

VIMOS SOFTWARE FAMILY

Using the VIMOS Kernel

14 February 2006



Thank you for your interest in our Vision Inspection and Optical Measurement System (VIMOS). In this manual you will find information about the main functionality of the system.

Before going on reading the manual, we kindly ask you to read the following

DISCLAIMER

This documentation is provided for reference purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this documentation, this documentation is provided "as is" without any warranty whatsoever and to the maximum extent permitted, atto-Systems Ltd. and Wolf Systeme AG disclaim all implied warranties, including without limitation the implied warranties of merchantability, non-infringement and fitness for a particular purpose, with respect to the same. Neither atto-Systems Ltd. nor Wolf Systeme AG shall be responsible for any damages, including without limitation, direct, indirect, consequential or incidental damages, arising out of the use of, or otherwise related to, this documentation or any other documentation. Notwithstanding anything to the contrary, nothing contained in this documentation or any other documentation is intended to, nor shall have the effect of, creating any warranties or representations from atto-Systems Ltd., Wolf Systeme AG or any of their suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of this software.

The described software is provided 'as is', without any warranty expressed or implied. No guaranty is given that the software is suitable for any given purpose.

COPYRIGHT

Under the copyright laws, neither the documentation nor the software may be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of atto-Systems Ltd or Wolf Systeme AG, except in the manner described in the documentation or the applicable licensing agreement governing the use of the software. All rights are reserved. Do not reverse-engineer. Do not modify or distribute without all of the documentation.

© Copyright atto-Systems Ltd. and Wolf Systeme AG

Atto-Systems Ltd.
Sofia, Bulgaria

Wolf Systeme AG
Karlsbad, Germany

All rights reserved.

TRADEMARKS

All trademarks and copyrights mentioned within the documentation are respected. They are the property of their respective owners.

CONVENTIONS USED IN THIS MANUAL



INFORMATION. This sign marks section in the manual, which is for information only. You can decide to read or skip this section.



ATTENTION. This sign marks section of the manual, which is particularly important for the general understanding of VIMOS. Please, make sure to read this section before proceeding with reading the manual.



TIPS & TRICKS. This sign marks a Tips & Tricks section. Here you can find some practical advises on using the system or get a more detailed explanation of some features. Reading this section may help you in solving a particular problem or give you some ideas but is not vital for understanding VIMOS.



PREMISE. This sign marks a section, which requires you to do something before proceeding with reading the manual. Usually this is a demo program, you have to run or something similar.

File Menu item

File > Open Sub-menu item

"1.1. About" Section name. If the section is within the current manual no manual name is specified. When the section is within external manual the name of the respective manual is also included.

Ctrl+E Hot-key combination. The first part of the combination specifies which system key to use. Possible values are: Ctrl, Alt, Shift. The second part specifies the normal key to be used in the combination.

CONTENTS

1. INTRODUCTION.....	6
1.1. HOW TO START VIMOS KERNEL.....	6
1.1.1. <i>Starting VIMOS kernel on simulator.....</i>	6
1.1.2. <i>Starting VIMOS kernel on camera.....</i>	6
1.2. BASIC MODES OF VIMOS OPERATION.....	7
1.3. MAIN SYSTEM LOOP	7
1.4. IMAGE ACQUISITION	7
1.4.1. <i>“Take image” tool</i>	8
1.4.2. <i>Automatic image acquisition.....</i>	9
2. MOUSE INTERFACE	10
2.1. MOUSE DEVICES AND PROTOCOLS	10
2.2. MOUSE COMMANDS	10
2.2.1. <i>Keyboard commands.....</i>	11
2.3. REMOTE CONTROL VIA THE SIMULATOR.....	11
2.4. USING A MOUSE DIRECTLY ON THE CAMERA	12
2.5. CONNECTING OTHER EXTERNAL DEVICE	13
2.5.1. <i>VIMOS setup for external device</i>	13
2.5.2. <i>Disconnecting external device, reconnecting mouse.....</i>	14
2.5.2.1. <i>Reconnect a PC</i>	14
2.5.2.2. <i>Reconnect a stand-alone mouse.....</i>	14
3. ENTERING REGISTRATION CODE.....	15
3.1. PROTECTED USER PROGRAMS	15
4. MENUS	16
5. DIALOGS	17
6. MENU STRUCTURE.....	18
6.1. RUN MAIN MENU	18
6.2. EDIT MAIN MENU	19
6.2.1. <i>User-program menu.....</i>	20
6.2.1.1. <i>Select element group menu</i>	21
6.2.1.2. <i>Configure program-element menu</i>	22
6.2.1.3. <i>User-program browser.....</i>	22
6.2.1.4. <i>Configuration dialog</i>	23
6.2.1.4.1. <i>Point arguments</i>	24
6.2.1.4.2. <i>Angle arguments</i>	24
6.2.1.4.3. <i>Arguments, linked to arbitrary results.....</i>	24
6.2.1.4.4. <i>Arguments, which can’t be linked.....</i>	25
6.2.1.5. <i>Link dialog</i>	25
6.2.2. <i>Configuration menu</i>	26
6.2.2.1. <i>Set run-mode type dialog</i>	26
6.2.2.2. <i>Set overlay color dialog.....</i>	27
6.2.2.3. <i>Configure serial device dialog.....</i>	27
6.2.2.4. <i>Calibrate dialog</i>	28
6.2.2.5. <i>General-purpose configuration dialog</i>	29
6.2.3. <i>Statistics menu.....</i>	30
6.2.4. <i>Inspect image menu.....</i>	30
6.2.4.1. <i>Picture binarization</i>	31
6.2.4.2. <i>Copy area</i>	31
6.2.4.3. <i>Histogram in rectangle</i>	31
6.2.4.4. <i>Line profile.....</i>	32
6.2.4.5. <i>Gray matrix</i>	32
7. DIFFERENCES BETWEEN CAMERA AND SIMULATOR	33

7.1. WORKING WITH PROJECTS	33
7.2. IMAGE ACQUISITION	33
7.2.1. <i>Image acquisition on camera</i>	33
7.2.2. <i>Image acquisition on simulator</i>	33
7.3. PC/CAMERA RESOURCES	34
7.4. PLC LINES	34
7.5. I/O TOOLS	34
7.6. FLASH PACKING	34
7.7. GENERAL-PURPOSE CONFIGURATION	34
7.8. LOGO DISPLAY	34
7.9. IMAGE FILES FORMATS	35
7.10. SIMULATOR KEYBOARD COMMANDS AND HINT LINE	35
7.11. "START EXEC" TOOL	35
7.12. TOUCH-SCREEN CALIBRATION	35
7.13. PROTECTED USER-PROGRAMS	35
8. VIMOS KERNEL ERROR CODES	36

1. Introduction

VIMOS kernel is the system component, which gets pictures and interprets user programs, i.e. it does the actual image processing. Read this manual to learn how to operate with VIMOS kernel on camera and on PC simulator. With the exception of some hardware-dependent differences, described later in the manual, VIMOS kernel looks quite the same on both platforms.

The main topics, included in the manual, are:


- ◆ Mouse interface
- ◆ How to use PC mouse and stand-alone mouse
- ◆ VIMOS graphical user interface (GUI)
- ◆ VIMOS configuration (setting VIMOS defaults)
- ◆ Creation and modification of user-programs
- ◆ Usage of statistics counters
- ◆ Usage of built-in functions for image inspection

1.1. How to start VIMOS kernel

This section describes how to start VIMOS kernel on PC (the kernel simulator) and camera.

1.1.1. Starting VIMOS kernel on simulator

The recommended way to get acquainted with VIMOS kernel is to use the VIMOS simulator program.


Select **Camera > Camera simulation** or press “Camera simulation” button  to start simulation of VIMOS kernel. The kernel loops on the current user-program (if any) and is ready to execute your commands. The mouse cursor disappears and from now on you will work with VIMOS single-color graphical interface, displayed inside the simulator window. The kernel GUI commands usually do not use a mouse cursor with two exceptions:

- The move/resize/rotate “**edit**” operations supported by kernel versions 2.80 and higher on PC and TI camera use mouse cursor to drag respective buttons.
- The GUI tools in “**run**” mode need a mouse cursor, enabled by left click (PC and TI camera).

Don't mix VIMOS kernel GUI with normal Windows interface, which is active when the mouse cursor is on. Switch between the two interfaces by **Alt+M**.

1.1.2. Starting VIMOS kernel on camera

Start VIMOS kernel on camera in two modes:

1. PC mode - the camera is connected to PC. Start simulator, select **Camera > Start Camera Vimos** or press toolbar button . Use PC mouse to work with kernel GUI on camera.
2. Stand-alone mode - the camera is not connected to PC. Use AUTOEXEC file to start VIMOS kernel on camera after power reset (see “2.4. Using a mouse directly on the camera”). Use a mouse device, connected directly to camera.

The graphical interface of the VIMOS kernel is displayed in the overlay picture of the camera monitor. With a few minor exceptions, the graphical interfaces on simulator and camera are identical (see “7. Differences between camera and simulator”).



ATTENTION. VIMOS kernel may run on cameras without video-output (VCM40, VCM50), where it is not possible to use overlay GUI. Use editor and simulator to develop user-programs for such cameras.

1.2. Basic modes of VIMOS operation

The VIMOS kernel has two modes of operation: run and edit. Execute user programs in run mode. Create/modify user-programs and configure the system in edit mode. When VIMOS kernel is started, it enters automatically in run mode and executes the default user-program in endless loop. Open the run main menu by right mouse click (run mode is not stopped). The menu will be displayed after finishing the current loop cycle. Terminate user-program execution, close the menu and enter edit mode by left mouse click on “Stop run-mode” option. Open the edit main menu and access all VIMOS edit-functions by right mouse click.

1.3. Main system loop

The VIMOS kernel executes user-program repeatedly in a loop, called the “main system loop”. Each loop pass takes a new image from the video-input (automatically or by the “Take image” tool), executes the user-program, and displays results in the overlay. The tools are executed sequentially according to their order in the user-program.

Each loop pass executes the following system modules:

- The serial communication module checks for commands, received through the serial port of the camera. Usually these are mouse commands, processed by the graphical interface module.
- The graphical interface module interprets mouse commands, received by the serial communication module. In edit mode it is used to process user commands for navigation through menus/dialogs. In run mode it is used to open or close the run main menu (the only menu which can be seen in run mode).
- The image acquisition module takes automatically images, which are processed by the image-processing tools of the user-program (see “1.4. Image acquisition”).
- The user-program execution module interprets the user program. Tools are executed according to their order in the user-program. Jumps and conditional branches may be specified by GOTO and IF tools. The module performs one pass of the user-program, which begins from the first tool and ends with the last tool.
- The drawing module draws GUI elements and tool’s graphical representations in the overlay screen. Tool drawings are based on tool results, calculated by the previous module.

Both system modes (run and edit) are based on execution of modules in the main loop. Depending on operation mode and other system settings, some modules may be skipped in edit or run mode..

1.4. Image acquisition

Let us first discuss some aspects of camera hardware, which concern image acquisition in VIMOS Kernel. The camera sensor takes continuously images with specified shutter speed. Sensor images are transmitted to the following destinations:

- *Camera video-output.* The video-output image is shown on the camera monitor.
- *Camera frame buffer.* This is a DRAM buffer where sensor images are stored on request. The process of storing sensor image into the frame buffer is called **image acquisition** or **image taking**. VIMOS image-processing tools work on pixels in the frame buffer. Don't forget to take image in each system cycle, otherwise your user program will use indeterminate or invalid data in the frame buffer.

The video-output of the camera has two data sources:

- *Sensor data.* Sensor images are continuously sent to video-output and we see **live picture** on the camera monitor.
- *Frame buffer.* Frame buffer image is sent to video-output and we see **still picture** on the camera monitor.

ADSP cameras have separate paths from sensor to video-output and to frame buffer, so we may see live picture on the monitor, while sensor images are not saved in the frame buffer.

TI cameras have no such data path. Sensor data is always stored in the frame buffer and the picture shown on the monitor is read from the frame buffer. Continuous storing of sensor images into frame buffer and display of frame buffer on monitor provides the live picture on TI cameras.

Video display modes affect performance of ADSP and TI cameras:

- *Good ADSP camera performance* - when live sensor picture is shown on the monitor.
- *Bad ADSP camera performance* - when still picture is shown on the monitor.
- *Good TI camera performance* - when still picture is shown on the monitor (sensor images are not taken and stored frame buffer).
- *Bad TI camera performance* - when live picture is shown on the monitor (sensor images are continuously taken and stored into frame buffer).

There are two possibilities to take image in each pass of the main loop:

- By a "Take image" tool in the user program (recommended).
- By automatic image acquisition (built-in function of VIMOS kernel). This is done before the execution of the user program.

Both options may be used simultaneously, but you will receive better performance if you disable the second option when using the first one.

1.4.1. "Take image" tool

The "Take image" tool stores sensor image into the frame buffer. The tool has certain advantages, compared to automatic image acquisition. You may configure the tool to:

- Specify dynamically shutter speed (link shutter argument to result of a previous tool).
- Specify video-mode, which should be set after the image taking operation (live or still).
- Take image, synchronized with external trigger.

By using a "Take image" tool instead of automatic image taking, you are able to:

- Take and process several images in one user-program pass.
- Use conditional image taking and achieve better system performance.



ATTENTION. Using a "Take image" tool for image acquisition on Simulator has some peculiarities. Each time the tool is executed, it gets new image from next image source (next image file or next AVI frame for example). Shutter is not supported. Setting post-operation video-mode is not supported. Trigger image taking is possible

if you use the simulated PLC-inputs.

1.4.2. Automatic image acquisition

A system parameter called “run-mode type” controls automatic image acquisition. Enter edit mode of VIMOS kernel and open the “**Set run-mode type**” dialog from **Edit main menu > Configuration > Set run-mode type...** Click on “Run-mode” toggle to select one of the modes, described in the table below. The last two table columns specify whether the corresponding mode takes automatically images and how it affects performance of ADSP and TI cameras. Note that different modes should be used on different camera models to achieve same functionality or good performance.

Mode	Description	ADSP camera	TI camera
Live	Show live picture on the monitor. Intended for manual inspection applications.	No image taking Good performance	Image taking Bad performance
Live & Shoot	Show live picture on the monitor. Intended for image-processing applications where you don't need to see the processed picture.	Image taking Good performance	Image taking Bad performance
Shoot & Show	Show still picture on the monitor. Intended for image-processing applications where you need to see the processed picture (testing).	Image taking Bad performance	Image taking Good performance
Show	Show still picture on the monitor. Intended for image-processing applications where you take images with a “Take image” tool.	No image taking Bad performance	No image taking Good performance



ATTENTION. The automatic image acquisition, controlled by the “run-mode type” parameter, is valid for VIMOS kernel running on camera. The PC simulator of the VIMOS kernel uses images, read from image files. Open the “**Image Source**” dialog from **Camera > Image Source...** to specify image files. Refer to manual “Using the Simulator” for more details.

2. Mouse interface

Mouse is the standard I/O device, used to control the VIMOS kernel. Mouse commands are accepted by the graphical user interface (GUI) of the system. From visual point of view the GUI represents a tree of menus and dialogs. The menus are composed of lines, called menu **options**. The dialogs are composed of **controls** – buttons, toggle controls, spin controls and static text. The menu options and dialog controls usually open other menus and dialogs or execute some VIMOS functions.

Due to the embedded system environment, the mouse operation differs from what you have used to see in most DOS and Windows programs. There is no mouse cursor. One menu line or dialog control is always selected (highlighted). Use vertical mouse movements to navigate through menus/dialogs and to highlight next/previous item. Single button clicks activate (execute) the highlighted item. Double clicks are not supported.

2.1. Mouse devices and protocols

The mouse devices that can be used with VIMOS kernel on camera are:

- The PC mouse if the camera is connected to PC (see “2.3. *Remote control via the simulator*”).
- A standard serial mouse (not a PS2 mouse) connected directly to the serial port of the camera (see “2.4. *Using a mouse directly on the camera*”). In this case we speak about stand-alone VIMOS operation.

VIMOS kernel supports two common protocols for serial mice:

- ◆ Microsoft Mouse
- ◆ Mouse Systems

When started, VIMOS sets the baud rate of the serial port to 1200 and attempts to recognize a mouse protocol. The recognition is done in two stages:

1. During the initial system setup (initialization of mouse driver) VIMOS toggles the mouse power and reads the mouse response. This stage recognizes Microsoft-compatible mice.
2. During the display of the system logo, which occupies several seconds, VIMOS reads mouse data and attempts to recognize the mouse protocol. Move the mouse while the logo is displayed to enable recognition of non-Microsoft mouse.



ATTENTION. The rules described above are valid for VIMOS, running on ADSP cameras. Observe the following rules on cameras with TI processor:

1. Do not move the mouse while the logo is displayed until the message “**Searching mouse...**” appears on the camera monitor.
2. Move the mouse while the logo is displayed to enable recognition of a mouse.

The PC mouse, used via the simulator, is always recognized as a Microsoft mouse at stage 1. There is no need to move the mouse during the logo display. Use these procedures to find mouse when VIMOS has been configured for other I/O device (see “2.5. *Connecting other external device*”).

2.2. Mouse commands

Use the following mouse commands when working with the graphical interface of VIMOS kernel:

- ◆ **Left button click.** Activate (execute) the selected menu option. Change state of selected dialog toggle or spin control and press dialog buttons.
- ◆ **Right button click.** Cancel selected option and/or close current menu/dialog. Set normal or fast operation of spin controls (fast increment: ++, fast decrement: --).
- ◆ **Vertical mouse movement.** Move mouse down/up without pressed button to select (highlight) next/previous menu option, dialog control or user-program tool when the user-program browser is open.
- ◆ **Arbitrary mouse movement.** Move tool icons on the screen (used for tool configuration).



INFORMATION. Other mouse commands and/or usage of mouse cursor may be introduced in future VIMOS versions, running on cameras with TI processor.

2.2.1. Keyboard commands


When working with VIMOS Kernel on Simulator, you may supplement mouse commands by key commands. Use the cursor keys ↑, ↓, ← and → to perform fine mouse movements with small pixel step in vertical and horizontal direction.


2.3. Remote control via the simulator

This section describes how to run VIMOS kernel on camera using the simulator.

Ensure that camera is not busy, i.e. VIMOS kernel or other program is not running on the camera. This can be done by a camera power reset. Ensure there is no autoexec file in the camera flash, which automatically starts program(s) after power reset.

Ensure that VIMOS kernel is properly installed on camera. Start simulator by double-click on the simulator icon. Go to **Camera > Camera Files Functions** and open the file-transfer dialog. Set correct COM port and baud rate. Press **Find camera** button to see a list with VIMOS files, stored in the camera flash. Depending on the camera model, the list may contain different sets of files, but the file **vm** should always be present. Refer to manual "Installation Guide" for details about installation of VIMOS kernel on camera.

Select **Camera > Start Camera Vimos** or press toolbar button  to start VIMOS kernel on camera and use the PC mouse for kernel control. The simulator captures mouse events and sends mouse commands to camera. Watch the response of your commands on the camera monitor.

While controlling VIMOS via the simulator, you will not see a mouse cursor on the PC monitor. Exit VIMOS kernel and release  to return to normal mouse operation in Windows. While in VIMOS, you may temporary return to Windows interface by **Alt+M**. Mouse is released (cursor appears on the screen) and you may use it for Windows commands. Next **Alt+M** captures mouse again and re-connects PC mouse to VIMOS kernel on camera.

Similar procedure is applied when you run VIMOS kernel on simulator. Now you will see VIMOS user interface inside the simulator window. Press **Alt+M** to enable Windows mouse without exiting the simulation. Next **Alt+M** captures mouse input, hides mouse cursor and returns control to kernel GUI.

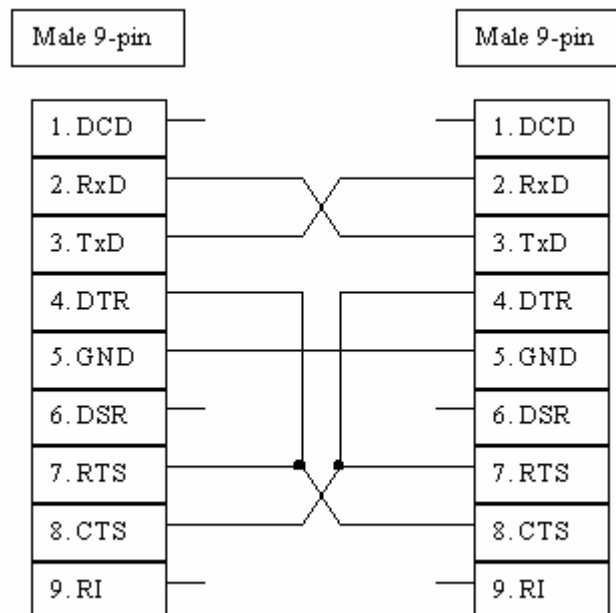
2.4. Using a mouse directly on the camera

Connect a standard serial mouse directly to camera and run VIMOS kernel without a PC. The stand-alone mouse provides the same capabilities, available when using a PC mouse via the simulator.



ATTENTION. The simulator starts the VIMOS kernel by sending the “**vm**” command to camera shell. In stand-alone mode there is no PC to send this command, so an autoexec file should start the kernel after power reset. The software package contains default autoexec file, which starts VIMOS kernel. Before you disconnect the PC, upload the file **AUTOEXEC.MSF** to camera. See manual “Using the Simulator” for details about file uploading.

Disconnect the serial cable of the camera from the PC (suppose you had a previous connection with PC). Connect the disconnected end of the cable to the cable of the serial mouse by a special transition cable. Both ends of the transition cable must have 9-pin male connectors. The connection scheme is presented below. The transition cable is symmetrical.



NOTE:

The following wires of the camera's serial cable should be connected to the 9-pin female connector. The table below shows colors of wires and V24 camera pins for ADSP cameras:

9-pin female connector	Color* of connected wire	Pin of the V24 camera connector
1. DCD		
2. RxD	Brown	2
3. TxD	White	3
4. DTR		
5. GND	Gray	5
6. DSR		
7. RTS	Green	1
8. CTS	Yellow	6
9. RI		

(*) Empty field means no connection. Leave the pink wire unconnected to the 9-pin female connector.

2.5. Connecting other external device

It is possible to connect non-mouse external device to the serial port of the camera (V&C IO-box with 8 inputs and 8 outputs for example). The device should communicate with a user-program, which contains I/O tools (**Send result**, **Receive result**). The device must send and receive data in format compatible with format of the I/O tools. Such device could be the PC when it runs a program, which observes the send/receive protocol.

2.5.1. VIMOS setup for external device

Connecting external device to camera requires setup operations before disconnecting the PC or the stand-alone mouse from camera:

- ◆ Create a user-program, which communicates with the external device by I/O tools. Send the user-program to camera flash. Start VIMOS kernel on camera and enter edit mode. Load the user program into by **Edit main menu > Load user-program from file...**
- ◆ Select **Edit main menu > Configuration > Serial device...** to open the **Configure Serial Device** dialog. Set **I/O device** toggle to **Other I/O Device**. Use **I/O Baud Rate** toggle to set baud rate, required by the external device. Click **OK** to exit the dialog.
- ◆ Select **Edit main menu > Save user-program to file...** and save current VIMOS settings into the user-program file.
- ◆ Exit VIMOS kernel on camera. The user-program, which was saved last to a file, becomes the default VIMOS program. When started, VIMOS will load and run the default user-program.
- ◆ Load AUTOEXEC.MSF into camera, which starts VIMOS kernel after power reset.


Now you've done all necessary setup and VIMOS is ready to run without a mouse. Switch off camera power, disconnect PC or stand-alone mouse and connect external device to the serial port of the camera. Switch on camera power to start VIMOS.

2.5.2. Disconnecting external device, reconnecting mouse

It is very likely to find errors in the user-program, while testing VIMOS communication with non-mouse external device. You need to gain control over VIMOS kernel and correct the user-program. Disconnect the external device and reconnect a PC or a stand-alone mouse to camera.

2.5.2.1. Reconnect a PC

Follow the instructions below to reconnect a PC:

- ◆ Switch off camera power and disconnect external device.
- ◆ Connect camera to PC serial port. You can control VIMOS kernel from simulator when the camera is not busy, i.e. you must skip starting of VIMOS kernel, done by the autoexec file. Use one of the following methods:
 - ❑ Start simulator and press  to enter the **Camera Files Functions** dialog. Specify correct COM port the camera is connected to (the default baud rate after power on is 9600). Press **"Wait boot..."** button and then switch camera on. Press **OK** to close the boot-message windows. See section "3.1.13. Wait for camera boot" in manual "Using the Simulator".
 - ❑ Start a general-purpose terminal program like HyperTerminal. Select correct COM port, configure the port with power-on camera settings (9600 bits/second, 8-bit word, 1 stop bit, no parity). While pressing the **Esc** key on the PC keyboard, switch the camera on. Release the **Esc** key when you see the logo of the camera operating system. Thus you will bypass the execution of the autoexec file, which starts VIMOS kernel. If you need, you may delete the autoexec file by the terminal command **"del autoexec"**.
- ◆ Use the simulator to start VIMOS kernel on camera (see "1.1.2. Starting VIMOS kernel on camera"). Use PC mouse to work with VIMOS kernel on camera.

2.5.2.2. Reconnect a stand-alone mouse

Follow the instructions below to reconnect a stand-alone mouse:

- ◆ Switch off the camera power and disconnect the external device.
- ◆ Connect a stand-alone mouse
- ◆ Switch on the camera. The autoexec file will start VIMOS kernel.
- ◆ When started, VIMOS always checks for a connected mouse at 1200 bits/sec. When VIMOS detects a mouse, it will ignore other I/O settings and you will be able to control VIMOS by the mouse.
- ◆ If the connected mouse is non-Microsoft, move the mouse while VIMOS logo is displayed on the monitor. Refer to section "2.1. Mouse devices and protocols" for details.



ATTENTION. To disable permanently settings for other I/O device, enter edit mode, go to **Edit main menu > Configuration > Serial device...** and open the **Configure Serial Device** dialog. Set **I/O device** to **Mouse** and **I/O Baud Rate** to 1200. The settings are saved on exit from VIMOS kernel.

3. Entering registration code

The copy-protection feature of VIMOS kernel aborts operation when unlicensed kernel files are copied from one camera to another. The registration code is provided by the program vendor in response to the serial number of the camera. In case of invalid or missing registration code the kernel returns error code **1002**.

Refer to manual "Installation Guide" and respective camera installation manuals to learn how to register VIMOS on camera. In general, there are no differences in the VIMOS registration procedures on different camera models.



ATTENTION. *If you erase camera flash and reload kernel files, you must re-enter the registration code.*

3.1. Protected user programs

The VIMOS kernel supports protected (encrypted) user-program files, which can be run on a given camera only. Such files have hidden tool structure and can be executed only. They can't be loaded and/or browsed by the user-program browser in edit mode. To run a protected user-program you should set in edit mode a default user-program file, for example by loading **up0.vm** (it doesn't matter whether the file exists or not). Exit VIMOS kernel, upload protected user-program to **up0.vm** on camera and restart VIMOS. An attempt to start protected user-program, which has not permission for this camera, results with execution of blank (empty) user-program.

The following limitations are valid when working with protected user-programs:

- You can't load protected program in edit mode by the "Load user-program from file..." option. VIMOS kernel will report "Invalid or missing user-program file !" error.
- You can't exit run mode and enter edit mode of VIMOS kernel when executing such program.
- The "Exit user-program" tool is disabled.
- You may use the "Load user-program" tool in your program to load a protected user-program from a normal user-program and vice versa. For example you may start unprotected program, which loads later a protected program. Once a protected program has been loaded, the edit mode is disabled.
- The PC simulator is not able to load and run protected programs.

Currently protected user-programs are supported on TI camera only. Instructions how to create protected user-programs can be found in the manual "Using the Simulator".

4. Menus

The graphical user interface (GUI) of the system consists of menus and dialogs. A brief GUI overview was given in section “2. *Mouse interface*”. Read next chapters to learn more about VIMOS menus and dialogs.

Each VIMOS menu has one top line, called ***title*** and one or more lines called ***options***. The current menu option is highlighted (shown in inverse color). Vertical mouse movements select (highlight) next/previous menu options. Left mouse click executes selected option. A menu option usually opens other menu, dialog or executes some VIMOS function. A menu option, which opens other menu, ends with “>” character. A menu option, which opens dialog, ends with “...” characters.

Right mouse click closes current menu. Depending on the operation context, a parent menu may be opened or not. If a menu is not opened automatically, use right click to open it. This is usually the case when the system returns to the root menu of the current mode of operation: **Edit main menu** or **Run main menu**. Right mouse click alternatively opens or closes the current main menu. Section “6. *Menu structure*” describes in details the menu tree VIMOS kernel GUI.

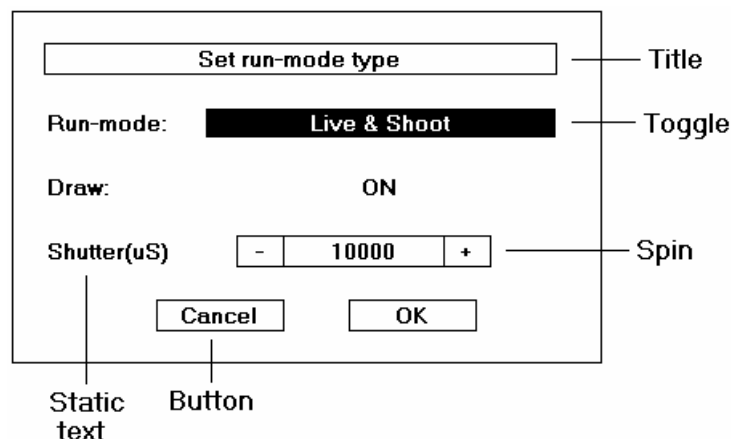
5. Dialogs

VIMOS dialogs have similar functions but more complex structure. The upper dialog area is occupied by the dialog **title**. The dialog consists of **controls**, which display information and/or allow the user to change system parameters. There is always one selected (highlighted) control in the dialog, which is shown in inverse color. Vertical mouse movements select next/previous dialog controls. Horizontal mouse movements have no effect. Mouse clicks manipulate the selected control. In contrast to the menu, the dialog is not closed by a right mouse click (see button description).

Types of dialog controls:

- ◆ **Static text.** Usually shows some system setting or a parameter state. Static text can't be selected or changed.
- ◆ **Toggle control.** Selects one of several possible states, represented by text strings. On entry, the dialog displays the current state of each toggle. Select next state by left mouse click. Select previous state by right mouse click. For example, the toggle used to show/change a line style, has two states: "Solid" and "Dashed".
- ◆ **Spin control.** Selects a numeric value in a given range. The spin control has two buttons with text: "-" and "+" and a numeric value, placed between the buttons. The spin control is selected when one of its buttons is highlighted. Left mouse clicks decrement or increment the spin value by a given step. Right mouse clicks change the spin operation to fast or normal mode. In fast mode the button titles are changed to "--" and "++" and the value is changed by greater step.
- ◆ **Button.** Executes some command when left-clicked. For example most dialogs have "OK" and "Cancel" buttons. Click "OK" button to accept changes and exit the dialog. Click "Cancel" button to ignore changes and exit the dialog. There are some exceptions of this rule. The "Cancel" button does not discard for example the linkage, made in a tool-configuration dialog by the "Link" sub-dialog.

Dialog example:



6. Menu structure

The user interface of the system is organized as a tree of menus and dialogs. Two root menus - **Run main menu** and **Edit main menu** are used to set the two basic modes of VIMOS operation - run mode and edit mode.

VIMOS executes user-programs in run mode. In run mode you can:

- ◆ Stop run mode, enter edit mode.
- ◆ Move tool icons, which are linked to mouse. You may use this feature to enter run-time data into user-program.
- ◆ Communicate with tools like PAUSE, which stops execution and waits for left click to continue.

In edit mode you can:

- ◆ Exit edit mode, start run mode.
- ◆ Configure the system.
- ◆ Create and/or change user-programs.
- ◆ Load/save user-programs to files.
- ◆ Execute built-in algorithms for image inspection.
- ◆ Read/modify statistics counters.
- ◆ Exit VIMOS kernel.

The screen of the camera monitor looks similar in both modes. The only GUI element, you can see in run mode, is the run main menu (opened or closed by right click). In edit mode you may have various menus or dialogs displayed on the screen (one at a time). A "X"-like marker marks the center of the screen.

Next sections describe menu options in details.

6.1. Run main menu

Open run main menu to stop run mode. Execute selected option by left click. Close menu by right click.

Run main menu options:

- **Stop run-mode.** Stop execution of user program and enter edit mode.

The menu is alternatively opened or closed by right clicks. Left click on "**Stop run mode**" option waits for end of the current system cycle (i.e. completes current user-program execution pass), closes the menu and enters edit mode. Right clicks do not change system mode.

6.2. Edit main menu

Use edit main menu to start run mode, to edit user programs and to configure VIMOS kernel. Execute selected option by left click. Close menu by right click.

Edit main menu options:

- ❑ **Start run-mode.** Exit edit-mode, close menu and start execution of user program.
- ❑ **Edit user-program.** Open user-program menu to create/modify current user program. See “6.2.1. *User-program menu*”.
- ❑ **Configuration.** Open configuration menu to set system configuration parameters. See “6.2.2. *Configuration menu*”.
- ❑ **Save user-program to file ...** Open the “Save user-program to file” dialog to save current user program to file.
- ❑ **Load user-program from file ...** Open the “Load user-program from file” dialog to load current user program from file.
- ❑ **Delete user-program file ...** Open the “Delete user-program file” dialog to delete user-program file.
- ❑ **New user-program ...** Discard current user program and begin creation of a new user program. This option opens a “System message” dialog, which prompts for confirmation. Click “**No**” button to reject or “**Yes**” button to confirm creation of new user program. Remember to save current user program to file before you start creation of new program.
- ❑ **Statistics.** Open the statistics menu to handle statistics counters. See “6.2.3. *Statistics menu*”.
- ❑ **Inspect image.** Open the “Inspect image menu” to execute built-in functions for image inspection. See “6.2.4. *Inspect image menu*”.
- ❑ **About ...** Open the “About” dialog, which displays system logo and version.
- ❑ **Exit.** Exit VIMOS kernel.

The edit main menu is alternatively opened or closed by right clicks. Right clicks do not change the system mode. Execute selected menu option and close the menu by left click.

Save / Load / Delete user-program options:

The save/load/delete user-program file options open file-name dialogs. Click on “File name” toggle to enter predefined name of user-program file - **up0.vm**, **up1.vm**, ..., **up29.vm**. Click “**Yes**” button to perform the file operation. Click “**Cancel**” button to cancel the file operation. Both buttons close the dialog. Reopen the edit main menu by right click.

Exit option:

If the user-program has been changed but not saved to file, the system opens a “System Message” dialog with the message “User program changed, but not saved. Exit anyway?”. Click “**No**” button to remain in VIMOS. Click “**Yes**” button to discard user-program changes and exit VIMOS. This dialog is displayed before the flash-packing dialog, described below.

Use the exit option to pack camera flash memory if necessary. A “System Message” dialog is opened on kernel exit, which displays the following message:

Free flash **uuuu/aaaa** KB. Pack it now ?

where:

uuuu is the used flash space in Kbytes

aaaa is the total available flash space in Kbytes

Click **"No"** button to exit without packing. Click **"Yes"** button to pack flash memory. The system will open a warning-message dialog. Click **"OK"** button and wait for packing-is-finished message. **DON'T** switch off the camera before the message appears on the screen. In general, a packing operation is needed when less than 64K bytes remain free ($aaaa - uuuu < 64$). The simulator does not support the flash-packing dialog.



ATTENTION. The flash packing dialog is not displayed if the "Flash auto pack" parameter is set to "On" (see "6.2.2.5. General-purpose configuration dialog").

Find more information about flash memory and flash packing in the following manuals:

- "Getting Started", section "Embedded environment"
- "Using the Simulator"
- "VIMOS Programming", section "Resources".

If VIMOS kernel is running on TI/VCRT 5.00 camera and the default file drive is changed from FD to MD, VIMOS kernel displays the message "Please wait while copying VIMOS files to MD device..." and copies kernel files from FD to MD device. Please wait while the "OK" message appears on the screen. See "6.2.2.5. General-purpose configuration dialog" for details about changing default drive. The simulator does not support this option.

6.2.1. User-program menu

Open this menu from **Edit main menu > Edit user-program** to add, delete or modify user-program tools (called also elements). Execute the selected option by left click. Close the menu by right click.

Menu options:

- ❑ **Add program element.** Open the child "Select element group" menu to add (append) an element at the end of the user program. The elements are organized in several groups. Use the child menu to select an element group and then select an element from the group. See "6.2.1.1. Select element group menu".
- ❑ **Insert program element.** Open the user-program browser and select user-program element (see "6.2.1.3. User-program browser"). A new element will be inserted into the user program in front of the selected element. Cancel the selection and return to the parent menu by right click. Accept the selection by left click, which opens the "Select element group" menu. Follow the instructions for "Add program element" option.
- ❑ **Delete program element.** Open the user-program browser and select an element, which should be deleted. Cancel the operation and return to the parent menu by right click. Delete selected element by left click.
- ❑ **Configure program element.** Open the user-program browser and select an element, which should be configured. Cancel the operation and return to parent menu by right click. Accept the selection by left click, which opens the "Configure program-element" menu. Use the menu to move or configure selected element. See "6.2.1.2. Configure program-element menu" for details.
- ❑ **Enable/disable program elements.** Open the user-program browser. Enable or disable element(s) by left click. Disabled elements are marked by exclamation mark (see "6.2.1.3. User-program browser"). They are neither executed nor displayed in run mode, but are displayed in edit mode. Close the browser and return to parent menu by right click.
- ❑ **Hide/show program element.** Open the user-program browser. Hide or show element(s) by left click. Hidden elements are marked by asterisk (see "6.2.1.3. User-program browser"). They are

executed but not displayed in run mode. Hidden elements are displayed in edit mode. Close the browser and return to parent menu by right click.

- ❑ **Exit.** Close the menu.

6.2.1.1. Select element group menu

Use this menu to add or insert element into the user-program. Select an element group and open a corresponding sub-menu from **Edit main menu > Edit user-program > Add program-element**. Select an element from the group sub-menu by left click. The element will be added in the user-program and you will see the tool's icon and the "Configure program-element" menu displayed on the screen. Move or configure the new element (see "6.2.1.2. Configure program-element menu"). Close the menu and return to parent menu by right click.





Menu options (element groups):

- ❑ **Image-processing tools.** Open sub-menu for selection of image processing tools, which read or modify image pixels. Select a tool and open move/configure menu by left click. Return to parent menu by right click.
- ❑ **Graphics & Calculations.** Open sub-menu for selection of graphics and calculation tools, which combine drawing and calculations. The tools do not work directly on images, but use results of image processing tools. Select a tool and open move/configure menu by left click. Return to parent menu by right click.
- ❑ **I/O tools.** Open sub-menu for selection of I/O tools, which communicate with external devices through PLC-lines and serial interface of camera. Select a tool and open move/configure menu by left click. Return to parent menu by right click.
- ❑ **Program flow.** Open sub-menu for selection of program flow tools. These tools make jumps and conditional branches inside the user-program by IF-ELSE-ENDIF and GOTO-LABEL constructions. Here you may count the number of user-program cycles, load another program or exit from current program. Select a tool and open move/configure menu by left click. Return to parent menu by right click.
- ❑ **Statistical tools.** Open sub-menu for selection of statistical tools. These tools operate on statistical counters - reset, increment, decrement, save/restore counters to/from file. Select a tool and open move/configure menu by left click. Return to parent menu by right click.
- ❑ **String tools.** Open sub-menu for selection of string tools for text manipulation. Text results of tools like Barcode and OCR are stored into a reserved memory buffer of VIMOS kernel. The text results may be displayed in the overlay or sent via the serial interface. Select a tool and open move/configure menu by left click. Return to parent menu by right click.
- ❑ **Point-list tools.** Open sub-menu for selection of tools, which process lists of points. Point-lists are stored into a reserved memory buffer of VIMOS kernel. Additional data could be associated with each point. Many tools store results and read arguments from the point-list buffer. Select a tool and open move/configure menu by left click. Return to parent menu by right click.
- ❑ **GUI tools.** Open sub-menu for selection of GUI tools, which are used to communicate with the user-program during its execution. GUI tools accept user input (mouse and keyboard commands) in "GUI" mode – run mode, in which a mouse cursor is visible on the screen. Use the mouse to click buttons, change spin values, check radio-buttons, etc. Other pointing device can be used as well.
- ❑ **Other tools.** Open sub-menu for selection of other tools, which don't fit into any one of the above groups – image taking, timer and pause tools, calibration, etc. Select a tool and open move/configure menu by left click. Return to parent menu by right click.

Sub-menu options coincide with tool groups from the manual "*Tool Description*". Refer to this manual for detailed description of tool operation, arguments and results.

6.2.1.2. Configure program-element menu

This menu is opened each time a tool (user-program element) is added, inserted or configured. Execute selected option by left click. Return to parent menu by right click. The menu has two options:

- ❑ **Move.** Move user-program element. Close the menu by left click. The selected element is marked on the screen. Move the element by arbitrary mouse movements without a pressed button. Return to parent menu by right click. Elements without graphical representations (icons) can't be moved. In kernel version 2.80 and higher this option is changed to **"Edit"**. It supports interactive move/resize/rotate operations by the mouse using the buttons ,  and  respectively (not available on ADSP-based cameras). The  button is used to move tools with 2 or more point arguments. Exit edit mode and return to parent menu by right mouse click.
- ❑ **Configure...** Configure user-program element. Open a corresponding configuration dialog by left click and configure the element (see "6.2.1.4. Configuration dialog"). Different element types have different configuration dialogs. Press **"OK"** or **"Cancel"** buttons to close the dialog and return to parent menu. Tools, which don't have configuration dialog, can't be configured. Refer to manual "Tool Description" for detailed information about of tool-configuration dialogs.

6.2.1.3. User-program browser

The user-program browser lists the entire user-program. It looks like a menu but it is possible to scroll up/down menu lines and to see hidden previous/next elements. The number of browser lines is set in the "General-purpose configuration" dialog, opened by the "Configuration" menu. The browser wraps from the last user-program element to the first one and vice versa.

Vertical mouse movements select consecutive elements. The graphical representation (if any) of the selected element is marked on the screen. Left click usually accepts the selected element and closes the browser. Right click usually cancels the select operation, closes the browser and returns to a parent menu or dialog.

Example:

User-Program Browser	
0. Circ.tool	[ML, (200,150), (200,250)] [P1,R1,R2,R3]
1. Textbox	[(571,158),P1] []
2. If	[<,P1,x,200,0] []

Each element is displayed in the following format:

n. name [argument 1, argument 2, ...] [result 1, result 2, ...]

where:

n = number of the element in the user program: 0, 1, 2, ...

name = name of the element.

An exclamation mark (!) is displayed before the name of a disabled element. An asterisk (*) is displayed before the name of a hidden element.

The format of the argument list depends on the link-state of each argument:

- Unlinked – an argument value is shown, for example: 5, 90, 2.74 for float, (120,110) for point, "text" for string argument, etc.
- Linked to the mouse - the string "ML" is displayed.
- Linked – the name of the result, the argument is linked to, for example: R3, P10, A2, C6, etc.

- **Steered** – the name of the linked result, followed by “+” for angles, “.x+” and “.y+” for X/Y steering of points, for example: A5+, (P2.x+,P3.y+), (120,P15.x+), etc.

The result list contains result names, for example: R7, P9, A4, etc. Refer to manual “*VIMOS Programming*” for more information about tool arguments, results and linkage process.

6.2.1.4. Configuration dialog

Select “Configure...” option from “Configure program element” menu to open a configuration dialog. Modify tool arguments or link tool arguments to results of other tools, which are executed before the current tool.

The format of the dialog depends on the type of the selected tool. The general format of the dialog is described below. This format is valid for tools with graphical representations (icons) on the screen. Some tools have configuration dialogs in different format.

Example for a configuration dialog:

Configure Rectangle Tool	
ARGUMENT:	LINK STATE:
Point:	Mouse Move Link
Angle:	Unlink Modify Link
Width:	- 60 +
Height:	- 40 +
Dash length:	- 0 +
Results:	R1 R2
Link All To Mouse Unlink All Cancel OK	

The dialog title displays the tool name. Tool arguments occupy separate screen lines. An optional text line under all argument lines displays the tool results (if any). Several buttons occupy the bottom dialog area:

- **Link All To Mouse.** Link all point arguments to the mouse.
- **Unlink All.** Unlink all linked arguments.
- **Cancel.** Close the dialog and ignore any changes with one exception - links, set by the “**Link**” button, which opens the link dialog, are not ignored.
- **OK.** Close the dialog and save all argument modifications in the user program.

Argument lines have different formats depending on argument type. Two basic groups of argument types are present in the dialog:

- Arguments, which may be linked to results (points, angles, floating-point numbers and strings).
- Style arguments, which are not linked to results (short integer, long integer, floating-point number, string). They usually specify some constant element characteristics, for example line type (dash length), size, etc.

6.2.1.4.1. Point arguments

Four controls are present for each point argument. A label shows the argument name. A static text shows the current link-state of the argument (unlinked, linked to mouse, linked to result or steered). The last 2 controls are buttons:

- Press **Move** button to close the configuration dialog and move the point by the mouse. Click a mouse button to terminate the move operation is and to reopen the dialog. Points previously linked to results are unlinked by the move operation. States of steered and linked-to-mouse points remain unchanged.
- Press **Link** button to close the configuration dialog and open the "Link" dialog. Use the "Link" dialog to change the link-state of the argument. Point arguments can be linked to point results only. After closing the "Link" dialog, the configuration dialog is reopened and the link-state text reflects the modified state of the argument. Note that links, made in "Link" dialog, are saved immediately into the user-program by the "OK" button.

6.2.1.4.2. Angle arguments

The angle arguments are shown in a way, similar to points. Four controls are present for each angle argument. A label shows the argument name. A static text shows the current link-state of the argument (unlinked, linked to result or steered). The last 2 controls are buttons:

- Press **Modify** button to close the dialog and change the angle by vertical mouse movements (down movement increases the angle, up movement decreases the angle). Click a mouse button to terminate the modify operation and reopen the configuration dialog. Angles previously linked to results are unlinked by the modify operation. States of steered angles remain unchanged.
- The **Link** button has the same function (see point arguments). Angle arguments can be linked to angle results only.

6.2.1.4.3. Arguments, linked to arbitrary results

A special group of arguments can be linked to results of arbitrary type. The argument line consists of 3 dialog controls - argument name, link-state text and **Link** button. The link-state has two options (unlinked, linked to result). The "Link" dialog contains toggle, used to select type of result, the argument is linked to. Typical tools with arguments from this group are:

Element	Argument
Text box	"Text 2" argument (string)
IF command	"Result" argument (floating-point)
ANDIF command	"Result" argument (floating-point)
ORIF command	"Result" argument (floating-point)
Tolerance tool	"Result" argument (floating-point)

6.2.1.4.4. Arguments, which can't be linked

The argument line consists of name and toggle and/or spin control(s). Toggle or spin controls specify short and long integer values. Two spin controls – mantissa (left) and exponent (right), separated by the letter 'E', specify floating-point argument values. A floating-point value is calculated according to the following formula:

$$\text{Floating-point value} = \text{mantissa} * 10 ^ \text{exponent}$$

6.2.1.5. Link dialog

Use "Link" dialog to change the link-state of an argument, further called the **selected** argument. Press argument's **Link** button in a configuration dialog to open the "Link" dialog. The functions of the dialog box controls are described below. On entry, the controls show the current link-state of the argument:

- **Link state.** Shows the link-state of the selected argument when the "Link" dialog is opened. It is not changed by the "Copy from" button.
- **Link to elem.** Shows current user-program element, the arguments and results of which can be used in the dialog (it is called the **referenced** element).
- **Select elem.** Press **Browse** button to open the user-program browser and select another referenced element.
- **Link type.** Use the toggle to set a new link-state of selected argument. The possible link states depend on the argument type:

Argument type	Possible link states
Angle	Unlink, Link to result, Angle-steer
Point	Unlink, Link to mouse, Link to result, X-steer, Y-steer
Float, string	Unlink, Link to result

- **Result.** Use this combination of a toggle and a spin control to specify a result, the selected argument is linked to. The toggle control specifies one of the following result types: **R** (float), **P** (point), **A** (angle) and **C** (counter). The result type is fixed and cannot be changed for angle or point arguments. The result type may be changed if the selected argument may be linked to an arbitrary type of result (see "6.2.1.4.3. Arguments, linked to arbitrary results"). The spin control specifies a result identifier. It has a minimum and maximum value, derived from the range of available results, which belong to the referenced element. A zero value in the spin control means that there are no results with type, specified by the "Result type" toggle (all results have identifiers greater than zero).
- **Argument.** Use the combination of toggle control and the button "Copy from" to copy other tool argument into selected argument. The toggle selects argument from the referenced element. The option is available if the selected argument is a point or an angle. An empty toggle indicates that there are no such arguments in the referenced element. Press the "Copy from" button to copy the argument. If the copied argument is not linked, its value is copied into the selected argument. If the copied argument is linked, its link-state is copied into the link-state of the selected argument. The "Link type" and "Result" controls are updated by the "Copy from" button to reflect the new argument's state. "Link state" is not updated.
- **Cancel.** Press this button to ignore changes and exit the dialog.
- **OK.** Press this button to accept changes and exit the dialog.

Notes:

1. Open the dialog twice to establish both X and Y steering for a point argument.
2. A warning message is displayed if an argument is linked to an incorrect result number.

Link dialog:

The Link dialog box is shown with the following fields and controls:

- Link state:** Unlink
- Link to elem:** 0. Coord.sys [P1] [P1,P2]
- Select elem:** Browse
- Link type:** Link to result
- Result:** P - 1 +
- Argument:** Copy from
- Buttons:** Cancel, OK

6.2.2. Configuration menu

Open this menu from **Edit main menu > Configuration** to configure VIMOS kernel. Select menu options by vertical mouse movements. Execute the selected option by left click. Close the menu by right click.

Menu options:

- ❑ **Set run mode type...** Open the “Set run-mode type” dialog (see “6.2.2.1. Set run-mode type dialog”).
- ❑ **Set overlay color...** Open the “Set overlay color” dialog (see “6.2.2.2. Set overlay color dialog”).
- ❑ **Serial device...** Open the “Configure Serial Device” dialog (see “6.2.2.3. Configure serial device dialog”).
- ❑ **Move logo.** Move the company logo by arbitrary mouse movements without pressed mouse button. The default logo position is the upper right screen corner. This option is available on camera only.
- ❑ **Calibrate...** Open the “Calibrate” dialog (see “6.2.2.4. Calibrate dialog”).
- ❑ **General-purpose...** Open the “General-purpose configuration” dialog (see “6.2.2.5. General-purpose configuration dialog”).
- ❑ **Calibrate touch-screen.** Enter calibration of touch-screen. Press the touch-screen spot, marked by small cross in the upper left screen corner. Then press the bottom right touch-screen spot. The system returns to configuration menu. Right mouse click aborts calibration and returns to parent menu. Calibration parameters are saved into system initialization file on exit, so you don’t need to make calibration on each kernel start.

Note: Touch-screen calibration is supported on TI cameras only.

6.2.2.1. Set run-mode type dialog

This dialog specifies run-mode parameters. The dialog controls are:

- ☐ **Run-mode** toggle
- ☐ **Draw** toggle
- ☐ **Shutter(uS)** spin

The **Run-mode** toggle specifies the mode, used by the camera to get and store pictures into video-memory. The toggle states are:

- ☐ **Live.** Show live picture on the monitor. ADSP camera - no image taking, good performance. TI camera - image taking, bad performance. Intended for manual inspection applications.
- ☐ **Live & Shoot.** Show live picture on the monitor. ADSP camera - image taking, good performance. TI camera - image taking, bad performance. Intended for image-processing applications where you don't need to see the processed image.
- ☐ **Shoot & Show.** Show still picture on the monitor. ADSP camera - image taking, bad performance. TI camera - image taking, good performance. Intended for image-processing applications where you need to see the processed image (testing).
- ☐ **Show.** Show still picture on the monitor. ADSP camera – no image taking, bad performance. TI camera – no image taking, good performance. Intended for image-processing applications where you take images with a “*Take image*” tool.

See section “1.4. *Image acquisition*” for details.

The **Draw** toggle enables or disables drawings in run mode. The toggle states are:

- ☐ **ON.** Redraw tools in each user-program pass (default). Use this option while designing and testing a new application.
- ☐ **OFF.** Do not redraw tools in each user-program pass. Use this option to improve performance during production. This option disables also the anti-flicker delay, set by the “General purpose configuration” dialog.

The **Shutter** spin control specifies shutter time in microseconds. VIMSO Kernel on Simulator does not support shutter speed.

6.2.2.2. Set overlay color dialog

Change the RGB color components of the overlay color.

6.2.2.3. Configure serial device dialog

This dialog configures a serial device, attached to camera. The dialog controls are:

- ☐ **Mouse type** toggle. Select a mouse model.
- ☐ **Vertical step** spin. Specify the number of vertical pixels, needed by the mouse to select a next or previous menu line, dialog-box control or user-program tool (i.e. the mouse sensitivity).
- ☐ **I/O Device** toggle. Specify serial device, attached to camera. Note that selected I/O device will be used **AFTER** restart of VIMOS kernel.
- ☐ **I/O Baud Rate** toggle. Select camera baud rate for communication with serial device. Note that selected baud rate will be used **AFTER** restart of VIMOS kernel.
- ☐ **I/O Delay** spin. Specify wait time for I/O device. Some serial devices need wait time during serial I/O.

NOTE:

When started, VIMOS kernel always searches for a connected mouse (see “2.1. Mouse devices and protocols”). In case of detected mouse, the kernel changes state to “mouse device” and ignores the I/O device, specified by the dialog. If mouse is not detected, VIMOS kernel changes state to the device type and baud rate, set by the dialog.

6.2.2.4. Calibrate dialog

Use this dialog to create or modify a calibration set.

The calibration is a method, used by the system to measure real-world objects in real-world measuring units, such as mm, cm, inches, etc. The camera normally operates on pixels. To obtain the real dimensions of some object, the pixel coordinate system must be transformed into a real-world coordinate system. VIMOS uses coordinate transformation, which preserves the origin and changes the measuring units on the X and Y-axis.

To specify a correct transformation, you must measure real distances on the X and Y-axes and specify a measuring unit. This information is stored into a data set, called calibration set. Up to 20 calibration sets can be created and used simultaneously. By default, all calibration sets are initialized 1:1 (no coordinate transformation).

Use the “Select Calibration Set” tool to activate a calibration set. All user-program tools, which follow this tool, use the selected calibration set and calculate results in real world units. Tools, which return distance for example, have two types of results: in pixels and in real-world units. Both results can be displayed or processed by other tools.

The “Calibrate” dialog:

The dialog controls have the following functions:

- **Calibration set** spin. Select a calibration set (1 to 20).
- **Measuring unit** toggle. Select measuring unit - pixels, mm, cm, inch and feet.
- **Copy** button. Copy the set, selected by the “**from set**” toggle into current set.
- **Set 1:1** button. Initialize current set to 1:1 (no calibration, one unit is one pixel).
- **Calibrate X** button. Measure real distance on X-axis. Press the button to display a vertical line on the screen. Move the line in horizontal direction to one end of a real object with known size. Click the left mouse button to fix the line and enable a second vertical line. Move the line to the other end of the object. Fix the second line and open the “Enter real distance” dialog by left click. Use the “Real distance” spin to enter the real distance between the two vertical lines. Use the

“Measuring unit” toggle control to change current measuring unit, set by the parent dialog (if necessary). Press “OK” button to close the dialog and return to “calibrate” dialog box. The calibration of the X-axis is done.

- **Calibrate Y** button. Measure real distance on Y-axis. The vertical calibration is similar to the horizontal one. Two horizontal lines are used to measure real distance in vertical direction.
- **Use Y cal. for X.** Press this button to use the calibration on the Y-axis for X-axis. Use this option when both axes have equivalent calibrations and Y-calibration is already done.
- **Use X cal. for Y.** Press this button to use the calibration on the X-axis for Y-axis. Use this option when both axes have equivalent calibrations and X-calibration is already done.
- **Cancel.** Cancel the calibration and close the dialog.
- **OK.** Save all calibration sets into file and close the dialog.

6.2.2.5. General-purpose configuration dialog

Open this dialog to configure general-purpose system parameters:

- ☐ **UP browser lines** spin. Specify number of lines in the user-program browser.
- ☐ **Default file drive** toggle. Specify new default file drive for TI camera/VCRT 5.00: **FD** or **MD**. The change becomes active after kernel restart. When opened, the dialog shows always the current (old) default drive.
- ☐ **Logo display** toggle. Enable or disable display of system logo in the overlay (default is enabled). Use “Off” option to speed up execution of user-programs in run mode.
- ☐ **Anti-flicker delay** spin. Anti-flicker delay time (ms), used to get rid of overlay flickering on the TI camera (default: 0=don't wait). Use zero delay time to improve the system performance in run mode. Use non-zero value to remove overlay blinking – please refer to the “Using VIMOS on TI camera” manual for more information. **NOTE:** This delay is ignored when you disable the overlay drawing by **Edit main menu > Configuration > Set run-mode type : Draw=OFF**.
- ☐ **GUI group id** spin. Specify group of GUI tools, which are visible in edit mode. All GUI tools in current user-program, which have “Group id” argument equal to selected group, are displayed in edit mode. The remaining GUI tools are hidden in edit mode. Use this option to edit groups of GUI tools, which occupy same screen area, but are executed in different branches of the user-program.
- ☐ **Load string file** toggle. Select a string file, which is loaded into string buffer on system start and when you exit the dialog by pressing the OK button (if the file is available). The default string file is **sb0.vm**. Use this option in combination with “DRAM buffer clear” = Off (see next option). The system-startup string file contains usually default strings for GUI tools (TI camera only).
- ☐ **DRAM buffer clear** toggle. Disable (default) or enable clearing of point-list and string buffers when loading a new user program (both in edit mode and dynamically by the load user-program tool).
- ☐ **Flash auto pack** toggle. Enable or disable automatic flash packing after the following operations:
 - On exit from VIMOS kernel to camera operating system (both TI camera and ADSP cameras).
 - Overwriting INI file (TI camera only).
 - Overwriting existing user-program file (TI camera only).
 - Deleting user-program file (TI camera only).
 - Overwriting existing statistics file (TI camera only).
 - Overwriting existing image file (TI camera only).

When auto packing is on, the kernel does not show the flash-packing dialog on exit. On TI cameras this option is valid when current file drive is FD.

6.2.3. Statistics menu

The system can collect statistics data and perform different actions depending on the accumulated data. Statistics is collected by means of counters, which can increment or decrement upon certain conditions. For example one counter could accumulate the total number of objects that have passed on front of the camera and another counter could accumulate the defective ones.

The system supports 1023 statistics counters with names C1, C2, C3, ..., C1023. Each counter holds a 32-bit signed integer value in the range $[-2\ 147\ 483\ 648, 2\ 147\ 483\ 647]$. The number of counters may be increased in future system releases.

The counters may be saved to file or loaded from file. Different sets of statistics can be stored into different files. For example, statistics data about production of red shoes in March can be saved into the file **st1.vm**. The production is switched to black shoes in April and statistics is saved in **st2.vm**. The production of red shoes is resumed in May. Select **st1.vm** to continue accumulation of red-shoe statistics. Separate sets of statistics data for red and black shoes are maintained in this way.

The “Statistics menu”, opened from **Edit main menu > Statistics**, provides the following statistics operations:

- ❑ **Save statistics to file...** Open a dialog to save statistics to file. Select file name and press “OK” button to save current statistics data into file.
- ❑ **Load statistics from file...** Open a dialog to load statistics from file. Select file name and press “OK” button to loads statistics data from file. This operation will discard current statistics data and load all counters with data, read from file.
- ❑ **Reset statistics...** Open a confirmation dialog box. Press “Yes” button to resets all statistics counters to zero.
- ❑ **View/set statistics entry...** Open the “Set counter” dialog to view and/or change counter value. The **Counter** spin selects a counter number from 1 to 1023. The **Value** spin displays and changes value of selected counter. When **Counter** selects a new counter, the **Value** spin is updated with its value, but the value of the previously selected counter is not changed. Press “OK” button to store value of selected counter and exit the dialog.

The user-program tools usually change the statistics counters. The “save” and “load” menu options change the name of the default statistics file, where the “Save Statistics” user-program tool saves counters. The default file name is **st0.vm**.

6.2.4. Inspect image menu

Open this menu from **Edit main menu > Inspect Image** to execute built-in functions for image inspection.

Menu options:

- ❑ **Take Picture...** Get camera picture and store picture into frame buffer for further image processing. This command acts like the “Take image” tool. It opens the “Take Picture” configuration dialog, where you can set shutter value in microseconds and specify video-mode to be set after picture taking. Use the “Freeze” option to make one shot and preserve the picture for further processing. The “Link” button is ignored. Set “No” option of “Wait ext. trigger” toggle. Press “OK” button to take picture or “Cancel” button to cancel the operation.
- ❑ **Copy picture from freeze buffer.** Copy picture from freeze-buffer into frame buffer.
- ❑ **Picture binarization ...** Open the “Picture binarization” dialog. Binarize the whole picture and show results in the overlay (see “6.2.4.1. Picture binarization”).
- ❑ **Copy Area ...** Open the “Copy area” dialog. Copy image area to freeze-buffer or save image area to file (see “6.2.4.2. Copy area”).

- ❑ **Histogram in rectangle...** Open the “Histogram in rectangle” dialog. Display histogram of rectangle area with pixel values (see “6.2.4.3. *Histogram in rectangle*”).
- ❑ **Line profile ...** Open the “Line profile” dialog. Display profile of pixel values, lying on a line segment (see “6.2.4.4. *Line profile*”).
- ❑ **Gray matrix ...** Open the “Gray matrix” dialog. Display pixel values from a 16x16 matrix (see “6.2.4.5. *Gray matrix*”).



INFORMATION. VIMOS kernel has two image buffers:

- *Frame buffer.* Images read from camera sensor are stored in this buffer. Image processing tools work on this buffer.
- *Freeze buffer.* Second image buffer, where the image from frame buffer can be saved.

6.2.4.1. Picture binarization

Binarize the picture, taken by the “Take picture” option, and display in the overlay screen. The picture binarization dialog consists of **Threshold** spin and **Close** button. Change spin value to perform binarization with new threshold and see results in the overlay. Press **Close** button to close the dialog and return to parent menu.

6.2.4.2. Copy area

Select rectangle area in current image, taken by the “Take picture” option. Copy the area into freeze-buffer or save the area to file.

Order of operations:

1. Press **Select** button to select an area of interest in current image. Specify top/left and bottom/right rectangle corners by left clicks. The specified rectangle is displayed with dashed lines. Coordinates and dimensions of selected area are shown in the upper left corner of the screen.
2. Copy selected area into freeze buffer by:
 - Enter destination coordinates in the freeze buffer by **Freeze X** and **Freeze Y** spin controls.
 - Press **Freeze** button to copy image area into Freeze buffer and to close the dialog.
3. Save selected area to file by:
 - Select file name by the **File** toggle control.
 - Press **Save** button to write image area into specified file and to close the dialog.

6.2.4.3. Histogram in rectangle

Select a portion of the picture and show its gray level histogram. The “Histogram in rectangle” dialog consists of **Height** and **Width** spins, **Move**, **Histogram** and **Close** buttons.

Use **Height** and **Width** spins to determine the size of the rectangle. The minimum size is 30x30 pixels, the maximum size is 500x500.

Press **Move** button to close the dialog and select rectangle position. Move the rectangle by mouse. Return to dialog by left or right click.

Press **Histogram** button to close the dialog and to display the gray level histogram in the overlay. The histogram is shown in relative units according to the percentage of the quantity of a given gray level value. Return to dialog by left or right click.

Press **Close** button to close dialog and return to parent menu.

6.2.4.4. Line profile

Use this dialog to draw profile of pixels, lying on a line segment. The dialog consists of **Define**, **Line profile** and **Close** buttons.

Press **Define** button to define a line segment. The dialog is closed. Move start and end segment points by mouse. Fix the points by left or right click. The maximum length of the segment line is 500 pixels. The dialog is re-opened after definition of end segment point.

Press **Line profile** button to close dialog and to draw profile of pixel values in the overlay. The profile is a 2-D chart with gray level values on Y-axis) and relative segment pixel positions on X-axis. Return to dialog by left or right click.

Press **Close** button to close dialog and return to parent menu.

6.2.4.5. Gray matrix

Use this dialog to display pixel values in 16x16 matrix. The dialog consists of **Move**, **Show values** and **Close** buttons.

Press **Move** button to close dialog and select matrix position. Move the matrix (rectangle with size 16x16) by mouse. Fix rectangle position and return to dialog by left or right click.

Press **Show values** button to close dialog and display matrix pixel values in hexadecimal format. Return to dialog by left or right click.

Press **Close** button to close dialog and return to parent menu.

7. Differences between camera and simulator

Although VIMOS kernel is practically the same on camera and PC simulator, there are some differences, caused by the different hardware platforms. The section describes these differences.

7.1. Working with projects

All files, used by VIMOS kernel on camera, are stored in the flash memory. There are no folders in the camera flash. Be careful when working with several projects on camera because you are able to use limited number of user-program and data files with fixed names. Projects usually consist of user-program files and data files, such as calibration data, image data, etc.

On the other hand, the PC simulator supports arbitrary number of project folders. You can change the project folder simply by several clicks (select **File > Change Workspace...**). The kernel simulator reads files from and writes files to current project folder, thus keeping all files in one place.

Use this simulator feature to maintain camera projects. Save into separate folder all files, used or generated by the kernel simulator during test and production phases of current project. Change current camera project by:

- Save current camera project by reading project files from camera to PC project folder.
- Delete project files from camera flash.
- Pack the flash to increase free flash space.
- Copy files from new PC project folder to camera flash.

7.2. Image acquisition

7.2.1. Image acquisition on camera

VIMOS kernel on camera gets image source from camera sensor. The sensor produces so-called “live” picture, because dozens of pictures can be received and stored into memory in one second.

7.2.2. Image acquisition on simulator

The kernel simulator on PC gets image source from:

- Sequence of image files
- AVI movie file
- Video-capture card or cheap USB-camera.
- VC camera

The built-in kernel function for image acquisition, accessed from **Edit main menu > Inspect image > Take picture...**, will use the image from current image file. The run-mode parameter, which specifies mode for automatic image taking (selected from **Edit main menu > Configuration > Set run-mode type...**), is ignored.

Each time the “Take image” tool is executed it gets new image from:

- Next image file.

- Next frame from AVI file.
- Next image from PC or VC camera.

The simulator does not support shutter configuration.

7.3. PC/camera resources

The hardware resources of PC and camera differ. A user program, which runs correctly on PC simulator, may fail on camera due to insufficient memory.

Known problems on ADSP cameras due to memory limitations:

Find BLOBS tool - limitations in **connected** mode:

- The maximum number of objects that can be processed is limited to **500**. The tool returns error 9005 in case of object overflow.
- The maximum width of the work area is about **540** pixels (no limitations for the height of the work area). The tool returns error 2 (memory allocation error) in case of too large width.

7.4. PLC lines

The circle indicators, placed on the status bar of the simulator window, simulate the PLC inputs and outputs of the camera. The first (left) 4 indicators represent the PLC inputs. Click on a given indicator to set or clear corresponding PLC input (from left to right – IN0, IN1, IN2, IN3). The right 4 indicators are PLC outputs. They are set or reset by the kernel simulator during user-program execution.

7.5. I/O tools

The kernel simulator does not support I/O tools, which send and receive data through the serial port of the camera. This limitation however may be removed in future VIMOS versions.

7.6. Flash packing

The kernel simulator does not support the packing-flash option on VIMOS kernel exit (see “6.2. *Edit main menu*”).

7.7. General-purpose configuration

The kernel simulator does not support the following general-purpose configuration parameters (see “6.2.2.5. *General-purpose configuration dialog*”):

- Default file drive. This parameter is valid only for VIMOS kernel, running on TI camera with VCRT 5.00 or higher.
- Logo display. Logo is not displayed in simulation screen.
- Anti-flicker delay. This parameter is valid only for VIMOS kernel running on TI camera.
- Flash auto packing. Valid on cameras only.

7.8. Logo display

VIMOS kernel on camera displays company logo in the overlay picture of the camera monitor. The default logo position is the upper right screen corner. Change logo position by **Edit main menu >**

Configuration > Move logo (see “6.2.2. *Configuration menu*”). You can replace the default logo by your custom logo (see manual “*Using the Simulator*”, section “4.1.1. *Installing custom logo on camera*”).

This option is not available on Simulator.

7.9. Image files formats

VIMOS Kernel on Simulator reads and writes image files in standard bitmap (BMP) format. VIMOS Kernel on camera reads and writes image files in special internal bitmap-like format. Conversion between the two formats is done on the fly by the file-transfer functions **Send** and **Get** of the Simulator “Camera Files Function” dialog (you must select file type **Image file**).

7.10. Simulator keyboard commands and hint line

The Simulator supports keyboard commands, which supplement mouse commands (see “2.2.1. *Keyboard commands*”). The Simulator displays hint line at the bottom of the simulation screen. Both options are not available on camera.

7.11. “Start Exec” tool

The kernel simulator does not support the “Start Exec” tool, which executes external user plug-in module.

7.12. Touch-screen calibration

Touch-screen calibration is not supported by simulator.

7.13. Protected user-programs

The simulator does not support loading and exporting of protected user-program files.

8. VIMOS kernel error codes

This section presents the VIMOS kernel error codes. There are two types of errors:

- Fatal errors – the system aborts execution and displays the message:

System error = **code**, Module = **module_name**

where:

code = system error code

module_name = name of module which caused the error

- Non-fatal run-time errors, which do not abort kernel execution. Such errors are for example the tool errors. You can check the result of a given tool by linking it to a “text-box” tool. In case of a tool error, the text-box will show an error code instead of the tool result.

General error codes:

Error	Error message macro	Description
0	R_OK	Success
1	R_ERROR	Undefined error
2	R_NOMEMORY	Not enough memory (unable to allocate memory)
3	R_INVALIDARG	Invalid argument was passed to the function.
4	R_NOMORE	No more elements/items (used in iterations)
5	R_NOTFOUND	Not found (when searching for specific item)
6	R_SMALLBUFFER	Memory buffer passed to the function was too small.
7	R_LIMIT	An internal limit has been reached.
8	R_NOTIMPLEMENTED	The function is not implemented.
9	R_PARMISMATCH	Internal parameter mismatch
10	R_FILEERROR	File operation failed.

System load and decompression error codes:

Error	Error message macro	Description
901	QX_RC_DRAM_OVF	DRAM overflow
902	QX_RC_DMEN_OVF	DMEM allocation error
903	QX_RC_INVALID_ARG	Invalid function argument(s)
904	QX_RC_FILE_NOTFOUND	Flash file not found
905	QX_RC_QX_STACK_OVF	QX stack overflow
906	QX_RC_CAM_STACK_OVF	Camera stack overflow
907	QX_RC_UNKNOWN_MODTYPE	Unknown sub-module type
908	QX_RC_INTERNAL_ERROR	Internal QX package error
909	QX_RC_TOO_MANY_MODULES	Too many modules in the list

910	QX_RC_MOD_NOTFOUND	Module not found in QXBUF
911	QX_RC_INV_BMPFILE	Invalid format of logo BMP file
920	RC_FILE_NOTFOUND	Flash file not found (read, delete)
921	RC_FILE_NOSPACE	Flash memory overflow (write)
922	RC_FILE_PGMERR	Flash program error
923	RC_FILE_INVTYPE	Invalid file type
924	RC_FILE_INVACCESS	Invalid file access
925	RC_FILE_CLOSE_ERR	File close error
926	RC_FILE_DMEN_OVF	DMEM allocation error
930	RC_GZ_INV_METHOD	Compression method not supported
931	RC_GZ_NO_MEMORY	Out of memory (inflate)
932	RC_GZ_INV_FORMAT	Invalid compressed data: format err
933	RC_GZ_INV_CRC	Invalid compressed data: CRC error
934	RC_GZ_INV_LENGTH	Invalid compressed data: length err
935	RC_GZ_READ_ERROR	Read error (unexpected end of file)
936	RC_GZ_WRITE_ERROR	Write error (flash overflow)
940	RC_GZ_INFILE_OPENERR	Unable to open input file
941	RC_GZ_INFILE_READERR	Input file read error
942	RC_GZ_INFILE_FMTERR	Non-GZIP format of input file
943	RC_GZ_OUTFILE_OPENERR	Unable to open output file
944	RC_GZ_OUTFILE_DELEERR	Output file delete error
945	RC_GZ_OUTFILE_CLOSERR	Output file close error
946	RC_GZ_DRAM_OVF	DRAM overflow

Main module error codes:

Error	Error message macro	Description
1001	MAIN_R_INIFILE_ERR	Invalid system initialization file
1002	MAIN_R_PROT_ERR	Protection error - invalid or missing registration key file

I/O module error codes:

Error	Error message macro	Description
2001	IO_R_ERROR	Common IO error.
2002	IO_R_COM_BUSY	Opening COM port failed.
2003	IO_R_NO_SPACE_IN_BUFFER	io_can_send() fail.
2004	IO_R_NO_ENOUGH_DATA	io_can_receive() fail.
2005	IO_R_BAD_CRC	io_check_crc() fail.

2006	IO_R_NO_SPACE_IN_QUEUE	io_add_packet() fail.
2007	IO_R_PPACKET_NULL	pointer to packet is null fail.
2008	IO_R_PDATA_NULL	io_build_packet() fail.
2009	IO_R_UNKNOWN_COMMAND	io_exec() fail.
2010	IO_R_NO_PACKET	io_receive_packet() fail.
2011	IO_R_BAUD	io_baud_up() or io_baud_down() fail.
2012	IO_R_JPEGERROR	Unable to create JPEG file.
2013	IO_R_RS232QUEUE_OVF	RS232 queue-buffer overflow.
2014	IO_R_BOX_ERR	I/O box error
2015	IO_R_RECV_DATA_ERR	Received different data type
2016	IO_R_RECV_ERR	CRC error or data error
2017	IO_R_SEND_ERR	Different data type for sending
2018	IO_R_INVALID_DEVICE	Invalid I/O device referenced
2019	IO_R_TIMEOUT	Timeout during I/O operation
2020	IO_R_DEVICE_OPEN	Device already open
2021	IO_R_DEVICE_OPEN_ERR	Failed to open specified device
2022	IO_R_DEVICE_NOT_OPEN	Device not open
2023	IO_R_INVALID_MODE	Invalid I/O mode
2024	IO_R_DEVICE_BUSY	I/O device busy

Universal serial I/O error codes:

Error	Error message macro	Description
2030	IO_SER_OPEN_ERROR	Serial open error
2031	IO_SER_GET_STATE_ERROR	Serial get state error
2032	IO_SER_SET_STATE_ERROR	Serial set state error
2033	IO_SER_GET_TIMEOUT_ERROR	Serial get timeouts error
2034	IO_SER_SET_TIMEOUT_ERROR	Serial set timeouts error
2035	IO_SER_DEVICE_NOT_OPEN	Serial device not opened
2036	IO_SER_CLOSE_ERROR	Serial close error
2037	IO_SER_FLUSH_ERROR	Serial flush error
2038	IO_SER_PURGE_ERROR	Serial purge error
2039	IO_SER_INVALID_ARG	Serial invalid argument
2040	IO_SER_READ_ERROR	Serial read error
2041	IO_SER_WRITE_ERROR	Serial write error
2042	IO_SER_WRITE_TIMEOUT_ERROR	Serial write timeout
2043	IO_SER_READ_TIMEOUT_ERROR	Serial read timeout
2044	IO_SER_HANDLE_ERROR	Invalid serial handle
2045	IO_SER_RS422_FIFO_OVF	Serial FIFO overflow

Beckhoff I/O module error codes:

Error	Error message macro	Description
2100	IOB_R_NO_MEMORY	Memory allocation error
2101	IOB_R_NO_CONFIG	Configuration file not found
2102	IOB_R_INV_FORMAT	Invalid format of configuration file
2103	IOB_R_DATSIZ_OVF	Total data size overflow (>255)
2104	IOB_R_MODBUF_OVF	Module buffer overflow – number of configuration file modules > IOB_MODULE_CNT
2105	IOB_R_DATABUF_OVF	Out/in data buffer overflow
2106	IOB_R_INTERNAL_ERR	Internal error
2107	IOB_R_TIME_OUT	Response time out
2108	IOB_R_CFG_MISMATCH	Mismatch b/n configuration file and received data size
2109	IOB_R_INV_CHECKSUM	Response checksum mismatch
2110	IOB_R_INV_MODNUM	Invalid module number for data access
2111	IOB_R_MODDATA_OVF	Module data buffer overflow

TCP error codes:

Error	Error message macro	Description
-2101	TCP_STREAM_BIND_ERROR	Camera: Stream socket bind error
-2102	TCP_MFCINIT_ERROR	PC: MFC initialization error
-2103	TCP_AFXSOCKETINIT_ERROR	PC: Socket initialization error
-2104	TCP_SOCKET_CREATE_ERROR	Socket create error
-2105	TCP_LISTEN_FAIL_ERROR	Listen failed
-2106	TCP_ACCEPT_FAIL_ERROR	Accept failed
-2107	TCP_NO_CONNECTION_ERROR	No connection
-2108	TCP_INVALID_IP_ERROR	Invalid IP address
-2109	TCP_CONNECT_FAIL_ERROR	Connect failed
-2110	TCP_NO_DATA_SOCKETS_ERROR	No data sockets
-2111	TCP_SEND_BYTE_ERROR	Send byte error
-2112	TCP_RECV_BYTE_ERROR	Receive byte error
-2113	TCP_SEND_ERROR	General send error
-2114	TCP_RECV_ERROR	General receive error
-2115	TCP_SEND_BLOCK_ERROR	Send block error
-2116	TCP_RECV_BLOCK_ERROR	Receive block error
-2117	TCP_GETSOCKNAME_ERROR	Socket name error
-2118	TCP_GETPEERNAME_ERROR	Peer name error
2200	TCP_NO_MEMORY	Memory allocation error

2210	TCP_SEND_IMAGE_HDR_ERROR	Send image - header error
2211	TCP_SEND_IMAGE_DATA_ERROR	Send image - data error
2212	TCP_SEND_IMAGE_REPLY_ERROR	Send image - reply error
2213	TCP_SEND_IMAGE_TIMEOUT	Send image - wait reply timeout
2214	TCP_SEND_PTLIST_HDR_ERROR	Send point-list - header error
2215	TCP_SEND_PTLIST_DATA_ERROR	Send point-list - data error
2216	TCP_SEND_PTLIST_REPLY_ERROR	Send point-list - reply error
2217	TCP_SEND_PTLIST_TIMEOUT	Send point-list - wait reply timeout
2218	TCP_SEND_RESULT_HDR_ERROR	Send result - header error
2219	TCP_SEND_RESULT_DATA_ERROR	Send result - data error
2220	TCP_SEND_RESULT_REPLY_ERROR	Send result - reply error
2221	TCP_SEND_RESULT_TIMEOUT	Send result - wait reply timeout
2222	TCP_SEND_STRING_HDR_ERROR	Send string - header error
2223	TCP_SEND_STRING_DATA_ERROR	Send string - data error
2224	TCP_SEND_STRING_REPLY_ERROR	Send string - reply error
2225	TCP_SEND_STRING_TIMEOUT	Send string - wait reply timeout
2310	TCP_RECV_IMAGE_NO_DATA	Receive image - no data
2311	TCP_RECV_IMAGE_BUF_OVF	Receive image - buffer overflow
2312	TCP_RECV_IMAGE_TIMEOUT	Receive image - timeout
2313	TCP_RECV_PTLIST_NO_DATA	Receive point-list - no data
2314	TCP_RECV_PTLIST_BUF_OVF	Receive point-list - buffer overflow
2315	TCP_RECV_PTLIST_TIMEOUT	Receive point-list - timeout
2316	TCP_RECV_RESULT_NO_DATA	Receive result - no data
2317	TCP_RECV_RESULT_BUF_OVF	Receive result - buffer overflow
2318	TCP_RECV_RESULT_TIMEOUT	Receive result - timeout
2319	TCP_RECV_STRING_NO_DATA	Receive string - no data
2320	TCP_RECV_STRING_BUF_OVF	Receive string - buffer overflow
2321	TCP_RECV_STRING_TIMEOUT	Receive string - timeout

GIM module error codes:

Error	Error message macro	Description
3001	GIM_R_INVELTYPE	Invalid type of user-program element
3002	GIM_R_ARGTYPERR	Argument type error
3003	GIM_R_ARGNOSUPPORT	Argument type not supported
3004	GIM_R_INVLINK	Invalid link
3005	GIM_R_DCNOSUPPORT	Dialog-control not supported
3006	GIM_R_INVMASK	Invalid toggle-selection mask

3007	GIM_R_STUBNOSUPPORT	Stub function not supported
3008	GIM_R_INVELSTRING	Invalid element string
3009	GIM_R_SETLINK	Unable to set link
3010	GIM_R_ELCNT_MISMATCH	Mismatch between UPM_ELCOUNT and element count in GIM.INI
3011	GIM_R_SYSSTAT_NOSUP	System state not supported
3012	GIM_R_ARGCNT_MISMATCH	Argument count mismatch in GIM.INI, GIM_APE and header file(s).

GUI module error codes:

Error	Error message macro	Description
4001	GUI_R_ERROR	Common GUI error
4002	GUI_R_READ_DRAM	mem_read_dram() fail.
4003	GUI_R_GET_PREV_SELECT	gui_GetPrevSelect() fail.
4004	GUI_R_GET_ITEM_TYPE	gui_GetItemType() fail.
4005	GUI_R_SET_SEL_TO_ITEM	gui_SetSelectedValueToItem() fail.
4006	GUI_R_GET_SEL_FROM_ITEM	gui_GetSelectedValueFromItem() fail.
4007	GUI_R_GET_NEXT_SELECT	gui_GetNextSelect() fail.
4008	GUI_R_GET_ITEM_ID	gui_GetItemID() fail.
4009	GUI_R_GET_ITEM_DATA	gui_GetItemData() fail.
4010	GUI_R_GET_ITEM_W_ID	gui_GetItemWithID() fail.
4011	GUI_R_GET_ITEM_TEXT	gui_GetItemText() fail.
4012	GUI_R_SET_ITEM_VALUE	gui_SetItemValue() fail.
4013	GUI_R_GET_ITEM_VALUE	gui_GetItemValue() fail.
4014	GUI_R_SET_ITEM_DATA	gui_SetItemData() fail.
4015	GUI_R_BAD_MNU_DLG_NUM	Bad menu/dlg # passed to gui_SetGIData
4016	GUI_R_GET_MNU_DLG_ADDR	gui_GetMnuDlgAddress() fail.
4017	GUI_R_MISSED_ID	Can't find item with such ID.
4018	GUI_R_WRITE_DRAM	mem_write_dram() fail.
4019	GUI_R_BAD_TYPE	Bad type - spin in GetText and etc....

GUI tool errors:

Error	Error message macro	Description
4101	RC_TG_INIT_ERROR	TG package initialization error
4102	RC_TG_MALLOC_ERROR	Memory allocation error
4103	RC_TG_MOUSE_ERROR	Mouse state error (I/O error)
4104	RC_TG_GUI_STACK_OVF	GUI stack overflow
4105	RC_TG_GUI_STACK_ACC_ERR	GUI stack access error
4106	RC_TG_WIN_SIZE_ERROR	Window size error (outside screen)

4107	RC_TG_WIN_STACK_OVF	Window stack overflow
4108	RC_TG_WIN_STACK_ACC_ERR	Window stack access error
4109	RC_TG_WIN_INTERNAL_ERR	Internal window error
4110	RC_TG_WIN_GUI_TYPE_ERR	Invalid type of GUI element
4111	RC_TG_WIN_NOOPEN	No opened window
4112	RC_TG_GUI_DATA_OVF	Overflow of GUI data buffer
4113	RC_TG_GUI_DATA_ERR	Invalid type of GUI data
4114	RC_TG_INVALID_ARG	Invalid argument passed to function
4115	RC_TG_INVALID_TYPE	Invalid type
4116	RC_TG_FONT_FILE_ERR	Font file error
4117	RC_TG_INVALID_FONT	Invalid font
4118	RC_TG_MEMBUF_OVF	Memory buffer overflow
4119	RC_TG_INVALID_BOX	Invalid box
4120	RC_TG_INVALID_FOCUS	Invalid id. of focus element
4121	RC_TG_KBD_TYPE_ERROR	Stand-alone virtual keyboard type not supported
4122	RC_TG_KBD_MISMATCH_ERR	Stand-alone virtual keyboard mismatch error
4123	RC_TG_KBD_OPEN_ERR	Stand-alone virtual keyboard open error
4124	RC_TG_KEYPAD_FIFO_OVF	Keypad FIFO buffer overflow
4125	RC_TG_KEYPAD_INT_ERR	Keypad internal error
4126	RC_TG_TOUCH_FMT_ERR	Touch-screen data format error
4127	RC_TG_INVALID_KEY	Invalid/missing registration key
4201	RC_GUI_BTN_IMG_ERROR	VIMOS button image error
4202	RC_GUI_SPIN_TYPE_ERROR	Invalid type of VIMOS spin tool
4203	RC_GUI_KBD_TYPE_ERROR	VIMOS virtual keyboard type not supported
4204	RC_GUI_KBD_MISMATCH_ERR	VIMOS virtual keyboard mismatch error

UPM module error codes:

Error	Error message macro	Description
5001	UPM_R_BADARGKIND	Bad argument kind (value/reference)
5002	UPM_R_ELDELETED	Element deleted (no operations allowed)
5003	UPM_R_CORRUPTED	User-program corrupted.
5004	UPM_R_TYPEMISMATCH	Data type mismatch.
5005	UPM_R_BADVERSION	Incompatible user-program versions.
5006	UPM_R_INVALIDRES	Tool result is not valid.
5007	UPM_R_BADRESID	The program contains an invalid result id.
5008	UPM_R_BADELCODE	The program contains an element with invalid code.
5009	UPM_R_SMALLELBUF	DMEM buffer gl.upm.elbuf[] is too small.

5010	UPM_R_NOMORERES	No more space for tool results.
5011	UPM_R_SMALLDRAMBUF	No more space in DRAM buffer.
5012	UPM_R_NOENDIF	IF without ENDIF in user-program.
5013	UPM_R_IFSYNTAXERR	Syntax error in conditional execution command.
5014	UPM_R_NORESULT	Result value not set.
5015	UPM_R_BREAK_UPROG	Not an error code. Break off user-program loop.
5016	UPM_R_BAD_DATA_RECV	Bad data received. (through serial interface)
5017	UPM_R_NOT_INIT	UPM not initialized, use upm_init()
5018	UPM_R_RTBUF_OVF	UPM_RTBUF overflow (too big user program)
5019	UPM_R_RTBUF_INV_INDEX	Invalid UPM_RTBUF index
5020	UPM_R_NOSUCHPROP	Property not found
5021	UPM_R_INV_FUNCTION	Invalid tool-exec function
5022	UPM_R_GUI_TOOL_OPENERR	GUI tool open error
5023	UPM_R_GUI_TOOL_CLOSEERR	GUI tool close error
5024	UPM_R_GUI_TOOL_INVOPT	GUI tool option not supported.
5025	UPM_R_CALL_STACK_OVERFLOW	Too many nested subroutine calls. Increase the macro UPM_MAX_CALL_STACK.
5026	UPM_R_UNDEFINED_SUB	Call to undefined subroutine
5027	UPM_R_DUPLICATE_SUB	Duplicate subroutine name
5028	UPM_R_TOO_MANY_SUBS	Too many subroutines. Increase UPM_MAX_SUBROUTINES
5029	UPM_R_DECRYPT_FAIL	Decryption failed - user-program encrypted for other camera
5030	PM_R_TRIGGER_BREAK	Trigger break event occurred - picture taken without trigger
5031	UPM_R_TENABLE_ERROR	tenable() error in picture taking operation
5032	UPM_R_PARAL_PICT_ERROR	tpstart() error in parallel picture taking

DRM module error codes:

Error	Error message macro	Description
6001	DPM_R_INV_FUNCTION	Invalid tool-drawing function

Graphic error codes:

Error	Error message macro	Description
7001	GR_R_OUTSIDE	Read/write outside screen boundary
7002	GR_R_BADALIGN	Video memory access not aligned

Camera-related error codes:

Error	Error message macro	Description
8001	FLASH_R_ERASERR	Unable to erase flash PROM sector
8002	FLASH_R_PGMSEC0	Attempt to program sector 0 of flash EPROM
8003	FLASH_R_PGMERR	Flash memory location already programmed
8004	FLASH_R_VERSERR	Invalid version of flash parameters
8005	GUI_R_OPENERR	Unable to open GUI data file
8006	GUI_R_FMTERR	Illegal format of GUI data file
8007	INIT_R_MEMERR	DRAM memory allocation error
8008	LOGO_R_FMTERR	Illegal format of LOGO data file
8009	GIM_R_OPENERR	Unable to open GIM data file
8010	GIM_R_FMTERR	Illegal format of GIM data file
8011	FLASH_R_NOTFOUND	Flash file not found
8012	FLASH_R_REOPEN	Attempt to open 2nd flash file
8013	FLASH_R_NOSPACE	Flash file write error: The flash memory is full
8014	FLASH_R_NOTOPEN	Flash file no opened: write failed
8015	FLASH_R_EOF	Attempt to read flash file beyond end of file
8016	FLASH_R_OPENERR	Flash file open error
8017	FLASH_R_CLOSEERR	Flash file close error
8018	FLASH_R_READERR	Flash file read error
8019	FLASH_R_WRITEERR	Flash file write error

Calculation module error codes:

Error	Error message macro	Description
9001	CALC_R_NORESULT	Tool produced no result
9002	CALC_R_ERR_CALIBRATION_SET	Invalid format of calibration set data
9003	CALC_R_FLOAT2FIXED_OVERFLOW	Overflow in float to fixed conversion
9004	CALC_R_NUM_OVERFLOW	Numeric overflow
9005	CALC_R_BLOB_OBJ_OVERFLOW	Too many objects for Blob
9006	CALC_R_BAD_CALIB_FILE	Calibration file corrupted
9007	CALC_R_BAD_CALIB_FILE_VER	Incompatible calibration file
9008	CALC_R_BAD_ARGUMENT	Bad argument

Object detection error codes

Error	Error message macro	Description
9100	BLOB_R_NO_MEMORY	Memory allocation error
9101	BLOB_R_IMG_OVF	Image buffer overflow
9102	BLOB_R_RLCBUF_OVF	RLC buffer overflow
9103	BLOB_R_BINIMG_ERR	Invalid format of binary image

9104	BLOB_R_MBUF_OVF	Object merge buffer overflow
9105	BLOB_R_OBJLABEL_ERR	Object labeling error
9106	BLOB_R_ODB_OVF	Object database (ODB) buffer overflow
9107	BLOB_R_INTERNAL_ERR	Internal program error
9108	BLOB_R_INV_OBJ_IDX	Invalid object index
9109	BLOB_R_TRACE_ERR	Contour tracing error
9110	BLOB_R_CONTBUF_OVF	Contour buffer overflow
9111	BLOB_R_INV_FEATURE	Invalid object feature
9112	BLOB_R_DELETED_OBJ	Deleted object
9113	BLOB_R_FEATURE_NOCALC	Feature not calculated
9114	BLOB_R_INVALID_ARG	Invalid input argument
9115	BLOB_R_DRAW_ERR	Feature drawing error
9116	BLOB_R_NO_ORIENT	Insufficient orientation data
9117	BLOB_R_COPY_PROT	Copy protection error
9118	BLOB_R_INVALID_ODB	Invalid ODB buffer
9119	BLOB_R_CONTOUR_LIMIT	Contour limit exceeded

Memory and file error codes:

Error	Error message macro	Description
10001	MEM_R_READFLASHERR	Error reading file
10002	MEM_R_WRITEFLASHERR	Error writing file
10003	MEM_R_BADFLASHFILE	Error opening file. File not found
10004	MEM_R_TOOMANYFILES	Too many open files
10005	MEM_R_BADSTRPOS	Invalid position within String buffer
10006	MEM_R_BADPTLPOS	Invalid point-list position
10007	MEM_R_BADSTRBUFFILE	Invalid string buffer file
10008	MEM_R_BADSTRBUFFILEVER	Incompatible string buffer file
10009	MEM_R_BADPTLISTFILE	Invalid point-list file
10010	MEM_R_BADPTLISTFILEVER	Incompatible point-list file

Statistics error codes:

Error	Error message macro	Description
12001	TSM_R_BAD_FILE	Statistics file corrupted
12002	TSM_R_BAD_FILE_VER	Incompatible statistics file