



Design and simulation of DNA, RNA and hybrid protein–nucleic acid nanostructures with oxView

Joakim Bohlin^{1,3}, Michael Matthies^{2,3}, Erik Poppleton^{2,3}, Jonah Procyk^{2,3}, Aatmik Mallya², Hao Yan² and Petr Šulc²✉

Molecular simulation has become an integral part of the DNA/RNA nanotechnology research pipeline. In particular, understanding the dynamics of structures and single-molecule events has improved the precision of nanoscaffolds and diagnostic tools. Here we present oxView, a design tool for visualization, design, editing and analysis of simulations of DNA, RNA and nucleic acid–protein nanostructures. oxView provides an accessible software platform for designing novel structures, tweaking existing designs, preparing them for simulation in the oxDNA/RNA molecular simulation engine and creating visualizations of simulation results. In several examples, we present procedures for using the tool, including its advanced features that couple the design capabilities with a coarse-grained simulation engine and scripting interface that can programmatically edit structures and facilitate design of complex structures from multiple substructures. These procedures provide a practical basis from which researchers, including experimentalists with limited computational experience, can integrate simulation and 3D visualization into their existing research programs.

Introduction

DNA and RNA nanotechnology¹ is a growing field in which DNA and RNA sequences are designed to self-assemble into nanoscale objects with defined geometries that are designed with nanometer precision. These nanoscale objects have found applications in diverse fields, including precision drug delivery, diagnostics, immunology, photonics and nanoelectronics^{2,3}.

Development of oxView

Several software tools have been developed to facilitate the creation of larger and more complex shapes^{4–12}. Molecular modeling packages specifically aimed at the large size- and timescales inherent in nanostructure design have been developed to characterize the mechanical properties of DNA and RNA nanostructures^{13–16}. Specifically, oxDNA and oxRNA^{17–21}, a nucleotide-resolution coarse-grained molecular simulation package that represents DNA and RNA nucleotides as single rigid bodies with force fields parameterized to capture the mechanical and kinetic properties of DNA and RNA, is a very popular tool for prototyping structures and rationalizing experimental results with nucleotide-level resolution of nanostructure behavior. Recently, we developed an extension to oxDNA and oxRNA²² that also includes a coarse-grained representation of proteins for evaluating protein–nucleic acid hybrid nanostructures through an anisotropic network model (ANM) that captures equilibrium fluctuations of protein components. The oxDNA and oxRNA models have been used in >150 works, ranging from DNA/RNA nanotechnology to studies of RNA and DNA biophysics¹³.

Until recently, setting up and evaluating oxDNA simulations had been a laborious task that required substantial knowledge of statistical physics, as well as Bash and Python scripting. To remedy this, we recently released a suite of web- and console- based tools that simplifies setting up simulations. These include TacoxDNA (<http://tacoxdna.sissa.it/>)²³, a website containing tools for converting structures from many other design tools into the oxDNA file format; oxdna.org²⁴, a website that provides users with a simple user interface for running oxDNA/RNA simulations on a graphics processing unit (GPU) cluster; oxDNA analysis tools (https://github.com/sulcgroup/oxdna_analysis_tools)²⁵, a Python library for analyzing oxDNA simulation results; and the primary

¹Clarendon Laboratory, Department of Physics, University of Oxford, Oxford, UK. ²School of Molecular Sciences and Center for Molecular Design and Biomimetics, The Biodesign Institute, Arizona State University, Tempe, AZ, USA. ³These authors contributed equally: Joakim Bohlin, Michael Matthies, Erik Poppleton, Jonah Procyk. ✉e-mail: psulc@asu.edu

subject of this protocol, oxView (oxview.org)²⁵, a web-based oxDNA configuration visualizer, free-form nanostructure editor and interactive simulation interface.

oxView was first made available in 2018 for a group of users and developers of oxDNA, and has since been used in many publications^{9,22,26–31}. It was used by our group to set up and evaluate simulations to gain insight into experimental systems^{26,28}, and the tool itself was first described in an article published in 2020²⁵. An additional publication covering some of the protein visualization features, which have since been extended, was published in 2021²². oxView can be used both as part of the structure computation and characterization pipeline as well as a stand-alone design tool. In this protocol, we describe how to use oxView to design new structures or edit imported ones, and how structure design can be coupled with coarse-grained simulations to characterize the behavior of the nanostructures. We also demonstrate its advanced features, which include the design of hybrid protein–DNA (or RNA) nanostructures and using scripting commands to automate structure editing. The tool is freely available at oxview.org.

Software overview

oxView is a single-page web application that runs in all modern browsers. It serves as both an oxDNA configuration visualizer and a DNA/RNA structure editing tool. Prior to oxView, oxDNA lacked good visualization tools, limited to a slow converter to Chimera³² and Cogli1, a visualization-only tool that struggled with large configurations and multi-gigabyte trajectories. oxView handles large structure files well; it can load files in excess of 1.6 million particles on a GPU-equipped desktop (see ‘Equipment’ in the ‘Materials’ section for hardware requirements) and load trajectories on the fly, allowing video creation from multi-gigabyte trajectory files. The majority of nanostructure design is well within the grasp of oxView’s size limitations, resulting in a seamless design and editing experience. In practice, oxView has an upper limit on the size of a single configuration input that depends on the JavaScript (JS) implementation of the specific browser being used. In Firefox, the limit is 1 GB per configuration, which corresponds to between 3 and 4 million particles. In practice, the biggest system ever simulated using oxDNA is <2 million nucleotides, so this limitation should not have a significant impact on users. While visualizing a structure, the user can create custom color palettes, either on a per-strand basis, through selection and coloring of particular sections, or through colormap files, which can be generated for various per-nucleotide mechanical properties via oxDNA Analysis Tools. Scenes can be exported from oxView into 3D modeling file formats for high-quality rendering and 3D printing, and trajectories can be made into videos or image series in-browser.

On the editing side, oxView is a free-form editing tool, so it lends itself to de novo design of small 3D nanostructures with complex geometries. Also, because of the numerous converters into the oxDNA format and the simplicity and freedom of the oxDNA file formats, oxView is a neutral ground between other design tools, allowing high-resolution editing of individual structures, for example, modifying junctions produced by 3D CAD designers or off-lattice extensions to structures designed in a lattice-based tool. oxView also allows one to combine structures designed in two different initial design tools into a single, larger structure with custom linkers. oxView has a scripting interface where users can use a library of custom-built functions alongside basic JS to perform mass-edits and interrogate structures.

oxView is written in TypeScript (~15,000 lines of code) using the Three.js 3D rendering library. For most uses, the web interface (oxview.org) is sufficient; however, if a user would like to create custom extensions of their own, all the source code is available from GitHub (<https://github.com/sulcgroup/oxdna-viewer>) under a GPL3 license and can easily be run on a local static web server. oxView is under continuous development, with a global user community providing feedback, feature requests and suggestions for further improvements. Complete documentation is also available on the GitHub page. Users with bug reports, questions or comments should post on the ‘issues’ page on GitHub.

Relationship with other programs

oxView was originally developed as a visualizer for oxDNA simulations. There have been many internal improvements since its introduction to the nucleic acid nanotechnology community; however, oxView is still intrinsically tied to the oxDNA simulation engine. Its visualization capabilities have developed alongside the simulation and analysis development of oxDNA. Protein visualization was added to support the ANM-oxDNA protein model²², which introduced protein representation

into oxDNA model for construction of hybrid protein–DNA or protein–RNA nanostructures. Analysis overlays were created to support visualization of the output from oxDNA Analysis Tools²⁵. Additional file reading options were created to support the TacoxDNA²³ and oxdna.org²⁴ servers, where it is integrated as part of the webpages. oxView complements existing design tools used in the DNA and RNA nanotechnology field^{4–12}, which can be used to obtain initial structure designs, often confined to a predefined lattice architecture or filling a polygonal shape in space. Designs from other tools can be loaded, edited, saved and exported. One of the unique capabilities oxView offers is that it is the only DNA and RNA nanotechnology design tool that can interactively edit and simulate nanostructure configurations using the oxDNA model. This allows users to immediately include feedback from simulations in the iterative design process. It also joins Tiamat⁵ and Adenita⁸ as apps that are fully free form and can also design RNA. In addition, oxView and Adenita are the only DNA/RNA nanostructure design tools that also allow the inclusion of protein structures. DNA nanostructure design is a complex artform¹² where no tool provides a complete design pipeline. Lattice-based tools such as caDNAno⁴ are generally superior for building large structures. However, they lack the complete freedom and support for complex junction types that can be built in free-form design tools. Large nanostructures will first be designed from scratch using one of the available design tools and imported into oxDNA (Tiamat or caDNAnano) or directly exported to the oxDNA format (in the case of scaDNAno¹¹, Adenita⁸, MagicDNA⁹ or vHelix^{6,7}). oxView can then be used to modify the structure or construct large-scale assemblies of multiple structures, run in-browser simulations or prepare these designs for simulations with oxDNA. The oxDNA simulations can then be imported and visualized with oxView.

Strengths and limitations of oxView

oxView offers flexibility and scale that is unmatched in the DNA/RNA nanotechnology software environment. Because it is fully browser based, it runs on most computers and does not require installation or any specialized computational knowledge to view and edit a structure. As a visualizer, it can load larger oxDNA simulations than other available visualizers (such as Cogli, or via conversion to Chimera or PDB-format viewers such as Visual Molecular Dynamics (VMD)) and also reads large trajectory files more gracefully, avoiding memory allocation issues that plague some visualization options, specifically Cogli 1. On the editing side, oxView is one of only a few fully free-form design tools, able to integrate designs from multiple sources into single simulations or strand lists for experimental preparation thanks to the variety of exporters into the oxDNA format. oxView is also the only editing tool that works directly in the oxDNA simulation data format. This makes it the best available tool for setting up and visualizing results of oxDNA simulations.

This flexibility has downsides, however. oxView is not a good tool for de novo design of large nanostructures as it does not have features such as scaffold routing or auto-stapling, which are key algorithms that have allowed the growth of the DNA and RNA nanotechnology field. oxView is not a tool for building an origami structure from scratch; the tools mentioned in the previous section each automate specific aspects of the process and have more abstract representations of DNA, which facilitates large-scale design. However, many of these tools have exporters and converters to the oxDNA format, which makes oxView extremely useful for assembling a single sequence list that combines multiple components, setting up simulations from such structures or making off-lattice edits to structures designed using lattice-based tools.

Experimental design

The general outline of any simulation-based DNA/RNA nanotechnology project is nanostructure design; conversion and relaxation; production simulation; evaluation; iteration. We have found that oxView represents a substantial improvement over previous tools for all steps except the simulation itself. Much of our work involves complex, multicomponent nanotechnology systems. We often receive the initial designs from collaborators or create them in one of the popular DNA nanotechnology design formats, most commonly caDNAno or Tiamat. We then load them in oxView as described in Procedure 1 and create any off-lattice components and combine the structures using oxView's editing tools as described in Procedure 2, where a DNA tetrahedron from Goodman et al.³³ is re-created. For simple projects, we then prepare the structure for simulation using the relaxation protocol described in Procedure 3. Relaxation simulations using oxServe show, in real time, the molecular dynamics simulation of the structure using the oxDNA model, and can hence also provide immediate feedback on the effects on any design change (such as extending single-stranded regions,

removing a duplex region or selecting two structures to bind together). We then export the combined design for oxDNA simulation and Python analysis as described in Procedure 4. The results of the simulation, including videos and data overlays, are then created and visualized using oxView. After evaluating the results, the simulation may reveal weaknesses in the structure or suggest further simulation experiments. If further edits to the structure are required, these can be done directly on the already-simulated structure (returning to the tools in Procedure 2) and exported back to simulation with a much shorter relaxation protocol, resulting in faster iterations and more time spent producing quality simulation data. For more complicated simulation experiments where mass edits of the structure are necessary, we make these edits programmatically using oxView's scripting interface, described in Procedure 5, to prepare the initial structure for simulation. If the design of interest is a hybrid structure also including protein or peptide components, Procedure 6 details how to create and parameterize a simple protein model for simulation alongside the DNA or RNA structure, while Procedure 7 re-creates a protein–DNA hybrid structure from Xu et al.³⁴. Procedures 8 and 9 demonstrate 3D rendering and printing and virtual reality (VR) as more immersive options for visualizing structures.

While some of the described procedures could be done in sequence, each procedure is a separate and independent task. One could use the specific outputs from Procedure 1, 2, 5, 6 or 7 as the input for Procedure 3 or 4; however, Procedures 3 and 4 can use any oxDNA file pair as input, and the other procedures are intended to provide example starting configuration/topologies.

The main use cases for oxView tool have been described in this protocol and the Supplementary Information. In the following Procedures, we describe examples illustrating the use of oxView to design a nanostructure and study it through coarse-grained simulations. The files used in these examples are available in the Examples directory on our GitHub repository (<https://github.com/sulcgroup/oxdna-viewer/tree/master/examples>), and also as a zip file in the supplement of this paper. The tool has other features (such as custom visualization options including nucleotide color/size, rendering materials and lighting), and it is constantly evolving, based on requests from the community as well as new challenges presented by experiments, simulations and new design software and file formats. The most up-to-date documentation and examples are kept on the tool GitHub repository github.com/sulcgroup/oxdna-viewer.

Applications of oxView

oxView's primary purpose as a tool is to facilitate structural analysis of large DNA and RNA structures using oxDNA/RNA. Previous applications of oxView's visualization capabilities include evaluating the differences between two similar design paradigms with vastly different yields of correctly formed product²⁸, verifying that distances between functionalized nucleotides match intended designs²⁹, confirming overall shape and stiffness evaluation for a library of large nanostructure designs⁹ to correlate local stiffness with cell uptake³¹, and calculating the dynamics of mechanically interlocked structures^{30,35}. oxView has also been used to design structures for simulation. In Yao et al.²⁸, the structures for simulation were built by combining two components originally designed in Tiamat⁵ with connections and external forces created using oxView. In Yu et al.³⁵, the structures for simulation were built from scratch using oxView's free-form design tools.

Materials

Equipment

oxView is a single-page web browser app that does not require the user to download or install any external libraries or code. The tool runs in all major web browsers (Chrome, Edge, Firefox and Safari) on major operating systems (Windows, Mac OS X and Linux) and standard office desktop/laptop hardware configuration (1.6 GHz CPU, 8 GB RAM) is sufficient for most nanostructure designs. Additionally, the application is packaged as an Electron executable available for Windows and Linux at <https://github.com/sulcgroup/oxdna-viewer/releases>. This allows oxView to be run as a standalone executable. For designs that exceed 30,000 nucleotides, a desktop or a high-end laptop with a GPU card (NVIDIA 1080 GTX or better) and larger RAM (16 GB or more) is recommended.

The visualization capabilities are also available on mobile browsers (iOS and Android) as well as WebXR-capable VR systems (tested on Oculus and Google Cardboard). oxView uses the popular Three.js 3D rendering library, which makes it easily extendable with custom scripting in the web console, through either default Three.js functions or an API that provides access to oxView's data structures.

Box 1 | File format descriptions

oxView: the oxView format is a JSON-based representation that can contain DNA, RNA or protein designs formatted in a way that is easy for oxView to read and write. Use this format to save your current work without any information loss. Details of the format specification and an example of a small structure represented in it can be found in section S1 of the Supplementary Information.

oxDNA: the oxDNA format, in the form of configuration and topology files that specify nucleotide position/orientation and connectivity, respectively, is used to save and load oxDNA simulation files. You can also load a single configuration file to update the currently loaded configuration. oxView can also load and display parameter (.par) files used in the ANM-oxDNA model.

PDB: the PDB format is a common all-atom format and can be used to import DNA, RNA or protein structures.

caDNAAno: the caDNAAno⁴ format is one of the most popular formats for DNA origami designs. Previously, caDNAAno files had to be converted into oxDNA files through TacoxDNA, which resulted in a loss of information about base pairs, strand colors and clusters. Using the tacoxdna.js library, oxView can now import caDNAAno files directly. Furthermore, a custom scaffold sequence can be set as the file is imported.

dnajson: Tiamat⁵ designs saved as '.dnajson' files can be imported directly into oxView. Designs in the binary tiamat format '.dna' need to first be opened in Tiamat and saved as '.dnajson'.

rpoly: the rpoly format is the output from the BSCOR software package^{6,7}, which converts polyhedral meshes into DNA structures that can be edited using the vHelix plugin in Autodesk Maya⁷. oxView can be used as an alternative, which does not require installation of Autodesk Maya, for the visualization and editing of the DNA nanostructures encoded in the BSCOR output.

oxDNA simulations either can be run on our `oxDNA.org` server or the user can install oxDNA on their local server or workstation. The oxDNA simulation package is supported on Unix operating systems and requires cmake (2.8 or newer) package and gcc (version 4.6 or newer) to compile. The GPU-enabled version requires an NVIDIA GPU card and the CUDA library (4.0 or newer).

The Python scripts to evaluate simulations require Python (version 3.6 or higher) with the following additional libraries installed: NumPy 1.16³⁶, Matplotlib 3.0.3 (minimum version 3.0)³⁷, BioPython 1.73³⁸, SciKitLearn 0.21.2³⁹ and Pathos 0.2.3⁴⁰. Most scripts can be used on any major operating system (Windows, Linux and Mac OS X); however, some require a working DNA analysis binary (part of the oxDNA package) for calculating energies, which is only available on Linux and Mac OS X.

Data format

The tool has an import feature from three popular design formats for DNA nanotechnology (caDNAAno, vHelix and Tiamat), as well as PDB format. Box 1 outlines relevant file formats compatible with oxView. Design files exported from other popular design tools can be converted as described in Procedure 1 below. Natively, the tool supports the file format that is used by the oxDNA model to represent DNA or RNA nanostructures, and ANM-oxDNA format for protein–DNA/RNA hybrids. A description of these file formats can be found in the supplement as well as in the official oxDNA documentation at <https://dna.physics.ox.ac.uk>. The tool also uses its own file format, oxview, a JSON-based structure description, and its format is described in the Supplementary Information. We note that, for the coordinate system, oxView uses the simulation units of the oxDNA format, where 1 distance unit corresponds to 0.8518 nm.

Procedure 1: importing designs from other software into oxView ● Timing Instantly to several minutes, depending on structure size

▲ **CRITICAL** oxView can currently import and load structures in any of the following formats: oxView, oxDNA, PDB, caDNAAno, dnajson (Tiamat) and rpoly (vHelix). oxView, oxDNA and PDB files can be opened by dragging and dropping them onto the oxView window or via the 'Open' dialogue in the File tab. caDNAAno, dnajson and rpoly files require user input during import and are loaded using the TacoxDNA dialogue in the File tab. Other design tools, including Scadnano¹¹, MagicDNA⁹ and Adenita⁸, have direct export into the oxDNA format, while others, including Athena¹⁰, have direct export into PDB format. Many formats (including PDB) can be converted to the oxDNA format using the TacoxDNA²³ webserver. Brief format descriptions can be found in Box 1. In this example, we load a caDNAAno structure of the linear actuator demonstrated in Benson et al.³⁰. While smaller structures can be imported almost instantly, very large caDNAAno structures can take several minutes. In the following example, the rail took 30 s to import while the slider was imported in 5 s.

- 1 Import the rail caDNAAno design (Fig. 1):
 - In the 'File' tab, click 'Import'.
 - Locate the 'rail.json' file in the examples folder in our GitHub repository.
 - Make sure that 'caDNAAno' is selected as file format.
 - Select 'Hexagonal' as lattice type.

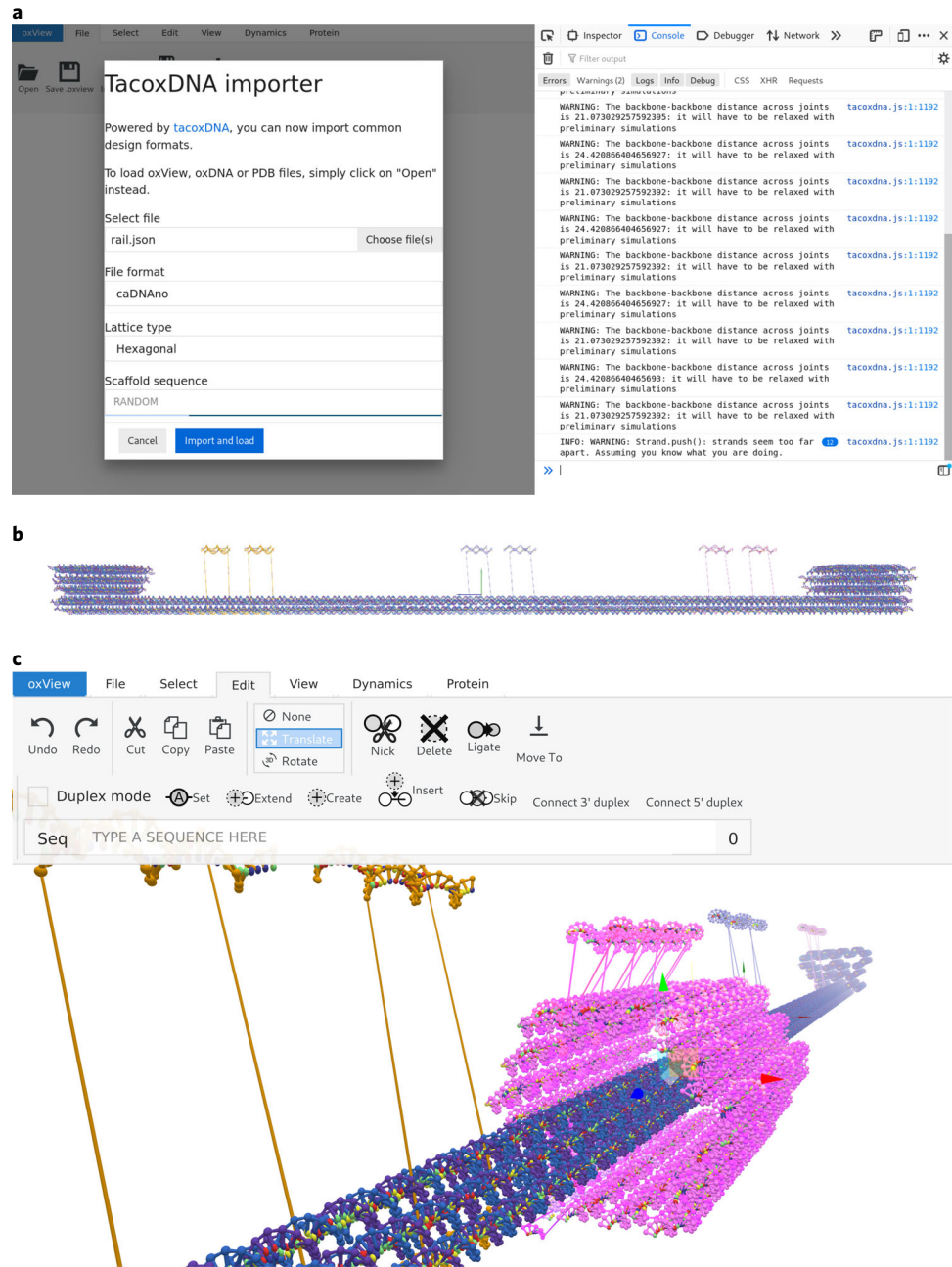


Fig. 1 | Importing and assembling designs created in caDNAno. a, caDNAno import options. Note the additional output in the web console to the right. **b**, Imported rail design from ref. ³⁰. Note that the custom colors match the staple colors painted in caDNAno. **c**, Imported slider design from ref. ³⁰. Select the entire slider design using the System selection mode, and use the translation tool (in the 'Edit' tab) to position the slider.

- Leave the scaffold sequence blank to set a random sequence. As caDNAno files do not contain the scaffold sequence (the longest sequence in a DNA origami), the scaffold sequence must be explicitly defined here to get the sequence identity intended in the design. If the exact base pairing is unimportant to the design, leaving this blank is acceptable as the default for the oxDNA model uses an average sequence model. If, however, the specific sequence is desired, the scaffold sequences for both the rail and slider can be found in the examples folder.
 - Click 'Import and load'. The import log is visible in the web console, shown in Fig. 1a. This step might take a few minutes, depending on the size of your design and the speed of your computer.
- ? TROUBLESHOOTING**
- 2 Repeat Step 1 for 'slider.json' found in the examples folder.

- 3 Position the slider on the rail:
 - In the ‘Select’ tab, enable the ‘System’ selection mode and click on the slider to select it.
 - In the ‘Edit’ tab, toggle the ‘Translate’ tool (or press ‘T’ on the keyboard). Use the arrows to position the slider on the rail, as seen in Fig. 1c.
- 4 (Optional) Use the ‘Cluster’ selection mode to select and translate the single-stranded clusters so that you avoid the extended backbone bonds seen in Fig. 1c. Make sure to place the slider overhangs on the inside of the slider.
- 5 (Optional) To connect the slider to a specific position on the rail, you can create mutual trap forces between the complementary single strands to pull them together (make sure to set their sequences accordingly) by individually selecting bases in an alternating fashion for all base pairs you want to pair, going to ‘Forces’ in the ‘Dynamics’ tab, and choosing ‘Create From Selection’. Alternatively, delete the single strands by selecting them then pressing the delete key, and replace them with new duplexes of the correct length by typing the sequence being replaced into the sequence text box under ‘Edit’ and with ‘Duplex mode’ checked, and clicking ‘Create’.
- 6 Download the oxView file, which saves the clustering, coloring and separate systems by clicking the oxView logo in the Save section of the File tab. The file can also be downloaded for simulation in oxDNA, see Procedure 4, by clicking the DNA helix in the Save section of the File tab.

Procedure 2: free-form nanostructure design and editing ● Timing dependent on complexity of the design

▲ **CRITICAL** While oxView started as a tool to simply visualize structures, it has become an increasingly capable tool for editing or even designing structures from scratch. To showcase the design capabilities in oxView, Procedure 2 describes how to draw a DNA tetrahedron, from Goodman et al.³³, as seen in Fig. 2. Free-form editing is not hardware limited, so the time it takes to design a structure depends only on the complexity of the design and the user’s familiarity with the interface.

- 1 Create a helix (Fig. 2a). To do this, input a 20-base sequence into the sequence text box, make sure “Duplex mode” is selected, then click the “create button” (see Fig. 3 for a list of all available editing tools.). We can just use a random sequence here, by typing “NNNNNNNNNNNNNNNNNNNNNNNN” (oxView uses the IUPAC one-letter codes for the nucleotides), but note that the given sequence can be changed later (using the “Set sequence” tool).
- 2 Duplicate and transform (Fig. 2b).
 - Go to the ‘Select’ tab, change the selection mode to ‘Cluster’ and click to select the created helix. Then use the copy (Ctrl + C) and paste (Ctrl + V) tools in the ‘Edit’ tab to create another helix.
 - Use the translate (T) and rotate (R) tools to position each helix. Although it is possible to move and rotate the helices in 3D, it is easier to keep them in a 2D plane for now. When the translate and rotate tools are active, arrows and planes will appear on the selected object. The red, green and blue arrows correspond to translation along or rotation around the X, Y and Z axes, respectively. In translation mode, clicking the yellow, magenta and cyan planes allows translation in the XY, XZ and YZ planes, respectively. The white square at the center of the arrows in translation mode and the yellow arrows in rotation mode correspond to translation and rotation in the plane perpendicular to the orientation vector of the camera.
 - Repeat Step 2 until you have created and oriented all six helices, as shown in Fig. 2b. Clear selections by double-clicking the background (or use the ‘Clear Selection’ button in the ‘Select’ tab).

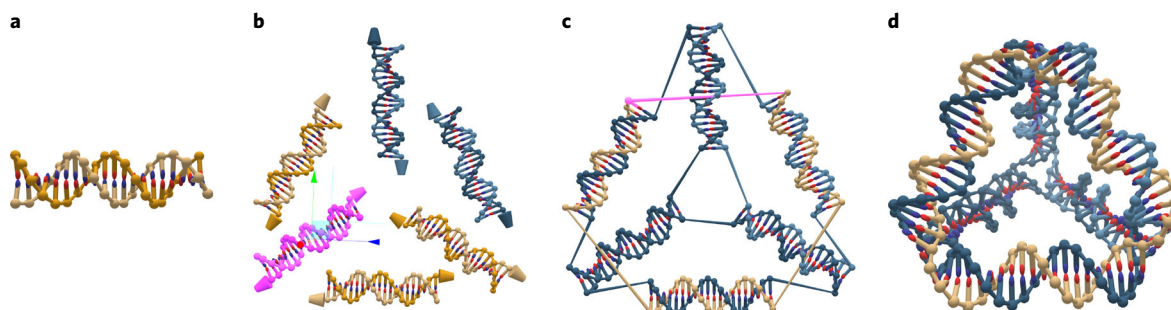


Fig. 2 | Designing a DNA tetrahedron using the oxView editing tools. **a**, The initial helix created. **b**, Duplicated helices being translated into place. The cyan square on the translation tool associated with the selected helix will correspond to in-plane motion for these strands. Note how 3’ markers have now been enabled (large cones marking the 3’ end of strands) to make it easier to identify which ends should be ligated. **c**, Strands ligated together. **d**, The resulting 3D tetrahedron shape, originally from ref. ³⁴, as seen after applying RBD.




















Tool	Description
	Create a new strand from a given sequence. select <i>duplex mode</i> to instead create a helix.
	Copy the selected elements (Ctrl+C).
	Cut the selected elements (Ctrl+X).
	Paste elements from clipboard (Ctrl+V to paste in original position, or Ctrl+Shift+V to paste in front of camera).
	Delete all currently selected elements (delete).
	Ligate two strands by selecting the 3' and 5' endpoint elements to connect (L).
	Nick a strand at the selected element (N).
	Extend strand from the selected element with the given sequence. select <i>duplex mode</i> to also extend the complementary strand.
	Insert (add) elements within a strand after the selected element.
	Skip (remove) selected elements within a strand.
	Rotate selected elements around their center of mass (R).
	Translate currently selected elements (T).
	Move to. move other selected elements to the position of the most recently selected element.
	Connect 3' duplex. connects the 3' ends of two selected staple strands with a duplex, generated from the sequence input.
	Connect 5' duplex. connects the 5' ends of two selected staple strands with a duplex, generated from the sequence input.
	Set the sequence of currently selected elements. select <i>duplex mode</i> to also set the complementary sequence on paired elements.
	Get. assigns the sequence of selected bases to the sequence input.
	Reverse complement. generates the reverse complement of a provided sequence.
	Search. highlights the position the provided sequence in each strand, if present.

Fig. 3 | Editing tools available in oxView. All tools are found under the 'Edit' tab. It is also possible to undo (Ctrl + Z) and redo (Ctrl + Y) edits.

- Ligate strands. With the monomer selection mode enabled, select exactly one 3' and one 5' end and then click ligate (L) to connect them. Connect the strands as shown in Fig. 2c. The backbones of the strands are slightly tapered toward the 3' end of the strand, which provides a constant visual identification of strand polarity. To make 3' ends even more visible, as seen in Fig. 2b, enable 3' markers in the 'View' tab.

? TROUBLESHOOTING

- Enable rigid-body dynamics (RBD) in the 'Dynamics' tab. Each helix was automatically assigned to a cluster when created, so it is now easy to toggle rigid-body relaxation to bring everything into a proper tetrahedron. To bring the clusters closer together, you can decrease the cluster repulsion constant and the connection length while increasing the spring constant, in the rigid-body settings (here we used repulsion = 100, spring = 100 and relaxed length = 1). The result should look something like Fig. 2d. For more details on RBD, see Procedure 3.
- (Optional) Finally, add spacers of a given sequence in the hinges using the insert tool and, if desired, cut strands using the nick tool.
- Download the oxView file, which saves the clustering data by clicking the oxView logo in the Save section of the File tab. The file can also be downloaded for simulation in oxDNA, see Procedure 4, by clicking the DNA helix in the Save section of the File tab.

Procedure 3: interactive design and simulation ● Timing Minutes to hours, depending on the size of the structure and how much relaxation is needed. The example relaxation used here took ~20 min

▲ **CRITICAL** There are two dynamic simulation options available in oxView. The first is an RBD simulation engine that runs in-browser, while the second, oxServe, runs oxDNA on a connected server and displays live results. RBD is a simplified model (described in detail in ref. ²⁵) that treats each cluster of nucleotides as a sphere that encompasses the entire cluster, with connections between spheres modeled as flexible springs. The RBD model does not correspond to any physical motion of the structure and is intended for structure relaxation purposes only, while the oxServe option runs an actual molecular simulation of oxDNA model. Both methods are also illustrated in Supplementary Video 1. External forces, in the form of mutual monomer traps, can be added through the oxView interface to influence the dynamics.

Both methods are part of 'relaxation', where nonphysical configurations output by design tools are prepared for simulation by moving particles into positions and orientations acceptable to the

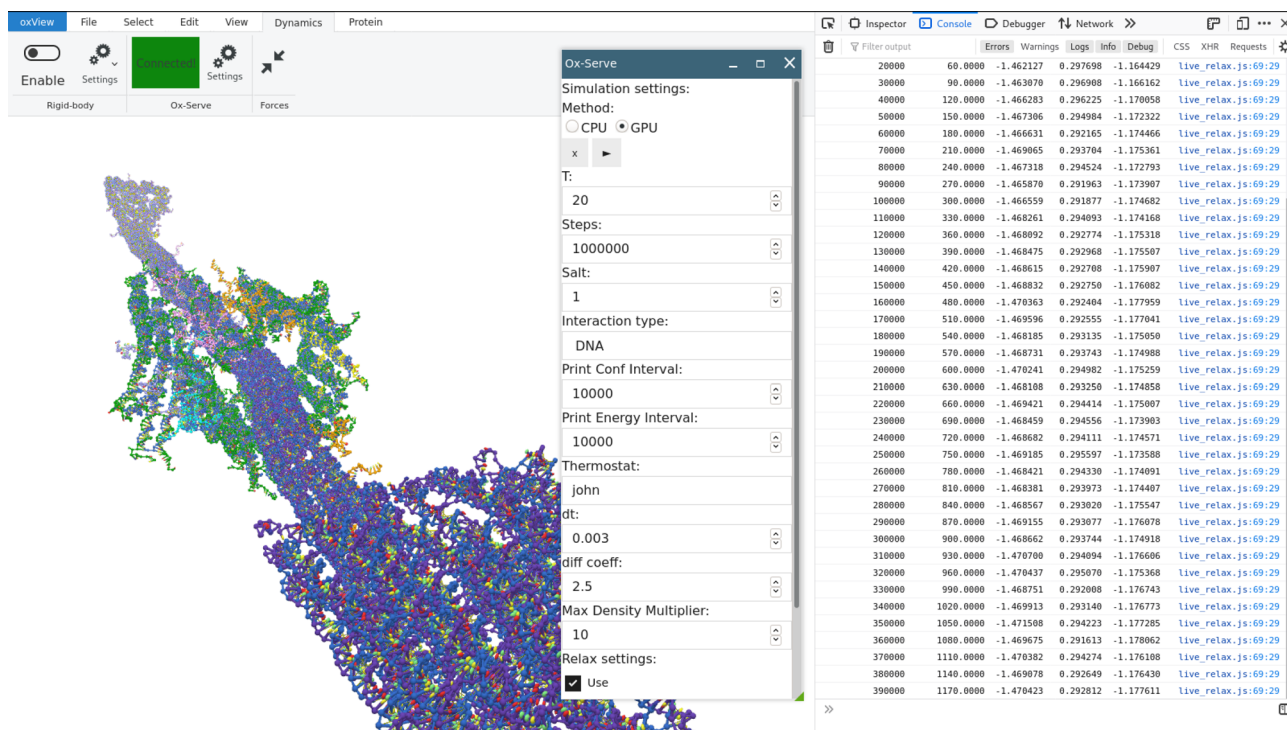


Fig. 4 | Interactive simulation in oxServe. Slider and rail caDNA designs from Benson et al.³⁰ imported and initially relaxed with RBD (as described in Procedure 1), here seen simulated using oxServe dynamics to relax the structure to a simulation-ready state. Note the simulation log in the browser console, which shows a potential energy between -1.4 and -1.5 , suggesting a well-relaxed structure.

oxDNA/RNA force field. RBD is first used to achieve correct orientation of groups of particles relative to one another, while oxServe allows the user to observe the results of oxDNA's relaxation force field in real time to verify that the trajectory is trending toward the intended structure.

Preparation

- 1 Open the file you would like to relax either by dragging and dropping the files onto the oxView window or via the Open button in the File tab. The steps outlined here are generic for any structure; however, for generating Fig. 4, we used the output files from Procedure 1 as the input to this procedure.
- 2 Add external forces.
 - Open the forces editor in the 'Dynamics' tab, where you will be able to (i) add mutual traps to selected particle pairs, (ii) automatically add mutual traps for all base pairs (this is helpful for avoiding duplexes being pulled apart during relaxation) or (iii) select and delete existing forces.
 - Current forces will be drawn as thin arrows in the 3D view.
 - Forces can be downloaded for use in later oxDNA simulations from the oxDNA download dialogue in the 'Save' section of the 'File' tab.

Rigid-body dynamics

▲ **CRITICAL** RBD is a simple simulation method in which clustered groups of particles are translated and rotated as a single rigid body that repels other groups of particles. Stretched backbones and mutual traps act as attractive potentials between clusters²⁵. RBD should be used when there are large blocks of particles that are knotted or incorrectly oriented in 3D space. The most common use case of this simulation method is relaxing imported 3D structures designed on a 2D caDNA lattice. The relaxation dynamics can also be used to facilitate free-form design, as exemplified in Procedure 2, where the tetrahedral helices are drawn on a plane and relaxed into the intended 3D shape. Applying RBD also makes it easier to spot certain errors in the designed shape in 3D, allowing the user to go back and update their design.

- 3 Make sure each intended rigid body is its own cluster by changing the coloring mode to 'cluster' under the View tab. Regions of the structure in red are not clustered and will not move during RBD. In general, each cluster should be internally close to ready for simulation, with connections

between clusters that are incorrectly oriented. Structures imported from caDNAno will already be clustered. Likewise, if you have duplicated or loaded multiple structures, they will already be separate clusters. Otherwise, you can use the clustering options (in the ‘Select’ tab) to either (i) automatically generate clusters through a density-based clustering algorithm (DBSCAN) or (ii) select and assign clusters manually, for instance, using the box selection tool.

? TROUBLESHOOTING

- 4 Enable the cluster dynamics. The clusters then repel each other through simple excluded volume interactions, while backbone connections or mutual traps between clusters act as springs.
- 5 Run the dynamics until you get the intended shape. Sometimes, you might need to ‘nudge’ the rigid bodies by translating or rotating them manually. Note that you will not be able to undo transformations done while the dynamics are enabled.
- 6 If the backbone bonds are still too extended, open the rigid-body settings dropdown and try lowering the cluster repulsion, increasing the spring constants or shortening the connection lengths. The shorter you get the bonds, the easier the following oxServe relaxation step will be. You may also be able to speed up the dynamics by increasing the dt (timestep length).

? TROUBLESHOOTING

oxServe dynamics

▲ CRITICAL This simulation option uses a web sockets connection to a separate host machine running an instance of oxDNA. The interface allows live visualization of a simulation as it is running, which is often used to monitor relaxation simulations to ensure that they do not ‘explode’ (when rounding errors compound owing to high forces/velocities and the kinetic energy of the simulation begins increasing exponentially and the structure completely falls apart) and are relaxing as intended.

- 7 Click the Connect button (in the ‘Dynamics’ tab), and input or select an oxServe host to connect to. There are three options for oxServe connections:
 - We offer a 1-GPU server as a public oxServe host at `wss://nanobase.org:8989`. This is the default option provided by the connection interface listed as `NANOBASE.ORG`. This should be the easiest to use for most users. Up to five connections are allowed at the same time.
 - You may set up and connect to your own server, following the instructions at <https://github.com/sulcgroup/ox-serve>⁴¹.
 - You may also use the oxServe code in a Google Colab notebook, as detailed on the GitHub page linked above.

? TROUBLESHOOTING

- 8 When connected, click Settings to open up the simulation options. The recommended workflow is as follows:
 - First, run a few Monte-Carlo steps (CPU) to make sure there are no overlapping nucleotides.
 - Then, run a Molecular Dynamics simulation (GPU) to relax the structure into a production-ready state.
 - Download the final configuration from the ‘File’ tab for use in your production simulation, either on `oxdna.org` or on your own cluster. We ask that researchers not run production simulations on the provided 1-GPU server as it is a shared resource intended for relaxation only. Simulation logs are written to the web console, as seen in Fig. 4.

? TROUBLESHOOTING

Procedure 4: external simulation and analysis of results ● **Timing** Hours to days, depending on the size of structure, length of simulation, and type of hardware. **Video creation and analysis** will take minutes to hours, depending on the size of the structure, the number of configurations in the trajectory and the speed of your computer.

▲ CRITICAL Once a structure has been designed and the initial relaxation performed using oxView, it can be used for a production oxDNA simulation. The specifics of oxDNA simulations are covered in detail elsewhere^{42,43}, but here in Procedure 4 we provide a brief protocol for exporting oxDNA simulations from oxView and running them on either an HPC cluster or `oxdna.org` and then analyzing the results. `Oxdna.org`²⁴ is a webserver hosted by our group that allows users to run and analyze equilibrium sampling simulations on a GPU server via a web interface. It is sufficient for sampling simulations to characterize DNA, RNA and protein nanostructures; however, for users who want specialized control over their simulations or to run many simulations, the web interface will not be sufficient, and users should run the simulations on their own HPC service. This procedure includes instructions for running oxDNA both from a command line and on `oxdna.org`.

- 1 Once a structure has been prepared in oxView (using the preparation methods from Procedures 1, 2, 5, 6 or 7 followed by relaxation using Procedure 3), download oxDNA simulation files from oxView by clicking the DNA button in the 'Save' section of the 'File' tab. If you also created a list of mutual traps using the forces interface, you can also download the oxDNA force file by selecting 'External Forces' in the dialogue.
- 2 If you are running simulation on your own HPC cluster, create an oxDNA input file that specifies simulation parameters and input/output options. Example input files can be found in the GitHub repository for oxDNA Analysis Tools and in the supplementary code from ref. ⁴². If you are using `oxdna.org` to run simulations, the input file will be automatically generated after clicking the Submit button at the end of filling out the job submission form.

If the simulation needs further relaxation or equilibration before the production simulation, the user can first run a shorter relaxation/equilibration simulation. A relaxation simulation often has the following modification of parameters from the example `input_run` file from oxDNA Analysis Tools:

- `max_backbone_force = 10`
- `dt = 0.0003`
- `external_forces = 1`
- `external_forces_file = <forces_file>`

Example relaxation files can also be found in the oxDNA Analysis Tools examples directory: one that runs a short Monte-Carlo simulation to separate overlapping particles, and a second file that runs a molecular dynamics (MD) simulation with a low `dt` setting. Both files use the `max_backbone_force` parameter that lets the simulation start with stretched backbones and also reduces the amount of force exerted by a stretched backbone so the simulation does not explode. The constant of integration, `dt`, determines the timestep between recalculating forces. In general, production simulations are run with `dt = 0.003`, and decreasing it by an order of magnitude can help reduce numerical instabilities due to the high forces and velocities that occur during relaxation. If you run your simulation on `oxdna.org`, a relaxation presimulation can be requested by checking the 'Needs relax' box in the input form. The number of steps and the timestep can be manually set in the 'View relax parameters' section.

- If running a relaxation simulation, it is recommended that you include artificial springs called mutual traps to nucleotides in the simulation to hold duplexes together against the forces exerted by stretched backbones. For details on creating these files using oxView, see Procedure 1. Mutual traps are defined by an additional text file with the following format:

```
{
type = mutual_trap
particle = p1
ref_particle = p2
stiff = 0.09
r0 = 1.2
PBC = 1
}
{
type = mutual_trap
particle = p2
ref_particle = p1
stiff = 0.09
r0 = 1.2
PBC = 1
}
```

where `p1` and `p2` are a pair of particles that need to be pulled together, and `stiff` is the spring constant of the potential. We recommend that 'stiff' be set to 0.09 for particles that are far apart and need to be joined and 3.0 for particles that are already bonded and need to be held together against the high forces generated during relaxation. By default, oxView produces files with a trap stiffness of 0.09; however, this can be modified by changing the stiffness option in the 'Forces' dialogue box before creating the trap (they cannot be edited once created, only deleted and re-created). The stiffness can also be edited after the file is created using find and replace in the text

editor of the user's choice. If you use `oxdna.org` for your simulations, a mutual trap file is automatically generated based on existing base pairs in the submitted structure.

- 3 Run the oxDNA simulation with the created input file. If running from a command line and oxDNA is added to the PATH, the command is:

```
oxDNA <inputfile>
```

Progress can be tracked by checking the header of the last configuration file to view the last timestep printed to the file. Energy information is also printed to stdout at an interval specified by the `print_energy_every` parameter in the input file.

If you use `oxdna.org`, the simulation will be run as soon as the simulation form is filled in and submitted and a GPU node becomes available. Progress can be checked by viewing the log file or by clicking on 'View last conf' on the jobs page. If the 'Needs relax' option was selected in the input file, a two-part relaxation is run. First, a short Monte-Carlo simulation is performed to remove excluded volume clashes followed by a longer molecular dynamics simulation with `max_backbone_force` set to further relax the structure. When the simulation completes, you can move on to creating a video of the simulation and analyzing the simulation trajectory.

Create a video of the simulation

- 4 Align the trajectory. To remove rotational degrees of freedom (oxView already automatically handles periodic boundary conditions and removes translations), the trajectory needs to be aligned. Alignment can be performed using the `align_trajectory.py` script from oxDNA Analysis Tools using the following command from the command line, assuming the package has been installed:

```
oat align_trajectory trajectory.dat aligned.dat
```

This will produce a new trajectory file, `aligned.dat`, with all rotations and translations removed. This file is also smaller than the original trajectory as the alignment script also removes velocities, so it is generally advised to perform this step before downloading if you are performing the simulations on a remote cluster. On `oxdna.org`, alignment can be performed using the 'Align Trajectory' section on the alignment page. The file will also be compressed to make downloading faster. Because of the way oxView streams files, the opened files must be local to the computer running the web browser, so the aligned trajectory must be downloaded from `oxdna.org` to your local computer before viewing it.

? TROUBLESHOOTING

- 5 Load the trajectory in oxView by dragging and dropping the aligned trajectory and the topology at the same time
- #### ? TROUBLESHOOTING
- 6 Click the 'Create video' button in the 'File' tab. This will open the video creation dialogue. The following video options are available:
 - Video type: Trajectory or XY Lemniscate. Trajectory will run through the trajectory and save each configuration as a frame. XY Lemniscate will move the camera in a figure eight around the current camera center (defaults to the origin) to show a 3D view of a single configuration
 - File format: webm, gif, png or jpg. The format in which to save the video. Webm video encoding is currently only available in Google Chrome. Gif and png/jpg image series are available in any browser. Image series can be converted to the video format of the user's choice using many free image/video manipulation tools such as ImageJ or FFmpeg
 - Framerate: the number of frames to show per second if the video is in WebM or gif format
 - Start: start the video creation with the chosen parameters
 - 7 After setting parameters, start the video capture by clicking 'Start'. This will automatically progress the trajectory or the camera motion and capture each image as a frame. If you move the camera during this process, it will be captured in the final video. A progress log will be printed in the browser console as the video creation progresses.
 - 8 When the video is ready, it will be downloaded by your browser. This video file can be quite large as it is not optimized. It is generally recommended to use some sort of video compression software to reduce the file size and change the file format to something more universally usable, such as mp4.

Analyze the simulation

▲ CRITICAL As this protocol is primarily focused on oxView, we will keep this section brief, with further discussion in Supplementary Information. The only analysis tool natively built into oxView is a distance finder. Distances can be calculated using the ‘Distance’ dialogue box. Select two particles and click the ‘Create from selection’ to view the distance between the selected particles. Most other common analysis types can be handled by the Python package, oxDNA Analysis Tools, which also has libraries of Python functions for reading, writing and managing oxDNA files to help users develop their own custom analyses. See the Supplementary Information for details on developing custom analysis scripts using the utilities offered by the package. As an example, here we demonstrate how to compute the mean structure and per-particle root mean square fluctuation (RMSF) of a simulation trajectory and visualize the results in oxView. We note that some of the scripts from oxDNA Analysis Tools are available through the GUI on the analysis page of the `oxDNA.org` server, and such analysis can be performed interactively on the `oxDNA.org` server after the simulation finishes. Trajectories can also be downloaded from `oxDNA.org` and used as input for the Python scripts as explained here.

- Run the Python script to obtain the mean structure, the per-particle RMSF values and the global root mean square deviation (RMSD) values for each configuration in the trajectory:

```
oat compute_mean -p 4 -d devs.json trajectory.dat
```

This will produce four files:

- `mean.dat`: a configuration file containing the mean position of each particle
- `devs.json`: an oxView color overlay file containing the RMSF of each particle in the simulation from the mean
- `devs_rmsd.png`: a graph showing the RMSD of each configuration in relation to the mean structure
- `devs_rmsd_data.json`: an oxView order parameter file containing the RMSD of each configuration in relation to the mean structure

? TROUBLESHOOTING

- Load the mean structure with the RMSF overlay by selecting the mean structure, the topology file and the deviations JSON file and dragging and dropping all of them at the same time onto the oxView window. This will show the mean structure with coloring in overlay mode, allowing the user to interactively visualize the flexibility of the structure

? TROUBLESHOOTING

- View the RMSD over the trajectory. Refresh the oxView page, and this time select, drag and drop the aligned trajectory and the topology. The ‘Trajectory’ tab should automatically open when the structure is loaded. Wait for indexing to complete, and click the ‘Order parameter selector’ button. Drag and drop the `devs_rmsd_data.json` file onto the order parameter dialogue box. This will show a graph with the RMSD plotted against the configuration number. Click on any node on the graph to jump to that configuration or press the left and right arrow keys or the play button on the trajectory tab to begin stepping through the trajectory.

Procedure 5: scripting in the browser console ● **Timing** The example scripts will have different execution time, depending on the system size; the examples here range from seconds to >30 min

- Load a structure that you want to edit into oxView.
- Open a JS developers console. It is usually accessible through the browser menu or through a keyboard shortcut, which varies by browser and operating system.

? TROUBLESHOOTING

Paste the script into the opened console, and press Enter. Note that, for some scripts, this can take a long time and you might get a notification from your browser that this tab is slowing down your browser. Do not stop the page; let it complete. Boxes 2–4 describe examples of three common use cases with results shown in Fig. 5. The designs are available in our online example repository. We provide more examples (including coloring a list of particles, printing the sequence of selected stands and visualizing the center of mass for a provided trajectory) and a more extensive introduction to the JS API in the Supplementary Information (Listings S8–S14).

Box 2 | Extending all 5' ends of a provided structure with single-stranded overhangs

Recently, Gopinath et al.⁴⁷ used single-stranded extensions of every staple strand in the structure to control its landing orientation on a silica wafer. Designing such a structure using conventional design tools like caDNAno is very tedious but is easily accomplished using the oxView editing API. Shown here is the code needed to modify the 'Death Star' design (Fig. 5a) to include a 25 T overhang on every staple strand (Fig. 5b)

```
function extend_overhang(strand) {
  if(strand.getLength() < 150) { // make sure we are extending only staples
    // select 5' end of the strand
    const p5 = strand.end5;
    // fetch 5' orientation vector
    const a = p5.getA1();
    // extend the strand with provided sequence
    const bases = edit.extendStrand(p5, "TTTTTTTTTTTTTTTTTTTTTTTTTTTTT");
    // figure out offset position
    const npos = new THREE.Vector3().copy(a).multiplyScalar(-1.5);
    // translate the extension bases in the desired direction
    bases.forEach(b => b.translatePosition(npos));
    // rotate the extension bases 90 degrees away from the structure plane
    rotateElements(new Set(bases), bases[0].getA2(), Math.PI/2, bases[0].getPos());
  }
}
// apply extension function to all strands in origami
systems[0].strands.forEach(extend_overhang);
// update the screen
render();
```

Box 3 | Visualizing a gold nanoparticle

Many nucleic acid structures contain different types of functional modifications such as fluorophore molecules or metallic nanoparticles. While oxDNA does not provide a way to simulate these modifications to the design, oxView can visualize these objects using standard Three.js functions to create a reasonable approximation. The following code creates a yellow sphere with a radius of 10 nm at position (-10, 30, -10), to represent a gold nanoparticle. We use the 'Death Star' design with 25 nt extensions as a demonstration (Fig. 5c).

```
// conversion from oxDNA to nanometer
const nmToOxDNA = 0.8518;
// compute radius of 10 nanometers
const radius = 10 / nmToOxDNA;
// Three.js functions to create a sphere
const geometry = new THREE.SphereGeometry(radius, 32, 16);

// Material defines the color of the Mesh (we use yellow)
const material = new THREE.MeshPhongMaterial({color: 0xffff00});
const sphere = new THREE.Mesh(geometry, material);
// move the sphere into position
sphere.position.set(-10, 30, -10);
// add to scene
scene.add(sphere);
// update the screen
render();
```

Box 4 | Constructing a crystal cluster from a provided origami design

oxDNA, being a coarse-grained model, is efficient enough to study crystal systems like the ones presented in Tian et al.⁴⁸. Figure 5 explains the logic to set up a cluster of 3 × 3 × 3 octahedral units connected in a cubic lattice. The octahedral unit is displayed in Fig. 5d, and the resulting primitive cubic lattice is shown in Fig. 5e. RBD is subsequently used to relax the stretched bonds and orient the duplex patches connecting the octahedral origami (Fig. 5f). As the system is very large, visualizing it during the relaxation slows the calculations down significantly, so it is recommended to hide all visible components prior to starting RBD and oxServe relaxation. The full JS code is provided in the Supplementary Information (Supplementary Listing S14).

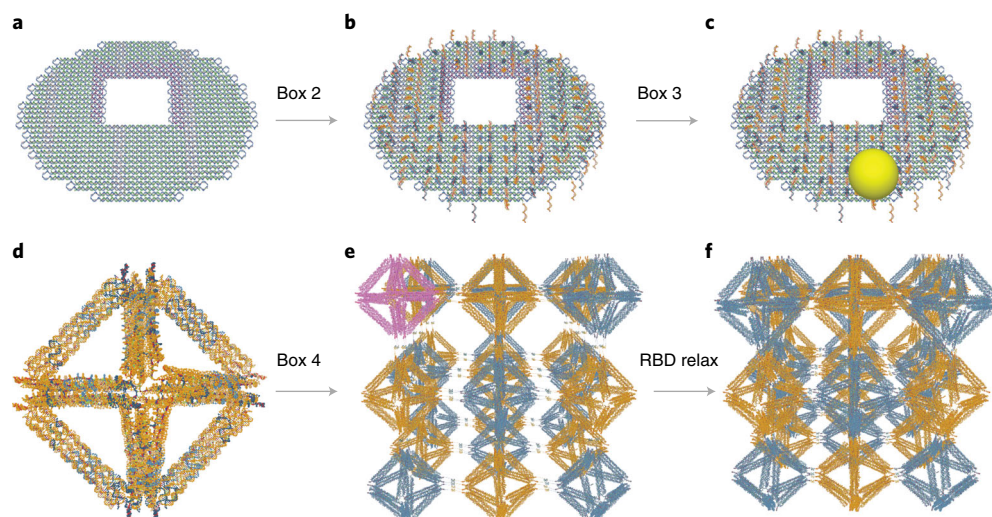


Fig. 5 | Examples of using the oxView scripting interface. **a**, ‘Death Star’ structure from Gopinath et al.⁴⁷. **b**, Applying the 5’ extension script from Box 2 to the ‘Death Star’ structure. **c**, Using the same structure to visualize a 10 nm gold nanoparticle (Box 3). **d**, Octahedron DNA origami structure from Tian et al.⁴⁸. **e**, A $3 \times 3 \times 3$ cluster of octahedral origamis, generated using the code in the Supplementary Information (Listing S14). **f**, Subsequent relaxation of the cubic lattice using RBD, to get a cell with less stretched bonds and orient the patches connecting the octahedra.

▲ CRITICAL STEP Note that the scripts are editing the currently loaded scene in the browser memory, so a scene must first be visualized in order to be edited. However, some draw operations can be costly; sometimes you can get a small amount of speedup by turning off the representations of some or all of the visualized particle components in the ‘View’ menu.

? TROUBLESHOOTING

Procedure 6: protein-nucleic acid hybrid structure visualization and design ● **Timing** The loading time for PDB files is strongly dependent on system size and composition. Loading PDB structures will take seconds at most, and parameterizing ANMs can take up to 10 min

Load a PDB file

- To load a PDB file, simply drag and drop the file onto the oxView window. oxView’s PDB file reader will import all amino acids and nucleic acids in the PDB file into the oxView scene. As PDB files are abundant and common, our users have a great starting library of proteins and nucleic acid structures. Additionally, oxView can load malformed PDB files (e.g., entirely missing chain information, duplicate chain identifiers, missing chain/model termination, two character chain identifiers, five digit residue numbers and six digit atom serial numbers) that are often present in the PDB databank as well as in popular programs, namely CHARMM-GUI⁴⁴, GROMACS⁴⁵, MODELLER⁴⁶ and many others. oxView can correctly load malformed PDB files generated by the aforementioned programs, large PDB files (<300 MB) containing biological assemblies and solution NMR PDB files (Fig. 6). Note that these files can be very large and take time to parse and convert into an oxDNA representation.

? TROUBLESHOOTING

Generate files to run an ANM-oxDNA model simulation with a loaded protein

▲ CRITICAL For details on the ANM-oxDNA model, see information in Box 5. Supplementary Video 2 also outlines these steps.

- Select all amino acids in the protein by switching the selection mode to ‘System’ under the ‘Select’ tab and clicking any amino acid displayed in the window.
- Select ‘Create Network’ in the ‘Protein’ tab to create a network object whose nodes are the selected particles.

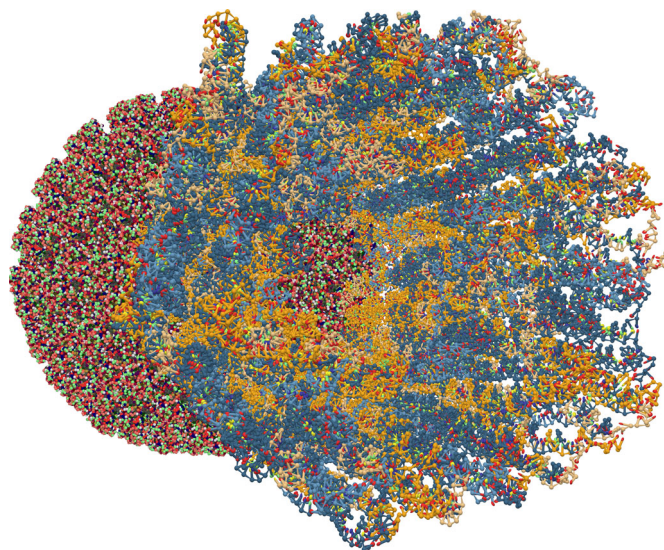


Fig. 6 | oxView visualization of a bluetongue virus core and a DNA origami barrel. The barrel is from Kube et al.⁴⁹, and both structures are imported from their PDB databank entries (2btv and 7as5, respectively).

Box 5 | Introduction to the oxDNA protein model

In addition to nucleic acids, oxView also supports protein visualization for design of protein–nucleic acid hybrid nanostructures. The protein representation is based on our ANM–oxDNA model where each protein residue is represented as a single spherical particle connected to other residues within a specified cutoff distance by a harmonic potential, an ANM. To run ANM–oxDNA simulations, the global spring constant (k) is fit to best reproduce the protein fluctuations (quantified by experimentally measured B factors) and provided along with the network topology as an external parameter file to the oxDNA simulation. For further details on the ANM–oxDNA model and its specifics, see Procyk et al.²².

The intended purpose of oxView’s protein visualizer is to both guide DNA nanostructure design around protein targets (such as a nanostructure binding to a virus, illustrated in Fig. 6) and to construct hybrid protein–DNA/RNA nanostructures for simulation. To expand oxView’s usability and accommodate the ANM–oxDNA model more completely, we also implemented a parameter file generator in oxView. Note that running protein simulations requires a unique branch of the oxDNA code that can be found at <https://github.com/sulcgroup/anm-oxdna>. Detailed instructions on installing and compiling this version of the code are included in the Supplementary Information. In Procedure 6, we describe how to import a protein structure into oxView and generate files to run simulations.

- 4 Construct the edges of the network. Define the cutoff distance in Angstroms inside the ‘Edge Cutoff’ box of the ‘Protein’ tab, then select ‘Fill Edges’ and, finally, ‘Cutoff (ANM)’. Our ANM is now complete: all particles within the specified cutoff distance are now connected to one another.
- 5 Visualize the network. Select the network you wish to visualize in ‘Select Network’. Then select ‘View Network’. To toggle the network visualization off, click ‘View Network’ again.

? TROUBLESHOOTING

Fluctuation solver

▲ CRITICAL The following steps solve for the network’s mean square fluctuations and linearly fit the predicted B factors of the network to the experimentally determined B factors of the protein. The ANM model aims to reproduce the equilibrium dynamics of a protein by fitting to B factors that describe the thermal fluctuations of each residue in the protein’s crystalline (often near-native) state. By direct evaluation of a user-defined ANM, we can solve for the predicted B factors of the ANM and fit the global constant (k) to best match the experimental B factors²². We can directly parameterize small protein systems (<1,500 residues) for simulation in our ANM–oxDNA model without leaving oxView. For larger

proteins, we point the reader to our Python scripts available at <https://github.com/sulcgroup/ANM-oxdna>, discussed further in the Supplementary Information.

- 6 Click 'Fluctuation Solver' in the 'Protein' tab to bring up a graphing window where per-residue B factor/mean squared fluctuation data is displayed. B-factor data are automatically loaded from PDB files to the 'Available Datasets' tab. Additional B-factor datasets can also be imported using the file dialogue in the window.
- 7 Set the temperature in the 'Temperature' box to the experimental temperature at which the B factors were obtained. This integer value should be in Kelvin and can be found in the header of most PDB files.
- 8 Select the dataset of the protein to fit the network to by clicking its entry under 'Available Datasets'. When the dataset is selected, the graph will update and show the per-residue fluctuation data of the selected dataset.
- 9 Solve the network. Select the network under 'Fitting Ready Networks' to start a web worker that builds the Hessian matrix of the network and calculates its pseudo-inverse to calculate the B factors of the given network. The global spring constant of the ANM model k is fit by linear regression with no intercept of target B factors to network B factors. When the webworker is finished, a new dataset will appear in the fluctuation window.
- 10 Save the network for simulation. Graphs can be saved as .png by selecting the camera icon in the 'Fluctuation Solver' window, B-factor datasets can be downloaded as JSON files by selecting download datasets in the 'Fluctuation Solver' window and simulation files with the now parameterized spring constants can be downloaded by selecting 'Simulation Files' under the 'Files' tab.

Procedure 7: design of protein–DNA hybrid cage ● Timing Variable, depending on available hardware

▲ **CRITICAL** Overall, the procedure to build the protein–DNA hybrid cage from Xu et al.³⁴ takes ~30 min in a single sitting. Simulation and computation time can differ greatly, depending on the hardware available.

Prepare the DNA component

- 1 Select the TacoxDNA icon under the 'File' tab. Choose the input file as the Tiamat design of the DNA cage in dnajson format, file format as Tiamat version 2, nucleic acid type as DNA and default base as random.
- ? **TROUBLESHOOTING**
- 2 Click Import and Load. The Tiamat design will be displayed in the viewer. Your oxView screen should match that of Fig. 7a.
 - 3 Go to 'Dynamics' and select 'Forces'. These strands are already base paired, but stretched bonds need to be relaxed, so the forces should be relatively strong. Set 'Stiffness' to 3.1 followed by 'Create Forces from Basepairs' to generate forces to relax the DNA cage.
 - 4 Close the forces window by again selecting 'Forces' under 'Dynamics'.
 - 5 Using oxServe (described above in Procedure 3), launch an MC CPU job followed by an MD GPU job to relax the DNA cage. While using oxServe to relax the cage, your screen should match that of Fig. 7b.
 - 6 Save the configuration, topology and force files by clicking the DNA icon in the 'Save' section of the 'File' tab. Select 'Topology', 'Configuration' and 'External Forces' in the dialogue and click export to save the structure and forces we generated previously. We will need these files to relax the full system later. Go ahead and rename it 'dnaforces.txt'.
 - 7 Refresh the window, then reload the system by dragging and dropping the topology and configuration files downloaded in the previous step.

Prepare the protein component

- 8 Go to <https://www.rcsb.org/> and download the desired protein, in our case KDPG Aldolase (PDB ID: 1EUA) in PDB format.
- 9 Drag and drop the PDB file onto the oxView window to load the protein structure into the scene.
- 10 Select the protein. Go to the 'Select' tab and select 'System' in the top left corner. Now, click any amino acid to select the entire protein.
- 11 Position the protein so it is not overlapping with the DNA cage. Use 'T' and 'R' (or Translate and Rotate under the 'Edit' tab) on the keyboard to switch between translation and rotation mode and position the protein away from the DNA cage. Your screen should look similar to Fig. 7c.

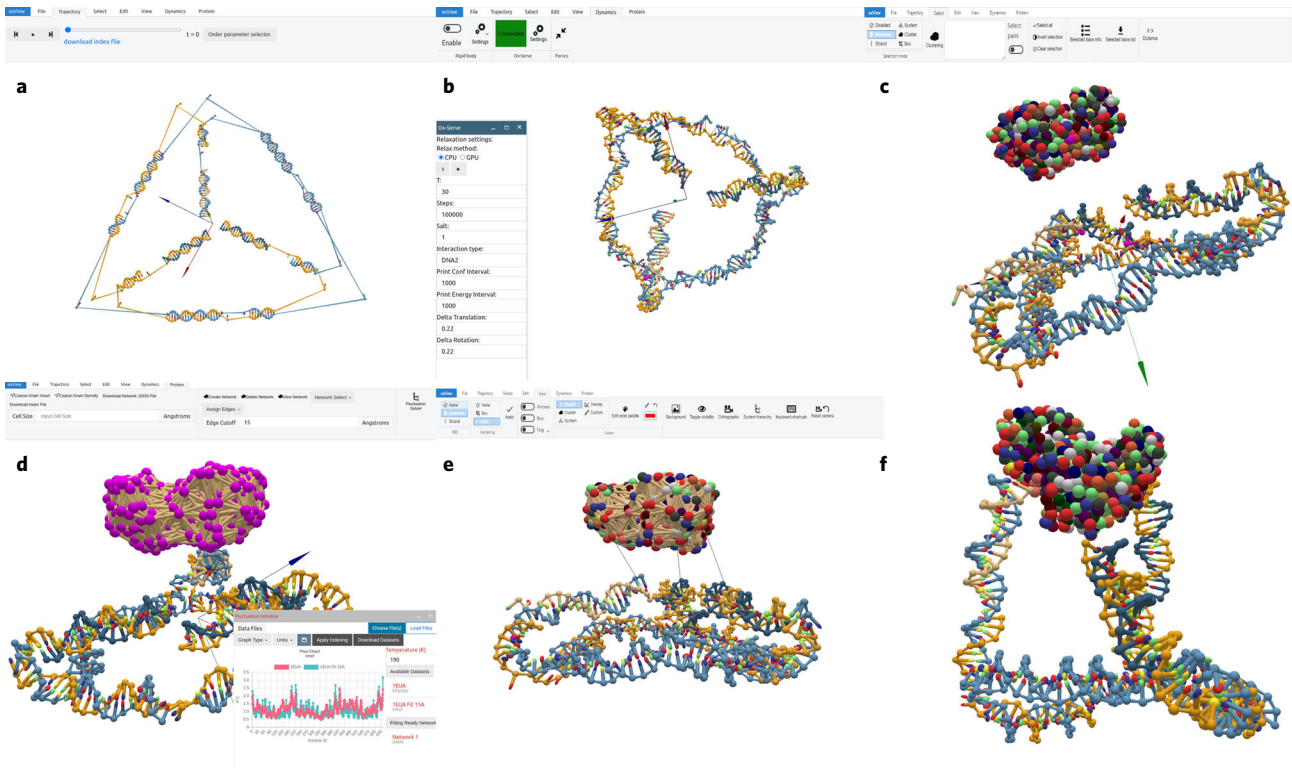


Fig. 7 | Various stages of construction of a protein-DNA hybrid in oxView. a, The unrelaxed imported Tiamat DNA cage design. **b**, oxServe relaxed DNA cage. **c**, Addition of protein component via PDB. **d**, Fitting an ANM to the protein (PDB ID: 1eua). **e**, Defining linker forces between the DNA and protein components. **f**, The fully relaxed structure, ready for simulation with ANM-oxDNA.

Make an ANM

- 12 With still only the protein selected, navigate to the ‘Protein’ tab and select ‘Create Network’. This initializes a network where nodes are the selected particles and edges are not yet defined.
- 13 To construct the ANM edges we first define the cutoff distance, typically 12–18 Angstroms, in the box titled Edge Cutoff. After typing in 15 (Angstroms) into the ‘Edge Cutoff’ box, select ‘Assign Edges’ and click ‘Cutoff (ANM)’ in the submenu.
- 14 Select ‘View Network’ in the Protein tab to view the network connections between particles. Your protein will look similar to the one depicted in Fig. 7d.

Fitting the ANM to B-factor data

- 15 Open the fluctuation solver by clicking ‘Fluctuation solver’ in the top right corner of the ‘Protein’ tab.
- 16 Select ‘1eua bfactor’ under Available Datasets. This displays the B-factor data of the ‘1eua.pdb’ as the mean squared fluctuation (msf) in Angstroms squared of each alpha-carbon atom in the protein.
- 17 Set the temperature in the ‘temp’ box to the temperature at which the B-factor data were obtained. In our case, we will set this to 190 K.
- 18 Solve the network. By selecting ‘Network 1’ under ‘Fitting Ready Networks’, a web worker will be launched to fit our defined ANM to the PDB B-factor data. This step can take some time, depending on the system size and cutoff used. Progress can be viewed in the console, and notifications are generated for the start and end. When it is finished, it will also generate a new dataset under Available Datasets titled ‘1eua Fit 15A’.

? TROUBLESHOOTING

- 19 View the B factors of the ANM. Select 'Ieua Fit 15A' under 'Available Datasets' to see the B factors of the network on the graph.
- 20 Save the parameterized ANM with the DNA cage. To save into a simulation-ready state, go to 'File' and select the DNA icon in the 'Save' section to download the topology, configuration and parameter files. Keep this oxView window open as we still need to define our forces in Steps 21–28.

Preparing combined system

- 21 Find the conjugated residues. In our experimental system, the protein was physically conjugated to the DNA cage at residue 52 cysteine (N terminus to C terminus) in all three sites of the trimer protein with an Ic-SPDP linker. This linker was previously simulated in fully atomistic resolution using GROMACS and approximated as a spring potential by matching the mean and standard deviation of the end-to-end length. With the protein in our viewer, open the browser console and type:

```
api.selectPDBIDS([52])
```

to select all residues with 52 as their PDB residue number. On KDPG Aldolase, we see three cysteine residues selected. Either make a mental note of the positions or use the custom coloring feature in the 'View' tab to highlight them.

- 22 While holding Ctrl, iteratively select each DNA nucleotide (the IDs should be 97, 195 and 293) followed by the nearest of the cysteine residues labeled in Step 22. Make sure the selection mode is 'Monomer' in the 'Select' tab during this step.
- 23 In the 'Dynamics' tab, select 'Forces' and select 'Create From Selection'. Our forces are now displayed in the forces window.
- 24 Download the forces file by clicking the DNA icon in the 'Save' section of the 'File' tab and selecting only 'External forces' in the dialogue. Rename the file 'linkerforces.txt'.
- 25 In 'linkerforces.txt', find and replace the default r0 and stiffness values (1.2, 0.09) of each mutual trap to match our linker approximation (1.071, 1.424) from ref. ²².
- 26 Delete the forces in the oxView window by selecting the left top corner checkbox followed by selecting 'Delete selected'.
- 27 Load the previous force file 'dnaforces.txt'. To ensure compatibility with the merged topology of the protein and DNA component, we must load in the forces we generated for the DNA component and export them for the merged topology.
- 28 Download the forces file for the new topology by selecting the file and clicking the DNA icon in the 'Save' section of the 'File' tab and selecting only 'External forces' in the dialogue. Name it 'relaxforces.txt'.

Relax/simulation of hybrid system

- 29 Double check that the system has been created correctly by refreshing the page to clear it and then loading the topology and configuration generated in Step 20 followed by the parameter file and 'linkerforces.txt'. Your system should look similar to Fig. 7e.

? TROUBLESHOOTING

- 30 Prepare a combined forces file. Concatenate the contents of the two forces files 'relaxforces.txt' and 'linkerforces.txt' into a new file named 'allforces.txt' using either copy-and-pasting in text editors or using the command line.
- 31 Prepare a relaxation directory. Gather all force files and the generated topology, configuration and parameter files from Step 2 into one directory.
- 32 Relax the system. Using either oxDNA directly or oxdna.org, first launch an MC job followed by a CUDA MD job to relax the structure using the 'allforces.txt' force file. The fully relaxed system will look similar to Fig. 7f.
- 33 Simulate the system. Using just the 'linkerforces.txt' as the external force file and the fully relaxed structure as the configuration file, we can now simulate our fully formed hybrid DNA–protein structure. This simulation's input file should have the following modifications from the standard input files:
 - `interaction_type = DNANM`
 - `dt = 0.002`

- `external_forces = 1`
- `external_forces_file = linkerforces.txt`

We use a slightly lower dt for ANM-oxDNA model simulations as proteins tend to be much denser than DNA and excluded volume interactions between amino acid residues can cause numerical instabilities if dt is too large.

▲ CRITICAL STEP Thanks to its use of the Three.js framework, it is easy to integrate additional visualization features into oxView. Two such features that can help public outreach activities by creating more interactive visualization options are the ability to export to common 3D formats (Procedure 8) and to view structures in VR (Procedure 9).

Procedure 8: export for 3D printing or advanced rendering ● Timing Exporting 3D files is usually very quick and should take at most a few minutes for large files. Importing into Blender can take over an hour for large files.

- 1 Optionally, customize components and scaling. In the ‘View’ tab, click the ‘Visible components’ button.
 - Select components to include in the export. For example, you may want to disable the nucleoside and connector meshes to only export the backbone shape
 - Set component scaling. An increased scaling can, for example, be used to make structures easier to 3D print
- 2 In the ‘File’ tab, click ‘3D Export’.
- 3 Select export format:
 - STL saves the shape as a mesh and is a common format for 3D printing
 - glTF saves additional information such as colors and materials, while still producing a smaller file size. Use this when you want to render your structure in an external 3D graphics program such as Blender
 - GLB is a binary version of glTF and can be used the same way. Being a binary format, GLB files are smaller in size but no longer human-readable
- 4 Select ‘additional options’:
 - Set material properties for backbone and nucleoside components
 - Set faces multiplier. A higher value will result in more faces per mesh, smoother surfaces and a larger file size
 - Enable flat hierarchy (only relevant for glTF). If enabled, the glTF structure will not keep oxView’s hierarchical structure of systems, strands and monomers. This will avoid the use of empty objects but will make the exported files harder to edit
- 5 Select a name for the output file.
- 6 Click the export button.
- 7 (optional) Import glTF into Blender:
 - In Blender (v. 2.9), click ‘File’ and ‘Import’ and select ‘glTF 2.0’
 - Navigate to and select your glTF file exported from oxView
 - Wait for the file to load; if you have started Blender from a terminal, you will see progress messages there
 - Select and remove the default cube
 - Position the camera, lights and other potential elements of the scene. You might want to scale the imported file by a factor of 0.1 to be more proportional to the Blender Defaults. For physically based rendering, select the Cycles renderer
 - Select ‘Render’ and ‘Render Image’ to start rendering.

? TROUBLESHOOTING

Procedure 9: VR visualization ● Timing Instantaneous

- 1 Open the oxView web app (oxview.org) using a WebXR-compatible device.
 - 2 Load a structure into oxView as described in Procedures 1 or 2.
 - 3 In the ‘View’ tab, click ‘Enable VR’.
 - 4 If your device is supported, click the ‘ENTER VR’ button at the bottom of the screen.
- ? TROUBLESHOOTING**
- 5 If you have a trajectory loaded, you can click your controller to step through the trajectory.

Troubleshooting

Troubleshooting advice can be found in Table 1.

Table 1 | Troubleshooting table

Procedure	Step	Problem	Possible reason	Possible solution
1	1	Import is taking a very long time	Converting from caDNAno is a very computationally intensive operation as all particle positions have to be calculated based on numbering in virtual helices	Likely, you can just wait a few more minutes. Check the web console for detailed output or possible errors
2	3	Notification saying ‘Please select one nucleotide with an available 3’ connection and one with an available 5’	The selected nucleotides have to be one 3’ end and one 5’ end	Check the strand directionality to make sure you have selected the correct ends to ligate. The diameter of the backbone decreases toward the 3’ direction. Toggle 3’ markers to make them more visible
3	3	Browser reports that the page is slowing down the browser	DBSCAN clustering can take a long time for large structures	Wait and do not hit the ‘Kill’ button the browser offers. If there is an error, oxView will notify you or the tab will crash. If it appears frozen, it is just thinking. Optionally, you can assign clusters manually
	6	The clusters just end up really far from each other	Elongated structures, for example the linear actuator rail from Procedure 1, are not well suited for RBD. This is because the repulsion pushes away everything within the maximum radius of the cluster	You can try tweaking the cluster dynamics settings, but in some cases it is easier to use the manual tools to move the clusters around
4	7	Notification: lost oxServe Connection	The connection to the oxServe server was lost (or could not be established)	Try connecting again, and check the web console for further errors. If that does not help, try another server or, if you manage the server yourself, make sure it is running
	8	oxServe relaxation was started, but nothing is happening	Communication with an oxServe server can sometimes be quite slow	Check the web console, as seen in Fig. 4, to see if there are any reported errors. If not, give it a bit more time
	2	Particles with forces between them do not end up pulled together	The external forces are not strong enough to overcome internal forces or something is incorrect in your trap file	Verify that you have put traps in both directions (e.g., p1 → p2 and p2 → p1). Try increasing the stiff parameter by 5× or 10×. Note that going too high can cause the simulation to explode
	3	Simulation does not start and reports an error message	Something is incorrect in the input file or oxDNA is not correctly compiled for your system	Check the error message for possible solutions. If you are using one of the premade input files from oxDNA Analysis Tools, all the parameters are correct, so double check that you have correctly specified your topology and configuration files. If you get an error from CUDA, ensure that you are running oxDNA on a computer with an NVIDIA GPU and CUDA installed and that your current CUDA version is the same as the one you used to compile oxDNA. For more information on compiling oxDNA, see dna.physics.ox.ac.uk/index.php/Download_and_Installation
4		Simulation ends with ‘Segmentation fault’	The structure was not sufficiently relaxed. Unrelaxed structures will explode during the simulation, resulting in numerical errors that crash the simulation engine	Run relaxation for longer. You might also need to decrease the dt of the relax simulation to make the relaxation more gentle. A relaxed structure where most nucleotides are bonded will have a potential energy between 1.4 and 1.5 oxDNA energy units. For more information on relaxation, see ref. ⁴²
	4	Video does not download	Video encoding, particularly if you made a gif, can take a long time	Wait a bit longer or check the browser console for errors. Make sure you do not open the browser console while the scene is still stepping or the change in image size will crash the encoder. It is fine to open the console before

Table continued

Table 1 (continued)

Procedure	Step	Problem	Possible reason	Possible solution
				starting video creation or while the postprocessing is occurring
	4, 9	Python reports an error	The most likely source of errors is missing libraries in your Python environment	Run oat config. This will check your environment and inform you if anything needs to be updated
	5, 10	Browser opens the dropped files in tabs instead of loading them into oxView	The browser did not capture the drop event	If your computer is a bit slow, it can help to move your mouse around for a second on the oxView window before dropping files to make sure that the browser registers the drop event correctly
5	1	Browser reports that the page is slowing down the browser	Large structures (>30,000 nucleotides) might take longer to be displayed	Wait and do not hit the 'Kill' button the browser offers. If there is an error, oxView will notify you or the tab will crash. If the browser appears frozen, oxView is just thinking
	2	Browser console is not opening	Depending on the browser and operating system you are using, the steps required to open the JS console might be different	Search your browser's manual page for 'how to start the javascript console'
	3	Script is refusing to run	Possible error in copying the script Changes to the scripting API	Double check that just the relevant JS code is copied for execution. Line numbers are not part of the JS syntax While oxView is written in a highly backward compatible manner, some new features might change the scripting interface. See the GitHub example pages for the newest version of the script code and documentation
6	1	PDB file will not load, or loads partially/incorrectly	Residue names in the PDB file do not follow the PDB databank Seqres format outlined at https://www.wwpdb.org/documentation/file-format-content/format33/sect3.html#SEQRES . Failure to format residues/nucleotides with their correct names results in errors about unrecognized residues in the console output and missing residues in the oxView display	Reformat your PDB file to contain accepted residue names only
	5	Clicking 'Visualize Network' displays nothing	The edges of the network were most likely not constructed	Repeat Step 2, and make sure no error message is produced
7	1	Browser reports that the page is slowing down the browser	Converting from Tiamat can be a very computationally intensive operation as all particle positions have to be calculated	Wait and do not hit the 'Kill' button the browser offers. If there is an error, oxView will notify you or the tab will crash. If the browser appears frozen, oxView is just thinking
	18	No networks shown under 'Fitting Ready Networks'	Error in network creation. When networks' edges have been constructed, the network should automatically propagate to the 'Fluctuation solver' window	Delete current network using 'Delete Network', and repeat Step 2. Successful network visualization will indicate that you completed the steps correctly
	29	The forces arrows drawn when reloading 'linkerforces.txt' connect to the incorrect nucleotides and/or amino acids	Edits to the DNA topology can change the indexing of nucleotides in the system, which results in incorrect references to particle IDs in the force file	Reload the downloaded system files from Step 2 and remake the forces files following Step 3
8	7	The exported file takes a long time to load in Blender	The blender glTF importer is very slow for large files	Keep waiting and the structure should load eventually. You may want to disable some components (in Step 1) to decrease the file size
9	4	The button says 'WEBXR NOT SUPPORTED'	The device you are using does not support WebXR	Make sure to read the documentation for your headset. If you are using an Android phone with Google Cardboard, make sure to use the Chrome browser

Timing

Procedure 1

Step 1, The conversion takes ~10 s.

Step 2, The conversion takes ~30 s.

Steps 3–5, If you just want to simulate the slider diffusing along the rail, it will take a few seconds to position the slider such that it does not overlap the rail overhangs. If you want to simulate the attachment between the rail and the slider, it will take ~10 min to set all the sequences and build mutual traps.

Procedure 2

Steps 1–3, For the demonstrated structure this will take ~5 min to create and position all the strands. Time will scale with the complexity of the design.

Step 4, For the demonstrated structure, RBD took <1 min. It will take longer if you draw a more complex structure or have more topological entanglements that need to be resolved.

Step 5, Adding spacers takes a couple of seconds per vertex.

Procedure 3

Step 1, Opening most oxDNA or oxView files will take a few seconds at most. As in Procedure 1, importing a file from another design tool can take a few seconds to minutes depending on the size of the design.

Steps 2–3, Creating mutual traps and assigning clusters manually will take a few seconds per trap that needs to be added. This can be several minutes in total for complex designs. Automatically generating forces or assigning clusters scales with the number of nucleotides will be almost instant for small structures and up to several minutes for an origami-sized structure.

Steps 4–6, The speed of RBD scales with the number of nucleotides and clusters in the scene. Relaxation to a final configuration will take a few seconds in small systems and several minutes for larger ones. You may need to stop the RBD and do some manual translation or adjusting of cluster parameters and continue the RBD.

Step 7, The oxServe connection will be established almost instantly.

Step 8, For the oxServe dynamics shown in Fig. 4, the Monte-Carlo CPU relaxation took 7 min to relax the first 1,000 steps using the nanobase.org server (after which it was possible to move on to MD relaxation). The Molecular Dynamics GPU relaxation is significantly faster, with the first 100,000 steps taking 2.5 min. Note that these timings will depend on your internet speed and the current load on the server.

Procedure 4

Steps 1–2, Preparing simulation parameters takes a few minutes whether you are starting from an example input file or filling out the form on oxDNA.org.

Step 3, The time it takes to run a production simulation depends on the hardware used and the number of nucleotides in the simulation. In general, there is an almost linear correlation between the number of nucleotides and the time per simulation step. On a modern GPU, most origami-sized simulations will complete 10^9 steps within 3–5 d. It takes ~4 d to run a simulation with 28,000 particles for 10^9 simulation steps on an NVIDIA V100 GPU. For a benchmark on oxDNA performance, see ref. ⁴³.

Step 4, Aligning the trajectory will take a few minutes for an origami-sized structure.

Step 5, Loading the trajectory file will be almost instant. The visualization may be a bit slow for a few seconds after the first configuration appears as a web worker indexes the trajectory. Indexing progress can be viewed in the ‘Trajectory’ tab.

Steps 6–8, Creating a video of an origami-sized structure can take 20–30 min for a trajectory containing 2,000 configurations of an origami-sized structure.

Step 9, Computing the mean structure will take a few minutes on 2,000 configurations of an origami-sized structure with the settings given (four CPUs). If you have additional CPU cores available, we have found that there is little benefit to running the analysis scripts on >20 cores.

Steps 10–11, Loading the structures in oxView will be almost instant. Indexing the trajectory will take 20–30 s, after which loading the order parameter file and jumping to any configuration will be instant.

Procedure 5

Steps 1–3, For all provided structures except the script generating the crystal cluster (Box 4), the execution time is <2 s using a high-end laptop or work station. The Box 4 example, run on a workstation with an AMD Ryzen 9 3950X processor, takes ~2 min to complete. The subsequent RBD relaxation step with all element visibilities turned off takes ~30 min to complete.

Procedure 6

Step 1, Biological assembly (2btv1.pdb) of a BlueTongue Virus Core consisting of >376,000 residues loads in ~10 s using a Google Chrome browser on an M1 Macbook Air laptop. Nucleosome system (6l9z.pdb) loads in ~2 s with 1,652 residues and 676 nucleotides on the same browser and device. Steps 2–9, Parameterization of the ANM's timing is system size dependent with a limit of 1,500 residues due to the computational demand. Example KDPG Aldolase (1eua.pdb) consisting of 632 residues takes 2.5 min. Both computationally intensive tasks of loading PDB files and parameterizing the ANM are offloaded to webworkers to avoid freezing of the user interface.

Procedure 7

Steps 1–8, Preparing the DNA component will take up to 5 min to import the file from Tiamat and the subsequent oxServe relaxation.
Steps 9–12, Loading and roughly positioning the protein will take <1 min.
Steps 16–21, Setting up and parameterizing the ANM will take ~5 min.
Steps 22–29, Creating the external force files will take <5 min.
Steps 30–33, Relaxation will take 15–20 min, depending on how close to relaxed the previous oxServe relaxation result was.
Step 34, A full 10^9 step production simulation will take ~4 d on a modern GPU.

Procedure 8

Steps 1–6, Exporting a file takes a few minutes at most.
Step 7, Loading large gLTF files into Blender can take up to hours, while smaller files will load within seconds. Positioning lights, camera and editing materials is its own art form and how long this will take depends on how complex your vision is and your familiarity with Blender. Rendering a publication-quality image once the scene is set up can also take up to a couple of hours.

Procedure 9

Steps 1–6, Loading a structure in VR is almost instant, just like loading a structure in a normal browser window.

Anticipated results

The structures, external files, input files and example results for Procedures 1–7 can be found in the examples folder of the oxView GitHub (also included as Supplementary Data) at <https://github.com/sulcgroup/oxdna-viewer/tree/master/examples> and can be downloaded and compared with the results obtained from completing each procedure. When comparing your results with the reference files in the example directory, keep in mind that your structures will look slightly different; you will not move components to the exact same point as we did while generating the examples, and molecular simulations are inherently stochastic processes that will not proceed in the same way in any two runs. Therefore, simulated configurations after both relaxation and production simulation will not be the same as the output configuration files (last_conf.dat) found in the examples. While the individual configurations are different, bulk parameters measured during execution of the procedures such as the mean structure, the energy of the simulation and the relative orientations of paired nucleotides should converge to the same values and distributions observed in the examples directory.

When using oxView to set up simulations whether from imported structures or a de novo design, the key indicator demonstrating that the simulation worked is if the structure does not explode or fall apart. Structures with overlapping particles, stretched bonds or topological impossibilities (parts of the structure passing through each other) will often cause numerical instabilities when physical rules are applied to them in simulations. Often, this causes the molecule to explode, which causes the whole structure to fall apart and the simulation to crash. After a successful simulation, always check that the energy values of the simulation fluctuate around a stable value and do not have a trend. A clear trend

in the energy values would suggest that the simulation was insufficiently relaxed before beginning, in which case the unrelaxed data need to be discarded and the simulation extended to ensure that the simulation is sampling physically relevant states and not artifacts of the initial configuration.

This protocol is primarily focused on setting up simulations and visualizing the results. Simulation convergence and analysis of order parameters is a field that entire textbooks have been written on. If you are interested in learning more about simulation methods using the oxDNA simulation tool, see refs. ^{42,43} for a more comprehensive overview of simulation methods themselves.

Data availability

The input files for examples in this protocols are available in the examples directory (github.com/sulcgroup/oxdna-viewer/tree/master/examples) and in Supplementary Data.

Code availability

All source code presented in this protocol are freely available under a GNU Public License at github.com/sulcgroup/oxdna-viewer. The Python analysis scripts used for analysis are available under GNU Public License at github.com/sulcgroup/oxdna_analysis_tools. The oxDNA simulation code and documentation are available from dna.physics.ox.ac.uk. The DNA/RNA-protein version of the model is in a separate branch available at <https://github.com/sulcgroup/anm-oxdna>. A bleeding-edge version of oxDNA that will replace the main branch at some point in the future can be found at github.com/lorenzo-rovigatti/oxDNA.

References

1. Seeman, N. C. Nucleic acid junctions and lattices. *J. Theor. Biol.* **99**, 237–247 (1982).
2. Pinheiro, A. V., Han, D., Shih, W. M. & Yan, H. Challenges and opportunities for structural DNA nanotechnology. *Nat. Nanotechnol.* **6**, 763–772 (2011).
3. Dey, S. et al. DNA origami. *Nat. Rev. Methods Prim.* **1**, 13 (2021).
4. Douglas, S. M. et al. Rapid prototyping of 3D DNA-origami shapes with caDNAno. *Nucleic Acids Res.* **37**, 5001–5006 (2009).
5. Williams, S. et al. Tiamat: a three-dimensional editing tool for complex DNA structures. 14th International Meeting on DNA Computing (Springer, 2008).
6. Benson, E. et al. Computer-aided production of scaffolded DNA nanostructures from flat sheet meshes. *Angew. Chem.* **55**, 8869–8872 (2016).
7. Benson, E. et al. DNA rendering of polyhedral meshes at the nanoscale. *Nature* **523**, 441 (2015).
8. de Llano, E. et al. Adenita: interactive 3D modelling and visualization of DNA nanostructures. *Nucleic Acids Res.* **48**, 8269–8275 (2020).
9. Huang, C.-M., Kucinic, A., Johnson, J. A., Su, H.-J. & Castro, C. E. Integrated computer-aided engineering and design for DNA assemblies. *Nat. Mater.* **20**, 1264–1271 (2021).
10. Jun, H., Wang, X., Bricker, W. P., Jackson, S. & Bathe, M. Rapid prototyping of wireframe scaffolded DNA origami using ATHENA. Preprint at *bioRxiv* <https://doi.org/10.1101/2020.02.09.940320> (2020).
11. Doty, D., Lee, B. L. & Stérin, T. scadnano: a browser-based, scriptable tool for designing DNA nanostructures. 26th International Conference on DNA Computing and Molecular Programming (DNA 26), 2020.
12. Glaser, M. et al. The art of designing DNA nanostructures with CAD software. *Molecules* **26**, 2287 (2021).
13. Doye, J. P. et al. Coarse-graining DNA for simulations of DNA nanotechnology. *Phys. Chem. Chem. Phys.* **15**, 20395–20414 (2013).
14. Maffeo, C. & Aksimentiev, A. MrDNA: a multi-resolution model for predicting the structure and dynamics of DNA systems. *Nucleic Acids Res.* **48**, 5135–5146 (2020).
15. Lee, J. Y. et al. Rapid computational analysis of DNA origami assemblies at near-atomic resolution. *ACS Nano* **15**, 1002–1015 (2021).
16. Kim, D.-N., Kilchherr, F., Dietz, H. & Bathe, M. Quantitative prediction of 3D solution shape and flexibility of nucleic acid nanostructures. *Nucleic Acids Res.* **40**, 2862–2868 (2011).
17. Snodin, B. E. et al. Introducing improved structural properties and salt dependence into a coarse-grained model of DNA. *J. Chem. Phys.* **142**, 06B613_1 (2015).
18. Šulc, P. et al. Sequence-dependent thermodynamics of a coarse-grained DNA model. *J. Chem. Phys.* **137**, 135101 (2012).
19. Rovigatti, L., Šulc, P., Reguly, I. Z. & Romano, F. A comparison between parallelization approaches in molecular dynamics simulations on GPUs. *J. Comput. Chem.* **36**, 1–8 (2015).
20. Ouldridge, T. E., Louis, A. A. & Doye, J. P. Structural, mechanical, and thermodynamic properties of a coarse-grained DNA model. *J. Chem. Phys.* **134**, 02B627 (2011).
21. Šulc, P., Romano, F., Ouldridge, T. E., Doye, J. P. & Louis, A. A. A nucleotide-level coarse-grained model of RNA. *J. Chem. Phys.* **140**, 235102 (2014).
22. Procyk, J., Poppleton, E. & Šulc, P. Coarse-grained nucleic acid-protein model for hybrid nanotechnology. *Soft Matter* **17**, 3586–3593 (2021).

23. Suma, A. et al. TacoxDNA: a user-friendly web server for simulations of complex DNA structures, from single strands to origami. *J. Comput. Chem.* **40**, 2586–2595 (2019).
24. Poppleton, E., Romero, R., Mallya, A., Rovigatti, L. & Šulc, P. OxDNA.org: a public webserver for coarse-grained simulations of DNA and RNA nanostructures. *Nucleic Acids Res.* **28**, e72 (2021).
25. Poppleton, E. et al. Design, optimization and analysis of large DNA and RNA nanostructures through interactive visualization, editing and molecular simulation. *Nucleic Acids Res.* **49**, W491–W498 (2020).
26. Matthies, M. et al. Triangulated wireframe structures assembled using single-stranded DNA tiles. *ACS Nano* **13**, 1839–1848 (2019).
27. Hong, F., Schreck, J. S. & Šulc, P. Understanding DNA interactions in crowded environments with a coarse-grained model. *Nucleic Acids Res.* **48**, 10726–10738 (2020).
28. Yao, G. et al. Meta-DNA structures. *Nat. Chem.* **12**, 1067–1075 (2020).
29. Wang, Y., Baars, I., Fördös, F. & Högberg, B. Clustering of death receptor for apoptosis using nanoscale patterns of peptides. *ACS Nano* **15**, 9614–9626 (2021).
30. Benson, E., Carrascosa Marzo, R., Bath, J. & Turberfield, A. J. Strategies for constructing and operating DNA origami linear actuators. *Small* **17**, 2007704 (2021).
31. Wang, Y. et al. DNA origami penetration in cell spheroid tissue models is enhanced by wireframe design. *Adv. Mater.* **33**, 2008457 (2021).
32. Pettersen, E. F. et al. UCSF Chimera—a visualization system for exploratory research and analysis. *J. Comput. Chem.* **25**, 1605–1612 (2004).
33. Goodman, R. P. et al. Rapid chiral assembly of rigid DNA building blocks for molecular nanofabrication. *Science* **310**, 1661–1665 (2005).
34. Xu, Y. et al. Tunable nanoscale cages from self-assembling DNA and protein building blocks. *ACS Nano* **13**, 3545–3554 (2019).
35. Yu, Z. et al. A self-regulating DNA rotaxane linear actuator driven by chemical energy. *J. Am. Chem. Soc.* **143**, 13292–13298 (2021).
36. Harris, C. R. et al. Array programming with NumPy. *Nature* **585**, 357–362 (2020).
37. matplotlib/matplotlib: REL: v3.5.1. *Zenodo* <https://zenodo.org/record/5773480#.YhgKyBtlBH4> (2021).
38. Cock, P. J. et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25**, 1422–1423 (2009).
39. Scikit-learn: machine learning in Python <https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html> (2011).
40. McKerns, M. M., Strand, L., Sullivan, T., Fang, A. & Aivazis, M. A. G. Building a framework for predictive science. Preprint at <https://arxiv.org/abs/1202.1056> (2012).
41. Matthies, M. ox-serve v. 1.0. *Zenodo* <https://doi.org/10.5281/zenodo.4551173> (2021).
42. Doye, J. P. K. et al. The oxDNA coarse-grained model as a tool to simulate DNA origami. Preprint at <https://arxiv.org/abs/2004.05052> (2020).
43. Sengar, A., Ouldrige, T. E., Henrich, O., Rovigatti, L. & Šulc, P. A primer on the oxDNA model of DNA: when to use it, how to simulate it and how to interpret the results. *Front. Mol. Biosci.* **8**, 551 (2021).
44. Jo, S., Kim, T., Iyer, G. V. & Im, W. CHARMM-GUI: a web-based graphical user interface for CHARMM. *J. Comput. Chem.* **29**, 1859–1865 (2008).
45. Berendsen, H. J. C., van der Spoel, D. & van Drunen, R. GROMACS: a message-passing parallel molecular dynamics implementation. *Comput. Phys. Commun.* **91**, 43–56 (1995).
46. Webb, B. & Sali, A. Comparative protein structure modeling using MODELLER. *Curr. Protoc. Bioinformatics* 5.6.1–5.6.30 (2016).
47. Gopinath, A. et al. Absolute and arbitrary orientation of single-molecule shapes. *Science* **371** (2021).
48. Tian, Y. et al. Ordered three-dimensional nanomaterials using DNA-prescribed and valence-controlled material voxels. *Nat. Mater.* **19**, 789–796 (2020).
49. Kube, M. et al. Revealing the structures of megadalton-scale DNA complexes with nucleotide resolution. *Nat. Commun.* **11**, 6229 (2020).

Acknowledgements

We acknowledge support from NSF grant 1931487 and ONR grant N000142012094. This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement no. 765703 (to J.B.). We thank users of the tool for their helpful feedback and bug reports, and members of Yan and Šulc labs at ASU and Doye, Louis and Turberfield labs in Oxford for testing the tool.

Author contributions

J.B., M.M., E.P. and J.P. contributed equally to this paper in code, testing and manuscript preparation, and their author order was determined alphabetically. J.B., M.M., E.P., J.P. and A.M. all contributed significantly to the oxView codebase since the previous publication on oxView. H.Y. contributed expertise in DNA nanotechnology and advised on the development of the tool. P.Š. designed and supervised the research tools development. All authors discussed the results and wrote the paper.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41596-022-00688-5>.

Correspondence and requests for materials should be addressed to Petr Šulc.

Peer review information *Nature Protocols* thanks David Doty, Manish K. Gupta and Christopher Maffeo for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Received: 13 July 2021; Accepted: 18 January 2022;

Published online: 06 June 2022

Related links

Key references using this protocol

Poppleton, E. et al. *Nucleic Acids Res.* **48**, e72 (2020): <https://doi.org/10.1093/nar/gkaa417>

Yao, G. et al. *Nat. Chem.* **12**, 1067–1075 (2020): <https://doi.org/10.1038/s41557-020-0539-8>

Procyk, J. et al. *Soft Matter* **17**, 3586–3593 (2021): <https://doi.org/10.1039/D0SM01639J>