# Mamba2 versus Mamba
# Detailed description of changes

Version 2 of Mamba introduced a lot of changes compared to the previous versions. These changes were made for performance and easiness improvement. Therefore, the scripts written for Mamba 1 are not compatible with Mamba 2. This document explains the main differences between the two versions. We expect that this description will allow you to easily modify your scripts. The list of changes is quite long and we tried our best to not forget anything important.

For further details, see the Mamba User Manual and the Mamba Reference manuals.

The first important change concerns the general structure of the Mamba library. All the Python sources (modules *mamba.py*, *mambaDraw.py*, *mambaExtra.py*, *mambaDisplay.py* and package *mambaComposed*) in Mamba 1 have been gathered inside two packages, *mamba* and *mambaDisplay* in Mamba 2. Functions and operators have also been reorganized by families inside these packages. This allows to produce a more structured documentation. This change simplifies deeply your script writings as you don't have to worry about the module or package (*mamba* or *mambaComposed*) which contains your morphological 2D operators. Obviously, this modification makes old imports of Mamba incompatible.

The *mamba3D* package, previously available as an add-on, has been added to the main library (both C and Python code). The **image3DMb** class no longer inherits from **sequenceMb**. This class is now defined independently (**sequenceMb** has been kept for compatibility purposes but it is simply an alias of **image3DMb**).

The source new organisation also impacts display with the creation of the package *mambaDisplay* which contains both 2D and 3D displays. New displays were added. Display methods in **imageMb** and **image3DMb** have been shortened (**show**, **hide**, **update** instead of **showDisplay, hideDisplay**, etc.). Palettes and opacity methods were removed from these classes. See the User Manual to know how to use the various display options (palette, mode, zoom, etc.) either in interactive mode or inside a script. The 3D display methods were also strongly modified by these changes. A new display (Player) has been added. The interactive operators **dynamicThreshold**, **superpose** and **interactiveSegment** still exist but they are contained in the *extra.py* module of the *mambaDisplay* package. Import them with the ***from mambaDisplay.extra import **** command.

Many changes were applied to the Mamba operators (we hope the following descriptions are exhaustive…). Some operators were added,  others were augmented or modified.

- A **div** operator**,** performing a pixelwise division between two images has been added. It manages the division by zero issue, see the Mamba Python Reference Manual.
- The **label** operator now supports 8-bit and 32-bit images and its default behavior regarding label values has been changed (no more missing label values). Without any specification, the label values are now contiguous.
- The **convert** operator can be used with all image depths.
- The **close** and **open** operators were renamed respectively **closing** and **opening** to avoid conflicts with Python standard functions.
- The **hitOrMiss** operator now works directly with a **doubleStructuringElement** object making **binaryHMT** useless.
- The **loadRaw** methods were homogenized for **image3DMb** and **imageMb**.

- **copyBitPlane** now supports 32-bit to binary and reverse copies. Any bit plane of a 32-bit image (numbered from 0 to 31) can be inserted or extracted directly.
- C functions regarding neighbors have been modified to gain performance. The corresponding Python functions (**supNeighbor**, **infNeighbo**r, **diffNeighbor**) are also impacted. The new operators can now perform operations with more than two points. It results that a **supNeighbor** or **infNeighbor** operators are practically equivalent to a **dilation** or **erode** operators. This however changes the neighborhood coding in the operators. The directions used in the neighborhood must be encoded. See the User Manual for more information.
- Support for hierarchical algorithms (watershed and build) on 32-bit images was added in C with removal of the specific Python functions (namely **watershedSegment32**, **basinSegment32**, **hierarBuild32** and **hierarDualBuild32**). These operators are now accessible through the standard functions (**watershedSegment**, **basinSegment**, **hierarBuild** and **hierarDualBuild**).

Mamba is now compatible with Python 3. This may have impact on the behavior of some functions (**getDirections**, for example, is based upon the Python **range** operator which does not have the same behavior in Python 3).

Mamba will no longer be supported for Python 2.6. Mamba 2 needs **ttk** (themed Tk) for its display and this module is not natively available in Python 2.6.

Mamba uses the **Pillow** library instead of the **PIL** one. You may have to update to **Pillow** if you were a **PIL** user. Both libraries are normally equivalent in functionality except for the import mechanisms. See the **Pillow** library documentation.

The C code is now compiled in a specific library (*mamba.dll* in Windows or *mamba.so* in Linux) with its include files available separately making it possible to build applications with the core functions of Mamba more easily. This has an impact on the compilation process which now use **CMake**.

Mamba now supports Windows 64-bit thanks to available win64 packages in the **Pillow** library.

Vectorisation in the C code was changed (see *mambaApi_vector.h*). This change improves the performance.

The documentation was modified. The directory containing doc in source was renamed, a better use of **Doxygen** was made to generate the C API doc, 2D and 3D docs were merged. The examples were removed from the User Manual and gathered in a separate documentation. The mamba style has been changed. Conversely, some documents have been merged into the User Manual.

The new version also needed to reorganize the source directory, the examples, the tests of the library (see the documentation to know how to launch the tests).

The result of these changes is a faster and, we hope, more user-friendly library.

*November 2, 2015*