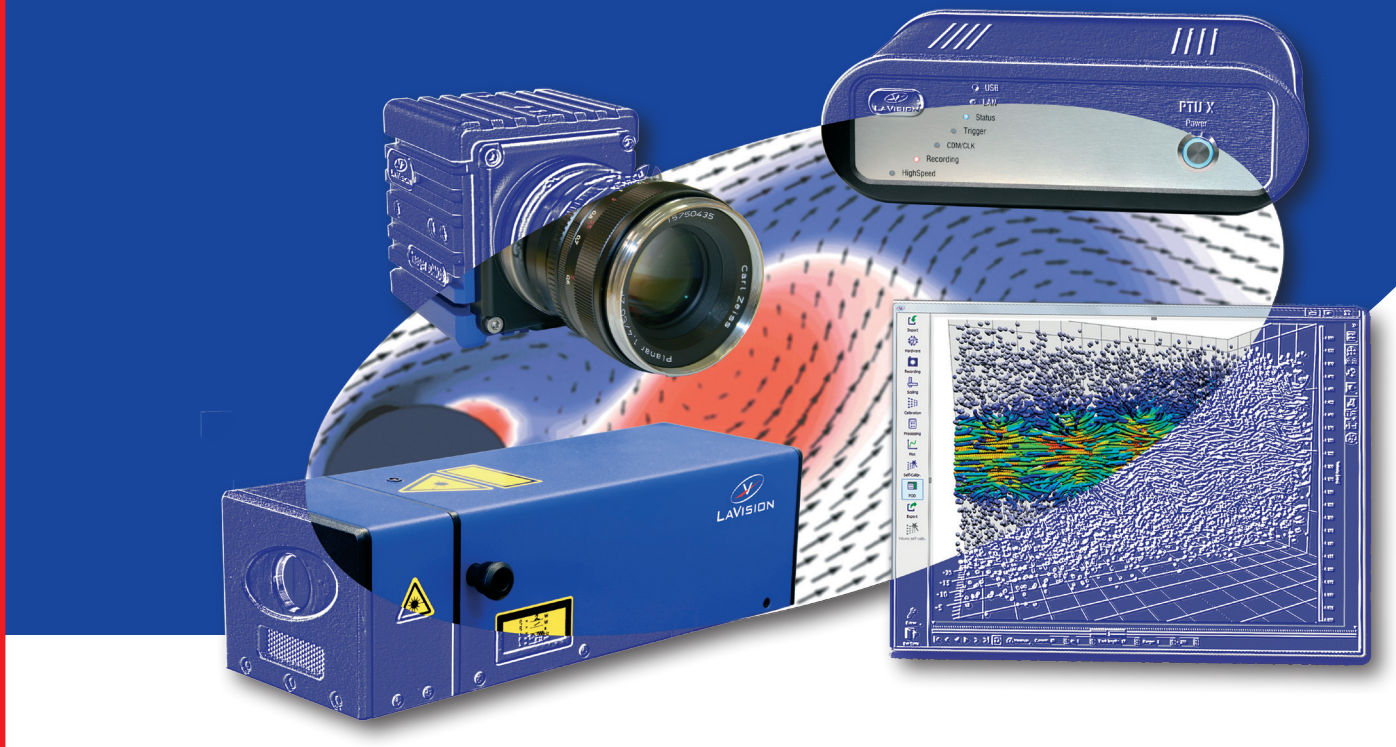
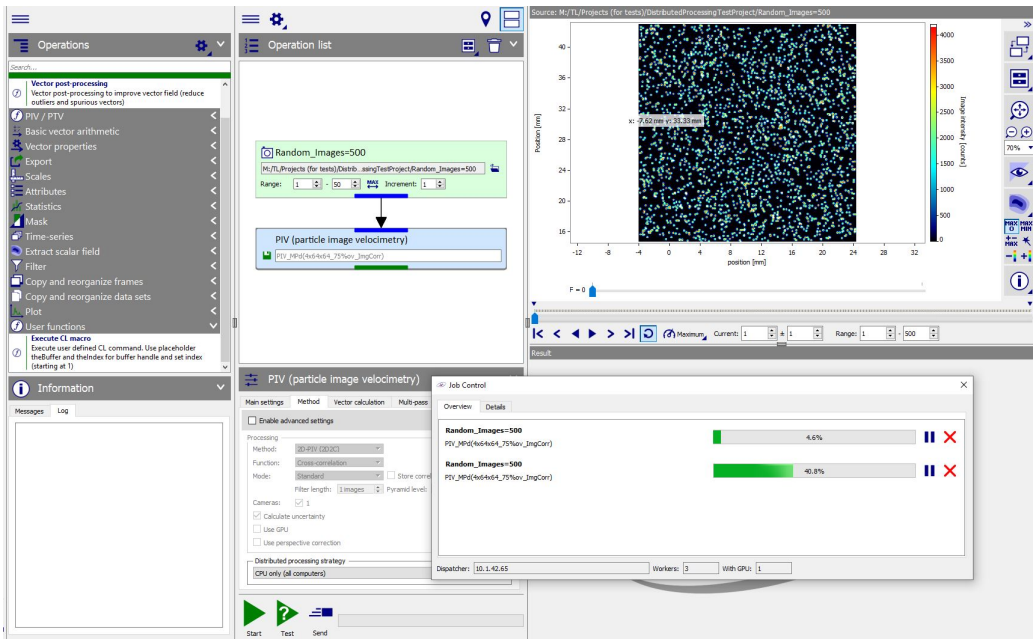


Product Manual

Distributed Processing in DaVis

Item Number(s): 1005xxx



LaVision

FOCUS ON IMAGING

Product Manual for **DaVis 10.2**

LaVision GmbH, Anna-Vandenhoeck-Ring 19, D-37081 Göttingen
Produced by LaVision GmbH, Göttingen
Printed in Germany
Göttingen, December 5, 2022

Document name: 1003013_DistributedProcessing_D10.2.pdf

Contents

1 Introduction

1.1 Three ways of distributed and scripted processing

This manual describes three different ways in **DaVis** to process data in a cluster or by a special worker version of **DaVis** without user interface and even on Linux systems.

Distributed Processing requires a fixed installation of several **DaVis** workers in the local network. The workers are usually starting as a service, need a special license and are available for Linux and Windows. The so called **Master DaVis** on a Windows PC is used to setup a processing list and sends this list via network to the workers for processing.

The **Cluster Script Processing** on a Linux cluster works similar to **Distributed Processing** but does not need a network connection to the **Master DaVis**. Scripts (e.g. bash scripts) are generated from the **Processing** dialog and must be distributed within the cluster network manually using some cluster tools.

Command Line Processing is working with the standard Windows and Linux versions of **DaVis** and simply starts **DaVis** with two parameters: the operation list file and the source set.

Available for processing on Linux systems are the standard imaging and vector operations and the PIV, Tomographic PIV, Strain, PTV and Shake-the-Box operations.

Not supported by the Linux version of **DaVis** are some of the imaging packages like Spray Geometry and Shadowgraphy. Also the Cuda processing mode of PIV does not run on Linux systems.

Setup should be executed by **DaVis** expert users, network and Linux experts, if needed with the help of network or cluster administrators. Support of the setup can be requested by **LaVision**, please ask our sales department for more information and read the chapter about service on page 73.

The different archives and installers in the download area of the **LaVision** website are described on page 10.

1.2 Overview of distributed and scripted processing modes

The following table gives a rough overview of the available ways to distribute the processing to different nodes.

	Distributed Processing	**Cluster Script Processing**	**Command Line Processing**
DaVis version	8.x, 10.1	8.4, 10.0.3 and later	10.0.3 and later
Number of "Master" licenses required	One	One (to generate script only)	One (to generate processing list only)
Master license continuously needed?	Yes in 8.x No in 10.1	No	No
Special license needed for usage?	No	Yes (to get the script generation dialog)	No
Licenses needed for worker nodes	No in 8.x Yes in 10.1	No	Yes
Tool used for job distribution	Master DaVis in 8.x Dispatcher in 10.1	OpenMPI e.g., other tools are supported	Via command line
Worker node OS	Windows or Linux	Linux	Windows or Linux
Worker can use GPU for PIV processing?	Windows only	No	Windows only
Several workers per PC (**)	Yes	Yes (if the tool supports this)	Yes
File Server OS	Windows or Linux (*)	Windows or Linux (*)	Windows or Linux (*)

	Distributed Processing	**Cluster Script Processing**	**Command Line Processing**
Pros	Feedback when jobs are finished No other tools required for job distribution	No worker node licenses required Master license not continually required	 Master license not continually required
Cons	Special licence for workers since 10.1	Customer must use own tool for job distribution	One licence needed per node

Table 1.1: (*) OS does not matter as long as the share is mounted into the standard file system.
(**) Useful on PCs with more than 64 cores as Windows splits all cores into two blocks of equal core numbers.

1.3 Supported Linux versions

Supported and tested Linux distributions running **DaVis** 10.1 and later:

- Distributions for x86_64 processors. A 32 bit version is not available.
- Ubuntu 16.04 or newer.
- Open SuSE Leap 42.3 or newer.
- Fedora 27.
- CentOS or Red Hat RHEL:
  - **DaVis** 10.1.0 requires versions 6.9 or 7.4: Special binary builds available. Setup automatically uses folder `linux64-centos69`.
  - **DaVis** 10.1.1 requires version 7.4 or later. Older versions are no longer supported because Qt 5.12 is not published for those versions.
  - **DaVis** 10.1.2 requires version 7.5 or later. Older versions are no longer supported because Qt 5.14 is not published for those versions.

  - **DaVis** 10.2.1 requires same version as 10.1.2 and is tested CentOS 7.9.

- Other distributions should be not too old and fulfill the general runtime library requirements:
  DaVis 10.1.0 needs the runtime libraries in version GLIBC_2.15, GLIBCXX_3.4.21 and CXXABI_1.3.9.
  DaVis 10.1.1 needs the runtime libraries in version GLIBC_2.23, GLIBCXX_3.4.21 and CXXABI_1.3.9.
  DaVis 10.1.2 needs the runtime libraries in version GLIBC_2.23, GLIBCXX_3.4.21 and CXXABI_1.3.9.
  Usage of binaries from folder `linux64-centos69` instead of `linux64` can be tried in **DaVis** 10.1.0. Since **DaVis** 10.1.1 the binaries are always built on a CentOS system and the `linux64-centos69` has been removed. See chapter on page 59 about the **Installation** for details.

1.4 Overview the different installers

The needed archives are either included in the `WorkerSetup` folder on the delivered USB flash drive or can be downloaded by registered customers from the **LaVision** website: `http://www.lavision.de`. The different archives and installers are available in the **DaVis Distributed Processing** section on the right side of the **Download - Software**, see figure 1.1.

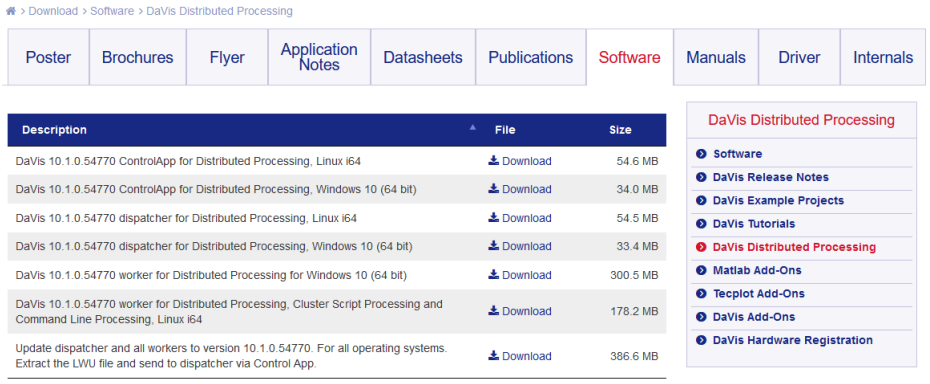


Figure 1.1: Download area for distributed processing and cluster script processing on the **LaVision** website.

The following installers are available. Detailed descriptions about the installation of the different setups and archives are given in the chapters

about installation of distributed processing on page 14 and about cluster script processing on 47.

Name	**Content**
SetupDaVis.exe	Installer for the Windows version of **DaVis** including the parts for the local installation with master, dispatcher and worker on the same PC.
ControlAppLinux.zip	Archive for the control app with Linux and Mac version.
SetupControlApp.exe	Installer for the control app on Windows.
DaVisDispatcherLinux.zip	Archive for the Linux version of the dispatcher.
SetupDaVisDispatcher.exe	Windows installer for the stand alone dispatcher. Should not be installed on the same PC as a normal master DaVis!
SetupDaVisWorker.exe	Windows installer for the stand alone worker. Should not be installed on the same PC as a normal master DaVis!
DaVisWorkerLinux.zip	Archive with the Linux worker to be used for distributed processing, cluster script processing or command line processing.
DaVisWorkerAutoUpdate.zip	Archive includes the lwu file needed for automatic update of dispatcher and all workers, see details on page 42.

Table 1.2: Installers and archives for distributed processing and cluster script processing.

2 Distributed Processing

2.1 How it works

A special installation mode of **Distributed Processing** with master **DaVis** (the standard **DaVis** used to configure processing lists), dispatcher and worker all together on one PC is available from the standard installer and can be used to test the behavior of job distribution and also as kind of a local **job manager**. Several different processing jobs can be send to the dispatcher changing e.g. some processing parameters, changing the source set or even changing the project. The user can work with **DaVis** all the time and does not need to wait for the results. The dispatcher manages the queue of jobs and asks the local worker to process them one after another in the background. With the help of standard system application priority the worker is allowed to use all cores of the CPU while the Master **DaVis** and other applications are inactive. While the user is busy with other applications, the worker falls to low CPU usage and does not disturb the user. The worker takes very low system resources in idle state and has no almost effect to other activities on the PC, just standard memory usage and few communication with the dispatcher. After finishing a job the **DaVis** project viewer displays the latest results.

The full network installation of **Distributed Processing** requires a fixed installation of several **DaVis** workers in the local network. The workers are usually starting as a service, need a special license and are available for Linux and Windows systems. All installations of **Master DaVis** software on different Windows PCs in the network can be used to setup processing list and to send these lists into the network to the dispatcher for further distribution to the workers.

One dispatcher must be installed in the network to handle the complete communication and the database of enqueued jobs, active jobs and finished jobs. The dispatcher can be installed together with a worker or even with the Master **DaVis** on the same PC, but the dispatcher should be seen as a server for distributed processing and therefore should be installed on

a PC running all the time. Worker PCs could be switched off or restarted at any time. Just running processing activities are interrupted and automatically redistributed by the dispatcher to other workers. Such jobs would not get corrupted or lead to missing results.

A stand-alone control application can also be used on any PC in the local network to watch the dispatcher activity, progress of all jobs and state of the workers. The control dialog is integrated into **DaVis**, too. The dispatcher can optionally send emails to the user who defined the processing job when processing finishes.

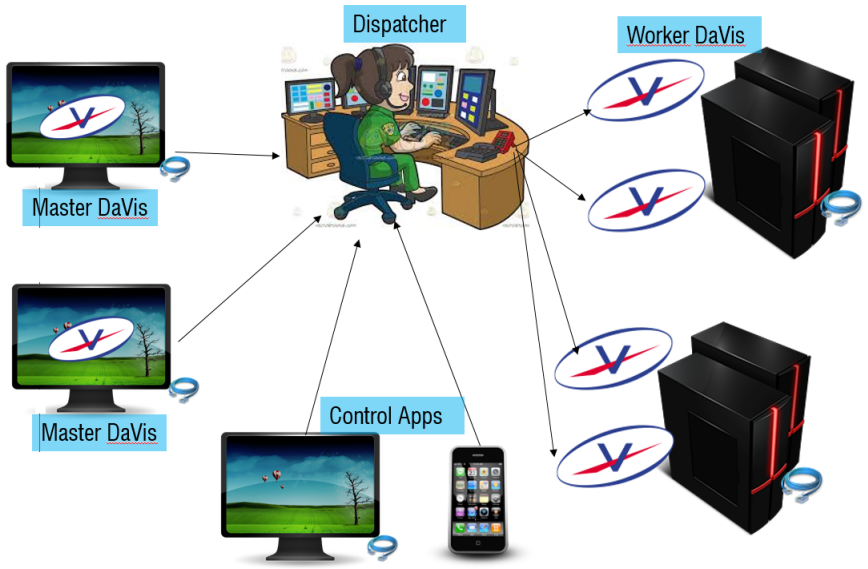


Figure 2.1: Rough overview of the different parts of distributed processing.

2.2 Installation

The standard **DaVis** installer for Windows can be used for the full installation of the local distribution system with master, dispatcher and worker on the same PC only. When installing a dispatcher or a worker on a separate system like a pure worker PC then the special installer must be used! Such installers for dispatcher only and for worker only and both for Windows and Linux are available on the **LaVision** website `http://www.lavision.de` for registered customers, see detailed description on page 10.

2.2.1 Master and Dispatcher and Worker on the same Windows PC

The installation of the Master **DaVis**, of the dispatcher and of one worker all on the same PC comes with the standard **DaVis** installer and works with the standard **DaVis** license. Select the first option in the setup dialog: **Install DaVis using license and configuration file *.lcfx**. Mode **Install a generic DaVis version** does not support installation of the local worker, because the worker license must be available during installation, otherwise the worker would fail to start. In the local configuration the worker will always use the same license as the master **DaVis**.

After selection the `lcfx` file the setup opens a dialog to select the **options**, see figure 2.2. Select **Local Distributed Processing** and optional the **Distributed Processing Control App**. This application is used to control all workers and jobs and is described in detail on page 36. The app is also included in the standard **DaVis** software but in this way could be installed on any PC without **DaVis**. There is also a small and simple installer available just for this app.

Afterwards the installer takes the standard configuration file and installs all three software packages into the same folder, using a special folder named `DistributedProcessing` for the dispatcher in child folder `Dispatcher` and the worker in subdirectory `DaVisWorker`.

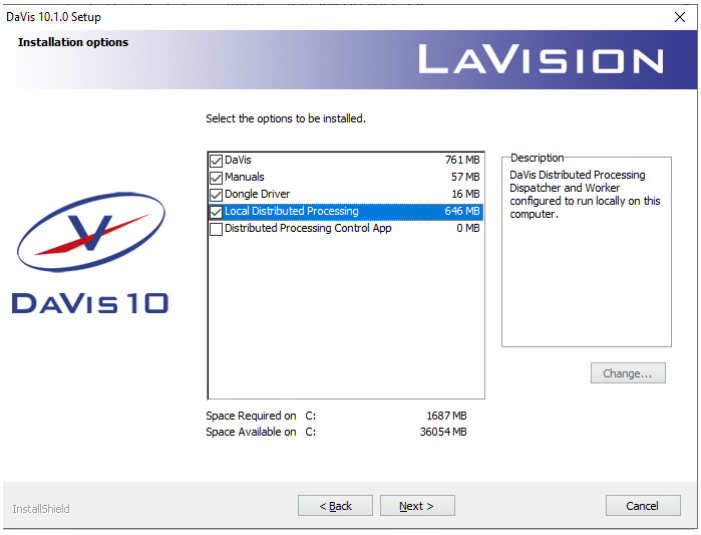


Figure 2.2: Installer options for distributed processing.

The installer also creates scheduled tasks on request to start the dispatcher and the worker during startup of the Windows PC, see page 21 for details.

The task needs account credentials to be started automatically. These credentials are requested by the installer, see figure 2.3, and used by the Windows system to start the tasks. This means, dispatcher and worker are running with the entered credentials, which usually should be the same as the current user.

If starting together with the Master **DaVis** software or manually, no credentials are needed and no scheduled tasks are created. A description of these options can be found on page 31. Changing to the start with Windows mode is not possible by a simple mode change in **DaVis** because the installer did not create the needed tasks!

In case of this full installation also no sharing of the **DaVis** project folder is needed. Master and worker **DaVis** are able to use the local project folder both from a local disk and from a network drive.

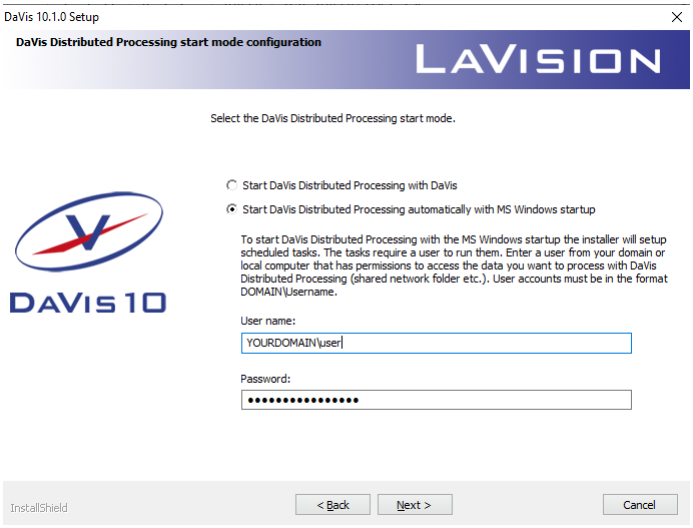


Figure 2.3: Installer requests account credentials for task creation.

Verification of installation success

If dispatcher and worker are installed as service then both are started by the installer at the end of the installation process. In case of a system reboot both applications are started automatically and the **DaVis** Master will already see the dispatcher and connect after startup.

A simple check is to start the master **DaVis** and open the control dialog from the **Extras** menu as **Dispatcher control**. The bottom line of this dialog should display the IP address of the **dispatcher** PC. If the IP ad-

dress is not given, then the master has not been able to connect to the dispatcher. Also the number of available **workers** is given in connected state. If the worker has been started and connected to the dispatcher, then the displayed value must be 1. Details on this dialog are given on page 36.

If all applications have started, the Windows task manager shows two **DaVis.exe** tasks and one **DistributedProcessingDispatcher.exe** task. Usually only one **DaVis.exe** task is listed in the top section of the detailed view under **Apps** while the second **DaVis.exe**, the worker, and also the dispatcher can be found under **Background processes**.

Multiple workers or dispatchers on the same system

The default mode after installation is the local mode which does not support usage of workers from other systems in the local network. Changing the configuration of dispatcher and worker from local mode to network mode is possible but needs some changes in configuration files and start scripts. Better run the stand alone installers for dispatcher and workers, but note that only one dispatcher can be installed on a system with default settings. To change the settings for a second dispatcher, please read the details about the dispatcher configuration on page 27 and on page 33. Also don't install a second worker unless the worker configuration, described on page 29, is changed manually and the descriptions on page 24 of the installation of two workers on the same system are followed.

Location of log files

All log files of dispatcher and worker are stored in files
`%APPDATA%/LaVision GmbH/DistributedProcessingDispatcher-<n>.log`
and
`%APPDATA%/LaVision GmbH/DistributedProcessingWorker-<n>.log`

2.2.2 Dispatcher on Windows

The explicit installation of the dispatcher is needed only once in the local network. For a special case with multiple separate dispatchers and workers in the same network, see page 33. But the default case is to have only one dispatcher on a Windows or on a Linux system.

Download the **SetupDispatcher.exe** installer from the **LaVision** website. The installer will ask for a common root folder for the installation which is required for the automatic update mechanism and which would also contain a parallel installed worker. The installer will also create a scheduled task with the account credentials of the active user to start the dispatcher automatically during the startup of the system, see figure 2.3. The name of this scheduled task is **DaVis Distributed Processing Dispatcher**, see page 21 for details. And a firewall rule is created, too, to allow the dispatcher to use the local network. See section about program arguments on page 27 for more information about the used ports.

All log files are stored in files

`%APPDATA%/LaVision GmbH/DistributedProcessingDispatcher-<n>.log`

2.2.3 Dispatcher on Linux

Download the **DistributedProcessingDispatcher** archive from the **LaVision** website `http://www.lavision.de` as registered customer. Create a new folder on the PC which should run the dispatcher. The user who installs and who runs the dispatcher does not need to be able to access any network shares e.g. with the **DaVis** projects. Also an installation of the dongle driver is not necessary. The dispatcher neither accesses any network share nor connects to the dongle.

Extract the archive and run `./start-dispatcher.sh` for tests. To install the dispatcher as service and start the dispatcher during startup of the system please call `sudo ./install-service-dispatcher.sh`

This command will install a cron job for the current user and tries to start the dispatcher every five minutes. Special arguments for the dispatcher can be given in the first lines of file `start-dispatcher.sh`, see the section below on page 27 for the details.

If a firewall is used then the dispatcher application must be allowed open network connections. At least the broadcast port and the websocket port should be available for incoming traffic. See section about program arguments on page 27 for more information about this ports.

All log files are stored in files

`~/.config/LaVision GmbH/DistributedProcessingDispatcher-<n>.log`

A subdirectory with name `DistributedProcessing` is created to store all internal information like active jobs and finished jobs and additionally the update archives.

The main log file can be watched e.g. by `tail` to control the activity and possible error messages:

```
tail -f ~/.config/LaVision GmbH/DistributedProcessingDispatcher-0.log
```

2.2.4 DaVis Worker on Windows

The explicit installation of the worker can be executed on as many systems in the local network as wanted. This installer should not be used on systems which are also used for a master **DaVis**!

Download the installer **SetupDaVisWorker.exe** from the **LaVision** website. The setup first asks for the account credentials to run the worker, see figure 2.3. This user must be able to access all network shares with are used for **DaVis** projects. And this user can be a special user just used to run the workers on all pure worker systems in the network. Afterwards the installer asks for the identifier of the network dongle giving the worker licenses, see figure 2.4. The ID must be defined during startup to avoid problems with multiple network dongles of the **DaVis** licensing system. The worker license must be available on the used network dongle, otherwise the start of the **DaVis** worker may fail or the worker would try to use another license. If no license is found, the worker shuts down.

The last question of the installer is about the installation path, which is registered with a key in the system registry:

```
Computer\HKEY_LOCAL_MACHINE\SOFTWARE\LaVision GmbH\DistributedProcessing
```

The installer creates a scheduled task named **DaVis Distributed Processing Dispatcher** for the automatic start of the worker at system startup time and also for a regular check every 30 minutes and possible restart in case of a crashed worker. And a firewall rule is created, too, to allow the worker to use the local network. Usually the worker is started from the installer at the end of the installation process. In case of a system reboot the worker application is automatically started.

All log files are stored in files
`%APPDATA%/LaVision GmbH/DistributedProcessingWorker-<n>.log`

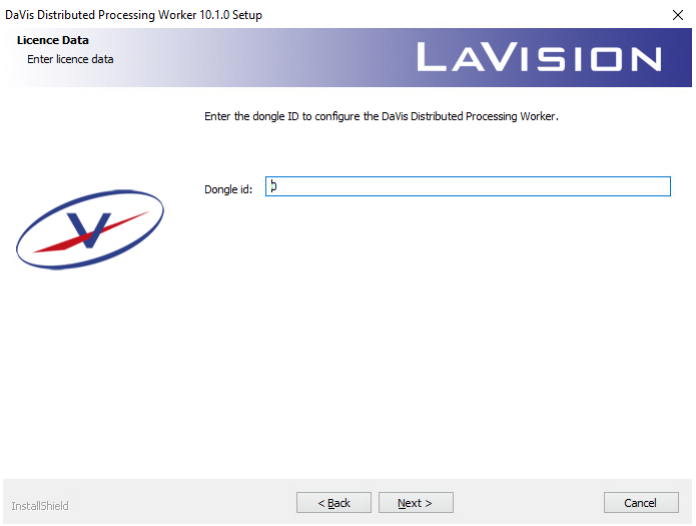


Figure 2.4: Installer requests identifier of the network dongle with worker licenses.

Note about changing the installation mode: After installing DaVis in the local mode together with dispatcher and worker on the local PC, another installation of the pure worker will not change the settings from local processing to network processing! Either start **DaVis** and change the settings in the preferences dialog or remove the old installation and install the network worker after.

2.2.5 DaVis Workers on multiple Windows PCs

The manual installation on a larger number of machines is an annoying routine. With two simple scripts the silent installation mode of Install Shield can be used. Copy both scripts to a network share and change the share and path names to your local infrastructure.

CreateTemplate.bat

Create a file on your network share with name e.g. `CreateTemplate.bat` and copy this batch code into the file. Also copy the `SetupDaVisWorker.exe` into this new folder. When executing this file via context menu and **run as administrator** on a worker PC the normal worker setup is running and should be clicked through with all required configurations. Afterwards the setup of the first worker PC has finished and the worker should start if installed as a scheduled task.

```
mkdir C:\WorkerInstaller
copy \\server\share\WorkerInstaller C:\WorkerInstaller
cd C:\WorkerInstaller
SetupDaVisWorker.exe /r /f1"C:\WorkerInstaller\worker.iss"
copy C:\WorkerInstaller\worker.iss \\server\share\WorkerInstaller
```

InstallTemplate.bat

Create another file on your network share with name e.g. `InstallTemplate.bat` and copy this batch code into the file. Then log into all the other worker PCs one after another and execute this batch file via context menu and **run as administrator**. It will install the worker in silent mode without any user interaction.

```
mkdir C:\WorkerInstaller
copy \\server\share\WorkerInstaller C:\WorkerInstaller
cd C:\WorkerInstaller
SetupDaVisWorker.exe /s /f1"C:\WorkerInstaller\worker.iss"
```

If the PCs are configured for Windows Remote Installation then this could be used to speed up the installation procedure even more. Please contact your IT departement for help.

2.2.6 Windows scheduled task

On Windows systems the installer creates a scheduled task for dispatcher and worker to start the applications together with the system at the end of the boot process and also tries to start the worker every 30 minutes in case of a crash of the executable. The **General** options are defining the account credentials, see figure 2.5, and the **Triggers** options are defining the start times for the task, see figure 2.6. The link to the application is defined by the **Actions** options, see figure 2.7, and is just using the batch file `worker-start.bat` in the root folder for the worker and `dispatcher-start.bat` for the dispatcher.

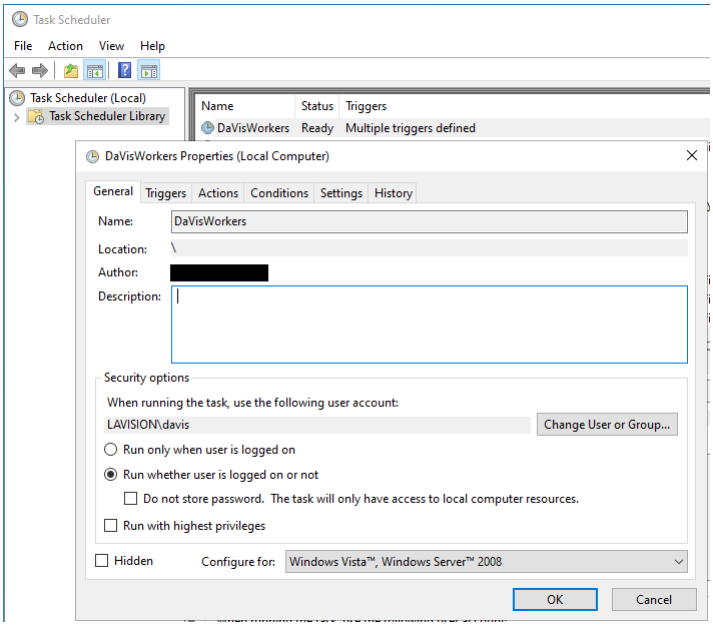


Figure 2.5: Windows task scheduler: General options for the DaVis worker.

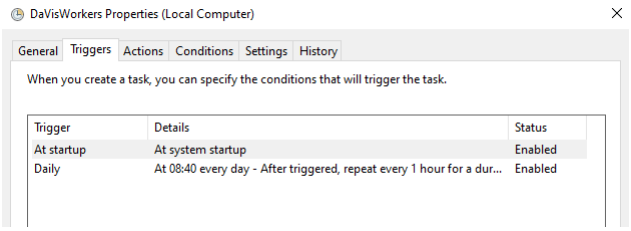


Figure 2.6: Windows task scheduler: Trigger options for the DaVis worker.

Company policy might disable task installation

In some companies the network policies disallow installation of scheduled tasks. After the first installation of a dispatcher or worker the installed scheduled task should be checked manually. Open the **task scheduler** in the Windows system settings (or **Aufgabenplanung** in the German version of Windows) and search for tasks with name **DaVis**. See figure 2.5 for a screenshot of the **task scheduler** dialog.

If company policies disallow the installation, then the task is missing. In this case either talk to you IT departement or create the task manually.

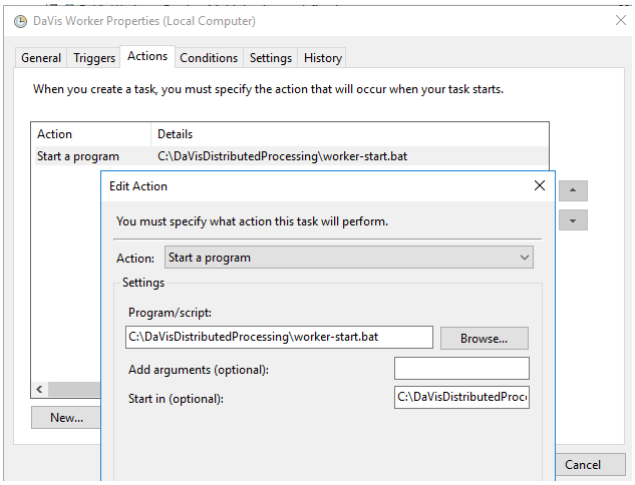


Figure 2.7: Windows task scheduler: Action options for the DaVis worker.

Solving access problems

Especially the start of the worker together with the system is a little bit critical concerning the access to network shares. On some systems the worker simply starts *too early* and never gets access to network shares. This may happen if the system configuration mounts shares after login of the user. Without login the worker service has to establish the network connection to a share manually.

The first possibility is to setup the system to auto login of the worker's account. If this is impossible and not allowed by the network administrators then you have to modify the `worker-start.bat` and establish the network connection before starting DaVis. Open file `worker-start.bat` with a text editor, remove the `rem` from the following line and replace `server` and `share` by your server name and share name and change drive `S:` to a drive you want.

```
rem net use S: \\server\share /u:%USERNAME% /persistent:yes
```

Entering the password for connecting to the file server might be needed. In this case modify the line to the following syntax and enter the password at the marked position:

```
net use S: \\server\share <password> /u:%USERNAME% /persistent:yes
```

2.2.7 Two workers on a multi core Windows PC

On a Windows PC with more than 64 cores the operating system splits all cores into at least two groups with both the same number of cores. An application is applied to one of the core groups and usually uses the cores of this group only. E.g. a PC with 80 cores gets two groups of 40 cores each. To be able to use all cores for distributed processing a separate **DaVis** must be installed for every group.

Just run the standard worker installation. Then open the root folder of the distributed processing software as defined during the installation. Create a copy of folder `DaVisWorker` in the same root folder and e.g. rename the copy to `DaVisWorkerGroup1`. Now modify the start script `worker-start.bat`, uncomment the line with affinity definition and define the group index:

```
set DP_AFFINITY=--process-group-affinity=0
```

Store the modified file, create a copy of the file and rename it e.g. to `worker-start-Group1.bat`. Edit the new file and modify both the core group affinity

```
set DP_AFFINITY=--process-group-affinity=1
```

and the path to the **DaVis** folder:

```
:start
echo Starting the DaVis worker...
set WORKERPATH=DaVisWorkerGroup1\win64
```

Store the modified file. Then open the Windows task scheduler and select the **DaVisWorker** task which had been created during installation. Open the **Properties** of this task and change to the **Action** options, see figure 2.8. Add a new action and select the created batch file.

After starting the task manually via **Run** item from the menu both **DaVis** workers should start. Verify their process affinity in the Windows task dialog. After sending jobs to both workers the Windows task dialog should show high CPU usage on all cores.

2.2.8 DaVis Worker on Linux

Download the **DistributedProcessingWorker** archive from the **LaVision** website `http://www.lavision.de` as registered customer. Create a new

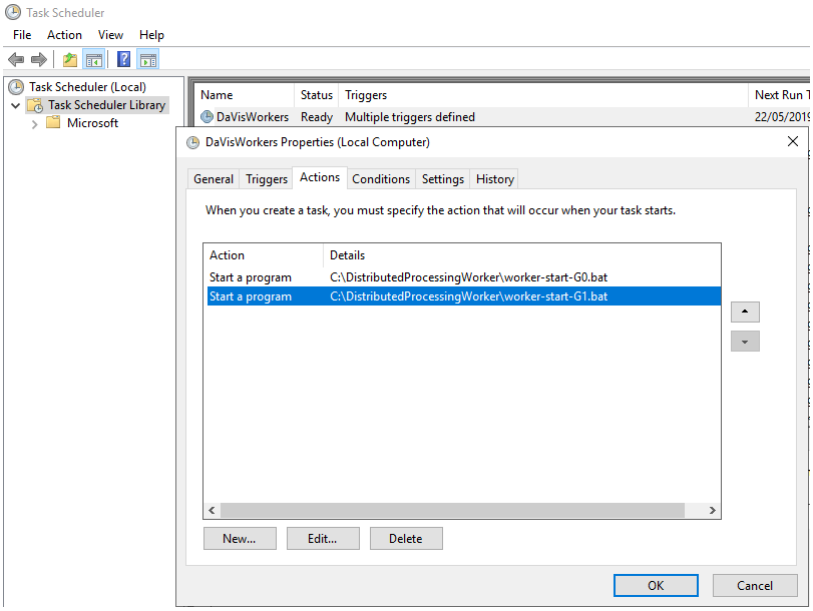


Figure 2.8: Windows task scheduler: Action options for system with two DaVis workers.

folder on the new worker node. The user who installs and who runs the worker must be able to access all used network shares with the **DaVis** projects which should be mounted at the node. An installation of the dongle driver is not required. Extract the archive and call `./davis-setup.sh worker <dongle-id>` with the dongle identifier of the used network dongle which gives the worker licenses.

To install the worker like a service and start the worker automatically after startup of the system please change into folder `DaVisWorker` and call `sudo ./davis-setup.sh service`. This command will install a cron job for the current user and tries to start the DaVis worker all five minutes. Special arguments for the worker can be given in the first lines of file `davis-setup.sh`, see section below on page 29 about the details.

All log files are stored in files `<user-home>/.config/LaVision GmbH/DistributedProcessingWorker-<n>.log`. The main log file can be watched e.g. by `tail` to control the activity and possible error messages:

```
tail -f ~/.config/LaVision GmbH/DistributedProcessingWorker-0.log
```

Don't forget to setup the drive mapping that allows the worker to access common file shares with the Windows master **DaVis**. See page 67 for details.

If a firewall is used the worker application must be allowed open network connections. At least the broadcast port should be available for incoming traffic.

2.2.9 Manual update

The manual update should be installed in case of the full local installation of DaVis, dispatcher and worker only. When updating several workers and the dispatcher, the automatic update should be used as described on page 42.

Windows

Updates of the software can easily be done after downloading the latest installer from the **LaVision** website: Start the installer, which will detect the old version and then update the old software using the old configuration. This will work for all installers: DaVis with dispatcher and worker all together, also for the standalone dispatcher and for the standalone worker.

In case of the full installation on the same PC all three applications are always of the same version. When starting the update procedure the installer searches for a running DaVis instance and asks the user to stop it. In case of the worker the application could have been started automatically during startup of the system and therefore is not available on screen. Please open the task manager, display the complete list of all tasks including the background tasks and search for the `DaVis.exe` in the list of background tasks. Then kill the application. Another way would be to open the **Control App**, change to **Details** and then to **Worker activity**, select the worker running on the local PC, press the right mouse button and select **Shutdown**.

The version of a standalone installation of a worker can be checked with the control dialog in card **Details - Clients** or with the Windows explorer showing the properties of the `DaVis.exe` or in the **Apps and Features** dialog of Windows.

Linux

Stop the dispatcher or worker service. Then remove the contents of the root folder for distributed processing which has been created during installation.

Extract the new archive in this folder, execute the setup bash script and then start the service again without new installation.

2.2.10 Uninstall

Windows

Uninstallation of the software is done via the standard Windows program management. Afterwards the folders for log files must be removed manually from folder `%APPDATA%/LaVision GmbH` and also the **DaVis** folder should be removed, which might contain some files created at runtime. The scheduled tasks and the firewall rules are removed automatically.

Linux

Stop the dispatcher or worker service and remove the cron job by calling `crontab -e` and deleting the configuration. Then remove the root folder for distributed processing which has been created during installation. Also the log files should be removed from folder `~/.config/LaVision GmbH/`.

2.2.11 Configuration of dispatcher and workers

Some arguments of dispatcher and workers are available in the start scripts only. See batch files `worker-start.bat` and `dispatcher-start.bat` in the Windows root folder of the installation,
`davis-start.sh` for the Linux worker and `start-dispatcher.sh` for the Linux dispatcher.

The available arguments are explained by comments in the scripts. Uncomment the arguments and fill their values to use them after a restart of the application. In the Windows batch files uncommenting means to remove `rem` and in the Linux shell scripts to remove the # symbol in the beginning of a line.

Dispatcher

The arguments of the **dispatcher** are:

- **--admin <user1>,<user2>,...**: Define the account names of users with admin privileges. Those users are allowed to change priority or break jobs of other users and to stop or restart workers. The admin users also are allowed to upload a patch file for the automatic update of the software. If no admin is defined, then all users get these privileges.

- **--mail-sender davis@localhost**: If a standard mail system like sendmail or postfix is installed on the PC of the Linux dispatcher, the dispatcher can send email notifications for finished jobs. The notifications are enabled by using this argument and defining the account name of the notification sender.

- **--mail-server <name:port>**: Define server name or IP address and port of the mail server to be used for sending mails. By default `localhost:25` is used.

- **--broadcast-port <port>**: This port must be equal for all Master **DaVis** and all workers. It is used to find each other and must not be used by any other application on the dispatcher's PC. The port number is 20049 by default.

- **--websocket-port <port>**: This port is opened by the dispatcher for communication with connected master and worker applications. It must be different to the broadcast port and must not be used by any other application on the dispatcher's PC. The port number is 20050 by default.

- **--local-clients-only**: If given then the dispatcher just accepts connections from master and client on the local PC. This option is enabled by the installer for master, dispatcher and worker on the same PC in local mode.

- **--send-job-after-connect <0/1>**: If enabled (value 1, this is the default mode) the dispatcher sends jobs to a new worker immediately after connection. If disabled (by value 0) jobs are sent after a while. This may be useful for testing in case of problems.

- **--activate-worker-after-connect <0/1>**: If enabled (value 1, this is the default mode) the dispatcher activates a new worker and could send a jobs immediately after connection. If disabled the user has to activate the worker manually in the control dialog.

- **--slow-worker-detection <min job time> <allowed speed factor, 0 = no timeout detection> <max slow counts>**: The first argument gives a timeout in seconds for each worker. If the worker does not send any progress about the active job to the dispatcher for at least this time, then the dispatcher removes the job from this worker and disables the worker. The second argument can be set to 0 to disable the timeout detection completely. Otherwise this value defines a factor for the processing time of each worker compares to the average processing time of all workers. If a worker process takes too long compared to the average processing time, then the worker is disabled.
- **--kill-other-dispatchers**: If defined then run this dispatcher as oldest dispatcher and kill other dispatchers in the network working on the same port.
- **--parallel-update-max <value>**: Restrict number of parallel updating workers, default = 5. If 0 then no limit. Used to decrease the network traffic when updating lots of workers at the same time.
- **--disable-self-update**: Disable self update of the dispatcher, used for tests of the update system.

Worker

The arguments of the **worker** are:

- **--broadcast-port=<port>**: This port must be equal for all Master **DaVis** and all workers. It is used to find each other and must not be used by any other application on the dispatcher's PC.
- **--worker-activity=<hours>**: Days and hours of activity. E.g. `18-6,sat,sun` restricts the activity of the worker to the night hours and to the weekend.
- **--worker-priority-normal**: By default the **DaVis** worker is starting with low priority on Windows systems. Use this option to start with normal priority.
- **--process-group-affinity=<group>**: Should be used on systems with more than 64 cores: Define the group of cores to be used by **DaVis**. The group can be 0,1,...

- **--local-dispatcher-only**: Worker is running with local dispatcher only and does not connect to another dispatcher on another PC.
- **--local-master-only**: Worker does not accept jobs from foreign masters, the dispatcher must send jobs from local master only.
- **--worker-users=<name1,name2>**: Worker accepts jobs from the given users only. Binds the worker to given users.
- **--disable-auto-update**: Worker is not allowed to install update from dispatcher. Used e.g. for dual clients.
- **-noruncheck**: Used to disable the security check on a **DaVis** executable started twice. Needed to start two workers from the same folder.
- **--disable-auto-update**: Auto update should be disabled when running e.g. several instances from the same folder or from a non-default folder like in systems with more than 64 cores.
- **-logpath=<path>**: Path for logfiles of the worker. The '##' is replaced by ' ' internally to avoid problems with double quoted parameters. Default path in Windows format: `%APPDATA%\LaVision##GmbH\Worker`

2.3 Preferences

There are some preferences available for distributed processing, see figure 2.9. Press the **Extras** button in the **DaVis** toolbar on the left, then select **Preferences** and in the preferences dialog in the left column select **Processing**.

The usage of distributed processing can be enabled or disabled. In the processing dialog the button to send a job to the dispatcher is visible only in enabled state. Also the control dialog is available in enabled state only.

The **broadcast port** is required for the communication between dispatcher, master and worker applications. The port number is 20049 by default and must be equal for all those applications in the local network. Otherwise master or worker can't establish a connection with the dispatcher. See next section for a sometimes useful change of the port number. Of course the port number may be changed on request of your IT department.

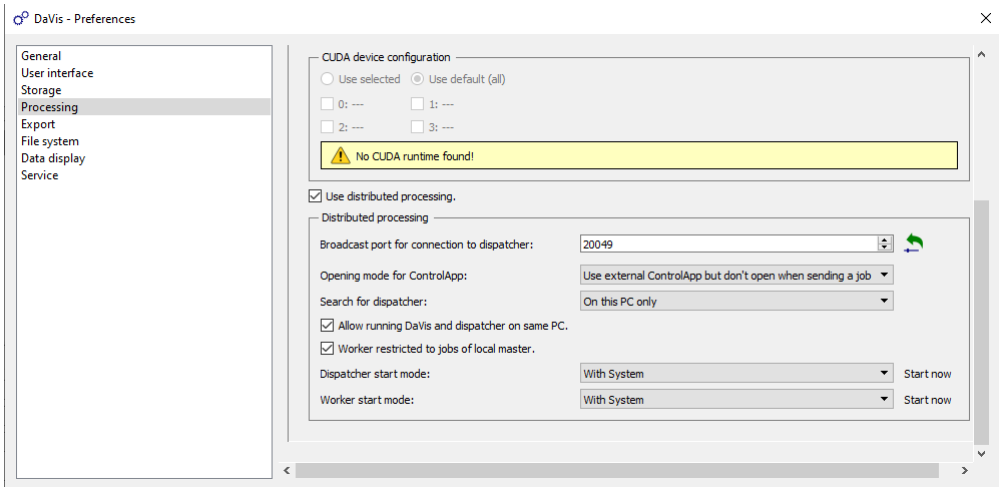


Figure 2.9: Preferences for distributed processing.

The **Opening mode for ControlApp** can be changed between automatic opening whenever a job is send from the processing dialog to the dispatcher or a silent mode, when the processing dialog does not open the control app. Also the control app can either be used as stand alone program next to **DaVis** or as part of **DaVis**.

Option **Search for dispatcher** allows to switch the Master **DaVis** between local mode and network mode. This is useful e.g. when running the full local installation with dispatcher and worker on a notebook and use the local mode while being offline. When returning to the company network switch to the network mode and use the dispatcher from the company network and all available workers in the network.

Note about switching between local and network mode: When the mode is changed a already running local worker will be asked to restart and connect to the new dispatcher. The local dispatcher is asked to stop and it's start mode is changed to **manually** to make sure the local dispatcher does not start again automatically. Please verify if the restart really happened, there are some rare occations when the worker doesn't restart. And verify in the task manager that the local dispatcher really stopped, or kill the `DistributedProcessingDispatcher.exe` process manually. Otherwise DaVis and the worker could contact the local dispatcher or the network dispatcher in undefined behaviour.

Allow running on same PC must be selected when the dispatcher or a worker is installed on the same PC together with the Master **DaVis**.

Worker restricted to jobs of local master is useful in a network with several Master **DaVis** installations using the same dispatcher and the same workers. In this case the local worker of PC 1 could be used be a job created on PC 2. But this could lead to high CPU usage and make other work on the local PC impossible for the locally active user. If this mode is enabled, only jobs created by the same account as running the worker will be processed locally. Jobs created from other accounts and from another PC will not be given to the local worker. This mode is always enabled when installing all applications together on the same PC.

The **root path of local installation** is defined during the installation of the local system with master, dispatcher and worker on the same PC and stored in the registry. For easy verification about the concrete used dispatcher and worker the value is given here in the dialog.

The **dispatcher start mode** and **worker start mode** can be changed when dispatcher and worker are installed on the same PC with the master DaVis. Depending on the user input during installation both dispatcher and worker are started by a scheduled task together **with system** startup and before a user logs into Windows. Sometimes it may be useful to start dispatcher and worker together with the Master **DaVis** (option **with DaVis**). If the user doe not need local distributed processing in every session, mode **by processing** might be the best choice to start local dispatcher and worker when sending the first job to the dispatcher. Last choice is to start the worker **manually** by clicking the **Start now** buttons in the preferences dialog. If not installed to be started with the system, a later change to this mode requires a new installation.

Maximum size of a subset given to a worker allows the dispatcher to restrict the size of parts of the source set given to each worker. By default the value is 0 and disables the limit. Otherwise a worker would never get more than the given number of source data in one step of the distribution. This limit is useful when processing very large sets with thousands or even 10.000s of images. If for example a set has a size of 40.000 images and is distributed to three workers, then each worker would get e.g. 8.000 images in the first step. The dispatcher does not distribute the complete set to all available workers in equal parts but spares some parts for the second step e.g. awaiting more workers to start later or allow fast workers to get another part while other workers have not finished. In this example with three workers of the same speed the two spare parts would be given to the first workers finishing the first step. In the second step the third

worker would be idle. With a restriction of the maximum set size e.g. to 2000 there would be many spare parts and the overhead of the last parts would be much smaller.

2.3.1 Multiple separate dispatchers and workers

In local networks with different work groups but all customers of **LaVision** and all using the distributed processing but with different network dongles and different accounting, the separation of dispatchers and workers might be a good idea. The easiest way to separate the systems is changing the port number of the network communication.

Therefore the port number must be changed in all Master **DaVis** installations. Also change the port number in the start scripts of dispatcher and all workers. During installation of the workers the different network dongle identifiers must be given.

2.4 Processing Dialog

If **Distributed Processing** has been enabled in the **Preferences** dialog, see page 31, the **Processing** dialog displays an additional button next to **Start** and **Test**, the **Distribute to dispatcher** button, see figure 2.10.

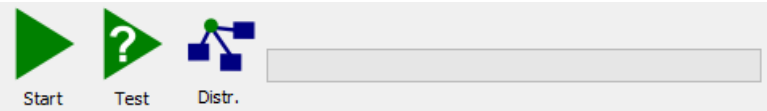


Figure 2.10: Button bar in the **Processing** dialog for **Distributed Processing**.

Pressing the **Distr.** button sends the defined processing operation list with the current **source sets** and the selected **range** and **increment** to the dispatcher for processing by the connected workers. The **Control** dialog, see page 36, opens and in case of an idle system the dispatcher immediately starts to distribute the operation list to the workers. The progress of processing can be monitored in the **Control** dialog. After finishing the operation list, the **Project Browser** of **DaVis** displays the results. Intermediate results are hidden and neither displayed in the **Processing** dialog nor in the **Project Browser** while partially calculated and partially stored in the project folder. In case of an email notification the dispatcher sends

such an email with information about the project, source set and operation list to the email address defined in the **Preferences** dialog, see page 31.

2.4.1 Shared folders

In the local installation of master **DaVis**, dispatcher and worker no shared folders are needed. The project folder can be located in a local drive or in a network share.

In case of the network configuration a job can be sent to the dispatcher only if **DaVis** detects a share of the project folder! This means the project folder must either be located on a network share or the local project folder must be shared to network users. Some instructions to setup those shares are given in a chapter on page 64. **LaVision** always recommends to talk to the IT department about creating and using shares! Most important about those shares is the ability of master and all workers to use the project with full write access.

2.4.2 GPU usage by dispatched jobs

So far the **PIV** operation is the only operation which takes advantage of GPU usage during distributed processing, see figure 2.11 for the parameters. The **distributed processing strategy** in the **Methods** card of PIV parameters can be switched between **CPU only (all computers)** and **If GPU available...**. The first mode simply will not use the GPU at all even if the local PC running the Master **DaVis** has a GPU and GPU usage is enabled. This mode uses all available workers regarless of their availability of a GPU at the worker PC. The second mode restricts processing to the workers with GPU and with GPU license. Workers without GPU are not used.

The advantage of this mode is to even allow a master PC without GPU the usage of workers with GPU! Anyway the master PC needs the license for PIV on GPU.

2.4.3 Remarks to the calibration

There is only one active calibration in the project. The calibration must not be changed while running jobs of the current project via distributed

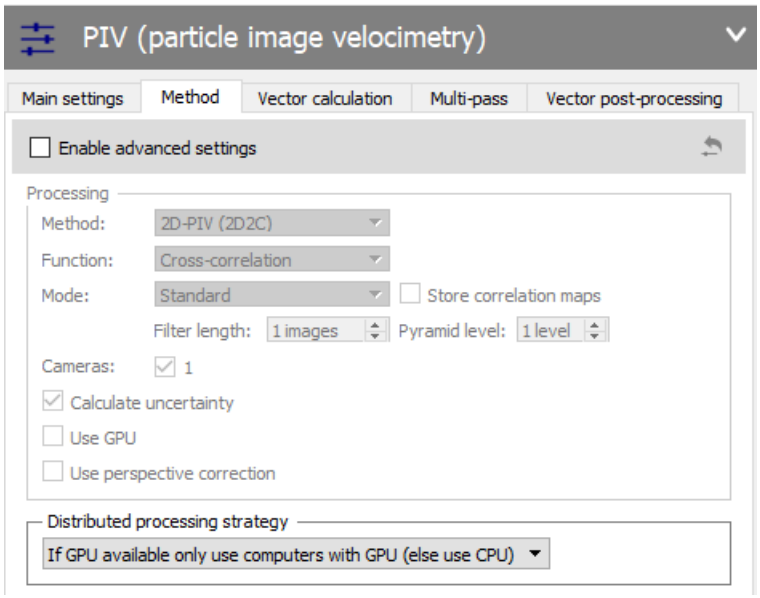


Figure 2.11: Special settings for **GPU** usage in the **PIV** operation.

processing. Workers are always loading and using the current active calibration at the moment when they are starting to process a job. When the active calibration is changed while a job is running, parts of the source images could have been processed using the old calibration while later processed parts are based on the new calibration.

2.4.4 Operations with local files as parameters

Some operations allow references to local files as parameter, e.g. the PIV operation with a reference vector field. This file is allowed to be located anywhere, it does not need to be part of the shared project folder. The distribution system copies the file before sending the job to the dispatcher into a location inside the project's properties folder and therefore the file is available to all workers.

2.4.5 Operations with restrictions to the operating system

In this **DaVis** version some operations can be processed by workers on Windows PCs only, e.g. Spray Geometry and some other operations from **Imaging** projects. A full list of those operations is available on page 69. But all PIV, Tomographic PIV and Shake-the-Box operations can be processed by Windows and Linux workers.

2.4.6 Operations disallowing distributed processing

There are a few operations which can't be sent to the dispatcher at all like export operations, which usually store their results on the local PC and not in the project folder structure. These are especially the **Export** operations in case of export to a local folder. See details on page 69.

2.4.7 User operations

The **Processing Wizard** allows to create CL files with operations by the user. Also the user can call own C-code functions compiled in a DLL, see the chapter about the DLL functions in the *Command Language* manual. This functions are executed via CL command `CallDll` and can be embedded into both the generated CL files from the wizard and into the simple CL operations **Execute CL macro**.

While the latter operation can be used for distributed processing without restriction, the CL files from the wizard located in the `CL_Autostart` folder and own DLLs must be transfered manually to the workers. When using Linux workers and own DLLs, the code must be compiled on Windows and on Linux systems!

2.4.8 Attribute ProcessedByHost

To verify the software version and the used worker for each processed result a special attribute is created and stored with the result files by the workers. This attribute is named `ProcessedByHost` and can even be displayed as overlay on the results like in figure 2.12. The value gives information about the used worker with its PC name and IP address, the operating system and the software version.

2.5 Control Dialog

The control dialog can be opened from the **Extras** menu as **Dispatcher control**. It also opens automatically when sending a job to the dispatcher in the **Processing** dialog and the **Preferences** are set to automatic opening. There are stand alone executables available for Windows, Linux or Mac OS which don't need a full installation but can be used by just extracting an

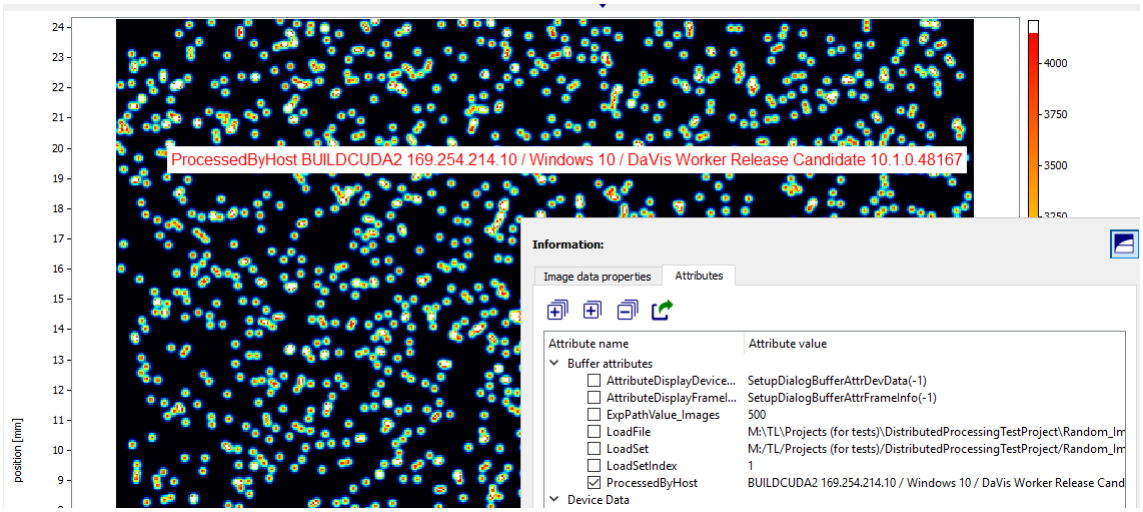


Figure 2.12: Attribute `ProcessedByHost` displayed as overlay.

archive. For the Mac OS version please contact **LaVision** service because it is not included in every new release version yet.

Note about mixing different versions of dispatcher and control app: Between software versions 10.1.2 and 10.2.0 the network protocol between dispatcher and control app changed! The amount of data to be transfered for display of the tables in the control app got reduced by aboout 80 percent. Control apps of a 10.1.x version are displaying incomplete columns when used with a dispatcher of 10.2.x and vice versa. In this case please update all parts of the software.

By default the Overview page in figure 2.13 is displayed, which shows all active jobs of the current user with their processing state, either waiting or in progress with a percentage value or paused. The buttons on the right of each job can be used to pause and resume a job or to cancel a job.

The bottom line of the dialog gives information about the connected dispatcher with it's IP address or the **disconnected** text if either the dispatcher is not running or can't be connected e.g. due to network problems or to a missing access rule in the local firewall. The other values give the number of available workers and of workers with GPU support.

Change to the **Details** card to get more information about the clients connected to the dispatcher, about the worker activity, about active jobs, waiting jobs and finished jobs.

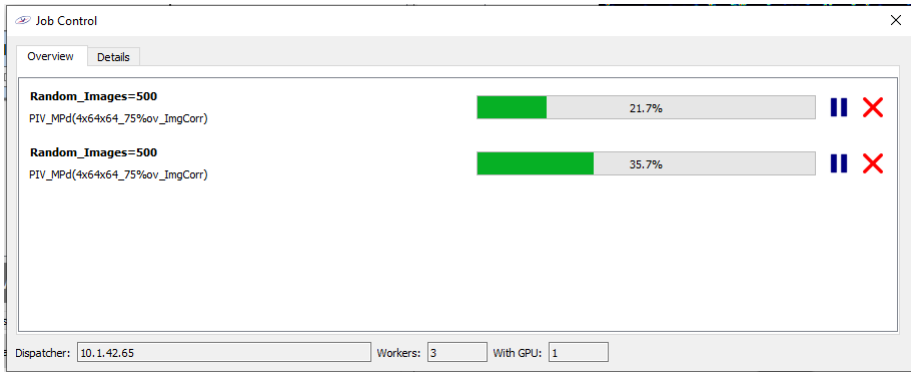


Figure 2.13: Overview page in the dispatcher control dialog.

2.5.1 Clients

The **Clients** card gives information about all connected workers and control applications, both Master **DaVis** and the stand alone control app. For each client the PC name is given together with the number of GPUs in the system, the **DaVis** version, time stamp of the last contact to the client, usage of CPU and memory by the system, and about the client state: This can be **control** for a Master **DaVis** or control app, **idle** for a worker in idle state, **processing** or **error**. In case of the error state a detailed error message is shown in card **Worker activity**. The special **updating** state is displayed while the worker is downloading the update archive, see page 42 for details.

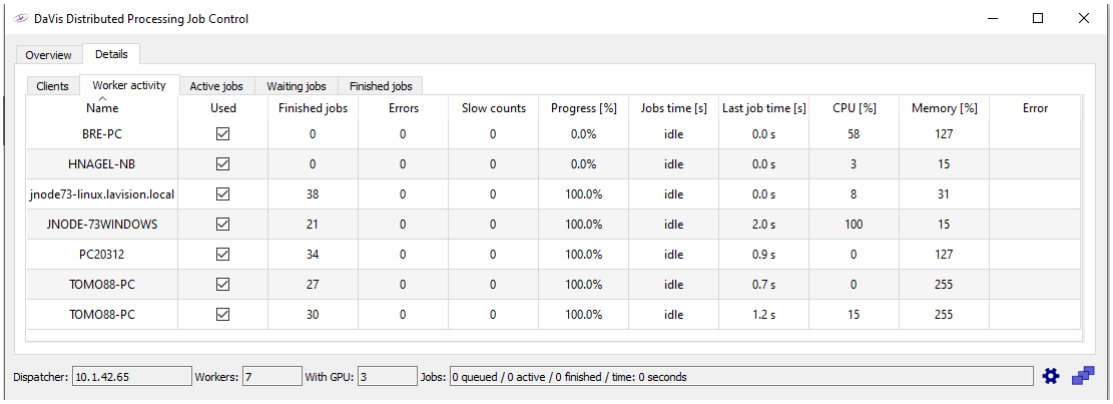


Figure 2.14: Information about clients in the control dialog.

2.5.2 Worker activity

The card of figure 2.15 shows all connected workers with their processing state and possible error messages. With the checkbox in the second column

a worker can be disabled or enabled. A disabled worker will stop at least after a short time with it's current job and the job is distributed to another worker by the dispatcher. By disabling and enabling the error state is reset. This can be useful e.g. if a share has not been mounted like in figure beyond and the problem is solved without need to restart the worker.

Disabling a worker and also opening the context menu via right mouse click to a worker's row is allowed for users with admin privileges only. The context menu gives access to commands to shutdown or restart a single worker, to restart all workers and to update all workers. See section on page 42 for details about the automatic update system.

Another option used in service cases is the storage of service files by a single worker or by all workers. The chosen destination path must be available in the network by all workers. E.g. select a workspace folder, containing the shared projects, and create a service folder as destination. Each worker will send the name of the created service file to the control app. Afterwards all `lsfx` files can be zipped again and send to **LaVision** service department.

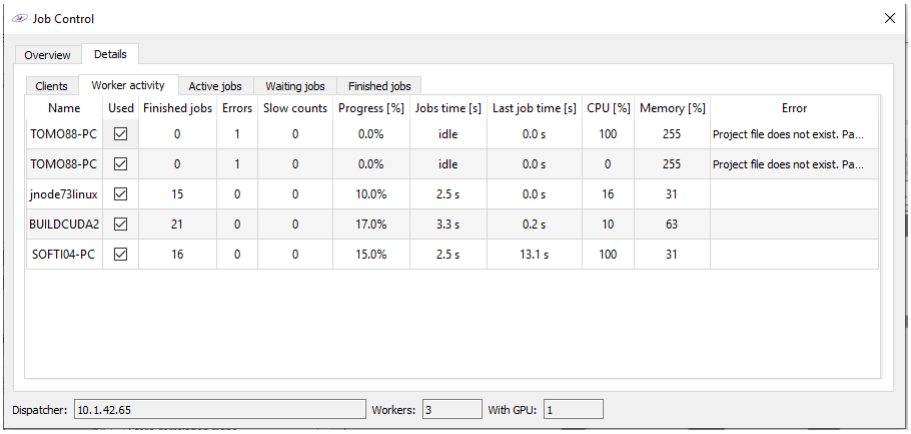


Figure 2.15: Information about the current worker activity.

2.5.3 Active jobs

The card in figure 2.16 shows all jobs which are just in the processing state. This is like a detailed description of the Overview page described on page 38, but this card also displays the active jobs from all other users. Additional informations are the current progress and the current processing time of each job.

The owner of the job and admins are allowed to pause and resume the job by clicking onto the checkbox in the **active** column. Via context menu those users are also allowed to **cancel** a job or to **cancel all** jobs. Processing of a canceled job will stop and the intermediate results will be visible in the **DaVis** project browser. The resulting set is incomplete, parts of the source range will be missing in the result. In case of longer operation list only the first operations might be processed already.

A special option of the context menu is to **kill all** jobs: This will simply clean both the active job list and the queue of waiting jobs, but temporary results and the job definitions will stay in the project without getting removed. The user has to clean the folders in the projects manually, especially `Properties/Processing/Jobs` and also the hidden result sets in the folders of the source sets.

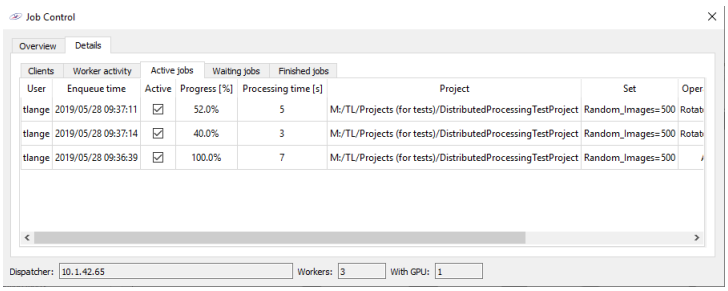


Figure 2.16: Information about all active jobs.

2.5.4 Waiting jobs

The card of figure 2.17 shows all waiting jobs in the queue of sent but not started jobs. The table gives the name of the user who enqueued the job, the enqueue time and the names of project, source set and operation list, which is automatically determined from the default names of the included operations.

The first column shows the name of the queue the job is sent to. By default all jobs are enqueued into the **normal** queue but can be moved via context menu into the **high** or **low** queue. This queue change is possible for the owner of the job and for admins only. When starting to process the next job the dispatcher first looks into the high, then into the normal and if both are empty into the low queue. This way the user can reorder the jobs after sending them.

The context menu also allows to cancel a job.

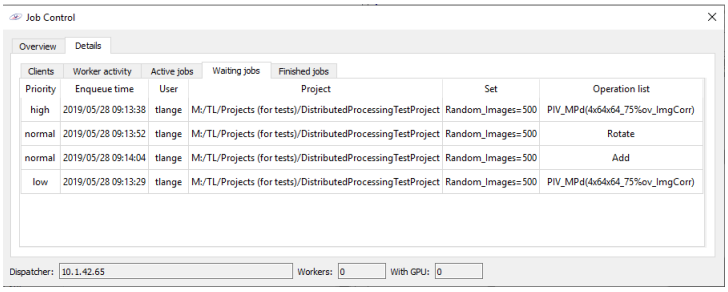


Figure 2.17: Information about all waiting jobs.

2.5.5 Finished jobs

The card of figure 2.18 shows all finished jobs by all users of the last day. The dispatcher automatically removes old jobs after some time. For each job the user name, enqueue time and finish time are given together with the processing time. A job is also described by the project folder, name of source set and name of the processing list, which is automatically determined from the default names of the included operations.

The **total processing time** sums up all partially distributed jobs of all used workers. This time can be compared to the local processing time by a stand alone **DaVis**.

The **active processing time** counts from the beginning of the distribution by dispatcher to the workers until the end of processing. This does not take the waiting time in the queue into account. The time decreases compared to to the **total processing time** depending on the number of used workers and on the performance of those workers.

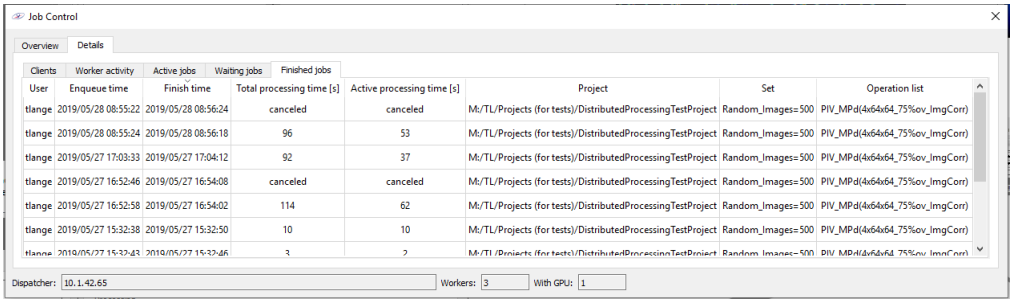


Figure 2.18: Information about all finished jobs.

2.5.6 Preferences

With the small button on the right in the bottom row the Control App can open the small properties dialog of figure 2.19. This dialog is available in the stand alone app only and not in the internal control dialog of **DaVis**. The options are about the **broadcast port** and about the **local or network search** for a dispatcher, see details on page 31 about the same settings in **DaVis**.

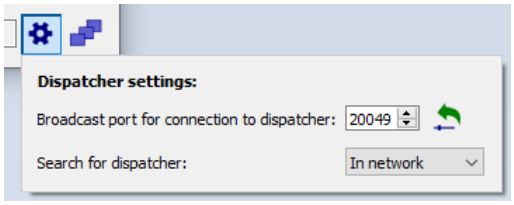


Figure 2.19: Connection properties of the control app.

2.5.7 Automatic update of workers

The automatic update must not be used for the local installation of master **DaVis**, dispatcher and worker on the same PC! In this case please use the standard update function of the **DaVis** installer `SetupDaVis.exe`!

In case of a manual multi worker installation of more than one worker on the same server the update must also be executed manually!

Admin users are allowed to update the dispatcher and all workers. The button next to the bottom right corner of the dialog, see figure 2.20, gives access to the **Update all workers and dispatcher** item, which opens a dialog to select the **LaVision DaVis worker update** archive with file extension `lwu`. This archive can be downloaded from the **LaVision** website `http://www.lavision.de` by registered customers.

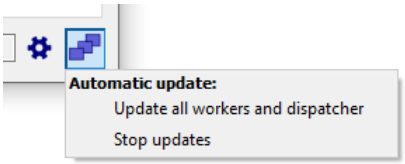


Figure 2.20: Starting or stopping automatic update of dispatcher and all workers.

The archive is verified after selection and then uploaded to the dispatcher, which stores all `lwu` files in the local settings folder. If the archive contains

another version then running on the dispatcher, the dispatcher extracts the archive, updates itself and restarts. The update is possible in both direction, returning to an older version and installing a new version.

After restart and reconnecting to the workers the dispatcher verifies the software version of each worker. If an update is needed, the worker downloads the archive from the dispatcher, patches itself and then restarts. The update archive includes binaries for all supported operating systems. Whenever a worker starts and connects to the dispatcher, the version verification is done and the worker is asked to update.

If the dispatcher did not receive an `lwu` archive so far but a worker is connecting with a wrong version number, the usage of this worker depends on the *version distance*, i.e. the dispatcher allows workers to process jobs if the difference in the version number is in the third value. E.g. a dispatcher version 10.1.0 works together with workers of version 10.1.x but not with workers of version 10.2.y.

Note and always think about possible changes of algorithms and fixes of failures in the different software versions! When using a new version of the **DaVis** Master but older versions of the workers, the changes could lead to different results if processing on the local master only or with the workers.

Cancel update

The uploaded automatic update can be canceled by admin users by selecting item **Stop updates** in bottom right corner of the dialog, see figure 2.20,. The dispatcher will remove all cashed archives and no longer ask workers with another version number to update themselves. This option is useful when e.g. installing several workers with a special version for tests but already used update archives before. After installing a worker the auto update option can also be disabled by a program argument, see page 29.

2.5.8 Update feature flags

Feature flags enable some special or new functions for certains customers. When a new function in **DaVis** is developed and given to some customers for testing, the function is enabled by a feature flag and not enabled by default for all customers. Another usage of the feature flags are special modes of operations or even complete operations, which are needed by

some customers only but don't require a special license. The configuration of those features is defined by a `FeatureFlags.xml` file and sent by **LaVision** developers or service to the customer.

If a customer gets such a feature and has to enable it for all installed workers, the new software version must be installed at first as described above. Then use the control app, press the button next to the bottom right corner of the dialog and press option **Update feature flags**. Select the `FeatureFlags.xml` and start uploading to the dispatcher, which will transfer the file to all available and running workers. Afterwards the workers are restarting. In difference to the software update procedure the dispatcher will send the file once to the connected workers. Offline workers will not receive the file.

2.6 Helpful batch scripts

2.6.1 Scripted Start of Jobs

Jobs with pre-defined operation lists can be sent to the dispatcher without using the master **DaVis**. See the following example of a bash script for Linux. This uses Windows drive `S:` mapped to Linux mount point `/mnt/server-share`. When using Linux workers only, the drive mapping could be commented and the paths could be given in Linux style. All commented parameters are optional.

The `START_AT` or `"at="` parameter is new in **DaVis** 10.1.2 and useful for testing the behavior of the system at different day times or weekdays. It might be interesting to compare the processing time depending on the general network traffic or traffic on the file server.

```
#!/bin/bash
PROCESSING_XML="S:/MyOperationLists/operation-list-name.xml"
PROCESSINGMODE=0
PROJECT_PATH="S:/MyProjects/Imaging"
SOURCE_SET="MyRecording"
FIRSTFILE=0
LASTFILE=9
STEPFILES=1
DISTRIBUTABLE=1
OPLISTSIZE=2
```

```
OPALIAS="alias=MyAliasName"
#REQUIRE_GPU=gpu
#REQUIRE_WINDOWS=windows
# Delayed start today at 11:30:  "at=1130"
# Delayed start tomorrow at 11:30:  "at=11130"
#START_AT="at=2000"
DRIVE_MAPPING="-drive-mapping S: /mnt/server-share"

./DistributedProcessingControlApp $DRIVE_MAPPING
    -enqueue-job "$PROJECT_PATH" "$SOURCE_SET" "$PROCESSING_XML"
    range=$FIRSTFILE-$LASTFILE-$STEPFILES listsize=$OPLISTSIZE
    $REQUIRE_GPU $REQUIRE_WINDOWS $OPALIAS $START_AT
```

The same program arguments are available for the Windows version of the DistributedProcessingControlApp. A batch script (bat file) is not given here, but would work the same way.

2.6.2 Scripted Restart of Windows Worker PCs

The **DaVis** worker is not able to reboot the PC. In case of a processing bug the **DaVis** worker could become unbreakable, meaning even with the restart option or by disabling a worker in the control app the worker continues processing.

In both cases the worker or the PC must be restarted manually. One way would be to logon to the PC, kill the **DaVis** process or reboot the PC. The following commands can be executed from another PC by a user with full access privileges to the worker PC to kill **DaVis** and restart the PC afterwards. Just replace `PC-NAME` by the real computer name.

```
taskkill /S PC-NAME /IM davis.exe
shutdown /r /m \\PC-NAME
```

To run this job for all worker PCs the following shell script can be used. The example is defined for 20 worker PCs with names `PC-NAME-01` to `PC-NAME-20`.

```
for /L %%X in (1,1,9) do (
    echo Restarting 0%%X
    taskkill /S PC-NAME-0%%X /IM davis.exe
    shutdown /r /m \\PC-NAME-0%%X
```

```
)
for /L %%X in (10,1,20) do (
    echo Restarting %%X
    taskkill /S PC-NAME-%%X /IM davis.exe
    shutdown /r /m \\PC-NAME-%%X
)
```

3 Cluster Script Processing

3.1 How it works

The basic idea is to create a shell script and parameter files by the known **DaVis** processing dialog. This script can be executed on a Linux cluster e.g. via `mpirun` command when using OpenMPI.

For a PIV processing operation this would process each image on another node of the cluster. Depending on the number of sources images and the number of nodes, the images are distributed more or less equally to the nodes.

This processing mode needs a special license for script generation on the **DaVis** Master! Please contact our sales department to get more information about the recent state.

3.2 Installation

For script generation the standard **DaVis** installation on Windows is required.

See page 59 about the worker installation on Linux nodes.

3.2.1 Network Configuration

Master PC and Linux worker PCs don't need to run in the same network. There is no direct communication between Master- and Worker-**DaVis**, but somehow the generated script and the project data must be available on both sides either via direct access to a file share or as copy.

3.3 Processing Dialog

The **Processing** dialog displays a new button to open a setup dialog for some additional settings. When all settings are defined, press button **Create** to create the script files.

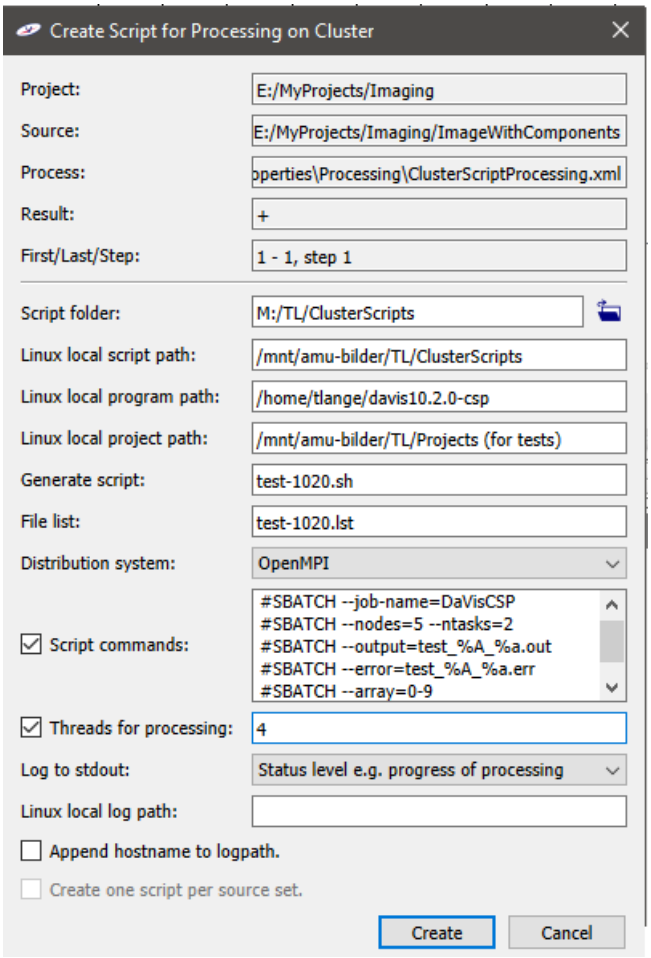


Figure 3.1: Setup parameters for Cluster Script Processing.

The top area of the configuration dialog from figure 3.1 gives information about the used project, the selected source set, the name of the result set and the location of the active processing list in the file system. Those values cannot be changed in the dialog and must not be modified later in the generated script.

The parameters of the second block must be defined by the user especially to give information about the path names seen from the Linux node: The **script folder** can be changed using the file select button on the right and defines the location on harddisc or better on a network share to be used to

store the generated script files. Every Linux node must have access to this share unless the user wants to copy everything by hand to the cluster. The local path on every Linux node to the script folder must be given one line below.

The **Linux local program path** is stored in the script to start **DaVis**. This path must be valid and equal on the complete cluster. The **Linux local project path** references the location of the project root path used by **DaVis** on Windows.

The **script name** should be changed when creating different scripts for different sources or for different operation lists. Enter a name for the **file list** if all files should be copied from the Master PC to the cluster. This text file is filled with the names of all required files, e.g. the project file and all files of the source set and the calibration of the project. The list can be used to copy everything which is needed for processing.

Distribution system changes the names of the environment variables for node index and total number of nodes. **OpenMPI**, **Sun/Oracle Grid Engine** and **Slurm Workload Manager** are predefined by the script generator. When using another system both names can simply be changed in the generated script. In this case please don't hesitate to contact **LaVision** to add the names from your distribution system.

Some **Script commands** can be defined and stored in the header of the generated script. This is useful e.g. for `sbash` commands of **Slurm**.

By default **DaVis** uses as much processor cores as possible during processing. Many algorithms like PIV or Tomographic PIV are running multi threaded and using all cores of the CPU(s). Sometimes on shared systems a process should not take all but only a part of the available cores. In this case enable **Threads for processing** and either enter a fixed number or the name of an environment variable, which is defined by the distribution system and should be used by **DaVis**. Some more descriptions are given in section 5.5.

The last settings are about logging: Usually **DaVis** gives lots of information to `stdout`. This output can be switched off completely or restricted to important information only like the active set and the index of the active file in the source set. The most detailed mode should be changed to testing only, this mode even prints progress information and intermediate information of the processing.

The **Linux local log path** should be defined in case of problems to store the log files of every node in a special directory. Optionally the name of the node can be added to the given path. In this case the used log path is like `/mnt/share/log/node-$HOSTNAME`. Those environment variables can be uses in the path definition, too.

3.3.1 Generated files

The script is generated in the above defined folder together with some other files and folders. In the example the script name is `test-1020-piv.sh` and the additional files are stored in `test-1020-piv.xml` and in folder `test-1020-piv`.

The values of the path and file names stored in environment variables are not allowed to be changed in the generated script. **DaVis** will detect this and stop with an error message. To run the script on another set of data or with another operation list, a new script has to be created with the changed settings.

Changes are allowed to the last line of the script with the start command for **DaVis**. Feel free to change the path of `davis`. The order of all other arguments especially in the `-csp` block is fixed and changed will cause error messages during startup.

The meaning of environment variable `PROCESSINGMODE` is 0 for the distribution of one source set to different instances of **DaVis** or 1 for the full processing of the source set by one instance only. When using Slurm the start of the script may differ between both modes. In the second case use one task only and start the script via srun instead of sbatch.

3.3.2 Copy all files to the cluster

Nothing has to be done if the cluster machines have direct access to the generated files and to the project folder.

In all other cases the generated files and folders must be copied to the cluster following the defined path names. Don't forget to make the `sh` file executable by calling `chmod +x *.sh` in the destination folder.

The project with all required source data must be copied, too. Minimum amount of data is the `exp` file of the project, the `Properties` folder and file

Properties.set in the project, and the complete directory hierarchy of the source set with the corresponding set files. Of cause the source set itself has to be copied completely together with its set file. And the generated result set file and folder have to be copied, too.

It's useful to enter a **file list** name in the dialog to get a text file with names of all folders and files to be copied to the cluster.

3.4 Start Processing

3.4.1 OpenMPI

For example type this command to start processing of the generated script myproc.sh on a OpenMPI cluster to run the script on both given nodes. The command returns when all nodes have finished processing.

```
mpirun -host localhost,192.168.0.42 \
    bash /home/iam/davisscripts/myproc.sh
```

Note: It is possible to use the same node for several jobs at the same time. In this case write access to the **DaVis** folder must be removed, see page 60. The command to start more instances on the same node could be like this:

```
mpirun -host localhost,192.168.0.42,192.168.0.42,192.168.0.73 \
    bash /home/iam/davisscripts/myproc.sh
```

3.4.2 SLURM

With SLURM job scheduler, the host list, task/node distribution and affinity parameters are already defined by SLURM once relying on the srun command, the command should be:

```
srun bash /home/iam/davisscripts/myproc.sh
```

4 Command Line Processing

4.1 How it works

Command line processing is working with the Windows and with the Linux version of **DaVis** and simply starts the software with two parameters: the operation list file and the source set. The **DaVis** runs without user interface. The operation list must be created with the Windows version of **DaVis** and stored on harddisc.

The idea behind command line processing is to give the user the possibility to create control scripts and run those scripts anywhere, usually on single Linux systems or on small clusters. When **DaVis** is started the project of the given source set is automatically detected and opened, then the processing list will be loaded and the source set is processed. The result set is stored using the naming policy of the operation list.

This processing mode needs no special license but checks the connected dongle or a network dongle for a valid license. See **DaVis Software manual** for a description about the installation of the dongle driver on Linux systems, where both local and network dongles can be used. **Note** that the installation of the dongle driver is NOT needed when using a network dongle!

4.2 Installation

The standard **DaVis** installation on Windows is required to use command line processing.

Refer to page 59 regarding the installation of the Linux version of **DaVis**. To be able to use command line processing on Linux systems either a network dongle or a local dongle with a valid **DaVis** license must be available. In case of a local dongle the dongle driver must be installed on the Linux system.

4.3 Start options

4.3.1 Linux

Starting a command line processing works with the following option definition using `op.OperationList.lvs` as path and name of the operation list and `source.set` as path and name of the source set. The source name is allowed to be defined without extension `.set`.

```
./davis -process <op.OperationList.lvs> <source.set>
```

Example with full output of all information and the complete progress in text form on stdout:

```
./davis -process /mnt/server/OperationLists/piv.OperationList.lvs
    /mnt/server/MyProjects/PIV/Recording73.set
```

This prints output like this:

```
StatusText:  Executing StartUp macro
StatusText:  DaVis 10.1.0
    serial# X000001-655616.46343L-64-190510
Project opened:  /mnt/server/Projects (for tests)/PIV-Test
Processing sequence loaded:
    /mnt/server/ClusterScripts/test110-piv.OperationList.lvs
Starting processing with source set:
    /mnt/server/Projects (for tests)/PIV-Test/Cam_Images=10_N=1
StatusText:  Progress:  0%
StatusText:  Progress:  10%
StatusText:  Progress:  20%
StatusText:  Progress:  30%
StatusText:  Progress:  40%
StatusText:  Progress:  50%
StatusText:  Progress:  60%
StatusText:  Progress:  70%
StatusText:  Progress:  80%
StatusText:  Progress:  90%
StatusText:  Progress:  100%
Processing finished
StatusText:  Executing ShutDown macro
StatusText:
```

```
StatusText:  done.
```

Use start option `-statusoff` to restrict the output to some general information:

```
./davis -statusoff -process /mnt/server/OperationLists/piv.OperationList.lvs
    /mnt/server/MyProjects/PIV/Recording73.set

Project opened:  /mnt/server/Projects (for tests)/PIV-Test
Processing sequence loaded:
    /mnt/server/ClusterScripts/test110-piv.OperationList.lvs
Starting processing with source set:
    /mnt/server/Projects (for tests)/PIV-Test/Cam_Images=10_N=1
Processing finished
```

With start option `-stdoff` the output can be disabled completely:

```
./davis -stdoff -process /mnt/server/OperationLists/test110-piv.OperationList.lvs
    /mnt/server/MyProjects/PIV/Recording73.set
```

Starting **DaVis** once for several runs for different operations list and/or different source sets is possible by just adding some more process options to the command line call:

```
./davis -process <op.OperationList.lvs> <source.set>
    -process <op2.OperationList.lvs> <source2.set>
```

All processing jobs are executed even in case of an error in one processing. Please check the output for all `Processing finished` lines.

The names of the result sets can be appended to a text file, which is created if not existing, by an optional argument:

```
-process-result-list <filename>
```

Example of a bash script to process all sets in a given project by the same operation list:

```
#!/bin/bash
PROJECT="/mnt/server/MyProjects/PIV-Test"
ls -1 "$PROJECT/"*.set | while read DAVISSET ; do
echo $DAVISSET
./davis -statusoff -process "/mnt/server/OperationLists/piv.OperationList.lvs"
"$DAVISSET"
```

```
done
```

4.3.2 Windows

Starting a command line processing works with the following option definition using `op.OperationList.lvs` as path and name of the operation list and `source.set` as path and name of the source set. The source name is allowed to be defined without extension `.set`. The following call must be executed in the `win64` folder in the **DaVis** installation. Using the arguments for a call to the `DaVis-Start.exe` is not possible.

```
davis.exe -process <op.OperationList.lvs> <source.set>
```

Example with simple progress bar in the startup dialog of **DaVis**:

```
davis.exe -process D:\OperationLists\piv.OperationList.lvs
    D:\MyProjects\PIV\Recording73.set
```

The Windows version of **DaVis** is not a console program. After starting from command shell the shell returns immediately and does not wait for the end of the software. During processing the progress bar of the start screen is running. The only way for automatic check of finished processing and of error is to read the logfiles, see page 62 about the location of those files, or to use the optional argument to store the name of the created result set in a text file:

```
-process-result-list <filename>
```

The name of the result set is appended to this file, the file is created if not existing.

4.4 Error messages

Errors can show up either at the startup of **DaVis** or during processing. To avoid processing errors please test the operation list on the Windows **DaVis**.

- **License problem: No active device found**: Dongle is not connected to the system.

- **No license for command line processing available**: No **DaVis** 10 dongle has been found or on Linux systems the wrong configuration is used like **cluster** instead of **processor**.
- **Not enough arguments for option '-process'**: Operation and or source set options are missing.
- **Processing operation list does not exist**: Invalid path and name of the operation list.
- **Source set does not exist**: Invalid path and name of the source set.

In case of processing errors the last line of the output looks like this on Linux systems:

```
Processing finished with error:  Mask not defined
```

On Windows systems the error details can be found in the log files only. See page 62 regarding the location of those files.

5 Installation on Linux

This chapter describes the installation of the **DaVis** software on Linux systems, which is available for 64 bit Linux operating systems only. For details of the requirements see page 59.

LaVision is not responsible for file loss or network problems caused by the usage of an incorrectly installed distributed processing or cluster script processing or command line processing with **DaVis**. With own macro programming a user could create own batch jobs, which could be responsible for large network traffic or high CPU usage on a **DaVis** worker PC and could even be used to delete or modify any file on the node in case of available access privileges.

5.1 Installation of DaVis on Linux

The Linux version of **DaVis** is not included on the standard installation medium and also not in the smaller installation archive from the **LaVision** website, see section above. **DaVis** on Linux is running without any GUI (graphical user interface) or even text interface. It can be executed as worker for distributed processing and cluster script processing and command line processing. It is not possible yet to execute the Linux version with user interface as known from the Windows® version, and it is not possible to control any hardware components.

5.1.1 System Requirements

The program is running and regularly tested on different Linux distributions. Regarding installation problems please contact the **LaVision** service. Some hints for problems with missing runtime libraries are given below.

Additionally a tool for script distribution is needed to be able to execute the script generated from the cluster script processing dialog. **DaVis** supports

script generation for OpenMPI, Sun/Oracle Grid Engine and Slurm Workload Manager. If another distribution tool is needed then please send the information to our service department. Adding a tool with its own script variables is usually no problem and can be done within days.

5.1.2 Installation Process

DaVis for Linux can be downloaded from **LaVision** website by registered customers, comes in a zip archive and can be extracted by the `unzip` command into a empty directory. A small configuration script can be executed via shell.

The installation of the worker is described in a separate section on page 24.

Please type `./davis-setup.sh cluster` for the **cluster script processing** configuration. A dongle driver installation is not needed, because the cluster script version of **DaVis** version does not connect to a dongle. The license is required for script generation only.

For the **command line processing** configuration please type `./davis-setup.sh processor` or `./davis-setup.sh processor <dongle-id>`. When using the first command **DaVis** tries to detect the dongle automatically, but the preferred way should be the second command to define the dongle during setup. The dongle driver can be installed by `sudo ./davis-setup.sh dongle`. Please see the **DaVis** software manual for details about the driver installation.

The directory with the installed and configured **DaVis** must be available on the complete cluster for every node. When many instances of **DaVis** are started from the same directory then all of them might write log files into that directory or store some settings when shutting down in the shared folder. It is better to mount this folder write protected or to set the folder itself as write protected via shell command `chmod -R a-w *` executed in the **DaVis** folder. When **DaVis** runs in a folder without write access some warnings will be given during startup and during shutdown. But most warnings are about log files and storage of settings and can be ignored.

The cluster PCs need access to the project folder of **DaVis** and to the script and settings files. If no direct access to the project folder is possible because of a special cluster network, all required files (source images, calibration data, ...) must be copied manually from the standard project folder to the cluster harddiscs.

5.1.3 Deinstallation

To uninstall **DaVis** just remove the **DaVis** folder from the system.

5.1.4 Fixing missing library issues

GLIBC_x.y

In **DaVis** 10.0 and 10.1.0 the following runtime libraries are required: `glibc 2.18` and `libstdc++ 3.4.21`. Usually these libraries are not available for older Linux versions. In this case during startup some error messages are displayed like:

```
linux64/davis:  linux64/lib64/libc.so.6:
version ‘GLIBC_2.17’ not found (required by /lib64/libglib-2.0.so.0)
```

This happens e.g. with CentOS 7.5 or later and can be fixed by removing the file `linux64(-centos69)/lib64/libc.so.6`.

libGL.so

Sometimes on server only Linux systems the `libGL.so.1` library is missing. In this case please install package `mesa_libgl` or `libglu1-mesa` depending on the distribution.

5.2 Startup of DaVis for Linux

The **DaVis** for Linux software does not include any window (X11, KDE, ...) or text (e.g. ncurses library) interface. So there is no startup window shown and no user input is possible, but all information and status messages can be printed to `stdout` for debugging purposes.

When started with command line parameter `-stdoff`, the informaton and status messages are not printed on screen and the program runs much faster during job execution. Sometimes information messages are useful but status messages creates too much output e.g. every progress change even for single image processing. The status messages can be disabled with start option `-statusoff`.

Whenever the worker tells about macro errors, detailed information is stored in log files, see section below. Of course in this case the write access must be available. Another way to get that information is to start the software manually and follow the standard output.

5.2.1 Settings folder

All variable settings of the software and all logfiles are stored by default in the **DaVis** subfolder `Users/<name>/log` with `<name>` as the user name of the active account. With this **portable mode** the user can copy the settings folder to any other PC into a **DaVis** installation of the same software version and use the same (copied) settings.

If a (empty) file of name `useAppData` exists in the binary folder next to the **DaVis** executable, then the so called **non-portable mode** is enabled: the **DaVis** folders are not touched and can be used in read only mode. In this case all settings are stored in folder `"%APPDATA%/LaVision GmbH/DaVis/10.1.0/` on Windows systems and `<user-home>/.config/LaVision GmbH/DaVis/10.1.0/` on Linux systems.

When starting for the first time, the software creates a `User.cl` file (until version 10.2.0) or a `ConfigureDaVis4Linux.cl` file (since version 10.2.1) in the folder `Users/<name>/CL_Autostart` with some default settings. The configuration defines a worker for cluster script processing of some default packages. Processing of other packages might be possible, but must be enabled manually.

5.3 Logging and error detection

Sometimes problems can be detected and solved only when looking into the worker's logfiles. By default files `LOG_<data>_<time>.txt` are stored in the **DaVis** subfolder `Users/<name>/log` until **DaVis** 10.2.0 and in subfolder `log` since version 10.2.1. The complete settings of the **DaVis** installation are stored in subfolder `Users/<name>`. Those files can be checked for macro error messages and for other information.

The storage of all standard logs and other files, depending on the package, is working only on a writable program folder. If the software has been installed with write protection or is running on a write protected

network drive, then the log folder can be changed by a command line parameter: `-logpath=<path>` where path gives the folder to store the logfiles, e.g. `win64/davis.exe -logpath=c:/temp` or `./davis-start.sh -logpath=/tmp/mydavis`. The folder has to exist before starting, it is not created or cleaned.

5.4 Temporary folders in DaVis

There are two different temporary folders in the system. The standard temporary folder is used by lots of functions in DaVis in most cases to store some small files e.g. during processing. This folder should be and is by default on the local harddisc. The second folder is used by the processing dialog only to store e.g. intermediate results during processing when the user disabled the storage mode for one of the operations. Its location is always in the active project folder in a child path under `Properties`.

The standard temporary folder is requested by macro command `GetTempDirectory()`. By default the system temporary folder on Windows is taken using environment variable `TEMP`. This variable can be defined on Linux, too. If `TEMP` is missing on Linux then the folder is created as `$HOME/tmp/davis_<hash from DaVis folder name>_<hostname>`. The automatic folder determination can be replaced by CL variable `DefaultPathTemp`, which is empty by default. It can be defined e.g. in the `User.cl` file (until version 10.2.0) or in the `ConfigureDaVis4Linux.cl` file (since version 10.2.1).

The second folder used by the processing module is requested by macro command `EXP_GetTempPath()`. The child path in the active project folder is defined as `Properties/Temp_<date>_<time>_<hostname>_<main task id on Linux only>`. This default folder naming can be replaced by environment variable `TEMP_PROCESSING_PATH` if this variable exists and is not empty.

5.5 Number of used processor cores during processing

By default **DaVis** uses as much processor cores as possible during processing. Many algorithms like PIV or Tomographic PIV are running multi

threaded and using all cores of the CPU(s). Sometimes on shared systems a process should not take all but only a part of the available cores. Therefore start option `--omp-max-threads=<` *value* `>` can be used to restrict the number of threads. This option is available for both the Linux and Windows version of **DaVis**.

A simple test on Linux is the following start via command line:

```
./davis -app ScriptProcessingApp -exec=SetupLinuxClient\(0\) --omp-max-threads=4
```

On a fresh extracted **DaVis** this call starts the software and generates macro file `User.cl` file (until version 10.2.0) or a `ConfigureDaVis4Linux.cl` file (since version 10.2.1). The standard output during startup displays the number of cores in the system and also the number of threads used by the processing.

On Windows systems **DaVis** can be started from the command shell like

```
./davis.exe --omp-max-threads=4
```

After startup press the `F7` key and enter this CL code to print the maximum number of used threads into the info dialog:

```
InfoText(GetEnvironmentVariable(":OMP_MAX_THREADS"));
```

5.6 File Server and Shares

5.6.1 File Server with Mapped Directories

At first create a network path for the project data on a file server. The Master-DaVis and every Remote-DaVis (Worker) must have read, write and creation privileges to this path and to the subdirectories. This privilege setup is the critical part of the complete installation.

The network path must be installed at every PC with the same drive character, that means the drive mapping of the project path must be equal on all PCs of the network! E.g. if the file server has the name `SERVER` and the share has been named `SHARE`, the network name of this path is `\\SERVER\SHARE` and must be installed e.g. with driver character `S:` on each Windows® -PC.

5.6.2 Master PC as File Server

When using a project path on the Master PC and storing all data on this PC, the project path must be shared to the network. The shared directory must be the project path or any upper level directory. E.g. when your project is located in `E:/MyData/MyProjects`, then you have to share either this directory or `E:/MyData` or `E:/` itself. The best way would be to share `E:/MyData`, then all projects of this folder can be used for processing.

DaVis searches automatically for the share and gives an error message, if no share can be found for the active project. Nothing additional has to be configured in the **DaVis** software.

5.6.3 Using a File Server with Shared Directories

The shared directories of the file server must not be mapped with drive letters to each PC. The project path or any parent path must be shared to the network, and the access privileges should be defined at least for the dummy user, maybe for a group or even for all users.

Setting up Shared Folder

Depending on the network environment and the operating systems the setup of the sharing access is different:

On Windows 7® click with the right mouse button to the folder, which should be shared. Select **Share to certain users** and add the users in the user selection dialog.

Troubleshooting: Windows 10 does not connect to mounted shares

Sometimes Windows 10 does not connect to mounted shares during startup. After login the shares are marked with red signs and the Windows notification service tells about not connected shares. The connection will be established by clicking on the share in the Windows explorer, but this is not possible for workers started automatically with the system, therefore workers are not able to work with this shares and give error messages during processing.

The first way to solve this problem is a manual modification of the startup script of the worker in the folder selected during installation. Open file `worker-start.bat` with a text editor and uncomment the line

```
rem net use S: \\server\share /u:%USERNAME% /persistent:yes
```

by removing the `rem` at the beginning. Then change `S:` to the driver you are using and `server\share` to the server name and share name you are using. Store the modified file and restart Windows to verify if this is working.

The second way to solve this problem would be a modification of Windows system settings for : Right click the Windows icon at the bottom left of the task bar or select the windows key + r. Within the run box, type `gpedit.msc`. Within the local **Group Policy Editor**, select **Administrative Templates - System - Login**, then modify the settings of **Always wait for the network at computer startup and logon** and enable the mode, see figure 5.1. As a small disadvantage the startup of the PC will take some seconds longer.

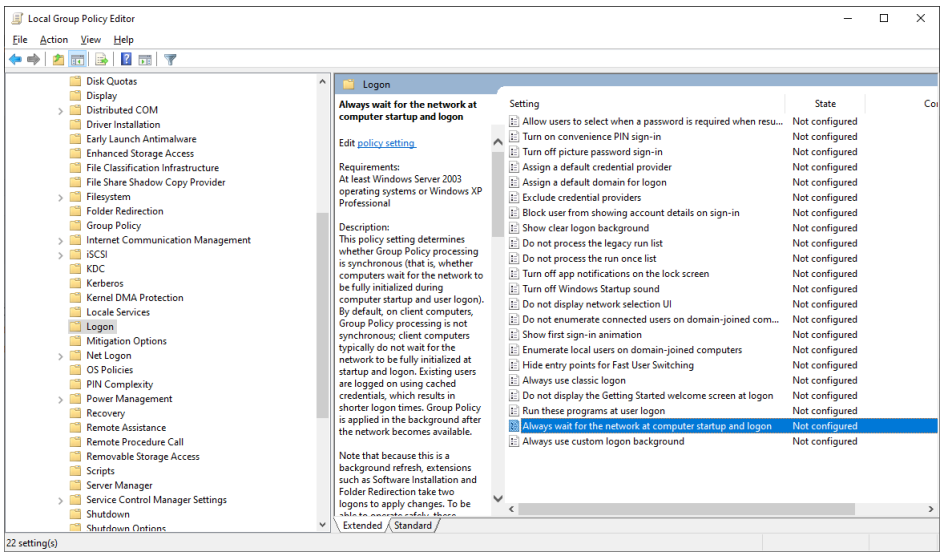


Figure 5.1: Group Policy Editor to modify the initial network connection of the PC.

Troubleshooting: Access timeouts on Windows 7 and modern Windows Server versions

There is a known issue with access problems between Linux cifs and Windows share service, which causes *mount error(12): cannot allocate memory* on the Linux side. A simple solution is to modify two registry values on

the Windows Server and then restart either the service or the machine, see description at https://wiki.archlinux.org/index.php/Samba#Troubleshooting

Set value to 1: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\LargeSystemCache

Set value to 3: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters\Size

5.6.4 Mapped drives and shares for the Linux worker

Configuration of a drive mapping between Windows drives and Linux mounts is required for distributed processing and useful for command line processing. Cluster script processing does not need this mapping.

During configuration of the **DaVis** worker via script ./davis-setup.sh (see section above) a file with some example definitions is created in folder Users/<name>/CL_Autostart as User.cl file (until version 10.2.0) or as ConfigureDaVis4Linux.cl file (since version 10.2.1). This file is automatically loaded by **DaVis** during startup. The lines to be modified are those with function calls to Linux_SetDriveMapping and Linux_SetShareMapping.

With Linux_SetDriveMapping a Windows drive letter like E: is mapped to a mounted share on the Linux system like /mnt/server-share. In case of a file server named SERVER and its share SHARE the Windows system would use E: as \\SERVER\SHARE and the Linux system would mount this share to /mnt/server-share then the drive mapping must be defined as:

```
Linux_SetDriveMapping('E',"/mnt/server-share");
```

It is also possible to map full share names. When a project is opened e.g. from \\SERVER\SHARE on the Windows system by **DaVis** then the Linux worker can be configured to map this share path to a locally mounted share:

```
Linux_SetShareMapping("//SERVER//SHARE","/mnt/server-share");
```

6 Operation Overview

The following tables give information about operations with special restrictions to the distribution behavior. This can be restrictions to Windows only, when an operation or a certain package is not available on the Linux worker. There are simple operations, which are able to be distributed to several workers, each worker calculating a part of the result set. These operations have a 1:1 relation, meaning every source image or source vector field has one result, and also the chosen processing range does not change the result. And then there are more complex operations like average or time filter, which need the whole source set and which have to be processed by a single worker. Sometimes this behavior depends on parameters, see table below.

In case of multi sets as source data the distributed processing will at least send each set to another worker. If the operation is allowed to run on different workers on each sub set, then each sub set will be distributed to different workers itself. There are a few operations on multi set which have to run on a single worker.

Operation	**Distributable to different workers**	**Running on single worker**
Basic image or vector arithmetics group	Yes	-
Image or vector properties group	Yes	-
Image mapping group	Yes	-
Image analysis group	Yes	-
Scales group	Yes	-
Attributes group	Yes	-
Statistics group	-	Yes
Masks group	Yes	-
Time-series group	-	Yes
Filter group	Yes	-

Operation	**Distributable to different workers**	**Running on single worker**
Copy and reorganize frames group	Yes, for some operations.	Yes, for some operations.
Copy and reorganize data sets group	-	Yes
Plot group	-	Yes
Convert image to RGB group	Yes	-
User functions group	Yes	-
Extract scalar fields group	Yes	-
Image reconstruction	Not in shift mode and not in surface height mode.	Yes
PIV image preprocessing	Not subtract minimum over time, not subtract sliding minimum, not local average normalization.	Yes
PIV	No time resolved processing. Auto mask type is not temporal. No sum of correlation (SOC), no sliding sum of correlation (SSOC) and no pyramid correlation.	Yes
Tomographic sizing image preprocessing	Not subtract minimum over time, not subtract sliding minimum, not local average normalization.	Yes
Tomographic PIV	For double exposure images only.	Yes
Shake-the-Box	-	Yes
Shake-the-Box double-frame with predictor	-	-

Operation	Distributable to different workers	Running on single worker
Multi particle field to particle field	-	Yes
Fine-Scale Reconstruction (VIC#)	-	Yes
Strain	-	Yes
DVC	-	Yes
BOS, Difference BOS, ESF, ESS	-	Yes
Data export	-	Yes, if exporting into the project.
Flame propagation export	-	-
Particle master export	-	-
Spray geometry export	-	-
Spray geometry processing	-	-

Table 6.1: Operations with special distribution behavior. This table is not complete for all of the more than 500 processing operations in DaVis. In case of the operation groups the behavior is usually the same for all operations in this group.

Project	Operation	Restriction
All	Export of data	If the destination is in the project, then a worker can process. A free destination can be processed on the local system only.
All	Export of views	Local system only.
PIV	PIV with GPU	Windows only, and GPU required for worker PC
PIV	PIV on CPU	Windows and Linux

Project	Operation	Restriction
Tomo	Fine-Scale Reconstruction (VIC#) with GPU	Windows only, and GPU required for worker PC
Strain	-	Windows and Linux
Exciplex	-	Windows only
Fluid Structure Interaction	-	Windows only
Fuel	-	Windows only
LIF	-	Windows only
Fuel	-	Windows only
Particle Master	-	Windows only
Pyrometry	-	Windows only
Raman	-	Windows only
Rayleigh Thermometry	-	Windows only
Soot Master	-	Windows only
Soot Pyrometry	-	Windows only
Spray Master LIF	-	Windows only
Spray Master	-	Windows only
Surface Flow	-	Windows only

Table 6.2: Projects and operations with restrictions to an operating system or to the distribution at all.

7 Support

If you have a technical problem or a question regarding hardware or software which is not adequately addressed in the documentation, please contact your local representative or **LaVision** service directly.

You can contact service at **LaVision** GmbH by:

e-mail: **service@lavision.de**

phone: **+49 551 9004 229**

Alternatively, you may submit your problem using the **Support Request Form** in the **Support** section of the **LaVision** website `www.lavision.com`.

In order to speed up your request, please include the following information:

- The order number of your system.
- The number of the used dongle (if available).
- Article number (if available).
- Serial number (if available).
- A short description of the problem.

7.1 Service file creation

In order to be able to reproduce a software problem, it could be essential to know the exact hardware setup and software parameters in **DaVis**. All currently used parameters and all error messages that have been shown since the last **DaVis** start can be extracted using the the following call in the **DaVis** root folder:

```
./create-service-file.sh
```

The `ZIP` file will be written automatically to the current folder. The name of the file contains the order number and dongle number that is extracted from the current settings. Send the `ZIP` file as attachment to your email together with the description of your problem to `service@lavision.de`.

Note: Files with a size of more than 20 MB should not be sent by email. **LaVision** can provide a link for uploading data via file drop. Please contact `service@lavision.de` for details.

Index

Manual No 1003013 - December 5, 2022

LaVision
FOCUS ON IMAGING

LaVisionUK Ltd
2 Minton Place / Victoria Road
Bicester, Oxon, OX26 6QB / UK
www.lavisionuk.com
Email: sales@lavision.com
Tel.: +44-(0)-870-997-6532
Fax: +44-(0)-870-762-6252

LaVision GmbH
Anna-Vandenhoeck-Ring 19
D-37081 Göttingen, Germany
www.lavision.com
Email: sales@lavision.com
Tel.: +49(0)551-9004-0
Fax: +49(0)551-9004-100

LaVision, Inc.
211 W. Michigan Ave., Suite 100
Ypsilanti, MI 48197, USA
www.lavisioninc.com
Email: sales@lavisioninc.com
Phone: +1(0)734-485-0913
Fax: +1(0)240-465-4306