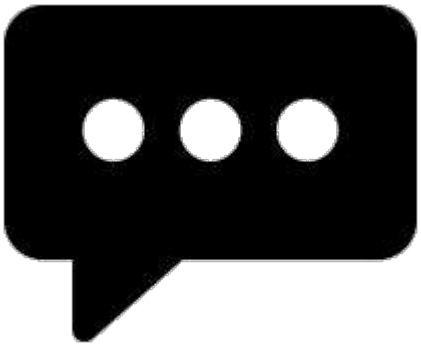
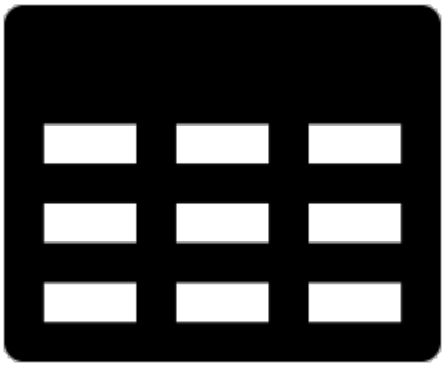
Building blocks for commodity immersive molecular modeling in web browsers (#36241)
1

First submission
Guidance from your Editor
Please submit by **22 May 2019** for the benefit of the authors (and your $200 publishing discount).

Structure and Criteria
Please read the 'Structure and Criteria' page for general guidance.

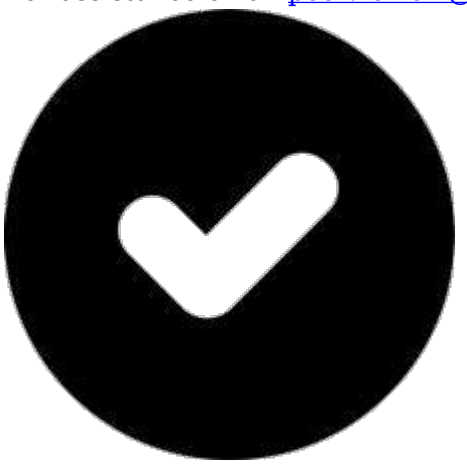


Author notes
Have you read the author notes on the guidance page?



Raw data check
Review the raw data. Download from the location described by the author.

Image check
Check that figures and images have not been inappropriately manipulated.
Privacy reminder: If uploading an annotated PDF, remove identifiable information to remain anonymous.
Files
Download and review all files from the materials page.
6 Figure file(s)
1 Other file(s)
For assistance email peer.review@peerj.com



Structure and Criteria
2

Structure your review
The review form is divided into 5 sections. Please consider these when composing your review:
1. BASIC REPORTING
2. EXPERIMENTAL DESIGN
3. VALIDITY OF THE FINDINGS
4. General comments
5. Confidential notes to the editor

You can also annotate this PDF and upload it as part of your review
When ready submit online.
Editorial Criteria

Use these criteria points to structure your review. The full detailed editorial criteria is on your guidance page.

BASIC REPORTING

Clear, unambiguous, professional English language used throughout.

Intro & background to show context. Literature well referenced & relevant.

Structure conforms to PeerJ standards, discipline norm, or improved for clarity.

Figures are relevant, high quality, well labelled & described.

Raw data supplied (see PeerJ policy).

EXPERIMENTAL DESIGN

Original primary research within Scope of the journal.

Research question well defined, relevant & meaningful. It is stated how the research fills an identified knowledge gap.

Rigorous investigation performed to a high technical & ethical standard.

Methods described with sufficient detail & information to replicate.

VALIDITY OF THE FINDINGS

Impact and novelty not assessed. Negative/inconclusive results accepted. *Meaningful* replication encouraged where rationale & benefit to literature is clearly stated.

All underlying data have been provided; they are robust, statistically sound, & controlled.

Speculation is welcome, but should be identified as such.

Conclusions are well stated, linked to original research question & limited to supporting results.

Standout reviewing tips
3

The best reviewers use these techniques

Tip
Example
Support criticisms with evidence from the text or from other sources
Smith et al (J of Methodology, 2005, V3, pp 123) have shown that the analysis you use in Lines 241-250 is not the most appropriate for this situation. Please explain why you used this method.
Give specific suggestions on how to improve the manuscript
Your introduction needs more detail. I suggest that you improve the description at lines 57- 86 to provide more justification for your study (specifically, you should expand upon the knowledge gap being filled).
Comment on language and grammar issues
The English language should be improved to ensure that an international audience can clearly understand your text. Some examples where the language could be improved include lines 23, 77, 121, 128 – the current phrasing makes comprehension difficult.
Organize by importance of the issues, and number your points
1. Your most important issue
2. The next most important item
3. …
4. The least important points
Please provide constructive criticism, and avoid personal opinions
I thank you for providing the raw data, however your supplemental files need more descriptive metadata identifiers to be useful to future readers. Although your results are compelling, the data analysis should be improved in the following ways: AA, BB, CC
Comment on strengths (as well as weaknesses) of the manuscript
I commend the authors for their extensive data set, compiled over many years of detailed fieldwork. In addition, the manuscript is clearly written in professional, unambiguous language. If there is a weakness, it is in the statistical analysis (as I have noted above) which should be improved upon before Acceptance.

Building blocks for commodity immersive molecular modeling in web browsers

Luciano A Abriata Corresp. 1

1 École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

Corresponding Author: Luciano A Abriata
Email address: luciano.abriata@epfl.ch

Almost every piece of scientific literature reporting advances in immersive interfaces for molecular modeling suggests or assumes that these technologies should greatly enhance thought processes in chemistry and structural biology, thus being useful for education, research and outreach in these disciplines. However, actual testing of immersive technologies for molecular modeling progresses very slowly, because most available tools require specialized hardware and software thus being hardly accessible to the community. As an alternative of much wider reach, I explore here state-of-the-art technologies for building immersive human-computer interfaces through augmented and virtual reality in web browsers. Specifically, I discuss and showcase how these technologies can be integrated into web apps that enable commodity, open-source, highly interactive human-computer interfaces for chemistry, structural biology and related disciplines, in ways accessible with standard computer devices and no software impediments. The prototype applications showcase wide applicability to education and research, ranging from visualization in interactive immersive 3D to basic modeling of physics and chemistry and on-the-fly comparison of experimental and simulated data. From this playground I outline how these emerging technologies could couple in the future to neural-network-based physics calculations, speech-based computer interaction, and sockets for concurrent collaboration through the internet -all technologies that are today maturing in web browsers- to deliver the next generation of tools for interactive, immersive, collaborative molecular modeling that can streamline human thought and intent with the numerical processing power of computers, at affordable costs. The prototypes and ideas introduced here further serve as starting points to build active content that everybody can utilize online, allowing for the first time actual mass testing of the power of immersive technologies for molecular modeling.

Building blocks for commodity immersive molecular modeling in web browsers

Luciano Andrés Abriata[1]

[1] Laboratory for Biomolecular Modeling and Protein Production and Structure Facility, School of Life Sciences, École Polytechnique Fédérale de Lausanne and Swiss Institute of Bioinformatics, CH-1015 Lausanne, Switzerland.

Corresponding Author:
Luciano A. Abriata[1]
AAB044 Station 19, CH-1015 Lausanne, Switzerland
Email address: luciano.abriata@epfl.ch

Abstract

Almost every piece of scientific literature reporting advances in immersive interfaces for molecular modeling suggests or assumes that these technologies should greatly enhance thought processes in chemistry and structural biology, thus being useful for education, research and outreach in these disciplines. However, actual testing of immersive technologies for molecular modeling progresses very slowly, because most available tools require specialized hardware and software thus being hardly accessible to the community. As an alternative of much wider reach, I explore here state-of-the-art technologies for building immersive human-computer interfaces through augmented and virtual reality in web browsers. Specifically, I first discuss and then introduce examples on how these technologies can be integrated into web apps that enable commodity, open-source, highly interactive human-computer interfaces for chemistry, structural biology and related disciplines, accessible on standard computers, tablets and smartphones with no software impediments. The example, prototype web apps advance wide applicability to education and research, ranging from visualization in interactive immersive 3D to basic modeling of physics and chemistry and on-the-fly comparison of experimental and simulated data. From this playground I further outline how these emerging technologies could couple in the future to neural-network-based physics calculations, speech-based computer interaction, and sockets for concurrent collaboration through the internet -all technologies that are today maturing in web browsers- to deliver the next generation of tools for interactive, immersive, collaborative molecular modeling that can streamline human thought and intent with the numerical processing power of computers, at affordable costs. The prototypes and ideas introduced here further serve as starting points to build active content that everybody can utilize online, allowing for the first time actual mass testing of the power of immersive technologies for molecular modeling.

Introduction

For a long time it has been suggested that immersive graphics technologies based in virtual reality (VR) and augmented reality (AR), as well as other advanced forms of human-computer interactions, have enormous potential in assisting thought processes in scientific research and in education, especially in disciplines that deal with abstract objects, objects much smaller or larger than human dimensions, objects that are hard to acquire and handle due to high costs, limited availability, fragility, very large amounts of visual data, etc.[1–3] Chemistry and structural biology are examples of such disciplines where AR and VR have been attributed high potential in education and research.[4–14] However, actual reach, user acceptance and educational effects of AR- and VR-based technologies still require deep evaluation [15–19], which has so far progressed slowly due to the complex software setups and the specialized hardware needed for currently available tools, which limit reach. As I intend to demonstrate here, client-side web technologies have matured enough to open up immersive molecular modeling technologies to the world at very affordable costs and with very simple setups.

This article begins by briefly reviewing the client-side web programming technologies that enable, in ordinary web browsers running on standard devices (computers, tablets and smartphones), developments of graphical interfaces for AR and VR, gesture- and speech-based commands, in-place or third-party calculations for molecular mechanics and for comparison to experimental observables, concurrent collaboration through the world wide web, and other human-computer interaction technologies. In a second stage the article showcases prototype web apps for specific tasks of practical utility in pedagogical settings and for actual research in chemistry, biophysics and structural biology. Being based on fully open technologies that require no installations and run on all modern browser-enabled devices, educators and researchers are free to try out all these examples right away. Furthermore, being the source code intrinsically open and modular, new applications can be easily created as exemplified.

Part 1: Survey of building blocks for immersive molecular modeling in web browsers

Virtual and augmented reality

At the core of immersive experiences there is visualization based on virtual or augmented reality methods (VR and AR). VR is the total replacement of the real world by a simulated environment,

whereas AR is the superposition of virtual objects on top of the real world. While VR is probably experienced best with VR goggles to suppress any side view of the real world, AR is not subject to this problem so it is amenable to other kinds of less immersive devices like desktop or laptop computers, tablets and smartphones (besides goggles, which are not incompatible) making it better suited for commodity solutions. A standard computer equipped with a frontal webcam can function as an "AR mirror" where the user sees him/herself with the virtual objects in the hands (Figure 1A-C), while a device with a webcam on its back serves for look-through AR, and a smartphone can be used for AR (or VR) by plugging it into cardboard goggles with proper eye lenses (Figure 1D). Comparing the actual utility of AR and VR, tests of object manipulation by human subjects showed significantly faster task completion time in AR over VR, possibly because the user seeing his/her own arms facilitates motion coordination.[2]

In web browsers, the WebGL API provides powerful 2D and 3D graphing capabilities (using GPU resources) in a format fully integrable with other web standards, HTML elements, APIs and JavaScript libraries, without the need of plug-ins and highly standardized across browsers. A handful of JavaScript libraries exploit WebGL to facilitate scene rendering, Three.js being probably the most widely used. In turn, tools like A-Frame provide entity component system frameworks that wrap Three.js into HTML language tags for building AR and VR scenes very easily. The examples presented here showcase either direct use of Three.js or Three.js through A-Frame, in AR and VR although the focus is on AR. Depending on the goal of the immersive web app, it can either (i) load pre-made models of the molecular systems (even animations, which could be derived from molecular simulations, for example) in OBJ/MTL or other formats (OBJ/MTL can be very easily exported from VMD, example in Figure 1C); or (ii) use WebGL primitives (like spheres, cylinders, etc.) to draw the molecular systems from scratch (Figure1A,B,D and most other examples in this article).

Looking into the future, web browser developers are working on directly integrating AR/VR as a native API, which should provide smoother and higher-resolution graphics; however this is largely experimental at the moment.[20]

Object detection and tracking

The other key component required for AR and VR is a means to detect and track objects or parts of the user's body such as his/her hands, in order to manipulate virtual objects. Applications

using *ad hoc* hardware use sensors and cameras that track the user's position in space and hand-held controllers, usually seen by the user as virtual tweezers, to directly move objects in space. Such hardware is however expensive, especially for educational settings outside developed countries. For commodity AR/VR in web browsers, solutions rely on computer vision through the webcam, essentially tracking algorithms like ARToolKit's JavaScript version, jsartoolkit, among other similar solutions.[21] This library essentially tracks user-defined 2D markers (examples in Figure 2A and in S1 ready to print at different sizes) in space as long as they are visible to the webcam, and make their computed coordinates available to the graphics algorithms. One particularly interesting implementation is AR.js and its A-Frame wrap, which enable highly simplified AR/VR, even using exclusively HTML code for simple developments.

It is interesting to note that in marker-based AR different viewers receive different perspectives of the same marker and hence of the rendered virtual object, just as if it was a real object in real space (Figure 2B). This easily enables multi-user AR/VR in a common room, as would be useful in a classroom setting where students and teachers look at the same virtual molecule.

An alternative to traditional marker-based AR should in principle be possible by using a plain hand-tracking JavaScript library like Handtracking.js. Another slightly more expensive approach but possibly better in tracking performance is using a device like the infrared-based Leap Motion Controller, which includes a JavaScript library to feed positional information from the device into the web app. Unfortunately, however, there are currently no ready-to-use libraries that couple these input tools to WebGL graphics.

Current JavaScript libraries for computer vision allow even more complex object tracking. One interesting example is gesture recognition by the WebGazer.js library, which analyzes face features to estimate where on the screen the user is looking at[22]. In molecular visualization this can be used for example to automatically move regions of interest to the frontof the visualization, as in the example in Figure 2C.

Speech-based interfaces

On top of gesture recognition, a speech-based interface can be highly useful for situations in which the user's hands are busy holding objects, as would happen very often in AR/VR applications. Current in-browser speech recognition APIs enable this very easily, especially

through libraries like Annyang [23] used in some of the examples of this article. These libraries usually allow working in two modes, one where the browser waits for specific commands (most accept variable inputs) and one where the browser collects large amounts of text that are then made available to the environment. The former allows direct activation of functions without the need for the user to click on the screen. The second option opens up the possibility of automatically detecting subjects, actions and concepts that are fed to artificial intelligence, or just predefined rules, that the computer will analyze in background. For example, when two users are discussing the interaction surface between two proteins and mention certain residues, the computer could "understand" this and automatically mine PubMed for mentions of said residue. This seems far-fetched, but is essentially the same technology that underlies automatic advertising and suggestions based on users' various inputs and usage statistics in ordinary devices, software and web portals.

Intensive calculations

As reviewed in a recent special issue of *Computing and Science in Engineering* [24], JavaScript has become a very powerful language through several avenues, including transcompilation from C/C++ reaching near-native execution speeds, language subsets specialized for speed, optimized just-in-time compilation, methods to program background scripts, and libraries to perform multi-core and on-GPUs computing to accelerate intensive calculations. I have developed on the impact this power has had in scientific computing in the last decade, especially regarding the development of tools for online analysis and visualization of data.[25,26] This power enables calculations that would not be possible with early JavaScript, which was interpreted and of slow execution. Possible applications, some of which are exemplified in some of the prototypes presented here, include simulating dynamics and experimental data, performing numerical data analysis and even handling data in neural networks, etc. If achieved fast enough, these possibilities enable applications where the user gets real-time numerical response from the web app as (s)he handles the molecular system. Several of the prototypes presented later on include such examples.

Besides the specific calculations one might code *ad hoc* for certain problems, there are several libraries that can be of use, saving large writing time and often being developed by specialists. While a large list is reviewed in[25] and kept updated at

https://lucianoabriata.altervista.org/jsinscience/, I mention here one particularly useful library called Cannon.js. This JavaScript library simulates rigid body mechanics, and integrates smoothly with Three.js and A-Frame, as in the example from Figure 1D where it is used to simulate thermal motions. Although rigid-body physics might look limiting for molecular modeling applications, they are often enough in settings requiring coarse-grain modeling like in integrative modeling; in fact the Integrative Modeling Platform[27] contains one such kind of engine for certain applications. Nevertheless, implementation of more complex force fields is certainly possible, as exemplified by a JavaScript transcompilation of the OpenMD engine.[28]

Further building blocks

Any other technology that facilitates interaction with the computer within a 3D environment, either to deliver or obtain information, might be of use. For example, haptic feedback would be desirable to confer a physical feel of interactions and clashes. Achieving a good experience in haptic feedback requires specialized devices, with possibly some less expensive alternative based on commodity robotics. Other rudimentary ways to achieve sensory feedback is by exploiting built-in vibration devices, which can be activated from JavaScript (as in the example in Figure 1D running in a smartphone), and touch-pressure sensing in touch screens which is also handled by JavaScript APIs.

A particularly interesting aspect of software running on web browsers is the ease with which different users can connect to each other, just over the internet. Web apps can exploit browser communication sockets to achieve browser-to-browser links over which data can be transmitted freely, with a server only intervening to establish the initial connection. As an example, I have authored a prototype in which two or more users can concurrently work on a JSmol session; essentially a chief user can apply rotations and commands on all other users' screens with a few ms of delay (because only instructions are transmitted, not graphics).[29] Such technology could be adapted to complex immersive environments to allow multiple users to work on the same problem at a distance, essential for collaboration, demonstrations, and online teaching[30].

Part 2: Prototype web apps showcasing sample applications

This section presents example web apps compatible with major web browsers in modern smartphones, tablets and computers, introducing features of increasing complexity. All examples are accessible through links in the Figure captions and at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/. To run these AR/VR web apps the user needs to print the Hiro and Kanji markers at the same scale (Figure 2A and S1). For simpler handling, the markers may be glued on a flat surface mounted on a device that can be easily rotated from the back, such as a small shaft perpendicular to the marker plane. Some examples (like that in Figure 1B) require printing and folding of a cube marker, available for download at the indicated URL.

Introducing web browser-based AR/VR

The simplest software setup for AR consists in using A-Frame to either (i) display a representation exported from VMD in OBJ/MTL format on a marker (example in Figure 1C); or (ii) display WebGL spheres at coordinates computed from atomic positions and colored by atom type (similar to the example in Figure 1A). Since only A-Frame is used in such examples, only HTML tags are needed, which the user can inspect in the source code of the pages listed in the caption to Figure 1 (or in the HTML section of the code shown in Figure S2).

Apps using A-Frame can gain interactivity through portions of JavaScript code that read atom coordinates and carry out calculations on them. In the example shown in Figure 1A the user drives a lysine side chain with the Hiro marker and a glutamate side chain with the Kanji marker. Each molecule is anchored to the center of its corresponding AR marker through its CA atom. Their protonation states correspond to neutral pH, so lysine is protonated hence its N atom (blue) is charged by +1, whereas glutamate is deprotonated hence its O atoms (red) bear a total charge of -1. Through simple JavaScript code the web app (i) adapts the positions of 10 yellow spheres that connect in space the lysine's N atom with one of the glutamate's O atoms; (ii) report the distance between these 2 atoms and the corresponding attractive electrostatic force in real time; and (iii) test and display clashes between any pair of atoms of the two molecules. The code for (i) is wrapped inside an auxiliary .js file, and the code for (ii) and (iii) is located between <script> tags at the end of the HTML file. The distance, electrostatic and clash calculations are computed inside a setInterval() function that is executed every 200 ms and uses the id identifiers of the sphere tags to locate them (see code in Figure S2). The distance is calculated in Angstrom

and includes a correction for a zoom factor that scales atom sizes and positions when the molecular coordinates are parsed into A-Frame HTML, to properly fit the screen. Clashes are detected as two spheres being within 3 Å and displayed as semitransparent A-Frame spheres centered on the affected atoms.

The latter example can be slightly modified to better fit in smartphone screens for visualization with commodity goggles (Figure 1D) by flipping the video horizontally and splitting the view in 2 images slightly differently oriented to render a 3D experience (which is handled by the A-Frame library itself). The example adapted for smartphones further features vibration each time a clash is detected.

All the above examples use a single AR marker per molecule, which is easy to setup but is limited to orientations where the marker is visible. With multiple coupled markers, for example a cube made of 6 different markers in its faces, the user can freely rotate the object in space, with the additional advantage that the estimation of position and orientation works better. One such example is shown in Figure 1B, from a web app that allows manipulation of any molecule loaded in PDB format.

Adding interactivity: examples on small molecules

An interesting modification of the example using lysine and glutamate side chains is incorporating a very simplistic emulation of hydrogen bond detection and proton transfer. Here, JavaScript code calculates and displays hydrogen bonds when the geometry permits, and randomly "transfers" one proton from lysine's N atom to one of the O atoms of glutamate if the lysine is protonated, or the other way around (actually spheres attached to each marker are hidden or shown as needed to emulate proton transfer). Protons "jump" only when they are within 2 Å of the receiving N or O atom; and they are set to jump back and forth to reflect 70% time-averaged population of protonated lysine and 30% of protonated glutamate, to convey the feeling of different acidic constants (in this case set to 70/30 = 2.33 favoring protonated lysine). At distances longer than 2 Å but shorter than 3 Å the web app displays a yellow dotted line that represents a hydrogen bond between the potential receiver heavy atom and the involved proton.

Similar emulation strategies could be easily used to build "interactive animations" for exploring chemical and physical phenomena, for example reactivity in Figure 3A which illustrates stereoselectivity in the Diels-Alder reaction in interactive 3D. This reaction occurs

between a dienophile and a conjugated diene in a concerted fashion, such that the side of the diene where the initial approach occurs defines the stereochemistry of the product. The web app in this example allows users to visualize this in 3D as they approach a molecule of 1,3-cyclohexadiene held with the left hand (Hiro marker) and a molecule of vinyl chloride in the right hand (Kanji marker). As the two pairs of reacting C atoms approach simultaneously, the two new bonds gain opacity until the product is formed. The product formed in this reaction is by itself an interesting molecule to visualize and move around in 3D through AR, because it contains two fused six-membered rings which are often hard to understand in 2D.

Looking into the near future, there is a very important point regarding the idea of emulating reactivity as in the proton transfer and Diels-Alder reaction examples. These examples are merely pictorial visualizations of the mechanisms, and they are not based on any kind of quantum calculations. Such calculations are extremely slow to be incorporated into immersive experiences where energies need to be computed on the fly. However, novel machine learning methods that approximate quantum calculations through orders-of-magnitude faster computations (like in [31–33]) could in a near future be coupled to AR/VR systems to interactively explore reactivity with realistic energy profiles in real time. Such tools would be useful not only for education but also for actual research, for example to interactively test the effect of chemical substituents on a reaction, estimate effects on spectroscopic observables, probe effects of structural changes on molecular orbitals, etc.

Another interesting tool to integrate with AR/VR is a physics engine, to add realistic mechanics to the simulation. The web app in Figure 1B uses Cannon.js to simulate thermal motions and thus give a sense of dynamics to the visualized system. And as presented in the first section, it is possible to adapt more complex force fields better suited for molecular interactions into JavaScript versions.[28]

Possible applications go well beyond emulating reactivity and dynamics, being especially interesting the possibility of using JavaScript to carry out on-the-fly calculations of experimental observables that can be compared in real time to a piece of actual experimental data, or simply to explore possible outcomes of an experiment. Figure 3B exemplifies this with the calculation of paramagnetically induced pseudocontact chemical shift and line broadening on a probe atom attached to one AR marker, as it is moved around the heme group of metmyoglobin with the other AR marker. This web app implements standard equations from the theory of paramagnetic

nuclear magnetic resonance[34] fed on-the-fly with the corresponding polar coordinates, then simulates the spectrum including noise and displays it using the Google Charts JavaScript library. More examples with realistic applications of client-side calculations in molecular modeling are introduced below, for biological macromolecules.

Immersive visualization and modeling of biological macromolecules

Figure 1C exemplified a VMD session directly imported into AR.js-A-Frame, for a very simple visualization-only AR experience in web br rs. Figure 4A does the same on a VMD scene reproducing the work of Berry and Board[6] but achievable with a web browser *i.e.* requiring no program, plug-ins nor driver installation, and accessible just on a web page. Besides using premade VMD scenes, the developer can also convert atomic coordinates to Three.js or A-Frame spheres as in the helical model of a nuclear export signal fro Figure 4B) which readily shows the functionally relevant amphipathic nature of these protein motifs. Moreover, with the web app depicted in Figure 1B one can load any file from the Protein Data Bank (although currently only small proteins are well supported) to look at its contents in AR, and even emulate thermal motions and protein unfolding at high temperatures through Cannon.js.

The next examples showcase the prototype incorporation of restraints, primitive force fields and on-the-fly simulation of experimental observables in web browser-based AR for molecular modeling.

The example in Figure 5A allows driving in space two molecules that are known to form a complex in solution, specifically ubiquitin (red trace) and a ubiquitin-interacting motif (UIM, blue trace) taken from PDB ID 2D3G[36]. The web app simulates on-the-fly the small-angle X-ray scattering (SAXS) profiles expected from the relative arrangement of the two proteins, overlaying simulated profiles over an experimental profile in real time as the user moves the proteins around. This offers a way to interactively test possible docking poses that are compatible with the experimental data. Although this urse could never be better than the extensive sampling achievable with molecular simulations, such an interactive tool could be useful (i) for preliminary analysis of SAXS data before starting complex calculations and (ii) to judiciously analyze the results from such calculations. [In this example the SAXS profile calculation is based on the Debye formula iterated through pairs of residues instead of iterating through all atoms as the full equation requires, for simplicity and speed; however, realistic SAXS

profiles of actual utility can be achieved with similar strategies and proper parameterization of the scattering centers[37]]. This web app further includes a rudimentary residue-grained force field to detect clashes, and a predefined binding coordinate which upon activation brings the two molecules together. Activation of SAXS simulation, clash-detecting force field and binding coordinate are controlled by voice commands, required because the user's hands are busy handing the markers. This proceeds through the browser's speech recognition API and does not consume much resources because speech recognition occurs in the cloud. The possibility of incorporating all these different elements (AR, 3D visualization, calculations and speech recognition) illustrates the superb integration capacity of libraries for client-side scripting in web browsers.[25] As a last note regarding this example, I highlig ow the modularity and simplicity of client-side web programming allows easy adaptation to other kinds of experimental data, as in the example on residue-specific paramagnetic relaxation enhancements at https://rrasia.altervista.org/HYL1_1-2/Hyl1_12_minima.html

Another example, presented in Figure 5B, shows how AR can help to explore residue-residue contact data. This information essentially points at pairs of residues that are in contact,[38] either inside folded proteins [39] or, more relevant to this example, in protein-protein complexes[40]. The example shows contacts predicted from coevolution analysis of large sequence alignments for the pair of proteins in chains A and B of PDB entry 1QOP, taken from the Gramlin server[40] at http://gremlin.bakerlab.org/cplx.php?uni_a=1QOP_A&uni_b=1QOP_B. Each protein is driven by one marker, and the predicted contacts are overlaid as dotted lines connecting the intervening pairs of residues. These lines are colored green, olive and red according to decreasing coevolution score as in the Gremlin website, and their widths reflect in real time the distance between pairs of residues, supposed to be minimal when contacts are satisfied. Such kind of tool could allow interactive inspection of residue-residue contact data to actively detect false positives through human intervention, to be removed before running restraint-guided docking.

The last prototype application shows rudimentary handling of highly disordered protein regions, here to test how a linker made of 6 glycine residues restricts the possible orientations of two well folded domains (Figure 6). Molecular motions "dynamics" or "flexibility", are essential for protein function and regulation, as well as for assembling complexes from their constituents. Broadly speaking, molecular motions range from restricted, coordinated structural changes in ordered domains, to extended dynamics in highly flexible regions. Accounting for

dynamics in faithful detail across such vast range of time and size scales constitutes an active goal of research, and requires complex force fields and molecular dynamics engines. As commented earlier, it is feasible to code complex simulation methods in JavaScript achieving near-native performance.[28] However, the example shown here considers only the dynamics of very flexible linkers within the very coarse approximation that beads describing amino acids behave as rigid spheres connected by fixed distances. Despite very simplistic, such rigid-body-like descriptions are often sufficient for describing several aspects of biological matter.[27] Anyway, as always with computer-based methods for simulations, the conclusions obtained from a given model must be consistent with its detail.

More specifically, the example in Figure 6 handles flexibility with Cannon.js, presented earlier for the example in Figure 1B but here wrapped into HTML through the A-Frame-physics extension. Each globular domain (corresponding to ubiquitin or the interacting helix) is modelled with 2 to 4 rigid beads per residue, *i.e.* a description similar to that of the MARTINI force field[41], while the flexible linker is modelled as backbone beads only (similar to the MARTINI description for glycine). As the user moves the two domains around, one attached to each AR marker, the web app updates the positions of the residues of the flexible linker based on the Cannon.js force field. This very simple model can help to answer questions related to the separation of the anchor points and the maximal extension of the linker when straight: How far can the two proteins go with the given linker containing 6 residues? Can the interacting partner be docked through certain interfaces yet allow for a relaxed configuration of the linker? Such investigations are in turn assisted by on-the-fly estimation of entropy associated to given extensions of the linkers, calculated here from a worm-like model,[42] and by the strain on the linkers which essentially tracks how much the consecutive glycines are stretched when the user pulls them apart beyond their equilibrium distance.

Discussion and Outlook

Achieving smooth and useful immersive chemistry is one of the key "grand challenges" for the simulation of matter in the 21st century,[43] that also apply to multiple other disciplines where interactive computer interfaces can help to more easily grasp and explore concepts. After around two decades of works slowly introducing AR and VR for chemistry and structural biology, the last two years saw four very inspiring pieces of literature showing the real potential of modern,

multiuser AR/VR in immersive molecular modeling and visualization. Goddard et al published a rich discussion about these technologies in the context of their own computer programs, including the widely used Chimera,[44] surveying advantages and disadvantages and presenting several interesting case studies.[18] O'Connor et al presented a VR-based system for multiuser handling of molecules in space which even accounts for molecular mechanics through a standard force field.[45] And Balo et al presented web-based software for VR molecular visualization on smartphones.[46]

Although the hardware required by current solutions for AR/VR (i.e. VR helmets/goggles with 3D handheld input devices) is much more accessible than a decade ago, it is still not what one could consider inexpensive and ly available across the world, especially for educational settings and outside developed countries. A few notable exceptions that achieve AR through webcams *i.e.* commodity hardware still require nonetheless somewhat complex software setups[5,6], in contrast to the methodology proposed here that relies entirely on HTML, CSS and JavaScript code running on standard web bro . There is no need for plugins and thus no installations, no manual updates, and no costs ciated to these web apps. In fact, the reader could right now print the AR markers from Figure 2A or Figure S1, access the webpages indicated in the figure captions, and test the examples right away.

The examples introduced here hence advance a potential for open, inexpensive, web-based AR and VR technologies in education and research in (bio)chemistry and related disciplines. In education, such tools could ace/complement tangible modeling kits, allowing virtually unlimited numbers and kinds of s, as well as augmenting models with additional information such as forces, charges, electron clouds and orbitals, data facts, etc. In research, such tools could help to visualize and probe molecular structure, simulate expected outcomes of experiments and test models and simulated data against experimental data, etc., everything through intuitive cues and fluent human-computer interactions.

Being these web apps so openly accessible, the next step is to develop content that teachers, students and researchers can put into use, to enable proper large-scale evaluations of the actual impact in learning and thinking processes. If successful, the near future will likely see these tools blended with modern molecular simulations and visualization methods, resulting in fully-fledged programs for deep immersive, interactive molecular modeling experiences that overcome the limitations of traditional software based on screen, mouse and keyboard.

Furthermore, happening all inside web browsers will seamlessly enable online concurrent collaboration among multiple peers as well as between students and teachers. Further adding haptic devices for force feedback, speech-based voice commands, numerical simulations and intelligent data and text mining –most of these technologies already built into web browsers, as presented- will bring research and education to the next level where human thought and intent couple with computer power to get the best out of both worlds.

Acknowledgements

I acknowledge all the members of the communities that develop the client-side web programming tools used here, as well as the very helpful communities who contribute to their improvement, bug detection and correction, documentation, and online help. Special thanks to J. Ethienne (AR.js developer), D. McCurdy (Google), T. Ater (Annyang developer), L. Stemkoski (Adelphi U., USA), A. Herráez (U. Alcalá, Spain), R. Rasia (U. of Rosario and IBR-CONICET, Argentina) and A. Papoutsaki (Pomona College, USA). I also acknowledge numerous researchers from the physics, chemistry and biology communities who have contributed several useful ideas and suggestions, inspiring some of the presented examples.

References

1. O'Donoghue, S. I. *et al.* Visualizing biological data-now and in the future. *Nat. Methods* **7**, S2-4 (2010).
2. M. Krichenbauer, G. Yamamoto, T. Taketom, C. Sandor & H. Kato. Augmented Reality versus Virtual Reality for 3D Object Manipulation. *IEEE Trans. Vis. Comput. Graph.* **24**, 1038–1048 (2018).
3. Sommer, B., Baaden, M., Krone, M. & Woods, A. From Virtual Reality to Immersive Analytics in Bioinformatics. *J. Integr. Bioinforma.* **15**, (2018).

4. Gillet, A., Sanner, M., Stoffler, D., Goodsell, D. & Olson, A. Augmented reality with tangible auto-fabricated models for molecular biology applications. *IEEE Visualization* (2004).
5. Gillet, A., Sanner, M., Stoffler, D. & Olson, A. Tangible interfaces for structural molecular biology. *Struct. Lond. Engl. 1993* **13**, 483–491 (2005).
6. Berry, C. & Board, J. A Protein in the palm of your hand through augmented reality. *Biochem. Mol. Biol. Educ. Bimon. Publ. Int. Union Biochem. Mol. Biol.* **42**, 446–449 (2014).
7. Vega Garzón, J. C., Magrini, M. L. & Galembeck, E. Using augmented reality to teach and learn biochemistry. *Biochem. Mol. Biol. Educ. Bimon. Publ. Int. Union Biochem. Mol. Biol.* (2017). doi:10.1002/bmb.21063
8. Wolle, P., Müller, M. P. & Rauh, D. Augmented Reality in Scientific Publications—Taking the Visualization of 3D Structures to the Next Level. *ACS Chem. Biol.* **13**, 496–499 (2018).
9. Hirst, J. D., Glowacki, D. R. & Baaden, M. Molecular simulations and visualization: introduction and overview. *Faraday Discuss* **169**, 9–22 (2014).
10. Maier, P., Tönnis, M. & GudrunKlinker, D. Dynamics in Tangible Chemical Reactions. in (2009).
11. Maier, P. & Klinker, G. Augmented Chemical Reactions: 3D Interaction Methods for Chemistry. *Int. J. Online Eng.* **9**, (2013).
12. Ratamero, E. M., Bellini, D., Dowson, C. G. & Römer, R. A. Touching proteins with virtual bare hands : Visualizing protein-drug complexes and their dynamics in self-made virtual reality using gaming hardware. *J. Comput. Aided Mol. Des.* **32**, 703–709 (2018).
13. Müller, C. *et al.* Interactive Molecular Graphics for Augmented Reality Using HoloLens. *J. Integr. Bioinforma.* **15**, (2018).

14. Martínez-Hung, H., García-López, C. A. & Escalona-Arranz, J. C. Augmented Reality Models Applied to the Chemistry Education on the University (article in Spanish). *Rev. Cuba. Quím.* **29**, 13–25 (2016).
15. M. Fjeld & B. M. Voegtli. Augmented Chemistry: an interactive educational workbench. in *Proceedings. International Symposium on Mixed and Augmented Reality* 259–321 (2002). doi:10.1109/ISMAR.2002.1115100
16. Pence, H. E., Williams, A. J. & Belford, R. E. New Tools and Challenges for Chemical Education: Mobile Learning, Augmented Reality, and Distributed Cognition in the Dawn of the Social and Semantic Web. in *Chemistry Education* 693–734 (Wiley-VCH Verlag GmbH & Co. KGaA, 2015). doi:10.1002/9783527679300.ch28
17. Bach, B., Sicat, R., Beyer, J., Cordeil, M. & Pfister, H. The Hologram in My Hand: How Effective is Interactive Exploration of 3D Visualizations in Immersive Tangible Augmented Reality? *IEEE Trans. Vis. Comput. Graph.* **24**, 457–467 (2018).
18. Goddard, T. D. *et al.* Molecular Visualization on the Holodeck. *J. Mol. Biol.* (2018). doi:10.1016/j.jmb.2018.06.040
19. Yang, S., Mei, B. & Yue, X. Mobile Augmented Reality Assisted Chemical Education: Insights from Elements 4D. *J. Chem. Educ.* **95**, 1060–1062 (2018).
20. Medley, J. Augmented reality for the web. (2019).
21. Kato, H. *ARToolKit*. (1999).
22. Papoutsaki, A. *et al.* WebGazer: Scalable Webcam Eye Tracking Using User Interactions. in *Proceedings of the 25th International Joint Conference on Artificial Intelligence* 3839–3845 (2016).
23. Ater, T. *Annyang*. (2019).

24. M. DiPierro. The Rise of JavaScript. *Comput. Sci. Eng.* **20**, 9–10 (2018).
25. Abriata, L. A. Web Apps Come of Age for Molecular Sciences. *Informatics* **4**, 28 (2017).
26. Abriata, L. A., Rodrigues, J. P. G. L. M., Salathé, M. & Patiny, L. Augmenting research, education and outreach with client-side web programming. *Trends Biotechnol.* (2017).
27. Russel, D. *et al.* Putting the pieces together: integrative modeling platform software for structure determination of macromolecular assemblies. *PLoS Biol.* **10**, e1001244 (2012).
28. Jiang, C. & Jin, X. Quick Way to Port Existing C/C++ Chemoinformatics Toolkits to the Web Using Emscripten. *J. Chem. Inf. Model.* (2017). doi:10.1021/acs.jcim.7b00434
29. Abriata, L. A. Concurrent interactive visualization and handling of molecular structures over the Internet in web browsers. *ArXiv E-Prints* (2017).
30. Lee, J., Kim, J.-I. & Kang, L.-W. A Collaborative Molecular Modeling Environment Using a Virtual Tunneling Service. *J. Biomed. Biotechnol.* **2012**, 7 (2012).
31. Smith, J. S., Isayev, O. & Roitberg, A. E. ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* **8**, 3192–3203 (2017).
32. Paruzzo, F. M. *et al.* Chemical Shifts in Molecular Solids by Machine Learning. *ArXiv* **1805**, 11541
33. Bartók, A. P. *et al.* Machine learning unifies the modeling of materials and molecules. *Sci. Adv.* **3**, (2017).
34. Bertini, I., Turano, P. & Vila, A. J. NMR of paramagnetic metalloproteins. *Chem.Rev.* **93**, 2833–2932 (1993).
35. Bologna, N. G. *et al.* Nucleo-cytosolic Shuttling of ARGONAUTE1 Prompts a Revised Model of the Plant MicroRNA Pathway. *Mol. Cell* **69**, 709–719.e5 (2018).

36. Hirano, S. *et al.* Double-sided ubiquitin binding of Hrs-UIM in endosomal protein sorting. *Nat. Struct. Mol. Biol.* **13**, 272–277 (2006).
37. Stovgaard, K., Andreetta, C., Ferkinghoff-Borg, J. & Hamelryck, T. Calculation of accurate small angle X-ray scattering curves from coarse-grained protein models. *BMC Bioinformatics* **11**, 429 (2010).
38. Simkovic, F., Ovchinnikov, S., Baker, D. & Rigden, D. J. Applications of contact predictions to structural biology. *IUCrJ* **4**, 291–300 (2017).
39. Abriata, L., Tamo, G., Monastyrskyy, M., Kryshtafovych, A. & Dal Peraro, M. Assessment of hard target modeling in CASP12 reveals an emerging role of alignment-based contact prediction methods. *Proteins Struct. Funct. Bioinforma.* **86 Suppl 1**, 97–112 (2018).
40. Ovchinnikov, S., Kamisetty, H. & Baker, D. Robust and accurate prediction of residue-residue interactions across protein interfaces using evolutionary information. *eLife* **3**, e02030 (2014).
41. Marrink, S. J., Risselada, H. J., Yefimov, S., Tieleman, D. P. & de Vries, A. H. The MARTINI force field: coarse grained model for biomolecular simulations. *J. Phys. Chem. B* **111**, 7812–7824 (2007).
42. Marko, J. F. & Siggia, E. D. Statistical mechanics of supercoiled DNA. *Phys Rev E* **52**, 2912–2938 (1995).
43. Aspuru-Guzik, A., Lindh, R. & Reiher, M. The Matter Simulation (R)evolution. *ACS Cent. Sci.* **4**, 144–152 (2018).
44. Goddard, T. D. *et al.* UCSF ChimeraX: Meeting modern challenges in visualization and analysis. *Protein Sci. Publ. Protein Soc.* **27**, 14–25 (2018).

45. O'Connor, M. *et al.* Sampling molecular conformations and dynamics in a multiuser virtual reality framework. *Sci. Adv.* **4**, eaat2731 (2018).

46. Balo, A. R., Wang, M. & Ernst, O. P. Accessible virtual reality of biomolecular structural models using the Autodesk Molecule Viewer. *Nat. Methods* **14**, 1122–1123 (2017).

Figure 1

Different implementations of WebGL for AR/VR in web browsers

(A) A lysine and a glutamate side chains attached to different AR markers, whose coordinates are processed in real time to deliver distance and electrostatic potential between charged groups and to calculate and display clashes. Graphics based in A-Frame with WebGL primitives. Accessible at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/smallmolclashdetection.html . (B) Use of a cube marker (made up of 6 different AR markers in its faces) to load any molecule in PDB format and handle and visualize it in 3D. Graphics based in Three.js with WebGL primitives. Accessible at https://lucianoabriata.altervista.org/jsinscience/arjs/jsartoolkit5/pdbloader5.html or with Cannon.js-based rigid body dynamics at https://lucianoabriata.altervista.org/jsinscience/arjs/jsartoolkit5/pdbloader6.html . (C) An object exported from VMD with the OBJ/MTL renderer and loaded onto a marker using pure A-Frame code. The green molecule is a double-stranded segment of DNA, bound to a homeodomain shown as pink cartoons (PDB ID 1FJL). Example accessible at http://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/obj-1fjl.html . (D) Example setup for VR using a cardboard goggle into which the smartphone is inserted (then adjusted to the head with straps, not shown). The example is uses only A-Frame for visualization, extended with JavaScript to detect and show clashes and to make the phone vibrate upon contact. Example accessible at http://lucianoabriata.altervista.org/jsinscience/arjs/arjs-phones.html .

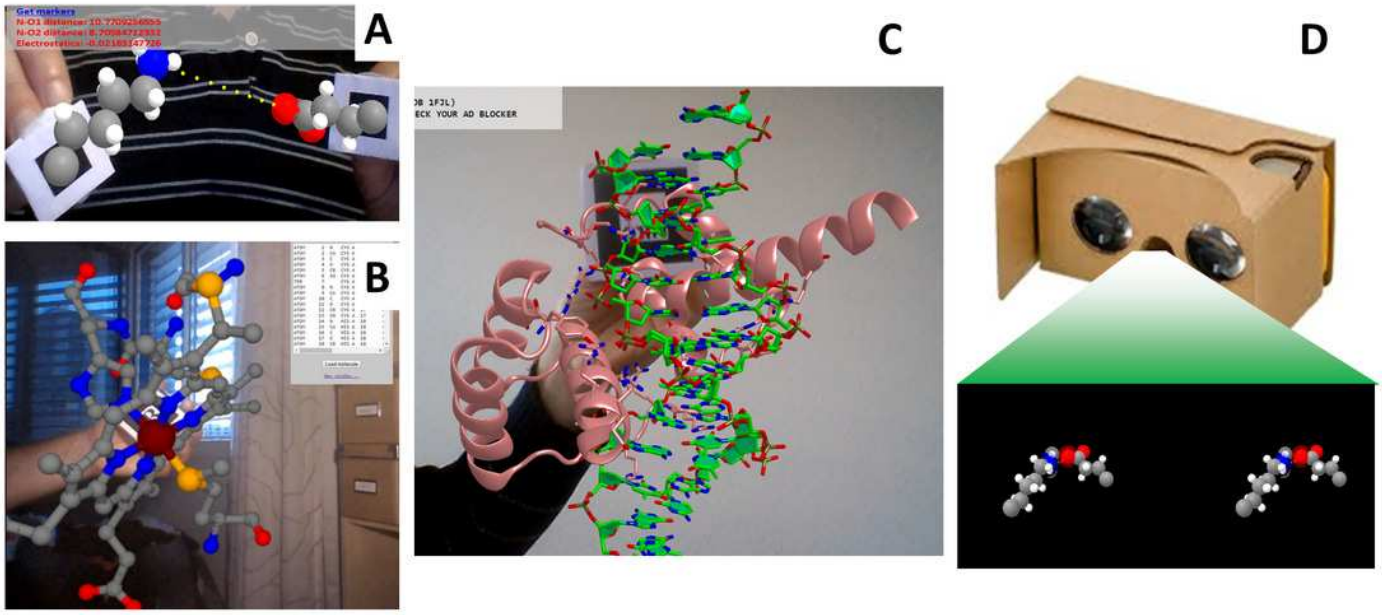

A
B
C
D

Figure 2

Tracking the real world with JavaScript libraries

(A) Kanji (left) and Hiro (right) markers that are built-in into AR.js (more sizes available in Figure S1). (B) Marker-based AR/VR facilitates concurrent view by multiple users. (C) WebGazer.js put to work to control JSmol, so that whatever region of the protein the user looks at comes to the front. I acknowledge Angel Herráez for help integrating both libraries. Example accessible at https://lucianoabriata.altervista.org/jsinscience/jsmolwebgazer/jsmolwebgazer.html .

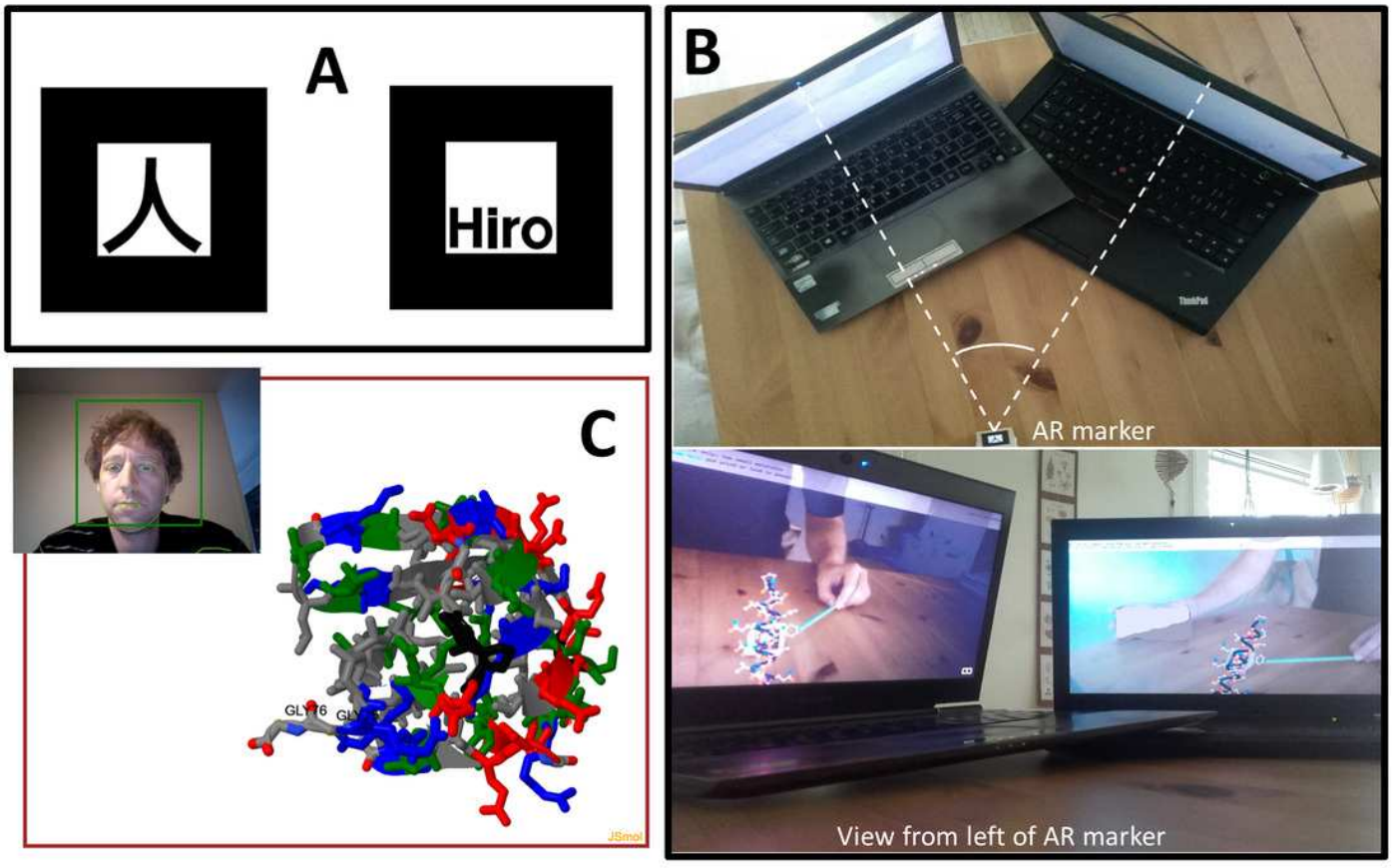


Figure 3

Interactivity in small molecules

(A) The product of a visual emulation of a Diels-Alder reaction, after the user moved the reagents close enough in space; this example further helps to visualize fused rings (as the diene reagent was a ring itself, leading to a double ring after reaction). Accessible at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/smallmoldielsalder.html . (B) As a probe atom (black sphere) is moved around a paramagnetic center, the user sees the paramagnetic effects simulated at the location of the probe atom in real time. Accessible at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/metmyoglobinfe3pcshift.html .

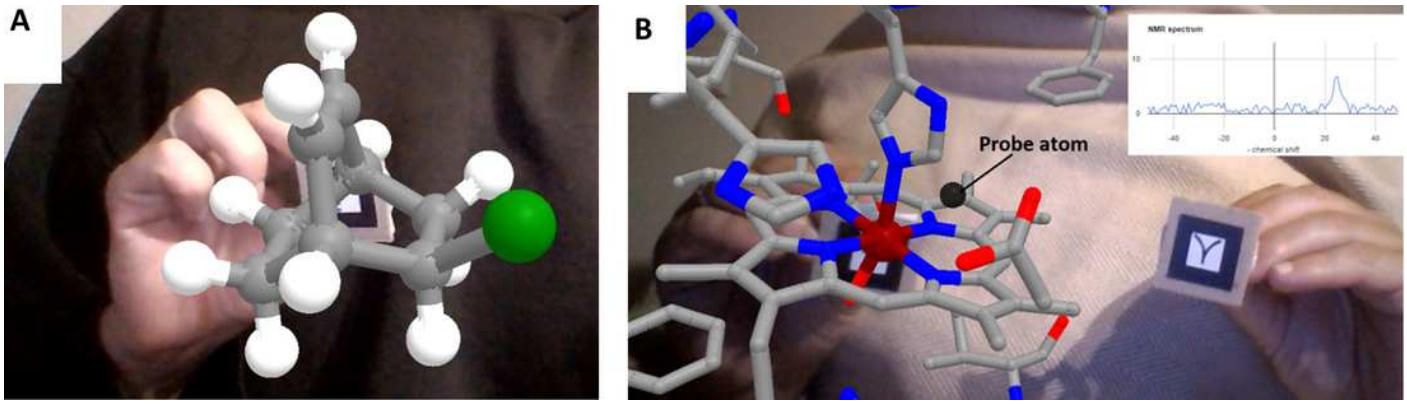


Figure 4

Visualization of biological macromolecules

(A) VMD-derived representation of PDB entry 1VYQ; this is essentially the same example shown by Berry and Board (*Biochem. Mol. Biol. Educ.* 2014) but in a version that runs in web browsers with no installs and no plugins. Example accessible at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/obj-1vyq.html . (B) Representation of an amphipathic alpha helix built with WebGL primitives, accessible at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/ubiquitinandNESatomistic.html

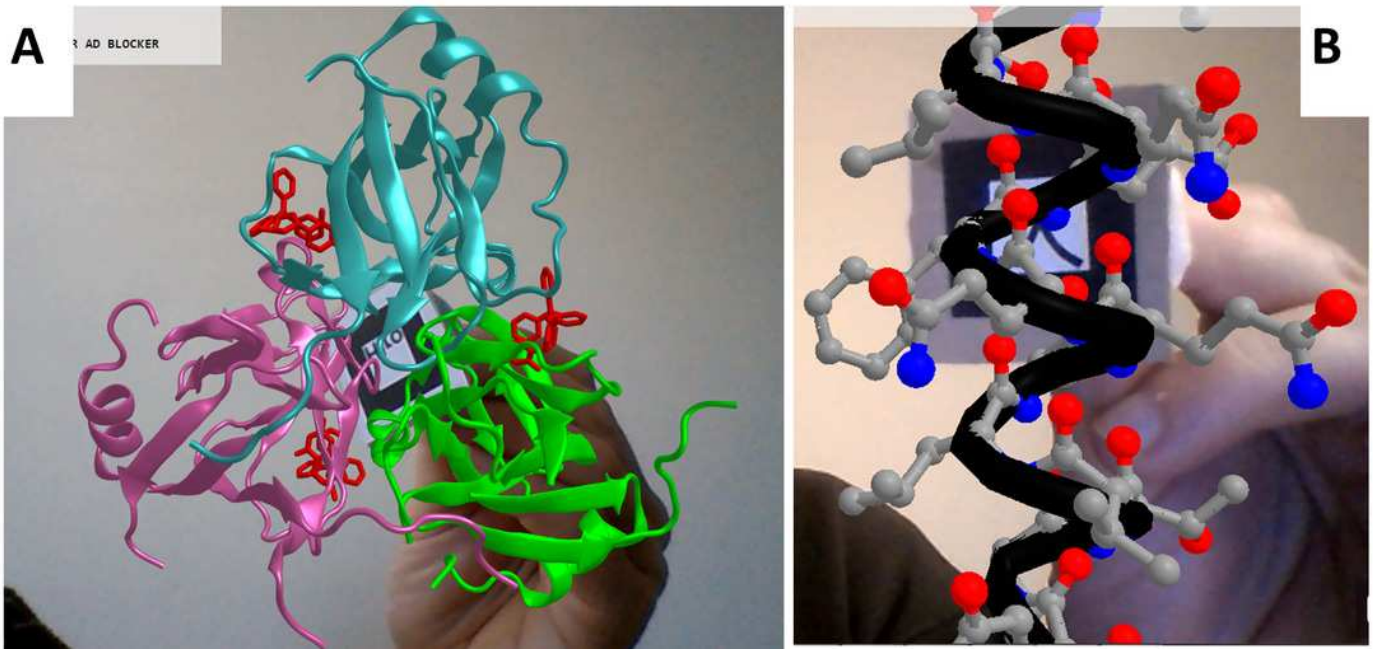
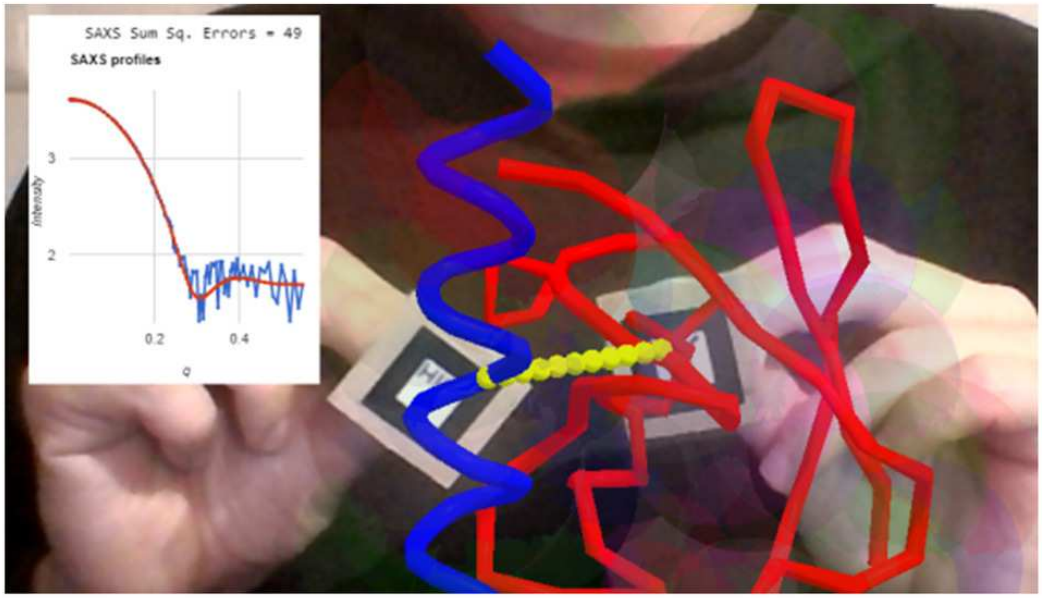


Figure 5

Interactive applications on biological macromolecules

Ubiquitin (red trace) and ubiquitin-interacting motif (blue) driven in 3D with two AR markers, as the web app computes the predicted SAXS profile and displays it overlaid on top of an experimental (simulated with noise) profile, together with a metric of the fit quality. This example can be tested at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/ubiquitinuimffvoicesaxs.html . (B) Interactive exploration of contacts predicted between two proteins (here from coevolution data) before (top) and after (bottom) manual docking. This example is 1QOP from Ovchinnikov et al, where contacts predicted with high score are colored green, contacts of intermediate confidence are olive, and the first contact of low probability is shown red (as taken from the original data). The thickness of the contact lines indicates distance, such that thin lines indicate the residues are close in space. Notice how the red contact, which has low probability, remains thicker than the well-scored contacts (green) upon manual docking. This example can be tested at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/coevol_1qop.html .

A



B

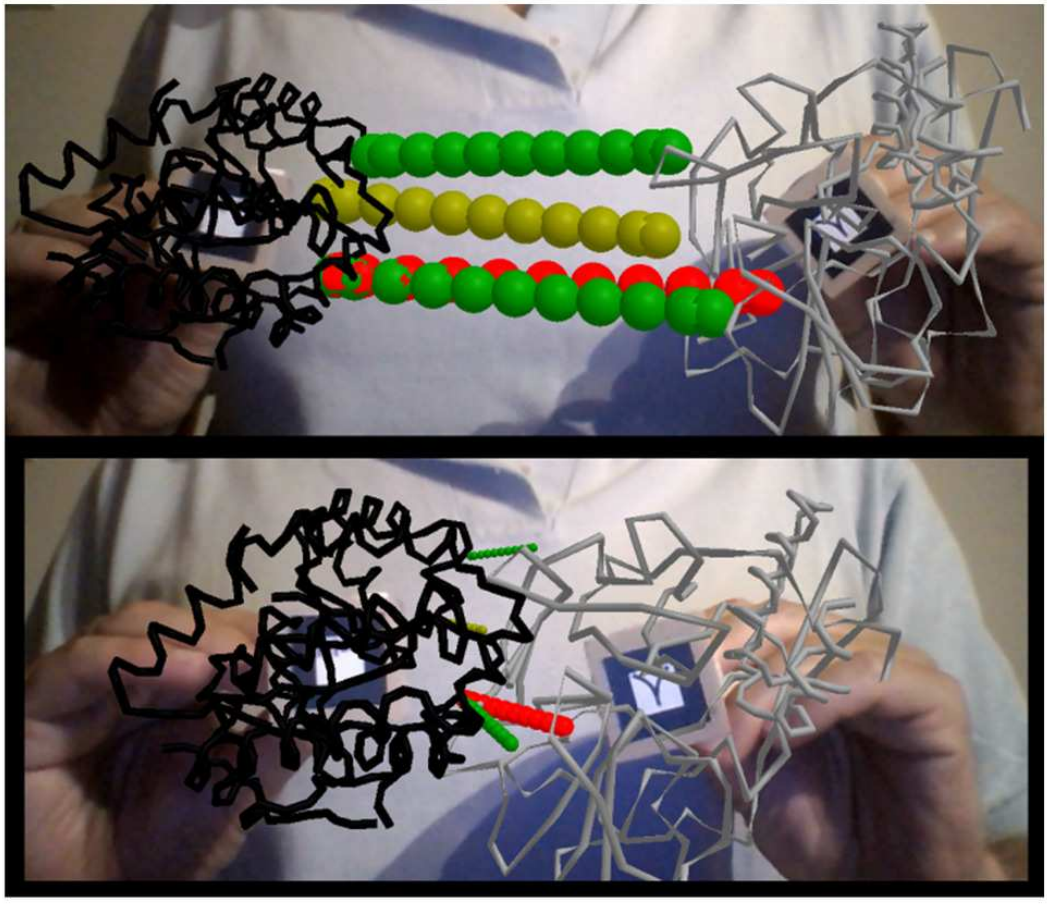
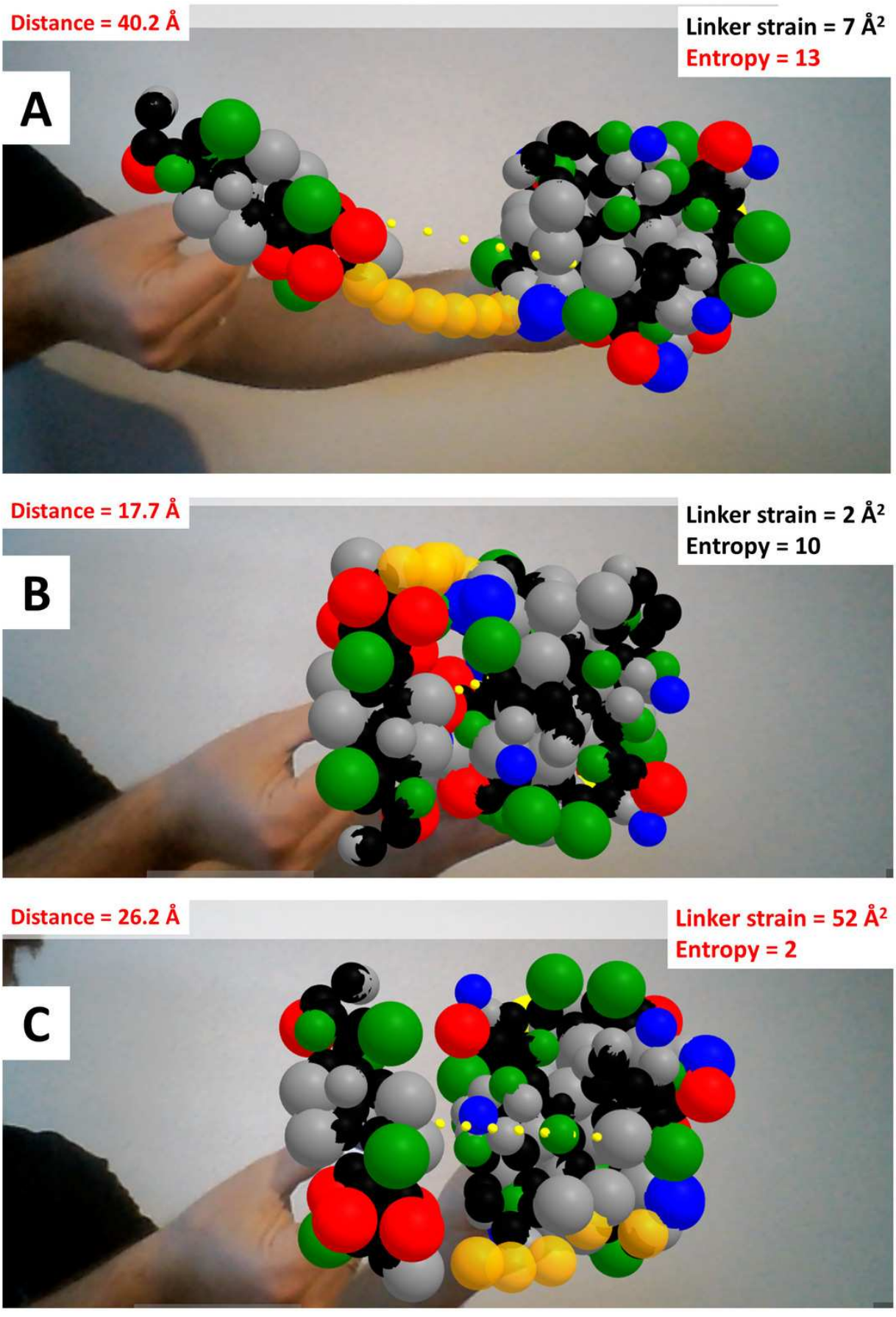


Figure 6

Dynamics of highly disordered segments modeled with rigid-body mechanics

Ubiquitin and ubiquitin-interacting motif modelled as 2-4 beads per residue, colored by physicochemical properties (grey=hydrophobic, red=negative, blue=positive, green=polar uncharged; backbone beads in black). The domains are driven independently in 3D with two AR markers. They are connected through a flexible linker of six backbone-sized beads (orange) whose dynamics are treated with the Cannon.js rigid-body force field. The web app reports in real time the distance between the centers of both domains, the entropy of the linker based on a worm-like chain model, and the linker strain computed from deviations of distances between consecutive linker beads from an equilibrium distance. (A) The two domains extended as much as possible while keeping the linker relaxed (although entropically unfavoured) illustrates the maximum possible separation with the given linker is of around 40 Å. (B) The binding pose between the 2 domains is geometrically feasible as it keeps the the linker relaxed. (C) This other binding pose is unachievable with a linker of this length. This example web app is available at https://lucianoabriata.altervista.org/jsinscience/arjs/armodeling/ubiquitin-uim-cannon.html .


Distance = 40.2 Å
Linker strain = 7 Å²
Entropy = 13
A
Distance = 17.7 Å
Linker strain = 2 Å²
Entropy = 10
B
Distance = 26.2 Å
Linker strain = 52 Å²
Entropy = 2
C