MeVisLab Release Notes before 3.0

Table of Contents

Release Notes before 3.0

This document lists the most relevant changes, additions, and fixes provided by the MeVisLab releases before 3.0.

Version 2.8.2 Stable Release (2016-10)

New Features

- Perform a sanity check of the module cache when loading. Faulty cache files are removed.
- Added `DrawVoxels3DMacro` module for interactively drawing into an image.
- Added field `emulateLegacyOpenGL` to shader modules which allows to switch off the emulation.
- Added blend mode parameter to `SoView2DMenu` module.
- Added possibility to override path center in `PathToKeyFrame` module.
- Improve scripting wrappers for `ml::KeyFrameList` by adding methods that allow adding/removing items.

Fixes

- Fixed file ending filters in various file selection dialogs.
- Avoid resize of "Find in Files" result area through long file names in the progress display.
- Make sure the right file is selected in the Session view of MATE if a document is selected that has the same name but a different path than another file.
- Fixed focus stealing bug in Pylint code checking with Windows 10.
- Fixed Pylint online installation.
- Fixed loading of PythonQt extension in Pylint.
- Added missing documentation of `MLGraphWrappers`.
- Added all include files of the `Eigen` library.
- Fixed documentation of `ml::DateTime` format string.
- Avoid crash in `SoView2DRigidRegistrationEditor` if the specified icon files do not exist.
- Fixed some bugs in `GraphicsViewPresentation`.
- Added a missing variable initialization in `AnimationRecorder`.
- Clamp output subimage against output image of `VoxelizeInventorScene` to avoid a crash.
- Do not lose LUT settings when re-connecting the input image of a `View2D`.
- Fixed shadow rendering of CSOs in fast rendering path.
- Try to avoid crash when no editor is active in `SoView2DCSOExtensibleEditor`.

- Fixed output of invalid voxel values in `DTFSkeletonization`.
- Copy marker ids in `XMarkerListTransformation`.
- `SoCSOManualCorrectionEditor` did not set the sub-type of contours correctly.

Version 2.8.1 Stable Release (2016-06)

New Features

- Added OpenInventor loop detection to IDE: If OpenInventor seems to be stuck in an endless update loop, a blinking "Loop!" indicator is displayed in the status bar of the IDE.
- Added `toggle` method to scripting class `MLABBoolField`.
- Added methods `findPrivateSlot` and `getPrivateTag` to scripting class `MLABDicomTree`.
- Added methods `setRGBA` and `getRGBA` to scripting wrapper of `ml::Skeleton`.

Fixes

- Do not require administrator rights when installing MeVisLab (depending on the install location).
- Do no treat unreachable code as error in Visual Studio 2015 (this was a problem with the Update 1 of Visual Studio 2015).
- Added missing file `createSolutionFromMLBuild.py` (used by the ToolRunner on Windows).
- Fixed module placement if modules should be placed at the current cursor position, but the module was selected by clicking in the quick search drop down box: In this case place the module at the center of the network.
- Fixed memory leaks in remote communication protocol.
- Avoid endless loop in new multi-threading host when throwing an error from `calculateOutputSubImage`.
- Fixed that some OpenInventor modules would cause crashes with some Quadro graphics cards (or rather their drivers).
- Avoid crash in MATE when closing documents of a session and auto save is enabled.
- Fixed that a negative integer could not be entered into a field control of an integer field when the minus was the first typed character.
- Update internal state of `TabView` before calling `tabSelectedCommand` or `tabDeselectedCommand`, as changing the tab again in these commands would leave behind a wrong state otherwise.
- Fixed that the `tabEnabled` and `tabTooltip` attributes of the `TabView` control didn't work.
- Fixed that **File** → **Create Local Macro...** would not retain the order of connections to a `SoGroup` (if only a subset of inputs connects from outside the local macro).
- Correctly determine output image min/max values of `ConstrainedConnectionCost` module in diff mode.

- Fixed a rendering difference between GPU and CPU rendering when using `SoRenderSurfaceIntersection`.
- Make `ImageToCSOVoxelSet` work for 3D images.
- Implemented possibility to override path direction for `PathProjection` mode of `PathToKeyFrame`.
- Fixed that `GetVoxelValue` would constrain the output value to the input data type range if the image had RescaleIntercept/RescaleSlope tags set.
- Fixed crash in `PathToKeyFrame` if the user managed to set a negative key index.
- In the `MacroModuleWizard` only check the `.mlab` file specified by the user for local macros.
- In `SaveBase` also regard backslashes when determining whether to append `.xml` to the filename.
- Fixed compile errors of PCL modules with Visual Studio 2015 Update 2.
- DICOM handling improvements:
  - Fixed some suboptimal import configurations of DICOM NM data.
  - Improved performance bottlenecks of non-DICOM-Imports with `DirectDicomImport` which make use of ITK loaders.
  - Commented risks of deactivated "Decompose MultiFrame Files" in the help of `DirectDicomImport`.
  - Fixed exceptions when exporting 2D DICOM frames manipulated with `DicomModifyImageTagsPlugin` which have an empty NumberOfFrames tag.

Version 2.8 Release

Fixes / Enhancements (MeVisLab)

- IDE:
  - Some of the defaults for network appearance and network interaction have been changed. If you don't like the new defaults, you can change them back in the Preferences dialog.
  - Modules in the network view now show small message indicators for warnings, errors, etc. besides the modules, which makes it easier to discern which module produced which messages.
  - Connections can be moved in the network without pressing Shift.
  - Implemented dropping of modules on other modules or connections to create a connection.
  - Show small Inventor connectors instead of rectangles when dragging Inventor fields to a Group/Separator. This way it is more obvious that you can add a connection between existing connections.
  - Allow to disconnect connections from the context menu of network connections.
  - Also allow to disconnect connections by selecting them in the network and pressing Delete/Backspace.
  - Make more network interaction undoable.
  - Status visualization of image input/output connectors by colors now extends to Base fields, too, if activated (a connector turns red if no Base object is set on the field).

  - The tooltip for Base field connectors now shows the type name for Python and VTK objects.
  - Entering an empty name when renaming modules will set the default name.
  - Video generation in MeVisLab (e.g., from the Screenshot Gallery) now uses OpenCV on Windows and Linux.
  - Module groups are now retained when creating a local macro from part of a network.
  - When creating a local macro you can't promote it to a global macro anymore. This didn't always work correctly.
  - You can now select a target directory in the dialog for creation of local macros.
  - Added a keyboard shortcut editor to MeVisLab and MATE.
  - The user can take screenshots of the current network with the F12 key.
  - The default dock layout of the IDE has been changed to a more sensible default.
- TestCaseManager:
  - Added `TestSupport.Macros.DEBUG_STOP` for debugging purposes in tests.
  - `TestCenter.py` can now run test suites from the command line.
  - Replaced "Use Debug" setting in TestCaseManager configuration (for secure mode) by an enum that also allows Automatic selection.
  - Python modules can now be reloaded first when pressing the reload button in the TestCaseManager panel. This can be configured on the Configuration tab.
- MATE (integrated editor)
  - Added option for silent auto reload of files.
  - Field expressions are now auto-completed.
  - Python import paths are stored in workspace projects (and used by PyLint, if installed).
  - Other small improvements.
- C++ API changes
  - Added class `BackgroundTaskMessageReceiver` to `BackgroundTask` framework.
- Controls
  - The slider controls honor the editable attribute of fields now.
  - Added MDL tag `caseSensitiveAutoComplete` for combobox mode to `Field` control.
  - Do not print warning about unsigned JavaScript in `WebView` control.
  - Show a double-tilde ("almost equal to") sign instead of a single tilde in front of numeric values that don't fit completely into a control.
- Scripting

  - Can disable busy state indication with `MLAB.enableBusyStateIndication(False)`.
  - Fixed problems with `destroyedCommand` on reloading of a module.
  - Added the methods `voxelVolumeInCubicMillimeters` and `voxelVolumeInMilliliters` to `MLABImageField`, which return the correct volume for sheared images.
  - Added new field wrappers for `ml::Vector6Field`, `ml::ImageVectorField`, `SoSFVec2s`, and `SoSFVec3s`. Previously a `MLABStringField` would be used for all these parameter types.
  - Added accessor for `QSqlQuery::lastQuery()` to `MLABScriptQuery`.
  - The `RemoteCallInterface` can now transmit None arguments.
  - Added methods `lastRenderedSize` and `currentRenderSizes` to the `RemoteRendering` class (and its wrapper).
  - Added `MLAB.getModuleForInventorObject()`, to be used with Inventor node wrappers.
- Other fixes and improvements
  - MeVisLab now has support for saving and loading of many image formats with sizes larger than 4GB (if the format supports this at all).
  - MeVisLab uses the Mesa OpenGL driver on Windows automatically if no hardware driver is present (e.g., because using it over a Remote Desktop connection) or if the environment variable `MLAB_FORCE_MESA` is set.
  - Added support for `visibleOn` and `dependsOn` attributes of (mainly input and output) fields.
- Third-party libraries
  - The GLEW library was updated to version 1.12.0.
  - libtiff has been updated to version 4.0.3, which includes support for files larger 4GB with the BigTiff format.
  - matplotlib has been updated to version 1.4.3.
  - Added the Python setuptools in version 1.6.
  - OpenCV was updated to version 3.0.
  - OpenSSL was updated to version 1.0.1r.
  - OpenInventor:
    - Added support for glVertexAttribArray rendering (and removed indexed color mode).
    - Class `SoDragger` got a field to disable interaction.
    - Primitive shapes allow to set a fixed tesselation now.

Fixes / Enhancements (Modules)

- General
  - The `ImageLoad` module can now load signed/unsigned 64 bit integer data types in RAW format.

  - Added raw loading to `Compose3dFrom2dFiles`.
  - Fixed writing of DICOM files in Japanese encoding (ISO_IR 13, ISO_IR 14).
  - `ImageStatistics` also shows the sum of voxel values now.
  - `ExtrudeDimension` was broken and was removed.
  - Added an undo framework for modules. See module `UndoManager` and the supported modules `SoVolumeCutting`, `CSOListContainer`, and `CSOManager`.
  - `RemoteRenderingEventRecorder` can now correctly record Num Pad keys.
- SoView2D and extensions
  - Introduced a shader framework for `SoView2D` (look for modules starting with SoView2DShader... and at `SoView2DExtensionSampler`).
  - Initial implementation of CPU rendering in `SoRenderSurfaceIntersection`.
  - Implemented event forwarding in `SoView2DScene`.
  - `SoView2DPickCamera` has been removed.
  - Added modules `VideoWriterML` and `VideoWriterInventor` which use OpenCV to create video files.
  - Enable `SoView2DMarkerEditor` in Managed Interaction mode if only selectingOn is TRUE.
  - `SoView2DPosition` can set new positions in Managed Interaction mode if `cooperative` is FALSE.
  - Added the module `SoView2DRigidRegistrationEditor`, which allows to modify an image transformation on screen.
- Other Open Inventor modules
  - Added new, faster transparency algorithms to `SoRenderers` library that require higher OpenGL versions.
  - Added module `PostEffectAntiAliasing`.
  - Make `SoDiagram2D` work with several viewers at the same time.
  - `So3DMarkerEditor` uses shared geometry for spheres now, which should speed up rendering.
  - Added new module `SoActionNotify` that triggers fields if it is rendered/gets event handling/etc.
  - `SoMouseGrabber` sets *Pressed, maskValid, and cursorPresent flags to false if switched off.
  - Added a forceNormalGeneration field to `SoSceneLoader`.
  - Improved and optimized `SoTensorFieldVis`.
  - Added a `So3DMarkerRenderer` which is suitable for the visualization of large marker counts.
  - The `SoVascularSystem` module now allows to color the skeleton tree from the root recursively until the radius of the branches drops below a defined cut-off value.
- CSO

- `SoCSOTransformEditor` allows rotation and isotropic scaling now.
- A `StylePalette` can now be attached to the `SoCSOVisualizationSettings` module to set visual attributes of individual CSOs or CSOGroups.
- The scripting wrappers for the CSOs have been adapted and extended.
- `CSOIsoGenerator` and `CSOSliceInterpolator` generate less seed points now.
- Speed-up of `CSOLoad` and `CSOIsoGenerator` when a lot of CSOs are loaded/generated.
- Can select index of displayed CSO/Group in `CSOInfo` now.
- Added the `SoCSOIsoEditor` module.
- Added the `CSOListContainer` module, which is a light-weight replacement for `CSOManager` if you don't need the extensive GUI interface (e.g. because you use scripting).
- Various other improvements.

- WEM
  - Removed field copyInputWEMs and replaced it by workDirectlyOnInputWEM on affected WEM modifying modules, the default is to copy the input WEM now.
  - Re-implemented `WEMClip` and split into three new modules.
  - `WEMSceneLoader` can optionally generated edges after loading now.
  - Fixed a WEM persistence bug with polygon patches.
  - Added creaseAngle support to `SoWEMRenderer`.
  - Added new shape Icosphere to `WEMInitialize`.
- ML modules
  - Added new module `VoxelizeInventorSceneGPU` (which requires a graphics driver supporting OpenGL 4.2 to work).
  - Added a `BaseClear` (similar to`ImageClear`) module.
  - Various fixes and improvements to the ConnectedComponents modules:
    - `ConnectedComponentsInfo` was extended in functionality.
    - `FilterConnectedComponents` can select by rank now.
    - Allow clamping of output image to bounding box of cluster(s).
    - Set user data on clusters and allow filtering on this information.
  - `BaseBypassOp` was deprecated, you should use `BaseBypass` or `BaseClear` instead.
  - Added a basic volume sculpting module `SoVolumeCutting`.
  - The `MPR` module now allows to specify the up vector (giving control over the rotation of the output MPR around the plane normal).

- Added option to get tri-linear interpolated values in `GetVoxelValue`.
- Added `CLAHE` filter module.
- Added new module `MLGenerateImageForBoundingBox` that returns an empty ML image for an Inventor scene's bounding box (e.g., for use with `VoxelizeInventorScene`).
- Added some OpenCV filters as MeVisLab modules. Look for modules starting with "cv".
- Fixed that the NGLDM and NGTDM modes of the `TextureFilter` module returned identical results for the first slices.
- Added example code for an image loading module. Check out the module `SimpleRawImageLoadExample`.

- Macro modules
  - Renamed `LocalFileName` to `LocalPath` and added an enum to select either a file or a directory.
  - `RunPythonScript` improvements (also see Details in module help):
    - Easier use of input fields (directly available as variables, even numerically typed).
    - Easier setting of output fields with `setOutputValue()` or `updateOutputValue()`.
    - Improved panel layout.
  - Replaced `CloneImageAsConstantImage` and `CloneImageAsTestPattern` with `CloneImageAsEmptyImage` which now persists the cloned empty image in a network.
  - Added the (work-in-progress) `AnimationRecorder` module.

Windows Edition

- Fixed that the user interface could lock up under high load for the Visual Studio 2013 (and later) versions.

OS X Edition

Note

This release requires at least OS X 10.11 (El Capitan) and Xcode 7.1 for development.

New Features

- Full support of OS X 10.11 (El Capitan)
- Most bundled command line tools are now usable from outside MeVisLab
- Full support for adding a signature to MeVisLab-based standalone Apps and PKGs (requires ADK)

  Add `$MACX_SIGN_APPLICATION 1` to your .mlinstall installer description file to use the default installer ID of your OS X keychain

Bug Fixes

- Fixed broken drag&drop of files onto module panels.

Removals

- The OsiriXBridge modules, the plugin, as well as the documentation have been moved to the GitHub community repository (https://github.com/MeVisLab/communitymodules/tree/master/Community/General/Projects/OsiriXMeVisLabBridge/) and are no longer shipped with MeVisLab

Modules and Libraries Provided by Fraunhofer MEVIS

Fixes/Enhancements (FME Modules)

- AlgorithmModule framework
  - Added prepared NetworkPanel code to include in `MLAlgorithmModule` and `AlgorithmMacroModule` derivates. Add:

    ```
    #include "$(MLAB_FMEstable_ReleaseMeVis)/Projects/AlgorithmModule/MLAlgorithmModule/Modules/NetworkPanel.mdl"
    ```

    to the NetworkPanel section of your module's MDL file. See example modules of AlgorithmModule framework for an example.
  - Added specialized error classes for use in `AlgorithmModule` derivates: `InputObjectError`, `InputParameterError`, `InternalError.`

    The new error classes automatically provide the corresponding status code to their base class `AlgorithmModule::Error`. The developer now only has to provide a meaningful error message.

    Have a look into the updated MLAlgorithmModuleExamples project.
  - Added specialized error classes for use in `AlgorithmMacroModule` derivates: `InputObjectError`, `InputParameterError`, `InternalError`. The new error classes automatically provide the corresponding status code to their base class `AlgorithmMacroModule.Error`. The developer now only has to provide a meaningful error message.

    Have a look into the updated AlgorithmMacroModuleExample.
  - Resolved an issue of the `AlgorithmMacroModule` base class which silently prevented that deriving modules could connect with module's `wakeupCommand` signal.
- CLI execution framework
  - `CLIImporter` offers a more convenient list view for the path configuration.
  - `CLIImporter` also discovers versioned Slicer app bundles on OS X.
  - CLI execution is performed in background, not freezing MeVisLab's GUI.
  - NetworkPanel feature is used for all CLIs now.
  - Uses new ctk_cli package from `https://github.com/commontk/ctk-cli` (part of ThirdParty).
- New module `VTKDefaultObject`: Allows the creation of a few VTK default objects via an enumerator.
- The module `SoVTK` supports connected vtkRenderers now; instead of a single vtkProp all vtkProps/vtkActors of the renderer are connected.
- New basic infrastructure modules for curves

  - `CurveInfo`: Gives information about connected curves and curve lists.
  - `CurveProperties`: Allows changing the properties of a curve list.
  - `ExtractCurveData`: Extracts a single curve from a curve list.
  - `CurveListContainer`: Provides a curve list for scripting access.
- New module `WEMExpandToMarkers`: Moves WEM nodes guided by a small number of XMarkers.
- New convenience context manager and comparisons in Python package `fmeTestSupport`.
- Test hierarchy analysis for complex dependencies
  - Python package `applicationTesting` for dependency analysis of various MeVisLab components, finding out test associations and executing tests.
  - Modules `RunFunctionalTestCases`, `RunGenericTestCases`, `RunAllTestCases`, `RunPythonTests`, `RunGoogleTests`, and `ApplicationAnalysisResultViewer` based on the Python libraries.
- New module `PythonCoverage`: Manually controls Python coverage analysis.
- New module `SplitList`: Splits a list of Base objects (a ListBase) into two parts.
- Module `itkImageFileReader`: Does no longer log a warning if the input file name, given via field unresolvedFilename, is empty (which is common behavior of other loader modules).
- New module `RekSave`: Saves the Fraunhofer EZRT file format (which can be imported with `DirectDicomImport`).
- Public `RecHeader.h` file provided by Fraunhofer EZRT now available in FME ThirdParty.
- Sped up `DMFileReader` plugin (used by all loaders based on MultiFileVolumes, such as `DirectDicomImport`), which supports the import of Digital Micrograph .dm3 and .dm4 files.
- DICOM
  - `DICOMSCSave`: Fixed a crash when inheriting tags.
  - `MultiFileVolumeListExport`: Fixed overwrite of incoming color images always with photometric interpretations RGB.
  - `DicomTreeCompare`: Provides option to hide remaining empty trees after hiding tags in Added and Removed result fields now.
  - New module `DICOMFileListFilter`: Searches for specific DICOM configurations in files.
  - `DirectDicomImport`
    - Checks now the imported data for inconsistencies between volume extents as described in DICOM tags and real extents and prefers real extents.
    - Fixed too strict warnings (*NumberOfSlices*) on imports for some retired PET DICOM tag configurations which are legal.
    - Fixed bad default setting of 'Minimum Num Frames in Volume' which lead to an undetected import of irregular positioned slices.

- When importing enhanced SOP classes they are corrected to single frame *SOPClassUIDs* when decomposing frames.
- Added an import-plugin for Fraunhofer EZRT (.rek and .raw) images.
- Fixed missing robustness against rare cases where the *PlanarConfiguration* tag is unreliable which lead to load failures.
- Fixed *OtherImporter* which uses `itkImageFileReader` with default 3D settings which hid higher dimensions in imported data.
- Fixed a problem where series with changing *BitsStored* tag were split.
- Adjusted tolerance values in default DPL configurations to match better commonly appearing data sets.
- Fixed: When caching *MultiFileVolumeLists* the property extension settings were lost.
- Fixed duplication of all tags in the *SharedFunctionFramesSequence* tag which blew up memory and time usage on large sequences.

- PCL
  - Added the Point Cloud Library (PCL) to MeVisLab.
  - Added new package PCL to MeVisLab containing some modules and interfaces based on this library:
    - `PCLModule`: Module base class.
    - `PCLCompare`: Compares point clouds.
    - `PCLInfo`: Displays information about connected point cloud and surface data structures.
    - `PCLNormalEstimation`: Estimates normals of point clouds .
    - `PCLCropBox`: Cuts out a sub-region of a point cloud.
    - `PCLMLImageToPointCloud`: Converts an ML image to a point cloud.
    - `PCLPointCloudToXMarkerList`: Converts a point cloud to an XMarkerList.
    - `PCLBaseListToPointCloud`: Converts a list-like ML Base object to a point cloud.
    - `PCLToInventor`: Simple display module for a connected point cloud and surface.
    - `PCLLoad`: Saves a point cloud in a file.
    - `PCLSave`: Loads a point cloud from a file.
    - `PCLMarchingCubesHoppe`: Creates a surface from a point cloud.
    - `PCLMarchingCubesRBF`: Creates a surface from a point cloud.
    - `PCLConvexHull`: Creates a convex hull from a point cloud.
    - `PCLEuclideanClusterExtraction`: creates Euclidean clusters from a point cloud similar to a connected component analysis.

    - `PCLPassThrough`: passes a selectable point member through an interval similar to `IntervalThreshold` or aligned box cropping.
    - `PCLVoxelGrid`: samples/voxelizes a point cloud to a grid aligned point cloud and also into an ML image.
    - `PCLSupportOutput`: provides a dedicated output for script manipulations.
    - `PCLSupportWrappers`: provides basic Python script access to objects in PCL Base connectors and point clouds.
    - `PCLVisualizer`: Wrapper around the `pcl::visualization:PCLVisualizer` class.
  - For details see the module documentations and example networks.
  - Notes
    - The binding is work-in-progress and the modules and bindings will undergo many changes in future.
    - All sources are in the MeVisLab Public Sources and can be found in an installed MeVisLab SDK under `<MeVisLabInstallPath>/Packages/FMEwork/PCL/Projects`.
- PlaneUtils: added new modules
  - `ImageSegmentationByPlane`: Segments an image by a 3D plane.
  - `ComposePlaneFromXMarker`: Composes a 3D plane from XMarkers
  - `ImageMirror`: Mirrors an image along a given plane.
- CSO
  - New module `FindCSOsOnSamePlane`: Provides information whether two CSOs are located on the same plane.
  - New modules `CSOBoolOp` and `CSOOffset`.
  - Changes in `MLCSOBoostGeometry` shared project
    - Added functions for computing area and symmetric difference.
    - Internal enhancements to ensure that output the CSOs have correct position and orientation.
  - New module `CSOCompare`: Compares two CSOLists by various criteria; can be used in regression tests.
  - New module `CreateCSOMapping`: Creates a mapping between closest CSOs in two CSOLists as input to CSOMerge.
  - New module `CSOVolume`: Computes the volume of an object defined by CSOs on all slices.
- New module `ImageShapeBasedInterpolation`: Interpolates slices of a mask image using the shape-based interpolation algorithm (same as `CSOShapeBasedInterpolation` but without CSOs).
- New module `DrawRectangle`: Masks out a 6D rectangular region in an image.
- New utility modules:

- `BaseCache`: Buffers the input object in memory and can protect it from input changes.
- `CalculateVoxelSum`: Computes the sum of all voxel values.
- `VolumeCenter`: Position of the center of a volume.
- `ImageSwitch`: Lets you select one connection between multiple inputs and outputs.
- `NormalizeOrientation`: Wrapper around OrthoSwapFlip module more convenient to use: Normalizes the volume to a given (orthogonal) orientation.
- `ChangeWorldMatrix`: Transforms the world matrix of an input image with another matrix.
- `XMarkerListCompare`: Compares two XMarkerLists, useful for testing.

- Overhaul of `XMarkerListConvert`, new modes for converting to/from integer voxel coordinates (useful for testing), completed tests, and moved from FMEwork to FMEstable.
- `Libssh` has been included in FMEwork/ThirdParty for the Windows and Linux platforms.
- Python module `mock` has been included in FMEwork/ThirdParty.
- Added new C++ library `MarkerConversion`, simplifying conversion of XMarkers and XMarkerLists between the three different image coordiante systems (world, floating point voxel, integer voxel (aka index)).
- Added convenient logging capabilities through library `SeverityChannelLogging` in directory `Packages/FMEwork/ReleaseMeVis/Projects/SeverityChannelLogging/Core/Sources`, a thin layer around Boost.Log
  - The framework supports the logging sources `scl::Logger` or `scl::ThreadSafeLogger`, has the severity levels *Debug*, *Info*, *Warning*, *Error*, and *Critical*, and logs an arbitrary channel identifier.
  - Supports different logging sinks, where the configured information is logged along automatic attributes like a timestamp. The sinks themselves allow for a detailed configuration of the output format.
  - New module `SeverityChannelLoggingSink`: Configures a logging sink with many parameters.
  - New module `SeverityChannelLoggingTester`: Configures a logging source for testing.
  - New module `SeverityChannelLoggingStringFieldSink`: Writes log records to a String field, is thread-safe.
  - New module `SeverityChannelLoggingPopupSink`: Sends a log record to a pop-up window.

Version 2.7.1 Stable Release (2015-10)

Note

Although this release includes a fix for OS X 10.11 (El Capitan), there are know issues on OS X 10.11. If you need the full functionality of MeVisLab, do not yet upgrade to OS X 10.11.

New Features

- Added auto-focus feature to `MLABGraphicsInventorRenderArea`: This item will automatically get the keyboard focus if the mouse moves over it.

- Added an `allowPopups` tag to the `WebView` control. Setting this to yes will allow HTML popup windows to appear (though not with the same feature support as the main control).
- Added a new pre-defined symbol "clear" for network panel buttons.
- ADK only: Added variable `$MAKE_ALL_OUTPUT_FILES_WRITEABLE` to installer wizard. But installers files are made writable in any case now if required (if they are signed or encrypted).
- The Visual Studio 2013 version of MeVisLab is now built with Visual Studio 2013 Update 5. We suggest you use the same version if you compile your own modules.

Fixes

- Fixed a bug where MATE would rename directories that have a file ending in their name and appear in the new project workspace. This especially affected Mac users (e.g., `MyProject.xcodeproj` was renamed to `MyProject`).
- Fixed random crashes when reading WEM files.
- Performance increase for WEM loading and generation (e.g., in `WEMIsoSurface`).
- Fixed that the output of `WEMIsoSurface` when generating in background was stretched if the sampling size in z direction was different from the voxel size in z. Also fixed timepoint handling for this case.
- Also fixed that `WEMIsoGenerator` could generate NaN values in rare cases.
- Fixed a crash in modules derived from `CSOModificator`.
- Restored removed CSO methods `removeFromGroup`, `addToGroup`, and `removeFromAllGroups` as deprecated methods.
- Added headers of `MLCSOBoostGeometry`.
- `SoView2DCSOExtensibleEditor`: `isCreatingNewCSO` field remained TRUE when current contour was deleted.
- `SoCSOCrossSectionRenderer`: Support non-orthogonal CSOs created on a MPR.
- `SoCSO3DVis`: Avoid to generate 3D objects for seed points that are effectively invisible.
- Fixed that some Inventor modules like `SoBackground` were painted a second time if modules like `SoAnnotation` are in the Inventor scene.
- Fixed `SoPostEffectRenderer` on Intel graphics and on OS X.
- Fixed `SoCSOFillingRenderer` on OS X.
- Fixed that `SoMouseGrabber` caused scene redraws on just turning the mouse wheel.
- Make sure that the `VTKGraphicsItem` can always be created in the `MLABGraphicsScene`.
- Fixed that some types of controls do not show in a `FreeFloat` control.
- Fixed a crash when accessing a `ButtonBar` control without an associated field from scripting.
- Fixed that calling `setFocus()` on a `Field` control did not work.
- `MLABCommonButtonGroupControl::removeAllButtons()` did not correctly remove the buttons.
- `SoTensorFieldVis`: Use correct color for ending points in vector rendering mode.
- Really allow to use an expression to set the button color of a network panel.

- Avoid a crash on reload of a module whose network panel has been removed.
- Installation of Pylint code checker failed because of missing functionality.
- De-highlight connected modules when a parameter connection is de-highlighted in the network view.
- Do not open internal network if there is no internal network.
- Fixed a problem with field connection dragging in networks on OS X. Also fix position of network tool tips on Retina displays.
- Fixed icon display problem in the MATE workspace area on OS X.
- Links in module documentation that point to another drive on Windows are correctly created now.
- Python class `ItemModelDataSource` now also supports items that are derived from `QObject`.
- Fixed some crashes in `GraphicsViewPresentation` example.
- Fixed a crash in `ConcatenateImages` if one manages to set `numberOfInputs` to zero.
- Fixed font issue on Mac with OS X 10.11 (El Capitan).
- Fixed ToolRunner required tool check on OS X.
- Inventor modules `SoCone` and `SoCylinder` were not correctly updated if the scaling factors were changed after first rendering.
- `QFileDialog` misleadingly allowed to select multiple directories and returned backslashes when selecting an existing directory on Windows Vista and later.
- Make sure `MLABScriptSqlQuery::allValues()` starts at the beginning even if `MLABSqlQuery::first()` was called before.
- Fixed a crash when two root instances of class `XMLTreeNode` (defined in project `MLParser`) were created in parallel. Also fixed a crash in `XMLTreeNode::readFromString()` on invalid strings.
- Documentation fixes.
- Some module panel layout fixes.

Version 2.7 Release (2015-06)

Fixes / Enhancements (MeVisLab)

- New features
  - MeVisLab now supports a new directory structure for modules: under the `Projects` directory (which is located directly in a package directory), sub-directories can have a `Modules`, `Sources`, `TestCases`, and `GoogleTests` directory, thus combining all sub-directories of a project.
  - The new multi-threaded ML host is now the default.
  - The restrictions for unlicensed use have been changed. The number of self-defined modules that can be used at the same time has been reduced to 5, but at the same time it is now possible to use regular Python scripting in modules with a file size of up to 2KB per module. This allows to check out the Python-specific features of MeVisLab like the debugger or the new Pylint integration. Note that the import of self-defined Python modules is not possible without a license.

- Modules can now inherit field documentation from prototype modules. This only works for ML or Open Inventor modules and requires that there is a module with the same name as the parent class of the inheriting module (as registered with the ML or Open Inventor class system). Invisible dummy modules can be added to the helpPrototype module group.
- Added `-userscript` command-line option for executing user scripts from the command-line.
- Improved number display for numbers with exponent or in too narrow widgets: The numbers are abbreviated to fit in the visible space and exponents are displayed in a more natural, non-programmer way (only while not editing).
- MDL syntax validation can now be extended by a user on the root level.
- Open Inventor fields which have `isContainerNotifyEnabled` not set are non-editable by default.
- Added support for the `LOCAL_BASENAME` variable for addressing, e.g., script files of a module in the MDL definition.

  Like: `source = $(LOCAL_BASENAME).py`
- IDE:
  - Improved network view: Modules can show a short info string and a small button in the network now. Also will some modules de-emphasize inactive inputs/outputs.
  - When multiple modules are selected all fields that are shared between the selected modules are shown in the Module Inspector and can be edited.
  - Mark non-default values of fields in the automatic panel/Module Inspector and add options to restore default values to the field/module context menu.
  - Some defaults for network appearance have been changed.
  - Added context menu for input and output connectors.
  - Added menu item **Copy Connection Info** to context menu of connections.
  - Added **Touch** entry to field context menu; this can be used to debug update problems.
  - Added **Copy Instance Name** to module context menu.
  - Added a **Recent Outputs** inspector.
  - Added a view for global MeVisLab MDL extensions (**Extras** → **Show Global MDL Definitions...**). This allows to find the definition of custom controls, wrappers, user scripts, and various other global MDL definitions.
  - Allow multi-line copy&paste into script console and behave more like an interactive Python console.
  - Added support for writing a trace file to the profiling view.
  - Improved quick search of documentation and menu items.
  - Added menu entry with shortcut **CTRL**+**T** for running tests of selected modules.
  - Users can now configure the Subversion command-line for Linux and Mac.

    - It is possible to use **CTRL**+**C** in the screenshot gallery (if it has the keyboard focus).
  - TestCaseManager:
    - Made it possible to run all tests of a package from the TestCaseManager.
    - The TestCaseManager now shows progress updates in secure mode.
    - Added menu entry and keyboard shortcut for repeating the last recent test case run.
    - One can now disable "info" message output in the TestCaseManager panel.
    - The debug console logging of a TestCase is now formatted in a human-readable way.
    - Allow to open the source code of a test at a specific function by invoking the context menu on the function.
    - Allow passing callables instead of truth values to the `disableTestFunctionIf` decorator.
    - Allow test groups to be disabled with the `DISABLED_` prefix.
    - `TestSupport.Base.expectError` and `TestSupport.Base.expectWarning` now take an additional regular expression as optional argument to specify the expected error or warning.
  - ADK:
    - The module dependency analyzer correctly detects Python modules under site-packages.
    - New tags have been added to the Deployment section of modules. See the description in the MDL Reference.
    - Setting `INCLUDEPDB` in an installer file will copy PDB files of required libraries.
- MATE (integrated editor)
  - Added workspace/projects feature: Files can now be organized into projects and are shown in a workspace area. The old session management is made obsolete by this.
  - Integrated automatic Python code-checking with Pylint (must first be installed from the Preferences dialog).
  - Python code completion has be rewritten using the jedi Python library. It supports much better completion on any Python object and the PythonQt wrappers. Hovering over text in Python now shows a tooltip with type information. Overload completion works much better now and only shows the matching signatures, depending on the current argument position. Goto reference and goto assignment have been added to the context menu. Completion now even works on script wrappers that are returned via `MLABMLBaseField::object()`.
  - Context sensitive help in Python code has been improved. All major script libraries are now supported, pressing **F1** (or context menu **Show Scripting Help**) on a selected variable will show correct help pages for: Python statements and base libraries, NumPy, VTK, OpenInventor, OpenCV, MLAB, etc.
  - Added **Find in Files** and **Replace in Files** support to find dialog.
  - Added option to **Find&Replace** dialog to replace all in selection.
  - Introduced a new graphical scrollbar similar to "RockScroll" (which can be deactivated).

  - Allow to run tests of an associated module from the toolbar.
  - Can also run the currently edited test case and test function (via context menu).
  - Fixed that root values in the **Watches** window in the Python debugger were not shown.
  - Documents can be saved automatically now.
- C++ API changes
  - Some deprecated functions and defines have been removed from the ML.
  - Introduced `setValue`/`updateValue`/`getValue` methods to ML fields.
  - Added `getVoxelVolume*` methods to `ml::MedicalImageProperties` (and thus `ml::PagedImage`).
  - Switched `MLABIntegerField` to use 64bit integers.
  - Added an `outputOnly` hint to ML fields. This tells MeVisLab to make these fields non-editable in the GUI.
  - Added `mlModuleProfiling.h` which offers the `ML_PROFILING_SCOPE` macros for user profiling in `ml::Module`.
- Controls
  - Ignored Open Inventor fields are now disabled in the module panel.
  - Removed support for the (broken) GUI tag `frameMargin`.
  - Can set `maxItemWidth` and `maxTextLength` property of `IconView` control now.
  - Support `@import` statement in `MLABWidgetControl::setStyleSheetFile()`. It allows to include other CSS files,

    either absolute: `@import "$(MLAB_SomeGroup_SomePackage)/Modules/.../some.css"`

    or relative: `@import "some.css"`
  - `ItemModelView` control:
    - The `ItemModelView` control can show and edit colors that are given in web notation (#rrggbb) in the item model.
    - Added column property `selectableAttribute` for controlling whether an item can be selected.
    - Added column property `floatDecimalPlaces` to determine the number of decimal places of floating point numbers.
    - Can set alignment of header with `headerAlign` tag.
    - `inspectorMode` now shows all attributes that are not used in fixed columns already.
  - Added new MDL tag `caseSensitiveAutoComplete` to `Combobox` control
  - Fix `Combobox` control initially touching enum fields under some circumstances.

  - Added new MDL tags `equalButtonWidths` and `equalButtonHeights` to `PushButtonGroup` and `ToolButtonGroup`.

- Scripting
  - MeVisLab now contains the complete VTK Python bindings. This allows to build arbitrary VTK pipelines using Python. The VTK objects in wrapped VTK modules can also be accessed via Python. For detailed information, have a look at the VTK Python Binding page.
  - `GraphicsView` now supports adding VTK scenes as a graphics item. Have a look at the VTKGraphicsViewExample.
  - Added Python bindings for Open Inventor.
  - We now ship the OpenCV Python bindings and some conversion scripts between ML and OpenCV images. Have a look at the OpenCVPlaygroundExample.
  - Added functions to query disk capacity to `MLABFileManager`.
  - Added scripting method `MLAB.createDicomUid()`.
  - Fixed resolving of images in `MLABPrinter.printHtml`.
  - Added scripting method `MLABEnumItem::isCurrentItem` which also checks deprecated item names.
  - Changed the location of Python stubs file (for supporting external Python editors): It is now located in the MeVisLab preferences directory under `PythonStubs/mevis.py`.
  - The content of the Python stubs file has been much improved. It improves code completion and static code checks for external editors.
  - Added convenience method `StandardItemModelWrapper::removeItem` which takes the index of the item to remove.
  - Added `CSOListWrapper::addCSOCopy(CSOWrapper*)`, `CSOWrapper::setId()`, `CSOWrapper::isSelfIntersecting()`, and `CSOWrapper::setSeedPointsAndPathPoints()`.
  - Added `isValid()` method to WEM scripting wrapper.
  - Added a function to `WEMGeometry` to compute the distance of a point to a triangle.
  - WEM script wrappers do range checking now.
- Other fixes and improvements
  - The ToolRunner now searches for profiles in the sub-directory `CodeTests` as well.
  - Remote modules now support non-scalar voxel data types for image outputs.
  - Image connections from remote modules will convey the detail information about C, T, and U dimensions.
  - Improved handling of `PixelRepresentation` tag in the `DicomTree` library.
  - `SoManagedInteraction` got a new action type: `SoOffsetAction` maps most naturally to mouse wheel rotation, but can also be mapped to mouse drags and key presses. See Managed Interaction Overview.

  - One can use the environment variable `MLAB_CURRENT_PACKAGE_DIR` in `.pro` files, it is set by `createProject.py`.
  - Copied `mlSystemWarningsDisable/Restore.h` to their own directory and renamed them to `ThirdPartyWarningsDisable/Restore.h`.
- Third-party libraries
  - boost has been updated to version 1.57.
  - Made `boost_chrono`, `boost_asio`, and `boost_log` available.
  - Open Inventor:
    - Added `SoSFTypedEnum<>` template to allow typesafe usage of enum fields.
    - Switched `SbBool` from int to bool.
    - Added `SoRef<>` smart pointer class for all `SoBase` derived classes.
    - `SoCube`, `SoCone`, `SoCylinder`, and `SoSphere` now support VBO rendering.
  - Added Qt translation tools to SDK.
  - Added support for intermediate SSL certificates to Qt.
  - Updated OpenSSL library to version 1.0.1m.
  - PythonQt now handles ownership of object instances correctly for most Qt methods.
  - The GoogleTest framework has been updated to version 1.7.0.
  - Python:
    - Python has been updated to version 2.7.9.
    - SQLAlchemy has been updated to version 0.9.7.
    - Added rarfile module.
    - Added jedi module; used for Python auto-completion in MATE.

Fixes / Enhancements (Modules)

- General
  - Some long-standing deprecated and unmaintained modules have been removed.
  - Fixed a problem with scrambled input event order when using break checking.
  - MeVisLab now supports remote rendering of VTK renderings using the new `VTKRemoteRendering` module.
- SoView2D and extensions
  - `SoView2D` keyboard shortcuts can be remapped when using Managed Interaction mode.

  - SoView2D got an additional cine mode CINE_TZ that iterates first over T dimension and then over Z dimension.
  - SoView2DLabel can now position text relative to the viewer's visible device rectangle.
  - Added an option to shade all exterior of SoView2DRectangle.
  - Added lineStyle option to SoView2DBorder and SoView2DRectangle.
  - Fixed overlay LUT of SoView2DOverlay sometimes not updating.
- Other Open Inventor modules
  - Fixed a crash in SoTexture2 with large non-power-of-two images.
  - SoDiagram2D, SoWEMRenderer, and SoCSO3DVis were switched to use the Managed Interaction framework for input event handling.
  - Added Managed Interaction support (through option flag) to So3DMarkerEditor.
  - Added new module SoPicking that is basically SoSelection2 with Managed Interaction.
  - Fixed origin translation of bounding box rendering and calculation of SoAxis.
  - Added new renderer module SoWeightedBlendedOIT as a faster alternative to SoDepthPeelRenderer.
  - Introduced a new general GLSL shader pipeline framework which supports a flexible way to build reusable shader functions and surface shaders, while interacting correctly with existing Open Inventor lights and fixed function state. It allows to separately modify the light model (BlinnPhong, Phong, Lambert, or your own custom model), the surface shading and how fragments are stored or discarded (for Depth Peeling, Ambient Occlusion, etc.).
  - The new module SoPointSpriteRenderer allows to render hundreds of thousands of correctly shaded spheres without generating a mesh geometry using OpenGL point sprites. The So3DMarkerEditor now offers a PointSprite render mode, which makes use of the SoPointSpriteRenderer to render huge amounts of markers as spheres.
  - The SoDrawInstanced module allows to render the same geometry multiple times with different color, scale, rotation, and translation. It uses instanced draw array/elements OpenGL calls on hardware that supports it and automatically falls back to multiple draw calls on older hardware. The module supports all Open Inventor geometry nodes that support VBO rendering.
  - The new GLSL post effect framework offers a complete OpenGL post effect framework for Open Inventor. It supports various screen-space post effects: FXAA antialiasing, ambient occlusion, glow, edges, depth halos, and custom effects. Have a look at the various example networks!
  - The SoDiagram2DCursor cursor position can be set from the outside now.
  - Print an error message when SoGVRRenderer is used below a depth peeling module - this is unnecessarily slow and should be avoided.
  - The hard-coded keyboard shortcuts in SoRenderArea and SoExaminerViewer can now be switched off through the enableKeyCommands field.
- CSO

- SoCSOSplineEditor got an option to close freehand CSOs on release.
- Added option for filtering of visible CSO ids to SoCSOVisualizationSettings.
- SoCSOFillingRenderer: can set OpenGL blend function now.
- Added flag provideVertexProperties to SoCSO3DVis which provides normals and texture coordinates to shaders if checked.
- Made CSOFilter not reorder the list of CSOs.
- Optimized CSOConvertToImage if CSO lies in one z-plane.
- CSOCasing: should not shift voxel position so that all components are positive.
- Fix path point generation in SoCSOSplineEditor for small voxel sizes.
- Activated the AUTO_CLEAR mode for CSOConvertToImage, CSOConvertTo3DMask, and CSOFilter.
- Implemented optional asynchronous loading/saving of CSOs in CSOLoad and CSOSave.
- SoView2DCSOExtensibleEditor can now restrict interaction to the image bounds.
- Many other fixes and improvements.

- WEM
  - WEMReducePolygons got an optional edge swapping post-processing step.
  - WEMSceneLoader generates an empty patch if degenerate faces are present.
  - Fixed normal generation in WEMSceneLoader.
  - WEMVascularSystem
    - Fixed that for large images 'bumpy' edges were created or missing at all.
    - Fixed that branches are cut off when using high smoothness at branchings.
    - Can now optionally generate smooth furcation joins (rims).
  - Added an XMarkerList output to WEMClip which encodes the WEMNodes' positions and normal vectors.
  - WEMIsoSurface can calculate in the background now.
  - Introduced a new version of the WEM file format with configurable content.
  - WEMGeodesicClip has been improved.
  - Removed WEMBoolOp: It didn't work correctly.

- ML modules
  - Changed module Reformat to support AutoUpdate/AutoClear modes.
  - MergeRegions: Fixed wrong output world matrix translation for non-copy merge modes with a 'use input N' region selection mode.

  - Fixed application of `worldToVoxelGridConversionTolerance` in `MergeRegions`.
  - `SetVoxelValue` supports all available voxel datatypes now.
  - Added modules to filter attributes (`ItemModelAttributeFilter`) and items (`ItemModelItemFilter`) from an ItemModel; item filtering uses a simple expression parser.
  - The original `ConnectedComponents` has been modularized and optimized.
  - Fix stencil buffer support in `RemoteRendering` and `OffscreenRenderer` modules.
  - Added new modes to `Arithmetic1`: Subtract constant and divide by constant.
  - Added new module `Round` which rounds voxel values to integer.
  - `MovieCreator` now uses the `VideoFoundation` framework under Mac OS X.
  - Added asynchronous loader modules `AsyncMemoryImageLoad` and `AsyncMemoryImageFetch` for simultaneously loading and displaying large image files.
  - Added `AccumulateImage` module for accumulating images consecutively, and module `CreateBoundingBox` for possibly creating a reference image first.
  - Added RGB to gray conversion to `ColorModelConverter`.
  - Added new module `ClusterToMarkers` which computes the centers of gravity of clusters of same integer voxel values in images.
  - Improved algorithm of `ConvexHull`.
  - Added some options to the `Threshold` module.
  - `MinMaxScan` bool field `preferUnsigned` was replaced by enum field `preferredIntType`.
  - Fixed `RegionGrowing` auto-clear bug and other bugs.
  - `ConstrainedConnectionCost`: Added support for page sizes > 1 in C, T, or U dimension.
  - Fixed border-handling errors in `ConstrainedConnectionCost`.
  - Added `Rank` filter mode `Max-Min`.
  - Deprecated `LoG` module.
- Macro modules
  - Added option to record (longer) pauses to `RemoteRenderingEventRecorder`.
  - `DicomTagBrowser` shows the content of `UN` tags if they seem to contain a string.
  - Removed `PythonProcessAllPagesExample` which was kind of pointless.
  - `RunPythonScript`: Can use `ctx` instead of `CTX` in code now.
  - Fields of `RunPythonScript` can be accessed via `ctx.owner().owner()` or `RunPythonScript_ctx`.
  - Added module `DrawVoxels3DPreview`.

- Added module `RemoteRenderingEventReplay` .

Windows Edition

- Opening files with MATE from the command-line raises an already running MATE to the front.
- Switched default Visual Studio compiler flag from `/Z:wchar_t-` to `/Z:wchar_t` for better compatibility with `boost_filesystem`.
- ADK: Fix Windows installer creation to support long (> 256 characters) paths.

Linux Edition

- Fixed installing to relative paths under Linux and check against non-empty directories.
- Fixed restart of MeVisLab not working under Ubuntu 14.04.
- Fixed problems with fontconfig under newer Linux versions.

OS X Edition

Note

This release requires at least OS X 10.9.5 and Xcode 6.2 for development.

New Features

- OS X 10.10 (Yosemite) is now supported.
- High resolution rendering is now enabled by default on capable hardware.
- Allow configuration of high-resolution rendering default for viewers via Boolean preferences variable `HiResRenderingDefault`.
- Allow the visibility of icons in menus be controlled by the `DontShowIconsInMenus` Boolean preferences variable.
- The application license (required to run MeVisLab-based applications) will no longer be installed within the application bundle. Instead the user is asked to choose whether the license will be available to all users or just the current user which will determine the storage place.
- New icons for MeVisLab, tools, and documents

Bug Fixes

- Menu items defined in MDL that only have an icon and empty text will now show the icon again. If the icon is set via scripting and there will be no text, the scripting function `setItemIconVisibleInMenu()` enforces its visibility.
- Fixed unresponsive behavior with specific wrapper executables used as Supportive Programs (e.g., open).
- Fixed status reporting for privileged tool execution.
- Fixed QuickLook issues for .mlab files
- Replaced deprecated QuickTime API by Audio Video Foundation API.

- Removed support for deprecated pixel buffers.

Modules and Libraries Provided by Fraunhofer MEVIS

ITK and VTK

- `itkFileWriter`: It turned out that the default setting of the module `itkImageFileWriter` is not high enough when writing an ITK-MetaImage file (mhd), so that some precision on offset and voxel size of the written output images could be lost. The internally used precision was increased to 17.
- `itkImageFileReader`: Fixed buffer overrun on long path names.
- `itkImageFileReader`, `itkImageFileWriter`: Replaced JavaScript with Python.
- `itkGradientDescentOptimizer` and `itkMutualInformationImageToImageMetric`: Fixed incorrect `PagedImage` updates and error messages.

Fixes/Enhancements (FME Modules)

- Macro module `BoundingBoxExt` has been reworked:
  - Renamed to `BoundingBoxWithMargin`
  - Reworked Python code and field interface. Module now makes use of the `AlgorithmMacroModule` template.
  - Updated documentation and example network.
- `MLAlgorithmModule`
  - The project wizard now adds `FMEstable_ReleaseMeVis` to `MLAB_PACKAGES` in the project file if the project is created in a package outside of FME package groups (~work, ~stable).
  - Reworked `Check` classes of Python module `AlgorithmModuleTestSupport`.
  - Per default logging on success is now disabled at Python modules Tests and Checks of `AlgorithmModuleTestSupport`.
  - Renamed example module.
  - Reworked automatic update/clear field interface: Replaced bool field `shouldUpdateAutomatically` with an enum field `onInputChangeBehavior`.
  - Moved include files `Description.mdl` and `Window.mdl` of `MLAlgorithmModule`. Adapt include commands to new location at .script files of modules using the files:

    ```
    #include "$(MLAB_FMEstable_ReleaseMeVis)/Projects/AlgorithmModule/
    MLAlgorithmModule/Modules/Description.mdl"

    #include "$(MLAB_FMEstable_ReleaseMeVis)/Projects/AlgorithmModule/
    MLAlgorithmModule/Modules/Window.mdl"
    ```

- `ConstantXMarkerList`
  - Enabled parsing of input without line feed.
  - Added `activateAttachments` to align module output with input string on reloading/instantiation.

    - Added parsing of vector components for `XMarker`.

- New modules:

    - `CSOWorldBox` calculates the world axes aligned bounding box of CSOs.

    - `CSOVoxelBox` calculates the image axes aligned bounding box of CSOs.

    - `AffineMatrixComposition` combines quaternion vectors for translation, rotation, scaling, and stretching to a 4x4 Matrix.

    - `AffineMatrixDecomposition` applies a polar decomposition of a non-singular 4x4-matrix into quaternions (translate, rotate, scale, and stretch).

    - `PolarVectorScaling` allows for sense- and meaningful scaling of quaternion vectors for translation, rotation, scaling, and stretching.

- Removed obsolete `ExpandFileName` which can be easily replaced by the `LocalFileName`.

- `DirectDicomImport`

    - Fixes:

        - Fixed bottlenecks when loading large result cache files.

        - Multiframe pixel data in tag dumps is shown correctly now.

        - Fixed crashes on imports where single and multiframes files were composed into same volume.

        - DPL configurations revised, numeric tolerances added, and parting options are listed first to avoid that parting tags are missed.

        - Fixed loss of some frame specific tags while importing multiframe files.

        - Support different data/voxel types in a DICOM series is possible with new DPL option now.

        - Fixed incorrect modality detection on Secondary Captures.

        - Programmatic updates of volume index updates highlighted row in volume list now.

    - Improvements:

        - DICOM tree generation does not create an SMF tree for a single frame any more but uses the original tree.

        - Installer generation adds loader plugins and private tag decoders via entries in deployment sections now.

        - DICOM frames without `ImageOrientationPatient` tags can be imported and sorted with user defined orientation.

        - Logging compacted and error handling unified.

        - DICOM frames with duplicate `SOPInstanceUID` tags are detected and rejected with error message before importing them.

        - New DPL option to skip private tags before sorting and importing to speed up import and reduce memory usage.

    - SMF files in result caches carry the prefix of the cache file now.
    - Enhanced US images without `ImagePositionPatient` tags can be sorted according `DimensionIndexValues`.
    - Sped up image data requests on images without `Image*Patient` tags.
    - Loader interface for the Digital Micrograph file format, version 3 and 4 available.
    - Added support to collect and compose meta data of non DICOM files.
    - Added an extensible loader interface for new file formats.
    - Improved page extent settings, sped up slow import and data retrieval from heaps of non DICOM files.
- Tag modification and DICOM export
    - Added new module `DicomModifyCreateTree` which allows the creation and modification of DICOM trees from scratch.
    - Added new module `ModifyDicomTree` for explicitly filtering a DICOM tree.
    - `DicomPixelModifier` supports different rescale/slope tag configurations on RT frames now.
    - `FileListFilterPlugin` supports most DPL configuration settings now.
    - `DicomTreeCompare` handles and compares multiframe pixels now.
    - `DirectDicomImportDcmtkOutput` modules renamed to `MultiFileVolumeListDcmtkOutputs`.
    - Fixed some misinterpretations of some private (0051,*) DICOM tags.
    - `DicomModifyTagsPlugin`: Fixed incorrect changing/adding of multiple values per tag with.
    - `DicomModifyImageTagsPlugin`
        - Sets up rotational matrix parts in global modification mode now.
        - Fixed missing support of DICOM frames without orientation or position information.
        - Fixed missing shifts between original and modified image.
        - Supports Secondary Capture images with missing `ImagePositionPatient` tags in `AdjustSlicewise` modes now.
    - `DicomTreeCompare` does not generate useless empty sections any more in cases where tags are ignored.
    - `DicomFIDSave` and `DicomREGSave` now set the `SOPClassUID` tag.
    - `ApplyDicomPixelModifier`: Fixed incorrectly modified frames with no `RescaleSlope` tags if other frames have valid ones.
    - `DicomModifyPrivateAdd(FieldAddOn)`: Fixed too strict clamps of element id to 0x10.
    - `MultiFileVolumeListImageOutput`: Fixed odd warning on empty directory selection.

  - `DicomTreeValidate`: Produces significant less false positives and can check allowed sizes of tag values now.
- DICOM
  - Deleted `RTPlanToML`, `RTRecordToML`, `RTIonPlanToML`, `RTIonRecordToML`, as they were obsolete. Corresponding info modules, e.g., `RTPlanInfo`, should be used instead.
  - Added shared project `MLDicomModifyFieldAddOns` that enables tag modification of various DICOM modules.
  - Added `MLDicomModifyRTPlugins` project with modules allowing modification of DICOM RT objects.
  - Added modules for creating RT IODs: `CreateRTStruct` and `CreateRTDose`.
  - Improved `RTObjectSave` module: Added **Save In Background** button, enhanced module panel, fixed some bugs.
  - Marked `MLToRTStruct` and `MLToRTStructureSetConverter` as deprecated, `CreateRTStruct` and `DicomModifyMLToRTStructPlugin` should be used instead.
- Other
  - Platform dependent print outputs in print tool functions fixed.
  - Added support and source code ports to new gcc4.9 and VC14 compilers.
  - Added shared project `MLCSOBoostGeometry` enabling calling some of the `boost::geometry` algorithms directly with CSOs.
- New public sources in package `FMEwork/ReleaseMeVis`
  - File reader plugins and derived classes:
    - `Sources/Shared/MLFileReaderPluginsBase`
    - `Sources/Shared/MLFileReaderPlugins`
    - `Sources/Shared/MLDMFileReader`
  - Simple ImagePropertyExtension storing only the original file name and a string for meta data:
    - `Sources/Shared/MLStringImagePropertyExtension`
  - Module sources:
    - `Sources/ML/MLColorFromName`
    - `Sources/ML/MLDicomAnalysis`
    - `Sources/ML/MLMultiFileVolumeListConverters`

Version 2.6.2 Stable Release (2015-03)

New Features

- Now allows to set JPG quality from `RemoteRendering` module.

- `ml::BackgroundTask::executeNextMessage()` now returns if there was any message to process.
- ADK: Now supports debug/release and version specific subdirs in package lib directories. Only adds existing directories to PATH.
- ADK: Added `HTML_ESCAPE_IN_VARIABLE` command to allow HTML/XML entity escaping.

Fixes

- Added missing files to the SDK.
- Changing the texts of a button group control now correctly updates its size.
- Corrected transformation for `WorldStartEndImageAxisAligned` mode in `SubImage` module.
- Fixed buggy stack image in `MPRPath` module.
- Fixed `setImageFile` method of Image control (didn't work if the newly set image had the same size as the previous image).
- `RemoteRendering` and `OffscreenRenderer` now support FBOs with stencil correctly.
- Matrix fields now work in remote modules.
- Fixed a crash when the number of backslashes in `DicomTagModify.modifyTagsString` was insufficient.
- `DicomModifyTagsPlugin` now allows to set private DICOM tag ids in the range 00..0F that were previously forbidden.

Version 2.6.1 Stable Release (2014-07)

Platform-independent Modifications

- Added scripting functions `MLAB.MEVISLAB_VERSION` and `MLAB.ML.MLGetVersion`.
- Added scripting method `getSeedPointPositionAt` to CSO wrapper.
- The sources for the MLOpenCLKernel1 library are now included in the release.
- Added source and library files of MLAlgorithmModule to the installer.
- The splash screen is not shown anymore when MeVisLab or a derived application is installed.
- Volume rendering in the `SoExaminerViewer` did not abort early when new input events arrived, which made the rendering of big volume data somewhat unresponsive.
- Fixed `WEMVascularSystem` creating 'bumpy' edges or totally missing edges for large images.
- Fixed that `SoView2DAnnotation` assumed that a string would not fit in width if (one of) the first lines in a block was empty.
- Pass on double-click information to an Inventor viewer in the GraphicsView control.
- Fixed icon drawing with class `View2DIcons`: The red and blue channels were swapped.
- Fixed access to preferences variables with ML functions: Only the first access worked, subsequent accesses always returned the same value.

- Fixed that changing the 'interpolate' field of `SoGVRMaskVolume` did not update the interpolation immediately.
- Allow to promote local macro modules to global macro modules with the Macro Module Wizard.
- Fixed handling of the `importPath` tag of modules, which led to some small problems with MATE and the debugger.
- Plugged some memory leaks in ItemModels and CSOs.

Windows Edition

- The matplotlib Python package was not available in the Visual Studio 2013 version.
- Adapted ToolRunner to generate Visual Studio 2013 solutions.
- Fixed broken event order of Managed Interactions when dragging causes longer calculations.

Linux Edition

- Added some tweaks for the Mesa llvmpipe renderer.
- Added V4L libraries, which are needed for OpenCV and the `VideoCapture` module.

Mac OS X Edition

- Pixel Scale Factor for Retina support is now correctly reported from the SoExaminerViewer.
- Fixed modal session related warnings with dialogs

Version 2.6 Release (2014-05)

Fixes / Enhancements (MeVisLab)

- New Features
  - Support for Visual Studio 2013.
  - You can now drag field connections from and to field buttons in a module's panel.
  - The default ML cache size is now a fourth of the available system RAM if not specified otherwise in the preferences.
  - The `min` and `max` tags of a field in a module description may now contain field expressions.
  - The version string displayed on the splash screen of MeVisLab applications can be overridden by setting the variable `UserSplashScreenString` in the preferences file.
  - Added `relatedFiles` tag to module description, to manually add files to the "Related Files" section of a module's context menu.
  - Support for the `Panel` control of modules moved to separate processes. The necessary sub-module fields are now automatically added to the interface of the proxy module.
  - `MenuItems` now support Python callables in addition to string names for script functions.
  - Introduced a new mechanism for getting third-party library dependencies for application installers.

- Implemented copy&paste of networks across MeVisLab instances.
- Parameter connections in networks now use different connection points for inputs and outputs to clarify the direction of the connection.
- The output field connection that is currently displayed in the Output Inspector is marked with a "halo" in the network.
- The `editable` and `persistent` attributes are now inherited from the internal field when a field is specified with `internalName`.
- Local macro modules may be taken from a sub-directory.
- Various improvements to the TestCaseManager:
  - The TestCenter supports generation of Python coverage data.
  - Support for grouping test cases with the `TestCaseSuite` tag.
  - Added a recent test cases menu to the "Files" menu of MeVisLab.
  - Added support for progress display, cancellation, and stop-on-error to the TestCaseManager.
  - Test cases are automatically reloaded when the Run button is pressed.
  - Secure testing is not the default anymore.
  - Added keyboard shortcut Ctrl+Alt+T to start the TestCaseManager.
- The Python library SQLAlchemy was moved to MeVis/ThirdParty.
- The Threading Building Blocks (TBB) library was added to ThirdParty and may be used in C++ modules by adding CONFIG += tbb.
- The diagnosis console got a message filter, which can be configured from the context menu.
- Removed conversion option for old 1.6 user packages from the project wizard.
- Better support for allowed DICOM string encodings - like certain Chinese encodings - in DicomTree library when loading DICOM files.
- Added new Qt widget explorer (Extras->Show Widget Explorer), which allows to debug Qt widget trees for CSS styling.
- When opening Visual Studio from a C++ module's context menu, prefer the version specified with MLAB_COMPILER_VERSION.

- MATE (Integrated Editor)
  - Removed option to run MATE inside of MeVisLab process, it was not tested anymore and some things have stopped working.
  - When clicking the header in MATE's session view, the documents are sorted.
  - MATE shows field info as tooltip over field names.
  - MATE shows variable values as tooltip when stopped in a debugger breakpoint. The currently selected stack frame is used.

  - Display of variable values improved for Python debugger.
  - MATE can show HTML pages now. This is used in the help generation.
- C++ API Changes
  - Support for resolution-independent rendering (See MeVisLab Resolution Independence API documentation).
  - Added templated class `TypedEnumField` to ML, which allows to store and retrieve enum values without casting.
  - Added class `TypedProcessing` to ML, which allows to call a templated process() method based on a given ML data type. This is similar to how the TypedHandlers work and can replace the old macro-based mechanisms for instantiating templated methods.
  - Removed the `MLSmallImageInterface`.
  - New MLUtilities function `MLClampIntervalToDataTypeRange` to clamp a min/max range against the range of a target data type.
  - Added double precision variants of existing single precision Open Inventor calls and classes, e.g., `SbVec3d`.
- Controls
  - The ItemModelView control gained a `visibleOn` tag for single columns.
  - Added `boldFontAttribute` and `italicFontAttribute` for columns to the ItemModelView control.
  - Added `autoExpandAll` attribute to ItemModelView control to automatically expand every new item which has children.
  - Renamed `doubleClickField` to `doubleClickedField` in ItemModelView control.
  - Radio check boxes in the ListView control can now be styled.
  - Added MDL tag `widgetName` to controls, to set the objectName of the underlying Qt widget. This can be used in CSS styling.
  - The orientation of the ProgressBar control can be set to vertical.
  - Added scripting method `setStepButtonsVisible` to NumberEdit control.
  - The EventFilter control provides a 'isKeyRepeat' attribute for key events.
  - The ComboBox control now also supports enum fields.
  - The controls Horizontal and Vertical now support the tags `preferredWidth` and `preferredHeight`.
- Scripting
  - Added a framework for providing registration information to, e.g., CSO visualization modules. Have a look at the scripting class `MLStandardTransformationProviderWrapper`.
  - Made SSL support available in standard Python modules.
  - Improved the CSO scripting wrappers.

- `ctx.unexpandFilename` prefers the longest, most specific variable match now.
- Added support for dynamic attributes to Python wrappers; use it by implementing the slot `py_dynamic_get_attrib`.
- Added new class `PythonItemModel` in Python module `ItemModelSupport` that gets its data directly from a Python object. This is an alternative to the `MLStandardItemModelWrapper` class.
- Added OutputInspector for ItemModels.
- Correctly check for lists and tuples in PythonQt.
- New scripting method `MLABFileManager.isPathInside`.
- New scripting method `MLAB.unsetVariable`.
- Added the functions `Logging.showDiff()` and `Base.createHtmlDiff()` to the TestSupport package.
- Added the functions `Image.getTrueMinMaxValues(outFieldName)` and `Image.verifyMinMaxPropertyValues(outFieldName, ...)` to the TestSupport package.
- Added `ctx.logError()` and `ctx.logWarning()` scripting methods.
- `MLABMLBaseField` and `MLABSoNodeField` got the scripting method `isNull()`.

- Fixes
  - The new multi-threaded ML host got many fixes and performance improvements, and many modules have been changed to make better use of multi-threading. Use of the new multi-threaded ML host is still not the default though and must be activated from the preferences.
  - Improved module dependency analyzer to find more DLLs that should go into an installer and also to cope with encrypted Python files. Also, when scanning networks, local macros of that network are now correctly regarded.
  - Fixed a crash in the eatDicom tool used by `DicomImport` (only seemed to occur under Windows 8 when compiled with Visual Studio 2012 and later).
  - Worker processes for modules moved to separate processes usually don't inherit file handles anymore.
  - The network scripting window honors the "Stay on top" setting from the "Panels" menu.
  - Several help editor and help generation fixes.
  - Opening files in MATE from the command line works now.
  - The field help tooltip in module panels is also shown for imported panels and fields of internal modules.
  - Fix setting the cursor shape from Inventor viewer in GraphicsView control when input events are marked as handled.
  - Context menu controls now inherit the style from their parent control.

Fixes/Enhancements (MeVis/Foundation)

- New LZ4 compressor available in `DataCompressor` library and available in `MLImageFormat` classes (used as default by `MLImageFormatFileCache`).
- Added overview help page 'Processing Large Volumes in MeVisLab'.
- Test coverage of `MLImageFormat`-Modules and classes improved.
- MeVisLab-wide improvements of C++11 compliance.
- Many warnings removed in tool classes and header files.
- Enhanced experimental compile settings available in .pro files to be enabled before the .pri file includes in .pro files:
  - `MLAB_ENABLE_QA_RELEVANT_WARNINGS = 1`

    If this is not empty, all warnings considered useful for QA relevant code are enabled.
  - `MLAB_HANDLE_QA_RELEVANT_WARNINGS_AS_ERRORS = 1`

    If this not empty, warnings which normally can be removed easily are handled as to errors. Especially in older projects a few warnings still appear in large amounts and cannot be removed without greater effort. These warnings are tolerated although they are not wanted any more.
  - `MLAB_HANDLE_ALL_QA_RELEVANT_WARNINGS_AS_ERRORS = 1`

    This is the preferred setting which switches all warnings to error and guarantees warn-free code. It should be used for new developments and preferably for all projects which can be ported.
- To be able to handle warnings from included third party files such as STL, boost, etc., the following includes can be used:
  - `#include <mlSystemWarningsDisable.h>`
  - `#include <TheThirdPartyInclude.h>`
  - `#include <mlSystemWarningsRestore.h>`

Fixes / Enhancements (Modules)

- General
  - Open Inventor events now get their double-click and auto-repeat flags from the operating system (previously double-clicks were self-detected).
- Open Inventor Modules
  - Added module `SoView2DOverlayMPR` for displaying an non-aligned overlay image on any underlying SoView2D/SoOrthoView2D using a fast software MPR algorithm.
  - The module `SoCameraInteraction` got buttons to set the camera to the six cardinal directions.
  - The module `SoCameraInteraction` got key commands for rotating the camera (which must be mapped to actual keys with `SoInteractionMapping`).
  - The module `SoView2DVectorFieldView` now also supports 4D vector fields.

- Added module `SoView2DLabel` for displaying a text label at a specific world position.
- `SoView2DSliceZoom` now has an option to zoom around the clicked position.
- Added module `SoView2DCurrentPixelSpacing` to get pixel spacing in current SoView2D.
- `SoVascularSystem` got more modes for interactive highlighting.
- `SoLUTEditor` did not like image ranges > 16 bit and also got quite slow with large ranges.
- Added module `SoRotateCamera` to perform an automatic camera rotation just as in `SoExaminerViewer`. This can be used with the `OffscreenRenderer` module.
- Added module `SoDiagram2DPan`.
- `SoDiagram2D`: opacity of curve area can be changed.
- `SoDiagram2DCursor`: can select which axis to show now.
- Fixed possible render problem when using multiple `OffscreenRenderer` modules (the OpenGL state was not always correctly set).
- `SoKeyGrabber` and `SoGenericCommandAction` know more key codes now.

- CSO
  - Lots of work happened on the `SoView2DCSOExtensibleEditor` and associated modules.
  - Distance lines can be snapped to certain angles in the `SoCSODistanceLineEditor`.
  - Fixes to the freehand drawing of `SoCSOLiveWireEditor`.
  - Added imageSumValue to module `SoCSOImageStatistics`.
  - Added module `SoCSOAnnotationDeviceCoordinates` that displays a list of visible CSOs and positions in device coordinates to anchor some widgets in a GraphicsView relative to the visible CSOs.
- WEM
  - Added loading of a subset of VRML to `WEMLoad`.
  - Improved loading of STL in `WEMLoad`.
  - `WEMLoad` can now handle empty patches in a .wem file.
  - Interpolation of PVL values made optional in `WEMReducePolygons`.
  - Added rendering of face normals as mode to `SoWEMRendererNormals`.
  - `WEMIsoSurface` was refactored and got a lot faster for large images.
  - `WEMSelectPatches` and `WEMComposePatches` were moved to the Standard package.
- ML modules
  - `MatrixArithmetic` computes determinant, trace, and sum of the output matrix now (instead of input matrix A).

- `OtsuThreshold` can completely ignore voxels below a threshold value.
- `ConnectedComponents` got overhauled.
- Improved `MPRPath` and `PathToKeyFrame` for stretched CPRs.
- `MPR` and `MPRPath` modules were rewritten to support slab projection/rendering.
- Created an `ImageToLUT` module which creates a LUT from an image (inverse operation to that provided by `LUTToMLImage`).
- `ImageSave` now saves RGB(A) DICOM images with contiguous color ordering.
- `MergeRegions`: computed world box now covers the border voxel region of the reference image entirely.
- `MergeRegions`: handles C/T/U dimension info.
- `MergeRegions` has an option now to refuse merging of images with non-aligned voxel grids.
- `MergeRegions`: fixed output image property computation when region mode is 'use input X'.
- `RegionGrowing`: fixed label computation on restore.
- `RegionGrowing`: fixed inversion of masked image output.
- `TestPattern` now sets a sensible color info on the output image.
- `ImageSave` will not refuse to save images with non-fitting color information.
- Added Overwrite mode to `Arithmetic1`.
- Added additional image input to `DicomTagModify` to copy selected DICOM tags from.
- `DeMosaic`: better detects matrix size in some cases and does forward DICOM information now.
- Added VoxelStartEndOffset mode to `SubImage` module.
- Added modules `CloneImageAsConstantImage` and `CloneImageAsTestPattern` which can be used to reproduce failing networks without providing the actual image data.
- Removed field `ignoreNextNotify` from ML module `Bypass`.
- Added module `DrawVoxels3D` as a successor for `Draw3D`.
- Re-factored and modularized `DicomQuery` for querying DICOM nodes. It should work better and faster now. This uses the new "DicomTools" scripting extension.
- `Resample3D`: invalidates output image if `useReferenceImage` is set and no second image input is provided.
- The `ImageClear` module got the `updateDone` trigger field. This can be used to solve certain update issues in complex image processing networks.
- `ImageHash`: added `update` and `autoUpdate` fields.
- Added optional Enter/Leave recording to `RemoteRenderingEventRecorder`.

- `ConcatenateImages` allows up to 8 inputs now (e.g., red/green/blue in c dimension, or several features in u dimension). C/T/U dimension information is also concatenated if present.
- Added field `suppressUpdates` to module `RemoteRendering`. This can help to suppress flickering when the scene changes.
- The modules `Diagram2D` and `ExtDiagram` have been removed and can be replaced with `SoDiagram2D` (which is available since MeVisLab 2.3).
- The modules `AccumulateImage`, `DrawPrimitives`, `Lut`, `MevisLogo` and `OrthoMPR` have been removed.
- Module `MLImageFormatFileCache`:
  - Fixed a rare bug which made region overwrite/update fail with error message.
  - Fixed a potential file leak when switching between enabled and disabled cache file preservation.
  - Uses now LZ4 data compresses as default setting.
  - Provides currently used cache file path in a field.
- Module `ImageCompare`: fixed rare comparison bug for voxel value epsilon > 0.

Windows Edition

- `UIDGenerator::getMac()` fixed if only Wifi adapters are available. This fixes the inability to save DICOM files in this case.
- Support for Visual Studio 2013 with VC12 compiler.
- Windows-related fixes of multiple `CLIImporter` problems.

Linux Edition

- Using fontconfig for getting Open Inventor fonts under Linux, and fixed buggy character alignment.
- Fix for `boost::TIME_UTC` compile problem under newer Linux versions - renamed to `TIME_UTC_` for all systems.
- Support for Ubuntu 14.04 LTS version (Trusty Tahr) with gcc4.8.2 compiler.

OS X Edition

Note

This release requires at least OS X 10.8.2.

New Features

- Support for retina display Macs.
- New artwork with retina resolution.
- ToolButton controls support the autoRaise tag on OS X as well.
- Added mevislab-start-release.app to MeVisLab.app/Contents/SharedSupport/Scripts folder.

Bug Fixes

- Use the correct font on OS X 10.9 (Mavericks).
- Fixed viewers showing garbage renderings on OS X 10.9 (Mavericks).
- Prevent MeVisLab from going into App Nap in Worker Mode on OS X 10.9 (Mavericks).
- Fixed focus issue with some of the IDE dialogs.
- Fixed keyboard shortcuts issue with module panels in 'stay in front'-mode.
- File dialogs again respect the given directory.
- Fixed ADK Application License Manager license detection issue.
- Fixed embedding of license file in DMG for OS X 10.9 (Mavericks).

Notes

- To remove the preference for the image processing cache size and to let MeVisLab automatically compute the value, type the following command in the Terminal:

```
defaults delete de.mevis.MeVisLab GlobalSettings.MLCacheSizeInMB
```

- To remove the preference for the amount of texture memory on the graphics card and to let MeVisLab automatically compute the value, type the following command in the Terminal:

```
defaults delete de.mevis.MeVisLab GlobalSettings.TextureMemoryInMB
```

Modules and Libraries Provided by Fraunhofer MEVIS

ITK and VTK

- ITK: upgrade of the ITK ThirdParty library to version 4.6.0 from December 27th, 2013.
- ITK: upgrade of wrapped ITK modules to new 4.6.0 version.
- ITK and VTK: the directory structure of generated wrapper projects has been revised. Manually and automatically created projects are separated, all generated help files are in mhelp format now.
- `ITKVTKGenerator` creates mhelp files instead of HTML files from now on. Generated wrapper codes use more package compliant directory structure and have less compile warnings, insecurities, and bugs.
- `itkImageFileReader` supports automatic ML data type selection for loaded files now.

Fixes/Enhancements (FME Modules)

- `SoCSOManualCorrectionEditor`: ported the SoCSOManualCorrectionProcessor to work with the `SoView2DCSOExtensibleEditor`.
- `TensorToEigensystem`: ported module to `MLAlgorithmModule` interface. Old updateMode and applyMode EnumFields have been replaced by a single BoolField "shouldAutoUpdate", manual adaption of existing networks required.

- `InterpolateXMarkerList`: added support for markers on different timepoints.
- `FFT`:
  - Fixed incorrect handling of reordering of frequencies when calculating the inverse FFT for images with odd number of elements in filtered dimension(s).
  - Introduced namespace FFTTools for tool-functions, renamed 'reorderFrequencies' to 'shiftFrequencies' and added additional tool functions.
- `CLIImporter`:
  - generated macro modules will re-use their input images on disk if possible.
  - added more auto-detected default paths (Slicer on Windows, downloaded Slicer extensions).
- New module `CSOShapeBasedInterpolation` to linearly interpolate parallel CSOs over slices.
- `VTKPolyDataToWEM`: can now convert color information and warns if the input is not of type VTKTriangle.
- `TubularTracking`: added an option to specify the start radius.
- New DICOM related modules, most of them are available as source codes in the `FMEwork/ReleaseMeVis` package:
  - `ApplyDicomPixelModifiers`: modifies/filters ML images according to (frame specific) DICOM tags.
  - `DicomModifyMultiFileVolumeExport`: exports DICOM files and a file list after adapting images and DICOM tags.
  - `ModifyDicomTreeAndImage`: adding, changing, switching, and modifying DICOM trees appended to images.
  - `DicomTreeInfo`: shows general DICOM tree information and permits value extraction from tag dumps with (regular) expressions.
  - `DicomTreeCompare`: for comparing and testing DICOM trees.
  - Experimental modules `DicomFIDSave` and `DicomTreeValidate` added.
  - `ModifyDicomTreeAndImage` and `DicomModifyMultiFileVolumeExport`: supports DICOM tree, tag, and image (property) modifications.
  - Plugin modules to modify DICOM tags and/or DICOM trees:
    - `DicomModifyTagsPlugin`
    - `DicomModifyImageTagsPlugin`
  - New modules displaying contents of radiotherapy objects: `RTDoseInfo`, `RTImageInfo`, `RTIonPlanInfo`, `RTIonRecordInfo`, `RTPlanInfo`, `RTRecordInfo`, and `RTStructInfo`.
- New features and changes in modules and wrappers:
  - Many modules (including `DirectDicomImport`):
    - Tag dump tab has new optional features:

    - Tag annotations.
    - Size control of hex dumps of binary tags.
    - Regular expression filter.
- `DirectDicomImport`
    - Added many more tags to default DPL configurations for more reliable sorting and partitioning.
    - Added experimental postprocessor for Siemens overlapping series' SetNGo-Files.
    - Safety-check added to avoid out-of memory situations with very large private tags during import.
    - MinMax-calculation bug on SEG-multi frame bit-image DICOMs fixed.
- `FileListFilterPlugin`
    - Can now also modify Processing Settings of `DirectDicomImport` for filtered DICOM files.
    - Supports private creator for private tags in Tag Value Filter.
- DICOM related modules in FME and Python wrappers: many modules provide Base connector(s) for DICOM trees to enable the handling of non-image DICOM information.

Source Codes, Wrappers, and Documentation

- Ported many module documentations from HTML to mhelp.
- Added overview help page 'DICOM in MeVisLab' about all DICOM related functionality in MeVisLab.
- General improvement of test coverage, documentation, and robustness, especially in DICOM functionality against bad DICOM configurations.
- Source code revision and new tool classes and class hierarchies:
    - Some Python wrappers use size_t or 64 bit typed arguments instead of too small 32 bit types.
    - Functionality from `DICOMAccessories` has been moved/refactored into other tool libraries.
    - A lot new DICOM functionality added to projects MLDICOMCachedIO, MLDICOMTags, ReleaseTools, FileListTools, MultiFileVolume, DicomAnalysis, DicomModify, DirectDicomImport, MultiFileVoluemListOutputs. Mostly available as public source codes.
    - `ReleaseTools` project provides a new base class `FieldAddOnBase` with a number of derived classes. For details see corresponding class documentations.
        - DICOMTagDumpFieldAddOn with derived classes TreeAndVolumeInputFieldAddOn, and StringLineFilterFieldAddOn.
        - DicomModifyFieldAddOnBase and its derived classes in MLDicomModify with derived classes DicomModifyOrdinaryFieldAddOn, DicomModifyOtherTagOperationsFieldAddOn, DicomModifyPrivateAddFieldAddOn, DicomModifyPrivateRemoveFieldAddOn, and DicomModifySequenceFieldAddOn.
        - DicomTagInterfaceBase and twelve derived classes in the project MLDicomModify dedicated to represent macros, IODs, and modules according to DICOM standard definitions, for example `DicomPatientModuleTagInterface`.

Version 2.5.2 Stable Release (2014-04)

Note

This is a special release that was not widely distributed because the 2.6 release was imminent.

New Features

- Added isDoubleClick and isAutoRepeat properties to Open Inventor mouse/key events.

Fixes

- Provide header files for the Eigen library.
- Improved event logging in SoActionLog.
- Create context menu events in RemotePanelRendering if createContextMenuEvents flag is set.
- Fixed crash in Python debugger if a Python file with breakpoints in it was removed.

Version 2.5.1 Stable Release (2013-11)

Platform-independent Modifications

- Fixed crash in MATE if back/forward is clicked and no document is present.
- Fixed a crash on module reload under certain circumstances.
- .mlinstall/.mli files: Exclude statements should respect the inputPath.
- .mlinstall/.mli files: Allow to redirect application files to different sub folder with variable OUTPUT_APPLICATION_ROOT.
- Avoid message boxes when MeVisLab is running as worker process.
- Fixed crash on remote module disconnect if MLAB.processInventorQueue is called then.
- Fixed strange behavior of Shift/Alt-Modifier keys in Inventor viewers.
- Fixed a code generation bug in the Inventor module wizard.
- Added .libs and .h for tools and data structures of the DTI library.
- Added dummy cursor shape IDs that can be overriden by setting the OverrideCursorDirectory variable in the preferences file, and adding PNG files to this directory. The filenames must be a lower camel case version of the enum value; a cursor hotspot position can be specified by adding !<x>,<y> to the filename before the .png suffix.
- Scripting:
  - Added MLAB.loadDicomTree method for loading a DICOM tree from file.
  - Added a scripting wrapper for the output of the RemoteRendering and RemotePanelRendering modules. This allows to add a virtual rendering slave.
  - Fixed list return of MLABTreeWrapper::getAll.

  - CheckBox control emitted toggledCommand initially when "checked" attribute is set.
  - MLABGraphicsFrame.setContentsMargins did not work as expected.
  - MLABGraphicsScene.addMDL did not initialize window state completely, which could lead to strange problems.
  - MLABWindowControl.close() removed all listeners even if the window was not actually closed.
  - Fixed a crash in PythonQt when calling virtual methods of wrapped objects during cleanup of those objects.
  - Fixed inplace operator assignment in PythonQt.
  - Added documentation for RemoteCall interface.
- New modules:
  - InventorSceneChangeObserver: Helps in determining the cause of Open Inventor scene redraws.
  - RemoteRenderingEventRecorder: Remote rendering slave that can record inputs events in a JSON file. This can be used in tests; for an example, have a look at the test for the SoView2DExtensibleCSOEditor.
- New modules by Fraunhofer MEVIS:
  - MLToRTStructureSetConverter: Converts a CSOList and a corresponsing image to a DICOM RT StructureSet object.
  - CLIImporter: Imports any number of CTK/Slicer CLI modules and makes them available as macro modules in MeVisLab.
- Enhanced modules:
  - RemoteRendering: Added facility to intercept double-clicks.
  - SoActionLog: Generates a more verbose log, optionally in JSON format. The JSON log string can be copied and cleared.
  - SoShowFPS: Can also show total number of redraws.
  - SoCameraInteraction: Added fields to select cursor shapes.
  - SoView2DSlicer: Can invert slicing direction by specifying negative sensitivity.
- Fixed modules:
  - DtfSkeletonization: Fixed issues with large graphs.
  - RegionGrowing: Fixed a problem with (incremental) updates and floating point images.
  - RegistrationManual: Was not initialized with identity matrix.
  - HistogramEqualization was overhauled.
  - SoView2DCSOExtensibleEditor: Removed warning about empty creator ID string, which is an allowed case.
  - SoCSOLabelRenderer: Store initial default font size in CSO.

- SoGenericCommandAction: Could not be disabled.
- RemoteRendering: Skipping mouse move events if more than one has accumulated in the event queue. This should fix lagging mouse interaction.
- RemotePanelRendering: Did not update cursors at the correct moments.
- ImageROIIsFileCache: Use $TMP directory for cache files.

Windows Edition

- Fixed out-of-bounds assertion in eatDicom (DicomImport) tool in Debug mode.

Linux Edition

- Fixed a problem with -ignoreprefs command line option.
- Fixed a boost compilation error on GCC 4.7.

Mac OS X Edition

- Fixed a focus issue when module panel is first shown.
- Don't execute main window shortcuts if a module panel has the focus.
- Updated documentation.
- Fixed a license check error when macro modules were moved to a separate process.
- Make windows not restorable on MacOS 10.7 and later.
- Use the correct system font on MacOS 10.9 (Mavericks).

Version 2.5 Release (2013-06)

Fixes / Enhancements (MeVisLab)

- New Features
  - Improved general multi-threading support for ML modules through a new ML host implementation than can take better advantage of the page-based image processing. This is quite new and we recommend users to stick to the "classic host" (see the preferences panel) for serious work for now, but we would like to hear about your experiences with your networks under the new host.
  - MeVisLab can move module execution to another process (see the context menu of ML and macro modules). This feature has existed since version 2.2, but wasn't generally visible. It has matured since then and might be useful in some scenarios. Use this to e.g. do some work in the background. This has some restrictions that are documented in the MeVisLab reference.
  - Improved network profiling.
    - Time spent in Open Inventor node rendering and event handling is recorded now.
    - When not combining call stacks, the "Functions" tab of the profiling view displays the actual call order. The old tracing feature which relied on manually placed macros in own code has been disabled.

    - Avoid saving geometry changes in .mlab files if nothing has been moved. This reduces differences e.g. for version control systems.

- MATE (Integrated Editor)
    - New "Watches" and "Evaluate Expression" views in Python debugger.
        - The "Watches" view manages a list of variables and expressions and always shows their values for the current stack frame while debugging (if applicable).
        - The "Evaluate Expression" view can be used to execute code for the current stack frame while debugging.
    - Can open IDE of connected MeVisLabs from MATE. This can be useful to debug (invisible) worker processes.

- C++ API Changes
    - `mlrange_cast` also works for `float` and `double` types now.
    - `mlTimerCounter.h` uses `MLint64` instead of `double` for time values now.
    - Introduced new function `Module/Host::processMissingPages` which only processes those pages that haven't been processed yet (in contrast to `Module/Host::processAllPages`, which does it for all pages).
    - `SubImage` got a new copy constructor that takes an additional translation offset. This can e.g. be used to translate an image in the `calculateOutputSubImage` method of a module.
    - Introduced new `update{Type}Value` methods on ML fields that only touch the field if the value changes. Previously one had to implement a comparison him/herself to avoid unnecessary field notifications.

- Controls
    - Extended expressions for `dependsOn` and `visibleOn` tags. We allow simple numerical expressions and comparisons for these tags now.
    - Pass reference to `EventFilter` control in command callback of `EventFilter` control.
    - Own controls can implement MDL tags also through `QObject` properties, which are automatically accessible from scripting.
    - Some MDL tags of `Slider` and `IntervalSlider` controls are also available as `QObject` properties.
    - The visible tab of a `TabView` can now be controlled through a field of type integer.
    - Fixed a problem with the `RemoteRendering` control when displaying the same rendering in several tabs.

- Scripting
    - New `updateValue` methods on fields that only touch the field if the value changes. Previously one had to implement a comparison him/herself to avoid unnecessary field notifications.
    - Added new methods `MLAB.setNumberOfImageProcessingThreads` and `MLAB.getNumberOfImageProcessingThreads` to override the default number of threads used for image processing.

- Fixes
  - The multi-document interface of MeVisLab and MATE has been replaced by a new implementation. Subsequently it is not possible anymore to have document sub-windows that can be placed and arranged, only one document/network is visible at a time.
  - Streaming output to `std::cerr` with <<isn't fragmented in the debug console anymore.

Fixes / Enhancements (Modules)

- General
  - The example modules have been updated.
- Open Inventor
  - Fixed a bug with nested uses of OpenGL frame buffer objects.
- Open Inventor Modules
  - New module `SoView2DSlider` that displays a slider control in an Open Inventor scene.
  - Added module `SoBlockNotification` that blocks update notifications in an Open Inventor scene graph.
  - New `SoMetaInformation` framework for passing information about the Open Inventor scene to e.g. a remote client.
  - Remove autoUpdateML field on `SoRenderArea`/`SoExaminerViewer`
  - `SoVascularSystem` can show an interactive preview of the branches that are about to be selected.
- CSO
  - Improved CSO scripting wrappers. Added methods to create primitives.
  - Speed-up for `CSOConvertTo3DMask`.
  - New extensible editor plug-ins `SoCSOLiveWireEditor`, `SoCSOBulgeEditor`, and `SoCSOArrowEditor`.
  - Fixed CSOLiveWireGraph's Laplace feature mode.
- WEM
  - `WEMIsoSurface` can generate surfaces for certain or all time points of an image.
  - `SoWEMRenderer` got an option to render a specific WEM patch from a set.
- ML modules
  - New masking features for `ConstrainedConnectionCost`.
  - Fixes to the `Histogram` module.
  - `ImageHash` calculates hashes independent from page size now and does not ignore the last row of an image.
  - `Mask2XMarkerList` has been renamed to `MaskToMarkers` and supports a value threshold now.

  - `(De)ComposeVector` has been renamed to `(De)ComposeVector4` and `(De)ComposeVector3` was added.

- Modules and Libraries Provided by Fraunhofer MEVIS
  - `DirectDicomImport:`
    - Many internals refactored to projects MLFileListTools and MLMultiFileVolume.
    - Volume list columns are better configurable now.
    - Some single-bit DICOM configurations are readable now.
    - Multiframe NM modality files with negative slice spacing are imported now.
    - New view/field showing source files of a volume.
    - Bug fixes:
      - Volume list view corruptions with special characters in tags.
      - Readabilty of cache files with special characters and spaces in file names.
  - Renamed modules:
    - `DirectDicomImportImportFilterPlugin` is now `FileListFilterPlugin.`
    - `DirectDicomImportOutput` is now `MultiFileVolumeListBaseOutput.`
    - `DirectDicomImportImageOutput` is now `MultiFileVolumeListImageOutput.`
    - `DirectDicomImportWaveformOutput` is now `MultiFileVolumeListWaveformOutput.`
    - `DirectDicomImportIteratorOutput` is now `MultiFileVolumeListIteratorOutput.`
    - `DirectDicomImportFilterVolumeList` is now `MultiFileVolumeListFilterOutputs.`
  - New module `DicomSCSave` added for saving secondary captures.
  - Some new tool classes for saving DICOM IODs in DICOM output project.
  - Port of Fraunhofer modules to new ML interfaces including many smaller C++ code cleanups and warning removals.
  - New tool routines added DICOMAccessories and ReleaseToolsDICOM.
  - `itkImageFileReader:` File meta data shown in tab.
  - Bug fixes in ITKVTKGenerator: wrong include code generation fixed.
  - MLCurveUtils: some bug fixes and performance improvements.
  - `VideoCapture:`
    - Made more efficient (on demand decoding).
    - Uses libv4l OpenCV backend on Linux.
  - `Calculator`

    - Removed "Off" mode.
    - Some bug fixes and performance improvements.
- ThirdParty:
    - OpenCV updated.
    - libv4l added.
- New libraries added to SDK and Public Sources:
    - MLFileListTools - Library to scan, filter (even on DICOM tag level) and manage file lists.
    - MLMultiFileVolume - Manages set(s) of 2D file/frame names as higher dimensional volume(s).
- DiffusionMRI modules:
    - `FiberBundleQuantification` - new module for quantifying tracked fiber bundle.
    - `SetFiberness` - new module for assigning fiberness values to points of tracked fiber bundle.
    - `SetFiberColor` - new module for assigning color values to points of tracked fiber bundle.
    - `CropFibers` - new module for cropping the endings of fiber bundle using a mask image.
    - `SaverFibers` allows for storing fibers in VTK polydata format.
    - `WrapFiberSet` additionally outputs the voxelized fiber bundle (sometimes called "envelope" of fiber bundle).

Windows Edition

- Introduced support for Visual Studio 2012.
- Only update .vcxproj files if the .pro file has changed.
- New visual on-the-fly network selector for selection with Tab key or mouse wheel.

Linux Edition

- ML modules can currently not be compiled under Ubuntu 13.04 because of a name clash between a boost header and a system header. This can be avoided by changing every occurence of `TIME_UTC` in the file Packages/MeVis/ThirdParty/Sources/misc/boost/boost/thread/xtime.hpp in the MeVisLab installation directory to `TIME_UTC_`. This release is tested with Ubuntu 12.04.
- Fixed "Restart with Current Networks" (Ctrl+K) for some distributions.
- Fixed `TestCaseManager`.

Mac OS X Edition

Note

This release requires at least Mac OS X 10.7.2.

- Fixes for MacBook Pros with Retina Display (full support is still work-in-progress).
- Do not disable or warn about certain OpenGL features if an Intel graphics chip is detected (full support for Intel 4000).
- Allow alternative locations for the prefs file (~/Library/Application Support/APPNAME/Settings, ~/Library/Application Support/APPNAME).
- Allow license file to be stored in ~/Library/Application Support/APPNAME as well.
- Added MacDependency application to MeVisLab bundle.
- SoCSO modules are again compiled with full optimization.
- StopWatch module now measures correctly in multi-threaded conditions.
- The OS X video camera driver has been ported from QTKit to AVFoundation and now supports exposure locking and furthermore reads the camera frame size from the OS driver.
- OsiriXBridge-plugin fully supports OsiriX running on Retina Display Macs.

Version 2.4 Release (2013-02)

Fixes / Enhancements (MeVisLab)

- New Features
  - The default handling of fatal errors is to abort in release mode now (just as it is in debug mode).
  - New zoom style zoom-to-mouse-cursor selectable from the preferences panel.
  - The internal network of a macro can be opened with middle click.
  - One can turn off field deprecation warnings in the preferences panel.
  - Network preview in tooltip of workspace tabs.
  - Performance improvements of network rendering.
  - Macro module wizard creates comment tags instead of MDL comments.
  - The ModuleDependencyAnalyzer sorts packages, files, and directories now
  - The ModuleDependencyAnalyzer did not correctly collect local macros
- MATE (Integrated Editor)
  - Searching whole words works as expected now.
  - Fixed timeout behavior when connected to MeVisLab.
- C++ API Changes
  - There is a new persistence interface for Base objects. See documentation in `mlAbstractPersistenceStream.h`
  - All MLLinearAlgebra matrix classes are derived from a common base class now.

  - Some compiler warnings have been turned into errors in our build system.
- Controls
  - ItemModelView can display vectors and matrices.
  - ItemModelView supports comboboxes for editing.
  - The LineEdit control supports inline controls now (for tool buttons like e.g. a 'Reload' button).
  - New DateTime control.
  - Added support for 'fieldDragging' tag to CheckBox and TextView, to allow easy field connections when using these controls.
  - A 'title' tag can be specified for fields.
- Scripting
  - Allow to set Python objects on MLBase type fields.
  - Python Booleans are really passed as Booleans to Qt slots now.
- Fixes
  - Remember (some) command line arguments when restarting MeVisLab.
  - Always update Inventor node name when instanceName is changed.
- Library Updates
  - boost updated to 1.48
  - Python updated to 2.7.3
  - NumPy updated to 1.6.2
  - assimp updated to 3.0 (affects `WEMSceneLoader` and `SoSceneLoader`)

Fixes / Enhancements (Modules)

- Open Inventor
  - `SoSFDouble` has been moved to the Open Inventor package, field connections to other Inventor field types work now.
- Open Inventor Modules
  - The new module `SoCreateCubeMapSampler` renders an Open Inventor scene to a cube map.
  - `SoView2DMarkerEditor` got a new overflow mode 'Remove New'.
  - `SoView2DAnnotation` can show an orientation cube and a center bottom text.
  - `SoView2DAnnotation` can specify user length/precision for individual tags.
  - `SoView2DPlane` was improved.
  - `SoView2DSliceShift` has been renamed to `SoView2DSlicePan`.

  - SoView2D will perform automatic slice grabbing now, i.e. in mosaic mode mouse interactions will be restricted to the starting slice until all mouse buttons are released again.
  - So3DXMarker got a marker threshold for faster rendering.
  - The SoSampler modules have been refactored.
  - SoView2D and many extensions support an alternative event handling mode called Managed Interaction that avoids clashes of button combinations and allows discovery and re-configuration of key and button combinations in an Open Inventor scene. Set the field useManagedInteraction to use this mode.
- GVR
  - Added new module SoGVRForceRenderingInBox to force rendering in a defined box.
- CSO
  - The CSO classes emit change events besides the old notification system. This is also accessible from scripting.
  - Added user-data facility to CSOs, based on the new MLVariant library. This is accessible from scripting.
  - New persistence format (version 5) for CSOs, supporting user data.
  - Added new extensible CSO editor framework.
  - Fixed seed point deletion on finishing CSOs in CSOLiveWireProcessor.
- ML modules
  - DicomTagModify got an alternative string interface to modify more than six tags at once.
  - SubImage has a center/width mode now.
  - SubImage: Fixed clamping for negative coordinates.
  - IntervalThreshold: optimized output range prediction and much improved GUI panel.
  - Fixed flipping in SwapFlipDimensions. Page size handling is now as one would expect.
  - Assorted fixes and changes on:
    - EuclideanDTF
    - RegionGrowing
- Macro modules
  - Changed LUT handling in View2D/View3D.
  - Fixed the DicomQuery module.
- Modules and Libraries Provided by Fraunhofer MEVIS
  - Many FME projects have been ported to the new ML interfaces.
  - For many modules, the build system no longer allows warnings.

- In the ReleaseTools library, new sublibraries, namely ReleaseToolsIO and ReleaseToolsMisc, have been added.
- DICOM support has been extended by refactoring the PrivateDicomTag decoders and adding new sublibraries (DICOMCachedIOTools) and functions to the DICOMCachedIO and DICOMAccessories libraries.
- Private tag decoding has been improved and can now decode many private sequence tags, special Toshiba tags and handle unknown private tags.
- `DirectDicomImport`:Several bugs have been fixed, including handling of negative slice spacings and the import of temporal series.
- `DirectDicomImport`:DICOM REG deformation fields can be imported.
- `DicomSRSave`: New module to export DICOM-compliant structured report files.
- `DicomREGSave`: New module to export DICOM-compliant registration deformation fields.
- `ColorFromName`: New module to determine and assign colors and transparencies from strings via regular expressions.
- Audio-Waveform playing is no longer support on Linux.
- `Arithmetic` and `Calculator`: Absolute minimum and maximum functions for vector types were added.
- `Arithmetic`: A bug in the propagation of the voxel-to-world matrix has been fixed.
- `Calculator`: The handling of domain errors can now be configured by the user.

Windows Edition

Note

This will be the last major version that will be released for **Visual Studio 2005** and **Visual Studio 2008**.

We plan to release the next major version of MeVisLab only for **Visual Studio 2010** (32 and 64 bit) and **Visual Studio 2012** (64 bit only). This should be only of interest for you if you develop your own ML or Open Inventor modules.

- Resolve deadlock with multi-threaded cout output under Windows.
- Improve screenshot functionality for panels containing OpenGL widgets.
- Simple gesture support for network area: One can close networks with a mouse gesture.

Linux Edition

- We have a new, nicer installer.
- Debug symbols are put into separate files when compiling.
- Known issues:

- Program arguments containing spaces cannot be passed correctly to any of the executable links in the bin-directory, because they are not escaped.

Mac OS X Edition

Note

This release requires at least Mac OS X 10.7.2.

- The panels of an inactive MeVisLab will no longer overlay the active application.
- The WorkerService is now a UserAgent instead of a SystemService and allows the use of the WorkerService on a per user base.
- Replaced all deprecated Apple Type System function calls with Core Text function calls in MLOSXSupport & Open Inventor.
- ADK
  - The installation of the deprecated PackageMaker application for the generation of guided installers (.pkg) is no longer required, also the build speed has been improved considerably
  - Welcome, Readme, License, Conclusion as well as Background image files may be supplied in various formats for the generation of guided installers. Use the .mlinstall file variables: INSTALLER_MACX_WELCOME, INSTALLER_MACX_README, INSTALLER_MACX_LICENSE, INSTALLER_MACX_CONCLUSION, INSTALLER_MACX_IMAGE.
  - The STANDALONE_APPLICATION_ARGS variable in .mlinstall files is now supported for standalone OSX applications.
  - The generation of application icons can now be done during the installer build process from .iconset, .pdf, or other images files if a native OSX icon cannot be supplied, if an image file is provided it should be at least of size 1024x1024.
- OsiriXBridge: An issue with non-ascii characters in path names of DICOM files in OsiriX database has been fixed.

Version 2.3.1 Stable Release (2012-09)

Platform-independent Modifications

- Fixed QMenu placement regression in Qt 4.8 when a submenu doesn't fit to the right of a parent menu, but fits even less to the left.
- Use correct size for tooltips when CSS style sheet is used and the font size is changed.
- Fixed showing the field help for modules with an instance name that differs from its type.
- MeVisLab will only write changed settings values to the registry, so that instances running in parallel will not overwrite each others settings completely.
- Load MATE user scripts also from user packages.
- Added the MLABFieldExpressionEvaluator class to the public API (for creating custom controls).

- The Field control ignored the stretchX/stretchY attributes.
- PythonQt did not always use the correct conversion to boolean when calling slots on QObjects.
- Fixed modules:
  - Arithmetic2: Fixed multiplication of vector image with scalar image to do the obvious operation.
  - EuclideanDTF: Various bugs have been fixed.
  - WEMMerge: Fixed a crash.
  - CSOFreehandProcessor: Don't remove the last seed point placed when in preview mode.
  - SoFramebufferSampler2D/3D: Added field clearAlpha.
  - FiberSetFilter3D: Set correct allowed type for output object.
  - IntervalMap: Optimized the calculation of the output image's min/max values.
  - DicomQuery: Did not work.

Windows Edition

- MeVisLab and MATE did not remember the correct monitor when maximized.
- Support NVidia Optimus technology, switching automatically to NVidia card when starting MeVisLab (with driver version 302 and above).

Linux Edition

- Execute MeVisLab as console application if DISPLAY environment variable is not set.

Mac OS X Edition

- File dialogs did not work correctly if the parent widget had the Qt::WindowStaysOnTopHint set.
- Support for OSX 10.8 (aka Mountain Lion).
- Could not enable Python Debugging in MATE.
- Fixed endless loop when executing formal TestCenter tests.
- TestCenter failed to generate chart when run in IDE.
- Fixed issue with module database caches not being rebuild for updated MeVisLab applications.
- Updated OsiriXBridge.

Version 2.3 Stable Release (2012-07)

Platform-independent Modifications

- Documentation added for modules from some Frauhofer MeVis packages.
- Fix annoying pause when opening welcome screen or test center results.

- Fix that the test center result page could show old screenshots if MeVisLab was not restarted between tests.
- Fixed modules:
  - Fixed memory leak in MLImageFormatFileCache.
  - Improved documentation on ItemModel wrappers and ItemModelView. The necessary headers for using MLItemModel are included in the SDK now.
  - Allow to specify color values as string for "colorAttribute" in ItemModelView.
  - The ConstrainedConnectionCost module doesn't try to set the image's maximum intensity value anymore.
  - SoRenderArea and SoExaminerViewer honor the grabKeyFocus flag now.
  - The Histogram module can show the standard deviation even for normalized histograms. Loading of old histogram files has been fixed.
  - Saving of curve data with multiple y curves has been fixed.
  - Module CurvatureEstimation: Fixed the erroneous determination of eigenvalues, which led to wrong results. fixed.

Linux Edition

- Fixed matplotlib integration.
- Including MeVisLabCommands.gdb in installers.

Mac OS X Edition

- Fixed matplotlib integration.
- Update proxy icon for Save As operations.
- Fix drag&drop between module panels.
- Fix file dialogs opened from module panels.
- Changed text display in editor to use default font.

Version 2.3 Release Candidate

Fixes / Enhancements (MeVisLab)

- New Features
  - Own commands can now be defined by Python scripting in the IDE, either in the main menu or in the tool bar.
  - Added auto-show feature for hidden connectors on modules in the network view.
  - Added multi-selection and re-arranging by drag&drop to **Snippets List**.
  - Added option to show IDE of worker processes to the context menu of remote modules.

- Added context menu to the tab bars in MeVisLab and MATE.
- Module tests can be excluded from automatic test runs by using the module tag `testsToOmit`.
- The module inspector now also shows indirectly related modules in the **Related** tab, sorted by relevance.
- Added the printing of a warning to the console when the licence is about to expire.
- Added two-colored halos for modules that are at the same time input and output for a selected module.
- Modules that lie over a group to which they do not belong are rendered differently now.
- New local macros will not silently overwrite existing ones anymore.
- When promoting local macros to global macros, existing fields can be imported and edited.
- New rendering mode "Comic" for MeVisLab networks in the IDE.
- New network autolayout supporting loops and groups.

- New MATE Features
  - Own commands can now be defined by Python scripting in the IDE, either in the main menu or in the tool bar.
  - Added cursor position history function with back / forward buttons to the toolbar.
  - MATE now contains a GUI Editor.
  - Added spell-checking (e.g. for the help editor).
  - Added new functions to move up/down the help outline.
  - MATE should block less often now if MeVisLab is busy.
  - Improved Python auto completion.
  - MATE does not connect to MeVisLabs of other users and does not use background or application MeVisLabs for auto-completion.
  - Speed improvements (needed for Qt 4.8).
  - Added new function "Go to Line Number".
  - Added session handling.
  - Added XML, HTML, and CSS syntax highlighting.
  - Implicit module creation for auto-completion can now be disabled.

- C++ API Changes
  - The ML does not use 'carrier types' anymore. If your module should support extended data types like vectors, etc., use them directly. You should probably switch to the new typed output handlers if you didn't already, since it is easier to support the desired extended types with them. For more information, see the documentation in the ML reference..

- You get a compile time assert if you use the C abs function on floats - we did this because this usually happens in error.
- Added new class `ml::EventSource` derived from `ml::Base` that implements a simple notification mechanism.
- Added `ml::AbstractItemModel` class for displaying (hierarchical) item lists from ML modules (including a control and scripting wrappers).
- The shared library MathUtils has been dismantled.
- The new ML function `MLGetPromotedTypeWithRange` can be used to get the 'best' output type for a module combining several input images of different type.
- Added new include file `mlRangeCasts.h` for integer casts with checks for valid ranges.

- Scripting
  - `MLAB.createMLBase` method allows to create reference counted base objects from Python (e.g. CSOList, WEM, ItemModel).
  - `MLAB.loadLibrary` method offers dynamic loading of shared libraries via scripting.
  - MDL controls can now be implemented using Python/PythonQt (see MeVisLab/Examples/Modules/Controls/PythonExampleControls).
  - Base object wrappers can now throw `std::exceptions` to cause a Python exception.
  - The `ModuleDependencyAnalyzer` is accessible from scripting.
  - sqlite3 Python module is available for Mac and Linux now.
  - `IconView` and `ListView` control got a `pressedCommand`, which can be used for starting drag operations.
  - Spell-checking is available from scripting.
  - The `Field` control can have a "More" button to show a bigger input dialog.
  - Added a new `GraphicsView` MDL control, which is based on the QGraphicsView Qt widget.
    - It offers:
      - Inventor render areas,
      - MDL panels,
      - Line, Pixmap, Polygon, Rect, Ellipse items,
      - WebKit and WebView, and
      - HotAreas with transitions.
    - See `TestGraphicsView` and `TestGraphicsViewHotArea` for example usages.
    - A presentation which presents the GraphicsView is included in MeVisLab, just run the `GraphicsViewPresentation` module.

    - The presentation itself is implemented using GraphicsView, have a look at the module's script files for detail.
  - Connecting to Qt signals in Python now has an alternative syntax.
  - The `ListView` control now allows to edit colors that have been set with setValue.
  - Allow to pass Python callables (functions) to e.g. ctx.callLater.
  - Added scripting access to type-handling ML functions through `MLAB.ML`.
- Fixes
  - Creating local macros from a selection retains module groups.
  - Fixed some issues with automatic screenshots in the module help.
  - Fixed a common crash related to diagnosis console output.
  - Fixed ML allocating a large buffer for big image extents.
  - Fixed 32-bit limitation and corresponding warnings of BitImage.
  - Fixed failing writes of memory chunks larger 2G in mlFileSystem.
- Library Updates
  - Added libjpg-turbo for fast JPEG compression.
  - Added Open Asset Import Library (assimp).
  - Added Python matplotlib (see `MatplotlibMDLExample` module).
  - boost serialization library can be used in own projects.
  - ITK was updated to version 4.1.0 (snapshot 2012-01-31) with some Get-functionality fixes and warning suppressions.
  - Qt has been updated to version 4.8 (with some back-ported fixes from 4.8.1).
  - VTK was updated to version 5.9.0 (snapshot 2012-04-28 with git tag "v5.10.0 rc 1") with some Get-functionality fixes and warning suppressions.
- Other
  - Dicom import does not do automatic adjustment of different rescale/slope values anymore.
  - Module panels and module help fiels have been updated.

Fixes / Enhancements (Modules)

- Open Inventor
  - SGI Open Inventor (OIV) was renovated:
    - OIV reference is now generated by doxygen.
    - GLEW was integrated into OIV to allow usage of OpenGL extensions inside of OIV.

    - Vertex Array and VBO rendering have been added to speed up rendering performance.
    - Added new `SoIndexedTriangleSet` for performance reasons.
    - Inventor fields can now be changed without updating the scene graph/causing a redraw.
    - For details, have a look at the Open Inventor release notes.
- Open Inventor Modules
    - New modules:
        - `SoMLVolumeBox` renders the bounding box of an ML volume in world coordinates.
        - `SoScreenSpaceAmbientOcclusion` offers screen space ambient occlusion for arbitrary geometry renderings.
        - `SoFixedFunctionShader` implements the fixed function OpenGL/OIV functionality as dynamically generated GLSL shader and offers per-pixel lighting and simple shader extension mechanisms.
        - `SoImageSampler` offers direct loading of images to textures, including compressed texture formats and cube maps.
        - `SoSceneLoader` provides loading meshes using the Open Asset Import Library (assimp), allowing to import almost all common 3D formats into Open Inventor.
        - `SoRenderSurfaceIntersection` offers interactive rendering of the intersection of 3D Open Inventor / WEM surfaces on 2D slices.
        - `SoShadowMapping` offers cascaded shadow mapping and soft-shadows (PCSS).
        - `SoShowDepthBuffer` displays the depth buffer for debugging purposes.
        - Replaced Diagram2D and ExtDiagram by `SoDiagram2D`.
    - `So3DMarkerEditor` will update externally changed markers automatically now.
    - Fixed a display problem between `SoView2D` and `SoText2`.
    - Support for vertex attributes in the SoShader framework.
    - State changes in the `SoMouseGrabber` and `SoKeyGrabber` module should not cause a scene redraw anymore.
    - Added support for sample buffers (graphics card native anti-aliasing) to SoRenderArea/ SoExaminerViewer.
    - Added anti-aliasing support (MSAA) also to various off-screen rendering modules.
    - Disabled Inventor viewers will not react to mouse clicks now.
    - Fixed `SoView2DPosition` moving the annotations in "Ellipse" mode.
    - `SoImageSamplers` can now load image formats supported by the ML.
    - Added support in `SoView2DMarkerEditor` for using line width from the style palette when using shapes.

  - Improved depth visualization of vectors in `SoView2DMarkerEditor`.
  - `SoView2DAnnotation` can now show "numSlices" and "sliceThickness".
- GVR
  - New modules:
    - `GVRDeepShadowMapping` offers deep shadow mapping for volume rendering.
    - `GVRGeometryClipping` provides clipping with OpenGL surfaces and implements consistent shading.
    - `GVRAmbientOcclusion` implements LAO (Local Ambient Occlusion) for volume rendering.
    - `GVRPreloadVolume` to preload a total GVR volume into the system file cache.
    - `GVRGradientVolumeSave` to calculate and save a gradient volume for an existing GVR volume.
  - `SoGVRDepthPeel` now works with GVR ray caster.
  - Fixed a LUT bug in the GVR with tagged volumes.
- CSO
  - CSOList is now derived from `ml::RefCountedBase`.
  - CSOs and CSOGroups can now be created with automatically created unique labels.
  - `CSOSliceInterpolator` and `CSOConvertTo3DMask` can take inner CSOs into account (and thus produce holes).
  - `CSOSliceInterpolator` got an option to remove the original CSOs to prevent double CSOs in merged copies.
  - `SoView2DCSOEditor` can do hole correction for CSOs of the same group.
  - Added grouping feature to the `CSOUndoRedoManager`.
  - New module `CSOTransformationProcessor` for interactive rotating, translating, and scaling of CSOs.
  - Updated selection and busy mechanism in CSO processors.
  - CSOs intersecting the viewing plane can now be represented by the intersection line.
- WEM
  - WEM visual attributes have been removed, since they where not used.
  - Allow to set alpha value of WEM nodes.
  - `SoWEMRenderer` is now extensible and a lot of debug functionality has been moved to extensions.
  - Added support for mapping primitive value lists (PVLs) as vertex attributes in the `SoWEMRenderer`.
  - `SoWEMRenderer` does not perform picking anymore while the mouse moves.
  - Added new module `WEMThreshold` to cut away parts of the surface below a certain PVL threshold.

- added new module `WEMSceneLoader` that loads meshes using the Open Asset Import Library (assimp), allowing to import almost all common 3D formats as WEMs.
- Added support for saving and loading WEM texture coordinates and normals in the .obj format.
- Added new module `MarkerListToWEMPlane` that approximates a set of markers by a smoothed plane.

- Modules and Libraries Provided by Fraunhofer MEVIS
  - Matplotlib was integrated into MeVisLab in cooperation with MMS.
    - Qt4 backend was ported to PythonQt to work in MeVisLab.
    - `MatplotlibCanvas` MDL control allows easy integration of plots into GUIs.
    - `MatplotlibMDLExample` example module demonstrates the features of the integration.
  - Added the registration module `MERIT`.

    The MERIT (MeVis Image Registration Toolkit) module is a software framework for image registration. It is particularly aimed at the rapid prototyping of registration methods often required in everyday work with MeVisLab. Among its core features are:
    - 2-D/3-D affine-linear image transformations (translation, rigid, similarity, rigid+scale, affine) with customizable component orders.
    - Newton-type optimization (and approximation thereof) with line-searching (linear or Armijo's rule).
    - Multi-resolution image pyramids with custom downsampling, stop levels, and number of levels.
    - Inherent pre-registration initialization schemes (e.g., initial matching of centers of gravity).
    - Five common image similarity measures (SSD, NCC, NMI, LCC, NGF).
    - Nearest neighbor, cubic, Lanczos, and linear image interpolation methods.
    - Multi-threading support using OpenMP processor threads.
    - Robust convergence criteria, e.g. Gill-Murray-Wright, for a unified convergence behavior of all algorithms.
    - Entirely graphical usage through a consistent user interface.
    - Error curve output, mask image support, plugin methods, and much more.
  - Added new module `CSOSplitSelfIntersection` that splits self-intersecting CSOs into not self-intersecting parts.
  - Added new module `IntervalMap` providing convenient user interface to map intervals of voxel values to user defined or default values.
  - The modules `Arithmetic` and `Calculator` have been revised, and come with several major changes:
    - The float and long double variables have been consolidated to double variables. Legacy mechanisms will keep field connections and expressions working, but usage of the old variable names will produce warnings.

    - Arithmetic will no longer work with images of different extents. Use a SubImage module to adjust sizes of input images.
    - Both modules are now capable of detecting domain errors while calculating and can react according to user settings.
    - Anonymous vectors and vector index operators in Calculator now accept any expression as components and index, respectively.
    - A length function for vector types has been added.
    - Calculator returns scalar results when vectors in the expression evaluate to scalars (for example, length or dot product).
    - Arithmetic can be used as a constant image by providing a type and an extent instead of working on input images.
    - In Arithmetic, expressions can use special variables to access the current voxel position.
- `DirectDicomImport` and related modules:
    - Some fixes in voxel size and world matrix calculation.
    - Improved robustness against non-standard or sloppy DICOM tag settings.
    - Imports JPEG2000 compressed files now.
    - Improved import of multi-frame DICOMs with "Decompose Multi Frame Files" flag.
    - Display of some private CSA Header tags with "Dump private tag values" in "Tags" tab.
    - Result caching can create single cache file for each imported file.
    - More tag flags for columns in Volume list.
    - Configurable tolerances for import of untypical or irregular slice distances.
    - Optional non-recursive (flat) directory import option.
    - `DirectDicomImportImportFilterPlugin`: added invert option for regular expressions.
    - `AccessDirectDicomImportCache`: allows to access the DICOM tree from Python, using the DICOM tree caching mechanisms of the DirectDicomImport.
- C++-tool and module libraries and public sources:
    - `MLDICOMAccessories` available in public sources now.
    - `MLDICOMCachedIO` (also available in public sources).
    - `MLGDCMPrivateDICOMTagDecoders` (also available in public sources).
    - `MLPrivateDICOMTagDecoders` (also available in public sources): allows implementing your own DICOM tags decoders.
    - `MLPrivateTags` available in public sources now.

- `MLReleaseTools` and `MLReleaseToolsStable` (also available in public sources): collection of C ++ tools for string conversions, index list parsers, directory scanning, progress logging, interrupt checking, simple path handling, (sub)image saving, icon generation, file type detection, etc.
- `MLITKManualBinding` available in public sources now.

Note

The package structure of the Fraunhofer MEVIS contributions has changed. The Release package located under FMEwork and FMEstable has been split in Release and ReleaseMeVis, which both still appear in the package groups FMEwork and FMEstable.

- ITK and VTK
  - ITK has been upgraded to version 4.1 from January, 31st, 2012, 12-05pm MET from the master GIT repository from http://www.itk.org. Basically FFT and deconvolution related functionality has been added by wrapping 52 new itk classes. 8 wrapped modules have been removed and 4 renamed. Some missing Get-functions and macros as well as some warnings have been fixed in the ITK-third party for that purpose. For the list of changed modules and further details please read the technical notes referenced in the help file of any modules wrapping itk classes.
  - VTK has been upgraded to version 5.10.0 from April, 28th, 2012, 18:08pm (MET) with tag "v5.10.0 rc 1" from GIT repostory from http://www.vtk.org. 156 new classes have been wrapped and 4 have been removed. Some missing Get-functions and macros as well as some warnings have been fixed in the VTK-third party for that purpose. For the list of changed modules and further details please read the release notes referenced in the help file of any modules wrapping vtk classes.
- Other ML Modules
  - Added new module `DicomFrameTagInfo` and improved module DicomFrameSelect.
  - `DicomTagModify` got an alternative interface for modifying an unlimited number of tags.
  - Added a SUM mode for `MergeRegions`
  - Added an optional input for a vector image in `MaskToMarkers`
  - Added new output inspector for `StylePalette`.
  - Added new module `DynamicStylePalette` with a flexible number of entries.
  - Extended remote rendering to also send whole module panels (but without OpenGL viewers).
  - `RasterFunctions` module can generate a list of functions for an input XMarkerList.
  - `RegionGrowing` and `RegionGrowingMacro` now allow to select the precision of the internal image data representation (previously always 14 bits). This improves threshold accuracy and/or memory efficiency in certain cases.
- Other Macro Modules
  - Added new macro module `HTTPFileDownload`.
  - Fixed the problems that some macro modules had with file paths containing non-ASCII characters.
  - Added new module `CompoundMatrixArithmetic` for calculation of arithmetic expressions for several 4x4 matrices.

Windows Edition

- MeVisLab now uses `user/AppData/Local/MeVis/MeVisLab/*` to store per-user data.
- Imported image data is now stored in `user/Documents/MeVis/ImageData`.
  - For backwards compatibility, `INSTALLDIR/data` is still used if it exists and is non-empty.
- License and prefs file may now be located in `user/Documents/MeVis`, `user/` and `user/Documents` is still supported.
- Improved `MLABWidgetControl::createScreenshot` on Windows.
- A fix for the integrated Python that previously shadowed DebugAsserts in the Microsoft runtime library in the release version of MeVisLab.

Linux Edition

- Preferences and other related files are now located in a common directory on Linux.
- Added Eclipse project file generation.
- Fixed that KDE plugins are no more loaded, which could lead to crashes when opening file dialogs.
- MoviePlayerControl support disabled, because Phonon support is disabled due to dependencies to incompatible system libraries.

Mac OS X Edition

Note

Use Xcode 4.2 on OSX 10.6 and Xcode 4.3 on OS X 10.7 for module development. This release has not yet been tested on OS X 10.8.

- The Option (Alt) key allows you to switch between Release & Debug mode using the Option key on OS X when requesting a 'New Instance' from the context menu in the Dock.
- Module database caches are now stored in the `~/Library/Caches/de.mevis.MeVisLab/ModuleCache/` folder.
- NSLog messages are now sent to the Debug console of MeVisLab.
- Added icon for `.mllut` files.
- Handle Dock preferences format of Mac OS X 10.7.2.
- Added support for OpenMP in Xcode 4.2 (using LLVM-GCC).
- Screenshot gallery will show Quicktime .mov files; Movie creation of the SoExaminerViewer will create a .mov file by default.
- The default application Info.plist file may be overwritten using the `MACX_APP_INFO_PLIST` variable in installer files.
- Multi-document windows in the MeVisLab IDE & MATE do no longer have a titlebar when maximized.
- MeVisLab Applications support the Mac OS X 10.7 fullscreen mode too when the tag `canGoFullscreen` is added to the Window control.

- `MLAB.runCommand` and `MLAB.runCommandStdInOut` use a dedicated thread to run the command. The spinning beach ball will no longer show up but an indeterminated progress indicator.
- MeVisLab IDE & MATE support Mac OS X 10.7 fullscreen mode.
- ToolRunner supports creation of Xcode4 workspaces for a list of .pro files.
- Adapted WorkerService for OS X.
- Fixed bug in `macx::Bundle::getBundleExecutable()` function: [NSBundle mainBundle] executablePath] returns the executable of the current process which is not always the same as the executable referenced by CFBundleExecutable in the bundle's Info.plist file.
- Fixed `SoShaderParameter*` bug on OS X 10.7 (the uniform location was not computed correctly).
- `OsiriXBridge` warns if a communication protocol version mismatch is detected (Plugin and module do not match).
- Removed `-fp` parameter from `OsiriXBridge` (DICOM conversion)

Common

- You should not need to set `QTDIR` anymore if you work with QT

Version 2.2.1 Stable Release (2011-09)

Platform-independent Modifications

- Fix hang when double-clicking in empty MATE window
- Fixed a crash when editing the help of a changed module.
- Improve error message when trying to access controls from the wrong FieldListener
- Fix a crash occuring with DynamicFrame under certain circumstances
- Fixed picking of Inventor nodes.
- ML Module Wizard: Changed code template to correctly use the ML_CALCULATEOUTPUTSUBIMAGE_NUM_INPUTS_N_CPP macro when numInputs > 5
- Fixed crash when exporting DICOM data and no network interface could be detected (workaround for a Windows bug)
- Don't generate module help on MeVisLab start as this can take quite some time. If needed this can be triggered from the "Extras" menu.
- ADK: Several fixes to the ModuleDependencyAnalyzer
- ADK: precompilePythonFiles.py is missing in the installed SDK / ADK
- Add DefaultProjectSetup.pri for Fraunhofer packages, needed for compiling modules from these packages

Standard Modules

- Prevent the tabs from SoTabPlaneDragger from getting infinitely big when viewed from the edge.

- Prevent LocalImage from resolving the default DemoDataPath file name on load.
- HistogramParameters: Corrected x scale of histogram curve, updated help
- SimpleRegionGrowing: Fixed a bug in the computation of the valid outbox' t-extension
- SwapFlipDimensions: Don't change the page size if the image size doesn't change
- ConvexHull: Correct documentation
- CSO modules: Calculate center of gravity correctly.
- FileDirectoy: Don't create paths that contain backslashes
- Fixed a bug in the OutputInspector for CurveData
- SoView2DMarkerEditor: Vectors not always visible in a slice
- So3DXMarker: Immediately update displayed markers if changed externally
- SoView2DMenu: Resetting draw color to white before rendering icons
- SoView2DMenuItem: Added an example network for a 3D tool button
- BaseSwitch: Correctly propagate field notifications
- BitMaskConvert: Avoid crash when second input gets invalid.

Fraunhofer Modules

- OpenCV has been updated to version 2.3.1, Visual Studio 2005 (VC8) is supported again.

Windows Edition

- Some few modules didn't reference the included xml2.dll but libxml2.dll (which was not included)
- Properly include MeVisPython executable in SDK
- Correctly show VC10 project entry in context menu of C++ modules.
- Fix a crash on startup for certain processors.

Linux Edition

- Added sqlite3 Python module
- ADK: Make sure LD_LIBRARY_PATH is set when running applications.
- Retain executable bit of installed scripts.

Mac OS X Edition

- Fixed MeVisLab restoration issues on Mac OS X 10.7
- Editable combobox style has been fixed for Mac OS X 10.7 (QTBUG-20261)
- Screenshot function and QuickLook support of MeVisLab network files have been fixed for Mac OS X 10.7
- MeVisLab-based applications will no longer try to run the MLABInstallSetup macro

- MLAB.runCommand, MLAB.runCommandStdInOut, MLABProcess.waitForExit, and MLABScriptProcess.waitForLaunch did not work correctly, they do now but will block the main thread. The spinning beach ball may be shown. A new, threaded implementation will be available with the next full MeVisLab release.
- MLABProcess.run does now correctly return the success state
- signfiles looks for a license in the same locations as MeVisLab itself
- Improved Xcode4 compatibility of the MeVisLab Project Generator
- An installation of Xcode in a folder other than /Developer is supported
- OsiriXBridge: fixed multiple modules overwriting each others eatDicom settings
- The OsiriX MeVisLab Bridge Guide has been updated

Version 2.2 Stable Release (2011-06)

Platform-independent Modifications

- New Public Sources:
  - There are new FME-modules and projects in the public sources providing efficient implementations of often used standard algorithms, demonstrating some new image processing features of the ML, or allowing for a closer look into the binding of VTK modules to MeVisLab or into the programming of plugins for DirectDicomImport:
    - TensorToEigenSystem (FMEstable project MLTensorToEigensystem)
    - GaussGradient and GaussHessian (FMEwork project MLGaussFilters)
    - ImageCumulativeSum (FMEwork project MLImageCumulativeSum)
    - ImageResample (FMEwork project MLImageResample)
    - QuadraturFilter and QuadraturToTensor (FMEwork project MLQuadratureFilter)
    - TubularTracking (FMEwork project MLTubularTracking)
    - DirectDicomImportIteratorOutput and DirectDicomImportWaveformOutput (FMEwork project MLDirectDicomImportOutputs)
    - FMEwork projects VTKSupport and MLABVTKRenderWidget
  - Additional SDK-Interfaces:
    - Additional DirectDicomImport-headers (FMEwork)
    - MLInternalPageCache (FMEstable)
- Fixed modules:
  - Fixed MLReplaceDeprecateAPI helper module.
  - VoxelizeInventorScene was not correctly restoring the Open Inventor state. Updated documentation.

  - CSOConvertToImage: values of CSOVoxelSet voxels are now filled with image values if the module should use image values.
  - CSOVoxelSets now retain the voxel-to-world-matrix of the source image that is used in CSOVoxelSetToMarkerList to create correct world marker positions.
  - The GVR renderer used GLSL shader code which caused an ATI driver crash on some ATI cards/drivers. The GLSL shaders have been modified so that the standard GVR slicer shaders used in View3D etc. should work again. The new GVR ray cast shaders and some advanced slicer shaders will still have the problem. Since this is an ATI driver bug, we filed a bug to ATI. Hopefully this will be fixed in a future ATI driver version.
  - View3D: Avoid clipping the annotations.
- SDK:
  - Fixed Snippets List feature by using a dummy image if no screenshot could be created (e.g. because of an insufficient OpenGL driver).
  - Added a MEVISLAB_VERSION define, current value is 202.
  - Fixed that DynamicFrame control crashes when its content does not contain a GUI control.
  - Fixed a seldom crash under 64 bit versions of MeVisLab that could occur with connections between Open Inventor fields of different types.
  - Fix that module help is not generated for deprecated modules.
  - Various small documentation updates.
- ADK:
  - Fixed that the ModuleDependencyAnalyzer macro does not find modules with an explicit instance name.
  - The standalone ModuleDependencyAnalyzer tool considers user packages now and can also consider an additional preferences file.
  - Fixes to the StandaloneInstallerWizard.

Windows Edition

- ADK:
  - Added missing zip.exe.

Linux Edition

- SDK:
  - Fixed wrapper scripts when used in directories containing spaces.
  - Fixed location of MeVisLabMovieWrapper.
- ADK:
  - Installer creation requires installjammer version 1.2.14, version 1.2.15 does not work.

Mac OS X Edition

- SDK:
  - Added initial support for Xcode 4:
    - Xcode 4 does no longer crash while trying to open project files created by qmake on behalf of the MeVisLab Project Generator (QTBUG-17247).
    - The MeVisLab Project Generator creates default Xcode 4 schemes for the Xcode projects. A **Debug in MeVisLab** scheme is created for MeVisLab module library projects.
- Documentation:
  - Updated MeVisLab Mac OS X Edition User's Guide

Version 2.2 Release Candidate (2011-05)

Platform-independent Modifications

Fixes/Enhancements (MeVisLab)

- New Features:
  - We moved to a new internal help file format (.mhelp) for modules which can be edited with the integrated MATE text editor. These are converted on-the-fly to HTML. The old HTML files are deprecated.
  - MATE has been made a separate program which communicates with MeVisLab instances (if available) for auto-completion and other needs.
  - MATE now contains the frontend for a new integrated Python debugger which support breakpoints, stopping on exceptions, call stack and variable introspection.
  - All modules from the MeVisLab/Standard package that still used JavaScript have been ported to Python.
  - MeVisLab does now provide its own Python version (called MeVisPython) for command line use. A separate installation of a Python version is not needed anymore.
  - Profiling has been improved and now contains a call graph view.
  - Implemented some performance optimizations for loading *.mlab files (especially when there are many field connections).
  - Added lazy script evaluation which can be enabled with `LazyScripting = true` in MeVisLab prefs file (this defers creating the script context until scripting is really used).
  - Added a "Snippets View": fragments from a network can be stored for future usage in a dock in the MeVisLab IDE.
  - Added an option to set module placement style in the IDE.
  - Added a customizable network view mouse interaction.

- The reloading of selected modules can be triggered by F5/Cmd+Alt+R and has a visual feedback now.
- Image inputs/outputs can be colored according to state in the network view for diagnostic purposes.
- Added a preview and navigation feature for internal networks - press Space when no module is selected.
- Added a context menu entry for modules to check library dependencies (calls DependencyWalker under Windows) for Linux and Mac.
- MeVisLab now contains the Qt libraries and header files that are needed to implement custom MDL GUI controls and to make use of Qt in MeVisLab modules. It is no longer neccessary to install a separate Qt SDK. Make sure that the QTDIR variable is NOT set in your environment, so that MeVisLab can make use of its internal Qt version.
- MeVisLab can now be installed at paths containing spaces even if Qt is used.
- The MeVisResearch package group has been renamed and split. The new names are FMEwork and FMEstable. (MeVis Research is now a Fraunhofer institute.) Users usually should not notice this change.
- Modules can be aligned with optional tool buttons.
- Added an option to convert a module group to a (local) macro.
- Added an attribute 'editable' to fields. This attribute sets the default for the 'edit' or 'editable' of controls representing this field. At the same time it sets the 'persistent' attribute if not explicitly given. Non-editable fields are also not editable in the automatic panel.
- Added a standalone tool for static module dependency analysis (for application deployment with the ADK add-on).
- Possibility to execute single ML or macro modules asynchronously in a separate process (with some limitations). This is an experimental feature which needs to be unlocked via the prefs file.

  Limitations:

  - Open Inventor modules/inputs/outputs do not work (but you can use the RemoteRendering module and the RemoteRendering control to do rendering in a remote module).
  - Image outputs work, but image inputs do not; read images from a file instead.
  - Base inputs and outputs work in principle, but need a Base type handler for the remote communication. Currently there exist only handlers for XMarkerLists and some specialized remote helper types.
  - The panel and scripting of a module will only work in remote modules if it does not violate some encapsulation rules.
  - Remote modules are executed asynchronously, so field updates that previously would happen immediately will happen delayed.

- C++ API Changes:
  - Various changes to the C++ Field/FieldContainer API: ml::Module is derived from ml::FieldContainer now, and some rarely used or ill defined methods have been removed.

- Those variants of the `FieldContainer::add*()` methods that take an external data pointer are renamed to `FieldContainer::addDeprecated*()`. Using these method versions is (and has been for quite some time) strongly discouraged.
- The regular FieldContainer::addInt/Float/...() methods now have the initial field value as second argument to avoid an extra setInt/Float/...Value call.
- Base fields can now be restricted to allowed sub-types of class Base (this is used in MeVisLab to do a type check when connecting Base fields).
- Base fields now support intrusive reference counting via a new RefCountedBase class (but this is not yet used in any modules).

- Scripting:
  - Module groups and note items in a module network are scriptable now.
  - MLABModule has a `hasWindow` call now.
  - Added `getFrameSpecificDicomTag` and `getFrameSpecificDicomTagById` to MLABImageField.
  - Added `storingToUndoHistoryCommand` and `restoredFromUndoHistoryCommand` commands to MDL - modules are now informed when they are moved to/from the undo buffer.
  - `MLABImageField::image().getSliceAsQPixmap()` and `MLABImageField::image().getSliceAsQImage()` allow to get a QPixmap/QImage from RGB and RGBA ML images. These images can then be used on MDL controls (e.g. `MLABButtonControl::setPixmap()`) or directly on the Qt API.

- Fixes:
  - Boost was updated to version 1.43
  - SQlite was updated to version 3.7.0.1
  - Crypto++ was updated to version 5.6.1
  - GoogleTest was updated to version 1.5
  - dcmtk was updated to version 3.6.0
  - libpng was updated to version 1.4.3
  - tiff library was updated to version 3.9.4
  - xml2 library was updated to version 2.7.7
  - zlib was updated to version 1.2.5
  - iconv library was updated to version 1.11.1
  - ITK was updated to version 4.0.0 (snapshot 2011-04-01)
  - VTK was updated to version 5.9.0 (snapshot 2011-04-11)
  - Added the following libraries:
    - Eigen 3.0.0 (linear algebra template library)

    - muParser 1.32 (math parser library)
    - OpenCV 2.2 (computer vision library)
    - vigra 1.7.1 (snapshot 2011-02-02) (computer vision template library)
  - Added the following Python extensions:
    - Mako 0.3.4 (template engine)
    - Sphinx 1.0.4 (documentation generator)
    - Jinja2 2.5.2 (templating language)
    - Pygments 1.3.1 (syntax highlighter)
    - docutils 0.7 (documentation utilities)
    - setuptools 0.6c11

Fixes/Enhancements (Modules and Controls)

- GVR:
  - Added temporal position support for secondary volumes.
  - Fixed the rendering of anisotropic GVR volumes (alpha correction and lighting).
  - The GVR Volume Renderer now offers a full featured GPU ray caster and many new modules.
  - Ray Cast Features:
    - Ray casting with empty space skipping and early ray termination.
    - Integration of opaque GL geometry and GL clipping planes.
    - Jittering (Half-step/Random).
    - First hit ray casting.
    - Support for large volumes using bricking/lookup textures.
  - Shader Pipeline Features:
    - SoGVRShaderFunctions work in both slicer and ray caster.
    - Improved coordinate system naming using postfixes.
    - Various new pipeline steps to make the ray caster extensible.
  - New Ray Cast Extensions:
    - SoGVRRayCastSettings
    - SoGVRFirstHitRayCastSettings
  - New Parameter Modules:
    - SoGVRShaderParameterPosition

    - SoGVRShaderParameterDirection
    - SoGVRShaderParameterPlane
  - New Extensions:
    - SoGVRPointLight
    - SoGVRLitSphereShading
    - SoGVRFirstHitAmbientOcclusion
  - SoGVRShaderDiagnosis was rewritten and now offers verbose introspection of the shader pipeline, the used shaders, shader errors/warnings, available state/uniforms and used uniforms.
  - SoGVRPipelineInspector was removed (it was replaced by functionality in SoGVRShaderDiagnosis).
  - GVRShaderFunctionEditor offers a new interface to edit SoGVRShaderFunctions using completion and auto-detection of used uniforms/state.
- CSO:
  - SoView2DCSOEditor now provides id of the CSO currently under mouse cursor.
  - CSOLiveWireProcessor can use input image directly as weight image.
  - CSOConvertTo3DMask now provides an option to use approximation instead on interpolation which makes the module more robust in some cases. The performance and quality of the algorithm has been improved.
  - Added option to render inner CSOs as holes.
  - More fine-grained control by groups over the display attributes of CSOs.
- WEM:
  - Added diagnostics for WEM patches in SoWEMDiagnosis.
  - Added standard and root mean square deviation to the WEMInfo module.
- Modules Provided by Fraunhofer MEVIS:
  - New module CSOManualCorrectionProcessor: allows for an intuitive editing of any CSO in 2D.
  - New module ImageFileCache: allows for a simple caching of standard datatype images with flexible update handling.
  - New module RunScript: simple utility module that allows for storing and executing a Python script in the network without having an explicit Python source file (or inline code in the .script).
  - New module ImageCumulativeSum: calculates the cumulative sum over an image dimension.
  - NotificationCounter: module for analyzing field notifications, now supports differentiation between notifications on valid and on invalid images.
  - BoundingBoxExt module now correctly detects and handles empty boxes.

- The VideoCapture framework has been extended:
  - Rare crashes while exiting MeVisLab have been fixed.
  - Access to driver-specific functions is provided via base-field access.
  - New SoVideoCaptureSampler2D module that uses a driver-specific function to provide access to the video frame in GLSL code (see SoShader modules).
- New VideoLoad module to load video files, the supported formats depend on the platform-specific backends used by OpenCV.
- New SoSimulateMouse module to simulate mouse interaction via scripting or network.
- New View2DTouch macro that offers additional interaction capabilities using gestures, an image dataset my be explored from the trackpad of modern notebooks or by using an external trackpad like the Apple Magic Trackpad.
  - New modules SoTouchpadDevice, SoView2DTouchControl, SoSwipeAccess.
- New modules: CurveSelect (selects curves from a CurveList), CurveImport (creates CurveData from ascii text).

- Other ML Modules:
  - The ConnectedComponent module has been reworked.
  - The MergeRegions module now allows to deactivate auto-update.
  - Added the ImageClear module, which automatically invalidates its output on input image changes - the user has to "manually" pass the input image through. This can reduce notifications.
  - DirectDicomImport:
    - New plugin mechanism available (see DirectDicomImportImportFilterPlugin) for filtering and classifying DICOM files before importing them.
    - New DirectDicomImportIteratorOutput module for iterating over volumes.
    - Radio-Therapy modules are also available on Linux platform now.
    - The eatDicom importer in DirectDicomImport has been removed, which leads to a number of "missing field" messages when reloading network built with older versions. Please resave those networks.
    - Many fixes, especially in voxel size and matrix calculations.
    - Many new features, such as sorting according to DICOM tags, additional import directories, experimental waveform output.

- Other Open Inventor Modules:
  - New module SoBypass to bypass an Inventor sub-graph.
  - Added mipmap support for 3D samplers.
  - Added a SoCameraViewAll module.

  - Improved external camera handling in the SoExaminerViewer.

- Other Macro Modules:
  - The Counter module gained the ability to count at regular intervals.
  - New modules StringSplit, StringListSplitter and SwapViewer.
  - LocalImage has been changed to return a new dataset by default that has anisotropic voxel size, is sagittal, has DICOM tags and does not have equal image extents.
  - ModifyRegion refactored to support non-CT world/voxel matrices (e.g. from MRI)
- Controls:
  - New MDL controls RadioButtonGroup, PushButtonGroup, ToolButtonGroup that shall replace the ButtonBar control.
  - `MLABListViewControl::setItemFilter` works recursively now.
  - Added a GLSLTextView control with input completion.
  - Updated MLABCheckBoxControl scripting.
  - Added `tabDependsOn` tag on TabViewItem control.
  - The IntervalSlider honors the "direction" tag now.
  - DicomBrowser is now a dynamically loaded control and supports .mlimage files.

Windows Edition

- Implemented a work-around for the Aero screenshot bug.
- Added support for Visual Studio 2010.

Linux Edition

- MeVisLab is now compiled with gcc version 4.4.
- Added support for DICOM RT.
- Library symbols are now exported explicitly under Linux; this should reduce library sizes.
- The mlbash script in <installpath>/bin has been renamed to mlshell to reflect that other shells than the bash can be used. The environment variable SHELL can be set to specify a shell. If it is not set, then the bash is the default.
- The prefix or infix MLAB was added to most third party libraries to avoid the MeVisLab loads same named, incompatible system libraries (e.g. libQtCore.so is now called libQtCoreMLAB.so, libfreetype.so is now libMLABfreetype.so).
- The compiler defines linux and Linux have been removed, so that only the define LINUX remains.
- Added include paths to generated Codeblocks projects to allow more code completion.

- New tool MeVisLabDependencyViewer: requires the unix tools nm, ldd and readelf. It can be started from the context menu of a module in MeVisLab (Debugging -> Show DLL Dependency).
- MeVisLab now tries to use the system fonts DejaVuSerif.ttf and FreeSerif.ttf for OpenGL font rendering if they can be found. Otherwise the not so nice font AntykwaTorunska-Regular.ttf is used as a fallback (this font is shipped with MeVisLab and was always used until now).

Mac OS X Edition

Important

Mac OS X 10.6 (Snow Leopard) or newer required

New Features:

- MeVisLab IDE:
  - Spotlight-based search is supported for the MeVisLab IDE and module help
  - To quick-start MeVisLab hold the function (**fn**) key while clicking the MeVisLab icon or double-clicking a MeVisLab network document
  - A new context menu for the MeVisLab icon in the Dock permits the launch of a new, separate MeVisLab instance
  - The MeVisLab IDE and MATE now provide you with a proxy icon for the current document
  - An AppleScript applet has been added to start MeVisLab in Debug Mode from the Mac OS X Dock. The applet is called `mevislab-start-debug.app` and is located at: `MeVisLab.app/Contents/SharedSupport/Scripts/`
  - The **Recent Files** submenu now limits the maximum number of recent documents according to the user settings. The global setting can be changed in the appearance panel of the system preferences of Mac OS X, but may also be modified on a per-app basis using the `NSRecentDocumentsLimit` key in the applications preferences, e.g. **defaults write de.mevis.MeVisLab NSRecentDocumentsLimit 20**
  - The location of application support files has changed: now the application name (`CFBundleName`) is used to refer to the root directory of settings, screenshot, snippets, etc. In effect, the application support files of the MeVisLab IDE are no longer stored at `~/Library/Application Support/MeVis/MeVisLab/...` but at `~/Library/Application Support/MeVisLab/...`.
  - Inspection of dynamic library dependencies is supported from the IDE using MacDependency 1.1 or later
  - MeVisLab will perform setup functions upon first startup of a new MeVisLab version for each user
  - Additional icons for .mhelp and .mlgraph document files
- ToolRunner
  - The ToolRunner supports the setup of additional environment variables for the tools
- SDK:
  - The `xcode-prepare-debug.applescript` script that sets up Xcode 3 for MeVisLab module debugging has now been transformed into an AppleScript applet. AppleScript applets

can be executed directly without the need to load the ScriptEditor. The `xcode-prepare-debug.applescript` script is now called `xcode-prepare-debug.app`. The script is still located at: `MeVisLab.app/Contents/SharedSupport/Scripts/`

  - The MeVisLab API Documentation is available to Xcode users as 'MeVisLab Toolbox Reference' Doc Set. The Doc Set is located within the MeVisLab bundle at: `MeVisLab.app/Contents/Packages/FMEwork/Release/Documentation/Publish/DocSets/`

  - The clang compiler is now supported as an alternative compiler (version 2.9 or later). Makefiles may be generated by executing a command line like the following: **/Applications/MeVisLab.app/Contents/Support/MeVisLabProjectGenerator.app/Contents/MacOS/MeVisLabProjectGenerator -qopt -spec -qopt /Applications/MeVisLab.app/Contents/Support/MeVisLabProjectGenerator.app/Contents/Resources/mkspecs/macx-clang -no-xcode -no-debug -o Makefile -auto**

  - The `xcode-prepare-unittest.app` AppleScript applet now adds the package environment variables to the unittest executable environment in Xcode 3 facilitating the location of test data

  - The MeVisLab Project Generator evaluates any additional environment variables set via the ToolRunner settings when launched in GUI mode (e.g. when launched via Finder or Xcode)

- ADK:

  - Package installers that are built using MeVisLab will use the new flat package format of Mac OS X 10.5 and later

  - MeVisLab standalone applications do no longer share their settings

- Documentation:

  - Updated MeVisLab Mac OS X Edition User's Guide

  - Updated OsiriX MeVisLab Bridge User's Guide

  - MeVisLab supports the generation of Xcode Doc Sets as part of the documentation tools

- OsiriX MeVisLab Bridge:

  - Series may be sent to multiple targets conveniently via a pop-up selector from the OsiriX toolbar, each OsiriXBridge module represents a target

  - Several series and whole studies can be sent at once if a DirectDicomImport module is used with the OsiriXBridge

  - The OsiriXBridge plugin is controlled exclusively via the toolbar icon in OsiriX

Resolved Issues:

- MeVisLab IDE:

  - The annoying beeps triggered by keyboard events due to a Qt bug (QTBUG-6444) have been fixed by back-porting changes from Qt 4.7

- MDL:

  - A rare crash in DynamicFrame control caused by Qt (QTBUG-4227) has been fixed

  - The additional margin around the viewer of a viewer control has been removed

- SDK:
  - Source code files in the `RELATEDFILES` variable of qmake .pro files are no longer built when targeting Xcode with the MeVisLab Project Generator
- Modules:
  - A bug in the OSX Image IO plugin has been fixed that could provoke a crash when reading images with embedded color space information

Version 2.1.1 Stable Release (2010-09)

Bug Fixes:

- Fixed a Windows resource leak when using OpenGL rendering widgets. (This fix was already included in updated 2.1 installers.)

  Fixed crash in OffscreenRender module when no input available, removed unused autoApply field to avoid wrong expectations.

  Fixed that boolean variables set with the FieldValueTestCaseGenerator are always false

  The mlbash script now sets MLAB_COMPILER_VERSION under Linux

  MeVisLabSDK readme on Linux now includes compiler version

  Added system requirements to Getting Started documentation

  Fixed a problem with scripting function MLABModule.removeTimer, where sometimes an unrelated timer could be removed.

  Missing source files of ITKVTKGenerator files added

  Allow for decompression of (jpeg) image data of single frame DICOM files.

  ml::BackgroundTasksMessage can now be sent to the GUI thread without having a sender()

  Disabled inline completion in MATE's search combo box, which would lead to wrong searches.

  Fixed crash when deleting multiple modules when the UndoManager is disabled.

  Fixed crash in CSOConvertToImage if an attached Bypass is set to No Bypass.

  Fixed crash when deleting ListView control (observed with deeply nested trees).

  clickedCommand and rightButtonClickedCommand of IconView didn't work.

  Fixed a problem with timers with intervals <20ms disappearing under Windows

Version 2.1 Stable Release (2010-06-24)

Bug Fixes:

- The MLCacheSize of MeVisLab 2.0 is no longer overwritten when running MeVisLab 2.1.
- The MeVisLab 2.0 inspector layout is no longer cleared when running MeVisLab 2.1.
- Improvements and bug fixes for the new TestCenter/TestCaseManager.

- Fixed debug mode Qt wrappers for PythonQt on Linux and MacOS.
- Project wizard line edits no longer jump to end of line while typing.
- Various small bug fixes to standard modules (including various WEM fixes).

New Features:

- A new "Search in Documentation" inspector has been added to the MeVisLab IDE and to MATE, which allows to search in the complete MeVisLab documentation (including all Doxygen references, all DocBook books, the Open Inventor reference, Python and NumPy). This feature is also available via the Help->Search in Documentation... menu item and via CTRL-SHIFT-F.
- The MeVisLab documentation was enhanced/corrected in various places.
- A lot of cross-links where added to the MeVisLab documentation. This makes it a lot easier to jump to relevant other documents/doxygen documentation while reading the HTML documentation.
- "Revert To Saved" menu option was added to the MeVisLab IDE.

Version 2.1 Release Candidate (2010-06-04)

New Features

- TestCenter/TestCaseManager
  - The MeVisLab Tester has been replaced by a new testing framework that is based on Python.
  - A TestCase consists of a Python script and a MeVisLab network that should be tested.
  - Each MeVisLab package may contain TestCases, which are managed in the TestCaseManager.
  - A simple ASSERT/EXPECT_EQ/NE/TRUE/FALSE API offers quick test writing.
  - Various other features, e.g., creation of screenshots and attachment of result files to the test report.
- ML BackgroundTasks
  - The new library MLBackgroundTasks adds support for asynchronous image processing to MeVisLab.
  - New example modules: AsyncProcessAllPages, AsyncTypedProcessAllPages, AsyncTileProcessingExample and AdvancedBackgroundTaskExample.
  - ImageSave and GVRVolumeSave now support asynchronous writing ("Save in Background" buttons).
  - The MeVisLab IDE contains a new BackgroundTasks inspector which shows running background tasks.
  - For more details, have a look at the "ML Tool Box Reference", which contains the detailed API documentation.
- MeVisLab Profiling
  - MeVisLab contains a new Profiling Inspector
  - Profiling measures:
    - time spent on ML image processing and field notifications.

    - memory usage of ML modules
    - number of field notifications

- GVR Shader Pipeline
    - The GLSL shaders used by the GVR are now extensible using the SoGVRShaderFunction module. When this module is used, the shaders are built as a pipeline of GLSL functions. The user can add custom functions, and replace existing functions.
    - New modules: SoGVRShaderFunction, SoGVRShaderInclude, SoGVRShaderPipelineInspector

Common Enhancements / Fixes (MeVisLab)

- ML:
    - The ML C++ API has been renovated and large portions have been marked as deprecated. The "ML Reference" contains a list of all deprecated functions/classes and a short introduction on the deprecation. Common abbreviations have been expanded e.g., getImgExt() is now called getImageExtent().
    - A new MeVisLab module called MLReplaceDeprecatedAPI has been added, which can be used to find and replace usage of the deprecated classes/functions.
    - Use DEFINES += ML_WARN_DEPRECATED in your profiles to enable compiler warnings about usage of deprecated APIs.
    - Use DEFINES += ML_DISABLE_DEPRECATED in your profiles to completely disable the deprecated APIs.
    - The deprecated API will stay part of the ML API for at least 2 years.
    - The image processing in an ML Module can now be implemented in an extra handler derived from CalculateOutputImageHandler, which is useful to separate the processing function from the module state, especially when multithreading support is desired.
    - CalculateOutputImageHandler is extended by TypedCalculateOutputImageHandler<> templates to support templated handlers.
    - processAllPages now supports a ProcessAllPages handler, this avoids the need to reuse the calculateOutputSubImage method of the ml::Module for processAllPages.
    - processAllPages now calls calculateOutputImageProperties() with the temporary output image, thus the user can specify the properties of the temporary output image, e.g., set the page extent or the datatypes of input sub images.
    - The bypassing mechanism of the ML has been rewritten and now works across multiple bypassed modules and for memory images.
    - ModulehandleInput supports a new enum ALLOW_INVALID_INPUT, which removes the need to use REDIRECT_TO_DUMMYOP, which has been deprecated. See the ML Guide for details.
    - ml::PagedImage now supports sparse page tables larger than 2^31 pages, thus the ML now supports very large images.
    - ml::PlaneField and ml::RotationField have been moved from MLTools to the core ML and are now available on ml::FieldContainer via addPlane() and addRotation().

- MLLinearAlgebra now contains ml::Rotation, ml::Plane and ml::Line.
- ml::SubImage now contains the source image extent and knows its valid region (see getSourceImageExtent() and getValidRegion().
- ml::SubImage now supports caching via the new method ml::SubImageallocateAsMemoryBlockHandle().
- New ml::TypeTraits<> class offers easy access to compile-time information on ML datatypes.
- New ml::PageIDIterator class allows to incrementally calculate page ids in a given region.
- New mlWarning(), mlError(), mlFatalError() macros have been added which support streaming additional information into them and are nicer to use than ML_WARNING, ML_ERROR macros.
- Methods like getTile(), updateImageProperties(),... that took a module and an output index now take a PagedImage, which is less error prone (deprecated old methods are still available).
- The preferences variable MLCacheSize is now called MLCacheSizeInMB. You may need to adapt this variable in your mevislab.prefs file if you used MLCacheSize before (the value to MLCacheSize had been in KB).
- Some incorrect operations/results in MLHolds/MLRangeOrder and MLPromotedType where fixed.
- Incorrect identification of all floating point vector voxels as double voxels fixed.
- MLuint64 voxel type introduced to all ML classes, macros and switches as well as in the modules.
- ToolsscanInput function of MLTools now also analyzes all components of non scalar voxels. This fixes incorrect min/max values in MinMaxScan and Scale modules when using real min/max values.
- The former TVector/Vector class (now named ImageVector) has been revised and supports n-dimensional vectors now.
- The integer voxel types Vector16, Vector, and Vector64 have been removed. Now they are provided as 8, 16, 32, and 64 bit versions for the dimensions 2,3,4,5,6,7,8,9,10,16,32, and 64 as for the floating point voxel types, too.

- Open Inventor:
  - Input devices derived from SoQtDevice are now registered to the run-time type system, hence can be created via run-time type system (see Inventor/Qt/devices/SoQtSubDevice.h).

- MDL:
  - The list view control (MLABListViewControl) has been re-implemented, which e.g., allows to
    - set foreground and background color and font attributes on list view items
    - store a user data object on list items
    - use a combo box when editing list cells
  - New ButtonBox control, which presents Button controls in a layout that is appropriate for the operating system's look & feel. The order in which the child Buttons appear depends on the window system.

  - Added role attribute to Button control that specifies the buttons role within the dialog and is used by the ButtonBox control
  - Fields can now be connected even if they are disabled
  - New MDL tags styleSheetString and styleSheetFile to set the Qt stylesheet for any MDL control recursively
  - The MDL attributes bgOrigin and ignoreMinSize have become obsolete
  - Labels can now contain unicode characters with codes >= 256 (provided the file contains the correct header for UTF8 encoding)
  - New WebView MDL control integrates a full WebKit based browser engine into the MDL (see TestWebView for examples)
  - The MDL offers a new interface to extend the MDL with user MDL controls written in Qt/C++. This requires a Qt 4.6.2 SDK installation which is publicly available from Nokia. Have a look at the ColorChooserExampleControlTest and DiagramExampleControlTest module.
  - Label control now supports HTML hyper links via new tags allowLinks and linkActivatedCommand/ linkHoveredCommand
  - The line edit control can show a hint text if it is empty and doesn't have focus.
- Scripting:
  - Python scripting has been upgraded to version 2.6.4 .
  - PythonQt binding has been upgraded to PythonQt 2.0
  - NumPy is now included in the Python distribution that comes with MeVisLab.
  - MeVisLab now offers complete access to the Qt libraries in Python, have a look at the MeVisLab Scripting Reference for details. This is an experimental feature.
  - Python C Extensions (*.pyd/*.so) are now supported anywhere (it is no longer required to place them in Package/lib/python), so you can place them to Package/Modules/Scripts/python as you would do with *.py files.
  - MLABImageFieldimage() offers a new API to the underlying ML image, including requesting image data via getTile() to do image processing in Python.
  - It is now possible to implement ML image processing modules in Python. Search for Python* in the quick search to see some examples.
  - New scripting classes MLABDicomTree and MLABDicomTag. MLABDicomTree is returned by MLABImageFieldgetDicomTree().
  - File dialog boxes now remember the last selected directory persistently
  - MLAB.mdlControlAtGlobalPosition() and MLAB.mdlControlAtCurrentCursorPosition() added to find an MDL control from a global screen position or the current cursor position.
  - Added classes to handle field-value test cases, that allow applying a parameterization to a network, verifing expected results and saving specified field values.
- IDE

  - A new quick search in network has been added (CTRL-F on the network), which allows to quickly find modules, fields and associated files.
  - A lot of network rendering improvements.
  - "Debug Output" inspector now supports exploring the containing folder of a file via holding CTRL key while clicking on a file link
  - Note items will now display non-ASCII characters in the network.
  - The IDE partly relied on the Qt3Support library from Qt. The remaining parts have been rewritten so that MeVisLab solely relies on Qt4.
  - "Extras->Reload Imported Python Modules" reloads all imported Python modules, which previously required a manual reload() call or a restart of MeVisLab.
- Packages
  - Packages now support a lib/release and lib/debug directory, which is placed into the PATH depending on the version of MeVisLab that is started (Release/Debug), this allows to place thirdparty DLLs into release/debug dirs if both debug/release DLLs use the same name
- Tools
  - added -prefs, -ignoreprefs and -scanpath commandline options to ToolRunner
  - added -prefs and -ignoreprefs commandline options to MeVisLabPackageScanner

Incompatible changes

- MLLinearAlgebra
  - TVector<> cannot be used as 6D vector any more, it has to be replaced with TImageVector<>.
  - Some constructors of ImageVector (the former Vector/TVector<> class) do not implicitly cast integer or differently types vectors any more, because this had many undesired side effects. These constructors must explicitly be cast now, i.e. instead of ImageVector v=17; use ImageVector v=ImageVector(17); now.
- ML
  - ml::Module::calcInSubImgProps has been removed and needs to be manually replaced by calls to ml::PagedImage::setInputSubImage*() calls on the output image in calculateOutputImageProperties().
  - ml::Module::connectInput() has been removed, replace it by using handleInput().
  - ml::SubImage is no longer derived from ml::ImageProperties.
  - ml::SubImage/ml::ImageProperties no longer have an isUpToDate flag.
  - ml::SubImage no longer stores min/max voxel values, you may get those value via getInputImage(N).getMin/MaxVoxelValue().
  - ml::Host/Module::processAllPages no longer takes two leading integer arguments, the second argument (the input image properties that should be used for the output image) has been removed and can be replaced by code in calculateOutputImageProperties() if it wasn't set to 0.

  - ml::Field::attachField(targetField) is no longer allowed with non Notify/SoNode/Base/OutputConnector Fields as targetField.
  - TSubImage<> no longer contains a cursor. Change your code to either make use of a TSubImageCursor<> or ConstTSubImageCursor<>. Alternatively you may use the TSubImageWithCursor<> class which offers the old cursor behavior of TSubImage<>.
  - The integer voxel types Vector16, Vector, and Vector64 have been removed. See new Common Enhancements / Fixes (MeVisLab) for replacement by new types.
  - ml::PagedImage now caches its state for invalid output images as well, since it turned out to be a performance problem when updateImageProperties is called on invalid output images recursively. In some situations, this requires that you call getOutputImageField(outIndex)->touch() in your code to notify the ML that the output image has changed and that it should call calculateOutputImageProperties again. This is especially the case if you want processAllPages(outIndex) to take into account the new output image state. If you do not want to cause update notifications, you can call getOutputImage(outIndex)->setUpdateOfImagePropertiesNeeded(true) instead, but this should rarely be the case.
- MeVisLab
  - The Vector2f, Vector3f and Vector4f fields have been renamed to Vector2, Vector3 and Vector4, this affects Python/JavaScript scripts that made use of MLABField.type or MLABField.shortClassName(), which now return "VectorX" instead of "VectorXf".
  - MDL variables OS, OSVERSION, ID, CPU and DATE removed. There is still LINUX/UNIX/MACOS/WINDOWS variables to check the OS

Common Enhancements / Fixes (Modules)

- A new event type has been added to the ObjMgr. The event 'ConnectedToDatabase' is sent on (re)connection of ObjMgrClients to the database and may be used to update dependent networks/scripts to the current database state.
- SoShader framework
  - Shaders attached via the shaderObject multifield are now aware of OpenGL context switches
  - Shaders may inject code into the main funtion via GLSL snippets (experimental)
  - SoShaderParameterMLImageProps adds a vec3 NameOfParameter_voxelSize uniform to access the voxel size of an ML image sampler
- SoGVR framework
  - SoGVRMaskSettings and SoGVRMaskVolume now allow to replace the WorldToVoxel matrix of the mask image (similar to SoGVRTransformedVolume).
  - SoGVRTransformedVolume and SoGVRMaskVolume now correctly load images with three and four color dimensions.
- New Modules
  - New CompareNetworks module allows to compare *.mlab files both textual and visual
  - New module SoGVRMeanIPMode offers Mean Intensity Projection for the GVR volume renderer
  - InventorRayPicker allows to apply a ray pick action to an inventor scene

  - SwapFlipDimensions allows to swap/flip two dimensions
  - New module WEMVascularSystem generates a smooth 2-manifold surface to represent a vascular system. This module complements SoVascularSystem.
  - The library MLVesselGraph is now available so own modules to compute on vessel graphs or to render vascular systems can be written.
  - New modules to create (FieldValueTestCaseEditor and -Generator), run (FieldValueTestCaseIterator) and convert field-value test cases to CSV (FieldValueTestCaseConvertor).
- File IO & Data Compression:
  - New wrapped itkImageFileWriter and revisions of wrapper for itkImageFileReader. See ITK/VTK.
  - New macro module LoadAny, a multi file extension/format loader with format detection for images, meshes, and other file types. Opens most readable MeVisLab file formats automatically.
  - New very fast LZF data compressor available in MLImageFormat modules and MLDataCompressorFactory for maximum compression speed (also in SDK and public sources).
  - New LZMA data compressor available in MLImageFormat modules and MLDataCompressorFactory for high compression ratios (also in SDK and public sources).
  - Performance of "Optimize Data" option for MLImageFormatSave/Load (de)compressors (and therefore also of file IO) significantly improved and set automatically if useful (i.e. the user selectable toggle disappeared).
  - New difference coding mode added to some data compressor and in MLImageFormatSave to improve compression ratios sometimes significantly.
  - New mode "Preserve Cache File" in MLImageFormatFileCache allows persistent cache files and reusing them (from) elsewhere.
  - Fix: ImageFormatFileCache does not request useless pages from input any more when reading data from cached file.
  - Performance fix in MLDicomTree: Significant optimization of opening times of .mlimage files with huge dicom tags.
  - Further revisions of all MLImageFormat modules (documentation, internal class hierarchy, browse dialogs, better use of variables in paths, showing all file information in all modules, additional buttons etc.)
  - ImageLoad module now also reads MLImageFormat files.
- Non scalar image support:
  - New integer voxel types for many dimensions and types available. See new ML features.
  - MPR, MPRPath AccumulateImage, Resample3D, OrthoMPR, Reformat, OrthoSwapFlip, and SoOrthoView2D also operate on non scalar image voxels now.
  - New module TypeFromDimension to create a non scalar voxel image from scalar components of a given dimension, simplifying and speeding up some often used operations.

  - Often requested mode for direct decomposition of non scalar voxels to a certain dimension (e.g., vector to color image or vector to time dimension) is now available with TypeToScalars by a newly selectable output dimension.
  - 16 new (and often requested) unary operations on non scalar voxels in Type/TensorArithmetic1 (especially operations with constants, for reordering, and sorting, Phong calculation). Some incorrect sign calculations in Comp Mul and Comp Sum modes fixed. Now these module also support float based processing, i.e. implicit conversions are reduced/removed to speed up operations.
  - View2D and View2DExtensions show all values of non scalar input voxels (enabled by default). This is configurable in View2D Options panels and its settings panel under "Voxel Value". Already saved modules reload former behavior to provide compatibility.
- Python
  - New module PythonArithmetic is similar to Arithmetic and offers to implement ML image processing by writing the processing code using inline Python/NumPy code in the panel of the module.
- ImageCalculationDiagnosis offers diagnosis information about the ML image calculation properties of the connected input image to debug and optimize ML modules.
- CSO:
  - Many improvements for CSOConvertTo3DMask.
  - Fixed multi-threading related bugs for CSOConvertToImage.
  - The module WEMClip now offers the generation of CSOs along the clipped WEM border.
- ITK/VTK:
  - Upgrade to VTK version 5.7 from 30th of April 2010, about 346 new modules added, 33 removed, many more changed. See any vtk module help for details.
  - Bug fix: Missing shader support has been fixed.
  - Upgrade to ITK version 3.19 from April, 29th, 2010, 10-20pm MET from the CVS repository from http://www.itk.org. See any itk module help for details.
    - Fix in conversion between ML and ITK patient coordinate system which occurred when coordinate systems were rotated.
    - Revised: itkImageFileReader supports non scalar voxel types, uses less memory, has new open and close buttons, provides better path proposals while browsing and the options to enable/disable sub voxel shift in patient coordinate system.
    - New: itkImageFileWriter wrapped, providing the option to save images in numerous file formats and scalar and non scalar voxels types.
    - Many modules now are compiled also for 6 dimensions to support non 2D/3D cases.
    - Some improved example networks (of some itkLevelSet-, itkFastMarching-, and itkVectorConnectedComponentImage filters).
    - Invalid usage of flat kernel elements removed.
    - Some modules compiled with incorrect input/output type combinations fixed.

    - Metric modules revised: New setting makes it configurable when to update the FixedImageRegions parameter.

    - ITKSupport is part of the public sources now.

- Misc

  - New module DicomTagBrowser which offers an improved view over DicomTagViewer of an image's DICOM tags

  - BaseBypass changed its behavior: It disconnects input and output if bypassing is off

  - MinMaxScan and BoundingBox update behavior changed: Since both modules cannot buffer the input image, (auto)UpdateMode == "off" was removed and redirected to "clear".

  - MergeRegions:

    - Fixed output image region computation for cases where the output world boundaries exactly hit the voxel boundaries (previously the box was one voxel to large in these cases). Now consistent with SubImage behavior, except that the MergeRegions module still allows for empty boxes.

    - Improved notification behavior

    - Many small improvements/fixes.

  - Marked output fields as non-persistent in many modules to improve diff-behavior of .mlab files

  - Decreased amount of unnecessary output notifications for various modules

  - Implemented bypassing for MinMaxScan when datatype does not change

  - Surround: Added border handling to UI (can be important if object boundary touches image border)

  - IntervalThresh Bugfixes:

    - For newMin/newMax: Disabled any clipping with respect to the input image type. This is no problem since no sliders and thus no min/max values are required (previously the value clipping could cause unwanted min/max values for subsequent images)

    - For center/width/min/max: Made sure to internally clamp the thresholds explicitly to the input image datatype range before their application (previously, a cast was performed which could result in false thresholds)

    - Now also clipping user values and computed output min/max against datatype boundaries.

  - Fixed Reformat crash: Previously, the Reformat module would crash in VirtualVolume mode under certain conditions: 1. Both previously invalid inputs were validated at the same time (possibly because they were derived from the same updated image). 2. The notification from the right input (input1) arrived first. 3. An ML module with an auto-update functionality requesting image data was located behind the reformat module.

  - Counter can now also handle floating point values

  - Added fieldHelper.js and fieldHelper.py script files for some convenience scripting functionality (mainly connecting fields/setting field values only if the field is not already connected/the field value is not identical, thus improving performance by preventing unwanted notifications)

- StopWatch now has a 'pause' functionality
- ExtractLabels now has up to 20 outputs and allows comments for each output
- ImageLoadMulti:
  - Fixed an error that could occur when pressing 'parse list' from the parser setting UI
  - Added an option to treat all paths as relative, even when starting with a \, / or C: (disabled by default).

- Fraunhofer MEVIS Release Packages (ThirdParty / Foundation / Release):
  - OpenCL support + MLOpenCLKernel1 module to experiment with OpenCL
  - OpenCV support (highgui Mac OS X 64-bit is experimental)
  - New OpenCV-based video camera back-end for VideoCapture module, supports Windows + Linux in addition to the QuickTime-based Mac OS X back-end.
  - New module FastMorphology for binary image morphology (dilation, erosion, closing, opening) (also in public sources)
  - New module ImageResample for fast(er) image resampling using integer resampling factors.
  - New module FFT for fast Fourier transformation in all 6 dimensions (also in public sources).
  - New module ImageBuffer.
  - New modules QuadratureFilter and QuadratureToTensor for filtering using log-Gabor filters and generation of structure/orientation tensors.
  - New XMarkerListModules: JitterXMarkerList, XMarkerListFilter, XMarkerAtIndex (also in public sources)
  - New project StochasticCollocation with many modules for managing piecewise multilinear functions and interpolation polynomials on sparse grids in arbitrarily many space dimensions.
  - New CSOUtils modules: CSOReduce, CSOSmooth, CSOMarkerGenerator and CSOConvertToXMarkerList (some parts are also in SDK and in public sources).
  - New project CurveUtils with modules CurveFilter, CurveGenerator, and CurveDownsample.
  - Many fixes in Arithmetic module.
  - Module TensorToEigensystem have a new eigenvalue sorting option and has been optimized for performance with a new math library.
  - New modules for DICOM import: DirectDicomImport, DirectDicomImportOutput, DirectDicomImportImageOutput, DirectDicomImportRTOutput, and DirectDicomImportSROutput. This newly developed package supports or example spectroscopy, radio therapy, structured reporting, gantry tilt files (some parts are in SDK interfaces).
  - New DiffusionMRI projects with ~30 new modules for diffusion imaging, including DTI preprocessing, visualiztaion, fiber tracking and fiber clustering
  - New experimental CTCompress2 data compressor available in MLImageFormat modules and MLDataCompressorFactory for CT and MR data (also in SDK and in public sources).

    - Some DICOM tool functions of project DICOMAccessories added to SDK interfaces.
- Removed the module Window2 and replaced it with the module Scale.
- Replaced ComposeVolumeFromSlices by ImageComposer module

qmake build system

- The default for TEMPLATE in *.pro files has changed from TEMPLATE = lib to TEMPLATE = app. If you are using profiles that generate shared libraries/dlls your profile now needs to contain a TEMPLATE = lib line (which was already part of the MeVisLab module wizard in 2.0, but maybe you have profiles which miss that line, these need to be adapted).

Linux Edition

- Known Problems
    - ImageLoad crashes if the png library encounters an error. This sometimes occurs when a png image is closed.

Mac OS X Edition

New Features:

- MeVisLab IDE:
    - MeVisLab has been *optimized for 64-bit* Mac OS X
    - Mac OS X 10.6 Snow Leopard is fully supported (Mac OS X 10.5.2 Leopard or newer required)
    - The GUI Toolkit Style can be changed during startup of MeVisLab by pressing the Shift key
    - The GUI Toolkit Style can be changed using the -style <style> command line option. Possible values for <style> are 'Native' and 'Application'
    - The GUI Toolkit Style is indicated by the application icon
    - MeVisLab provides audio feedback to the user after a pressed key modifier has been recognized by the MeVisLab startup process
    - Experimental AppleScript support (see MeVisLab Mac OS X Edition User's Guide)
    - Help requested via command line is printed to the console (--help, -help, -h option)
- MDL:
    - Added support for MoviePlayer control
    - Added experimental MacControl control to enable native Mac OS X user interface mockups referenced from Apple Interface Builder Nib files (see MacControlTest example module). Full source code of the control is included
- SDK:
    - The MeVisLabProjectGenerator provides a contextual menu to .pro files in the Mac OS X 10.6 Finder
- Documentation:
    - Updated MeVisLab Mac OS X Edition User's Guide

    - Updated OsiriX MeVisLab Bridge User's Guide
- OsiriX MeVisLab Bridge:
    - OsiriX plugin:
        - Patient series can be sent directly to MeVisLab from the OsiriX Database Browser (plugin type is now 'Database')
        - Added XML-RPC support to the OsiriX plugin ('SendSelectedSeriesToMeVisLab' method)
    - OsiriXBridge MeVisLab module:
        - The user's temporary directory is used for conversion instead of /tmp if possible
        - DICOM import failure messages are displayed in the OsiriX MeVisLab Bridge module panel as well
        - Experimental support for DirectDicomImport module

Changed Behavior:

- Modules:
    - The Accelerate.framework is no longer used by default if the module has a dependency on cblas or clapack (`CONFIG+=cblas / clapack`). Add `macx_accelerate` to the `CONFIG` variable in the projects .pro file (`CONFIG+=macx_accelerate`) if you want to employ the optimized version for Mac OS X.
    - SoGVRTransformedVolume, SoGVRTransformedVolumeSettings: The transformation matrix supplied by the user now replaces the WorldToVoxel matrix of the transformed volume, and is not multiplied to it any more.

Resolved Issues:

- MeVisLab IDE:
    - Flickering of undetermined progress indicator
    - The undetermined progress indicator did not hide if the application became inactive
    - Displaying the Help menu could take a long time
    - IDE Toolbar icons were of different sizes depending on the chosen style
- MeVisLab ADK:
    - Shipping MeVisLab standalone applications indicated the error state in the icon which is not desired
- Modules:
    - Recording of QuickTime movies could fail on Snow Leopard (VideoCapture)
    - Apple Remote Control device support did not work for newer MacBook Pros (SoAppleRemoteDevice)
    - MeVisLab could crash while sending DICOM data from OsiriX if the conversion directory had not been set (OsiriXBridge)

- Mouse double-click timing is no longer a fixed value but read from the user's settings in Mac OS X
- Slicing in the View2D did not work well with pixel-scrolling mice (e.g., Mighty Mouse / Magic Mouse)
- Improved sensitivity of mouse wheel in SoMouseGrabber for pixel-scrolling mice
- MatlabScriptWrapper did not work with MATLAB 2010a because of a name clash in the boost libraries

Version 2.0 final (2009-06-03)

Additional Fixes for the Final Release Version

- Improved performance of VCPROJ generation from *.pro files on Windows (qmake).
- Reworked example networks of SoShader library.
- IDE now also deletes .auto file when saving with "Save Copy As".
- CSOConvertTo3DMask:
  - Added an option to use only the seed points for generating the implicit surface function (in contrast to using only the path points).
  - The CSOs can now optionally be smoothed internally by a spline approximation to diminish the oscillation of the resulting surface function.
- CSOLiveWireProcessor: on changing the slice while generating a CSO, the module does not crash anymore and prints out a message that the current CSO (with its id) is invalid and should be removed.
- SoView2DCSOEditor:
  - Fixed a bug where seed points of CSOs with only 1 or 2 seed points could not be moved.
  - Seed points under the mouse cursor are now rendered a little bit bigger.
- WEMInfo: fixed an issue with fired notifications during processing which caused the IsoSurface macro to crash under certain circumstances.
- TestPattern: fixed the interpolation of XRamp and YRamp for complex data types (vec, mat, complex, etc.).
- fixed typos and errata in various documentations.
- Added the ADK documentation to the ADK installer.
- Fixed crash on exit of standalone MeVisLab applications.
- Fixed example networks (replaced modules by publicly available modules): CSOMarkerGenerator, XMarkerListBounds, XMarkerShortestPath, SoGVRDrawOnPlane, SoViewPortRegion, SceneInspector.
- Added GUI panels for: ComplexArithmetic1, ComplexArithmetic2, ComplexToReal.
- HistogramPeakEstimate: fixed a bug which caused the y-values of the peaks to be set to their x-values.

Version 2.0 rc (2009-05-08)

Common Enhancements / Fixes (MeVisLab)

- New MeVisLab package structure:
  - MeVisLab Modules are now managed in *Packages*.
  - All Tools work on the new package structure.
  - New MeVisLab Tool Runner that facilitates daily developer tasks:
    - Building documentation:
      - *doxygen* for source code.
      - *DocBook* for HTML and PDF documentation.
    - Generating project files:
      - .vcproj for Windows compilers (VC8 / VC9).
      - .xcodeproj for Mac OS X Xcode (instead of using the dedicated MeVisLabProjectGenerator).
      - Makefile for Linux gcc compilers.
    - Building of projects:
      - Supports IncrediBuild on Windows platforms.
      - Supports the analysis of project dependencies and the building of multiple projects in the correct order.
- Revamped network rendering:
  - New module colors (distinguishable for all types of color-blindnesses).
  - Advanced selection highlighting:
    - Selected modules, attached connections and attached modules are visualized.
    - Hold SPACE to let the highlighting appear more articulate.
  - Mouse over highlighting for all network items.
  - Context menu for connections:
    - Display values of parameter connections.
    - Remove single / all connections.
  - Added interactive mini map for a better navigation.
  - Added visual grouping for modules.
  - Added notes for annotating a network.
  - Added placeholders for in/out connectors in internal networks of a macro.

    - Enabled the horizontal moving of connections.
    - Added many short cuts, for example:
        - For opening the automatic panel.
        - For opening the internal network of a macro.
        - For editing the instance name of a module.
    - Print modes for network (simple rendering / black & white rendering).
- An enhanced way to place instantiated modules on a network did not make it into the release.
- Enhanced Scripting methods:
    - MLAB.startXxxDrag methods can now take an icon and a hotspot location.
    - MLABFileManager.getExecutable().
    - MLABFileDialog.getExistingDirectory().
- New Scripting methods:
    - MLAB.allModulesForPackageIdentifier()
    - MLAB.compilerInfo()
    - MLAB.qobjectInherits()
    - MLAB.runCommandDetached()
    - MLAB.showColorDialog()
    - MLAB.stringToEncodedUrlString()
    - MLABField.isConnectedToFieldInSameNetwork()
    - MLABFileManager.getExecutableOnLocalMachine()
    - MLABFileManager.macGetBundleSharedSupportDir()
    - MLABModule.createConnectors()
    - MLABModule.getPackage()
    - MLABModule.isDescendentOf()
    - MLABModule.package()
    - MLABModule.scriptVariable()
    - MLABModule.setScriptVariable()
    - MLABSqlQuery.lastInsertId()
    - MLABTabViewControl.setTabText()
    - MLABTabViewControl.setTabToolTip()

  - MLABTabViewControl.tabText()
  - MLABTabViewControl.tabToolTip()
  - MLABTextViewControl.selectedText()
  - MLABTree.findChildrenRecursive()
  - MLABWindowControl.isMinimized()
  - MLABWindowControl.showMinimized()
- Removed Scripting methods:
  - MLAB.buildInformationVariable(): use information from MLABPackage
  - MLAB.buildInformationVariables(): use information from MLABPackage
  - MLAB.hasBuildInformationVariable()
  - MLABFileManager.getDocsDir(): use MLABPackage.documentationPath()
  - MLABFileManager.getExecutableDir(): use MLABFileManager.getExecutable()
  - MLABFileManager.getFileInfo()
- Updated / added documentation:
  - Reviewed the *ML Guide*.
  - Added an elaborate *Getting Started* document that covers all first steps from building networks to building own modules in C++.
  - Added a *MeVisLab Reference Manual* that covers all GUI elements of MeVisLab.
  - Added documentation concerning the new package structure and all new tools.
- Added support for Postgres and SQLite databases (see the MLABSqlDatabase and MLABSqlQuery scripting classes for general database support).
- Other changes to the IDE:
  - The Python scripting console can be set to execute default commands on startup. These can be edited in the preferences dialog. Default is "from mevis import *".
  - MeVisLab got a new commandline option "-quick" for faster startup performance when the module database has not changed. Under Windows, look out for the MeVisLab icons with the red arrows on it to start MeVisLab quickly.

Common Enhancements / Fixes (Modules):

- Renamed a number of modules. The old name is left as the 'deprecatedName' to ensure compatibility with old networks. Renamed modules are: CalcGradient, CalcVolume, CatCurves, ConcatImages, ConnectComp, ConstImg, ExtConvolution, ImgLoad, ImgPropConvert, ImgSave, ImgStat, JointHist, JointHistMask, MarkerStat, StdDeviation, Subimage, SubImageStat, VoxValueRescale, SoFrameBufferSampler, SoGVRVolRen, SimpleImgStat. The new names are mostly the non-abbreviated version, e.g. ImgStat becomes ImageStatistics.
- Added SoShadow* modules group to cast and interact with shadows.

- ML image file format default file extension has changed from .mif to .mlimage (.mif will still be recognized though).
- SoTexture2 no longer crashes on Win+VC8. As a side effect it is no longer possible to load SGI Image files (.rgb). Supported are JPEG + GIF. See also SoMLTexture2 for additional image formats.
- Added event consumption mechanisms for SoView2DExtensions and SoMouseGrabber.
- SoView2DAnnotation has more display options.
- Added new module SoViewportRegion to allow rendering into a specified region of the render area and to combine several renderings in one render area.
- Added MovieCreator module, to allow writing image sequences as movie files.
- The new SoMovieScripter module allows to rotate Open Inventor scenes automatically. It also offers the linear interpolation of a variety of variables over time.
- New Arithmetic module for image arithmetic with general mathematical expressions.
- New Calculator module for general mathematical expressions.
- New TubularTracking module for tracking tubular structures and vessels.
- New GaussSmoothing module for fast filtering with a Gaussian filter kernel.
- New GaussGradient module for calculating the image gradient using a derivative-of-Gaussian approach.
- New GaussHessian module for calculating the image Hessian using a derivative-of-Gaussian approach.
- New Vesselness module for calculating a vesselness measure of an image.
- New ImageWarp module for deforming an image according to a dense vector field.
- New TensorToEigensystem module for calculating eigenvectors and eigenvalues for a tensor/matrix image.
- Added DeMosaic module which extracts 4D data sets from fMRI and DTI data e.g. provided by SIEMENS scanners.
- GVRImageToVolume and GVRVolumeLoad got a parameter "additionalTransformation" to change the world position of GVR volumes.
- XMarkerLists:
  - New module InterpolateXMarkerList for interpolating new points in an XMarkerList.
  - New module XMarkerShortestPath for finding the shortest path between two points in an XMarkerList.
  - New module SmoothXMarkerList for smoothing the positions in an XMarkerList.
  - New module TravelingSalesman for finding the shortest route visiting all positions in an XMarkerList.
  - New module XMarkerListBounds for finding a bounding box or sphere enclosing all points in an XMarkerList.
  - New module XMarkerSamplePattern for sampling pre-defined point patterns.

  - New module XMarkerListMaxDistance for finding the max distance between points in an XMarkerList.
  - New module MarkerToMask for converting an XMarkerList into an image mask.
  - New module XMarkerListConvert for converting between world and voxel coordinates.
- Added two new options to SigmaFilter that are disable by default:
  - Use Linear Weighting: If enabled, the kernel voxels in the Sigma Interval are not all weighted equally, but relative to their distance to its center.
  - Local StdDev Max: Specifies whether and against which value the computed local standard deviation is to be clamped prior to the definition of the sigma interval. This can be useful if there is a known maximum of the noise level, but adaptive filtering is still desired in the areas with lower noise.
- RegionGrowing:
  - Fixed a missing enum string bug that could cause a crash on some platforms. Now all 4D neighborhoods are associated with the correct enum item name.
  - Previously, the DICOM tags RescaleIntercept and RescaleSlope were only read correctly if 'auto update unit type' was enabled. Now this happens always if 'HU' is selected as unit type (no matter if automatically or manually).
  - Previously, the unit type was not auto-updated when 'auto update unit type' was enabled with an already connected input image. This is now fixed.
  - Now the default HU intercept for images with no such DICOM information is 0 instead of -1024.
  - Reduced the number of (unnecessary) output notifications.
  - Output values are no longer clamped to the uint16 range.
  - Fixed some memory leaks introduced by the 4D extension.
  - Fixed boundary overlap computation for 4d images.
  - Improved 64 bit support.
- MergeRegions:
  - MergeRegions now supports up to 20 in- and outputs (max number can be changed in MergeRegions.h).
  - Explicit region selection adapted, requires re-save of network to remove warnings. Deprecated modes still supported though, should be fully backwards compatible.
  - Inactive in/outputs are hidden in the MeVisLab module representation. Default number of in/outputs now 2.
  - Added some new merge modes.
  - Reduced the amount of inconsistencies in the merge modes, more consistent and better documented fillValue handling.
- New MacroModule CombineLabels: The CombineLabels module is designed to simplify label image handling by allowing to add objects to an existing label image or combine them to create a new

one. In contrast to mere OR operations with specified bitmasks, the module supports the invertible encoding of all pow(2, n) objects that can be represented by an n-bit label image and thus their separate extraction using the ExtractLabels module.

- New Module ExtractLabels: The ExtractLabels module is designed to simplify label image handling by allowing to extract up to 10 objects identified via a specific (sub)bitmask from a given (integer) label image. In contrast to a mere AND operation with the bitmask, it allows the separate extraction of any of pow(2,n) objects that can be represented by an n-bit label image.
- CSO:
  - CSOFilter: Implemented the filtering much more efficient.
  - CSOConvertTo3DMask: The starting condition for scanning the surface is more stable now. Also added a debug mode for very large surfaces / huge amount of CSOs.
  - CSOConvertToImage: Fixed an issue with the calculation of the voxel bounding box.
  - CSOBase: Fixed some issues with the computation of the normal and area of a CSO.
  - SoView2DCSOEditor:
    - The display of and the interaction with a CSO can now be limited to CSOs that lie in the current viewer's plane.
    - If a CSO is edited so that it would be not in-plane anymore, it can optionally be auto-leveled.
  - SoCSO3DVis supports now a StylePalette.
- WEM:
  - New module WEMPVL that generates PVLs for existing WEMs and can be used for testing.
  - Fixed issues with the copying of WEMPVLs; PVL values are also now generated for newly added primitves (i.e. in WEMSubdivide).
  - New module WEMMerge that merges all patches of incoming WEMs into one output WEM.
  - Added methods to automatically close holes in a WEM.
  - New modules SoView2DWEMContour, SoWEMInteract and WEMExtrude: can be used to draw contours on a 3D viewer and to extrude those contours to become a WEM which in turn can be used with the WEMBoolOp to interactively shape surfaces.
- ITK:
  - Upgrade to ITK version 3.9 from September, 4th, 2008. Some modules wrapping classes from itk review directory have been removed since licensing is not clear or since they were not useful. For details see technical notes in module help of any wrapped itk module.
- VTK:
  - Upgrade to VTK version 5.3 from September, 3rd, 2008. For details see technical notes in module help of any wrapped vtk module.

Mac OS X Edition

- New Features:
  - MeVisLab IDE:

    - This release has been optimized to run on Mac OS X 10.5 (Leopard) or better. As a result, MeVisLab does no longer run on Mac OS X 10.4 (Tiger).
    - The MeVisLab application bundle now contains a debug and a release mode version of MeVisLab that can be started separately (press the Option key during launch to toggle the version; other methods to choose a version include the Preferences dialog or command line options).
    - An option has been added to the Preferences that allows you to choose an alternative IDE toolkit style (used for cross-platform application development).
    - MeVisLab is now Time Machine aware, log and temporary files are excluded from backup.
    - The Busy Cursor has been replaced by a more descriptive panel using an indeterminated progress indicator + text.
    - Additional document icons for ML Image (.mlimage), GVR (.gvr), and Open Inventor (.iv) files have been included.
  - Image Formats:
    - A new ML Image IO (a.k.a. MFL) plugin has been included that interfaces with the Mac OS X Image IO framework and makes most of the Mac OS X image formats readable by MeVisLab (e.g. ImageLoad). Some of the new formats now supported are: JPEG-2000, Adobe Photoshop, Camera-RAW (Canon, Nikon, ...), GIF, Radiance, Quicktime Images, DNG, PICT, Targa, SGI Image, Windows + Mac Icons, PDF Images. Images may contain multiple layers.
  - DicomService:
    - The DicomService is now part of the MeVisLab application bundle and does no longer require a separate download.
  - SDK:
    - The MeVisLabProjectGenerator (a.k.a. MacProfileTranslator) has been completely rewritten and may now be used in GUI mode (default) or from the command line (see '.../MeVisLab.app/Contents/Support/MeVisLabProjectGenerator.app/Contents/MacOS/MeVisLabProjectGenerator --help' for available options).
    - Support for the RELATEDFILES project variable (.pro files) has been added. It allows you to add related files (e.g. .def or .html files) to a 'Related Files' group in the 'Groups & Files' list of your Xcode workspace window for quick access.
  - Documentation:
    - Expanded and revised users guide *MeVisLab Mac OS X Edition*.
    - New document *OsiriX MeVisLab Bridge*.
  - Modules:
    - New ML module VideoCapture, that uses Core Video to provide live video input for all video devices supported by Mac OS X - try your iSight (also in SDK interfaces).
    - Updated OsiriX MeVisLab Bridge that supports and requires OsiriX 3.3 or later.
- Bug Fixes:

- MeVisLab IDE:
  - The hidden panel issue has been fixed (module panels did not show up sometimes).
  - The styles font size mismatch between Mac OS X and Windows version has been resolved (now the same font size given in point results in approximately the same pixel size on Mac OS X and Windows).
  - The automatic calculation of available texture memory for newer graphics cards has been improved.

- Important:
  - Some of the MeVisLab helper tools have changed their name and/or id. Sometimes the Mac OS X application database (a.k.a. launch services database) does not detect the changes as soon as you would like. To be on the safe side, execute the following command-line in the Terminal to rebuild the application database after you've installed MeVisLab:
  -
```
/System/Library/Frameworks/CoreServices.framework/
  Frameworks/LaunchServices.framework/Support/lsregister
   -kill -r -domain local -domain system -domain
   user
```

- Notes:
  - Leave the available texture memory setting in the MeVisLab preferences untouched for the new MacBook Pro with NVIDIA 9400&9600 chipset (5th generation), MeVisLab will automatically calculate the exact amount of texture memory for the currently active graphic chip (according to the energy settings). You may remove any manual overwriting of the texture memory setting by executing the following command-line in the Terminal:
  -
```
defaults delete de.mevis.MeVisLab
  GlobalSettings.TextureMemoryInMB
```

Windows Edition

- Known issues:
  - 64bit version of MeVisLab is missing the ctypes Python module, because this cannot be build on 64bit Windows.

Linux Edition

- Changes of the development scripts:
  - MeVisLab.sh and init.sh do not exist anymore.
  - New scripts exist in >MeVisLab-Install-Path</bin. Their purpose is to call commands with certain environment variables set:
    - MeVisLab and MeVisLab_d will start MeVisLab in release and debug mode.
    - mlmake will build MeVisLab projects. It will create the makefiles if they do not exist already by calling the bash script next to the pro-file.
    - mlgdb starts the gdb. If no arguments are given, then MeVisLab will be debugged.
    - MeVisLabPackageScanner runs the MeVisLab package scanner.

- ToolRunner starts the tool runner.
- mlbash starts a bash session.
- mlcodeblocks starts the codeblocks IDE.

Version 1.6.1 (2008-08-24)

Common Enhancements / Fixes (MeVisLab)

- ML:
  - Subtle bug in priority management of the ML Cache fixed. It required an ML interface change and update to version number 1.8.68.23.87 in mlVersion.h. Relink all libraries dependent on the ML.
- MDL:
  - The abstract Control got an attribute destroyedCommand mirroring the initCommand.
- Scripting:
  - MeVisLab pointer objects (e.g. controls) can now directly be used as keys in Python dictionaries.

Common Enhancements / Fixes (Modules)

- MemCache:
  - Fixed: Module MemCache always returned the first output of a connected module, not the connected output.
- eatDicom:
  - fixed the scanning of scouting images, where some mandatory information could be left unset, like the PatientOrientation; in those cases, a message is printed to the console and the according structures are filled with 0.
- DICOM macros:
  - Added new module DicomQuery to query and retrieve data from a DICOM server. It's quite basic at the moment, suggestions are welcome.
- CSO:
  - CSOLiveWireProcessor: fixed a memory leak
  - CSOIsoGenerator: fixed a memory leak
  - CSOPathPointIterator: new helper structure
  - CSOAffineTransformation: fixed numerous update issues
  - CSOFreehandProcessor: added a field for adjusting the interpolation fineness, so that contours can now be drawn onto very small images with a very high zooming level
  - CSOFilter: fixed a bug that could lead to crash if the filter string was left empty
  - CSOFilter: added filter criterion for CSOs and CSOGroups: timepoint

  - CSOLoad: uses now ifstreams internally (enables the use of UTF8 characters in the path)
  - SoView2DCSOEditor: added a missing CSO_FINISHED notification
  - SoView2DCSOEditor: the picking is now more accurate
  - SoView2DCSOEditor: the module can now render correctly in 3D
  - CSO library: added undo/redo for removing/adding whole sets of CSOs
  - CSO API: changed 'get'-methods to be 'const' where possible
- WEM:
  - WEMClip: re-added module
  - WEMBoolOp: fixed triangulation issues
  - WEMLoad: added an 'Unload' function, fixed a bug that could lead to a crash if a lot of small WEMPatches were loaded
  - WEMSubdivide: fixed a bug that could lead to a crash if the surface has holes
  - WEMSubdivide: enhanced the speed of the module greatly
  - SoWEMRenderer: fixed an issue with the coloring of nodes
  - WEM API: added new temporary structure: WEMIndexedFace
  - WEM GUI API: added new GUI substructures
- Viewers:
  - Stereo viewing couldn't be activated on the Open Inventor viewers, even if the graphics card supported this. (You need to enable the context menu for this and set this from the preferences dialog.)

Mac OS X Edition

- MeVisLab startup time has been greatly reduced

Windows Edition

- Fixed a problem when building projects where the project file name contained a dash

Version 1.6 (2008-03-25)

Common Enhancements / Fixes (MeVisLab)

- New welcome screen for MeVisLab.
- Automatic detection of usable screen space.
- New integrated TextEditor (called MATE) with syntax highlighting, indention, completion and a context sensitive help.
  - **Note:** Currently, MATE uses latin-1 encoding for text files on all platforms

- Improved Scripting consoles
  - Advanced JavaScript and Python completion.
  - Context sensitive help.
- Enhanced Scripting methods:
  - MLAB.showInformation()
  - MLAB.showWarning()
  - MLAB.showCritical()
- New Scripting methods:
  - MLAB.showQuestion()
  - MLAB.computerModelInfo()
  - MLAB.[macro|ml|inventor]Modules(), MLAB.all[Macro|ML|Inventor]Modules()
  - MLABFileManager.moveFileToTrash()
- Python binding
  - Fixed writing of bad .pyc files when .py files have a syntax error.
  - Fixed auto-reloading of modules when a python file was changed after it had a syntax error.
  - Python errors are now printed with the courier font so allow locating the ^ marker in syntax errors.
  - If a MDL command is connected to a Python function/method, the Python function may now have less arguments than the command provides. Surplus arguments are ignored.
- MDL
  - TextView now supports showLineNumbers = BOOL and syntaxHighlighing = (Python|QSA|MDL|GLSL).
  - LineEdit: new command lostFocusCommand.
- SDK
  - Module wizard for ML modules improved with many new features and tooltips with references into detail documentation of *ML Guide*.
  - Now the *Toolbox Reference* uses improved doxygen layout, many classes and files have better, fixed or completed documentation and alphabetical class index.

Core Revision

- Glut removal: *Glut* was only provided as a prebuild binary for the windows platform. Options would have been to provide a 64bit version of *glut* or to remove it from *netsrc*. The second choice has been selected (*Glut* was used for font rendering only, but libraries like *ftgl* can handle that much better.).
  - Removed *glut* from *netsrc*.
  - Added a bridge to MLOpenGL (see mlGL2DFont.{h,cpp}).

  - Examples of usage can be found in MLMiscModulesInv (see mlDiagram2D.{h,cpp}) and SoView2D (extends the GL2DFont class toward text rendering and usage of unicode strings).

- GL and GLU removal: Both libs should be available on any recent system (as far as it is known, exceptions have been Win95, Win98 and WinME).

- MLSystemInfo: Inlined assembly is not supported by the VC8-64 compiler. Therefore gathering system information has to be done with outlined assembly on this platform. This is done inside this new project that returns information on available extensions (like MMX, SSE, 3DNow, ...), number of virtual processors, cache sizes and other CPU info.

Common Enhancements / Fixes (Modules)

- WEM library reworked: a WEM is now actually a list of WEMPatches of different type (WEMTrianglePatch, WEMQuadPatch or WEMPolygonPatch).

  - Split the WEM library into a core part and into a modules part (MLWEM and MLWEMModules).

  - Removed: WEMBrowser, WEMClip, WEMMerge

  - Renamed: WEMCollapseEdges -> WEMReducePolygons

  - Renamed: WEMSplitEdges -> WEMSubdivide

  - Renamed: WEMPurge -> WEMDemergePatches

  - Use WEMInfo instead of the removed WEMBoundingBox

  - Use BaseSwitch instead of the removed WEMSwitch

- CSO: fixes and new modules

  - Split the CSO library into a core part and into a modules part (MLCSO and MLCSOModules).

  - Reworked CSOConvertToImage: this module works now with a page-based concept and has a OnDemand mode.

  - New processors: CSOModifyProcessor and CSOBulgeProcessor, both to modify existing CSOs.

  - New processor: CSOLiveWireProcessor for a semi-automatic segmentation.

  - SoView2DCSOEditor displays closed and fully visible CSOs in a filled manner.

  - CSOIsoGenerator offers several filtering and smoothing options.

- SoGVRVolRen no longer supports legacy hardware LUTs (using EXT_paletted_texture) , which was last seen on NVidia GeForce3 cards (but rendering still works on these cards).

- SoGVRSecondaryLUT and SoGVRSecondaryVolume support a variable number of LUTs and additional octree volumes in the renderer.

- SoGVRTransformedVolume and SoGVRTransformedVolumeSettings allow so set additional volumes as 3D textures.

- SoGVRShaderDiagnosis allows to view the shaders which the GVR generates.

- SoShader: fixes and new modules

  - Shader modules now have GLSL syntax highlighting.
  - Added SoMLSampler1D and SoMLSamplerCubeMap.
- Some new modules have been introduced:
  - AccumulateImage: Accumulates images (MPRs) in one image.
  - PointCloudToMask: Converts a list of pairs of points and normals (surface) to a smoothed distance image or a 3D mask.
  - TextureFilter: Applies statistic texture filters to an image.
  - MarkerStat: Computes statistical information on a marker list.
  - DtfSkeletonization: 3D distance transformation by homotopic thinning, skeletonization (replaces module Skeletonization).
  - SoCameraAnimation: Rotates a camera around an Inventor scene in a smooth pan.
  - SoMLLUTChangeColor: Modifies output of SoMLLUT modules.
  - SoMLSampler1D, SoMLSamplerCubeMap: similar to SoMLSampler2D and SoMLSampler3D.
  - BaseBypass: Like the Bypass module just for Base inputs.
  - OSType: Provides basic operating system information.
  - SoBoundingBoxVis: Computes and visualizes an axis aligned bounding box for an arbitrary Open Inventor scene.
  - SoVascularSystem: visualizes vascular systems (vessel graphs) generated by DtfSkeletonization.
  - DRR: module for computing a 2D digitally reconstructed radiograph image out of a 3D CT-image.
- ColorModelConvert: Now supports color models LAB and XYZ.
- CoreControl: Has been extended.
- DistFromXMarkerList: Added calculation of maximum distance.
- EuclideanDTF: Has been re-implemented.
- FuzzyCluster: Has been re-worked.
- HistogramParameters: Calculates a second user defined quantile.
- ImgLoad: automatic load can be suppressed.
- ImgPropConvert: Voxel Size and World Matrix are now kept consistent even with no input image connected.
- MergeRegions:
  - Fixed a bug that could cause a 1-voxel translation of output regions created as intersections or unions of input images for non-diagonal world matrices.
  - Fixed a potential problem in output world matrix creation for merging regions in color images.

    - Improved UI.
- MPR, MPRPath: Got new visualization parameters.
- Reformat: Added a field for the Euler angles.
- RegionGrowing: Added 4D capabilities.
- Sobel3D: Fields have been renamed.
- StringUtils: New mode ToASCII added for conversion of character values to ASCII numbers.
- Switch: Now up to 25 inputs are available.
- So3DXMarker: Has two drawing modes now: fast or correct.
- SoDepthPeelRenderer: Has been reworked and should produce better results.
- A lot of work was done on SoView2D and its extension modules, some changes are:
    - All help files and example networks have been reworked.
    - SoView2D: Added field "sliceZoomSynchronized" which updates the slice-origin when zooming. This allows to keep the viewing center in the middle of the view when updating the sliceZoom via a field synchronization.
    - SoView2D: Added field "updateInventorEvent". If this is set to "true", the following inventor nodes will not react to events that have been handled by a SoView2DExtensions.
    - SoView2D: New field "plane" contains the plane of the current slice.
    - SoView2DAnnotation: Some fields introduced to get a reliable automatic font sizing.
    - SoView2DMarkerEditor: Added option "snapToSlice" to translate any moved marker to the current slice.
- BoolInt: Also returns inverted bool value.
- DicomTool: Can take name template for file names.
- MarkerListInspector: Lots of new options.
- There are also a lot of other minor changes which don't appear in this document.

Mac OS X Edition

- MeVisLab changes:
    - *MeVisLab does no longer run on PowerPC-based Macintosh computers. This release has been optimized to run on Macintosh computers with an Intel processor.*
    - Added Mac OS X 10.5 (Leopard) compatibility; MeVisLab 1.6 runs on Mac OS X 10.4.11 or 10.5.x
    - Added 'Restart With Current Networks' feature (File menu).
    - Improved handling of non-latin1 language settings.

- Added multiple instances support, now more than one MeVisLab application can be used at the same time.
- Added command-line support ('cmdline' license feature required).
- Added Mac OS X Spotlight metadata importer (MeVisLabDocuments.mdimporter) that allows indexing of uncrypted MeVisLab script (.script) and definition files (.def).
- Added Mac OS X Quick Look generator (MeVisLabDocuments.qlgenerator) for MeVisLab network files (.mlab) that enables previewing of MeVisLab networks. It requires networks to be saved by MeVisLab running on Mac OS X 10.5 (the preview gets added to the resource fork of the MeVisLab network file, thus does not affect loading and parsing of the network).
- Added Mac OS X Quick Look support for MeVisLab script (.script) and definition files (.def) via standard Quick Look generators for text on Mac OS X 10.5.
- Adapted shortcuts to be more consistent on Mac OS X.
- Autodetect amount of video RAM on the graphics card (used by the volume renderer and other GPU-intensive tasks).

- MeVisLab Mac OS X Edition User's Guide changes:
  - PowerPC sections removed
  - Spotlight importer plugin added
  - Quick Look integration added
  - Setup of User Projects Location added
  - Using the Module Wizards added
  - Creating an Xcode project from a .pro file updated
  - Debugging MeVisLab modules with Xcode added
  - Developing on Leopard for Tiger added
  - Obtaining Source Codes of MeVisLab Modules added
  - Integrating MeVisLab with OsiriX added
- MeVisLabDicomService changes:
  - Added Mac OS X 10.5 compatibility. There are separate installers for Mac OS X 10.4 (Tiger) and Mac

    OS X 10.5 (Leopard). Please choose the appropriate installer for your Mac OS X version.
- Module changes:
  - Added an experimental bridge between MeVisLab and OsiriX 3.0 (Advanced Mac OS X PACS Viewer) consisting of the OsiriXMeVisLabBridge plugin for OsiriX 3.0 and the MeVisLab module OsiriXBridge. This bridge enables OsiriX to send datasets directly to the MeVisLab imaging pipeline using a standard ImgLoad module (see the example network of the OsiriXBridge module). After processing the dataset, MeVisLab may send the modified data back to OsiriX. The plugin and the

module require Mac OS X 10.5 (Leopard). The source codes of both the plugin and the module are included in the MeVisLab public sources package

  - Added check for user interaction in volume renderer (GVR); rendering large volumes should be more responsive now.

- Scripting changes:

  - Added MLAB.macIsApplicationRunning(bundleId)

- SDK changes:

  - Added MacOSSupport SDK that wraps Mac OS X specific functionality in C++ classes. See the *MeVisLab Toolbox Reference*.

  - Added AppleScript xcode-prepare-debug.applescript to automatically setup an MeVisLab module Xcode project for debugging (located within the MeVisLab.app bundle in /Contents/SharedSupport/Scripts/).

Linux Edition

- All projects are now linked with the option '-z defs' in order to check all symbols at linking time. To turn this option off, add a CONFIG += NO_LIBRARY_DEPENDENCY_ERROR_CHECK to a project's .pro-file.

MeVisLab Public Sources Installer

- Sources of many projects added:

  - MLBase

  - MLBinaryTree

  - MLGeometry

  - SoLUTTools

  - SoMLSupport

  - SoFlash

  - SoShader

  - SoQtMeVis

  - SoTensorFieldVis

  - SoVirtualVolume

  - OsiriXMeVisLabBridge & MLOsiriXBridge

ML, mlLinearAlgebra, mlUtils and Image Processing For details about the following changes see the ML release notes in mlDoc.h.

- New version number 1.8.67.23.86 which includes binary and compile interface changes. Some source will require updates before they compile again.

- Fixed: day() function in DateTime class now returns the correct day instead of always 0.

- Many constants and macros (e.g. from mlLinearAlgebraDefs.h), ML and mlUtils without ML_-prefix renamed or even removed if they are available on common system.
- Many interfaces, parameters and internal structures changed for 64 port, activation of 64 bit warnings on VC8 and non WIN32 platforms.
- New and changed image processing properties in BaseOp class for bypassing, multithreading.
- Revision of basic multithreading support in mlUtils, new class Mutex introduced which replaced mlCriticalSection.
- File system interface in mlUtils changed for 64 bit fixes and new functions added.
- Partial refactoring and revision of header files of project MLBase. Some stuff now located in its own header files such as CurveData, CurveList (from mlDiagramData.h), ParserBase, BaseItemParser (from mlListParser.h).
- Access to id of BaseItem only via functions from now on.
- New macros ML_PRINT_INFO and ML_PRINT_INFORMATION.
- Bugfixes in cursor movements and new functions for bounding box calculations and BitImage to BitImage copying in BitImage class.
- Fixed: Calculation of image statistics in ImageStat module when image has more than 3 dimensions.
- New error codes in mlTypeDefs.h and some MLErrorCodeDescriptions renamed or added, e.g MLOk string renamed to Ok.
- Runtime type system in mlUtils has new functions or paarmeters for class, dll, and inheritance dependencies.
- Fixed: MLImageFormatLoad cannot read nonwritable files.
- Many sections in *ML Guide* improved with new details and examples.
- Fixed: TextureFilter module crashing on non Win32 platforms.
- Fixed: KernelEditor creating wrong kernel strings on non Win32 platforms.
- Fixed: TimeCounter in mlUtils used/returned incorrect time units on different platforms. This lead to complete revision of the class interface. Interface was not clear enough such that a complete revision was necessary. It is not compile compatible any more and requires revision of the places of use. See header or SDK documentation for new interfaces.
- New module FloydSteinberg.
- New module ImageCompare.
- New Module CommandNotifyer.
- New module BitImageArithmetic, MLToBitImage, and BitImageToML in project BitImageTools as well as improved source box/region suppport and bug fixes.

ITK & VTK:

- Help pages revised and additional module specific information and limitations added.

- Many ITK and VTK modules now support correct conversion of world matrix and voxel size settings to itk/vtk orientation and voxel scaling. In some cases (e.g. ITK registration modules) this may change module behaviour slightly.
- Wrapper modules for itk/vtk classes have hyperlinks in parameter panels to doxygen class documentation.
- ITKVTKGenerator now supports the suppression of export symbols from user defined libraries and supports creation of platform dependent modules.
- Fixed minor bugs, typos in GUI, additional information.
- ITK:
  - Missing Sigma parameters and limitations fixed in itkGradientRecursiveGaussianImageFilter, itkGradientMagnitudeRecursiveGaussianImageFilter, itkLaplacianRecursiveGaussianImageFilter, itkHessianRecursiveGaussianImageFilter.
  - Fix in region calculation of itkLaplacianRecursiveGaussianImageFilter.
  - Fixed invalid connectors of modules:
    - itkCannySegmentationIF,
    - itkCurvesLevelSetIF,
    - itkGeodesicActiveContourLevelSetIF,
    - itkGeodesicActiveContourShapePriorIF,
    - itkLaplacianSegmentationIF,
    - itkShapeDetectionLevelSetIF,
    - itkThresholdSegmentationLevelSetIF, and
    - itkVectorThresholdSegmentationLevelSetImageFilter.
  - New itk classes wrapped as modules:
    - itkBinaryMorphologicalClosingImageFilter,
    - itkBinaryMorphologicalOpeningImageFilter,
    - itkBinaryProjectionImageFilter,
    - itkBinaryThresholdProjectionImageFilter,
    - itkMaximumProjectionImageFilter,
    - itkMeanProjectionImageFilter,
    - itkMedianProjectionImageFilter,
    - itkMinimumProjectionImageFilter,
    - itkMorphologicalWatershedFromMarkersImageFilter,
    - itkMorphologicalWatershedImageFilter,

    - itkNarrowBandCurvesLevelSetImageFilter,
    - itkNarrowBandThresholdSegmentationLevelSetImageFilter,
    - itkRegionalMaximaImageFilter,
    - itkRegionalMinimaImageFilter,
    - itkStandardDeviationProjectionImageFilter,
    - itkSumProjectionImageFilter,
    - itkValuedRegionalMaximaImageFilter, and
    - itkValuedRegionalMinimaImageFilter.
- VTK:
    - Pop up window of vtk modules suppressed; errors are redirected to MeVisLab console and ML error manager now.
    - 56 new vtk classes wrapped, 51 useless classes removed. See the *module comparison* for details.
    - vtkInputInfos shows more information about connected input classes.

Version 1.5.2 (2007-11-09)

MeVisLab Core and IDE:

- Fixed a bug causing screenshots to capture the wrong display area on Windows and Linux systems
- Fixed uncheckable MLABPopupMenuItems
- Fixed handling of html file urls with spaces
- *Mac OS X Platform:*
    - *The 1.5.x editions are the last releases for the PowerPC-based Macintosh computers. The next major revision will be only available for Macintosh computers with an Intel processor.*
    - Fixed crash occuring when closing MeVisLab with image connector preview on
    - Fixed placing of About and SystemInfo windows in panel child mode
    - Removed obsolete 'Default FileDialog path' field in preferences panel
    - Improved creation of Xcode 3 projects from *.pro files on Mac OS X 10.5 (*Leopard is not officially supported yet*)

MeVisLab MDL/Scripting:

- Fixed MLABGraphics/MFLImageIOPlugin bug: bmp files are written correctly now
- Fixed preferences file parsing: system and debug/release variables are now defined before reading prefs
- Fixed usage of QWidget.setVisible()

- Added operation system version number variable for Mac OS X (MACOS_10_4 or MACOS_10_5)
- Fixed signing of Python files for Mac OS X and Linux
- Fixed shouldCloseCommand, which was not able to prevent window closing

Core Libraries:

- DICOM:
  - ImgSave: Fixed memory leak saving DICOM slices
  - ImgSave: Fixed swapping of voxel size x/y in stored DICOM images
  - eatDicom: Fixed number format in output information with the -rules option
- 2D Viewing:
  - Fixed a potential memory leak in SoView2D if in mosaic mode and a slice step other than 1 is used
- Volume Rendering:
  - Fixed crash in GVRVolumeSave when a none DICOM file was attached and automatic filename was enabled
  - Fixed bug in SoGVRVolRen occuring with color dimension c==2 was given. Color == 2 is NOT supported by the GVR
  - Improved interaction with large volumes by adding render abort on interaction for Mac OS X
- CSO:
  - Fixed memory leaks
  - Fixed handling of Group and List rules
  - Fixes in the CSOMath class
  - CSOConvertTo3DMask now reduces the amount of considered points automatically, making the module much faster and stable
  - Fixes in the convertor modules (double computation, internal hash keying, output image min/max values)
  - Fixes in the SoView2DCSOEditor: editing of non-parallel CSOs (relative to the current viewer's plane) and moving a CSO while editing
  - CSOIsoProcessor: Enabled further editing of existing CSOs
- Object Manager:
  - Fixed crash in ObjLoader occuring when 'use loader with name' is non-empty and does not refer to a valid loader name

Modules:

- MLOffscreenRender works better now, fields for transparency and stencil buffer added
- Diagram2D: Fixed occasional crash on disconnecting input object

- MergeLists: Fixed a bug that lead to a crash under Windows Visual Studio 2005 (VC8)
- SoLUTEDitor: Fixed a bug that caused the module to hang if a new range with a min and max value both outside the original range was set
- JointHist: Fixed a bug that caused an invalid diagram output of the histogram of the first image if its bin size was greater/equal than of the histogram of the second input image
- ImageStat: Fixed invalid bounding box calculations on images with c-, t-, or u-extent > 1
- Draw3D: Fixed incorrect 2D and 3D fill operations
- MLImageFormatSave: Missing documentation of compression options and their additional parameters added

Add-On's:

- ITK: Many itkLevelSetImageFilters with invalid function connectors have been fixed (itkCannySegmentation-, itkCurves-, itkGeodesicActiveContour-, itkGeodesicActiveContourShapePrior-, itkLaplacianSegmentation-, itkShapeDetection-, itkThresholdSegmentation-, itkVectorThresholdSegmentationLevelSetImageFilter),

Version 1.5.1 (2007-08-31)

MeVisLab IDE:

- Fixed inconsistencies when pressing the up or down key in the quick module search.
- Fixed editing of list view items that were not added initially.
- Fixed a bug when module windows in an unregistered MeVisLab were not expanding.

MeVisLab MDL/Scripting:

- MDL validator did not include the LineEdit's lostFocusCommand.
- An unnamed LineEdit control now shows a blinking cursor.
- Fixed a bug in the ButtonBar: It had always a minimum size, which was not a desired behavior.
- Fixed the shell script template MLModule.sh, which is used by the MLModuleWizard.

Core Libraries:

- Fixed a bug in eatDicom where number to string conversion resulted in an endless loop.
- Fixed eatDicom problems related to output paths with umlauts and non-latin1 encoded input files.

Modules:

- HistogramPeakEstimate: crash if the given VOI was empty.
- LoadBase: crash if an empty list was loaded.
- CSOConvertToImage: crash if CSOs were to be converted to different time points.
- SoTensorFieldVis: wrong colors if a color image and a vec3/vecf3/mat3 image was attached.

- RasterFunction: corrected a half-voxel issue, so that primitives with a scale value of 1 are correctly rastered now.
- RasterFunction: a scale value of 0 now leads to an empty image and no matrix errors are printed out anymore.
- StringUtils: case-insensitive string comparsion always returned 'EQUAL'.
- MPR in ForceUpright mode now keeps the size of the field of view.
- Added a field Time Point to the macro View3D.
- Changed the evaluation of the parameter depthVisibility on the SoView2DMarkerEditor.
- Fixed export of time point information in ImgSave and DicomTool modules
- MLABMoviePlayerControl: the loop option did not control the looping of the player (Windows).

Mac OS X Platform: **Note:** The 1.5.x editions will be last releases for the PowerPC-based Macintosh computers. The next major revision will only be available for Macintosh computers with an Intel processor.

- GUI inconsistencies and MeVisLab style support have been fixed.
- Screenshots of OpenGL renderings are now fully supported.
- Movie generation within MeVisLab has been added using Apple's Quicktime.
- Better font size match for certain GUI controls.
- A dialog misplacement of the 'save as' dialog in the screenshot gallery has been fixed.
- Frequent authorization requests during DicomService installation have been fixed.
- The 'storescp' zombie when terminating the DicomService is fixed.
- More robust font management for SoView2D.

Version 1.5 (2007-05-21)

Installation:

- New platforms supported:
  - Windows with VC7 or VC8 development environments
  - Mac OS X
    - Minimum system requirements:
      - Macintosh computer with PowerPC G4, G5, or Intel Core processor; 1 GB RAM
      - ATI Radeon 9600 or NVIDIA GeForce FX 5200, 1280x800 screen resolution
      - 2 GB of available hard disk space
      - Mac OS X 10.4.8

    - Apple Xcode 2.4 or later for module development
  - Recommended system:
    - Mac Pro, 4 GB RAM
    - ATI Radeon X1900, 1920x1200 screen resolution
  - **Note:** The main platform supported by MeVisLab is still the Windows VC6 environment. The Windows VC7/8 and Mac OS X versions are offered as beta versions. The Mac OS X PowerPC version is completely unsupported and only provided for non-commercial installations.
- The MeVisLab 1.5 settings are partly incompatible with the MeVisLab 1.4 settings, especially the user defined IDE layouts will be lost.

MeVisLab Core:

- Major revisions in all core libraries
  - Migrated from application framework Qt3 to Qt4
  - Open Inventor-Qt-binding SoQt (Coin3D) has been replaced by SoQtMeVis (port from SoXt to Qt4). The modified sources and additional documentation are available for download at www.mevislab.de/inventor.
  - All third-party libraries have been updated to current versions.
- New PythonQt binding for Qt4 (see also pythonqt.sourceforge.net)
  - Update to Python 2.5
  - Support for Python remote debugging (see MeVisLab Scripting Reference: Python: Section on Debugging)
  - New auto-completion feature in Python scripting console
  - The new Python binding changes the interface to several QVariant types, including QSize, QRect, QPoint, QDate, QDateTime, QTime. Where these classes are used within Python scripting, the property access typically has to be changed into a method call (e.g. instead of size.height user size.height()).
  - New builtin help() function which can be called with any Python or MeVisLab Object as its argument

MeVisLab IDE:

- New output inspectors for marker lists and CSO contour objects (see below), improved existing inspectors
- New tab bar for switching between open network windows
- New command to search for a module in the current network (Ctrl+F)
- Plain (un-fancy) network rendering mode has been removed.

MeVisLab MDL/Scripting:

- The TabView control now supports "mode = left" and "mode = right" alignment of Tabs.

- The Splitter control now has "color" and "shadow" tags.
- The Box control has a "boxCheckedField" tag which can be used to enable/disable a Box with a checkbox.
- The ColorEdit control has an "edit" tag that can be used to disable editing of the color.
- The Field control has a new tag "useSheet" to control the use of "sheets" for dialog displays (only used on Mac OS X).

Open Inventor:

- Open Inventor is now part of the MeVisLab SDK, no separate installation is needed.
- New getName() method available in the C++ code of an Inventor node, returns the module's instance name
- Improved font handling in the SoFont and SoFontStyle modules:
  - On Windows, a font file with the specified font name and one of the extensions "", ".ttf", ".pfa", ".pfb", ".otf" is searched in the Windows font directory.
  - On Linux, a font file with the specified font name and one of the extensions "", ".ttf", ".pfa", ".pfb", ".otf" is searched in /usr/share/data/fonts or in the directory pointed to by the FL_FONT_PATH environment variable.
  - On Mac OS X, fonts are loaded using the Apple Type Services. A font can be specified by its Postscript font name (e.g. 'LucidaGrande-Bold') or its regular font name (e.g. 'Lucida Grande Bold').
- New input device drivers
  - Via SoCustomExaminerViewer: SoQtSpacemouse (Windows + Mac OS X only), SoQtPowerMate (Mac OS X only), SoQtAppleRemote (Mac OS X only) - device drivers for the 3Dconnexion Space Traveler 6DOF input device, the Griffin PowerMate input device, and the Apple Remote Control device
  - SoSpacemouseAccess, SoPowerMateAccess, SoAppleRemoteAccess - provide access to the input devices (buttons, sliders etc.)

ML and Related Libraries:

- New support for numeric min/max/epsilon handling in mlUtils
- Fields vec2, vec3, vec4, SubImgBoxf and long double use correct precisions in string representation now.
- Quaternion support added to mlLinearAlgebra and as a new voxel type in MLTypeExtensions
- New support for extensible image properties (classes ImagePropertyExtension, ImagePropertyExtensionContainer). This mechanism is used by the new representation of DICOM image properties (class MLDicomTreeImagePropertyExtension, see below).
- DataCompressor classes and factory added, as well as specializations for BZip2 and ZLib data compressors. User implemented compressors are supported.
- New functions and platform independent support for very large files even on 32 bit systems (mlFileSystem in mlUtils)

- Several new error codes and utility functions
- For details see ML Library Reference.

Other Core Libraries:

- DICOM:
  - Access to DICOM image properties is now available through the class MLDicomTreeImagePropertyExtension as part of the extensible image properties framework.
  - DICOM data is represented using the new DCMTree library, including support for DICOM sequences.
  - The DICOM import in MeVisLab now preserves frame-specific DICOM data and provides them through the StructuredMF interface in the DCMTree library. The new module DicomFrameSelect can be used to access frame specific DICOM tags.
  - The old DicomTags library is no longer supported.
- New MLImageFormat package with dedicated ML image file format, supporting 6D, compression, extended voxel types, DICOM information, user tags, files > 4GB, and image file caching. C++ interfaces are included in the SDK.
- New Contour Segmentation Object (CSO) library for generating, visualizing and processing contours, with C++ interface and several new modules:
  - CSOAffineTransformation, CSOConvertTo3DMask, CSOConvertToImage, CSOFilter, CSOInfo, CSOIsoGenerator, CSOLoad, CSOManager, CSOSave, LocalCSOLoad
  - CSOFreehandProcessor, CSOIsoProcessor, CSOPrimitiveProcessor
  - SoCSO3DVis, SoView2DCSOEditor
- New SoShader framework providing a framework to use the OpenGL Shading Language (GLSL) in MeVisLab networks.
  - Example source available for the SoGoochShader module
  - New modules: SoCheckShaderSupport, SoClearShaderState, SoVertexShader, SoFragmentShader, SoShaderProgram, SoShaderParameter1f, SoShaderParameter2f, SoShaderParameter3f, SoShaderParameter4f, SoShaderParameter1i, SoShaderParameter2i, SoShaderParameter3i, SoShaderParameterColor, SoShaderParameterMatrix, SoShaderParameterMLImageProps, SoShaderParameterMLImageSize, SoMultiplePass, SoGLRenderState, SoMLSampler2D, SoMLSampler3D, SoFramebufferSampler, SoMultiPassFramebufferSampler, SoInheritedFramebufferSampler, SoGoochShader
- 2D Viewing:
  - SoView2DMarkerEditor: The depth visibility parameter is now used correctly for picking and visualizing.
  - SoView2D has options to automatically unzoom on input image change and to disable writing to the z buffer in 3D renderings.
  - SoMouseGrabber and all View2D extensions support changing the cursor shape.
  - SoView2DBorder/Highlight can use the information of SoFocus to draw a highlighted border.

  - SoView2DOverlay supports a different data filtering instead of inheriting the SoView2D settings, see inheritFilterMode and filterMode fields.
  - SoView2DOverlay can draw check-tiles, see checkerTileSize, isCheckerTiling, areCheckerTilesInverted fields.
  - SoView2DTransRot has been modified to allow full mouse configuration instead of fixed settings, e.g. for MPR interaction.
  - SynchroView2D has been reimplemented.
  - New module GVROrthoView2D is a fast alternative to the SoView2DOrthoView by using the SoGVRVolRen module to render the images.
- Volume Rendering:
  - New module SoLUTEditor2D allows to create 2D LUT's that can be used with the SoGVRVolRen module and tag volumes.
  - SoGVRVolRen now uses OpenGL Shading Language shaders when available and supports on-the-fly gradients (set gradientQuality field to "Highest").
  - SoGVRVolRen has a new "filterVolumeData" mode that supports Nearest, FilterLinear, FilterLinearPreClass and an experimental FilterCubic, the "classificationMode" has been removed.
  - SoGVRQualitySettings has a flag "forceHighQualityRendering" that forces rending in the highest possible quality (e.g. for offscreen rendering).
  - GVRVolumeLoad now has a Close button to close the opened file.
  - GVRImageToVolume, GVRVolumeSave now support a "padSize" field to set the padding of the Octree (currently PAD1, PAD3 and AUTO), PAD3 is needed for on-the-fly gradients.
- WEM:
  - New module IsoSurface allows to easily create WEM iso surfaces.
  - Added dynamic user value vectors to the WEMNodes that can be used in networks (internally in modules) and for LUT control.
  - WEMImageData module supports trilinear interpolation and RGB images, setting of LUT range is fixed.
  - WEMIsoSurface module optionally generates quad meshes (in addition to triangle meshes) and supports using only the maximum iso value.
  - WEMModify module facilitates rotation.
  - WEMLoad/WEMSave support binary and ASCII STL formats.
  - WEMSmooth module reworked, offers Laplacian surface smoothing and normal smoothing. Both smoothing algorithms can be used in serial.
  - WEMSurfaceDistance contain new fields for the min/max normalized distances for the closest points positions.
  - SoWEMRenderer module supports face sorting for correct transparencies without a SoDepthPeelRenderer.

- New adaptors to VTK

Modules:

- Image I/O, DICOM
  - New modules BitImageLoad, BitImageSave: Saving/Loading of bit images from/to base objects or ML images
  - ImgLoad/Save support BMP format.
  - ImgLoad now supports loading raw images of datatype "double".
  - ImgSave allows to select subsets of DICOM tags to be discarded (per-frame tags, private tags, ...).
  - DicomLUT now supports VOI LUT's and non-linear modality LUT's.
  - DicomRescale problems for slope!=1 fixed.
- Image Transformations:
  - New module SubImageStat allows to do statistics/projections on selectable dimensions of the input image.
  - New module MPRLight provides functionality of MPR without Open Inventor features.
  - New module OrthoTriplePlaneViewer displays an image in oblique orthogonal 2D views.
  - New module TransformEdit for numerical, incremental and manual editing of transformations.
  - AffineTransformation3d has new modes selectable via a new transformationMode field.
  - EuclideanDTF uses an improved algorithm which is two times faster.
  - MPR,MPRPath: drawImageOn can be used to disable image drawing.
- Visualization and Interaction:
  - New module SoFocus allows to handle focus changes on viewers.
  - New module SoView2DDeformationGridView displays a grid showing a vector field deformation.
  - New module OffscreenRenderer allows to render Inventor Scenes into images using offscreen render buffers.
  - SoExaminerViewer has a new field "mouseInteraction" that allows complete mouse interaction re-mapping.
  - SoLUTEditor now has fields "window" and "level" to allow interactive windowing of the current LUT, a field "currentIndex" to view/edit the current point's index, and an option "newRangeMode" to either clip or scale the LUT on range change.
  - SoFlash has fields for a target color and a color output.
  - SoDepthPeelRenderer has a new "sceneDepthCompare" mode that renders the depth peel layers with correct depth layers (to include it with other renderings in the depth buffer).
  - VoxelizeInvScene now allows to voxelize triangles and lines in an anti-aliased manner. Thickness can be specified in either voxels or world units.

  - SoTensorFieldVis module now accommodates to vector data automatically.
  - SoRLAxis: Bug fix for negative annotation range or negative origin
- Miscellaneous:
  - New arithmetic modules: Arithmetic0 (scalar and vectors), BoolArithmetic (boolean values), MatrixArithmetic (matrices)
  - New module StringUtils provides string helper functionality (e.g. concatenation).
  - New module FieldBypass allows to enable/disable propagation of field notifications.
  - New module RasterMacro offers test images of geometry primitives.
  - New module TestPrinting shows how to create PDFs from rich-text using Python or JavaScript.
  - ImageIteratorStart/End now support file caching.
  - PrintCurves was improved.

Add-On's:

- VTK:
  - Update to version 5.1 of April 16th, 2007
  - New converter modules between WEM, ML, points, arrays lookup tables and vtk
  - New VTKQtRenderWindow and VTKQtRenderWindowInteractor
  - Hiding of multi inputs controllable in panels to reduce amount of connectors
  - Module inspector for vtk added
  - About 260 additional modules from wrapping process of vtk classes, including many new readers and writers.
  - Bug fixes related to non-scalar voxel types and voxel spacing in ML/VTK conversion. Fixed display of large module panels.
  - Some modules removed that caused problems due to unresolvable symbols, platform dependencies or other issues
  - For detailed release notes see any vtk module help.
- ITK:
  - Update to version 3.3 of April 16th, 2007
  - Four modules removed, 22 functions and 14 filters/modules added
  - For detailed release notes see "Technical Notes" in any itk module help.
- New Wrapper code generator module ITKVTKGenerator for itk and vtk.
- MeVisLab Public Sources – Sources of several projects added:
  - misc/MLDataCompressors/MLDataCompressor

- misc/MLDataCompressors/MLBZip2DataCompressor
- misc/MLDataCompressors/ZLibDataCompressor
- misc/MLDicomTreeImagePropertyExtension
- std/ML/MLImageFormat
- std/ML/MLImageIterator
- std/ML/MLBitImageTools
- std/ML/MLTypeExtensions
- std/AddOns/VTK/VTKAdapters
- std/AddOns/VTK/WEMVTKAdapters

Known issues:

- The SoGVRVolRen module creates border artifacts when rendering in 2D texturing legacy mode (this should only be an issue on very old graphics boards or older notebooks).
- VC7 and VC8 versions of MeVisLab do not contain the msvc*d.dll debug DLLs, because the Microsoft EULA forbids deploying these files.
- In the Mac OS X version of MeVisLab, "Restart with current network" is not and probably never will be supported.

Version 1.4 (2006-05-31)

MeVisLab Core:

- Fixes in macro reloading process and several other bug fixes

MeVisLab IDE:

- Improved output inspector:
  - In addition to ML image and Open Inventor inspectors, specialized inspectors for different MLBase datatypes have been added.
  - New inspectors for LUT, CurveData, WEM and GVRVolumeData objects
  - ML image and Open Inventor inspectors have been improved.
  - Users can add own inspectors for MLBase derived datatypes
  - Inspector settings (e.g. details on/off toggle state) are persistent.
- Preferences now contains a "Limits" panel that allows to set the available resources, e.g. texture memory and volume data cache sizes.
- Fixed bug with editing instance name in module inspector
- Improved module reference pages

MeVisLab MDL/Scripting:

- MLABModule::remove() and MLABNetwork::removeModule() are now non-undoable (which is the desired behaviour in most cases), new methods MLABModule::removeUndoable() and MLABNetwork::removeModuleUndoable() remove a module undoable (which is what the former methods did in the previous version).
- MLABField::connectFrom() and MLABModule::connectField() now check if connection is possible and return true on success.
- New MDL tag tabEnabled = BOOL for items of TabView controls. In previous versions, tab view items could only be enabled/disabled by scripting commands.

Open Inventor:

- SoQtSpacemouse: Driver for Spacemouse/Spacetraveller device (SoQtSpacemouse/SoQtSpacemouse.h in InventorWrapper)

ML:

- Minor changes: Warnings removed, bug fixes in (T)SubImage and DateTime classes, some support macros for extended data types added. For details see ML Library Reference.

Other Core Libraries:

- Lookup Tables (MLLUT):
  - New modules LUTCompose, DicomLUT, TableLUT
  - Fixes in handling of invalid LUTs
  - Added sigmoid LUT primitive
  - Added interpolation mode "constant" to LUTFLinear/SoLUTEditor
- Volume Rendering (GVR), 2D/3D Visualization:
  - New module SoDepthPeelRenderer implements order-independent transparency of unsorted meshes
  - New module SoGVRDepthPeel allows to mix transparent geometry into the SoGVRVolRen volume renderer
  - SoGVRVolRen and View3D have new "gradientQuality" field specifying the quality of the gradients in illumination mode
  - SoView2DScene has been improved and now allows to render a 3D scene with a given slab size onto any SoView2D slice
  - SoGVRSlabHint in connection with SoGVRVolRen allows the VolumeRenderer to render slabs on any SoView2D/SoOrthoView2D, see SoGVRSlabHint example network
  - Fixed bug in updating View2D extensions
- Winged-Edge Meshes (MLWEM/SoWEM):
  - New modules:
    - SoWEMDiagnosis: Handling and visualization of errors

    - SoWEMSynchroViewer: Synchronized views of WEMs
    - WEMImageData: Set image data to the WEM nodes' LUT values
  - Enhancements:
    - SoWEMRenderer: More detailed impacts on visualization parameters, optional display of statistical data on the WEM
    - WEMBoundingBox: Added object-aligned bounding box computation and visualization
    - WEMClip Added control of the output parts of the clipped WEM
    - WEMCollapseEdges: Enhanced algorithm, edges are collapsed based on angular information now, produces better results due to local smoothing
    - WEMInfo: Added statistical output compatile to Diagram2D
    - WEMInitialize: Added erroneous test surface
    - WEMLoad/WEMSave: Added a proprietary binary .wem format for fast loading and saving
    - WEMPerformance: Redesigned module panel and added a module filter
    - WEMPurge: Now shows a list of all connected components that can be removed and permits interactive choice, also added an automatic selection option
    - WEMSurfaceDistance: Faster computation
  - Bug fixes:
    - WEMIsoSurface: Fixed a bug with the lutValue interpolation
    - WEMSmooth: Fixed Displacement Correction algorithm
  - General features:
    - Delayed computation of edges for a faster loading/saving/display
    - Several new toolbox classes and basic data structures
- Object Manager:
  - Fixed a rare event notification bug due to recursive notification

Modules:

- New ML and Open Inventor modules:
  - Blinking material: SoFlash
  - Segmentation with automatic threshold selection: OtsuThreshold
- Enhancements:
  - New modes in TypeArithmetic1 and TypeArithmetic2 for processing vector and matrix images
  - Performance optimization of BoundingBox (for integer images), RegionGrowing (output generation), ConnectComp (completely new, more efficient algorithm)

  - VoxelizeInvScene is more time efficient now and offers a filling of closed surfaces (scan lines).
  - ContourManager: Modification mode for freehand contours, drawing of label string at contours, copy and paste functionality, group ID for 3D objects, binary mask can be restricted to contours with an individual group ID.
  - JointHist: New feature to compute a 2D histogram from a single image (i.e. image value vs. x,y,z,c,t or u-coordinate), improved module panel, AutoApply bug resolved.
- Bug Fixes:
  - DicomTagModify: AutoApply bug resolved
  - OrthoSwapFlip/OrthoReformat3: Fixed bug with certain oblique voxel coordinate systems
  - ImgSave: Correctly handle suppression of special tags in case the DICOM tag list is not associated to a file

Add-On's:

- VTK:
  - Upgrade to VTK 5.0: About 80 new modules, some modules removed
  - A large number of bug fixes (especially in writer modules)
  - New module VTKInputInfo to retrieve class parameters (e.g. scalar range)
  - New module VTKToMLImage to embed VTK image processing in ML image pipelines and vice versa
  - Module MLBaseToVTKPoints provides conversion of Base points, vectors etc. to VTK PolyData.
  - For details see main page of VTK module documentation.
- ITK:
  - Upgrade to ITK 2.7
  - A few new 2.7 modules added (itkMorphologicalGradient-, itkInvertIntensity- and itkModulusImagefilter)
  - Some filters with incorrect paging support have been changed to work globally (itkScalarConnectedComponent- and itk*Reconstruction*ImageFilters).
  - Metrics support optional mask images now.

Version 1.3 (2006-01-30)

MeVisLab Core:

- Support for user module HTML documentation and index creation
- Destructors of modules are now always called on exit of MeVisLab (to allow user cleanup of resources)
- Several bug fixes

MeVisLab IDE

- Automatic show/hide of open panels when active network is maximized (Panels->Hide Panels of Invisible Networks)
- New tab card "Related" in the FieldInspector for quick access to referenced modules and modules in the same genre/library
- New "Quick Add Module" feature: Typing with the focus on a network automatically activates the module quick search
- New menu items "Show Module in Network" and "Module Context Menu" in the context menu of Field controls and in the widget debug context menu
- Auto-save of untitled networks, optionally restored at restart of MeVisLab after a crash
  - (This is especially useful if MeVisLab is started in a debugger and the debugger kills the MeVisLab process.)
- Flexible drag-and-drop handling of files to networks allows users to extend file dropping with their own modules by file extension
- Several bug fixes

MeVisLab MDL/Scripting:

- Added initCommand tag to all GUI controls
- ColorEdit control has new mode = triangle to allow HSV inplace editing of colors
- New tags for MenuItem and SubMenu
- Added additional scripting access methods to various GUI controls

Open Inventor:

- Added a graphical material editor to the SoMaterial node
- See Open Inventor Changes

ML:

- Revision of mlLinearAlgebra, new classes mat2, mat5 and mat6.
- Several new vector and matrix voxel types
- Many new features in mlAPI and in classes Host, SubImg, TSubImg, BaseOp, PagedImg, Vector and SubImgBox
- Improved documentation
- (See detailed release notes in the ML library reference)

Other Core Libraries:

- Lookup Tables (MLLUT):
  - Support for 2D- and 3D-LUTs (especially useful for tagged volume rendering, see GVR)
  - New method LUTFunction::isValid() to check whether a LUT function can be rendered
  - New module SoLUTEditor for interactive specification of RGBA lookup tables

    - New LUT modules LUTChannelMap, LUTCombiner, LUTConcat, LUTBlend, LUTSelect
- Volume Rendering (GVR):
    - The GigaVoxelRender (GVR) library has been completely reorganized and rewritten
    - A lot of new features and performance optimizations, see the documentation of SoGVRVolRen
    - New image object "GVRVolume" representing an image as a multi-resolution octree
    - New modules to handle GVRVolumes: GVRImageToVolume, GVRVolumeToImage, GVRVolumeLoad, GVRVolumeSave
    - New GVR extension modules: SoGVRTagVolume, SoGVRGradientVolume, SoGVRIlluminationSettings, SoGVRSubVolumeSettings, SoGVRQualitySettings, SoGVRIncrementalUpdater
- Object Manager:
    - Improvements and bug fixes
- Kernel Filters (MLKernel):
    - Support for separable kernel filters
    - Support for multiple inputs and outputs
    - Support for different input and output types
    - New code examples for kernel classes

Modules:

- New ML modules:
    - New arithmetic and conversion modules for improved processing of vector- and matrix-valued images: TypeArithmetics1, TypeArithmetics2, TypeToScalars, TypeFromScalars, TypeComposer*, TypeDecomposer* (* = 8, 16, 32, 64)
    - Fast Fourier transform: FFT1D, FFT2D
    - Orthogonal min/max/sum/average projection: OrthoProjection
    - Geometry: ImageSeparator, InterleaveDimension, SwapDimensions, TransformEdit
    - Switch image input streams: Switch1toN
- New Open Inventor modules:
    - Display 3D geometry in a SoView2D viewer: SoView2DScene
    - Display vector fields in a SoView2D viewer: SoView2DVectorFieldView
    - Render into textures, fill rate reduced rendering: SoFramebufferRenderer, SoFramebufferTexture, SoCheckFramebufferObjectSupport
    - Load 2D textures from ML images, switch according to complexity requests, enable multi-texturing, etc: SoMLTexture2, SoComplexitySwitch, SoImagePlane, SoTextureUnit

    - Display orientation hints within a 3D scene: SoOrientationInset, SoOrientationModel
    - New SoTensorFieldVis for 3D tensor and vector visualization
    - New 3D viewer with new camera control buttons, customizable buttons and 3D input support: SoCustomExaminerViewer
- New library for dealing with surfaces represented by "winged-edge meshes" (WEMs), including several ML (MLWEM*) and Open Inventor (SoWEM*) modules for generating (iso-surface, load/save), processing (smoothing, primitive reduction, subdivision etc), visualizing (Open Inventor rendering in different modes with LUT support) and for an interactive modification of surfaces
    - Special thanks to Bart De Dobbelaer (KU Leuven, Belgium) for his great contribution to this library
- SmallImageInterface (with examples) for simple image processing examples in educational contexts

Add-On's:

- New download package "MeVisLabPublicSources.exe" with complete source codes of selected MeVisLab modules and libraries, considered as additional documentation to the MeVisLab developer
- ITK and VTK:
    - ITK Add-On 1.1:
        - Based on ITK 1.3 (Nov. 2005), several new modules
        - Includes registration framework
        - Several new example networks, e.g. from the ITK book
    - VTK Add-On "PreAlpha":
        - Several hundred VTK modules, some adaptors, and numerous example networks
    - Special thanks to Julien Jomier (CADDLab at UNC, North Carolina) for his great contributions to the code generators for ITK and VTK

Version 1.2 (2005-07-28)

MeVisLab Core:

- MeVisLab Core refactored
    - Performance of module destruction improved, large speedup when freeing networks/modules
- Improved handling of debug/release libs on Linux systems (with/without _d suffix, as on Windows)

MeVisLab IDE:

- Output Inspector: Improved formatting and layout

Modules:

- New OpenGL support using GLEW and MLOpenGL library
- New module SystemInfo showing details on the OS and OpenGL driver that MeVisLab is running on

- Diagram/curve data classes now available as MLBase objects in MLBase/mlDiagramData.h
- SoView2D and extensions:
  - Supports upto 16 bit hardware LUTs on cards that support ARB_fragment_program or OpenGL Shading Language
  - LUT support for RGB and RGBA images
  - Editing functionality in SoView2DPlane and MPR to allow editing an MPR plane in a 2D viewer interactively
  - New module SoView2DBorderHighlight to draw a border around a selected range of slices
- Image import/export:
  - Support for JPEG and PNG file formats
  - Support for files up to 4GB
- New module DicomRescale

Version 1.1 (2005-04-28)

MeVisLab Core:

- Full unicode support in:
  - Networks
  - MDL
  - Image, Dicom and Base I/O
- MDL:
  - Translation support
  - New MDL controls: EventFilter, Log, ContextMenu
  - PreloadDLL allows to load DLLs on MeVisLab startup (e.g. for adding new types to the ML)
  - Various new tags have been added to different controls
- New support for Python scripting language
- New scripting classes/methods:
  - New scripting API for LineEdit control
  - MLABProcess class for asynchronous process control
- GUI Rescaling improved

MeVisLab IDE:

- Improved Preferences dialog, includes settings controlled by global modules (error handling, tracing, debugging). The corresponding global modules are no longer visible.

- Added "Scripting Assistant" docking window
- Improved advanced search, now including "name+keywords", "common tags" and "seeAlso"
- Bug fixes and improvements in module wizards

Open Inventor:

(See Open Inventor release notes for a complete change log)

- New documentation
- New option to catch, report, and suppress runtime errors and to provide error tracing information
- UTF-8 support for path names
- Several bug fixes

ML:

(See detailed release notes in the ML library reference)

- Tracing functionality added to ML and to many modules. Now runtime catching, tracing and reporting runtime crashes is possible if the ML is configured to do so.
- Platform independent interfaces for unicode strings and file access
- Support for images larger than 4 Giga voxels, handled by several page-based modules
- MLint data type has changed from 32 to 64 bit, which results to 64 bit voxel indexing in several classes (Vector, SubImg, SubImgBox, VirtualVolume, ...).
- Some new extended data types for float vectors (vec2f, ..., vec32f)
- New MultiField classes for signed and unsigned 32 bit integers
- Various new features in classes TSubImg, ErrorOutput, VirtualVolume, and other toolbox classes, in TemplateSupport and API
- Many bug fixes
- Improved ISO and coding style compliance of ML, examples and modules
- Example programs and modules improved and extended

Modules:

- New DicomService module and Windows-NT service
- Image import/export:
  - Added "Save DICOM header file only" field in ImgSave for DICOM/TIFF images
  - Enabled LZW compression in ImgSave for TIFF images
  - Added "Allow overwrite" field in ImgSave
  - Added "Page Size Hint" field in ImgLoad to control page size of loaded image data

    - Fixed handling of DICOM position coordinates
- SoView2DMarkerEditor:
    - New field "selectionBoxesForCurrentOnly" to suppress manipulators of non-current vector markers
    - New field "selectiveDrawing"
    - Bug fix: markerSize of a connected StylePalette is used now
- New base classes and modules for lookup tables, see misc/MLLUT and ML/MLLUTTools
- Other new modules:
    - SoView2DVoxelView module for voxel value inspection (View2D extension)
    - Compose/DecomposeArray
    - SoSilhouette and SoEdgeEmphasize to draw the outer contour and inner feature lines
- New scripting helper functions:
    - assert(), see scripts/debugAssert.js
    - Convenience functions for creating HTML tables, see scripts/hypertextHelper.js
- Renamed GlobalStatBase to ImageStat
- Disabled currently unmaintained SoMeasurement modules, select "Deprecated SoMeasurement" group to re-enable
- Bug fixes and improvements in many modules:
    - MPR, Reformat, OrthoSwapFlip, OrthoReformat and OrthoView2D support large datasets via paging and virtual volume concepts
    - LocalMaxima: Min/max of the output image are set correctly now if the "Return source value" option is enabled
    - SmartMarker: Added delay to avoid resetting a smart marker immediately after delete on click
    - MorphologicalGradient: Fixed a bug that occurred when "threeDimensional" was on
    - IntervalThresh: Fixed bug where the output image was not invalidated on disconnected/invalid input
    - Added/enhanced drag-and-drop support for LoadBase, ImgLoad, OpenImage, FileDirectory, ObjLoader, MakeName
    - ModifyRegion: Fixed "make extensions dividable by" mode
    - ComposeBaseList now allows the specification of IDs for both input objects
    - ConnectComp: Fixed a few bugs, one of them concerning the largest cluster mode
    - LoadBase/SaveBase: Added keyboard shortcuts
    - Resample3D bugfix: Several bug fixes, new field "filterAlwaysTolerance" for compensation of rounding errors

- More fixes...

Version 1.0 (2004-08-18)

Core:

- Fixed bug in *.mlab persistence (strings which started with * could not be saved/restored)
- Fixed #ifnset bug in MDL parser
- various minor MeVisLab core fixes and improvements
- Improved multi-user installation
- added **UserLibraryPath** to Preferences and mevislab.prefs file, DLLs found in **UserModulePath** are copied into the **UserLibraryPath** if it is set, it is automatically added to the PATH/LD_LIBRARY_PATH

InventorWrapper:

- SoSFXVImage was renamed to SoSFMLImage, please adjust your projects to use **#include <SoSFMLImage.h>** and **SoSFMLImage** instead of SoSFXVImage
- InventorWrapper API and doxygen docs were cleaned and improved
- removed deprecated sources: SoDatatypes.h (SoMarkerList, SoVectorList)

Wizards:

- Fixed support for target paths which contain spaces
- Fixed minor bugs in Inventor Module Wizard

EatDicom:

- Fixed recursive data scan on Linux

Module changes:

- Fixed popupMenu of SoExaminerViewer
- Minor fixes and improvements to several modules

New Modules:

- AnonymizeMacro
- CacheView
- Checksum
- DicomReceiver
- ImageComposer
- TypeComposer
- TypeDecomposer

- RuntimeDump

New ObjectManager Modules: The ObjectManager is a new concept that can be used for application-wide sharing for datastructures and event handling. See the ObjMgr online help for details.

- ObjBaseTypeConnection
- ObjConstrainedIterator
- ObjDump
- ObjFieldConnection
- ObjInfo
- ObjInspector
- ObjLoader
- ObjLoaderDump
- ObjMgr
- ObjMgrEventClient
- ObjMgrMultiClient
- ObjMgrMultiClient2
- ObjVolume

Version 1.0b2 (2004-06-16)

Wizards:

- All wizards have been improved, the settings of a wizard can now be saved to a ".wiz" file, so that one can create templates for later modification/extension. Be careful: re-creating the module/project in the same place will overwrite existing files!
- New wizard for Inventor modules, allowing the creation of SoNodes and SoView2DExtensions

Module changes:

- Improvements to ContourManager, LiveWire
- Fixes to DICOM support
- Various diffusion filter modules have been merged to a single, optimized Diffusion module. The experimental ansitropic diffusion filter module AnisoDiffOp has been removed.
- Minor fixes and improvements to several modules

Core:

- New MDL tags Label:textFormat, ToolButton:textPosition, ToolButton:autoScale
- MDL tag CheckBox:command no longer supported, use a FieldListener instead
- @IF (ML_HELP)

- Added MLAddEventFilterCB() function to allow modules to process native window events
- Renamed MLMin(), MLMax() functions to MLDataTypeMin/Max(), resp.
- Added FieldContainer::removeField() method
- @ENDIF
- Fixed various MeVisLab core bugs
- Documentation improved, although still not perfect...

Version 1.0b1 (2004-05-18)

First public beta version of MeVisLab.