (at least 13 bytes), where the null-terminated output string shall be stored to. The function has no return value.

Supported format: hh:mm:ss.ssss

`void` **cmpack_decodejd**`(double jd, int *year, int *month, int *day, int *hour, int *minute, int *second, int *milisecond);`

The function converts Julian date to numeric representation of a corresponding date and time. The *jd* parameter specifies the Julian date, the next seven parameters are pointers to the variables, where the output values shall be stored to. The function has no return value.

`void` **cmpack_jdtostr**`(double jd, char *datestr, char *timestr);`

The function converts a Julian date to string representation of corresponding date and time. The *jd* parameter specifies the Julian date, the next two parameters are pointers to the allocated buffers (at least 11 and 13 bytes), where the null-terminated output strings shall be stored to. The function has no return value.

Supported formats: see *cmpack_timetostr* and *cmpack_datetostr*.

4.1.3 Mathematical functions

`void` **cmpack_robustmean**`(int n, float *A, float *mean, float *sig);`

The function computes robust mean value of *n* single precision real numbers stored in the *A* array. The variable pointed by the *mean* parameter receives computed mean value, while the standard deviation is passed out by the *sig* parameter. The function has no return value.

`void` **cmpack_fastsky**`(int nx, int ny, float *ccd, int nmax, float lobad, float hibad, float *skymed, float *skysig);`

The function computes rough estimation of value of sky brightness by means of the robust mean algorithm. This routine is optimized to very fast estimation of frame background, because it applies the robust mean algorithm to a subset of pixels of a source frame.

The *nx* and *ny* parameters are dimensions of the frame in pixels. Frame data is given in the *ccd* parameter in the form of a array of single precision real numbers. The *nmax* parameter specifies the upper limit for number of pixels used in computation. There are also two values, which are intended to discard bad pixels: all pixels, which value is less than or equal to the *lobad* parameter or greater than or equal to the *hibad* parameter are not included in the computation. The variables specified in the *skymed* and *skysing* parameters receive values of mean value and standard deviation respectively. The function has no return value.

`void` **cmpack_fullsky**`(int nx, int ny, float *ccd, float *skymed, float *skysig);`

The function computes a value of sky brightness by means of the robust mean algorithm. This routine is optimized to the precision, because, in countrast to the previous routine, it uses all pixels of a source frame.

The *nx* and *ny* parameters are dimensions of the frame in pixels. Frame data is given in the *ccd* parameter in the form of a array of single precision real numbers. The variables specified in the *skymed* and *skysing* parameters receive values of mean value and standard deviation respectively. The function has no return value.

4.1.4 Astronomical functions

`double` **cmpack_siderealtime**`(double jd);`

The function computes value of Greenwich Mean Sidereal Time. The *jd* parameter specifies the date and time in for of Julian date, corresponding value of sidereal time in **days** is returned.

`double` **cmpack_airmass**`(double jd, double obj_ra, double obj_dec, double obs_lon, double obs_lat);`

The function computes value of air-mass coefficient. The *jd* parameter specifies the date and time in form of Julian date. The *obj_ra* parameter is right ascension of an observed object in hours, *obj_dec* parameter is declination of an object in degrees. Next two parameters specify observer's geographical coordinates in degrees. The function returns output value, which is positive if the object is above the horizon and negative, if the object is below the horizon.

`void` **cmpack_rdtolb**`(double obj_ra, double obj_de, double *obj_la, double *obj_be);`

The function transforms coordinates from the equatorial system to the ecliptical one. The *obj_ra* parameter is right ascension of an observed object in hours, *obj_dec* parameter is declination of an object in degrees. The next two parameters are pointers to the variables, which receive corresponding ecliptical coordinates in degrees. The function has no return value.

`void` **cmpack_sun**`(double jd, double *sun_ls, double *sun_rs);`

The function computes celestial position of the Sun and distance between the Sun and the Earth. The *jd* parameter specifies the date and time in form of Julian date. The *sun_ls* parameter receives the ecliptic longitude of the Sun and the corresponding distance between the Earth and the Sun in astronomical units (AU) is passed out by the *sun_rs* parameter.

`double` **cmpack_hcor**`(double obj_la, double obj_be, double sun_ls, double sun_rs);`

The function computes value of heliocentric correction. The input parameters specify the ecliptic coordinates of the observed object in degrees (*obj_la* and *obj_be*) and the ecliptic longitude and the distance of the Sun in degrees and astronomical units respectively (*sub_ls* and *sun_rs*). The function returns output value in days.

4.2 CCD frame reduction

4.2.1 Frame conversion

The *konv_xxx* functions are intended for reading CCD frames in all formats supported by C-Munipack software and converting them to the FITS format. These files are designated to form a unified interface for fetching CCD data to memory before displaying them on the screen.

`int` **konv_format**`(const char *filename, int *format);`

The function opens a file specified by full path and name in the *filename* parameter and checks its format. Then, the file is closed. If the file format is supported, it returns zero and the format identifier (see table 4.3) is stored to the *format* parameter. If failed, the error code is returned.

`int` **konv_init**`(const char *parfile, int flipx, int flipy, long tcor, int mode);`

This function parses the configuration file and initializes the internal variables for conversion of CCD frames to the FITS format. The function allocates several internal memory buffers, so it's necessary to call *konv_clean* when the processed is finished.

If the *pafile* parameter isn't NULL, it should specify the path and name of the configuration file. This file contains the parameters, which shall be inserted to the converted files, and their respective values. See the Files section for more details. The *flipx* and *flipy* parameters turns on image flipping in horizontal and vertical direction respectively. The *tcor* parameter is the time correction applied to the time of observation in seconds. The last parameter, *mode*, sets the conversion mode – see table 4.4. If the initialization is successful, it returns zero. Otherwise, the error code is returned.

`int` **konv**`(const char *in, const char *out);`

This function performs conversion of the source file specified by path and name in the *in* parameter. The output data (if they are generated) are stored to the file specified by *out* parameter. If the output data are not generated in this mode, the value of the second parameter can be NULL. It returns zero on success or error code on failure. You must call *konv_init* routine first to initialize the internal variables and set the conversion mode.

`int` **konv_clean**`(void);`

This function frees the memory allocated by calling the *konv_init* routine.

Here is the recommended order of calling the above mentioned functions. Names of the input and the output files are stored in the *in* and *out* arrays respectively.

```
konv_init(NULL, 0, 0, 0, 0, NULL);
for (i=0;i<count;i++)
  konv(in[i],out[i],NULL);
konv_clean(NULL);
```

`int` **konv_open**`(void **handle, const char *filename);`

This function opens a file with CCD frame of any supported format and creates a descriptor, which is used for accessing file data. You must call the *konv_close* file to close the file and to free allocated memory. The file can be stored in one of the supported CCD frame formats, the automatic detection algorithm is used to determine the format of the file.

The *filename* parameter specify the path and name of the file. The *handle* parameters is the pointer to the variable of *void** type, which receives the pointer to the internal descriptor.

`int` **konv_openf**`(void **handle, const char *filename, int format);`

This function opens a file with CCD frame of specified type and creates a descriptor, which is used for accessing file data. You must call the *konv_close* file to close the file and to free allocated memory. This function doesn't perform format detection.

The *filename* parameter specify the path and file name of the file. The *handle* parameters is the pointer to the variable of *void** type, which receives the pointer to the internal descriptor. The *format* parameter is one the *MPK_FORMAT_xxx* constants – see table 4.3.

`int` **konv_gets**`(void *handle, const char *key, char *buf, int buflen);`

This function reads single string data from the file header.

The *handle* parameter is the handle created by *konv_open** function. The *key* parameter specify which parameter shall be read. The value is stored to the memory buffer, which pointer is given in the *buf* parameter, and the length of the buffer in bytes is in the *buflen* parameter. The function returns zero on success or error code on failure.

Note: Besides the keywords stored in the file (which are dependent on the file format), the set of "virtual" keys is provided – see table 4.1.

`int` **konv_getl**`(void *handle, const char *key, long *val);`

This function reads single integer data from the file header.

The *handle* parameter is the handle created by *konv_open** function. The *key* parameter specify which parameter shall be read. The value is stored to the variable, which pointer is given in the *val* parameter. The function returns zero on success or error code on failure.

Note: Besides the keywords stored in the file (which are dependent on the file format), the set of "virtual" keys is provided – see table 4.1.

`int` **konv_getd**`(void *handle, const char *key, double *val);`

This function reads single real number data from the file header.

The *handle* parameter is the handle created by *konv_open** function. The *key* parameter specify which parameter shall be read. The value is stored to the variable, which pointer is given in the *val* parameter. The function returns zero on success or error code on failure.

Note: Besides the keywords stored in the file (which are dependent on the file format), the set of "virtual" keys is provided – see table 4.1.

`int` **konv_gkyn**`(void *handle, int index, char *key, int keylen, char *val, int vallen, char *com, int comlen);`

This function is intended for sequential reading of all data stored in the file header.

The *handle* parameter is the handle created by *konv_open** function. The second parameter is the index of the keyword starting by zero. The keyword is stored to the *key* buffer, the value is stored to the *val* buffer, and the comment is stored to the *com* parameter. The *keylen*, *vallen* and *comlen* parameters specify the size of the buffers in bytes. The function return zero on success or non-zero value, if the index is negative or beyond the end of the header.

Note: Format independent set of virtual keys is not supported by this function.

`int` **konv_g2de**`(void *handle, float *buf, int flipx, int flipy);`

This function reads image data from the file to the array of single-precision real numbers. Image flipping is supported.

The *handle* parameter is the handle created by one of the *konv_open** functions. The second parameter is the pointer to the allocated memory buffer, which must be large enough to keep whole image. Set the *flipx* and *flipy* parameter to non-zero value to flip the image in x and y axis respectively.

`int` **konv_close**`(void *handle);`

This function closes the file and destroys the internal descriptor. After calling this function, the handle is no more valid and must not be used in any of the *konv_xxx* function.

The *handle* parameter is the handle created by one of the *konv_open** function. The funtion returns zero on success or error code on failure.

Keyword	**Data type**	**Description**
background	real	Background level in ADU
bitpix	integer	Number of bits per pixel
date	string	Observation date (yyyy-mm-dd)
exposure	real	Exposure duration in seconds
filter	string	Name of the pass band (filter)
height	integer	Image height in pixels
magic	string	File format descriptor
range	real	Pixel value range in ADU
temperature	real	CCD temperature in deg. C
time	string	Observation time (hh:mm:ss.sss)
width	integer	Image width in pixels

Table 4.1: Format independent (virtual) keys for reading CCD frames

4.2.2 Time correction

This functions are intended for modification of the time of observation stored in FITS files. Use the conversion functions first if the source file is saved in another format. It is possible to modify the header of the original frame or to make a new FITS file.

`int` **tcor_init**`(long tcor);`

This function initializes the internal variables for time correction of CCD frame files. Although the function doesn't allocate any internal buffers, you should call the *tcor_clean* function when the processed has been finished (for compatibility reasons).

The *tcor* parameter specify the time correction applied to the time of observation in seconds. Positive values of the correction shift the time to the future and negative values shift it to the past.

`int` **tcor**`(const char *sci, const char *out, char *date, char *time);`

The function makes time correction of the single FITS file. The source file is identified by its path and name in the *sci* parameter. The *out* parameter specify the name of the output file. If the second parameter is NULL, the source file will be changed. You can optionally provide two memory buffers in the third and four parameters, where the string representation of corrected date and time shall be stored to. Zero is returned on success or error code on failure.

`int` **tcor_clean**`(void);`

Even though this function does nothing, it should be called after processing, because of compatibility of your code with future versions.

4.2.3 Dark-frame correction

This functions are intended for dark-frame correction of FITS frames. Use the conversion functions first if the source or the correction file is saved in another format. It is possible to modify the header of the original frame or to make a new FITS file.

`int` **dark_init**`(const char *drk);`

The function reads the correction frame and initializes the internal variables. You must call the *dark_clean* function when the processing is finished to free the memory allocated by this function.

The *drk* parameter specify the path and name of the file with dark correction frame. The function returns zero on success and error code on failure.

`int` **dark**`(const char *sci, const char *out);`

The *dark* function makes correction of the single FITS file.

The source file is identified by its path and name in the *sci* parameter. The *out* parameter specify the name of the output file. If the second parameter is NULL, the source file will be changed. Zero is returned on success and error code on failure.

`int` **dark_clean**`(void);`

This function frees the memory buffers allocated by the *dark_init* function.

4.2.4 Flat-frame correction

This functions are intended for flat-frame correction of FITS frames. Use the conversion functions first if the source or the correction file is saved in another format. It is possible to modify the header of the original frame or to make a new FITS file.

`int` **flat_init**`(const char *flt);`

The function reads the correction frame and initializes the internal variables. You must call the *flat_clean* function when the processing is finished to free the memory allocated by this function.

The *flt* parameter specify the path and name of the file with flat correction frame. The function returns zero on success and error code on failure.

`int` **flat**`(const char *sci, const char *out);`

The *flat* function makes correction of the single FITS file.

The source file is identified by its path and name in the *sci* parameter. The *out* parameter specify the name of the output file. If the second parameter is NULL, the source file will be changed. Zero is returned on success and error code on failure.

`int` **flat_clean**`(void);`

This function frees the memory buffers allocated by the *flat_init* function.

4.2.5 Photometry

This set of functions is designed for detecting stars on FITS frames and measurement of their brightness. The output is written to a photometry file (see chapter 5.3).

`int` **mphot_init**`(const char *optfile, const char *apfile);`

The function reads the configuration files and initializes the internal variables. You must call the *phot_clean* function when the processing is finished to free the memory allocated by this function.

The *optfile* parameter specify the path and name of the photometry configuration file *muniphot.ini*, the *apfile* parameter specify the path and name of the aperture configuration file *apertures.ini*. If NULL value is given, the program uses default path and name. The function returns zero on success and error code on failure.

`int` **mphot**`(const char *sci, const char *out, double *jd, int *nostar);`

The *mphot* function performs photometry on a single FITS file.

The source file is identified by its path and name in the *sci* parameter. The *out* parameter specify the name of the output file (it cannot be NULL). If the *jd* parameter isn't NULL, the time of observation is passed out through it. Optionally, you can set the *nostar* parameter to the pointer to variable, where the number of the stars detected shall be stored to. Zero is returned on success and error code on failure.

`int` **mphot_clean**`(void);`

This function frees the memory buffers allocated by the *mphot_init* function.

4.2.6 Matching stars

This set of functions is intended for matching stars in a set of photometry files against one reference frame. The source file must be a photometry file, reference file can be either a photometry or a catalogue file. The output is written always in photometry file format.

`int` **match_init**`(const char *reffile, const char *optfile);`

The function reads the reference file and the configuration file and initializes the internal variables. You must call the *match_clean* function when the processing was finished to free the memory allocated by this function.

The *reffile* parameter specifies the path and name of the reference file. The *optfile* parameter is the path and name of the configuration file *munimatch.ini*. If NULL value is given, the program uses default path and name. The function returns zero on success and error code on failure.

`int` **match**`(const char *infile, const char *outfile, int *mstar);`

The *match* function performs star matching on a single photometry file.

The source file is identified by its path and name in the *infile* parameter. The *outfile* parameter specify the name of the output file (it cannot be NULL). If the *mstar* parameter isn't NULL, the total number of stars matched is passed out through it. Zero is returned on success and error code on failure.

`int` **match_clean**`(void);`

This function frees the memory buffers allocated by the *match_init* function.

4.2.7 Listing

This set of functions are used for matched processing photometry files and making output data reports in various format.

`int` **mlist_init**`(const char *optfile, int aper, int outfrm, int nstar, int *stars, const char *ra, const char *dec, const char *lon, const char *lat);`

The function reads the configuration file and initializes the internal variables. You must call the *mlist_clean* function when the processing was finished to free the memory allocated by this function.

The *optfile* parameter specify the path and name of the configuration file *munilist.ini*. If NULL value is given, the program uses default path and name. The aperture index is set by the *aper* parameter. The *outfrm* is bit mask of constants, which specify the output format. If the output format requires selection of the stars, the *nstar* parameter must be the number of stars selected and the *stars* parameter must point to the array of indices of selected stars. Some formats require object's celestial coordinates or observer's geographical coordinates. The function returns zero on success or error code on failure.

`int` **mlist_read**`(const char *infile, int frame);`

The functions reads and processes single photometry file.

The *infile* parameter specify the path and name of the source file. The second parameter, *frame*, is the ordinal number of the frame, which is hereafter used for identification of the frame.

`int` **mlist_getpar**`(int *ncols, int *nrows);`

This function stores the number of columns and the number of rows stored in the internal memory buffer.

Number of columns is stored to the variable specified in the *ncols* parameter, the *nrows* parameter specify the variable, where the number of rows shall be stored to. The function returns zero on success or error code on failure.

`int` **mlist_getcol**`(int col, char *buf);`

Call this function to get the name of particular column. Columns are numbered starting by zero.

The *col* parameter is the column index (starting by zero), the *buf* parameter points to the allocated memory buffer, where the name shall be stored to as null-terminated string. The function returns zero on success and error code on failure.

`int` **mlist_getx**`(int *frm, double *jd, double *hcor, double *amass);`

The function reads table data in special columns for all rows.

All four parameters must are pointers to the allocated memory, where the data shall be stored to. The number of items of the arrays must be equal to or greater than the number of rows (see *mlist_getpar* function). The function stores frame identifiers to the *frm* array, Julian dates are copied to the *jd* array, values of heliocentric correction is stored to *hcor* array and values of air-mass coefficient is stored to the *amass* array. The function returns zero on success and error code on failure.

`int` **mlist_gety**`(int col, double *val, double *err);`

The function reads table data in particular column for all rows.

The first parameter is the column index (starting by zero). The *val* parameter is the pointer to the allocated array of doubles, where the values shall be stored to. The *err* parameter (if present) specify the address of the memory buffer, where the errors of the values shall be stored to.

`int` **mlist_write**`(const char *outfile);`

This function writes table data to the file specified in the only parameter.

`int` **mlist_clean**`(void);`

This function frees the memory buffers allocated by the *mlist_** functions.

4.3 Miscellaneous functions

4.3.1 Dark-frame averaging

This functions are used in the *meandark* tool for averaging of the dark correction frames. The source files and the target file are stored in the FITS format.

`int` **mdark_init**`(const char *optfile);`

The function initializes the internal variables. You must call the *mdark_clean* function when the processing was finished to free the memory allocated by this function.

The *optfile* parameter is not used in this version, you should set it always to NULL. The function returns zero on success or error code on failure.

`int` **mdark_read**`(const char *infile);`

This function reads single FITS frame and stores it to the memory.

The *infile* parameter specify the path and name of the file. The function returns zero on success. Otherwise, the error code is returned.

`int` **mdark_write**`(const char *outfile);`

The *mdark_write* function computes the target frame by averaging corresponding pixels on source files. The result is stored to the file in the FITS format.

The *outfile* parameter specify the path and name of the target file. If it exists, it is overwritten. The function returns zero on success. Otherwise, the error code is returned.

`int` **mdark_clean**`(void);`

This function frees the memory buffers allocated by the *mdark_** functions.

4.3.2 Flat-frame mastering

This functions are used in the *autoflat* tool for merging of the dark correction frames. The source files and the target file are stored in the FITS format.

`int` **aflat_init**`(const char *optfile);`

The function reads the configuration file and initializes the internal variables. You must call the *mflat_clean* function when the processing was finished to free the memory allocated by this function.

The *optfile* parameter specify the path and name of the configuration file *autoflat.ini*. If NULL value is given, the program uses default path and name. The function returns zero on success or error code on failure.

`int` **aflat_read**`(const char *infile);`

This function reads single FITS frame and stores it to the memory.

The *infile* parameter specify the path and name of the file. The function returns zero on success. Otherwise, the error code is returned.

`int` **aflat_write**`(const char *outfile, float *skymed, float *skysig);`

The *aflat_write* function computes the target frame by means of the robust mean algorithm on all corresponding pixels on source files. The result is stored to the file in the FITS format.

The *outfile* parameter specify the path and name of the target file. If it exists, it is overwritten. The optional *skymed* parameter receives average brightness of the output frame and the *skysig* parameter is set to the value of standard deviation of its brightness. The function returns zero on success. Otherwise, the error code is returned.

`int` **aflat_clean**`(void);`

This function frees the memory buffers allocated by the *aflat_** functions.

4.3.3 Finding variable stars

This functions are designated to help the user with finding variable stars on a set of CCD frames.

`int` **mfind_init**`(const char *optfile, int ap, int thr);`

The function reads the configuration file and initializes the internal variables. You must call the *mfind_clean* function when the processing was finished to free the allocated memory.

The *optfile* parameter specify the path and name of the configuration file *munifind.ini*. If NULL value is given, the program uses default path and name. The *ap* parameter is the aperture index, the *thr* parameter is the threshold in % (see 3.3.4 for further details). The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_readsrt**`(const char *srtfile);`

This function reads single photometry file and stores it to the memory. The *infile* parameter specify the path and name of the file. The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_readall**`(const char *srcfile);`

This function reads photometry data stored in the file in the READALL format and stores it to the memory. The *srcfile* parameter specify the path and name of the file. The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_compute**`(int ref);`

Call this function after reading all source files. It computes mean magnitude and standard deviation of the stars discarding those of poor quality. The output table is stored in the memory. If the *ref* parameter is not zero, it specifies the index of the comparison star. Use zero value to enable automatic detection of best comparison star. The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_getpar**`(int *nstars, int *nimages, int *ref);`

The functions reads several internal variables filled by means on the *mfind_compute* function. The *nstars* parameter receives the number of table rows (number of stars), the *nimages* parameter receives number of frames processed, and the *ref* points to the variable which receives the index of the comparison star used. The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_get**`(double *mag, double *stddev);`

The function reads table data from internal memory computed by means of the *mfind_compute* function. The parameters point to two arrays of doubles, where data shall be stored to. The *mag* array receives mean magnitudes and standard deviations are stored to the *stddev* array. The array must be large enough – call *mfind_getpar* function to determine the number of stars. The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_gdata**`(int star, double *jd, double *mag, double *err);`

The function makes a light curve of a particular star. First parameter specifies the index of the star. Magnitudes are always relative to the comparison star. Call *mfind_getpar* function to determine the number of frames and the index of the comparison star. Next three parameters points to the arrays, where the data shall be stored to. The *jd* array receives Julian dates, the magnitudes and their errors are stored to the *mag* and the *err* arrays. The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_write**`(const char *outfile);`

This function saves the table stored in the internal memory to file. The *outfile* parameter specify the path and name of the output file. The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_wdata** `(int star, const char *outfile);`

This function makes a light curve of a particular star and writes it to a file. First parameter specifies the index of the star. The *outfile* parameter is the path and name of the output file. The function returns zero on success. Otherwise, the error code is returned.

`int` **mfind_clean** `(void);`

This function frees the memory buffers allocated by the *mfind_** functions.

4.4 Data types and constants

Following constants are used on the library interface. They are declared in the *cmunipack.h* public header file.

4.4.1 Error codes

The constants presented in the table 4.2 are used for indication of a reason of failure.

Konstanta	**Význam**
MPK_MEMORY	Insufficient memory
MPK_CANNOT_OPEN_SRC	Cannot open the source file
MPK_CANNOT_OPEN_OUT	Cannot open the destination file
MPK_CANNOT_OPEN_PAR	Cannot open the parameter file
MPK_NO_DIRFILE	Dirfile not found
MPK_NO_INPUT_FILES	No input files
MPK_NO_DATA	No data loaded in memory
MPK_UNKNOWN_FORMAT	Unknown format of source file
MPK_INVALID_SIZE	Invalid dimensions of image
MPK_INVALID_PAR	Invalid value of parameter
MPK_INVALID_DATA	Invalid data in source file
MPK_INVALID_HEADER	Error in header of the source file
MPK_INVALID_DATE	Invalid format of date or time of observation
MPK_ERROR	General error
MPK_KEY_NOT_FOUND	Key not found
MPK_DATA_OVERFLOW	An underflow has been occurred during computation
MPK_DATA_UNDERFLOW	An overflow has been occurred during computation
MPK_CLOSED_FILE	Operation not allowed on closed file
MPK_FRAME_NOT_FOUND	Frame not found
MPK_INVALID_OBJRA	Invalid format of right ascension
MPK_INVALID_OBJDEC	Invalid format of declination
MPK_INVALID_OBSLON	Invalid format of longitude
MPK_INVALID_OBSLAT	Invalid format of latitude
MPK_MISSING_OBJCOORDS	Missing object's equatorial coordinates
MPK_MISSING_OBSCOORDS	Missing observer's geographical coordinates
DRK_CANNOT_OPEN_DRK	Dark frame not found
DRK_DIFFERENET_SIZE	Dimensions of dark-frame and scientific image are different
FLT_CANNOT_OPEN_FLT	Flat frame not found
FLT_DIFFERENET_SIZE	Dimensions of flat-frame and scientific image are different
FLT_MEAN_ZERO	Mean value of flat frame is zero
FLT_DIVZERO	An zero value has been occurred on flat frame
MAT_CANNOT_OPEN_REF	Reference file not found
MAT_FEW_POINTS_REF	Too few stars in the reference file
MAT_FEW_POINTS_SRC	Too few stars in the source file
MAT_MATCH_NOT_FOUND	Coincidences not found
MAT_XML_PARSE	Error in the catalogue file
AFL_DIFFERENET_SIZE	Input frames are not compatible (different sizes)
MDK_DIFFERENET_SIZE	Input frames are not compatible (different sizes)
MFI_TOO_FEW_DATA	Too few data in source files
MFI_REF_NOT_FOUND	Reference star was not found

Table 4.2: Error code constants

4.4.2 Source file format constants

The constants presented in the table 4.3 are used for identification of source file format in *konv_** functions.

Constant	**Description**
MPK_FORMAT_UNKNOWN	Unknown format
MPK_FORMAT_FITS	FITS image format
MPK_FORMAT_SBIG	ST-x image format
MPK_FORMAT_OES	OES Astro format

Table 4.3: Source file format constants

4.4.3 Conversion mode constants

The constants presented in the table 4.4 are intended for specification of the working mode in *konv_** functions.

Constant	**Description**
MPK_CONVERT_NORMAL	Normal conversion
MPK_CONVERT_HEADER	Create only header of FITS file
MPK_CONVERT_INFO	Print short info only
MPK_CONVERT_DETAILS	Print detailed info only

Table 4.4: Conversion mode constants

4.4.4 Destination file format constants

The following set of constants (see table 4.5 is intended for specification output data format. The values are given in parameter of the *mlist_init* function and the bitmap should contain exactly one of the constant out of *format specifiers*, which defines file format (see table 4.5, and optionally one or more modifiers from the *Format modifiers* category.

Category	**Constant**	**Description**
Format specifiers	MPK_FORMAT_DIFFMAG	Differential magnitudes
	MPK_FORMAT_INSTMAG	Instrumental magnitudes
	MPK_FORMAT_READALL	The „Read-all“ format
	MPK_FORMAT_TRACKLST	Track-list format
Format modifiers	MPK_FORMAT_JDHEL	Store helioc. JD instead of geocentric one
	MPK_FORMAT_HELCOR	Include value of heliocentric correction
	MPK_FORMAT_AIRMASS	Include value of air-mass coefficient

Table 4.5: Destination file format constants

5 Files

In this chapter, you will find the description of all file formats used in the C-Munipack project. Following text is arranged into four sections. In the first part, the supported formats of input files are introduced. The detailed description of output files is included in the second section. Next part consists of the description of photometry and catalogue files. In the closing part, the configuration files are described.

All keywords and invariant phrases, which are included directly in the files are printed in *cursive font*. Names of files and directories are printed in `teletype face`. Default values of parameters are places in the [square brackets].

5.1 Input files

In this section of the user manual, the overview of CCD file formats is introduced. The FITS format, which has become a standard in practice, is most common of them. In the C-Munipack project, this format has been also chosen for storing temporary image data between the individual parts of the reduction pipeline.

The *Flexible Image Transport System* (FITS) format is used for storing CCD frames by most of the existing CCD astronomy applications. Since the specification is intended to be quite general, if you decide to implement a software accepting FITS files, one important disadvantage takes place: there is no widely adhered specification of data and time format and that's why you can find many of them in practice. The C-Munipack software can automatically recognize several widely used formats.

All programs included in the C-Munipack project uses the *CFITSIO* library, version 3.006, for manipulating with FITS files. The library was published under open-source license and it is available on the internet.

The *CCDOPS* program uses special file format. The advantage of this format is the simple compression algorithm, which considerably reduces the size of the files. Also old models of cameras made by the *OES GmbH*, Germany, uses its own format for storing CCD data. These two formats are supported by the C-Munipack project. Own implementation has been made in accordance to the available documents.

5.2 Output files

Final products of the reduction process are stored to output data files in simple ASCII text format. Such files can be imported to most of existing post-processing software and they are compatible with original Munipack data files, too.

5.2.1 Light curve data files

Light curves (dependency of brightness on observation time) are usually final product of the reduction process. A light curve table consists of observation time in Julian date form followed by magnitudes and their errors of all selected stars. Two basic output working modes are provided: in differential mode, magnitudes stored in the table are differences between each pair of the stars. On the other hand, instrumental mode allows you to print out raw values computed by photometry. Note, that instrumental magnitudes cannot be compared to absolute ones, which is found in the photometry catalogues, without further processing.

Output files are stored in the ASCII format; the end of the line is represented by `CR+LF` in the DOS/MS Windows environment and by `LF` in the Unix/Linux environment. First line contains always a list of column names separated by single space character. Second line consists of additional information (aperture, filter, etc.) and has no special formatting, it must not start by number, though.

On the following lines, the table values are stored. The values are separated by tab character or single space, rows are separated by the end-of-line character (see above). Parsers must ignore all additional white characters. Empty lines indicates, that the corresponding frame was not successfully processed and thus a brightness of a variable or a comparison star could not be determined. See table 5.1 for short description of columns.

Mode	Keyword	Description
Common part	`JD`	Geocentric Julian date of observation
	`JDHEL`	Heliocentric Julian date of observation (optional)
	`HELCOR`	Heliocentric correction in days (optional)
	`AIRMASS`	Air-mass coefficient (optional)
Differential magnitudes	`V-C`	Difference of variable and comparison star
	`s1`	Error of V-C value
	`V-C1`	Difference of comparison and check star #1
	`s2`	Error of C-C1
	`...`	...
Instrumental magnitudes	`MAG0`	Brightness of star #1
	`ERR0`	Error of MAG0
	`MAG1`	Brightness of star #2
	`ERR1`	Error of MAG1
	`...`	...

Table 5.1: Description of light curve file

5.2.2 Track-list data files

Track lists (dependency of frame shift on observation time) are usually used for determining the value of periodic error of the telescope mount. A track-list table consists of observation time in Julian date form followed by shifts in the horizontal (X) and the vertical (Y) axes in pixels relative to the reference frame.

Output files are stored in the ASCII format; the end of the line is represented by `CR+LF` in the DOS/MS Windows environment and by `LF` in the Unix/Linux environment. First line contains a list of column names separated by single space character. Second line consists of additional information and has no special formatting, it must not start by number, though.

On the following lines, the table values are stored. The values are separated by tab character or single space, rows are separated by the end-of-line character (see above). Parsers must ignore all additional white characters. Empty lines indicates, that the corresponding frame was not successfully processed and thus shift values could not be determined. See table 5.2 for short description of columns.

Keyword	Description
`JD`	Geocentric Julian date of observation
`OFFSETX`	Relative shift in horizontal direction
`OFFSETY`	Relative shift in vertical direction

Table 5.2: Description of track-list file

5.2.3 Munifind's ouput data file

An output file consists of table of indices, mean magnitudes and standard deviations for all detected stars. Such data is intended for detecting variable stars on a set of CCD frames. Magnitudes are always in differential form, they are relative to the comparison star.

Output files are stored in the ASCII format; the end of the line is represented by `CR+LF` in the DOS/MS Windows environment and by `LF` in the Unix/Linux environment. First line contains a list of column names separated by single space character. Second line consists of additional information and has no special formatting, it must not start by number, though.

On the following lines, the table values are stored. The values are separated by tab character or single space, rows are separated by the end-of-line character (see above). Parsers must ignore all additional white characters. See table 5.3 for short description of columns.

Keyword	Description
`INDEX`	Ordinal number of a star in the reference file
`MEAN_MAG`	Mean relative magnitude of a star
`STDEV`	Standard deviation of previous value
`GOODPOINTS`	Number of measurements used for computation

Table 5.3: Description of munifind's output file

5.3 Photometry files

Photometry files are made by the photometry phase during the reduction of CCD frames. The photometry file consists of a header, which carries important information on the corresponding frame, and a list of all detected stars and their photometry data (magnitudes and errors). Output file of the matching routine is stored to another set of files, they have the same format and can be distinguished by an extension. Files saved by photometry have the `srt` extension and files generated by matching program have the `mat` extension.

Photometry files are stored in the ASCII format; the end of the line is represented by `CR+LF` in the DOS/MS Windows environment and by `LF` in the Unix/Linux environment. The header is always stored on the first three lines. In the first line, there is a list of keywords, which define meaning of values written on the next line, see following table 5.4. The position of keywords mentioned in the table is decisive, but values and keywords can be aligned to the left or to the right. Empty space must be filled by space characters. Third line must be always empty.

Note: Because this format is not extensible, it has been designed to be superseded in the version 1.2 in favour of new XML based format.

Keyword	Positions	Description
`NL`	1 – 3	Format version, always 2
`NX`	5 – 8	Frame width in pixels
`NY`	10 – 13	Frame height in pixels
`LOWBAD`	15 – 21	Lowest valid pixel value in ADU
`HIGHBAD`	23 – 29	Highest valid pixel value in ADU
`THRESH`	31 – 37	Threshold ???
`AP1`	39 – 45	Radius of first aperture in pixels
`PH/ADU`	47 – 53	A/D converter gain (e- per ADU)
`RNOISE`	55 – 61	Readout noise ???
`JD`	63 – 77	Julian date of center of the exposure
`FILTER`	79 – 94	Filter (band)
`EXPTIME`	96 – 105	Exposure time in seconds
`FWHM`	107 – 113	Mean computed value of FWHM

Table 5.4: Description of photometry file - header

From the fourth line to the end of file, each triple of lines carries information on particular star. The values are stored in the ASCII format on first two lines, positions are decisive, see the table 5.5 for details. Values can be aligned to the right or to the left. Empty space must be filled by space characters. Each third line must be always empty.

Line	Positions	Description
Line 1	1 – 6	Ordinal number or numeric identifier
	8 – 15	Horizontal position of center of a star
	17 – 24	Vertical position of center of a star
	26 – 33	Instrumental magnitude of a star (aperture #1)
	35 – 42	Instrumental magnitude of a star (aperture #2)
	44 – ...	...
Line 2	1 – 13	Mean sky level v ADU (local background)
	15 – 19	Standard deviation of sky level
	21 – 25	Unused, always zero
	26 – 33	Error of magnitude (aperture #1)
	35 – 42	Error of magnitude (aperture #2)
	44 – ...	...
Line 3		Always empty

Table 5.5: Description of photometry files - data

5.4 Catalogue files

Using catalogue files can speed up your work, in case you often observe particular star field, for example long-term monitoring of a variable star. The structure of the data stored in a file is very similar to a photometry file, but it contains the selection of stars (variable star, comparison star, etc.). When a catalogue file is used instead of a reference file in matching phase of reduction, the selection is automatically restored from that file.

Catalogue files are saved in XML-based format. The EXPAT library has been used for reading the files in XML based formats, see the *References* section for details.

File structure

cat_file - root element of XML document
Attributes: None
Model: (info, selection, stars)

info - file header
Attributes: None
Model: (object?, ra2000?, dec2000?, observer?, observatory?, telescope?, camera?, filter?, fov?, orientation?, comment?)

object, **ra2000**, **dec2000** - object's designation and coordinates (optional)
Attributes: None
Model: #PCDATA

observer, **observatory** - observer's name, observatory (optional)
Attributes: None
Model: #PCDATA

telescope, **camera**, **filter** - Description of telescope, camera, filter (optional)
Attributes: None.
Model: #PCDATA

fov, **orientation** - field of view (width x height), rotation (optional)
Attributes: None.
Model: #PCDATA

comment - user comments *Attributes*: None.
Model: #PCDATA

selection - table of selected stars *Attributes*: None.
Model: (select*)

select - assigns designation to a star by its identifier. Variable star is labelled as 'var', comparison star 'comp', check stars 'chk1', 'chk2', etc.
Attributes: id - identifier, label - designation
Model: Empty

stars - table of all stars on the frame *Attributes*: width, height - frame size in pixels
Model: (s*)

s - reference position and magnitude of stars
Attributes: id - identifier; x, y - position in pixels relative to left top corner of the frame, m - instrumental magnitude, e - std. deviation of magnitude
Model: Empty

5.5 Configuration files

Configuration files of the C-Munipack package are text files; every parameter is written on a separate line. There is always a parameter (key), followed by an equals sign, then a value is present. Optionally there can be a comment separated by the sharp character. In some cases the value can be enclosed in parentheses or in single quotes. The spaces or tabs (white space) are ignored if they are at the beginning of the line or around an equals sign or at the end of the line. The end of the line should be represented by `CR+LF` in the DOS/MS Windows environment and by `LF` in the Unix/Linux environment.

It is allowed to add comments to the file – each line starting with a sharp character up to the end of the line is considered a comment and ignored. The lines containing only white spaces or comments are ignored, too.

5.5.1 Apertures.ini

The definitions of the apertures used in the photometry are stored in the `Apertures.ini` file. The values are always in pixels. Not all of the lines need to be defined; if there is some unused aperture, it is skipped and the brightness of the star with this aperture is not calculated.

- *A1, A2, ... A12* – radius of the apertures no. 1..12 [0 = the aperture shall not be used]
- *IS* – inner radius of the aperture for measuring the brightness of the sky; [20]
- *OS* – outer radius of the aperture for measuring the brightness of the sky; [30]

5.5.2 Munifind.ini (Parameters for searching the variables)

The parameters used in the *Munifind* function are stored in this file.

- *APERTURE* – number of the aperture used (from 1 to 12) [1]
- *THRESHOLD* – this value determines the percentage of the measurements in the set of the input files that are needed so that the star can be added to the output file [30]

5.5.3 Munilist.ini (Parameters of the output files)

This file contains the parameters for the *Munilist* function.

- *APERTURE* – no. of the aperture used [1]
- *OUTFORM* – format of the output file: 0=differential mag.; 1=instrumental mag. [0]

5.5.4 Munimatch.ini (Parameters for the matching function)

This file contains the parameters for the *Munimatch* function.

- *RSTARS* – number of stars from the reference file which will be used for matching [10]
- *ISTARS* – number of vertexes of the searched-for polygons [5]
- *CLIP* – threshold for determining the inaccuracy of the polygons; in the multiplies of standard deviation (sigma) [2.5]

5.5.5 Muniphot.ini (Photometry options)

In this file, the parameters for the *Muniphot* function are saved.

- *READNS* – the level of the noise per one image in ADU [15.00]
- *GAIN* – the number of electrons per ADU; the correct value should be set according to the CCD camera manual
- *LODATA* – the lowest valid value of a pixel [7.00]
- *HIDATA* – the highest valid value of a pixel [65535.00 = 16 bits)]
- *FWHM* – expected width of an object halfway to the maximum in pixels (*F*ull *W*idth at *H*alf *M*aximum) for which the calculation should be optimized. For the first try use 3.00
- *TRESH* – threshold determining the level of the dimness of the stars that should be detected; for the first try use 4.00
- *LOSHARP* – lower bound for the sharpness of the stars [0.20]
- *HISHARP* – upper bound for the sharpness of the stars [1.00]
- *LOROUND* – lower bound for the roundness of the stars [-1.00]
- *HIROUND* – upper bound for the roundness of the stars [1.00]

Alphabetical list of error messages

A

Aperture radii must be between 1.0 and outer sky radius

One or more aperture radii are less than one pixel or grater than outer sky annulus. Probably the size of the annulus was changed, but there are some large apertures in the table. Check the table of aperture radii and specified sky annulus.

C

Cannot open the destination file

The destination file cannot be created or modified. Check: a) the destination media and the target directory are writable, b) the file is not opened in another application, c) you have got proper access rights for writing the file, d) you had entered valid path and file name.

Cannot open the parameter file

The reading of the configuration file failed. Check: a) you had entered the path and the file name correctly and the specified file exists, b) the file is not opened in another application, c) you have got proper access rights for reading the file. If you see this message in the Muniwin software, it is probably a bug in the software. Please send a bug report to the author.

Cannot open the source file

The source file is not available for reading. Check: a) the file is not opened in another application, b) you had entered the path and the file name correctly and the specified file exists and c) you have got proper access rights for reading the file.

Coincidencies not found

The coincidencies between the reference file and the source photometry file were not found. In most cases this error indicates, that the shift between two frames is too high, or you have mixed frames, which belong to different star field. This is not critical failure, this frame will be skipped in the following process.

Cannot copy file ... to ...

The copying of the file failed. Check: a) the destination media and the target directory are writable, b) the files are not opened in another application, c) you have got proper access rights for reading and writing the files, d) you had entered valid paths and file names and the source file exists.

D

Dark frame not found

The specified dark correction frame is not available for reading. Check: a) the file is not opened in another application, b) you had entered the path and the file name correctly and the specified file exists and c) you have got proper access rights for reading the file. If you see this message in the Muniwin software, it is probably a bug in the software. Please send a bug report to the author.

Data overflow has been occured during computation

The result of computation is above its upper limit. This is not critical failure, but one or more pixels on the resulting frame aren't valid and they will be skipped in the following process. Usually this error indicates bad quality of a dark or a flat frame used.

Data underflow has been occured during computation

The result of computation is below its lower limit. This is not critical failure, but one or more pixels on the resulting frame aren't valid and they will be skipped in the following process. Usually this error indicates bad quality of a dark or a flat frame used.

Default directory ... could not be created. Please check the configuration

The standard directory specified in the configuration is not valid. Please, open the "Preferencies" dialog in the "Tools" menu and check the specified paths.

Dimensions of dark-frame and scientific image are different

It is not allowed to use this specified dark frame for correction of the specified exposure file, because they have different size (width or height).

Dimensions of flat-frame and scientific image are different

It is not allowed to use this specified flat frame for correction of the specified exposure file, because they have different size (width or height).

Directory ... could not be created

The creation of the specified directory failed. Check: a) the destination media and the target directory are writable, b) you have got proper access rights for writing to the destination directory, c) you had entered valid path and directory name.

Dirfile not found

The file with list of input frames doesn't exists or cannot be opened for reading. Check: a) the file is not opened in another application, b) you had entered the path and the file name correctly and the specified file exists and c) you have got proper access rights for reading the file.

E

Error in header of the source file

Invalid format, keyword or value was found in the header of the file. This error indicates the corruption of the input file.

Error while reading file

Reading of the input file has failed during the operation.

F

File is open in read-only mode

If you see this message, it is definitely a bug in the software. Please send a bug report to the author.

Flat frame not found

The specified flat correction frame is not available for reading. Check: a) the file is not opened in another application, b) you had entered the path and the file name correctly and the specified file exists and c) you have got proper access rights for reading the file. If you see this message in the Muniwin software, it is probably a bug in the software. Please send a bug report to the author.

Frame not found

The required frame is not available for reading. If you see this message, it is definitely a bug in the software. Please send a bug report to the author.

G

General error

If you see this message, it is definitely a bug in the software. This is critical failure. Something is wrong, but the program cannot recognize the reason. Please send a bug report to the author.

I

Input frames are not compatible (different sizes)

The operation could not be completed, because some of the source frames in the set have different size (width or height). This error is issued if you are trying to make a master flat frame or a master dark frame from a set of incompatible frames.

Insufficient memory

The allocation of memory buffer failed. It can indicate invalid value in the source frame or data file, for example too large size of the frame. If you see this message, it is definitely a bug in the software. Please send a bug report to the author.

Invalid data in source file

Invalid value was found in the body of the file. This error indicates the corruption of the input file.

Invalid dimensions of image

A dimension (width or height) of the specified frame is not valid. This error indicates the corruption of the input file.

Invalid format of date of observation

The format of the observaton date is not valid. Check the user manual for syntax of the date/time values.

Invalid format of date or time of observation

The format of date or time is not supported. It can indicate the corruption of the input file. Unfortunately, there is not any standard how to save a date and time of observation to FITS file, so every program uses its own special format. Although current version of C-Munipack can read many different formats, it may happens, that your program uses the another one. Send me one frame or two as sample (see email address above) and I will add it to the sources promptly.

Invalid format of latitude

The format of the geographical latitude is not valid. Check the user manual for syntax of the geographical coordinates.

Invalid format of longitude

The format of the geographical longitude is not valid. Check the user manual for syntax of the geographical coordinates.

Invalid format of declination

The format of the object's declination is not valid. Check the user manual for syntax of the equatorial coordinates.

Invalid format of right ascension

The format of the object's right ascension is not valid. Check the user manual for syntax of the equatorial coordinates.

Invalid value of parameter

This error is issued when a value of a parameter is outside the range or is not valid. Check the configuration file and consult the user manual for meaning of parameters. If you see this message in the Muniwin software, it is probably a bug in the software. Please send a bug report to the author.

Invalid format of time of observation

The format of the observaton time is not valid. Check the user manual for syntax of the date/time values.

It is not allowed to delete the reference file

It is not allowed to delete this frame from the table of input file, because this is the reference file.

K

Key not found

If you see this message, it is definitely a bug in the software. Please send a bug report to the author.

M

Mean value of flat frame is zero

The specified flat frame is not usable for flat-field correction, because it is completely black (all pixels are set to zero).

Missing object's coordinates

The object's equatorial coordinates are needed for this operation, but they haven't been specified.

Missing observer's coordinates

The observer's geographical coordinates are needed for this operation, but they haven't been specified. Open 'Preferences' dialog and set the coordinates.

N

No data loaded in memory

The required data is not available, because table is empty. This error message definitely indicates a bug in the software. Please send a bug report to the author.

No file was successfully processed

The operation has been completed without critical failure, but no source file was processed correctly.

No input files

There have been no input files specified on command line or in the dirfile. Check the user manual for syntax of the command.

No stars selected

It is not allowed to perform this operation, because there are no selected stars. Run "Choose stars" dialog in "Plotting" menu first.

O

Operation not allowed on closed file

It is allowed to perform reading or writing on closed file. If you see this message, it is definitely a bug in the software. Please send a bug report to the author.

R

Reference file was not found

The specified reference or catalogue file is not available for reading. Check: a) the file is not opened in another application, b) you had entered the path and the file name correctly and the specified file exists and c) you have got proper access rights for reading the file. If you see this message in the Muniwin software, it is probably a bug in the software. Please send a bug report to the author.

Reference star was not found

Specified reference or comparison star was not found. If you see this message in the Muniwin software, it is definitely a bug in the software. Please send a bug report to the author.

Resulting image cannot be empty

The dimensions or the resulting image are zero. Check the specified size of the image.

Resulting image is too large

The dimensions or the resulting image are too high. Check the specified size of the image.

S

Syntax error in the catalogue file

The specified catalogue file is not valid. This error indicates the corruption of the catalogue file.

T

There are no valid entries in table of input files

You cannot perform this operation, because the table of input files is empty. Use "Add files" dialog in "Files" dialog to add files to the table.

Too few data in source files

There are too few frames to perform this operation. This error is issued by Varfind utility, if there are less than two input frames. Check the user manual for requirements of this operation.

Too few stars in the reference file

There are not enough stars in the reference or catalogue file. In most cases this error indicates, that the frame is of very low quality (clouds, closed shutter, ...). Select another reference frame and repeat the mathing process again.

Too few stars in the source file

There are not enough stars to perform star matching. In most cases this error indicates, that the original frame is of very low quality (clouds, closed shutter, ...). This is not critical failure, this frame will be skipped in the following process.

U

Unknown error status

If you see this message, it is definitely a bug in the software. This is critical failure. Some routine or module reported an error code which is not in the code table. Please send a bug report to the author.

Unknown format of source file

The format of the source file is not supported. It can indicate the corruption of the input file.

Z

Zero value has been occured on flat frame (can’t divide by zero)

The result of computation is not valid, because divisor is zero. This is not critical failure, but one or more pixels on the resulting frame is not valid and they will be skipped in the following process.

References

Algorithms and libraries:

Main part of the algorithms of reduction of CCD frames originate from the Munipack software by Filip Hroch. Graphical user interface was inspired by the Munidos software.

- Pence, William Dr.: CFITSIO library, http://heasarc.gsfc.nasa.gov/fitsio/
- Clark, James: EXPAT library, http://www.libexpat.org/
- Hroch, Filip: Munipack, http://munipack.astronomy.cz/
- Novák, R. and Král L.: Munidos, http://munipack.astronomy.cz/
- Král, Lukáš: Varfind, private communication
- Bakis, Volcan: Air-mass computation algorithm, private communication
- Pokorný, Zdeněk Dr.: Astronomické algoritmy pro kalkulátory, HaP hl. m. Prahy, 1984
- Drolon, Hervé et al.: FreeImage project, http://freeimage.sourceforge.net/

Development tools:

The source codes were compiled by freware C/C++ compiler Minimalist GNU for Windows. Graphical user interface was developed in the Borland DEPHI version 7 environment. Binary installation packages were created by means of the Nullsoft NSIS scriptable engine. The Microsoft's HTML Help Workshop was used to create electronic help. The documentation was typed in LaTeX.

- MinGW - Minimalist GNU for Windows, http://www.mingw.org/
- Nullsoft Scriptable Installation System, http://nsis.sf.net
- MiKTeX project, http://www.miktex.org/

Variable stars catalogues:

Optionally, the program uses catalogues of variable stars to search the object's coordinates, which are neccessary for computation of heliocentric correction and air-mass coefficient. The catalogue files are not included in the packages, but they are freely available at the following locations:

- Sternberg Astronomical Institute: General Catalogue Of Variable Stars,
  http://www.sai.msu.su/groups/cluster/gcvs/gcvs/index.htm
- Zejda, M. a kol.: Katalog BRKA, http://var.astro.cz/brno/

2. VERBATIM COPYING

3. COPYING IN QUANTITY

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

5. COMBINING DOCUMENTS

6. COLLECTIONS OF DOCUMENTS

> Copyright ©YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.