A Free and Source-Available Rendering Workflow for Ligand Binding Conformity for Biochemical Engineering

ABSTRACT

Biochemical 3D rendering solutions have consistently lagged behind the industry standard solutions present for 3D design, often due to the specificity of their intended workload. Improving render quality would allow for more clarity and intentionality in figures present in publications, and potentially improve scientific engagement during outreach programs and conferences. Unfortunately, many of these 3D rendering solutions are both costly and time-consuming to learn, placing them firmly outside the reach of non-commercially focused entities such as academic laboratories or prospective students. This is especially hindering in areas of biochemistry which benefit heavily from visualisation, such as biochemical engineering and medicinal chemistry. In an effort to resolve this issue, this report describes a novel methodology through which molecule binding conformities can be rendered at a quality exceeding the current industry standard, and which is entirely cost-free, using source-available and open source solutions wherever possible.

METHODS

Docking & Conformity – AutoDock Vina and SwissDock

Desired ligand and target are virtually docked together using either SwissDock (*Bugnon et al, 2024)(Röhrig et al 2023*) or AutoDock Vina *(Eberhardt et al, 2021*)(*Trott & Olson, 2010)*. AutoDock Vina is recommended due to the open-source aims of this report. SwissDock, whilst not open source, allows use of AutoDock Vina within it's interface, or alternatively the proprietary attracting cavities docking mode. After determining likely binding conformities through examination of ΔG, AC and SwissParam scores, download or export the docking result. This will provide a ".dock4" or similar file.

As a worked example in this report, porcine cathepsin H was docked with davunetide. The porcine cathepsin H structure was obtained from RCSB PDB (8PCH) *(Berman et al., 2000) (Guncar et al, 1997)*, whilst the structure of davunetide was obtained from PubChem *(Kim S et al, 2025)(National Centre for Biotechnology Information, 2025).*

Initial Visualisation & Modification – PyMOL

By default, PyMOL *(Schrödinger & DeLano, 2020)* will not be able to import predictive docking files. Due to this, we must install the "PyViewDock" extension *(Boneta, 2021)*. After installing, import the docking file generated in the previous step by following the PyViewDock documentation. After import, you will see multiple frames imported by

default, with each frame being a different binding conformity generated by your docking software. Select the desired binding conformity by selecting it's frame.

Here, any chemical modifications can be applied if necessary to your ligand. This should be limited to changes that do not alter the chemical properties of the molecule, such as removal of suspension or free hydrogens. If you wish to alter the chemical properties of the ligand, do so prior to docking using your favoured software.

After making any necessary modifications, export the structure from PyMOL using the ""Export Structure," and "Export Molecule" options, selecting all structures and using the frame selector to identify the conformity you wish to render. Export in the "PDBx/mmcif" format.

Final Modifications & Rendering – Blender

By default, Blender cannot import 3D data from the PyMOL outputted file format, so you must first install the Molecular Nodes extension *(Johnston, 2022)* in order to import your ligand into Blender. Consulting the Molecular Nodes documentation is strongly suggested hereafter. Import the PyMOL file using Molecular Nodes from local using the ball and stick method. Switch to rendered view. Here, you may notice geometry issues regarding the bonds. If this is the case, switch to the geometry nodes tab and locate the checkbox under named attributes "Find". Enable this, and bonds should appear in the correct position.

Next we must import the target. Once again, using Molecular Nodes import the .pdb of your target molecule from local – this time selecting the surface method. Use the .pdb included in your docking output, as it will automatically scale and orient itself to the exact correct conformity as seen in PyMOL.

Here is where any subjective changes are made to both the target and ligand to prepare it for final render. This involves determining material, polycount, scene lighting, and camera position amongst a vast quantity of other variables. If interested in tweaking the scene further, consult the Blender documentation *(Blender Team, 2025)*.

RESULTS

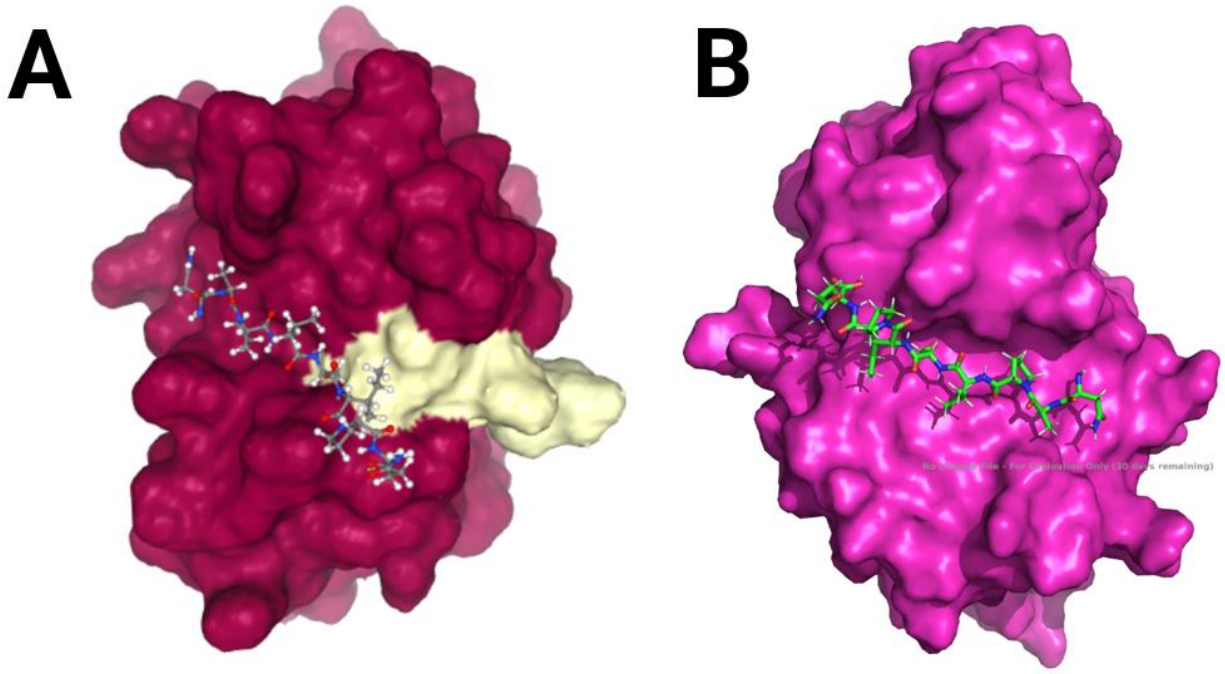

*Figure 1 – **1A** shows the top predicted binding conformity of porcine cathepsin H and davunetide rendered by SwissDock. **1B** shows the same binding conformity rendered by PyMOL 3.1.*

The above renders shown in *Figure 1A* and *1B* illustrate the capabilities of the rendering solutions present in SwissDock and PyMOL. Whilst both renders clearly show the binding configuration of the ligand to the target, resolution and rendering quality is extremely limited. Source resolution is also massively limited in *Figure 1A*, although it has been scaled in this report to alleviate the issue.

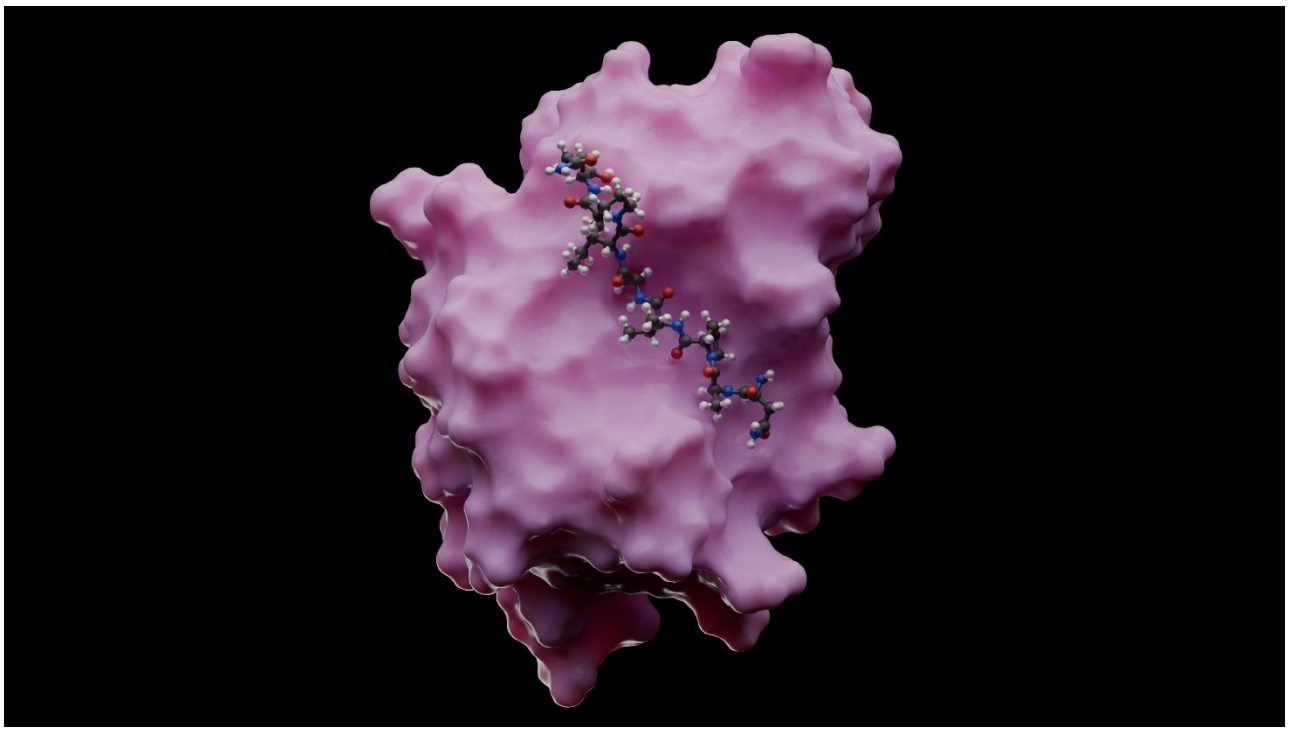

Figure 2 – Top predicted binding conformity of porcine cathepsin H and davunetide rendered in Blender 4.4 via Cycles.

As is clearly demonstrated in *Figure 2*, the subjective qualities of the Blender render are significantly improved whilst still maintaining the clarity present in *Figures 1A* and *1B*. This render has been tweaked from the default state after import, and has custom materials defined for both the ligand and target.

Full size renders ranging from 4K to 1K have been made available in the GitHub repository, alongside all files used to create said renders in each piece of software.

Also made available in the GitHub repository are brief animations of the type often found at scientific conferences, one rendered in Blender and the other in PyMOL. The integrated SwissDock renderer does not provide this functionality, and as such has been excluded.

DISCUSSION

I believe that the renders shown in *Figures 1* and *2* illustrate a clear quality improvement in favour of the Blender Cycles render. Given that this entire pipeline is either source-available or open-source and entirely free to implement, it seems it could be used in a myriad of ways across bioscience. This includes both students and professionals given the prevalence and ease-of-use of Blender and PyMOL.

Another aspect where blender also shows drastic improvements over PyMOL is granularity and control. Not only can you modify the appearance of the molecule and protein, but the scene around them to better incorporate them into presentations or publications as necessary. This pipeline could also be modified to fit medicinal chemistry modification of compounds and create high-quality renders to improve scientific communication efforts at presentations and conferences.

Regarding drawbacks, the only significant one is rendering time. For rendering still images, the time difference is negligible if using GPU-accelerated rendering. Without this capability, however, rendering times would increase significantly. As an alternative, Blender includes a less strenuous renderer – EEVEE *(Blender Team, 2025)*, which can easily be run on limited hardware and provides a less detailed render. Despite this potential limitation, you still gain all of the granular control inherent to working in Blender when compared to other software, and renders still appear of a significantly higher quality when compared to those in *Figures 1A* and *1B*.

To conclude, I believe that this report has clearly shown the merits of this rendering pipeline, and the vast majority of this must be attributed to the extremely talented developers of Molecular Nodes, PyViewDock, AutoDock Vina, PyMOL and Blender. Open-source and source-available software is vital in the educational ecosystem and the accessibility of each of these programs makes it possible to drastically improve render quality and better scientific communication.

REFERENCES

Berman, HM., Westbrook, J., Feng, Z., Gilliland, G., Bhat, TN., Weissig, H., Shindyalov, IN, Bourne, PE. 2000. *The Protein Data Bank*. Nucleic Acids Res. **28**:235-242. doi: 10.1093/nar/28.1.235.

Blender Team. 2025. *Blender Documentation*. Retrieved from https://docs.blender.org/. [Accessed July 6, 2025].

Boneta, S. 2021. *PyViewDock*. Retrieved from https://github.com/unizar-flav/PyViewDock. [Accessed July 6, 2025].

Bugnon, M., et al. 2024. *SwissDock 2024: major enhancements for small-molecule docking with Attracting Cavities and AutoDock Vina*. Nucleic Acids Res. **52**(W1):W324–W332 doi: 10.1093/nar/gkae300.

Eberhardt, J., Santos-Martins, D., Tillack, A.F., Forli, S. 2021. *AutoDock Vina 1.2.0: New Docking Methods, Expanded Force Field, and Python Bindings*. Journal of Chemical Information and Modelling **61**(8):3891-3898. doi: 10.1021/acs.jcim.1c00203.

Guncar, G., Podobnik, M., Pungercar, J., Strukelj, B., Turk, V., Turk, D. 1997. *Crystal structure of porcine cathepsin h determined at 2.1 angstrom resolution: location of the mini-chain c-terminal carboxyl group defines cathepsin h aminopeptidase function.* Structure **6**(1):51-61. doi: 10.1016/S0969-2126(98)00007-0.

Johnston, B. 2022. *Molecular Nodes.* Retrieved from https://extensions.blender.org/add-ons/molecularnodes/. [Accessed July 6, 2025].

Kim, S., Chen, J., Cheng, T., et al. 2025. *PubChem 2025 update*. Nucleic Acids Res. **53**(D1):D1516-D1525. doi:10.1093/nar/gkae1059.

National Centre for Biotechnology Information. 2025. *PubChem Compound Summary for CID 9832404, Davunetide*. https://pubchem.ncbi.nlm.nih.gov/compound/Davunetide. [Accessed July 7, 2025].

Röhrig, UF., Goullieux, M., Bugnon, M., Zoete, V. 2023. *Attracting Cavities 2.0: Improving the Flexibility and Robustness for Small-Molecule Docking*. Journal of Chemical Information and Modelling **63**(12):3925-3940. doi: 10.1021/acs.jcim.3c00054.

Schrödinger, L., DeLano, W. 2020. *PyMOL*. Retrieved from http://www.pymol.org/pymol. [Accessed July 6, 2025].

Trott, O., Olson, AJ. 2010. *AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading*. Journal of Computational Chemistry. **31**(2):455-461. doi: 10.1002/jcc.21334.