Visualization of Molecular Machinery Using Agent-Based Animation

Daniel Gehrer*
Supervised by: Ivan Viola†

Institute of Computer Graphics and Algorithms
Vienna University of Technology
Vienna / Austria

Abstract

This paper proposes an agent-based model for animating molecular machines. Usually molecular machines are visualized using key-frame animation. Creating large molecular assemblies with key-frame animation in standard 3D software can be a tedious task, because hundreds or thousands of molecular particles have to be animated by hand, considering various biological phenomena. To avoid repetitive animation of molecular particles, a prototypic framework is implemented, that employs an agent-based approach. Instead of animating the molecular particles directly, the framework utilizes a behavior description for each type of molecular particle. The animation results from the molecular particles interacting with each other as defined by their behavior. Interaction between molecular particles is enabled by an abstract model that is implemented by the framework. The methodology for creating the framework was driven through learning by example. Two molecular machines are visualized using the framework. During this process, the framework was iteratively improved. The resulted animations demonstrate that agent-based animation is a viable option for molecular machines.

Keywords: molecular visualization, molecular machines, agent-based visualization

1 Introduction

Every second, an overwhelming number of molecular processes take place in our cells. A great variety of macromolecular complexes are involved in these processes by performing specific tasks. These complexes have some analogies to machines in the macroscopic world, therefore they are often called molecular machines [4].

Biologists examine molecular machines using X-ray crystallography, NMR spectroscopy and cryo-electron microscopy [8]. Based on the spectroscopy data, biologists develop models on how such machines may work. The gained knowledge then has to be conveyed to fellow scientists and students. Textual descriptions are hard to imagine. Simple two-dimensional (2D) visualizations provide better insights, but fail to convey the richness of a three-dimensional (3D) environment [10]. Animated 3D visualizations proved to be powerful tools for describing complex molecular processes to diverse audiences. This kind of visualizations are not only an impactful way to communicate complicated molecular processes but are also widely appreciated for their beauty [6].

Existing 3D software allows biologists to create animated models of molecular machines, but have also drawbacks for this specific task. This kind of software often requires weeks to months of training and regular use to create a basic molecular animation [11]. Sometimes biologists have to hire programmers and animators to create models and animations [16]. One of the major challenges is to model large molecular assemblies which can involve thousands of molecules. Animating such a large number of objects can be very tedious [11].

The goal of this work is to simplify and accelerate the animation process of molecular machines in their environment. This is achieved by identifying repetitious and automatable tasks as a first step. The identified tasks are then used to create a framework that supports the biologist in the animation process. To learn about the visualization and animation process, two molecular machines were visualized and animated from scratch. The lessons learned by each machine are then used to iteratively evolve the framework. Although some tools already exist that simplify the visualization and animation process [9, 14, 16], this very basic bottom up approach is used to learn about the core problems of the process and how to approach them.

The result is a prototypic framework that animates molecular environments using an agent-based model. The model represents an abstract molecular environment. It can be extended and customized to adopt the desired behavior. A lot of animation tasks are abstracted away by the framework. Instead of animating a scene using key-frames, the framework requires the biologist to describe the behavior of molecular particles. These molecular particles act as agents that interact with the environment ac-

*daniel.gehrer@cg.tuwien.ac.at
†viola@cg.tuwien.ac.at

cording to their behavior description. They can respond on environmental conditions like the concentration of specific ligands and can also actively participate by initiating or releasing molecular bonds. A molecular machine is built of molecules [8]. By describing the behavior of each molecule, the functionality of the whole machine can be animated.

2 Related Work

There is a range of tools that support the visualization and animation process. Some of them cover only a small part of the whole visualization process, like providing structural information of molecules [5], or animating molecular structures [1, 12]. But there are also tools that support the biologist through the whole process of importing, animating and rendering molecular environments [9, 16].

Visualization of large molecules is based on scientific data. The Protein Data Bank (PDB) is a publicly available repository for such data. It contains 3D structures of biological molecules. The structures range from tiny proteins and parts of DNA to complex molecular machines [5].

Molecular machines depend largely on docking capabilities of various distinct molecules. KBDOCK is a database system containing binding site information for protein docking. Molecules need some very specific structural and physical properties for docking. KBDOCK uses protein family classifications with coordinate data from PDB to analyze the spatial arrangements of protein interaction [7].

The scientific molecular data is the basis for the visualization and animation of molecular machinery. The next step is to import and visualize this data. A few tools exist that are specialized for this step [1, 3, 12].

The Embedded Python Molecular Viewer (ePMV) is an open-source plugin for the 3D software Blender, Cinema4D and Maya [12]. It allows scientists to import molecular structures from PDB into the 3D software. As soon as the structures are imported, the visualization capabilities of the 3D software can be used to animate and render the scene [12].

A similar tool is Molecular Maya (mMaya) [1]. It is a free plugin for the 3D software Maya and offers functions to import, build and animate molecular structures. The biologist can open PDB files or download them directly from the Protein Data Bank. Molecular Maya supports various representation forms [1].

BioBlender is a software package for visualizing biomolecules in the open source 3D software Blender [3]. The package extends Blender's capabilities to allow biologists to import molecular structures from PDB and provides notable visualization options for special surface properties [3].

cellVIEW is a tool that solely focuses on efficiently rendering large molecular assemblies [13]. The system is capable of interactively visualizing large datasets representing viruses and bacterial organisms consisting of millions of atoms in real-time at 60 Hz display rate. cellVIEW integrates various acceleration techniques like a level-of-detail scheme and dynamic generation of DNA strands directly on the GPU. cellVIEW is freely available to be used and extended [13].

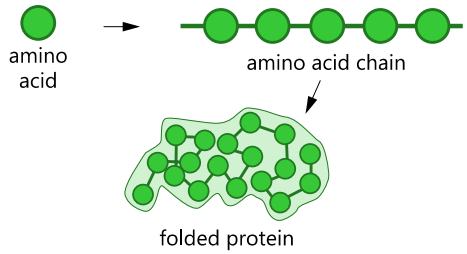

Figure 1: A Protein consists of a chain of linked together amino acids. In a watery environment the chain folds into a stable conformation [2].

Another category of tools supports biologists in creating their own molecular animations [9, 14, 16]. One of these tools is the Molecular Flipbooks [9]. The intended audience are molecular biologists who want to visualize molecular models to communicate their ideas to their peers, their students or the public. It is only meant for simple molecular models and not for complex or cinematic quality animations. Molecular Flipbook utilizes a simple user interface which allows the user to create plain key-frame animations [9].

SketchBio is another tool that enables biologists to rapidly construct molecular animations [16]. It incorporates a two-handed manipulation technique using two six-degree-of-freedom magnetic trackers to edit a scene. That enables novel interaction patterns that accelerate common tasks when animating molecular models. The tool aims for an easier usage than standard 3D software [16].

UCSF Chimera is not only a tool for making simple videos, but a program to interactively visualize and analyze molecular structures [14]. It also makes use of related data like sequence alignments, docking results, trajectories, conformational ensembles, etc. Beside a long list of analysis features, UCSF Chimera can also be used to generate high-quality images and animations [14].

3 Biological Background

All living things are made of cells. Cells are small, membrane-enclosed units filled with water and other chemicals. In an approximate bacterial cell, water makes up 70 % of the cell interior. Macromolecules like DNA, proteins and polysaccharides make up 24 %. The remaining 6 % are phospholipids, small molecules and ions. The composition of an animal cell is similar [2].

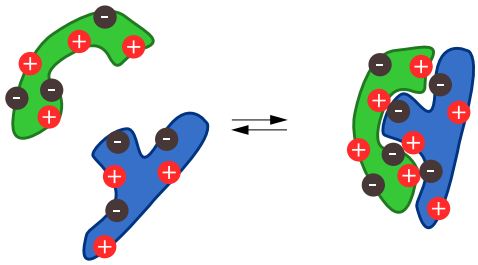

Figure 2: Binding of two molecules due to complementary shape and charges on their surface [2].

Macromolecules are polymers that are constructed by linking smaller subunits (called monomers) together. Especially proteins play a crucial role in cells as they are versatile and can perform thousands of distinct functions. The reason for this versatileness is that a protein is made of a chain of amino acids. Depending on the sequence and the number of amino acids in the chain, the resulting protein can be of very different size and shape. Each amino acid has unique physical properties. Some are negatively or positively charged, some are chemically reactive, some are hydrophobic, etc. Depending on the sequence of the amino acids in the chain, the protein is suitable for different tasks. The cell has capabilities to produce proteins on demand, out of amino acid sequence templates encoded in the DNA [2].

In the water environment of a cell, the amino acid chain folds into a conformation of lowest energy, as illustrated in Figure 1. Normally, each protein folds into a single stable conformation. This conformation, however, often changes slightly when the protein interacts with other molecules in the cell. These small shifts in shape are crucial for its functionality [2].

Large molecules can bind together through complementary shape and charges on their surfaces. A simple illustration of this effect is shown in Figure 2. This kind of binding can be very specific and allow only certain molecules to bind. When multiple macromolecules bind together, they can form large complexes with multiple moving parts that can perform specific tasks [2]. These complexes are called molecular machines [4].

4 Method

Animations are used by biologists to communicate ideas to fellow scientists and students. To create animations, biologists often use standard 3D software to model molecular environments according to their ideas [10]. A few specialized tools also exist. Similar to standard 3D software, these tools also focus on animating 3D objects in a scene using key-frame animation [9, 16]. This paper proposes another way of producing an animation. Instead of animating 3D objects directly, the biologist describes the behavior of molecular particles on a high level. The molecular particles with the behavior description attached, are then placed in an environment. Each molecular particle interacts with other particles according to their behavior description, which causes the animation. If the behavior description is sufficiently detailed, the molecular particles also react correctly, when environmental conditions change. This agent-based approach should speed up the animation process, because not every molecular particle has to be animated by hand anymore. The animation also scales very well, since the description has only to be defined once for each type of molecular particle. In contrast, manual animation requires the biologist to animate each particle independently which can be a very tedious task [10].

Since the goal is to communicate ideas, the biologist should not have to focus on low-level physical properties or forces. Hence, the framework aims to allow the biologist to describe the behavior on a level as high as possible. At the same time, the high-level description must not limit the biologist in achieving an animation that contains all details necessary to communicate the idea. To achieve this, an abstract molecular model was created, that is used for the behavior description. This model includes common molecular details and features that are discussed later.

To test the agent-based animation approach for molecular machines, a prototypical framework was created which implements the agent-based model. The methodology for developing the framework was driven by examples. Two molecular machines were visualized to learn about the challenges of the animation process. The animations were created from scratch. No tool presented in the related work section was extended. The reason for this design choice is to get an in-depth understanding of the challenges involved and avoid being biased by possible solutions. Only cellVIEW is used for efficient rendering [13].

The framework is created and evolved iteratively by following these steps: Choose a simple molecular machine to start with. Get familiar with it and its functionality. Create a framework that can be used to visualize and animate the machine. Make sure that the framework supports all tasks necessary, from importing molecular data, to generating a model for the chosen molecular machine, to render the machine. Also take care the framework does not require the user to do repetitious tasks. Repetitious tasks often expose as molecular features that can be abstracted and reused. When the chosen molecular machine is sufficiently visualized, choose another more complex one. Then repeat these steps with the new machine but instead of creating a new framework, advance the existing one, if necessary. When repeated for a few iterations, the resulting framework should support quite a variety of molecular machines.

4.1 Model

As discussed before, the lessons learned from animating the two molecular machines are used to create an abstract

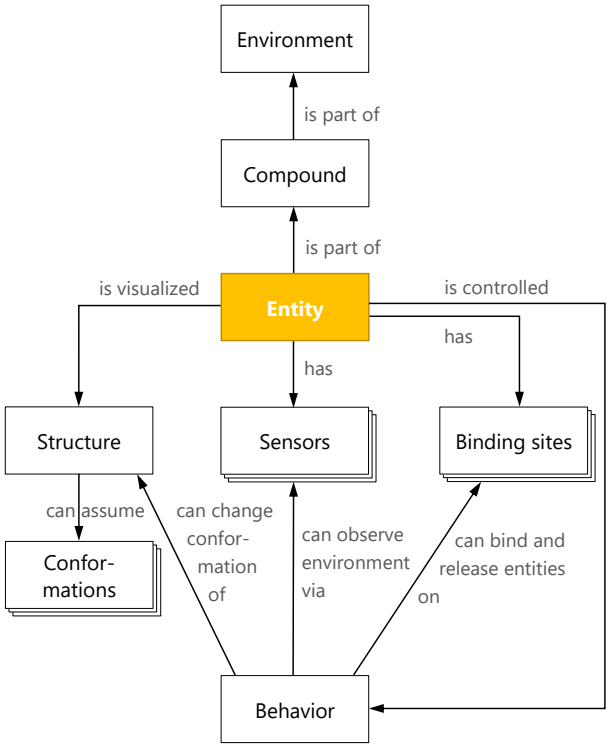

Figure 3: Interaction of model components

model. This model can be extended and customized to animate a desired molecular machine in an agent-based way. Agents are suitable, since molecules can be seen as autonomous units that interact with each other in an environment [8]. An overview over the model is illustrated in Figure 3.

To have a defined space, where all animations happen, the model defines an environment. The environment includes everything that should be visualized. It includes all molecular particles, all behavior descriptions, all 3D models and everything else needed for the visualization. The environment can be seen as the equivalent to a scene in standard 3D software.

In biology, the environment is inhabited by molecular particles. In the model, molecular particles are represented by so-called entities. Entities are defined as the smallest, indivisible molecular particles in the environment. In most cases a molecule equals an entity, but that is not always true. The special cases are discussed later. Entities are the central components in this model. They act as autonomous agents.

In biology, molecular particles move and interact with each other according to physical forces [2]. In existing solutions, these effects have to be animated by the user [9, 16]. Instead of animating each molecular particle by hand, the proposed model requires the user to provide a behavior description for each type of entity. The animation is then generated according to the behavior description.

The atomic structure of a molecule is very important in biology. Depending on its size and shape, it can interact with other molecules [2]. This atomic structure is reflected in the entity structure. As described in Section 3, the shape of certain molecules can shift slightly, which is important for various processes [2]. Such shapes a molecule can assume, are represented in the model as entity conformations. Each entity type has a set of conformations that it can assume.

Other than in biology, in the model the entities are controlled by behavior descriptions. The behavior has to be aware of other entities nearby, since the interaction of molecules is vital in biology. Therefore sensors are introduced to the model. A sensor allows the entity to detect other nearby entities in the environment. It can then decide if it wants to interact with one of them or not. In biology, molecules do not have sensors. The binding and unbinding is a result of physical attraction and repulsion forces. This framework does not aim to be a biological simulator, but a tool to for visualization. Therefore, the concept of a sensor is used to mimic these physical behavior explicitly.

Another very important phenomena in biology is, that molecules dock or bind together to perform a task. Molecules have areas with a specific shape and charge profile where other molecules with complementary shape and charge profile can bind, as described in Section 3 [2]. In the model, such areas are represented as binding sites. Entities that are bound together in the model form a compound.

The following paragraphs describe each concept of the model in more detail:

Entity: Entities are the central objects in an environment. Each entity acts as an agent and can operate individually, according to a defined behavior. Furthermore, entities can have binding sites and sensors attached, that allow the entity to interact with the environment. Its visual representation is defined by its structure.

An entity is the smallest, indivisible molecular particle in the environment. That can be either a molecule, an atom or an ion. Molecules that can split into smaller parts, are not atomic and therefore not represented as single entity. Instead they are represented as compound of smaller entities, in order to be able to split apart. What the smallest, indivisible molecular particles are, is dependent on the environment. For example, an environment that visualizes a biological process that reduces adenosine triphosphate (ATP) to adenosine diphosphate (ADP) for power [2], could represent these molecules using two entities: ADP and phosphate (P_i). ATP would be represented as an ADP entity bound to a P_i entity.

Structure: The structure contains the visual representation of an entity. It contains information about all the atoms that the entity is made of. As discussed in Section 3, certain molecules can change their conformation, which means that the atom positions slightly shift [2]. Each possible conformation an entity can assume is also defined

in its structure. A conformation encloses the spatial location of all entity atoms, as well as the locations and directions of binding sites and sensors. An entity structure can change its color depending on the current conformation for visual guidance.

Conformation changes can be initiated by the behavior. To avoid jumpy conformation changes, the transition between two conformations is animated for a user defined amount of time. Atom positions, binding site and sensor positions as well as binding site and sensor directions are interpolated linearly. The hue, saturation and value components of the color are also interpolated linearly.

Behavior: The behavior is the agent logic of an entity. Each entity is linked to a behavior which controls its interaction with the environment. It can use the sensors to determine the concentration of a specific ligand or find nearby entities. This information can be used to initiate or release bindings on the binding sites or to change the conformation. It is also possible to attract or repel other entities to mimic physical forces. Many simple behaviors can be comfortably described as state machines.

Sensor: Sensors are used by the entity behavior to find other entities that are nearby. To achieve this, the sensor checks for each entity in the environment if it is located inside a cone at the sensor site. The cone is defined by the sensor position and direction that are relative to the origin of the entity. The apex of this cone is located at the specified position, the direction reflects the sensing direction. The size of the cone is dependent on additional aperture and range properties. Aperture is the angle at the apex and range is the height of the cone.

Binding site: At a binding site, an entity can establish a temporary or lasting bond to a binding site of another entity. Therefore, entities with the ability to bind to other entities must have one or more binding sites.

In biology the binding sites are areas with a specific shape and charge profile [2], as described in Section 3. In this model, the complex biological representation is reduced to a single point and a direction. When two entities bind together, they form a compound and are aligned to each other depending on the location and direction of their binding sites. The location and direction of a binding site is stored in each conformation of the entity. As a consequence, binding sites can change their location and direction as a result of a conformation change.

The behavior can bind a binding site of the controlled entity to a binding site of another entity and release such a bond again. When two binding sites bind, it is not checked if they are compatible in a complementary shape and charge way, like it is in biology. Therefore the behavior description is responsible to only bind to compatible binding sites.

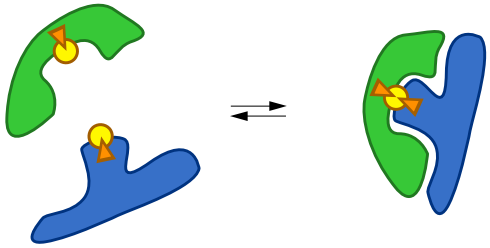

Figure 4: The binding site positions are depicted as yellow circles, the directions as orange arrows. When bound, the positions match and the directions are pointing at each other.

Compound: All entities that are bound together are organized in compounds. Compounds act like one big entity, though with the ability to separate. As soon as one or more entities split, they leave the current compound and form a new one. When an entity is not bound to any other entity, it is still part of a compound, but it is on its own.

The entities in a compound cannot move freely. Their position and rotation relative to each other is well-defined by the position and direction of the binding sites on which the entities are bound together. To enforce these constraints, the position and rotation of each entity is recalculated in every frame. Each compound has an arbitrary root entity. Starting from there, an algorithm traverses along all bindings and recalculates the positions and rotations for each entity.

A binding site is located at a certain position relative to the entity origin. In addition to that position, a direction is provided. When two binding sites are bound, the entities are aligned in such a way that those directions point at each other, and the positions match, as illustrated in Figure 4.

Compounds are not only used to hold entities together but also to locate them spatially in the environment. There are three different types how a compound can be located: The first is ”fixed”. The compound is at a static position and cannot be moved. Fixed is used, for example, when a compound should be placed at a certain position in the cell wall. The second location type is ”floating in compartment”. The environment can be divided into separated areas called compartments. Compounds that are floating, move randomly inside such a compartment, mimicking Brownian motion. The third location type is ”trajectory”. During compounds are attracted or repulsed by an entity, compounds move along a trajectory. Behaviors can change the location type to move around entities in the environment.

5 Results

The result of this work is a framework that implements the model proposed in section 4.1. The framework is implemented using the Unity game engine [15] and utilizes

cellView to render the environment [13]. Furthermore, the implemented framework is used to visualize and animate two molecular machines. At first, the relatively simple Hemoglobin was visualized, then the more sophisticated ATP-Synthase.

5.1 Hemoglobin

Hemoglobin is a molecule for oxygen-transport in red blood cells. It is a small molecular machine that picks up oxygen in the lung, and delivers it where it is needed. Hemoglobin has four binding sites where oxygen can bind to. Oxygen does not binds to all binding sites simultaneously, but one after another. Once the first oxygen is bound, a small change in the conformation is induced. This structural change makes it easier for the next oxygen to bind. Thus, binding the first oxygen is the hardest, but from then it gets easier and easier [2].

In the lungs, where oxygen is plentiful, the first oxygen binds easily and then quickly fills up the remaining binding sites. As the blood circulates through the body, it passes areas where oxygen level is low. Here the Hemoglobin releases its bound oxygen. As soon as the first oxygen drops off, the remaining binding sites release the other oxygen molecules more easily, as the shape changes back [2].

Framework Model

The Hemoglobin is modeled as one entity with 5 conformations: deoxygenated, 25 % oxygenated, 50 % oxygenated, 75 % oxygenated and 100 % oxygenated. The structure is imported from the two PDB entries 1hho (100 % oxygenated) and 2hhb (deoxygenated). Intermediate conformations are linearly interpolated. The Hemoglobin model contains four binding sites and four sensors. One sensor for each binding site.

The behavior is programmed to evaluate the oxygen concentration on each sensor. If the concentration exceeds a certain bind-threshold, the nearest oxygen molecule is attracted and bound to the binding site. The conformation changes depending on the sum of oxygen molecules bound on all binding sites. Moreover, the higher the number of bound oxygen molecules, the lower is the bind-threshold for the next one.

If the measured oxygen concentration on a sensor drops below a release-threshold, the oxygen on this binding site is released again. Like the bind-threshold, the release-threshold is also dependent on the number of oxygen molecules currently bound. A screenshot of the visualization is shown in Figure 5.

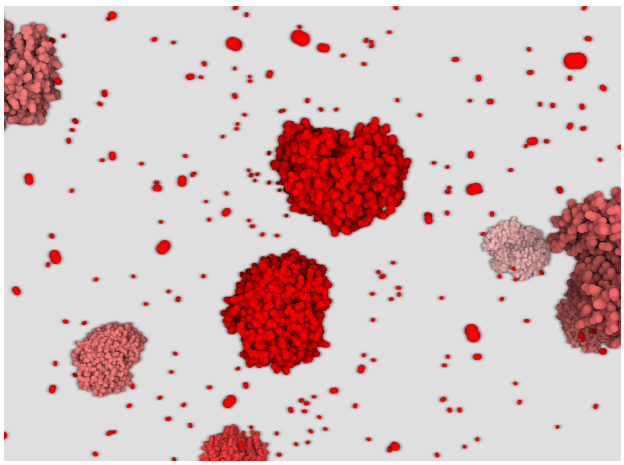

Figure 5: Hemoglobin and oxygen environment. The redder the Hemoglobin, the more oxygen is bound

5.2 ATP-Synthase

Most cellular processes are powered by adenosine triphosphate (ATP). When energy is needed, ATP breaks into adenosine diphosphate (ADP) and phosphate (P_i). This liberates energy that was stored in the bond. The liberated energy is then used to power the cellular process. ATP-Synthase is a molecular machine that binds ADP and P_i back together to form ATP, so that it can be used again. The energy needed for this process is obtained from a proton gradient over a membrane [8].

ATP-Synthase provides a channel for protons from outside the membrane to get inside. Since the proton concentration outside is higher than inside, the protons want to move in. The energy from this proton flow drives a rotor. The rotor, again, drives a generator that is capable of binding ADP and P_i back together [8].

Framework Model

Since this machine is too complex to model using a single entity, the model is composed of different parts. One entity forms the basic frame for the machine. It provides binding sites for the stator, rotor, three $F_1\alpha$ entities and three $F_1\beta$ entities. The parts are shown in Figure 6. The frame also has a sensor attached. It senses to the outside of the membrane enclosed area. As ATP is not atomic, it is not modeled as a single entity. Instead ATP is represented as ADP bound to P_i.

The behavior for the ATP-Synthase is described like this: First, use the sensor to find a proton from outside the membrane. Bind it to the F_0c next to the stator. Rotate the rotor by one step. Release the proton bound to the F_0c that is now next to the stator and release it to the inside of the membrane. These simple steps cause the rotor to rotate. The F_1 parts act dependent on the rotation progress. First they bind to separated ADP and P_i entities. In the next step, these two entities are bound together. In the third step the regenerated ATP is released and the cycle repeats.

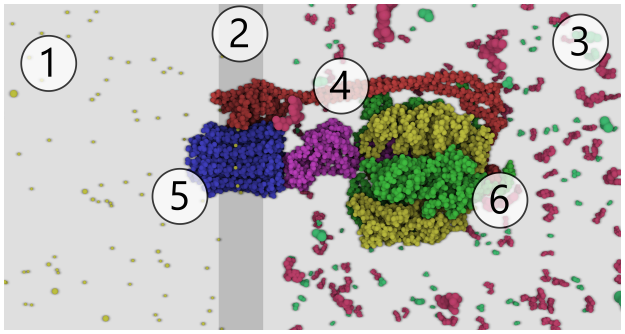

Figure 6: ATP-Synthase. (1) Protons outside membrane, (2) membrane, (3) ADP and P_i inside membrane, (4) stator, (5) rotor consisting of F_0c proteins, (6) $F1$ part consisting of $F_1\alpha$ and $F_1\beta$ proteins

6 Discussion

The framework introduces a novel approach for animating molecular machines. In already existing tools, the biologist animates the molecular environment via key-frame animation [9, 16]. In this framework, the animator provides a behavior description of molecular entities. These entities act as agents and behave according to the provided description in the molecular environment. Usually, the number of distinct entity types is small compared to the total number of entities. Since the behavior description only has to be formulated once for each entity type, such an animation scales very easy, even for a large number of molecular particles in an environment. Moreover, the animated machine reacts automatically to changing environmental conditions, without having to be explicitly animated. This is because the animation is inferred from the behavior description, which defines the desired actions for different conditions.

The framework is only implemented at prototypical level and requires programming skills to create environments. Therefore it is not yet ready to be used productively by biologists. Hence, if and how much time is saved using this approach cannot be measured yet. However, the animations created with the framework demonstrate that agent-base animation is a viable option for visualizing molecular machines.

6.1 Level of Abstraction

One goal of this framework is to find problems in the animation process, then to generalize them and find a solution that suits a broader domain. The challenge is to find a suitable level of abstraction to maximize the benefits for the user.

Independent from the animation approach, the biologist has to provide some sort of description of what should happen in the animation. Finding an appropriate way of describing this is challenging. In current tools, the biologist provides key-frames as description [9, 16]. This approach gives the animator the most control over the animation. As a consequence the animation task often is tedious. On the other extreme, the user could model the physical properties of an entity. The animation would then be computed by a simulator. In that case, the behavior and ultimately the animation would results from physical properties. Such a description is very abstract and highly reusable. However, it gives very less and indirect control over the resulting animation. Since the goal of this framework is to communicate ideas via animations, this is not a suitable approach. The biologist should not have to fine-tune physical properties to achieve an expected animation. It is more straightforward to describe the behavior more directly. Hence, a more low-level form of behavior description is appropriate.

This framework aims for an abstraction level that has the most advantages for the biologist. It should be abstract enough to avoid repetitive tasks, yet not take away the ability to precisely influence the resulting animation. It is intended to find a certain level of abstraction for the framework that allows to describe most machines easy and well, but does not limit the possibilities either. The chosen level of abstraction provides a good starting point but may be adjusted in future work to better fulfill the requirements.

6.2 Limitations

The framework does not include a graphical user interface (GUI) for designing environments. This makes it difficult to precisely model spatial components like binding sites and sensors. Assembling multiple PDB datasets together is also a challenging task without visual guidance. Since only an application programming interface (API) is provided, the framework is only usable by users that are already experienced in programming.

Animations of conformation changes in entities are interpolated linearly. This is not realistic since it can happen that two or more atoms are at the same spatial position during the animation. An interpolation mode which considers some constraints would be more suitable.

6.3 Future work

The framework depicts only a few biological features of two well-known molecular machines. Future work can focus on finding and abstracting more biological features, to support a broader domain of molecular machines. Such features could be, for example, elasticity in molecular structures or degrees of freedom on binding sites [2].

To make the framework more attractive for biologists, future work should provide a GUI for creating and editing molecular environments. The API as only way to interact with the framework is also a limiting factor for broader usage. Focusing on a more intuitive way of describing entity behavior, for example via a domain specific language could be also beneficial.

References

[1] Molecular maya toolkit – mmaya. http://www.molecularmovies.com/toolkit/, 2017. Accessed: 2017-01-31.

[2] Bruce Alberts, Alexander Johnson, Julian Lewis, David Morgan, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of THE CELL*, volume 6. Garland Science, 2014.

[3] Raluca Mihaela Andrei, Marco Callieri, Maria Francesca Zini, Tiziana Loni, Giuseppe Maraziti, Mike Chen Pan, and Monica Zoppè. Intuitive representation of surface properties of biomolecules using bioblender. *BMC Bioinformatics*, 13(4):S16, 2012.

[4] Roberto Ballardini, Vincenzo Balzani, Alberto Credi, Maria Teresa Gandolfi, and Margherita Venturi. Artificial Molecular-Level Machines: Which Energy To Make Them Work? *Accounts of Chemical Research*, 34(6):445–455, 2001.

[5] H M Berman, J Westbrook, Z Feng, G Gilliland, T N Bhat, H Weissig, I N Shindyalov, and P E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

[6] Katrina T Forest, Christopher P Hill, Katrina T Forest, and Christopher P Hill. ScienceDirect Editorial overview : Macromolecular machines and assemblies : Rise and fall at the molecular level. *Current Opinion in Structural Biology*, 31:vii–viii, 2015.

[7] Anisah W. Ghoorah, Marie-Dominique Devignes, Malika Smal-Tabbone, and David W. Ritchie. KB-DOCK 2013: a spatial classification of 3D protein domain family interactions. *Nucleic Acids Research*, 42(D1):D389, 2013.

[8] David S. Goodsell. *Bionanotechnology: Lessons from Nature*. John Wiley & Sons, Inc., 2004.

[9] Janet Iwasa, Gael McGill, Piotr Sliz, Ronald Mourant, Mike Pan, and Rise Riyo. Molecular flipbook. https://www.molecularflipbook.org/, 2017. Accessed: 2017-01-31.

[10] Janet H. Iwasa. Animating the model figure. *Trends in Cell Biology*, 20(12):699 – 704, 2010. Special issue - CellBio-X.

[11] Janet H Iwasa. ScienceDirect Bringing macromolecular machinery to life using 3D animation. *Current Opinion in Structural Biology*, 31:84–88, 2015.

[12] Graham T. Johnson, Ludovic Autin, David S. Goodsell, Michel F. Sanner, and Arthur J. Olson. epmv embeds molecular modeling into professional animation software environments. *Structure*, 19(3):293 – 303, 2011.

[13] Mathieu Le Muzic, Ludovic Autin, Julius Parulek, and Ivan Viola. cellview: a tool for illustrative and multi-scale rendering of large biomolecular datasets. In Katja Bühler, Lars Linsen, and Nigel W. John, editors, *Eurographics Workshop on Visual Computing for Biology and Medicine*, pages 61–70. EG Digital Library, The Eurographics Association, September 2015.

[14] Eric F. Pettersen, Thomas D. Goddard, Conrad C. Huang, Gregory S. Couch, Daniel M. Greenblatt, Elaine C. Meng, and Thomas E. Ferrin. UCSF Chimera - A visualization system for exploratory research and analysis. *Journal of Computational Chemistry*, 25(13):1605–1612, 2004.

[15] Unity Technologies. Unity – Game Engine. https://unity3d.com, 2017. Accessed: 2017-02-06.

[16] Shawn M Waldon, Peter M Thompson, Patrick J Hahn, and Russell M Taylor. SketchBio: a scientist's 3D interface for molecular modeling and animation. *BMC Bioinformatics*, 15(1):1–17, 2014.