Noordelijke Hogeschool Leeuwarden

Computer Vision

VisionLab development environment

27 August 2008

j.van.de.loosdrecht@tech.nhl.nl, jaap@vdlmv.nl

Overview

- Development environments (general)
- VisionLab
  - Overview of system
  - Image types
  - Co-ordinate system
  - Dyadic operators
  - Displaying of images
  - Using camera
  - File Drop Cam (*)
  - File Cam (*)
  - Online help
  - Script language
  - Speeding up scripts (*)
  - Adding own script as operator (*)
  - Adding own C++ operators (*)

8/28/2008 VisionLab 2

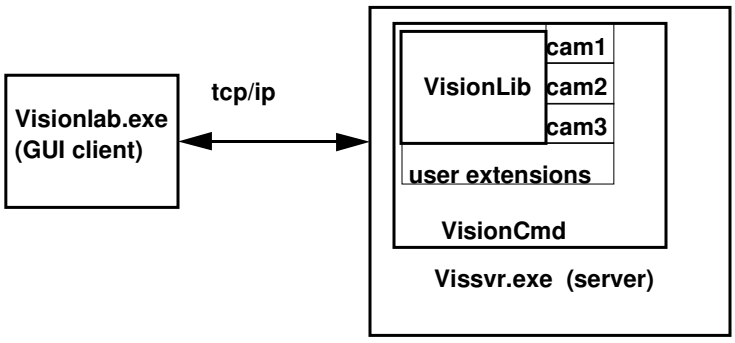
Development environments (general)

- Image acquisition
- Library with standard operators
- Experiment with operators
  - Interactive
  - Scripts
  - Graphical programming
- Add user defined operators
- Stand-alone applications

- Survey of 10 image processing packages by NHL and TNO TPD

8/28/2008 VisionLab 3

Overview of VisionLab



Demo version can be downloaded from www.vdlmv.nl (Windows and Linux)

8/28/2008 VisionLab 4

Overview of VisionLab

- **VisionLib:**
  Library of more the 200 operators for image processing, classifying with neural networks and interfaces to cameras, written in ANSI C++.
- **VisionCmd:**
  Command interpreter for scripts. The scripts can be used for easy and fast development of applications or prototypes.
- **VisionServer:**
  Shell with a network interface around VisionCmd. With VisionServer it is possible to communicate over the internet with a remote VisionCmd.
- **VisionClient:**
  Graphical user interface with which it is possible to experiment in a comfortable way with the VisionLib library and to develop and test scripts. An online help is available.
- **VisionLibNet:**
  .NET class library around VisionCmd, making it accessible by all .NET languages.
- **VisionLibDLL:**
  dll library around VisionCmd, making it accessible other languages. Delphi wrapper is available.

8/28/2008 VisionLab 5

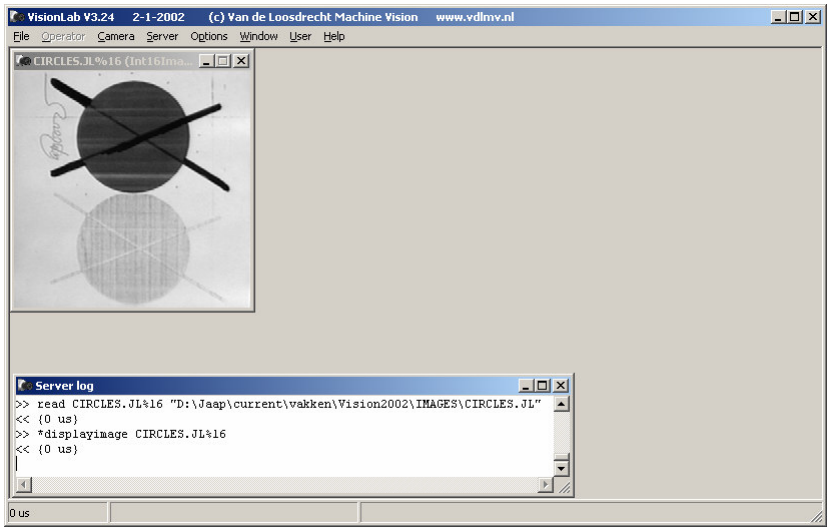
Demonstration VisionLab

- **Start-up VisionLab**
- **Show console server**
- **Explain Server Log window of client**
- **Open circles.jl and show server/client communication**
- **Demonstrate opening of *.bmp image, is automatically converted to RGB888Image, use dark_flower.bmp in images directory**

- **Source image circles.jl: Lex van de Voort van de Kleij (NHL)**

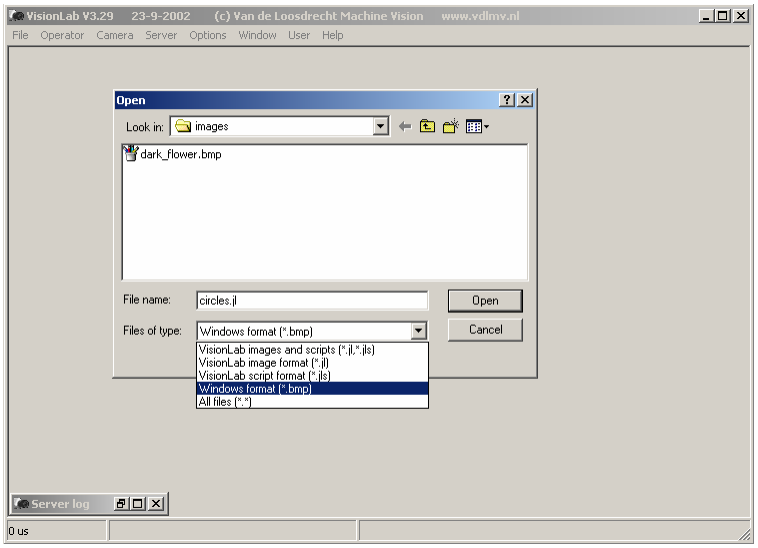
8/28/2008 VisionLab 6

Demonstration VisionLab



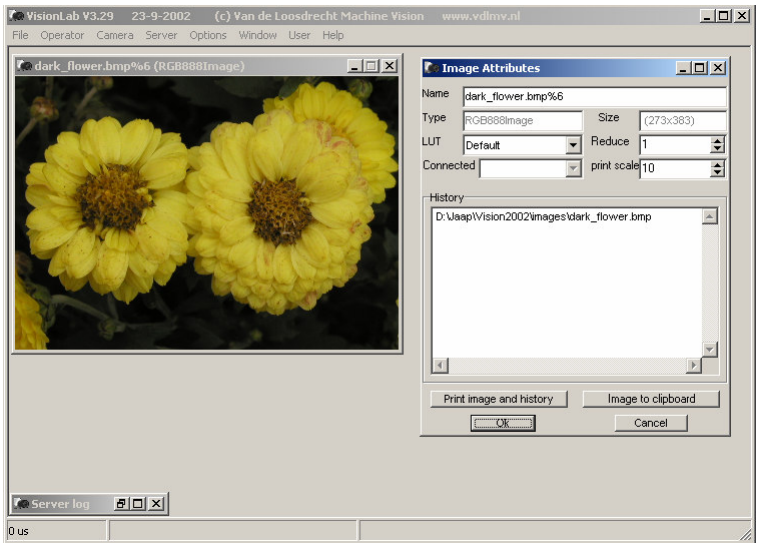
8/28/2008 VisionLab 7

Open *.bmp image



8/28/2008 VisionLab 8

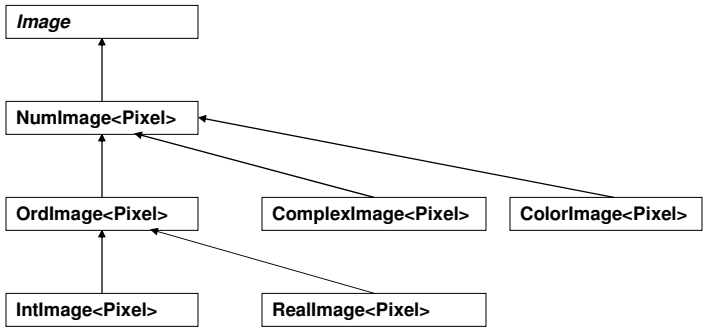
Conversion to RGB888Image



8/28/2008 VisionLab 9

Image types (*)

Class hierarchy using templates:



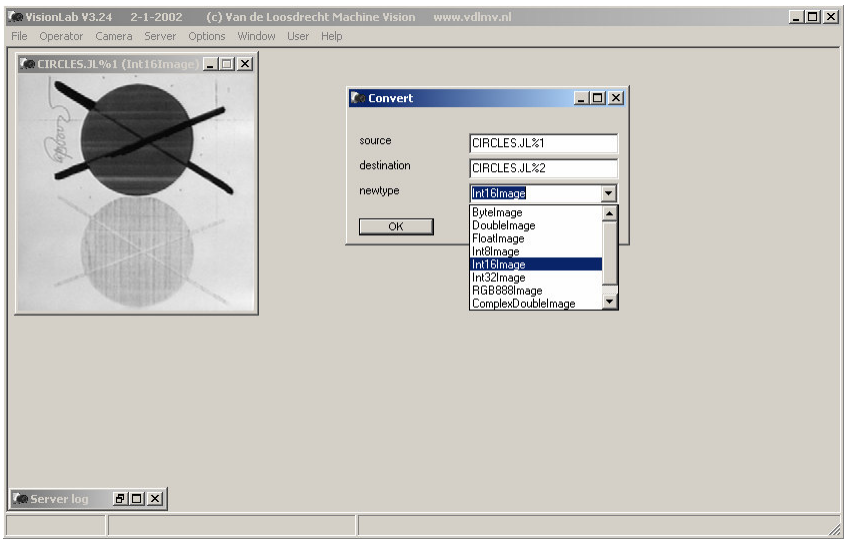
8/28/2008 VisionLab 10

Image types

- ***Image***
  - **NumImage<Pixel>:**
    - **OrdImage<Pixel>:**
      - **IntImage<Pixel>:**
        - **ByteImage**
        - **Int8Image**
        - **Int16Image**
        - **Int32Image**
      - **RealImage<Pixel>:**
        - **FloatImage**
        - **DoubleImage**
    - **ColorImage<Pixel>:**
      - **RGB888Image, RGB161616Image**
      - **HSV888Image, HSV161616Image**
      - **YUV888Image, YUV161616Image**
    - **ComplexImage<Pixel>:**
      - **ComplexFloatImage**
      - **ComplexDoubleImage**

8/28/2008 VisionLab 11

Conversion between Image Types



8/28/2008 VisionLab 12

Co-ordinate system and Dyadic operators

Co-ordinate system

- **Origin: top left corner**
- **X values increase from left to right**
- **Y values increase from top to bottom**

Dyadic operators:

- **Use select second**

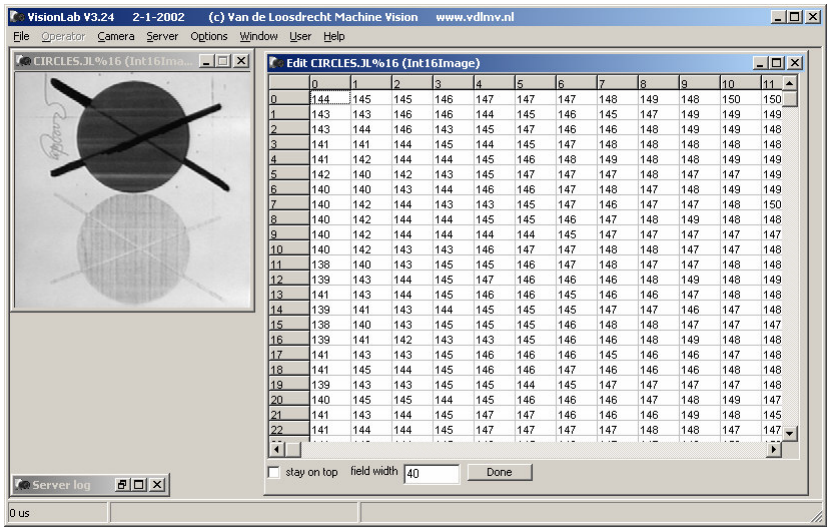
8/28/2008 VisionLab 13

Demonstration co-ordinate system and DisplayLUTs

- **Show co-ordinate system with Edit**
- **Demonstrate select second :**
  - **DisplayLUT for Int16Image = Stretch**
  - **Copy circles**
  - **Add the two images**
  - **No visible difference!**
- **Show pixel values of both images with Edit**
- **Explanation in slide about DisplayLUTs**

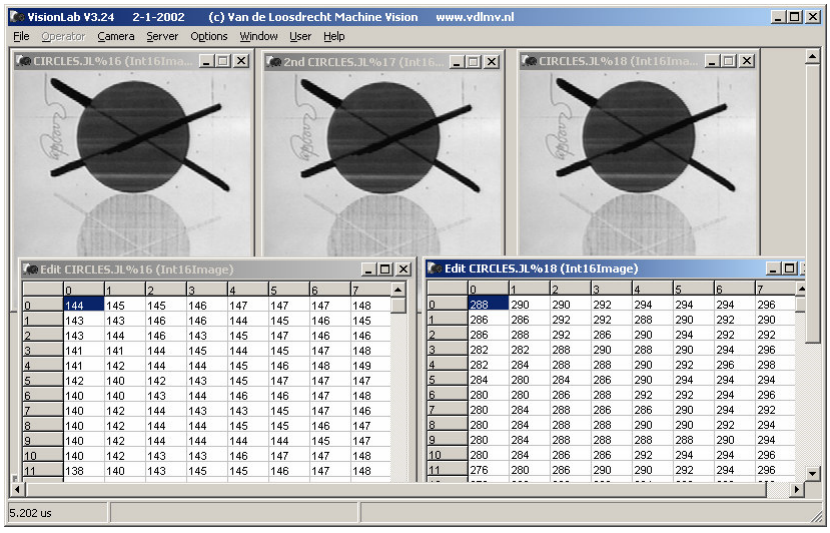
8/28/2008 VisionLab 14

Demonstration co-ordinate system



8/28/2008 VisionLab 15

Add images using select 2nd



8/28/2008 VisionLab 16

Displaying of images

Server:

- **List of images on which operations act**

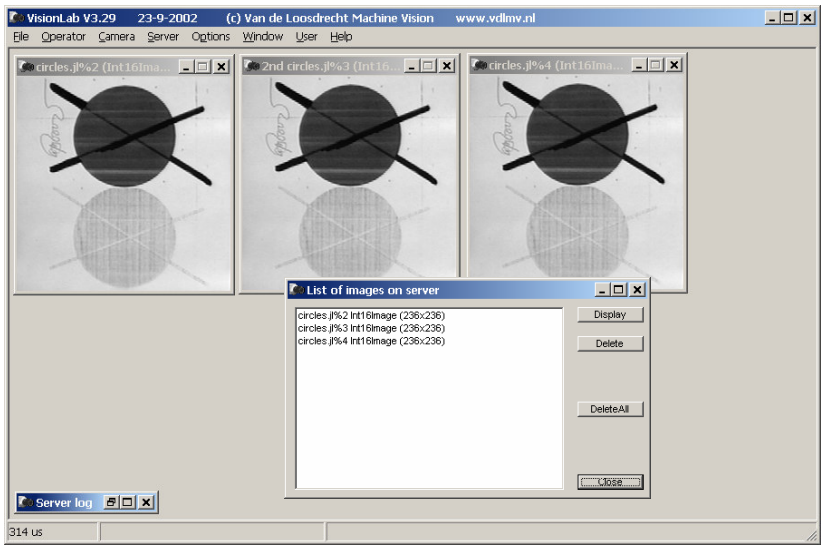
GUI client:

- **Display image with use of LookUp Table (LUT)**

Demonstration of displaying of images

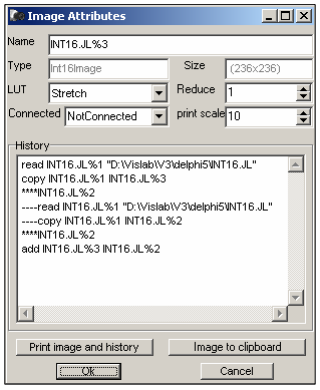
- **Show list of images on server**
- **Client is 'stupid', it knows nothing about vision operators**
- **Client can only translate commands and display images**
- **All vision intelligence is in the server**
- **Demonstrate applying different LUT's on image, see next slide**

List of images on server



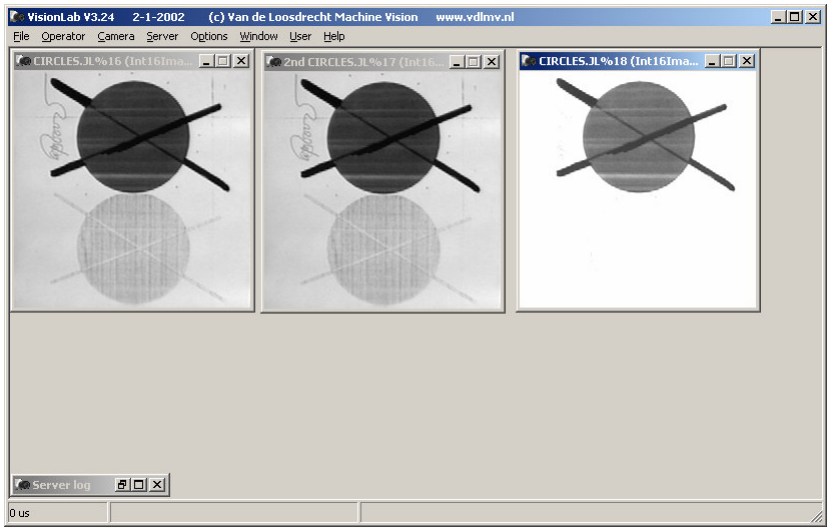
8/28/2008 VisionLab 19

Set LUT for result image to Clip



8/28/2008 VisionLab 20

See result of adding the two images



8/28/2008 VisionLab 21

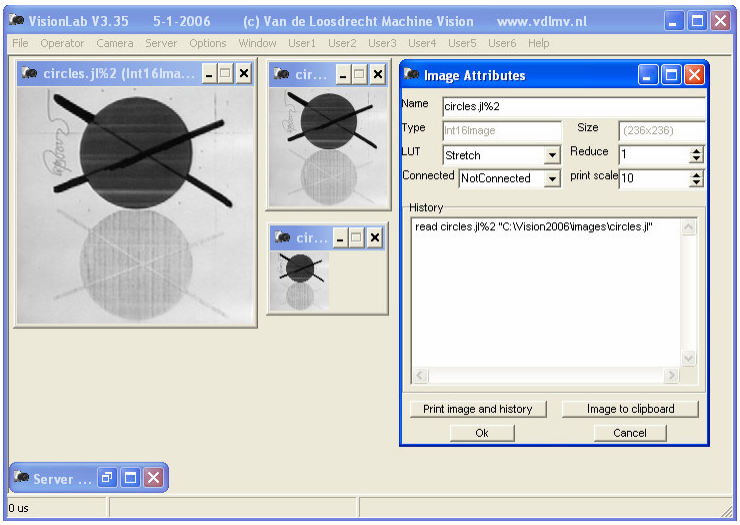
Displaying of images

DisplayLUT's (LUT = LookUpTable):

- **IntImage**
  - **Stretch**
  - **Clip**
  - **Labelled**
  - **Binary**
- **FloatImage**
  - **Linear**
  - **Logarithmic**
- **ColorImage**
  in colour depending on screen settings
- **ComplexImage**
  power spectrum displayed as FloatImage

8/28/2008 VisionLab 22

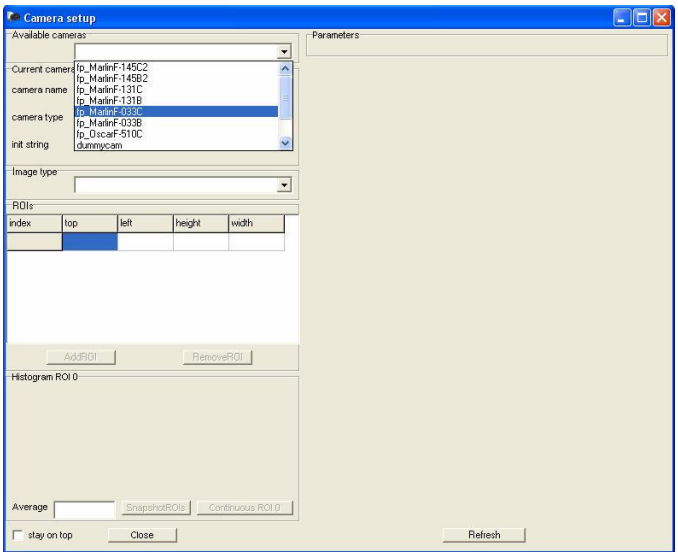
Change reduce factor with scroll wheel or + or - button



Camera setup

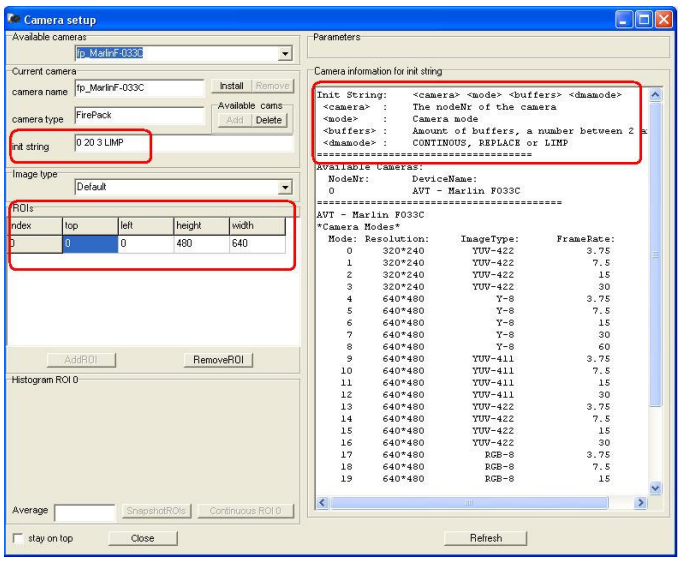
- **Multiple camera's**
- **Multiple windows (ROI's) per camera**
- **Dynamic loading of parameters**
- **Support for:**
  - **Fire wire (FirePackage and CMU)**
  - **Ellips rio (analogue video)**
  - **File cam**
  - **DirectX (beta)**
- **Open camera interface,**
  easy to add new camera's or frame grabbers
- **Camera recorder**

Select camera



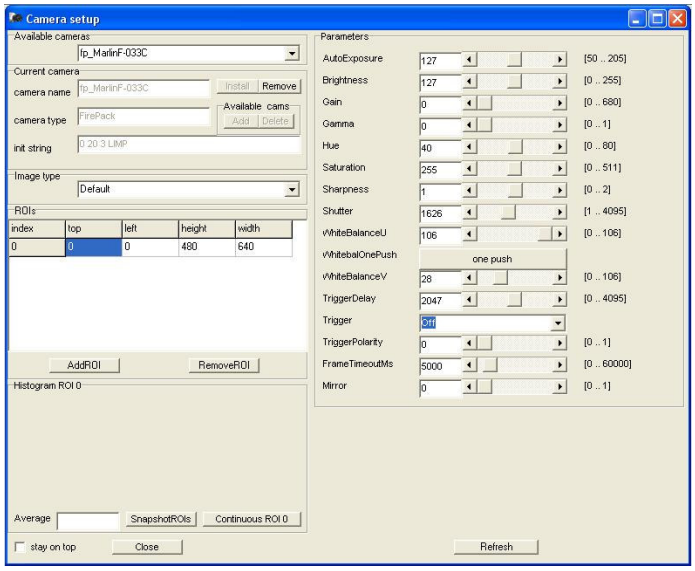
8/28/2008 VisionLab 25

Set init string, select mode



8/28/2008 VisionLab 26

Ready for taking snapshot



8/28/2008 VisionLab 27

Exercise

- **Experiment with camera setup**

8/28/2008 VisionLab 28

File Drop Camera (*)

Used as a rudimentary interface to cameras who are able to write bmp or jpg images to file with a file name and a sequence number.
The file name consists of a <File base name> + <sequence nr> + . + <File extension>. The sequence nr is padded with leading 0's if <nr of digits> is greater then 0.

After pressing the start button the image with the specified filename is searched for. If the specified file is not found, processing is postponed until the specified file is created.

If the file is found or created:
- the image is opened and displayed
- the image is selected as script image (%current image)
- the script with <Script name> is executed with the specified <Parameters> the function result of the script is assigned to the specified <Result> variable
- <Sequence nr> is incremented
- the search or wait for the next image starts

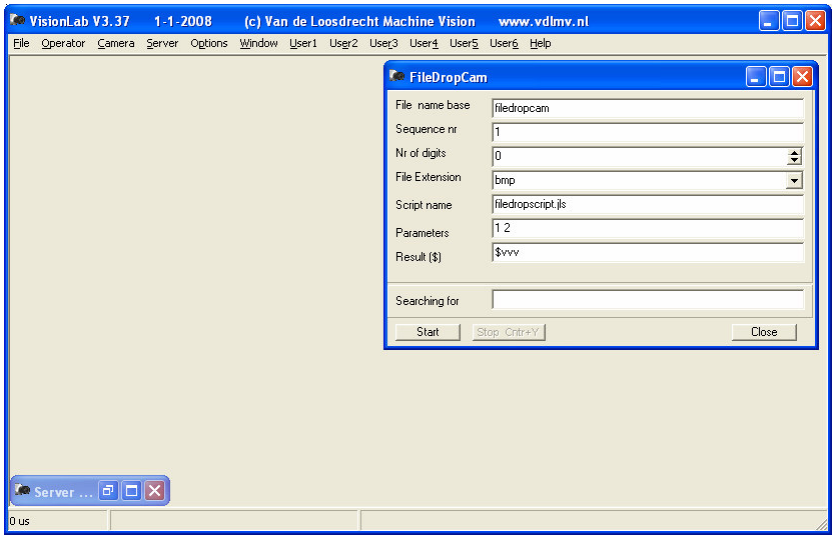
8/28/2008 VisionLab 29

Demonstration File Drop Camera (*)

- **Move file filedropcam2.bmp from directory to the desktop, leave filedropcam1.bmp in current directory**
- **Open File Drop Cam form in Camera menu**
- **Specify the parameters as in the next slide**
- **Press start button**
- **File filedropcam1.bmp is in current directory, so the image is opened and selected as %currentimage and script filedropscript.jls is executed**
- **Because file filedropcam2.bmp is in not in the specified directory, processing is postponed.**
- **Drop file filedropcam2.bmp back in specified directory, processing continues, image is opened and script is executed**

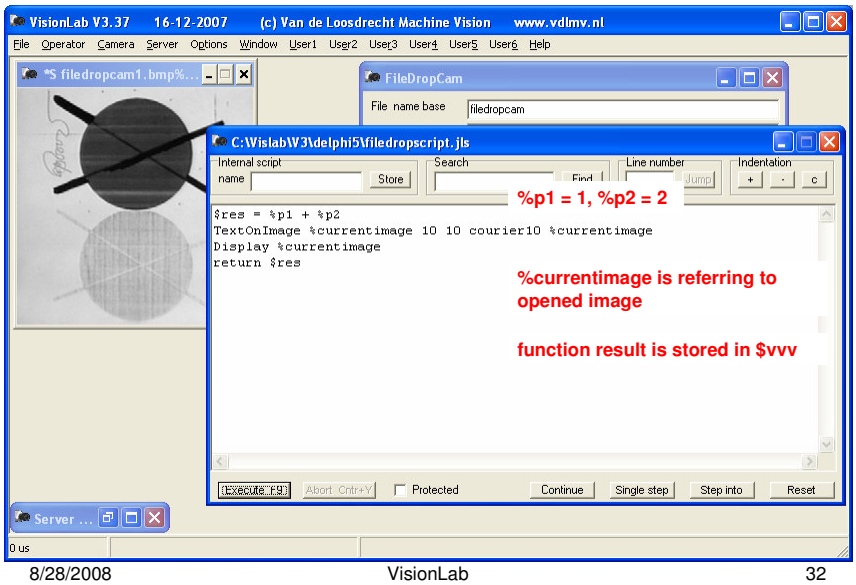
8/28/2008 VisionLab 30

File Drop Cam form in Camera menu (*)



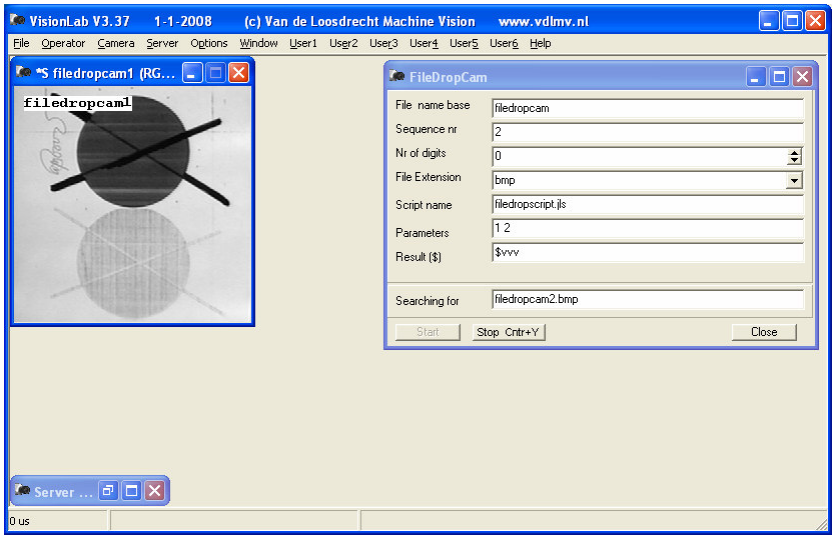
8/28/2008 VisionLab 31

Image is opened %currentimage and script is executed (*)



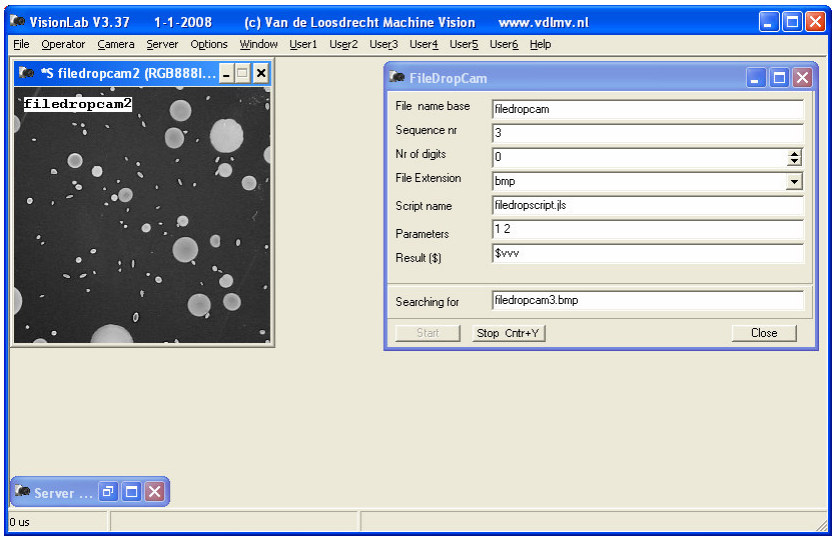
8/28/2008 VisionLab 32

Wait for drop file filedropcam2.bmp back in specified directory (*)



8/28/2008 VisionLab 33

File dropcam2.bmp is processed (*)



8/28/2008 VisionLab 34

FileCam (*)

FileCam is used to 'snapshot' images from a file from disk.

The init string should be used to specify in which mode the camera is operated. Syntax:

<imageType > <fileName>

The file is in normal text file with on each line the file name of an jl image. It is adviced to use a fully specified file names including the full direcory path.

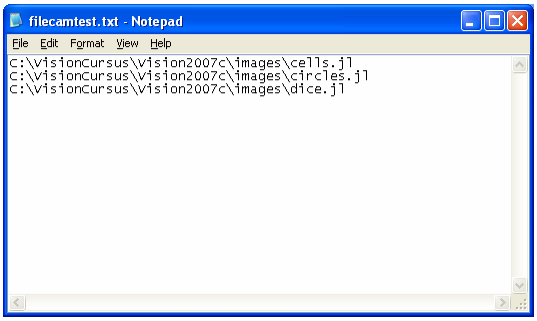
Snapshoting will start from the first image in the file.

If more snapshots are taken then images specified in the file, snapshoting will start again from the first image

Use the command LastFileCamName to retrieve the fileName of the last snapshot image.

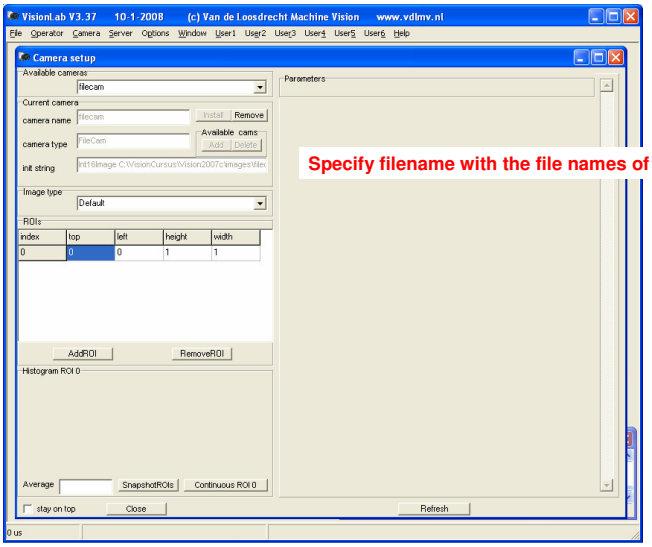
8/28/2008 VisionLab 35

Example: using a FileCam (*)



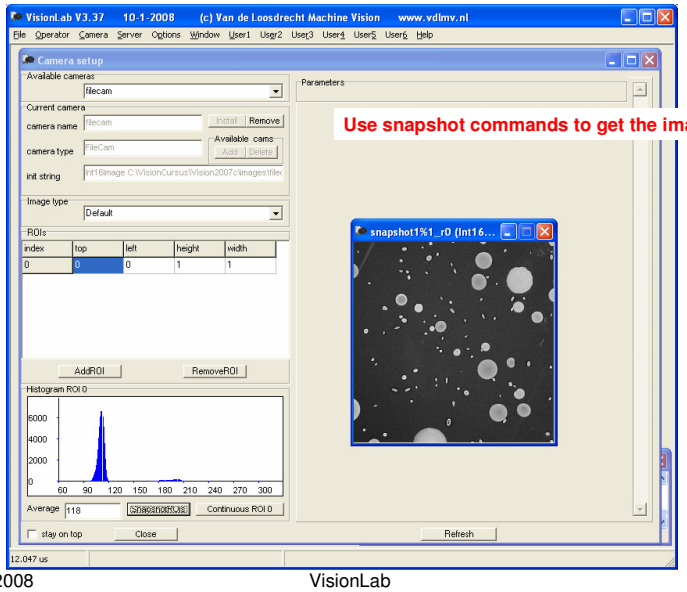
8/28/2008 VisionLab 36

Example: using a FileCam (*)



8/28/2008 VisionLab 37

Example: using a FileCam (*)



8/28/2008 VisionLab 38

Online help

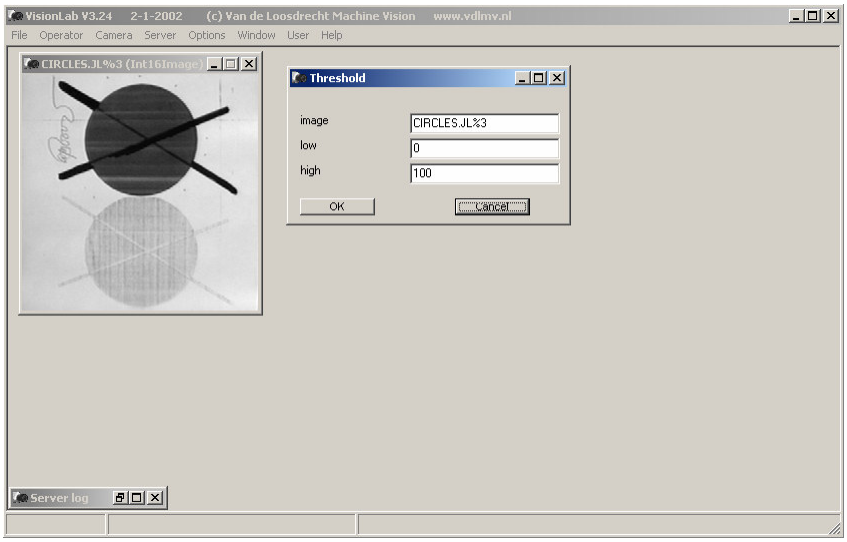
Online help is available:

- **F1**
- **Main menu**

Demonstration online help

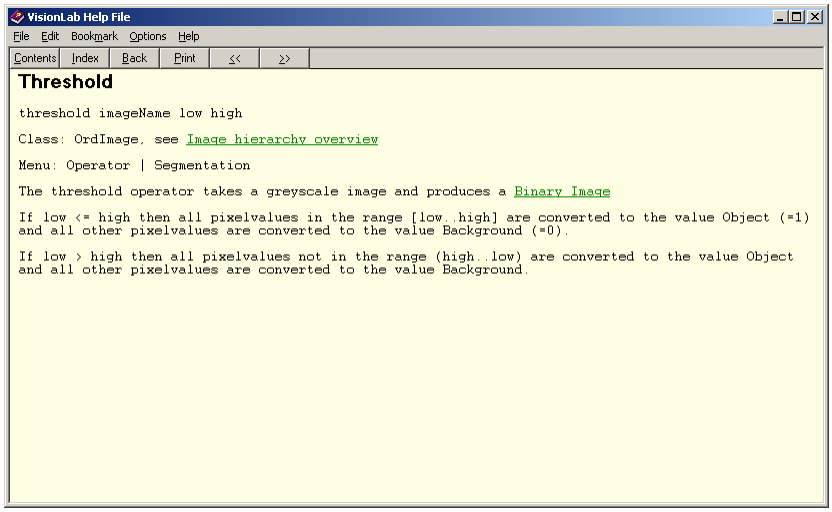
- **Search for Threshold operator**
  - **Open image circles.jl**
  - **Select threshold operator and press F1 for online help**
  - **description operator and position in operator menu structure**
  - **Select help from main menu**
  - **Select Find in VisionLab help file window and type threshold**
  - **C++ header file in System Development Kit**

Open image and select threshold operator



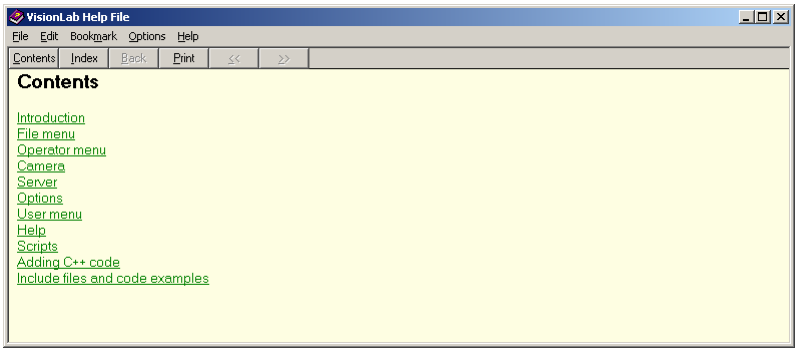
8/28/2008 VisionLab 41

Press F1 for help for Threshold (GUI)



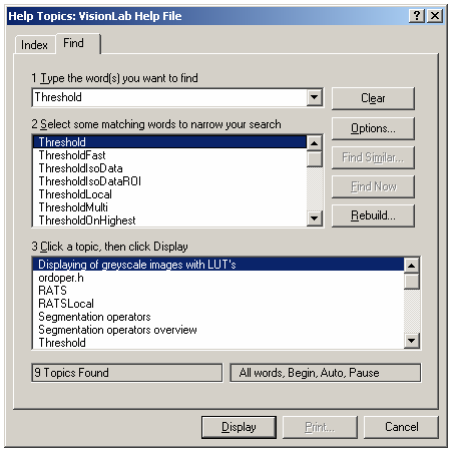
8/28/2008 VisionLab 42

Select help from main menu



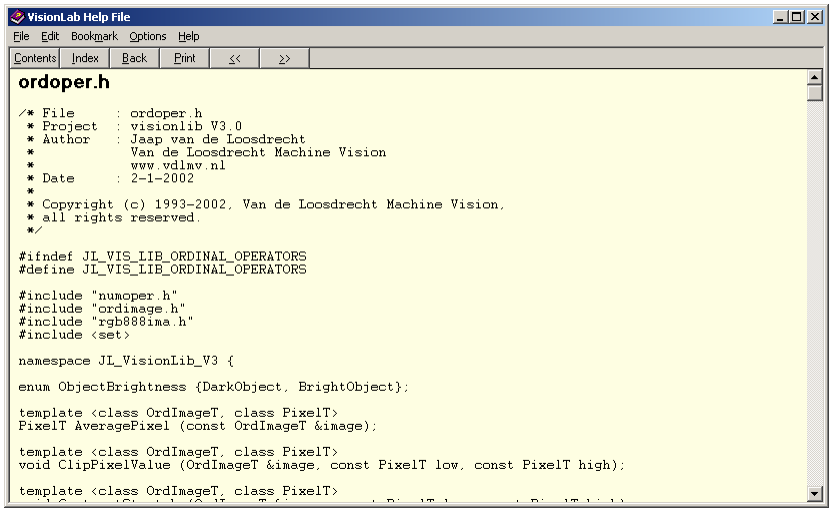
8/28/2008 VisionLab 43

Select Find in VisionLab help file window

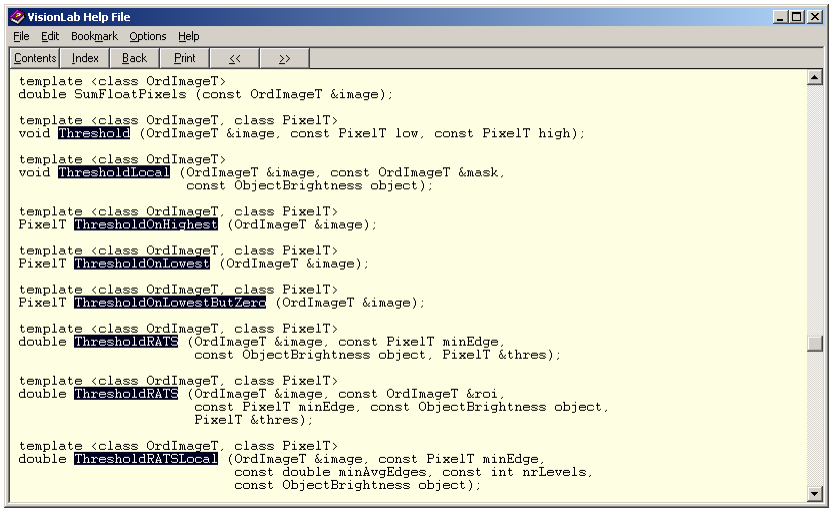


8/28/2008 VisionLab 44

Help for Threshold (SDK) (*)



Help for Threshold (SDK) (*)



Script language

Scripts can contain different kind of commands:

- **image operators**
- **variables and expressions (*)**
- **control statements (*)**
- **internal commands (*)**
- **special server commands (*)**
- **pre-processor commands (*)**

Calling scripts (procedure/function call): (*)

Scripts can be added as new operators to GUI (*)

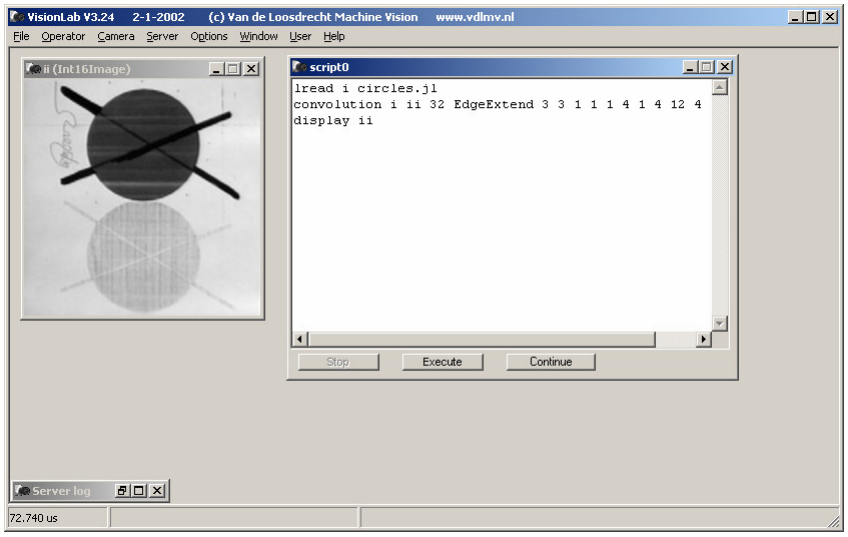
8/28/2008 VisionLab 47

Demonstration script image operators

- **Create new script, add the lines by selecting the operators from the menus (not necessary to type them):**
  - **lread i circles.jl**
  - **display i**
- **Execute script**
- **Insert with mouse threshold command, and adapt 'imageName', explain why**
- **Execute script**
- **Demonstrate convolution instead of threshold:**
  - **convolution i ii gausian3x3**
  - **display ii**
- **Change to SobelNS by doubleclick on convolution line in script**

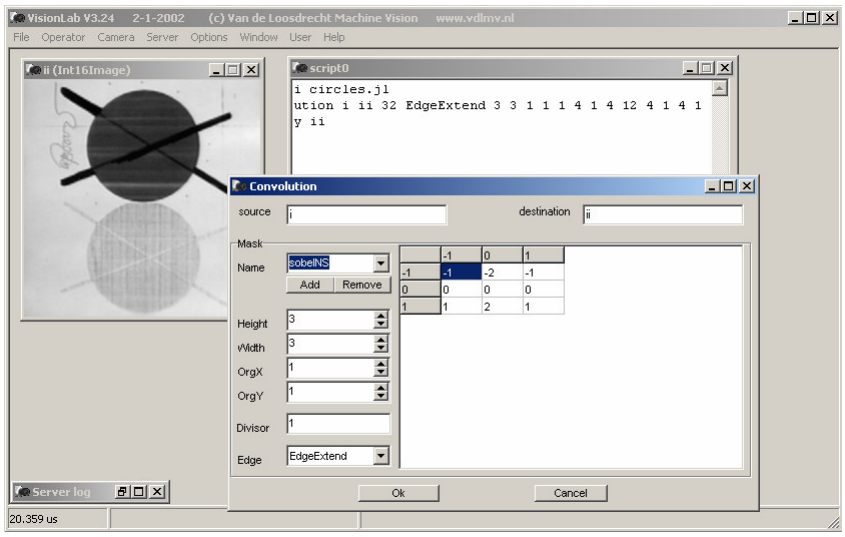
8/28/2008 VisionLab 48

Example script



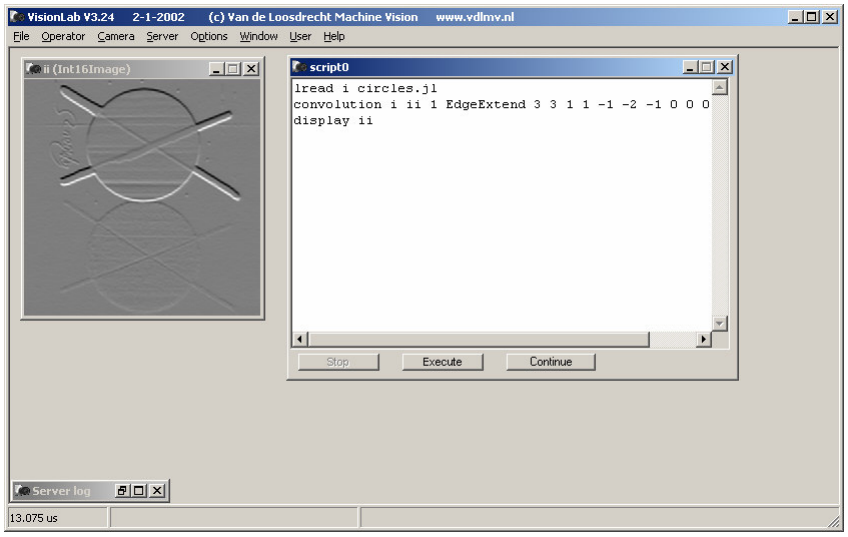
8/28/2008 VisionLab 49

Double click on line with convolution and …



8/28/2008 VisionLab 50

… select mask Sobel NS



8/28/2008 VisionLab 51

Demonstration script variables and control statements

This script calculates the average area of the selected blobs

- **Open script vb_area.jls**
- **Open Variable window (Server | Examine variables)**
- **Press 4 times Single step button, watch variable window**
- **Press Continue button: execution stops at breakpoint**
- **Comment out break (// break)**
- **Press Execute button to finish script**

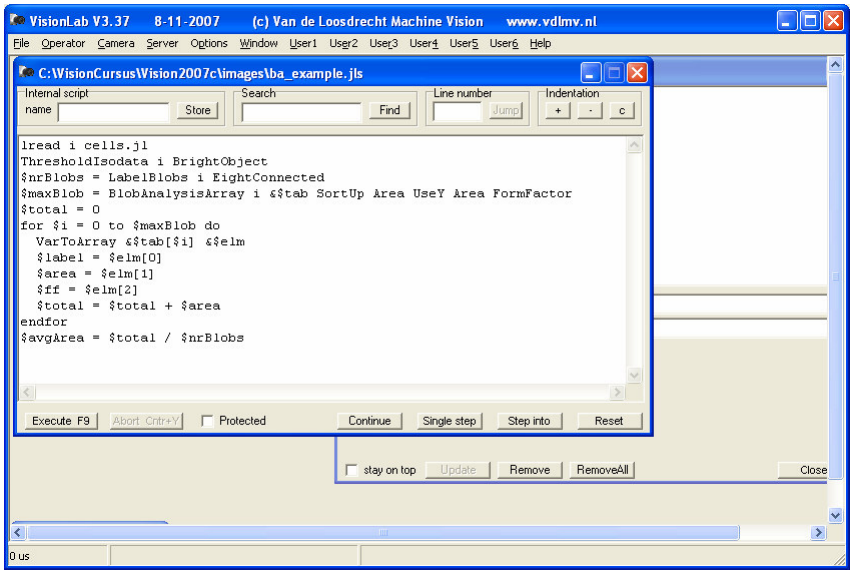
8/28/2008 VisionLab 52

Demonstration script variables and control statements

- This script calculates the average area of the selected blobs
- Open scripts ba_example.jls
- Open variable screen (Server menu | Examine variables)
- Single step through script
- Notes:
  - result is returned to an array with name tab
  - Click on array name in top window of variable screen to examine details of array
  - each element of the array contains a line with:
    <labelnr> followed with the specified measurements
  - Each line is extracted from the array tab to an array elm
  - The element with index 0 of array elm is the labelnr
  - The element with index 1 of array elm is the area
  - The element with index 2 of array elm is the formfactor

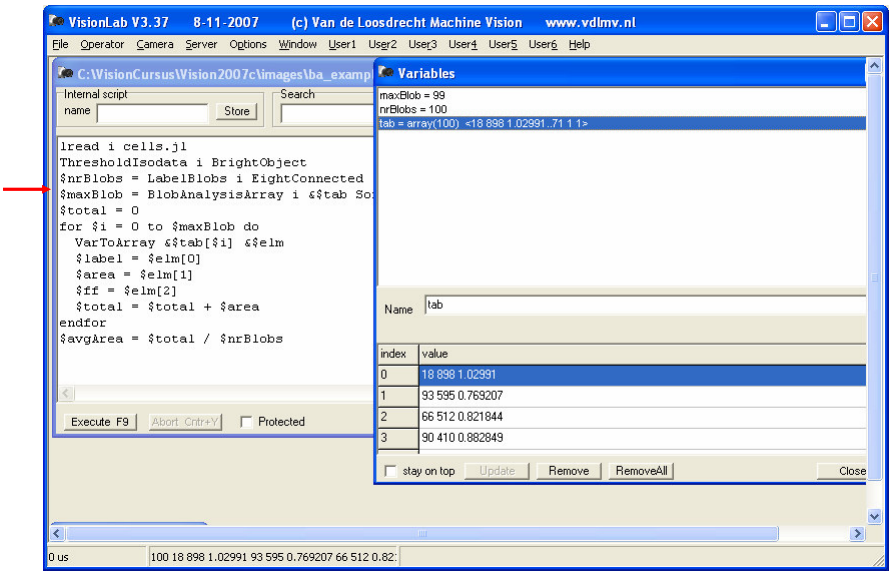
8/28/2008 VisionLab 53

Demonstration script variables and control statements

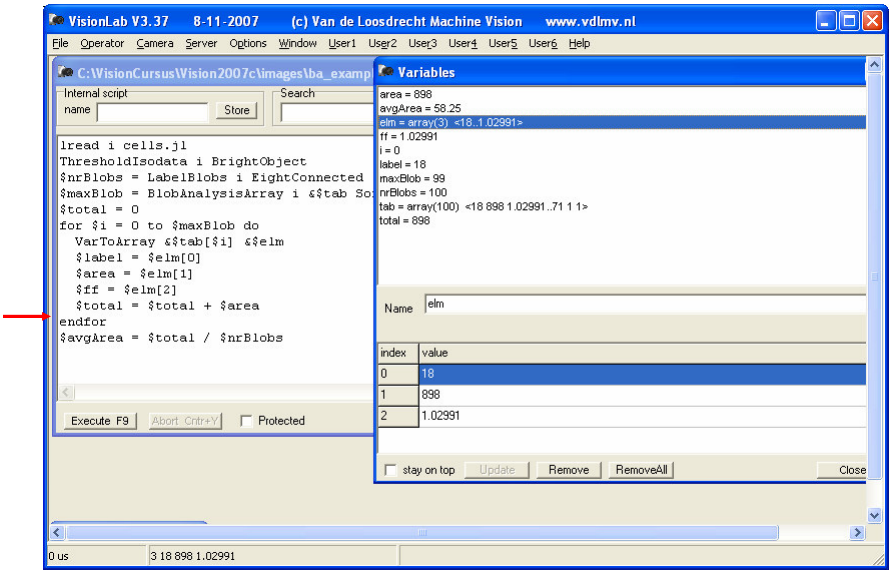


8/28/2008 VisionLab 54

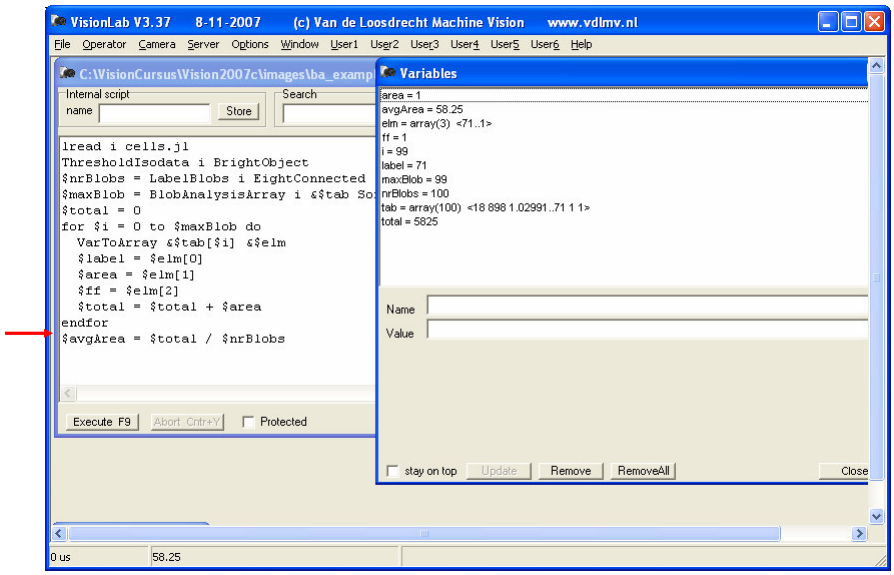
Demonstration script variables and control statements



Demonstration script variables and control statements



Demonstration script variables and control statements



8/28/2008 VisionLab 57

Exercise

- **Experiment with scripts**

8/28/2008 VisionLab 58

Script variables (*)

Variables are of type string. When necessary a conversion is applied to type integer, float or boolean.

There are two type of variables:

- **global: Scope is all scripts and outside the scripts. Lifetime is from creation to the moment it is removed explicitly by the user. The name of a global variable starts with one $ sign, example: $global**
- **local: scope and lifetime is the script in which the variable is created. The name of a local variable starts with two $ signs, example: $$local**

The value of the variables can be examined and modified with the 'Examine vars' item in the server menu.

8/28/2008 VisionLab 59

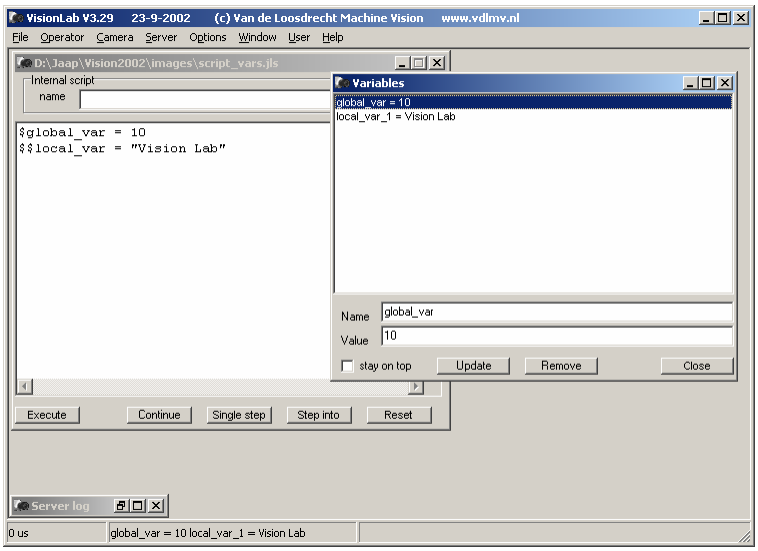
Demonstration script variables (*)

- **open script script_vars.jls**
- **execute script**
- **open window 'Examine vars' in the server menu**

- **explain: Main menu | Options | Synchronize scripts**

8/28/2008 VisionLab 60

Demonstration script variables (*)



Array's of variables (*)

Array's of variable are also possible using [and] for indexing

First index = 0
Indexes must be either constants, variables or indexed arrays
Expressions as index are not allowed
$array is a short notation for $array[0]

Example: $array[$i] = $array[$tab[0]]

Array's of variables (*)

Operators for manipulation of array's:

- **ArrayToVar &$array &$var, convert array to var**
- **CopyVar &$src &$dest, copy array from src to dest**
- **GetDimVar &$array, result is dimension of array**
- **SetDimVar &$array dimension**
- **VarToArray &$var &$array, convert var to array, each word in var is new element in array**
- **ArrayToFile <fileName> &$array**
- **ArrayFromFile <fileName> &$array**
- **LArrayToFile <fileName> &$array**
- **LArrayFromFile <fileName> &$array**

8/28/2008 VisionLab 63

Script expressions (*)

Simple forms of expressions are possible:

- **<var> = <var>**
- **<var> = <constant>**
- **<var> = <var> <operator <var>**
- **<var> = <var> <operator> <constant>**

Note there is not yet support for brackets ( ) in expressions.

The following operators are supported: + - * / mod and or ! == != < <= > >= .

8/28/2008 VisionLab 64

Script expressions (*)

**Arithmetic and Boolean expressions are possible,
use brackets ( ) for priorities in Boolean expressions.**

**Note: all operators and must separated by spaces.
Examples:
$r = ($x + $v) * 4
$b = ! ($b1 and (bt[$i] or $b2))**

It is not possible to call a function in expressions.

**Prohibit evaluation by using “”, example:
$cg = “(10,15)”**

8/28/2008 VisionLab 65

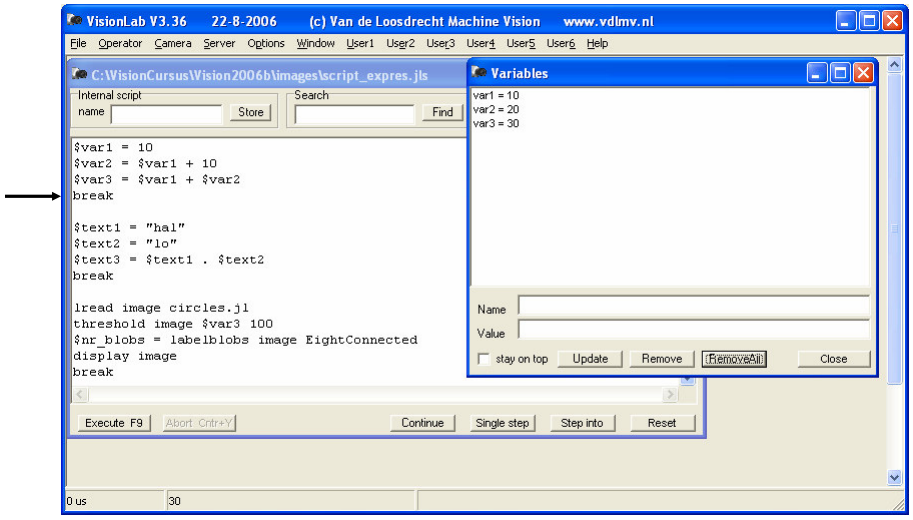
Demonstration script expressions (*)

- **open script script_expres.jls**
- **execute script**
- **open window 'Examine vars' in the server menu**
- **click continue script**
- **click continue script**

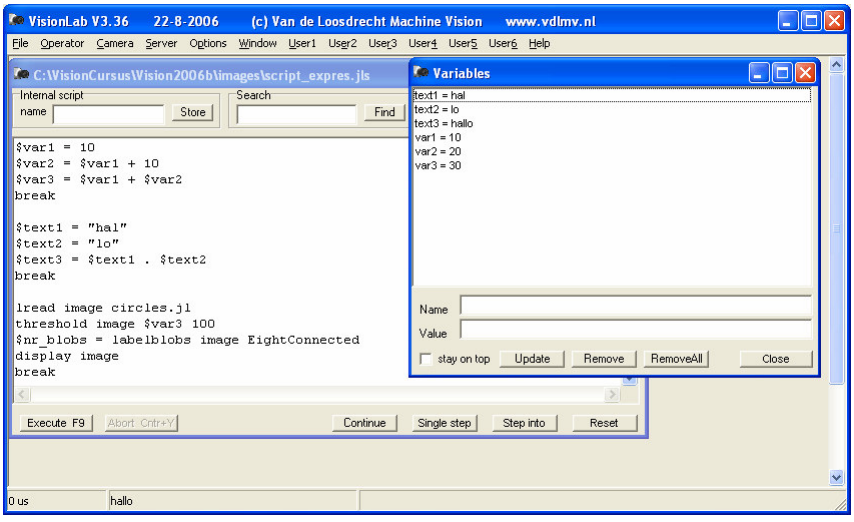
- **explain break(points), reset and single stepping scripts**

8/28/2008 VisionLab 66

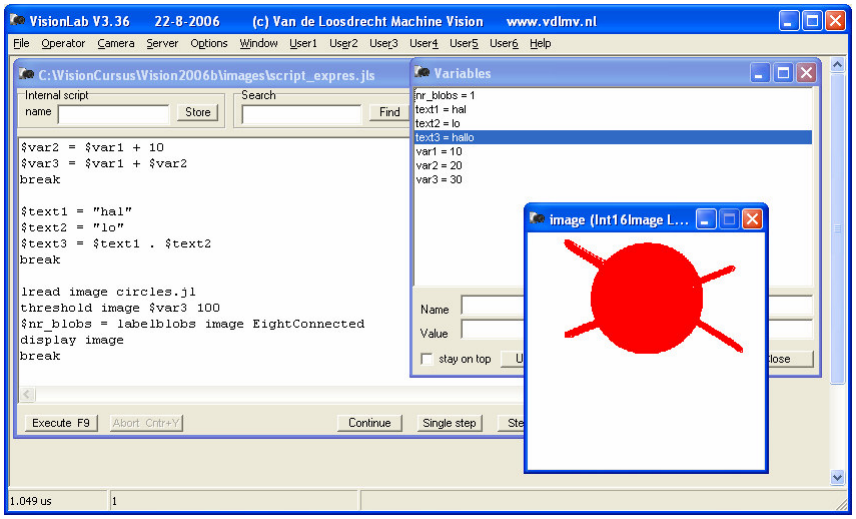
Demonstration script expressions (*)



Demonstration script expressions (*)



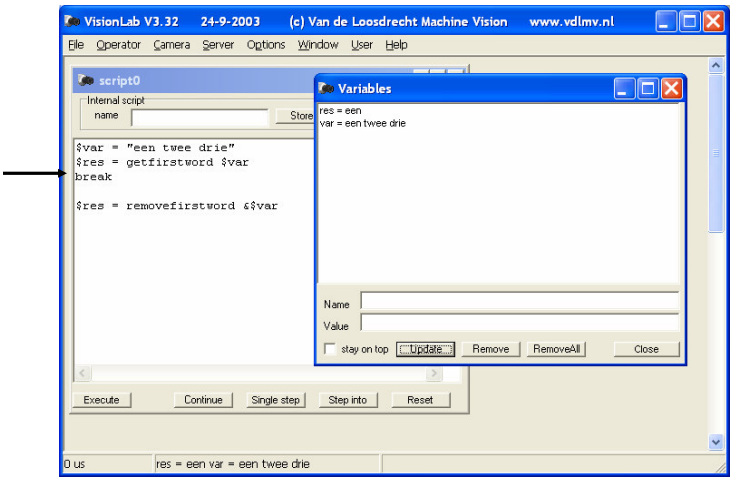
Demonstration script expressions (*)



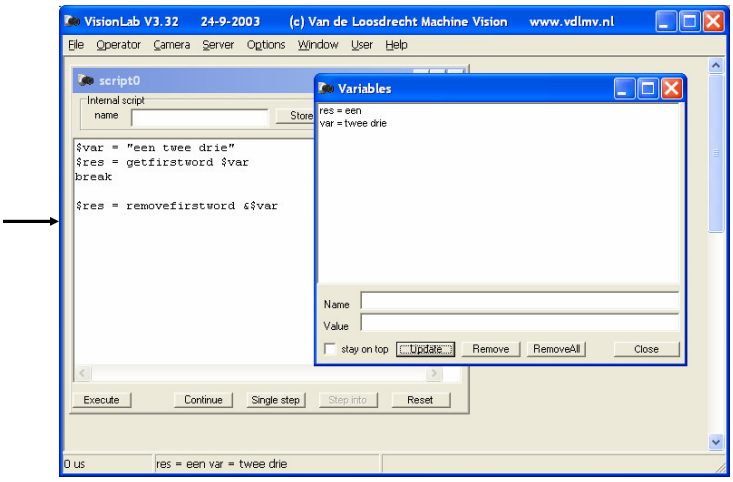
Operators on variables (*)

- **Operators:**
  - **for string manipulation**
  - **math operators for number manipulation**
- **Parameter types:**
  - **by value, <var>**
  - **by reference, <&var>**
- **Examples**
  - **by value: $res = getfirstword $var**
  - **by reference: $res = removefirstword &$var**

Parameter by value (*)



Parameter by reference (*)



Operators for string manipulation (*)

A variable can be initialized with a string using double quotes.
Example: $s = "here are spaces allowed"

. is dyadic operator used for string concatenation.

Note this operator is used for concatenation of chars in a word.
If used as $v . $w, both $v and $w must be non empty variables with one word as value.
Use for concatenation of words the string operator Concat.

8/28/2008 VisionLab 73

Operators for string manipulation (*)

A variable can be initialised with a string using double quotes.
Example: $s = "here are spaces allowed"

- **getfirstword <var>**
- **getfirstnwords <n> <var>**
- **getnthword <n> <var>**
- **getnthfromvector <n> <var>**
- **vector <i1> <i2> <i3> .....**
- **listallvars**
- **removefirstword <&var>**
- **removefirstnwords <n> <&var>**
- **removeuntilword <word> <&var>**
- **strip <&var>**

8/28/2008 VisionLab 74

Operators for special string manipulation (*)

- **getvar <&var>**
- **isequalvar <&var1> <&var2>**
- **removevar <&var>**
- **removeallvars**
- **setvar <&var> <value>**
- **isvar <varname without $>**
- **testequalvar <&var1> <&var2>**
- **concat <sequence of <var> and text>**
- **substr <&var> <offset> <count>**
- **length <&var>**
- **find <&str> <key> <offset>, find substring in string**
- **findfirstof <&str> <key> <offset>, find character in string**
- **findlastof <&str> <key> <offset >, find character in string**

For more details of operators, see online help

8/28/2008 VisionLab 75

Math operators for vars (*)

- **asin <x>**
- **acos <x>**
- **atan <x>**
- **atan2 <y> <x>**
- **ceil <x>**
- **cos <x>**
- **cosh <x>**
- **exp <x>**
- **fabs <x>**
- **floattoint <x>**
- **floor <x>**
- **fmod <x> <y>**
- **initrandomgen <x>**
- **log <x>**
- **log10 <x>**
- **mod <x> <y>**
- **pow <x> <y>**
- **random <low> <high>**
- **randomint <low><high>**
- **sin <x>**
- **sinh <x>**
- **sqrt <x>**
- **tan <x>**
- **tanh <x>**

8/28/2008 VisionLab 76

Script control statements (*)

Nesting to arbitrary depth is possible

```
if <condition> then
   <statement block>
else
  <statement block>
endif

while <condition> do
   <statement block>
endwhile

 for<var> = <expression> (<to>|<downto>) <expression> do
    <statement block>
 endfor
```

8/28/2008 VisionLab 77

Script control statements (*)

```
switch <expression>
case <tag>
  <statement block>
case <tag>
  <statement block>
default
  <statement block>
endswitch

return <expression>   // function return

label <labelName>
goto <labelName>
```

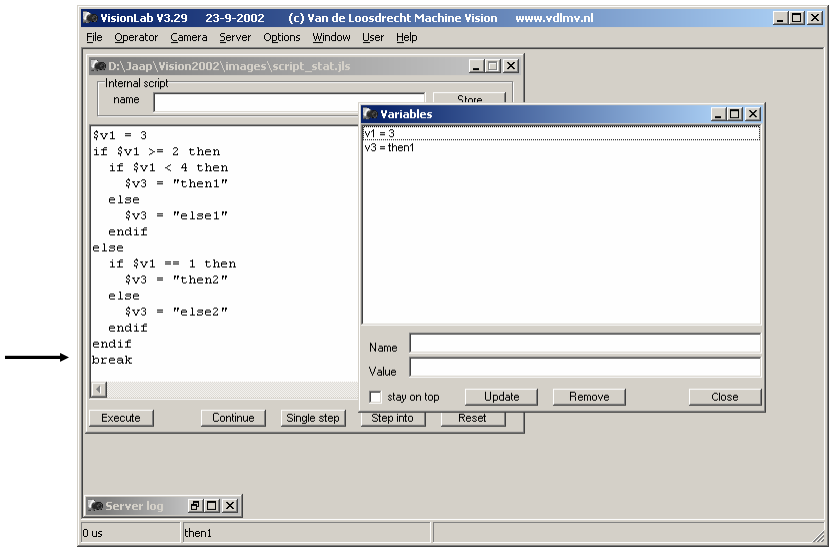
8/28/2008 VisionLab 78

Demonstration script control statements (*)

- open script script_stat.jls
- execute script
- open window 'Examine vars' in the server menu
- click continue script
- click continue script

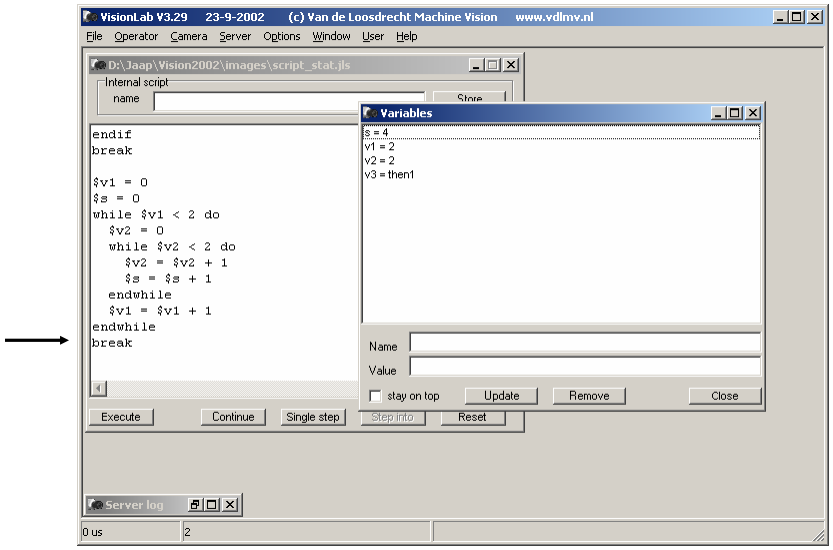
8/28/2008 VisionLab 79

Demonstration script control statements (*)

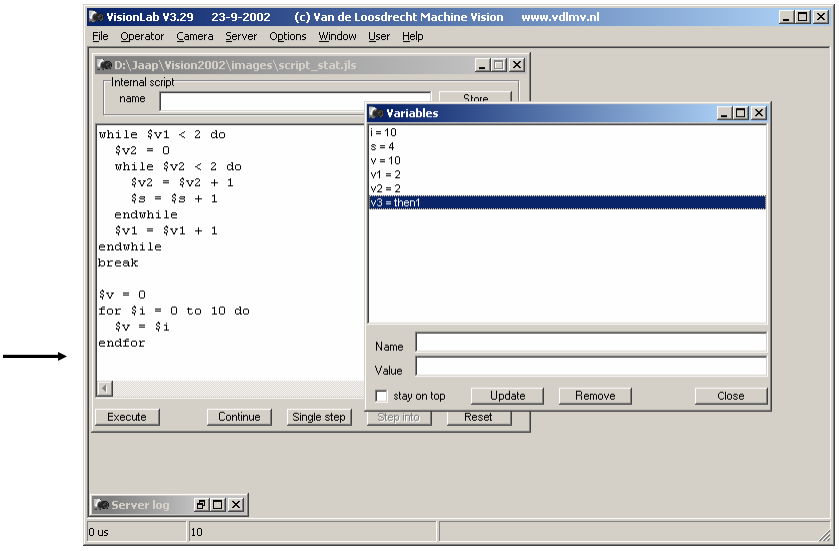


8/28/2008 VisionLab 80

Demonstration script control statements (*)



Demonstration script control statements (*)



Internal commands (*)

Internal commands are special local commands for which there is not a (direct) server command.

Examples :

- **lcwd <path>: change working directory of client to path**
- **lpwd: return working directory of client.**
- **lread <imagename> <filename>: The file with <filename> is read on the client and added to the server as image with <imagename>**
- **lwrite <imagename> <filename>: The image with <imagename> is written on the client as file with <filename>**
- **display <name>: The image with <name> is displayed**
- **GUIForm: create dialog box from script**
- **see also on-line help for more examples**

8/28/2008 VisionLab 83

Demonstration GUIForm: create dialog box from script (*)

GUIForm <caption> {<spinedit> | <editbox>}

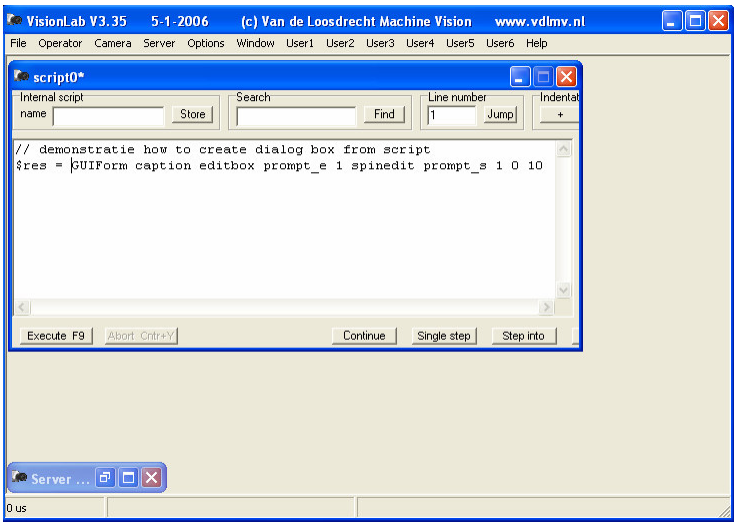
Displays a dynamic form from a script in order to query the user with inputs, result is a string with the answers.

- spinedit: <prompt> <default> <low> <high>
A spinedit field is added to the form with a prompt and a default value. Low and high specify the extreme possible values.

- editbox: <prompt> <default>
An editfield is added to the form with a prompt and a default value.

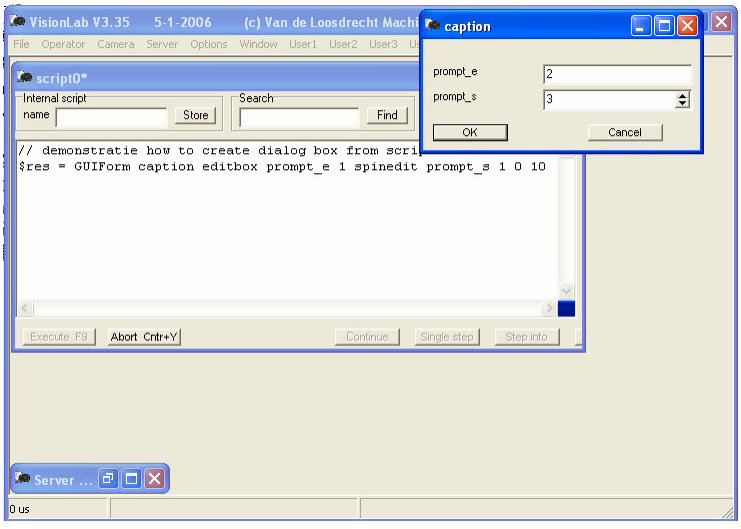
8/28/2008 VisionLab 84

Demonstration GUIForm: create dialog box from script (*)



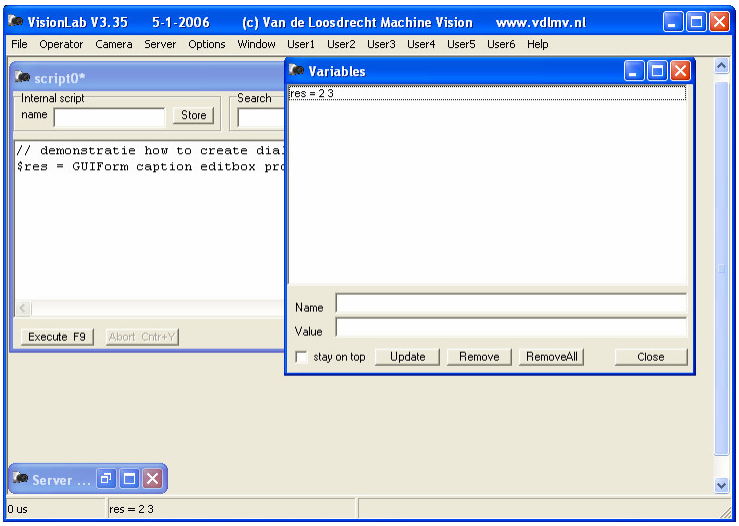
8/28/2008 VisionLab 85

Demonstration GUIForm: execute script (*)



8/28/2008 VisionLab 86

Demonstration GUIForm: examine result variable (*)



Special server commands (*)

Examples :

- **cwd <path>: change working directory of server to path**
- **pwd: return working directory of server**
- **read <imagename> <filename>: The file with <filename> is read on the server and added to the server as image with <imagename>**
- **write <imagename> <filename>: The image with <imagename> is written on the server as file with <filename>**
- **Date and time command, see next slide**
- **see also on-line help for more examples**

Date and time commands (*)

- **Time: returns <day> <month> <daylnMonth> <hh:mm:ss> <year>**
- **Delay <secs>: delay operations for secs seconds time.**
- **MicroDelay <us>: delay operations for us micro-seconds time.**
- **MilliDelay <ms>: delay operations for ms milli-seconds time.**
- **ResetTimeStamp: reset time stamp counter to zero**
- **MicroTimeStamp: return a time stamp in micro seconds, note: maximum time for overflow after reset is 35 minutes.**
- **MilliTimeStamp: return a time stamp in mille seconds, note: maximum time for overflow after reset is 583 hours.**

8/28/2008 VisionLab 89

Pre-processor commands (*)

A command is pre-processed before it is executed. The following pre-processor commands are only available in the client:

- **%currentimage: this command is replaced with the name of the current selected image or the name of the image selected as script image**
- **%secondimage: this command is replaced with the name of the image selected as second image**
- **%thirdimage: this command is replaced with the name of the image selected as third image**
- **%startupdir: directory name from which the client has been launched**

8/28/2008 VisionLab 90

Script procedure calls (*)

There are three different kind of scripts:

- **local scripts: script is executed in the client or called with the *lcall* command from a script executed in the client**
- **remote file scripts: script is executed in the server. Script is started with a *call* <remote fileName> <params> command**
- **internal script: script is executed in the server. Before being started with a *icall* <internal scriptName> <params> command the script has to be stored in the server**

The formal parameters have the form %pn, where n starts at 1
The first parameter is %p1, the next %p2 etc

A script can return a function result with the return command
return <function result>

Local scripts can be aborted with <control y>

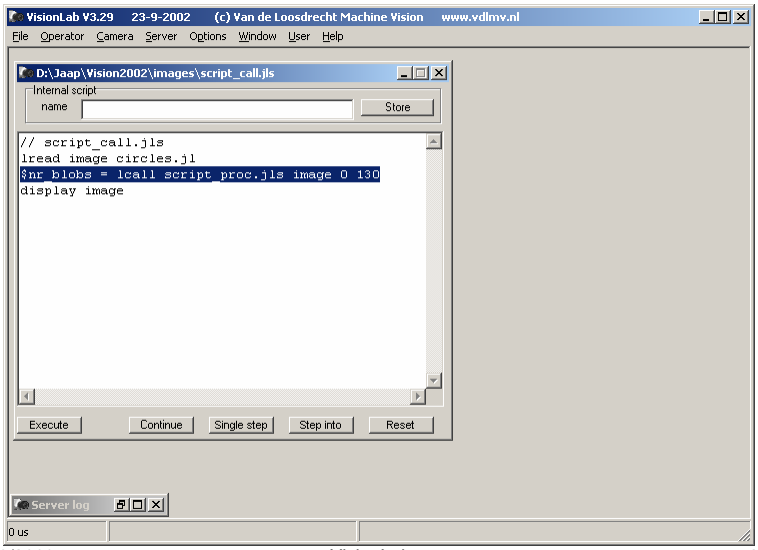
8/28/2008 VisionLab 91

Demonstration script procedure calls (*)

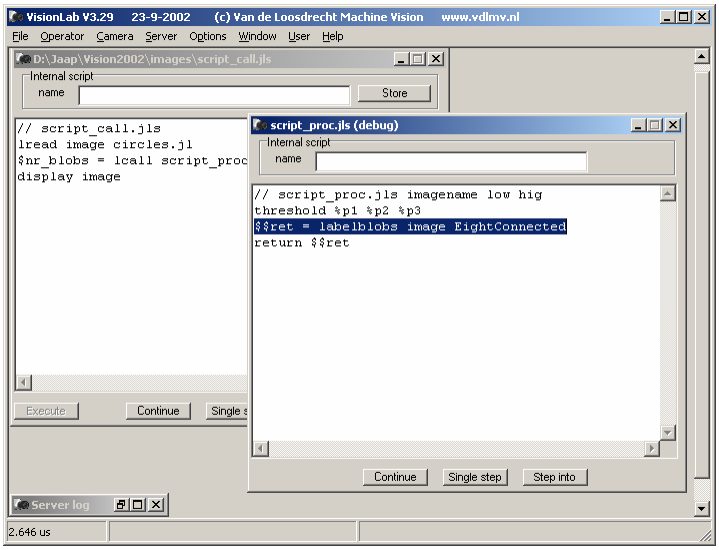
- **open script script_call.jls**
- **click single step**
- **click single step**
- **click step into**
- **click single step**
- **click single step**
- **open window 'Examine vars' in the server menu**
- **click continue script**

8/28/2008 VisionLab 92

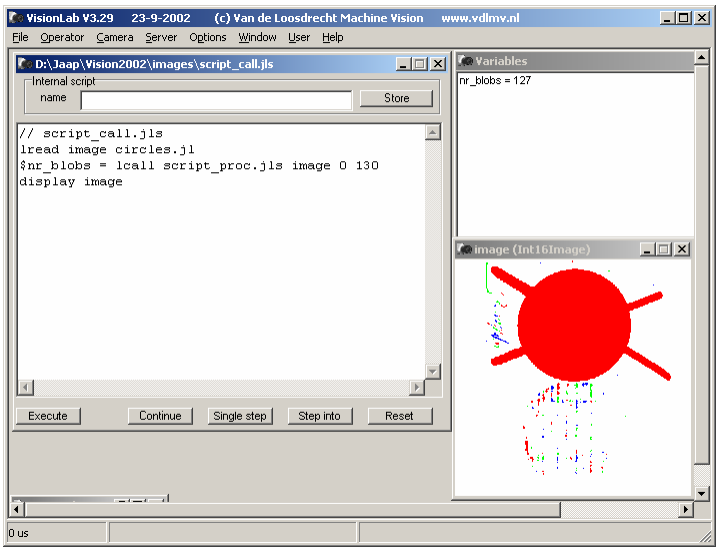
Demonstration script procedure calls (*)



Demonstration script procedure calls (*)



Demonstration script procedure calls (*)



8/28/2008 VisionLab 95

Speeding up scripts (*)

- **Using internal scripts**
- **System settings**
  - **Disable client logging**
  - **Disable server logging**
  - **Disable history mechanism**

8/28/2008 VisionLab 96

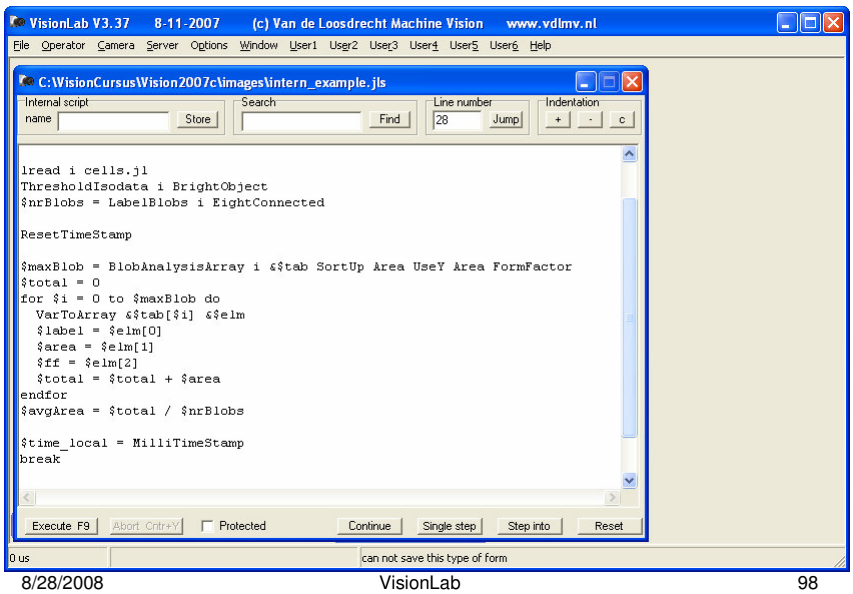
Demonstration speed up by using internal script (*)

- open script internal_example.jls
- run script to first break point
- examine variables
- examine script intern_script.jls
- run (continue) second part of script and compare executing times
- $time_local = 6411 ms
- $time_intern = 176 ms

- Note: this example is just to demonstrate internal scripts. A much faster way to calculate the average area is to count the number of object pixels and divide this number with the number of object

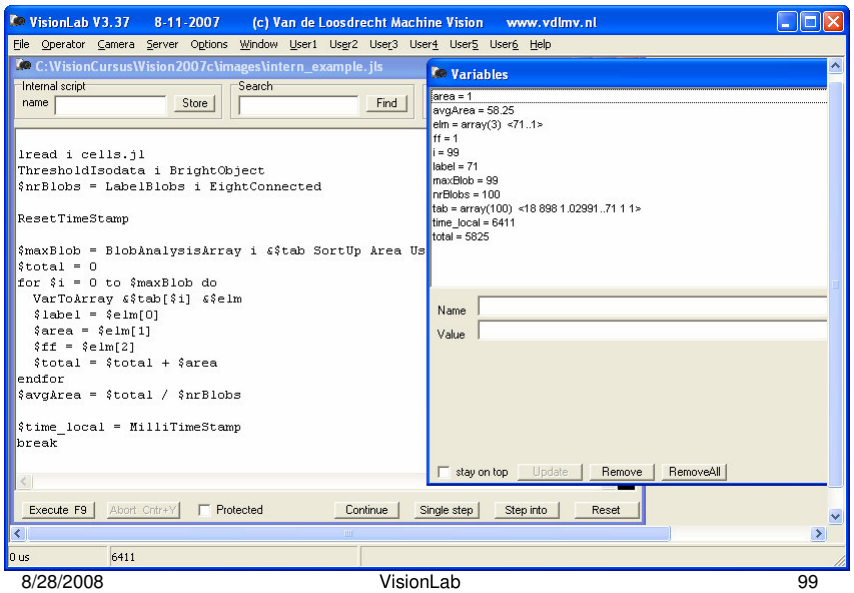
8/28/2008 VisionLab 97

open script internal_example.jls (*)

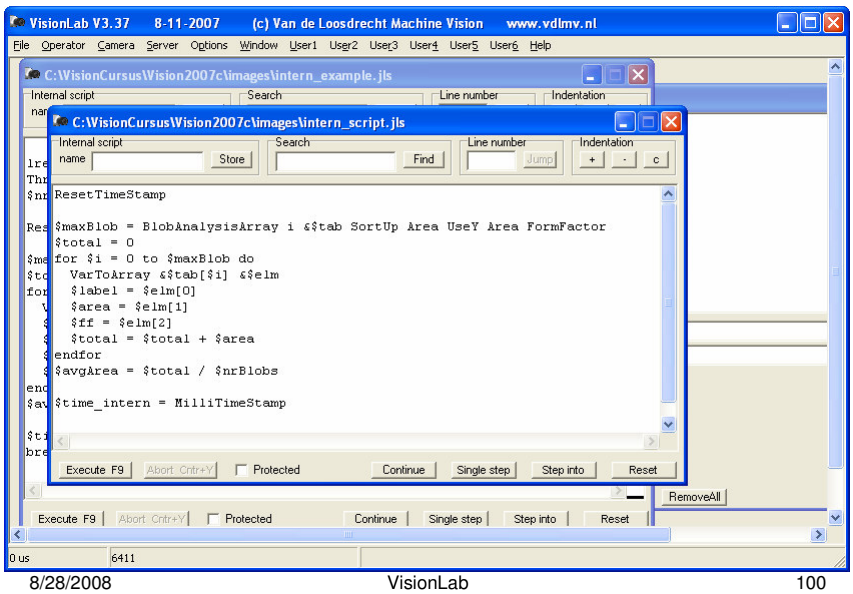


8/28/2008 VisionLab 98

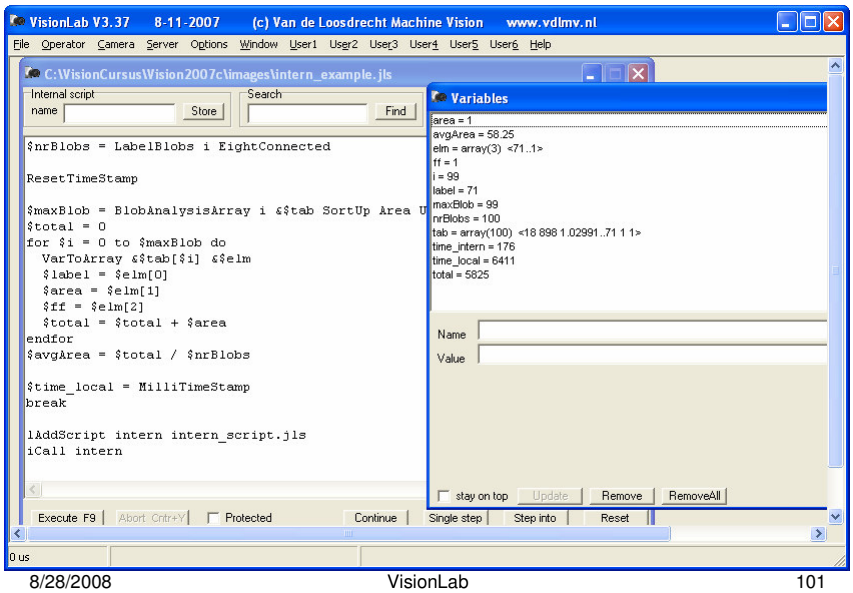
run script to first break point (*)



examine script intern_script.jls (*)



run second part of script and compare executing times(*)



8/28/2008 VisionLab 101

Speeding up scripts (*)

- **System settings**
  - **Disable client logging:**
    set in main menu item server log to “no log”
  - **Disable server logging and Disable history mechanism:**
    close visionlab
    open file visionlab.ini
    search in file for “[localserver]”
    next line has item “exename” with syntax
    vissvr.exe <port> <byteorder> <timeout> <maxhissize> <EchoOn|EchoOff> <(no)debug>
    set <maxhissize> to 0
    set <EchoOn|EchoOff> to EchoOff
    example: “exename=vissvr.exe 2066 NativeByteOrder 5 0 EchoOff nodebug”
    save and close file visionlab.ini
    start VisionLab again

8/28/2008 VisionLab 102

Adding a script as new operator (*)

New operators can be added to the system by editing the file visionlab.ini. See for more details Adding a new operator to the client

Example:

An operator with the name Script must be added to the usermenu. When the operator is selected the script with name test.jls is to be executed and it will receive three parameters.

In order to make this possible add the following lines to the file visionlab.ini and startup the system.

```
[examplescript]
menu=UserMenu
caption=Example &Script
class=IntImage
script=test.jls
paramform=scriptparam
selectedimage source
editbox low 0
editbox high 80
```

8/28/2008 VisionLab 103

Adding a script as new operator (*)

Explanation:

[examplescript]: In [ ] the name of the operator. Note: no white space behind] is allowed.

menu: The menu where the operator is added to the system.

caption: The name used for the operator in the menu. The & give the letter for the shortkey.

class: The highest class in the Image hierarchy overview on which the operator can work.

script: The filename of the script which is to be executed.

paramform: Specifies the form which to be used to ask the user for the parameters. Currently for user scripts only 'scriptparam' is supported.

8/28/2008 VisionLab 104

Adding a script as new operator (*)

The section following paramform specifies the parameters for the operator. The maximal number of parameters is 25. The parameters are displayed and send to the server in the same order as they are declared. There are the following possibilities:

- **selectedimage <prompt>: An edit field is added with a prompt and the image name of the currently selected image**
- **2ndselected <prompt>: An edit field is added with a prompt and the image name of the 2ndselected image**
- **3rdselected <prompt>: An edit field is added with a prompt and the image name of the 3rdselected image.**
- **genimagename <prompt>: An edit field is added with a prompt and a generated image name**
- **spinedit <prompt> <default> <low> <high>: A spinedit field is added with a prompt and a default value. Low and high specify the extreme possible values**

8/28/2008 VisionLab 105

Adding a script as new operator (*)

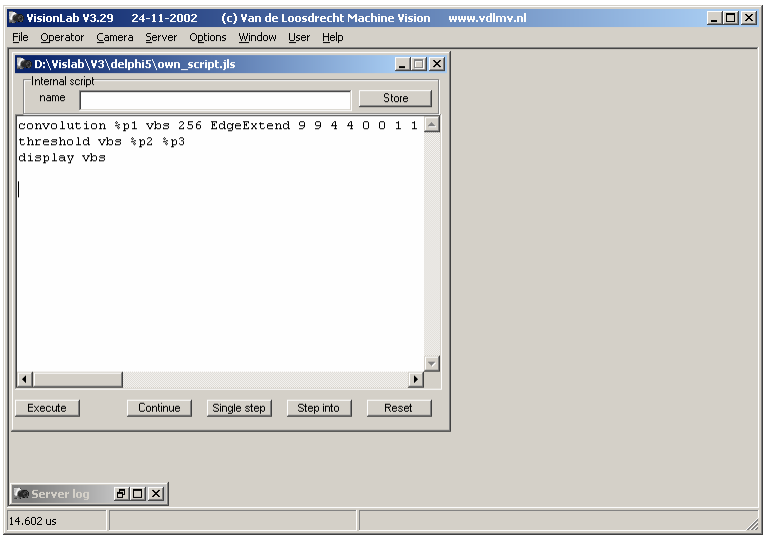
- **editbox <prompt> <default>: An edit field is added with a prompt and a default value**
- **combobox <prompt> <default> <item1> ... <itemN>:**
  A combo box is added with a prompt and a default value. The combobox is filled with the specified items. If prompt equals to 'none' combobox is invisible

8/28/2008 VisionLab 106

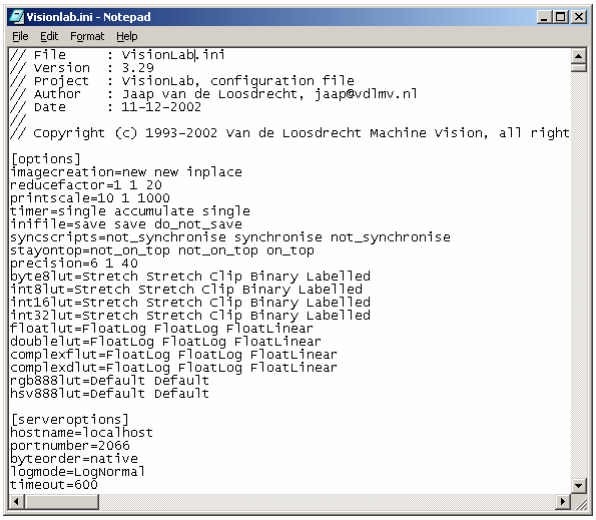
Example: Adding an own script as operator (*)

- **Demonstrate own_script.jls in User Menu item 'example script' on circles.jl and look at vislab.ini for added script as new operator.**

Example: Adding an own script as operator (*)

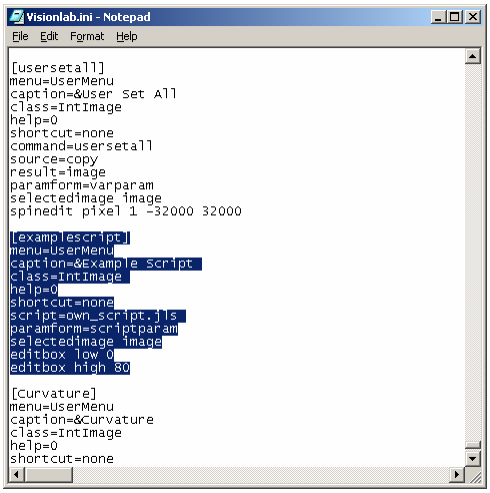


VisionLab.ini configuration file (*)



8/28/2008 VisionLab 109

Adding script own_script.jls as operator in visionlab.ini (*)



8/28/2008 VisionLab 110

Adding an own C++ operator (*)

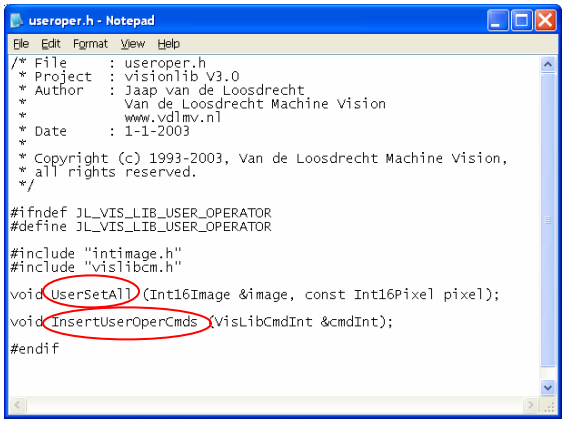
Add new operator to server to set all pixels to same value
- Write new operator in C++ (UserSetAll)
- Write envelope for operator (SetAllCmd)
- Add envelope to cmd interpreter (Main in visserv.cpp)

- Command send to server will be: usersetall <image name> <pixel value>
- The server will use the first word as name of the command ("usersetall") and will use this name to find the associated envelope function SetAllCmd
- SetAllCmd will get as input stream the command without the first word: "<image name> <pixel value>"
- SetAllCmd will:
  - read the image name and pixel value from the inputs stream
  - check whether image name is a valid image
  - call UserSetAll with image and value as parameter
  - the call to userSetAll is surrounded with timer functions which supports the displaying of the time needed for the operation in the left bottom corner of the GUI
  - The call AddLastCmdToHistory adds the command to the history list of the image

Add new operator to GUI
- Edit vislab.ini

See for example the files in the src directory of the SDK:
useroper.h, useroper.cpp and visserv.cpp

useroper.h (*)



useroper.cpp (*)

useroper.cpp - Notepad

File Edit Format View Help

```
using namespace JL_visionLib_V3;

// the new operator itself

void UserSetAll (Int16Image &image, const Int16Pixel pixel) {
   for (int y = 0; y < image.GetHeight()-1; y++) {
      for (int x = 0; x < image.GetWidth()-1; x++) {
         image(x,y) = pixel;
      } // for x
   } // for y
   /* alternative faster solution using pointers:
   Int16Pixel *ownPtr = image.GetFirstPixelPtr();
   Int16Pixel *lastPtr = image.GetLastPixelPtr();
   while (ownPtr <= lastPtr) {
      *ownPtr++ = pixel;
   } // for all pixels */
} // UserSetAll

// the following code envelope is necessary to add the new operator to the server

static void SetAllCmd (istream &is, ostream &os, VisLibCmdInt &cmdInt) {
   QString imageName;
   Int16Pixel pixel;
   is >> imageName >> pixel;
   if (is.fail()) throw(Error("[SetAll] input error"));
   if (CheckIsInt16Image(cmdInt, imageName, os)) {
      Int16Image *src = dynamic_cast <Int16Image *>(cmdInt.GetImage(imageName));
      cmdInt.StartTimer();
      UserSetAll (*src, pixel);
      cmdInt.StopTimer();
      cmdInt.AddLastCmdToHistory (imageName);
      os << "example of a result string";
   }
} // SetAllCmd

// called by the main program
void InsertUserOperCmds (VisLibCmdInt &cmdInt) {
   cmdInt.InsertCmd ("usersetall",  CmdIntCommand(SetAllCmd, cmdInt), "<imageName> <pixelvalue>");
} // InsertUserOperCmds
```

the operator to add

the envelope function

to be called from main

visserv.cpp (*)

visserv.cpp - Notepad

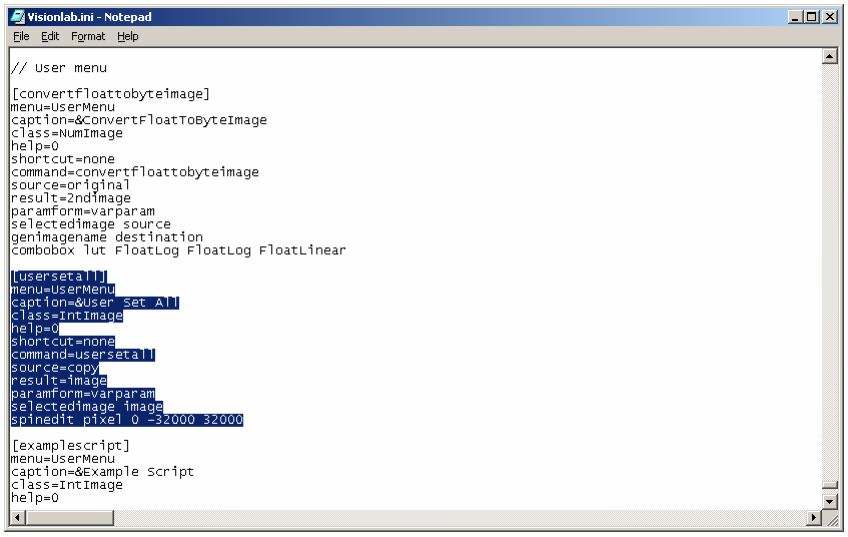
File Edit Format View Help

```
.....
int main (int argc, char *argv[])
   .....

      VisLibCmdInt cmdInt(debug);
      VisLibSvr server(cmdInt, port, byteOrder, timeOut);

      // register camera's
      // RegisterFireCam(cmdInt);
      RegisterDummyCam(cmdInt);
      RegisterFileCam(cmdInt);
      //RegisterRioCam (cmdInt);
      // examples:
      InsertUserOperCmds (cmdInt);   // simple user defined operator
      // insert your own cmds <----------
      server.Run();
   }
   ....
} // main
```

Add new operator to GUI (*)



8/28/2008 VisionLab 115

Adding a new C++ operator to the client (*)

New operators can be added to the client by editing the file visionlab.ini. The new operators will be active the next time the system is started.

The descriptor for the operator with name usersetall looks like:

```
[usersetall]
menu=User1Menu
caption=&User Set All
class=IntImage
help=0
shortcut=none
command=UserSetAll
source=copy
result=image
paramform=varparam
selectedimage image
spinedit pixel 1 -32000 32000
```

8/28/2008 VisionLab 116

Adding a new C++ operator to the client (*)

[usersetall]: In [ ] the name of the operator. Note: no white space behind] is allowed.

menu: The menu where the operator is added to the system.

caption: The name used for the operator in the menu. The & give the letter for the shortkey.

class: The highest class in the Image hierarchy overview on which the operator can work, or "NoImage" if operator does not work with images.

shortcut: The name for a key on the keyboard, which can activate the operator without using the menu. Possible values: none, ctrl+a, .. ctrl+z, f1, .. f12, ctrl+f1, .. ctrl+f12, shift+f1, .. shift+f12, shift+ctrl+f1, .. shift+ctrl+f12, ins, shift+ins, ctrl+ins, del, shift+del, ctrl+del, alt+bksp and shift+alt+bksp.

command: The command name which is sent to the server.

8/28/2008 VisionLab 117

Adding a new C++ operator to the client (*)

source: Specifies on which image operation is performed, possible values:

- **original: operation is performed on the source image.**
- **copy: operation is performed on a copy of the original, the name of the copy is either specified by a imagename parameter (see below: description of paramform) or generated by the system.**
- **none: operation is not performed on an image.**

result: Determines how the result of the operation is to be displayed, possible values:

- **image: The image (= first parameter) on which the operation was performed is displayed.**
- **2ndimage: The image specified as second parameter in the parameter list of the server command is displayed.**
- **3rdimage: The image specified as third parameter in the parameter list of the server command is displayed.**
- **4thimage: The image specified as fourthparameter in the parameter list of the server command is displayed.**
- **2nd3rdimage: The images specified as second and third parameter in the parameter list of the server command are displayed.**

8/28/2008 VisionLab 118

Adding a new C++ operator to the client (*)

result (continued):

- **2nd3rd4thimage: The images specified as second, third and fourth parameter in the parameter list of the server command are displayed.**
- **bothimages: image and 2ndimage are displayed, see above.**
- **box: As result a string with the format '(x1,y1) (x2,y2)' is expected from the server. A box with lefttop (x1,y1) and rightbottom (x2,y2) is drawn in the current active image.**
- **circle: As result a string with the format '(x,y) r h' is expected from the server. A circle with centre (x,y) and radius r is drawn in the current active image. h is the number of hits.**
- **circles: As result a string with the format 'n (x,y) r h ... (x,y) r h' is expected from the server. N circles with centres (x,y) and radius r are drawn in the current active image.**
- **line: As result a string with the format '(x1,y1) (x2,y2) is expected from the server. A line from (x1,y1) to (x2,y2) is drawn in the current active image.**
- **polarline: As result a string with the format '(r,phi) h' is expected from the server. A line with polarcoordinates is drawn in the current active image. h is the number of hits.**

Adding a new C++ operator to the client (*)

result (continued):

- **polarlines: As result a string with the format 'n (r,phi) h ... (r,phi) h is expected from the server. N lines with polarcoordinates is drawn in the current active image. h is the number of hits.**
- **string: As result a string is expected from the server which is displayed in the info bar.**
- **stringlist: As result a list of strings is expected from the server which is displayed in a window.**
- **nothing: Nothing is displayed.**

paramform: Specifies the form which to be used to ask the user for the parameters. Currently for user C++ operators only 'varparam' is supported.

The section following paramform specifies the parameters for the operator. The maximal number of parameters is 25. The parameters are displayed and sent to the server in the same order as they are declared.

Adding a new C++ operator to the client (*)

There are the following possibilities for the parameters :

- **selectedimage <prompt>: An editfield is added with a prompt and the image name of the currently selected image.**
- **2ndselected <prompt>: An editfield is added with a prompt and the image name of the 2ndselected image.**
- **3rdselected <prompt>: An editfield is added with a prompt and the image name of the 3rdselected image.**
- **genimagename <prompt>: An editfield is added with a prompt and a generated image name.**
- **spinedit <prompt> <default> <low> <high>: A spinedit field is added with a prompt and a default value. Low and high specify the extreme possible values.**
- **editbox <prompt> <default>: An editfield is added with a prompt and a default value.**
- **combobox <prompt> <default> <item1> ... <itemN>: A combo box is added with a prompt and a default value. The combobox is filled with the specified items.**

8/28/2008 VisionLab 121

Adding a new C++ operator to the client (*)

Possibilities for the parameters (continued):

- **checklistbox <prompt> <defaults> <|> <item1> ... <itemN>: A checklistbox is added with a prompt and default check values. The listbox is filled with the specified items.**
  NOTE: this parameter can ONLY be used as the last parameter of a command.

8/28/2008 VisionLab 122