**MAGI-Dock: a PyMOL companion to Autodock Vina**

Jurgen Kaftalli

**Abstract**

**Summary**: Molecular docking simulation of small molecule drugs to macromolecules is a valuable tool in structural biology and medicinal chemistry research, also thanks to many available free resources. Like many open-source programs in the field, free docking software relies on a command-line interface, and steps requiring interaction with molecular structure make use of external graphics software, which are often limited in number and very specific. In addition, such features usually need a re-training of the staff, which may hamper the usage, especially in a company environment. Here we present MAGI-Dock, a graphical user interface that brings together the power of two of the most used free software for docking and graphics, Autodock Vina and PyMOL. MAGI-Dock is a free PyMOL plugin that assists the user along the docking workflow, focusing on docking box set-up.

**Availability and implementation:** MAGI-Dock is a free open source software available at [https://github.com/gjonwick/Docking- Box.git](https://github.com/gjonwick/Docking-%20Box.git)

**Contact:** jurgenkaftalli@hotmail.com

1. **Introduction**

Molecular docking simulation of small molecules to bio-macromolecules is a powerful tool for structure-based drug discovery that has reached widespread use also thanks to the many available resources in the free domain. In this field, AutoDock Vina is one of the fastest and most widely used open-source docking engines, providing flexible receptor, hydrated docking, support for macrocycle molecules, multiple scoring functions, etc. Moreover, simultaneous docking of multiple ligands and batch mode for virtual screening allows for exploiting large, customized virtual libraries of compounds and fragments (ZINC [ref]) and orphan drugs (DrugBank [ref]). Once the macromolecular target and the ligands to be screened are defined, the docking process is carried out in four steps: *(1)* the preparation of the target and ligands (with setting up of rotatable bonds for the latter) by conversion in a suitable format (PDBQT); *(2)* the setting up of a search box defining the binding site on the target, including the set-up of flexible sidechains; *(3)* running the docking simulation; *(4)* analysis of the result. The command-line nature of Autodock and the AutoDockTools [ref] companion package allows for automating the whole process in a high throughput fashion by, e.g. shell scripting [ref], except for the set-up of the search box on the target. This step is usually carried out once and requires user-supervised molecular graphics interaction with the target to fine-tune the box boundaries and the extent of flexibility of side chains forming the binding site. The support for such box drawing is limited to a few molecular graphics software [ref] or discontinued [ref seeliger]. Above all, a modern implementation is missing in one of the most widely used macromolecular visualization tools at present, PyMOL. Since PyMOL is written in Python, one of the most popular programming languages, it can easily be extended to Python plugins. Using this feature, we provide MAGI-Dock, a free PyMOL plugin for setting up ligands, search box and flexible sidechain and analyzing results.

1. **Features**MAGI-Dock allows users to run all the steps needed for a docking job. With the convenient interface PyMOL provides, users can run docking effortlessly without switching back and forth between PyMOL and other molecular modeling softwares. Moreover, such a feature allows users to use the molecular modeling power of PyMOL and run docking directly on customized structures.

The plugin is conveniently organized in tabs, each one managing a sequential step, described as follows:

**Receptors**  
The first tab deals with the generation of receptors. Users can select the PyMOL object, to be treated as the receptor and generate the corresponding *pdbqt* file. Once the receptor is ready, it's up to the user to add flexible residues. The receptor selected in the generated receptor list will also be automatically loaded into “memory” and be used for docking.

**Ligands**

The second tab deals with the preparation of ligands. Similar to the receptor tab, users can choose the selections representing the ligands, and click the *Add* button. This notifies the plugin about the ligands that are going to be prepared. In order to actually prepare the ligands, users have to select the ligands from the ligand list and click the *Generate*button - this will launch the preparation of all selected ligands. The successfully prepared ligands will appear on the prepared ligands list.

A recommended approach to prepare multiple ligands, is to first load them using PyMOL (especially if they are in .sdf format) and refresh the list in the ligands tab. Then, all the ligands can be prepared in one go, by just selecting all of them and clicking on the generate button.

**Docking**

Finally, in the third tab, users can run the docking job. The receptor on which docking will be run can be loaded into "memory" by clicking one of the generated receptors in the first tab as explained previously, while the ligands can be selected from the prepared ligands list in the Docking tab. Clicking on the *Run Docking Job* button will launch Vina and execute docking on the selected ligands and receptor.

Users can also optionally specify the exhaustiveness, number of poses, minimum RMSD, etc. - options which are useful to Vina.

**Box**  
The plugin also provides a convenient interface to generate the docking box. Users can either make a selection in the PyMOL viewport and by clicking the generate box button a box will automatically be created around the selection, or they can type in the object’s name and a box will be generated around the object. The box can be moved by scrolling over the coordinates and dimensions of the box. Additional options regarding the box include: an option which lets users specify the rate of movement (useful when sensitive movements of the box are needed), an option to hide/show the box, an option to fill the box (colors the planes of the box with different colors, while being transparent; very useful when dealing with box positioning), and finally an option to save the box.

**Config**

In the first tab users can specify the required configurations to run docking.

If the user is not working in a system where the tools can be loaded as modules, it is mandatory to specify the path to the python executable provided by MGL Tools installation, where all the AutoDock scripts are located, the vina executable path, as well as the config file containing the docking box parameters; specifying a working directory is also recommended.

1. **Software requirements and installation**The plugin is built using PyQt - one of the most popular GUI toolkits for Python - and its libraries are provided by the PyMOL package itself.

The plugin requires Vina and MGLTools to be installed in the system. Depending on the operating system, the path of the software should be specified by the users in the config tab of the plugin.

Note: Regarding the python executable in the MGLTools directory, while recommended, it may be omitted.

If the users will work on systems where the required tools can be loaded as modules (via the ***module load*** command) then no configuration is needed. However, the required tools must be loaded before starting PyMOL (this scenario will usually occur only when working with remote clusters; a guide is provided in supplementary materials).

*Installation guidelines*

In order for the plugin to use the AutoDock scripts, the environment should have already been set up for command line usage of AutoDock - the instructions are available at the MGLTools manual or the readme file that comes with the MGLTools installation.

A vina executable should also be available on the system - we suggest to name the vina executable as “vina”.

Finally, in PyMOL, click on Plugin → Plugin Manager → Install New Plugin → Choose file... and select the \_\_init\_\_.py file in the directory MAGI Dock was downloaded. The plugin should now appear in the Plugin submenu.

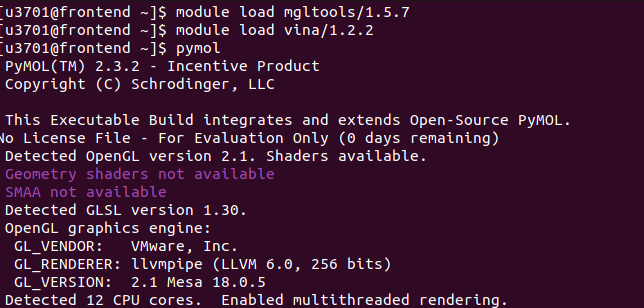
The plugin is tested on Windows 10, as well as in Ubuntu and CentOS linux distributions.

1. **Conclusions**

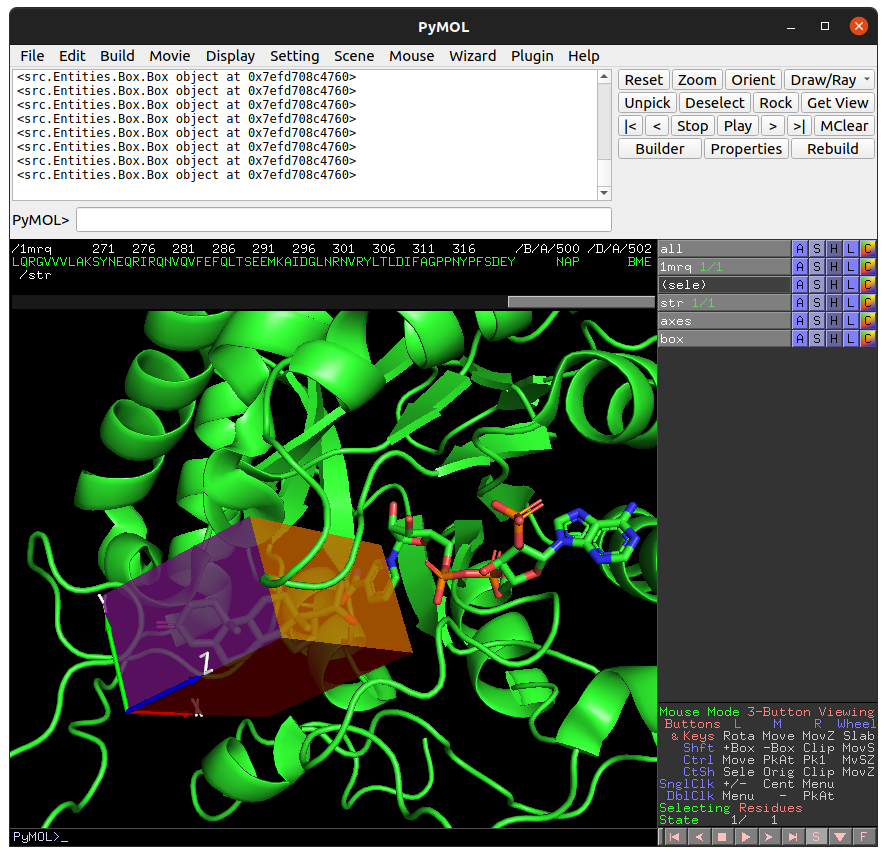
MAGI-Dock is a PyMOL interface to Autodock Vina that fills the gap between the power of the popular docking software's text-driven commands and the user interaction experience given by the widespread molecular graphics software. The plugin provides a GUI for each command-line preparation or simulation command. In addition, it provides an interactive drawing of the simulation box, a fundamental, one-time step to be carried out under user supervision. Moreover, the coupling of Autodock Vina with PyMOL through a graphical interface is conducive to lowering the learning curve of training in molecular docking simulation, with benefit for trainees in both academia and company/enterprise environment.

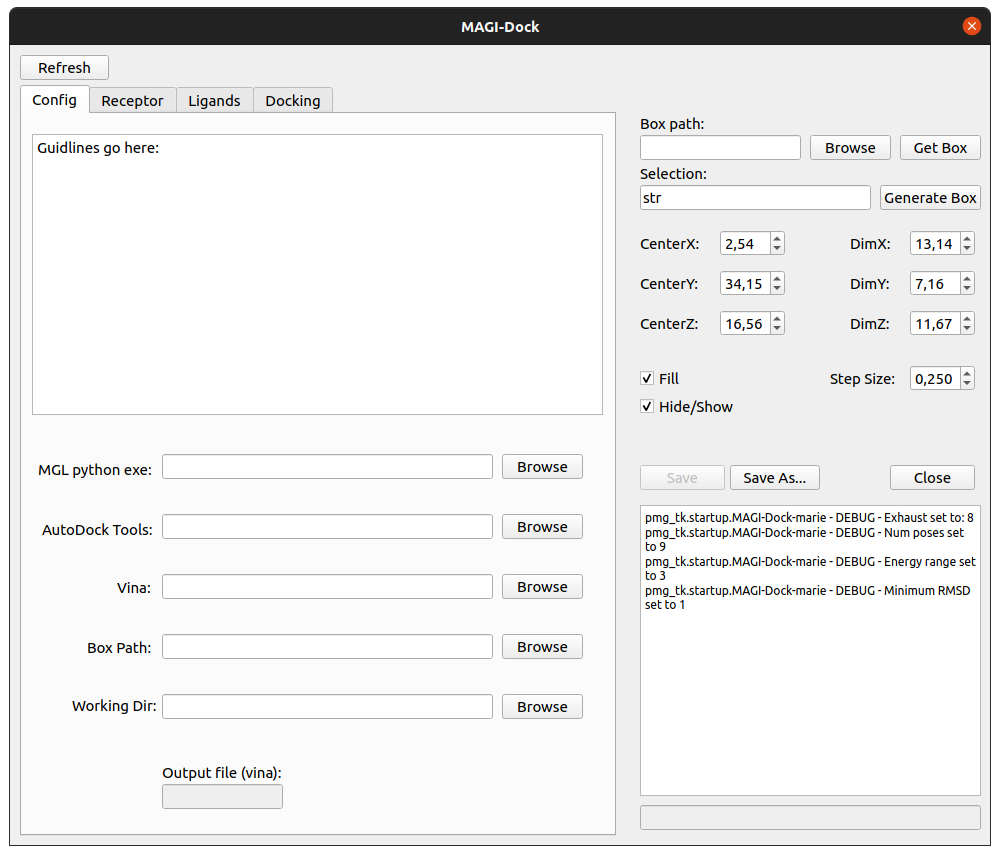
**SUPPLEMENTARY:**

Starting PyMOL in a remote cluster (mgltools, and vina should be loaded first)

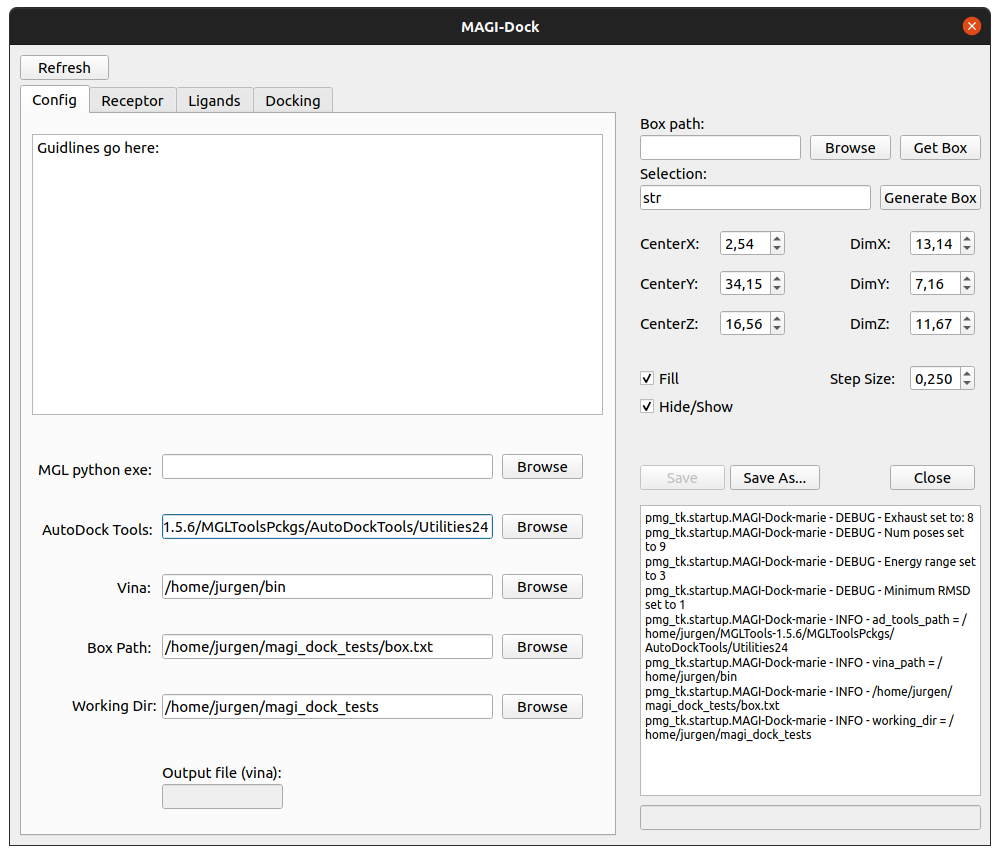


A typical workflow:

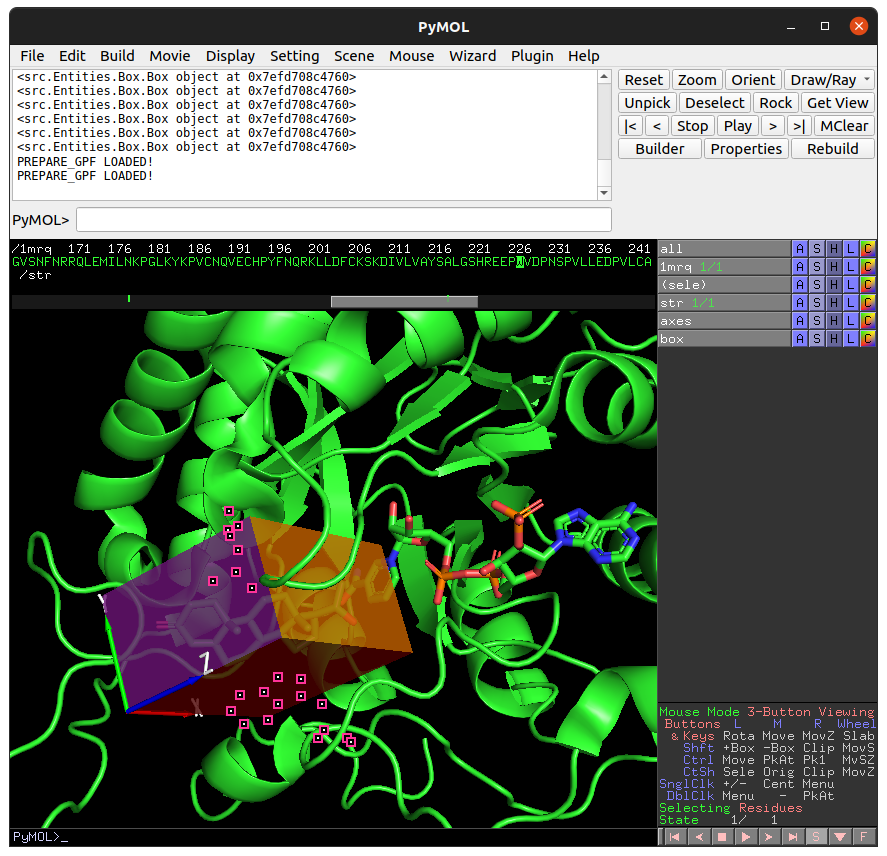


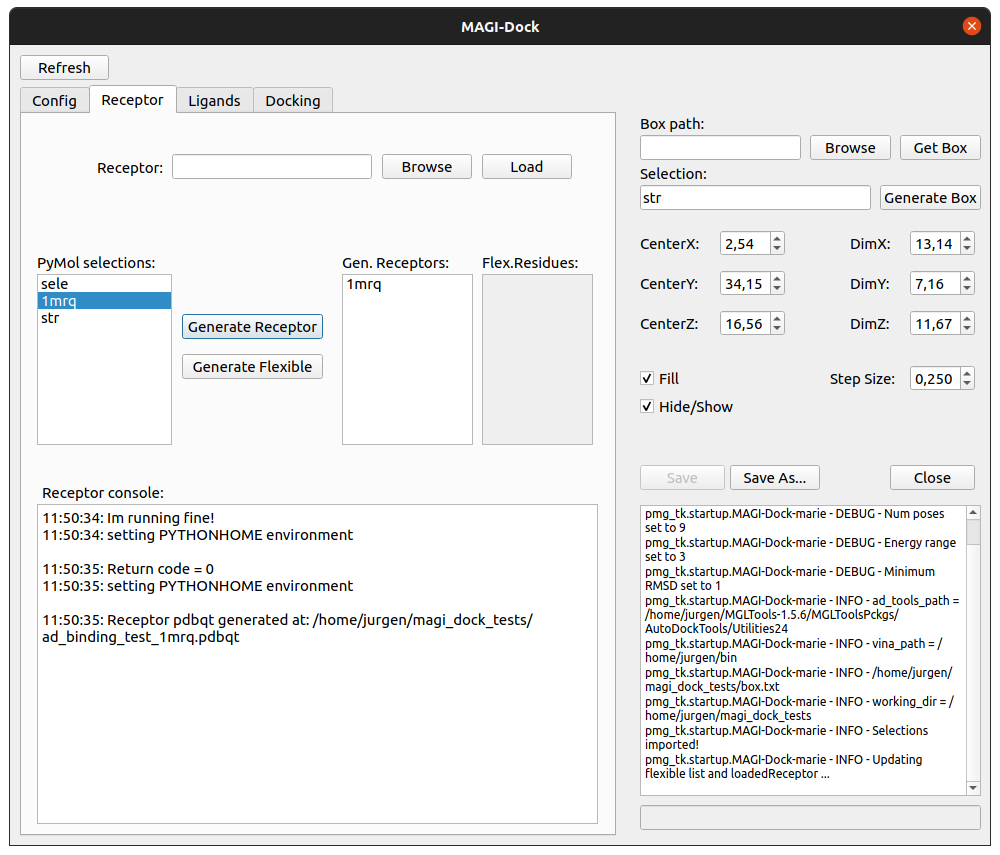


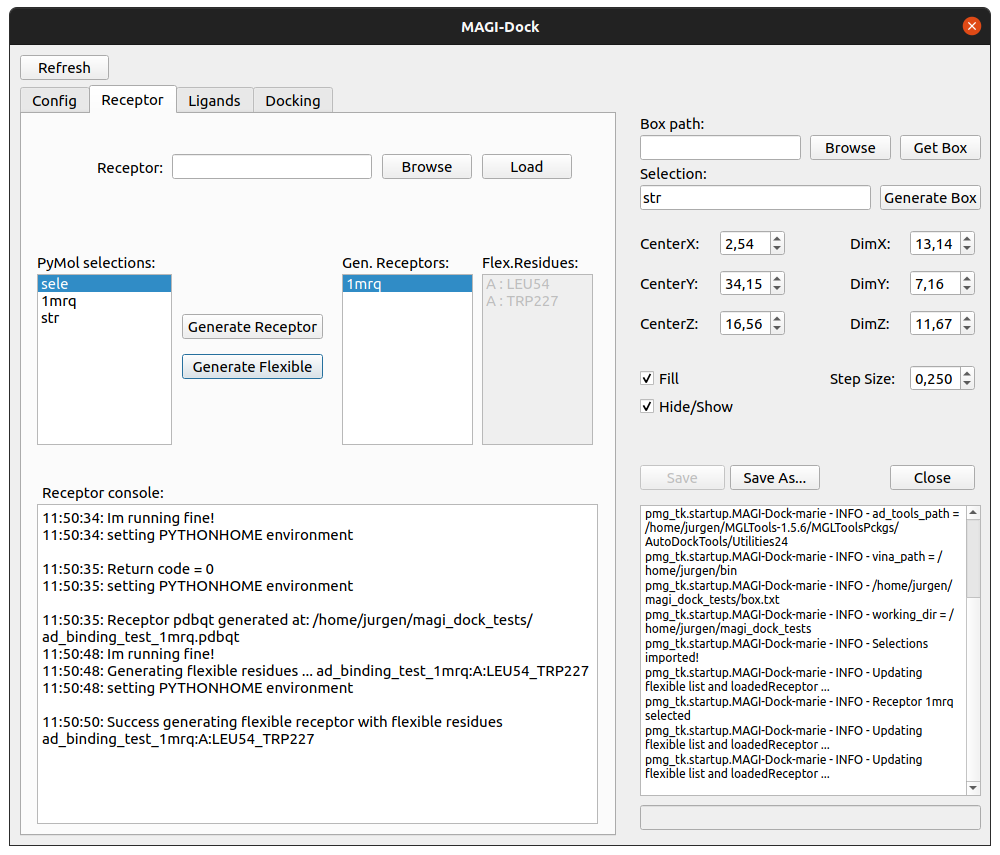
Generating a box around the steroid.



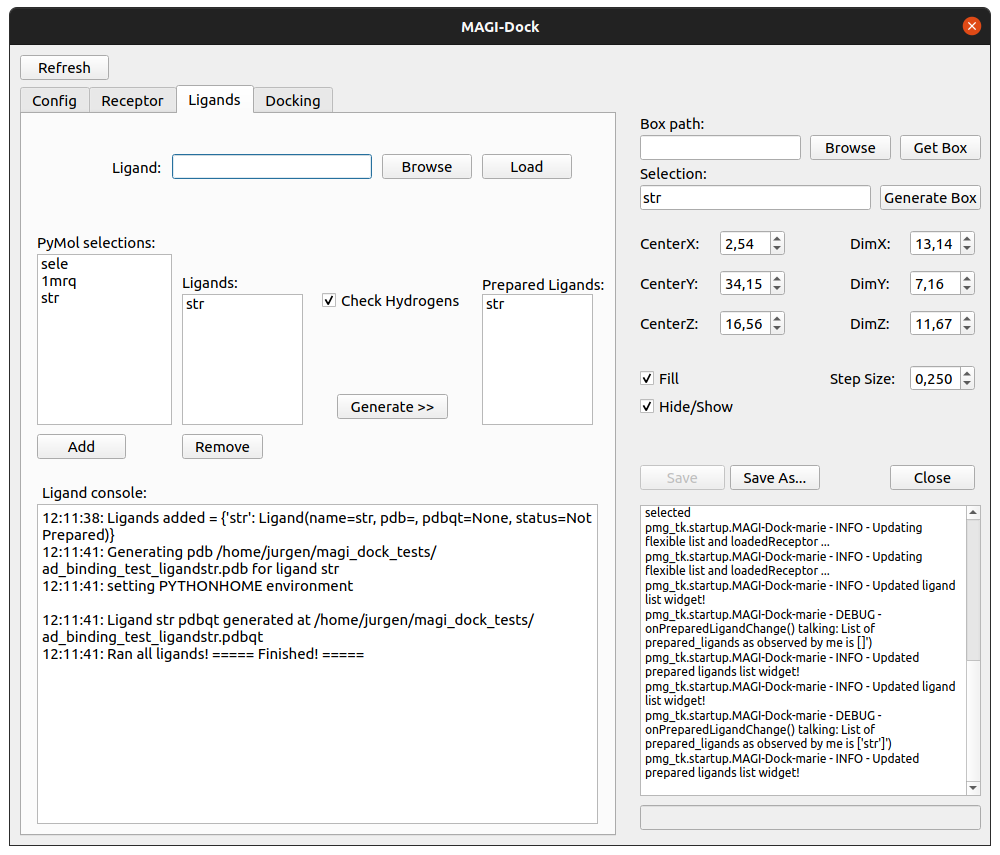
Specifying the paths (depending on the system, you can get away without specifying the mgl python exe path).



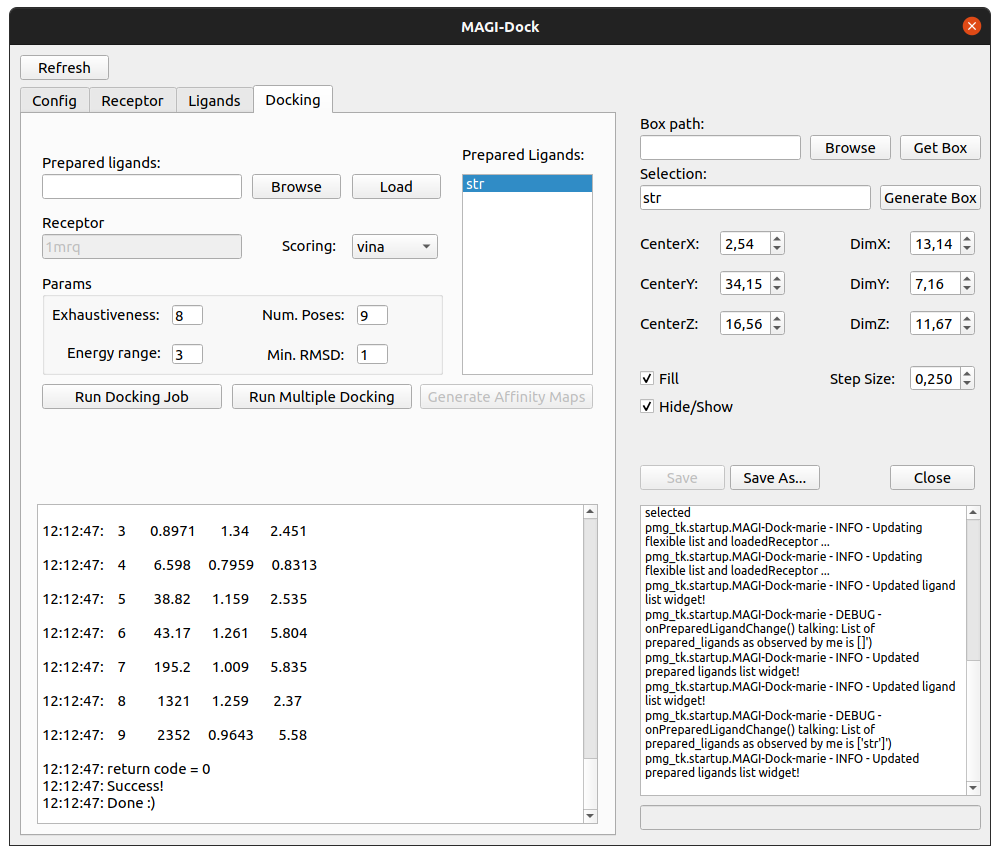




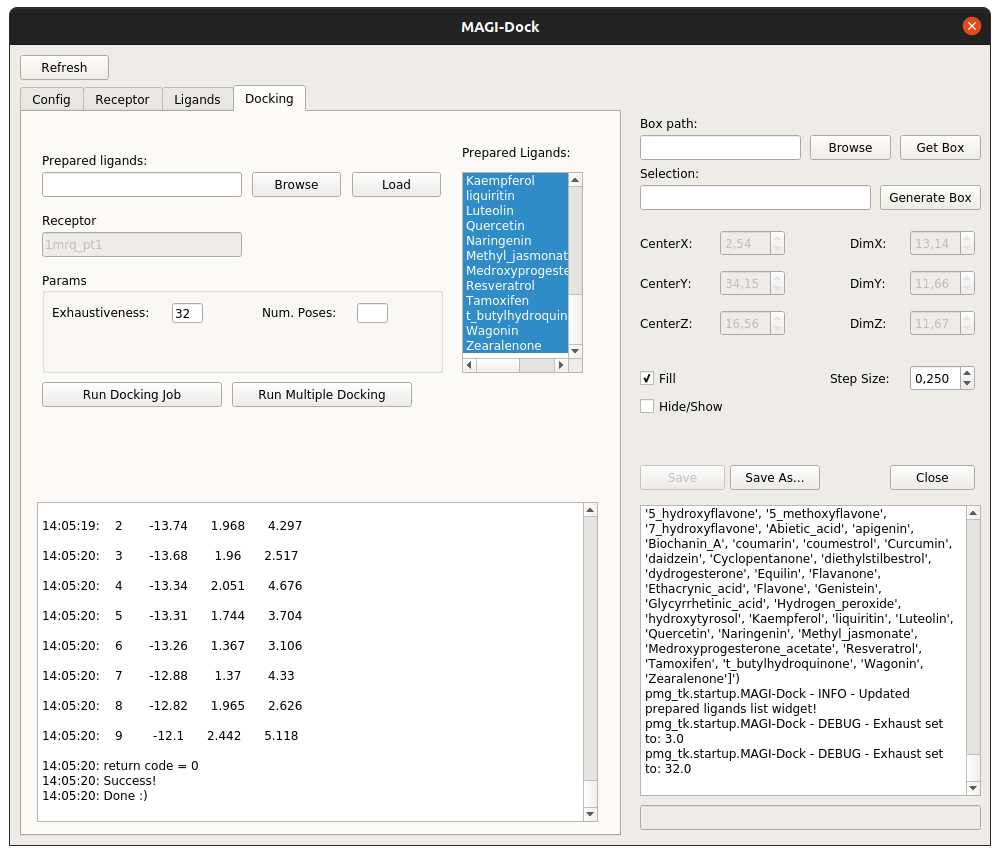
Generating the receptor and the flexible residues (specified in the ***sele*** selection).



Preparing ligands.



Running docking job.



Running docking on multiple ligands.