FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Instant Construction of Atomistic Models for Visualization in Integrative Cell Biology

DISSERTATION

zur Erlangung des akademischen Grades

eingereicht von

Dipl.-Ing. Tobias Klein
Matrikelnummer 01129861

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Assoc. Prof. Dipl.-Ing. Dr.techn. Ivan Viola
Zweitbetreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller

Diese Dissertation haben begutachtet:

Thomas Ertl	Timo Ropinski

Wien, 23. November 2019

Tobias Klein

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Instant Construction of Atomistic Models for Visualization in Integrative Cell Biology

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

by

Dipl.-Ing. Tobias Klein
Registration Number 01129861

to the Faculty of Informatics

at the TU Wien

Advisor: Assoc. Prof. Dipl.-Ing. Dr.techn. Ivan Viola
Second advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller

The dissertation has been reviewed by:

Thomas Ertl	Timo Ropinski

Vienna, 23rd November, 2019

Tobias Klein

Erklärung zur Verfassung der Arbeit

Dipl.-Ing. Tobias Klein

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 23. November 2019

Tobias Klein

Kurzfassung

Ergebnisse rechnergestützter Modellierung haben die integrative Zellbiologie auf vielfälltige Weise weiterentwickelt. Insbesondere im biologischen Mesoskalenbereich, der Skala zwischen Atomen und zellulären Umgebungen, verbessern Computermodelle das Verständnis und die qualitative Analyse biologischer Strukturen. Die Mesoskala ist ein bedeutsamer Bereich, da sie den Bereich der Skalen darstellt, die für einzelne experimentelle Techniken nicht vollständig zugänglich ist. Komplexe molekulare Zusammensetzungen innerhalb dieser Größenordnung können mit Hilfe von Kristallstrukturanalyse visualisiert werden, allerdings nur in Isolation. Modelle in der Mesoskala zeigen, wie Moleküle zu komplexeren subzellulären Umgebungen zusammengefügt werden. Diese Molekülkomplexe bilden die grundlegenden Prozesse des Lebens. Durch die geschickte Kombination von Resultaten aus bildgebenden und experimentellen Methoden können Wissenschaftler einen Einblick in die stattfindenden Prozesse erlangen. Erst in jüngster Zeit haben Biologen damit angefangen, die Resultate verschiedener biologischer Quellen zu vereinheitlichen. Durch diesen Ansatz ist es möglich, komplexe biologische Umgebungen wie Viren oder Bakterien digital zu modellieren und anschließend zu visualisieren. Aus diesen und folgenden Gründen leben wir derzeit in einer richtungweisenden Zeit für die Erforschung der integrativen Strukturbiologie. In erster Linie macht die Fülle der vorhandenen Daten, durch Quellen wie bekannte Online-Datenbanken, Strukturinformationen über biologische Objekte öffentlich zugänglich. Darüber hinaus bildet der Fortschritt von parallelen Prozessoren die Grundlage für die direkte Konstruktion und Visualisierung großer biologischer Umgebungen im atomistischen Detail. Zusätzlich ermöglichen neue wissenschaftliche Fortschritte in der Visualisierung die effiziente Darstellung komplexer biologischer Phänomene mit mehrenen Millionen von Struktureinheiten.

In dieser kumulativen Dissertation, präsentieren wir eine Vielzahl von neuen Techniken, die den sofortigen digitalen Aufbau von Strukturen der Mesoskala erleichtern. Die gemeinsame methodische Grundlage der Techniken und Erkenntnisse in dieser Arbeit ist "Rechnen statt Speichern". Dieser Ansatz reduziert die Komplexität der Speicherverwaltung und ermöglicht die sofortige Änderung der erstellten Modelle. In Kombination sind die vorliegenden Techniken in der Lage, große biologische Umgebungen unter Verwendung der grundlegenden strukturellen Bausteine von Zellen interaktiv zu konstruieren. Diese Bausteine bestehen hauptsächlich aus Nukleinsäuren, Lipiden und

löslichen Proteinen. Für die Erzeugung von langen linearen Polymeren, die aus Nukleinsäuren bestehen, präsentieren wir eine parallele Konstruktionstechnik, die einen Mittelpunktsverschiebungsalgorithmus verwendet. Die effiziente Erzeugung von Lipidmembranen wird durch einen Textursyntheseansatz realisiert, der das Wang-Kachelung-Konzept ausnutzt. Für die Verteilung löslicher Proteine wird ein stufenweiser paralleler Algorithmus vorgestellt. In diese Arbeit päsentieren wir die Integration der direkten Modelierung in eine visuelle Umgebung und beleuchten wie dies mehrere Aspekte signifikant verbessert. Erstens ermöglicht die Integration ein sofortiges Feedback über die erstellten Strukturen und die Ergebnisse von Parameteränderungen. Darüber hinaus bildet die Integration von Konstruktion in die Visualisierung die Grundlage für Visualisierungssysteme, die darauf abzielen, große biologische Lebenswelten direkt zu erstellen. Dies fördert die qualitative Analyse biologischer Umgebungen, welche eine Vielzahl von synthetisierten Modellen benötigen. Um die Physiologie biologischer Modelle der Mesoskala der Allgemeinheit näherzubringen, wird in ein neuartiges Konzept präsentiert, welches die Erstellung von prozeduralen Animationen auf mehreren Skalen vereinfacht.

Abstract

Computational models have advanced research of integrative cell biology in various ways. Especially in the biological mesoscale, the scale between atoms and cellular environments, computational models improve the understanding and qualitative analysis. The mesoscale is an important range, since it represents the range of scales that are not fully accessible to a single experimental technique. Complex molecular assemblies within this scale have been visualized with x-ray crystallography, though only in isolation. Mesoscale models shows how molecules are assembled into more complex subcelluar environments that orchestrate the processes of life. The skillful combination of the results of imaging and experimental techniques provides a glimpse of the processes, which are happening here. Only recently, biologists have started to unify the various sources of information. They have begun to computationally assemble and subsequently visualize complex environments, such as viruses or bacteria. Currently, we live in an opportune time for researching integrative structural biology due to several factors. First and foremost, the wealth of data, driven through sources like online databases, makes structural information about biological entities publicly available. In addition to that, the progress of parallel processors builds the foundation to instantly construct and render large mesoscale environments in atomistic detail. Finally, new scientific advances in visualization allow the efficient rendering of complex biological phenomena with millions of structural units.

In this cumulative thesis, we propose several novel techniques that facilitate the instant construction of mesoscale structures. The common methodological strategy of these techniques and insight from this thesis is “compute instead of store”. This approach eliminates the storage and memory management complexity, and enables instant changes of the constructed models. Combined, our techniques are capable of instantly constructing large-scale biological environments using the basic structural building blocks of cells. These building blocks are mainly nucleic acids, lipids, and soluble proteins. For the generation of long linear polymers formed by nucleic acids, we propose a parallel construction technique that makes use of a midpoint displacement algorithm. The efficient generation of lipid membranes is realized through a texture synthesis approach that makes use of the Wang tiling concept. For the population of soluble proteins, we present a staged algorithm, whereby each stage is processed in parallel. We have integrated the instant construction approach into a visual environment in order to improve several aspects. First, it allows immediate feedback on the created

structures and the results of parameter changes. Additionally, the integration of construction in visualization builds the foundation for visualization systems that strive to construct large-scale environments on-the-fly. Lastly, it advances the qualitative analysis of biological mesoscale environments, where a multitude of synthesized models is required. In order to disseminate the physiology of biological mesoscale models, we propose a novel concept that simplifies the creation of multi-scale procedural animations.

Contents

Part I

Introduction

CHAPTER 1

Overview

Visualization and computer graphics have experienced significant advances in the procedural modeling of nature. Many natural phenomena, such as water, smoke, fire, wood, and stone have already been procedurally generated for several decades [EM03]. Also man-made structures such as the facade of buildings [IMAW15] or even entire urban landscapes [SKK+14b, SKK+14a] can be efficiently generated in a procedural manner. Technological advances in structural biology coupled with the rapid increase in computational capabilities have now opened new possibilities to construct and analyze large biological systems. There has been a shift from studying individual proteins to analyzing larger systems, such as viruses, bacteria, or even portions of eukaryotic cells. The modeling of a whole cells is generally seen as "the grand challenge of the 21st century" [Tom01]. This includes the integration of heterogeneous datasets to provide a unified representation of knowledge about different organisms [ILO+16]. Many of the processes that happen inside the cells and viruses of our body cannot be understood by observing smaller structures separately. However, already small structures are challenging to visually capture, since molecular structures are typically smaller than the wavelength of light, and thus cannot be directly observed. If taken out of their environment, experimental techniques, such as X-ray crystallography, NMR spectroscopy, and electron microscopy provide information about molecules in up to atomistic resolution. These techniques capture either various characteristics of small molecular structures, or information about larger complexes on the cellular scale. Nevertheless, the biological mesoscale, the intermediate scale between molecular and cellular biology, is still hard to observe. Only a combination of different experimental techniques reveals a glimpse of the processes that are happening here. In general, the biological mesoscale concerns how molecules are assembled into more complex subcelluar environments that orchestrate the processes of life. Biologists have started to unify various sources of information about the biological mesoscale. There are first prototypes, which computationally assemble complex biological environments, such as

viruses or bacteria, from smaller molecular building blocks.

The visualization and dissemination of the knowledge about such biological environments is a complex task, but many approaches have already been developed. Visualization of molecular structures has a rich history [MKK+19], where in the past, visual representations focused on small structures like atoms and the bonds between them. With the technological progress of the GPU, new systems and techniques have emerged that are capable to render several billions [MAPV15] or trillions [SMK+16] of particles in realtime. They do not only rely on the computational power of the GPU but also on advanced Level of Detail and Occlusion Culling techniques. Additionally, much research has been conducted to visually represent molecular structures. Dynamic multi-scale color schemes [WWLM+19], sophisticated cutaways [LMMS+16], and labeling methods [KČK+18] are only a few examples that provide effective methods for the perception enhancement of the displayed structures. While these approaches facilitate the visualization of molecular structures, they are usually separated from the data acquisition.

Endeavours to disseminate gathered knowledge about cellular environments are generally laboriously hand-crafted, made from highly skilled experts that are educated in biology, as well as in illustrating or 3D modeling. Examples of such work include the well-known illustrations by Wilhelm et al. [WMT+14] or David Goodsell [Goo91]. They are either created by hand drawings or with a combination of 3D modeling and visualization tools. Public databases [BWF+00] offer detailed information about the shapes of proteins, nucleic acids, and complex assemblies. However, larger environments are usually still created by manual methods and displayed in 2D illustrations or 3D visualizations. More recent approaches [JAAA+15, DA14] make use of computation to synthesize biological environments with vastly more detail than images that result from microscopy data. These environments are assembled on the basis of smaller structures, of which the exact atomistic arrangements are known. The main drawback of the existing constructive approaches is their computation time, which spans from minutes up to hours depending on the desired quality. This makes the construction a slow offline process, which does not provide interactive feedback.

In this thesis, three publications are presented, introducing novel constructive techniques for the visualization of molecular structures. Our work shows that the tight integration of construction into visualization opens up new opportunities to display larger, more complex structures including their dynamic behavior. The integration also facilitates the generation of molecular structures on the GPU to directly visualize them without any further data transfer. The presented research acts as the foundation for visualization systems, which strive to construct biological environments instantaneously in order to show, analyze, and hypothesize about limitless biological environments.

1.1 Problem Formulation

The challenge of constructing and visualizing biological mesoscale structures starts with understanding the hierarchical organization of living systems. On the lowest level, atoms build bonds to form larger molecules like proteins or nucleic acids. On higher levels, molecules compose into viruses, bacteria, or eukaryotic cells. These mesoscale environments have a distinct multi-scale behavior and comprise a multitude of instances of only a few different protein types. Blood plasma, for example, consists of mostly water (up to 95% by volume) and a few different types of dissolved proteins, which are mainly serum albumins, globulins, and fibrinogen. More complex environments, such as the HIV Virion [JGA+14], also consist of only small magnitudes of different protein types. This fact can be computationally leveraged through multi-instance techniques, which have the potential to efficiently construct and visualize large and complex biological environments. Biological mesoscale environments are usually organized into several compartments. Compartments build the shape of biological structures and are often composed of lipids forming a membrane. The membrane lipids are tightly packed, and thus feature difficult characteristics from the perspective of digital construction. On top of the structural challenges for mesoscale construction comes the task to explain their functions. The behaviour of living systems cannot be effectively explained with static models. Keyframed animations are the typical approach to explain how a mesoscale structure works. However, they are tedious to create and are sometimes difficult to change when new findings are made. In addition to that, the construction of the structure and the animation have to take the multiscale nature of living systems into account. Such systems exhibit a vastly different behavior and structural properties depending on the scale.

Typically, visualization has been dealing with multiscale aspects in the sense that there was measured data or simulated models from which the depiction was directly inferred. Therefore, challenges were associated with storage and efficient data transfer to computational resources that generate the image. A classic example is Guthe's Wavelet-based real-time rendering framework [WS04]. The situation in this work is different, since we no longer have one simulation, or one measured dataset. The integrative sciences create models that are assemblies of knowledge from various areas. The overall biological environment was never observed as such, it is a synthetic geometric representative of all the knowledge that has been put into the process of creating it. For this reason, the digital representation of the environment is rather a set of rules than a particular geometric instance. However, scientists still want to study it through the geometric representation of exemplary structures. Under these conditions, technological approaches can apply a strategy, which is also known from procedural modeling, to the construction of digital biological environments: instead of storing a model and managing its transfer, the model is computed instantly. The methodological strategy and key insight of this thesis is: "compute instead of store". This eliminates the storage and memory management complexity, and enables instant changes of the model, so that parameter changes instantly recompute the entire model.

The instant construction approach is a key technology for hypothesis generation and visualization. While this strategy can be applied everywhere where visualization of representative models is appropriate, instead of particular acquired data (e.g., medical diagnostic examination), this thesis describes a focused set of solutions for bridging structural biology from molecules to cells. Therefore, the basic building blocks of a cell drive the scientific curiosity and technological advances presented in this thesis. A cell consists of compound structures, such as fibrous elements forming elongated genetic molecules, or lipids mapped on 2D manifolds of a membrane surface, or soluble proteins distributed in 3D space. As biology is dynamic, the animation of biological environments needs to be considered as well. For all these aspects the mission statement: "compute instead store" is systematically investigated. To efficiently construct animated biological mesoscale models, the following research questions are addressed:

- *How can linear polymers be instantaneously constructed?*
- *How can lipid membranes be efficiently mapped on a texture patch?*
- *How can molecular structures be efficiently packed in a compartment?*
- *How can overlapping molecular structures be relaxed in realtime?*
- *How can the molecular construction approach be integrated into the visualization?*
- *How can constantly changing domain knowledge be incorporated into the construction and visualization approach?*
- *How can molecular structures be procedurally animated in a multi-scale manner?*

1.2 Research Goals and Methodology

Manually constructing digital biological systems in their finest detail would take a lifetime of a biologist. On the other hand, substantial information about these systems is already available through public databases and experimental research data. The overall goal of this thesis is to create a symbiosis between construction and visualization algorithms for integrative structural biology to disseminate available knowledge and drive the hypotheses generation and validation of new research ideas. In this thesis, we focus on the biological mesoscale, where many structures and processes are still not completely understood. We subdivide the overall research aim into the following goals:

- Goal 1: A problem characterization for the construction and animation of multi-scale biological mesoscale models.
- Goal 2: An efficient algorithmic formulation to construct and visualize linear polymer structures.

- Goal 3: A texture synthesis approach for the construction of a non-periodic mapping of densely packed 3D structures on arbitrary shapes.
- Goal 4: An instant packing technique to populate structures in 3D compartments.
- Goal 5: A multi-scale animation approach to disseminate biological physiology.
- Goal 6: A joint construction and visualization of integrative structural biology.

The research process of this thesis was strongly accompanied by domain scientists who supported and critically validated the results with respect to the presented research goals. First and foremost, the research was driven through a close collaboration with structural and computational biologists from the Scripps Research Institute. They have participated in numerous discussions of the research direction, potential solutions and also the evaluation of the developed techniques and results. Furthermore, we have evaluated the effectiveness of our techniques and results with various other domain scientists ranging from professional animators, illustrators of biological images, biochemists, and experts in biomedical visualization. Computational aspects were compared to the performance of previous techniques and solutions. Whenever possible, we have measured the correspondence of the biological construction results to experimental data obtained from real-world examples.

1.3 Background and Related Work

"Structural characterization of biomolecular mechanisms across a broad spectrum of scales is key to our understanding of life at the molecular level" [ILO+16]. The characterization is driven by the wealth of information originating from the revolution in imaging techniques, with examples like cryo electron microscopy (cryo-EM), cryo electron tomography (cryo-ET), and superresolution light microscopy. Also the information of biological molecules and their roles, relationships, and actions in cells has grown tremendously through the so called 'omics' revolution [LSM+19]. This includes the study of genomes (genomics), the examination of proteins (proteomics), and the analysis of chemical processes involving RNA molecules (transcriptomics) and metabolites (metabolomics). Experimental methods are constantly improving the knowledge about mesoscale structures and are extending their applicability to larger subjects. Recent advances in imaging techniques, like liquid-cell electron tomography [DSV+19], even facilitate to observe specimen in solution. Using cryo-EM, the specimen is typically frozen and trapped in a static state of existence, representing only a single moment of the lifetime of the sample. Liquid-cell electron tomography, on the other hand, encloses the specimen in a solution allowing experts to image materials in 3-D. Figure 1.1 depicts such an image captured with liquid-cell electron tomography. However, imaging techniques usually still fall short of being able to determine the atomic structure of living cells. For example, cryo-EM provides the detailed structure of the envelope of the Zika virus, but the inside, i.e., the randomly

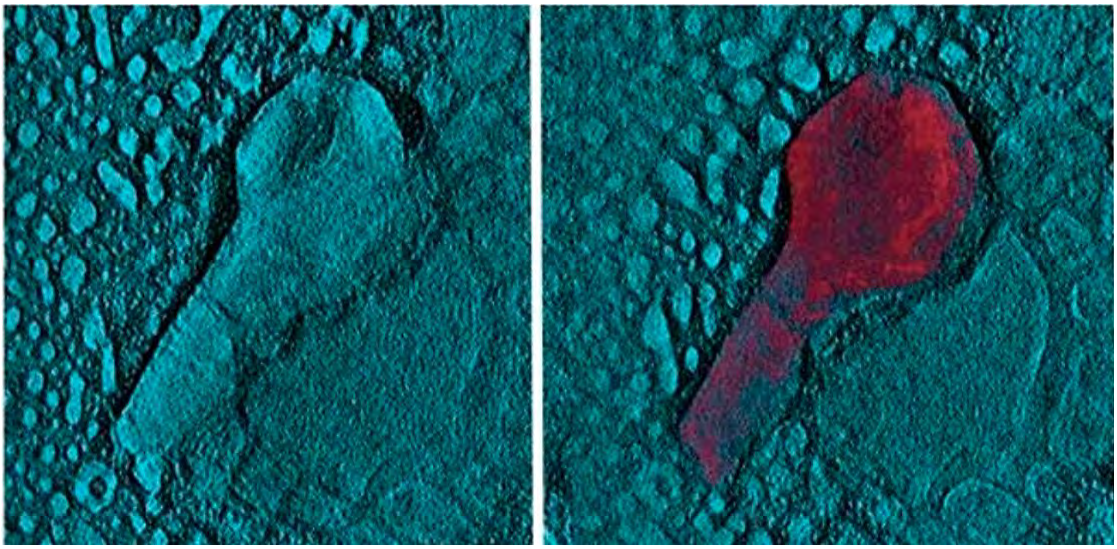
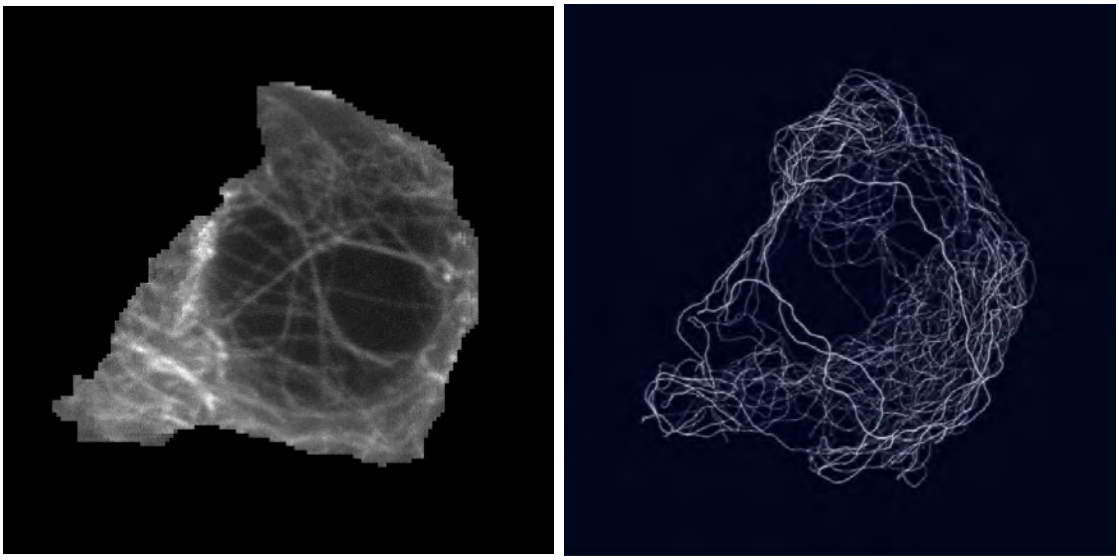


Figure 1.1: Unsegmented (left) and segmented (right) image of a bacteriophage, captured through liquid-cell electron tomography [DSV+19]. In contrast to cryo-EM, liquid-cell electron tomography facilitates to observe specimen in solution instead of in a frozen and static state. This enables the imaging of materials in 3-D. Image source: Dearnaley et al. [DSV+19].

arranged genome, is still largely inaccessible to experiments [SCS+16]. Many groups use computational approaches to construct mesoscale models with atomic detail, combining experimental techniques with atomic structural information [ILO+16, RS16]. The majority of research focuses on the individual structures of the mesoscale, which can be grouped into membranes, fibrous structures, and soluble components. For instance, the tool lipidWrapper [DA14] and the work by Jo et al. [JLKI09] present specialized algorithms to construct lipid membranes. Lipids are rather small molecules consisting of only several dozens of atoms. They consist of a head group that is soluble in water and one or more tails that are insoluble. The lipids themselves are roughly oriented perpendicularly with respect to the membrane surface. Their dense packing on the membrane makes them a particular challenge in computational modeling. Other tools like GraphiteLifeExplorer [HLLF13] focus on modeling the structure of DNA and its interaction with proteins. DNA is one of the most omnipresent structures in mesoscale systems and is basically a long polymer of four different types of nucleotides carrying the genetic information. Extremely long strands of DNA are tightly packed into small compartments. For instance, human cells compact two meter of DNA (lined-up) into a cell nucleus of about five micrometers in width. The spatial architecture and dynamics of the genomic material play an important role in biological processes [RZ14]. While eukaryotes achieve the tight packing by wrapping the DNA around proteins called histones, most prokaryotes do not have histones and achieve the packing of DNA through supercoiling. Supercoiling describes the process that already coiled DNA

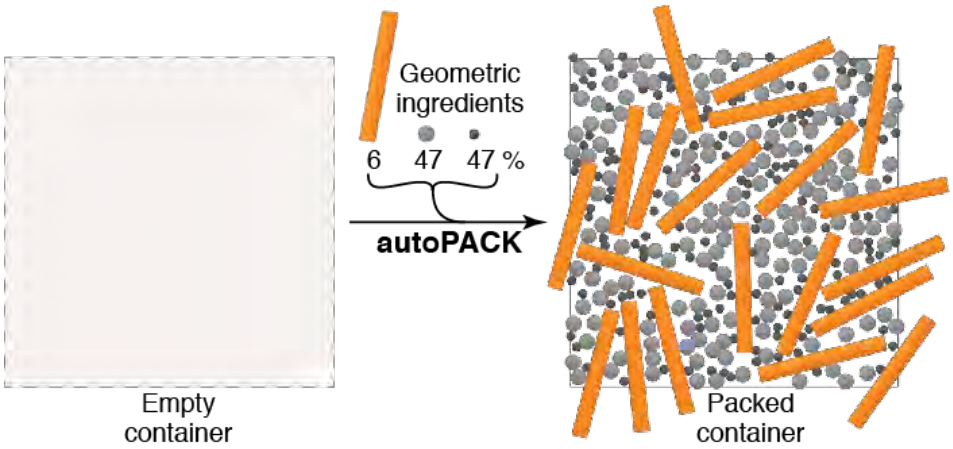


(a) (b)

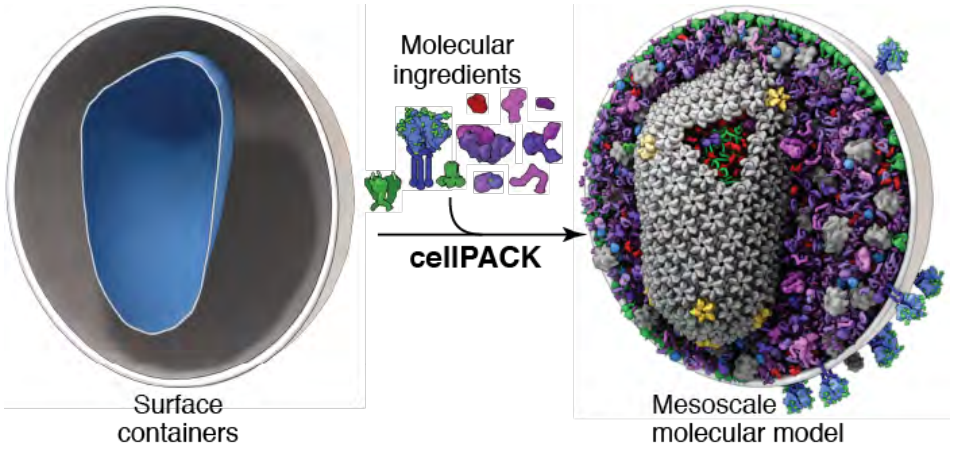
Figure 1.2: The image on the left (a) shows a fluorescence microscopy image of microtubules in a human-induced pluripotent stem cell. The right image (b) is a visualization of microtubules, where the spatial structure is generated with our fiber generation approach. It is constrained by a segmentation mask from the scan of the same cell. Image source of (a): Allen Institute for Cell Science, 3D Cell Viewer (https://www.allencell.org/3d-cell-viewer.html).

structures coil again on a higher level. Typical approaches to computationally model the structure of DNA concatenate building blocks in many iterations [JAAA+15] or use lattice-based approaches [GAO18]. In the work of this thesis, we present a novel parallel approach to generate DNA structures, adapting the midpoint displacement algorithm. The basic pipeline of our approach is also able to generate other kinds of polymers, such as microtubules, as illustrated in Figure 1.2. With the segmentation mask from fluorescence microscopy images, we generate the spatial characterization of the microtubules' structure. The third component of mesoscale models are soluble molecules. They occupy the free space in mesoscale compartments and consist of molecules with different shapes and complexities. In terms of modeling, their placement boils down to a stochastic spatial distribution, constrained by various biological rules.

A homogeneous approach of constructing full structural mesoscale models is realized by more recent approaches like cellPACK [JAAA+15] and LIFEEXPLORER [VLF11, HLLF13]. cellPACK is able to assemble large mesoscale models, such as whole viruses, utilizing packing algorithms, as illustrated in Figure 1.3. A simple example of a packing problem is to place several geometric ingredients, such as rectangles and squares, into a squared container, as shown in Figure 1.3a. In the case of biological systems, the problem



(a)



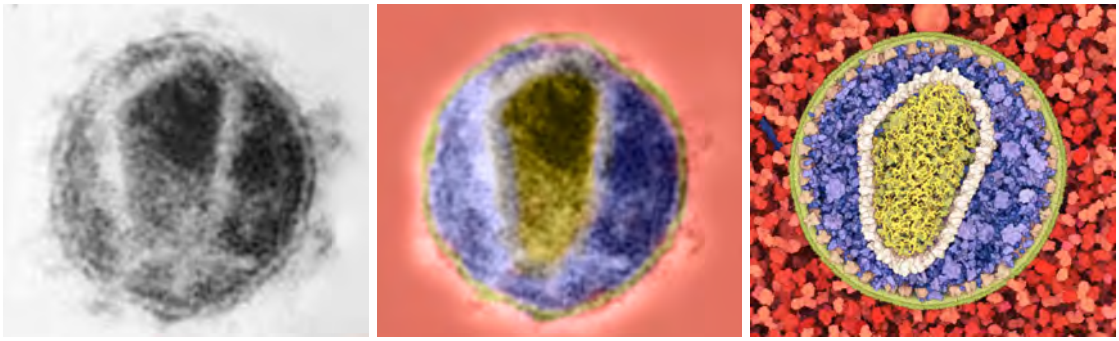
(b)

Figure 1.3: cellPACK [JAAA+15] is a computer program that assembles large-scale models at the biological mesoscale. It integrates data from structural biology and systems biology using a packing algorithm (a) to generate large-scale biological models in molecular detail. The cellPACK approach makes use of recipes, which are designed to reflect protein concentrations, distributions, and the compartment shapes of a certain structure (b). With this information, the approach can generate large-scale structural models of viruses, bacteria, and eukaryotic cells down to the level of atoms. Image source: Johnson et al. [JAAA+15].

is much more challenging. Here, the ingredients consist of differently shaped three dimensional molecular structures with specific spatial dependencies, as depicted in Figure 1.3b. cellPACK integrates multiple sources of biological information into a 'recipe' and is able to assemble structural mesoscale models with atomistic detail. The recipe describes the constitution of the desired environment, defined through the concentration of different protein types and the rules concerning their packing

and interaction. The environment of the created model is partitioned into membrane-bounded compartments, defined by mesh surfaces. Each compartment of the model is populated with different ingredients. These ingredients can be soluble molecules inside the compartment, molecules embedded in the membranes, fibrous structures, like RNA and DNA, as well as lipids in membranes. First, cellPACK defines the compartments of mesoscale structures through meshes, which can be derived from various imaging methods. Then, it packs soluble proteins into the compartments based on the concentrations acquired from experimental procedures. Furthermore, cellPACK is able to add DNA and RNA structures to the model by using a random walk approach to model the underlying spatial structure. To complete the model, cellPACK incorporates membrane structures, which can be generated from tools like lipidWrapper [DA14]. Together, the different components provide a comprehensive structure of the mesoscale model. While tools like cellPACK lead to the desired result, their main drawback is the computational effort. Depending on the complexity, the computation spans from minutes up to hours. In most biological scenarios, the exact structural outcome of the resulting model is hard to estimate and parameters must be repeatedly adapted to match the findings from scientific experiments. This makes the construction process a tedious task and the generation of large model ensembles infeasible with commodity hardware. In this thesis, we follow the recipe concept of cellPACK, but introduce novel parallel algorithms to produce the models instantaneously. Inspired by numerous approaches from procedural modeling, we have adapted existent methods and developed new approaches to instantaneously construct and visualize biological mesoscale models. Our algorithms are implemented on the GPU, facilitating a direct way to visualize and investigate all of the components in one common framework. This improves the workflow of structural biologists and also enables users to directly utilize the models for public dissemination.

Another major challenge to disseminate biological mesoscale models is their visualization. Structural biology is undergoing a transformation driven through advances in data visualization and computer graphics [MKA+19]. Tools like Chimera, COOT, PyMOL, or VMD enable structural biologists to visualize molecular structures and help them building a bridge to computer scientists. Visualization of the biological mesoscale is faced with the challenge of rendering very large numbers of molecules, thus, requires specialized solutions. Advances in large dynamic particle rendering [FGKR16] and the development of systems like MegMol [GKM+14] have made the interactive rendering of mesoscale models possible. Such advances also include methods to visualize the dynamics of molecular systems [KBE09]. While a large part of research has focused on the molecular visualization of protein structures, the visualization of nucleic acids [LBLH19] and membranes [Baa19] has shown substantial progress as well. Many visualization techniques make use of the specific geometric properties of molecular environments. For instance, molecular rendering techniques often exploit the fact that biological structures typically consist of recurring molecular substructures [LBH12] to improve memory and performance efficiency. Another significant visualization challenge is the mismatch between data size and output resolution. For



(a) (b) (c)

Figure 1.4: Illustration of the connection between an electron microscope image of the HIV virus (a) and the visualization of our computational model (c). Picture (b) shows the electron microscope image enhanced with a similar color scheme as in our visualization. Image source of (a): Hans R. Gelderblom, Robert Koch Institute.

this reason, Ibrahim et al. [IWR+17] introduce a novel concept of screen-space normal distribution functions, which represent the distribution of surface normals. This concept is especially useful for the visualization of large molecular dynamics simulations. Another well-known approach to deal with the large number of atoms, compared to the small number of pixels on a screen, is the use of structural level of detail. Such approaches [LVRH07, LMPSV14, VDZLBI11, PJR+14] visually abstract molecular structures, reduce visual clutter, and overall improve the rendering performance. The repetitive nature of the intracellular proteins simplifies hardware optimization so that even large amounts of data fits on the memory of GPUs [FKE13]. The visualization of biomolecular structures is already well reviewed, thus, we refer to the work of Kozlíková et al. [KKF+17], Johnson and Hertig [JH14], and Schatz et al. [SKPE19] for a more comprehensive overview. All the illustrations and animations presented in this thesis have been initially created with cellVIEW [MAPV15] and subsequently with the marion framework [MKS+18]. In order to address the requirements of this work, the marion framework has been constantly enhanced in terms of performance and functionality. In general, the visualizations are based on measured data, wherever available, and the comprehensive knowledge of our domain scientists. Figure 1.4 illustrates the difference between data acquired from imaging methods and the visualized outcome from our computational models. Image (a) depicts an example of a HIV virus captured by an electron microscope. The simple addition of colors in image (b) highlights the different regions of the virus and its surrounding blood plasma. In image (c), we depict our computed model with atomistic detail enhanced with a similar color scheme as in picture (b).

Structures of molecular biology carry out various tasks important for the function of a cell. While the structural information is required for many aspects of molecular models, it does not explain the physiology of living systems. Physiology is typically

showcased through computer animation. As stated, "animation is the most effective approach for exploring the dynamic nature of the mesoscale" [GFH18]. Additionally, Stith [Sti04] argues that "lectures using animation lead to more complete understanding of certain cell biology concepts than lectures that use only static illustrations". There are various ways to create molecular animations showcasing a biological phenomenon. Certainly, the most realistic way is to make use of simulations. Simulations can be obtained from complex molecular dynamics models [SB+01] or specialized agent-based systems. While molecular dynamics simulations are often used to study biophysical systems, all-atom simulations do not scale well to larger mesoscale structures or longer time spans. To circumvent this problem, biologist often make use of coarse grained models [ILU+14] or agent-based simulations.

An agent-based model represents a stochastic rule-based approach, which focuses on the interactions among the individual agents. To facilitate an efficient computation of complex biological systems, there are already first approaches to parallelize agent-based simulations [FOE+11]. A key characteristic of agent-based models is their ability to abstract. They are able to exclude non-essential detail producing a more simplified model. However, it still remains a challenging task to visualize and focus on events of interest since their occurrences are hard to determine and follow. To tackle this issue, Muzic et al. [LMPSV14] present a novel way of visualizing molecular reactions using omniscient intelligence to control passive agents. The omniscient intelligence controls the molecular reactions and directs passive agents preventing the initiation of autonomous reactions. Earlier, the work of Falk et al. [FKRE10] also uses an agent-based simulation and continuous 3D concentration map, where the local density is computed and visualized with volume rendering techniques. Another approach by Reisacher et al. [RMV16] makes use of computer graphics techniques, such as employing glowing effects and clipping planes, to guide the user's attention.

From the perspective of scientific storytelling, the major drawback of simulations is their low controllability. While animations with storytelling purposes strive to be scientifically accurate, they also aim to abstract the subject to focus on its essential meaning. Additionally, many scientific animations attempt to show processes that are not yet completely understood, or are not simulated precisely. In such cases, more traditional 3D animation tools are deployed to construct visual narratives for the dissemination of phenomena [Iwa15]. These animation tools usually originate from the entertainment industry and are extended to import scientific data [McG08]. They are capable of producing either animations or 'sandbox'-like environments, where the user can freely interact and explore. Specialized tools like SketchBio [WTHT14] try to provide simple interfaces to allow scientists to directly construct models and animations. However, the generation of a high-quality molecular animations is typically performed by trained scientific animators, a profession with a steep learning curve [Iwa14]. The creation of such stories usually starts with the story description and the definition of the audience. Animations targeted for biological experts should be truthful to the data and should be minimalistic and clear. On the other hand, research has also suggested

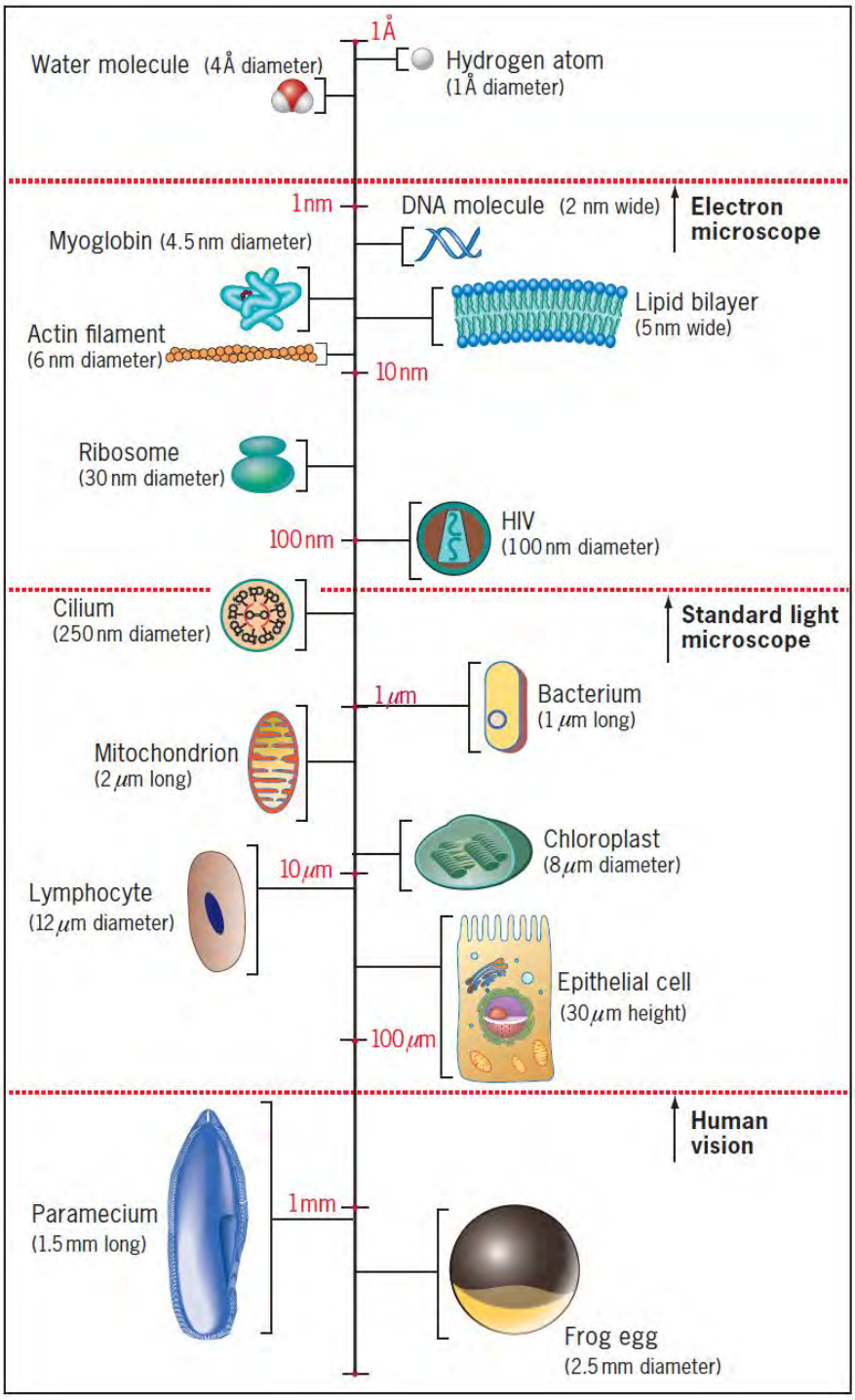


Figure 1.5: Relative sizes of cells and cell components. The illustrated structures differ more than seven orders of magnitude in size [KIW15].

that complex representations are desirable in certain learning contexts for conveying the dynamic nature of cellular phenomena [JM12]. In case a broad audience is targeted, the visual appeal, narration, and background are of greater importance. After the target audience is defined, a storyboard is drawn to outline all the actions in detail. The storyboard serves as fundamental basis for discussions between biological experts and animators. Goodsell et al. [GFH18] describe in their work how mesoscale landscapes can be used to construct visual narratives. After the story has been defined, the required molecular models are constructed and animated. New advances in molecular graphics and tools like the embedded Python Molecular Viewer (ePMV) [JAG+11] simplify this part of the pipeline. However, in most cases the construction of models and the animation are still laborious steps, where the animator uses carefully chosen keyframes. Whenever the story changes or different phenomena are explained, the animation needs to be remade. Only recent attempts formalize approaches to reuse molecular animations through templates [SMR+17].

In the research of this thesis, we focus on procedural modeling to generate visual stories about the biological mesoscale. We argue that for storytelling aspects, procedural animation is able to accurately mimic various kinds of complex mesoscale behaviors. In general, the concept of procedural animation is well-known in computer graphics and used to animate structures with a high degree of complexity. Procedural models are able to generate natural phenomena, such as forests [MKM89] and man-made structures like cities [PM01]. They scale to even large environments like whole urban landscapes [WMW+08], or infinite cities in real-time [SKK+14b, SKK+14a]. Procedural animation approaches are also able to create dynamic structures like clothing [RPC+10], ocean waves [HNC02], or clouds [SSEH03], mainly for purposes of the video-game industry. The animation makes use of mathematical functions to create visually plausible results, thus simplifies the authoring process in comparison to more precise simulations. For example, a combination of different wave functions can animate the surface of water with reasonable quality. This approach is much simpler and more efficient in comparison to a complex physics-based simulation of water. In contrast to existing animation approaches, mesoscale animations face the specific issue that they are inherently multi-scale. Figure 1.5 illustrates the relative sizes of cells and cell components, which differ more than seven orders of magnitude in size. While there are attempts to seamlessly visualize across multiple structural levels of the biological mesoscale [HMK+19], the involved processes also feature multi-scale behavior. For instance, while atoms move quickly in a chaotic Brownian motion, large structures like microtubules undergo changes on a different time scale featuring in the growth and shrinking. This makes the creation of procedural animations in the molecular domain a particular challenge.

1.4 Structure

This thesis is organized into two parts. The first part provides an overview of the research and contextualizes the publications. It delineates the individual construction

and visualization techniques that were developed in the course of this dissertation. The second part includes three peer-reviewed journal publications that constitute the main part of the thesis.

CHAPTER 2

Modeling and Visualization of Integrative Cell Biology

The grand vision of our research is the creation of in silico models of living organisms to visually explore their structure and dynamics. This includes the construction of a fully animated eukaryotic cell with a size of 50 microns in diameter and formed by more than 100 trillion atoms interacting with its environment. The research presented in this thesis creates the foundation that leads towards this goal. It puts a special focus on the biological mesoscale. In order to achieve the intended goal, the thesis proposes the integration of construction and visualization methods into a common framework so that biological structures can be constructed and explored in the same environment. This provides an intuitive feedback for the validation of biological models, serves to directly explore them, and also to disseminate them among peers or laymen. Furthermore, this thesis presents new efficient techniques to construct models for integrative cell biology, which facilitates the comprehensive understanding of biological processes as a whole. It also proposes a new multi-scale animation approach, which is able to reveal the intrinsic functions of living organisms.

We live in an opportune time for researching integrative structural biology. Several factors that support the dissemination and analysis of mesoscale structures have evolved concurrently. First, the wealth of data, driven through sources like the Protein Data Bank [BWF+00], makes structural information about biological entities publicly available. Second, the computational capabilities of parallel processors have progressed tremendously. Devices with parallel processors have become ubiquitous enabling a wide audience to visually experience the fundamental processes of life in detail. Third, the progress in visualization research of molecular structures has produced sophisticated techniques, which are capable to efficiently display and explain biological phenomena. This thesis builds on top of these advances to further progress the research

in providing a visual overview of the fundamental biological processes that make up every human being.

2.1 Research Outline

The endeavor of constructing integrative biology models does not strive to build a certain instance of a biological environment, but rather aims to construct a representative model that incorporates the currently available knowledge. As such, an integrative biological model follows and represents the underlying rules of the constructed structure instead of creating a certain image of a dataset. In the best case, the constructed model shows an integrative depiction of the structure and the behavior over multiple scales. It starts from individual atoms, to larger molecular assemblies, up to the overall shape of the generated environment. A constructed model of integrative biology facilitates to illustrate macromolecular life forms as a whole, reflecting the current structural and functional observations. Structural construction approaches range from assembling several molecules to a molecular machine, concatenating pieces of DNA to a long strand, to populate large environments like viruses or potentially even whole human cells. In order to construct a biological environment, the fundamental building elements and their characteristics have to be understood. The building blocks of mesoscale models can be roughly organized into three different kinds of structures, namely linear polymers, lipid membranes, and soluble proteins.

Linear polymers, such as DNA or RNA, are long strands that are tangled up inside certain compartments of the biological environment. For instance, DNA is an omnipresent structure in the human body and mostly located in the cell nucleus. It is fundamental for all forms all life. In the typical construction process, the generated spatial organization is represented by a curve, which is then replaced by the buildings blocks of the DNA, i.e., the DNA base pairs.

Lipid membranes feature different spatial characteristics. Here, small lipid molecules form a densely packed membrane that builds a continuous barrier around cells. The membranes of almost all organisms and many viruses are made of a lipid bilayer, which prevents ions, proteins and other molecules from diffusing into neighboring areas. A lipid membrane contains two opposed layers of lipids, where each lipid comprises several dozens of atoms. They consist of a head group and one and more tails. For each biological structure, the bilayer contains lipids of a unique type with the common property that they are densely packed and oriented roughly perpendicular to the membrane. The construction of lipid bilayers is a complex and laborious task due to the dense packing of the lipids and their large number of occurrences compared to other surrounding molecules. From the perspective of construction, the main challenge is to incorporate the known structural information and to avoid unnatural repetitive patterns on the membrane's surface.

As the third kind of structure, soluble proteins and assemblies fill the compartments of biological environments. The task of populating soluble components boils down to the

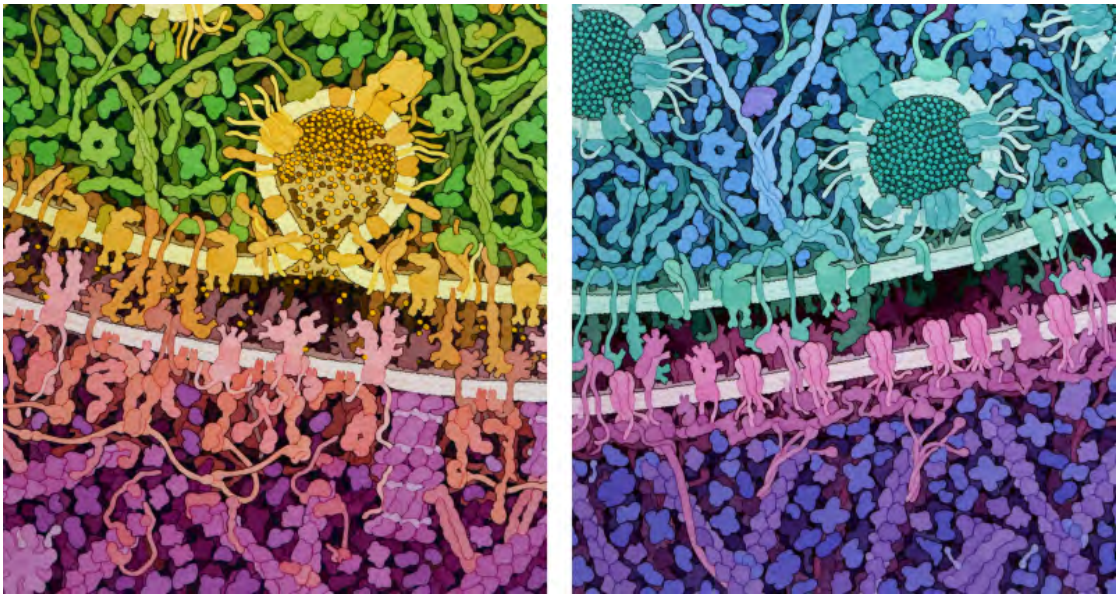


Figure 2.1: Artistic conception by David Goodsell showing an example of an excitatory synapse on the left, and a synaptic vesicle releasing a neurotransmitter into the synapse on the right. Such illustrations are based on a deep understanding of the involved biological processes, combined with the artistic talent of abstracting and realizing them in the form of watercolor images. Image courtesy of David Goodsell.

specification of their spatial distribution while avoiding overlaps with other structures. With these three basic building elements and their corresponding characteristics and constraints, most of the biological phenomena can be computationally constructed at a basic level.

So far, constructional approaches of integrative structural biology were mainly developed by trained biologists who are also skilled in computer science or illustration. For instance, well-known approaches of illustrating biological processes were made by David Goodsell. He applies his knowledge of integrative biology combined with artistic skills to create watercolor images, which abstract biological phenomena to a simplified and more understandable representation, as illustrated in Figure 2.1. The visualization community has greatly advanced visual approaches, which take structural biological models as input data. Much progress has been achieved in the effectiveness and efficiency of visualizing data of integrative structural biology. In this thesis, however, we propose a new strategy, which integrates the construction of integrative structural biology into visualization. We follow the methodological strategy "compute instead store". This approach is motivated by several reasons. First, the integration of construction and visualization simplifies the process of digitally building and validating biological phenomena. The integration of construction into visualization provides a direct feedback loop for the visual analysis and verification concerning intended

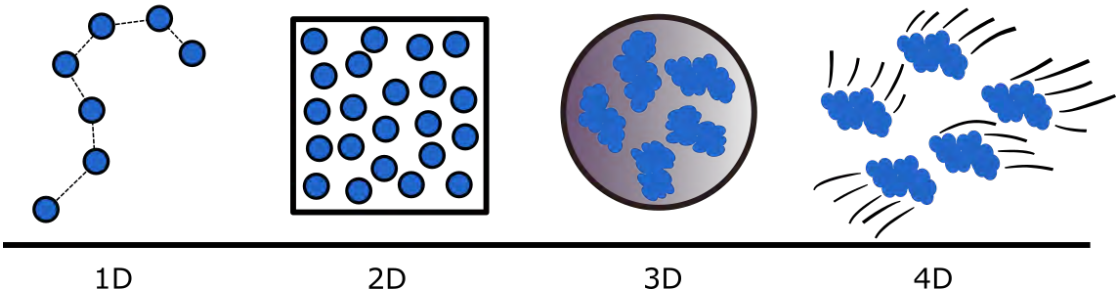


Figure 2.2: Overview of the research conducted in this thesis. The research is divided into four categories, namely the construction of linear polymers like bacterial genomes (1D), the mapping of densely packed molecular structures onto texture patches (2D), the population of molecules in biological compartments (3D), and the animation of the functionality of molecular structures (4D).

biological structures. We hypothesize that this new alternative of the visualization pipeline, improves the efficiency of constructing biological environments and also their quality. Additionally, the integration of construction and visualization provides various synergies. If biological data is directly generated on the GPU, data does not need to be transferred anymore from CPU memory to GPU memory for visualizing. This opens the door for view-driven approaches, which adaptively generate biological structures depending on the viewing frustum of the camera. Furthermore, the optimization of the construction process improves the effectiveness of presenting biological environments and also increases the potential for a qualitative analysis. An immediate approach of constructing integrative biology allows researchers to compare a multitude of instantiated biological models to observed experimental data.

In summary, the research conducted in this thesis comprises the construction of the basic structural biological mesoscale elements for visualization purposes . This includes linear polymer structures, lipid membranes, and soluble molecules. The thesis extends the concepts and techniques of animating these structures in order to disseminate them to peers or laymen. The work is divided according to the spatial characteristics of the corresponding structures. In detail, we organize the structural components into the following categories (see also Figure 2.2):

1D: Construction of linear polymer structures like DNA.

2D: Mapping of densely packed lipid structures on texture patches.

3D: Population of molecular structures into biological compartments.

4D: Animation of the dynamic behavior of biological structures.

Each part comes with certain properties, characteristics, and challenges, which are explained in the following sections.

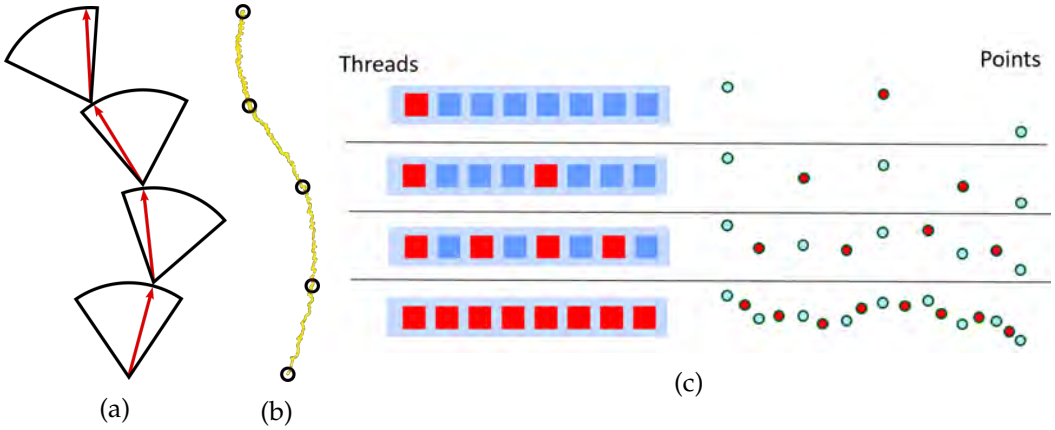


Figure 2.3: Illustration on how to construct linear polymers, such as DNA [KMA+19]. (a) shows an iterative approach to build a linear polymer, which is realized through a random walk. Each step of the random walk incorporates the stiffness of the polymer through constraining the next direction. (b) indicates the resulting polymer (DNA in this instance) depicted through a yellow curve. Our approach of constructing linear polymers makes use of a more efficient strategy, realized through a parallel implementation of the midpoint displacement algorithm. (c) illustrates the parallel construction approach, where active threads of a parallel processor (left) and the corresponding point, which are produced, (right) are highlighted in red.

2.2 Constructing Linear Polymers - a Stochastic Problem

An example of linear polymer structures are bacterial genomes, which are long fibrous macromolecules and typically modeled through a stochastic process like a random walk. A random walk starts at a given location and chooses a random, sometimes constrained, direction to find the next location. This approach is repeated until the desired length of resulting structure is reached. In the context of bacterial genomes, the constraints of the random walk induce a certain stiffness of the fiber structure. The process is illustrated in Figure 2.3. It starts with a random location inside of a given compartment. At each step, the random walk chooses a new random direction (red arrow), where the directional distribution is constrained by the stiffness of the biological model. The constrained direction of the random walk is indicated by the circular sectors, depicted in Figure 2.3a. Subsequently, the resulting points are used to define the curve of the bacterial genome. A further subdivision of the curve and replacement of points with corresponding molecular building blocks leads to a structural model of a bacterial genome, as indicated in Figure 2.3b.

While this approach leads to the desired result, the high computational effort of the method makes it an offline process. Visualization systems that intend to show multiple bacterial genome structures, or depict only the parts of a genome that is visible to the camera, benefit from an instant construction approach. Furthermore,

instant construction builds the foundation for computational hypothesis generation and verification that requires the construction of a multitude of bacterial genome models.

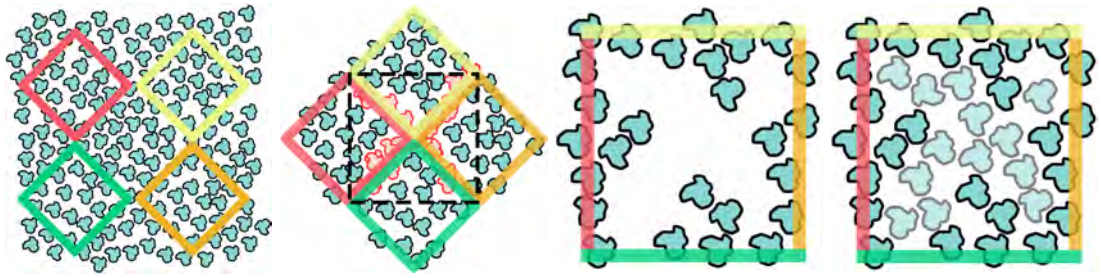
We propose a novel technique to generate bacterial genomes. It makes use of the divide-and-conquer concept and a midpoint displacement algorithm to map the process on the hardware of a parallel processor. First, a random walk generates a rough backbone of the desired structures by generating only a small number of points. The backbone is then refined with a parallel adaption of a midpoint displacement algorithm. In the simple 1D case, parallel midpoint displacement starts with two points. The midpoint of the line between the two points is displaced perpendicular to the line by a random amount. This process is repeated until the desired level of refinement is attained, while the random displacement magnitude is reduced in each step. The midpoint displacement approach became widely popular after it was extended to generate random heightmaps [FFC82], which are frequently used for terrain generation.

In contrast to an iterative random walk, the midpoint displacement algorithm maps well to the hardware of a parallel processor. In each iteration of the midpoint displacement, the procedure can be divided up and assigned to independent processes. The illustration in Figure 2.3c highlights the active threads of the parallel processor (depicted as red squares) corresponding to the newly generated points (depcited as red circles) for each iteration. The approach demonstrates that the stiffness characteristics of a linear polymer can be realized through the shift in the midpoint displacement algorithm. Our parallel midpoint displacement approach outperforms previous sequential methods in the construction of bacterial genome models like the Mycoplasma genitalium, with 580,000 DNA base pairs, more than 2600-fold.

2.3 Constructing Lipid Membranes - a Mapping Problem

In contrast to soluble proteins, lipid membranes consist of smaller molecules, which typically compose into bilayers and form the outer shape of a compartment. They are densely packed on the membrane and roughly oriented perpendicular to it. This makes modeling a laborious task. Previous tools, which model membranes by individually populating lipids one-by-one on the membrane are computationally unacceptable for interactive systems. Even more sophisticated systems like the LipidWrapper [DA14] are slow in computation. LipidWrapper uses pre-defined patches of lipids to populate a mesh, which represents a membrane. However, the edges of the populated patches do no fit together. Overlapping lipids are removed and resulting holes are filled with newly introduced lipids.

We propose a more efficient approach to populate lipid membranes, which makes use of the Wang tiling concept [Wan61]. Wang tiles have been originally proposed to create large non-perodic patterns from a small set of tiles. The concept has become popular, since it has been used for the procedural synthesis of textures. Wang tiles are a set of tiles, where each tile edge has a fixed color encoding. Wang tiles are arranged so that each edge matches the color of the adjacent neighbor. There are several approaches



(a) (b) (c) (d)

Figure 2.4: Illustration of the Wang Tile synthesis for lipid membranes [KAK+18]. (a) Four diamond-squared texture patches are extracted from a lipid membrane model. (b) A Wang Tile is assembled through the four texture patches (the Wang Tile is indicated with dashed lines). (c) Lipids outside of the Wang Tile are removed, as well as overlapping lipids at the border region of the four texture patches. (d) The final Wang Tile is created by filling holes in the tile with new lipids.

that extend the concept of Wang tiles to the 3D space, in order to texture arbitrary topological surfaces [FL05]. We presents a new concept that adapts the concept of Wang tiles to populate lipid membranes. The major challenge of the approach is to synthesize the individual tiles so that seamless transitions across tile edges of the same color are ensured. We follow the process presented by Fu and Leun [FL05] and divide the approach into four steps, as shown in Figure 2.4. In the first step (see Figure 2.4a), we use a piece of an existing lipid membrane model to extract diamond-squared patches, where each patch is encoded with a different color. In the second step, we use the patches to build a Wang tile (see Figure 2.4b). The colored edges of the Wang tile correspond to the color of the corresponding diamond patches. The excess parts of the patches are cut off and the remaining lipids form the Wang tile. Using this concept, tiles with the same color will always fit together. Nevertheless, the creation of a Wang tile will potentially result in overlapping lipids at the connecting edges of the diamond patches, the diagonals of the Wang tile (see Figure 2.4b). To resolve the overlaps, we follow the approach of the LipidWrapper system. In step 3, overlapping lipids are removed (see Figure 2.4c) and replaced through non-overlapping lipids in step 4 (see Figure 2.4d). This concept facilitates the efficient construction of lipid membranes on arbitrarily shaped compartments.

2.4 Constructing Soluble Proteins - a Packing Problem

Cells and viruses are typically crowded with soluble components, which fill their compartments. They are enclosed in shapes, which are usually lipid membranes and appear in large quantities. The population process of soluble proteins boils down to spatially distributing the components inside of the compartments. Previous work [JAAA+15] has already achieved this task in a sequential manner, which can

take up to hours to construct a model of a virus. They describe the task as a packing problem, which is a challenging task in the context of biological systems. Here, the ingredients consist of differently shaped 3D structures, namely molecules, and have specific spatial dependencies.

We propose a parallel approach to pack soluble components into shapes reducing the computational effort and allowing for interactive rates. To make the procedure applicable to a parallel processor like the GPU, we divide the procedure into three steps:

1. The compartment space is partitioned using voxelization.
2. The space is populated with the soluble proteins, while first ignoring overlaps.
3. The remaining overlapping proteins are detected and relaxed.

This algorithm staging leads to the major advantage that each step can be processed in parallel. There are efficient approaches for GPU-based voxelization, such as the one by Schwarz and Seidel [SS10]. If the shape of the compartment is static, this step can be even computed in an offline process. The voxelization of the compartments provides a valuable structure to determine which locations belong to a specific one. Additionally, it is used in the packing process to determine which locations are already occupied by other structures. In the second step, proteins are populated into a compartment. In a typical packing approach, this process has strong sequential constraints. A populated protein must not overlap with any other protein that is already generated. To make the population applicable for parallel processing, we first ignore any potential overlaps. Overlaps that are generated during the population step are relaxed subsequently with a force-based system. The relaxation is loosely based on standard rigid body dynamics, where corresponding forces are computed if points are in contact. However, the performance of the relaxation system is dependent on the degree and number of existing overlaps. In order to reduce the workload of the system, the population step should result in well-spaced and evenly distributed proteins. Standard random number generators usually do not result in well-spaced points. For this reason, we populate compartments with the help of Halton sequences [Hal60]. A Halton sequence produces well-spaced points and are often used for numeric methods, such as Monte Carlo simulations [DSHG08]. This three-step approach maps well to the GPU and is able to instantaneously construct the soluble structures of large biological environments.

2.5 Procedural Animation of Molecular Structures - a Multi-scale Problem

While the previous techniques provide an efficient way of constructing structural mesoscale models, they do not reveal the physiological nature of the biological systems.

We present a novel approach to procedurally animate dynamic behavior in a multi-scale fashion. The technique presented in the paper facilitates to communicate the mechanisms and functions of a biological system to peers and laymen. So far, molecular dynamics are typically animated through keyframes, where each individual frame has to be carefully chosen and designed. Whenever new findings about a system are discovered, the animation needs to be updated to reflect the new state of knowledge. Another approach, used by scientists, is molecular dynamics simulation. A simplified molecular system is given a set of initial conditions and a Newtonian physics simulation is employed in the modeling of the system's behavior. Such a system is hard to control beyond the specification of the initial conditions. This motivates the use of procedural animation instead of using standard keyframed animations, or simulations. A procedural animation does not require a thorough specification of animation details and at the same time has a predictable outcome. The concept of procedural animation is well-known in the computer graphics community. It captures the behavior of a system in order to generate a variety of animations from the same system. Procedural animations typically incorporate the motion behavior of characters, fluids, or flexible objects like clothes. Most procedural animations deal with behavior on only one scale. Biological systems, however, show a vastly varying behavior depending on the observed scale. This makes the creation of procedural animations a complex task.

There are various approaches to create animations for molecular structures. For the perspective of animators, simulation is the most simple one. Simulations can be extremely complex and are often not feasible to showcase the functions of certain structures. Additionally, some molecular environments have too many unknown characteristics to apply simulation. For these reasons, keyframing is typically the tool of choice. Keyframing refers to the creation of snapshots of the system's state at different moments in time. An example of keyframing is shown in Figure 2.5a. The initial keyframe (left) consists of randomly scattered molecules. The positions of the molecules interpolate (middle) to the target keyframe (right) resulting in a sphere. While the animation results in the desired organization of the molecules, it might not correctly reflect the assembly process of the structure. We introduce the concepts of *instant-dependent timing* (IDT), *instance-dependent transition function* (IDTF), and *time-varying keyframes* (TVK) to create realistic looking animations of biological processes. IDT modifies the interpolation timings based on the individual instances, as shown in Figure 2.5b. Using IDT, all molecules of the example shown in Figure 2.5b now arrive at their destination in the correct order. The animation is staged and shows how the sphere is assembled. While using IDT the timings are correct, but the trajectory of the molecules might still be wrong. Therefore, we propose a IDTF, which modifies the individual trajectories of the molecules, as depicted in Figure 2.5c. IDT and IDTF produce animated transitions between two static states, which can be described with two simple keyframes. TVK further enhance the concept by defining the interpolation between two dynamic states. For instance, this enables a simple animation description that incorporates the Brownion motion of the free-floating molecules and the spinning motion of the molecules that are assembled in the sphere, as illustrated in Figure 2.5d.

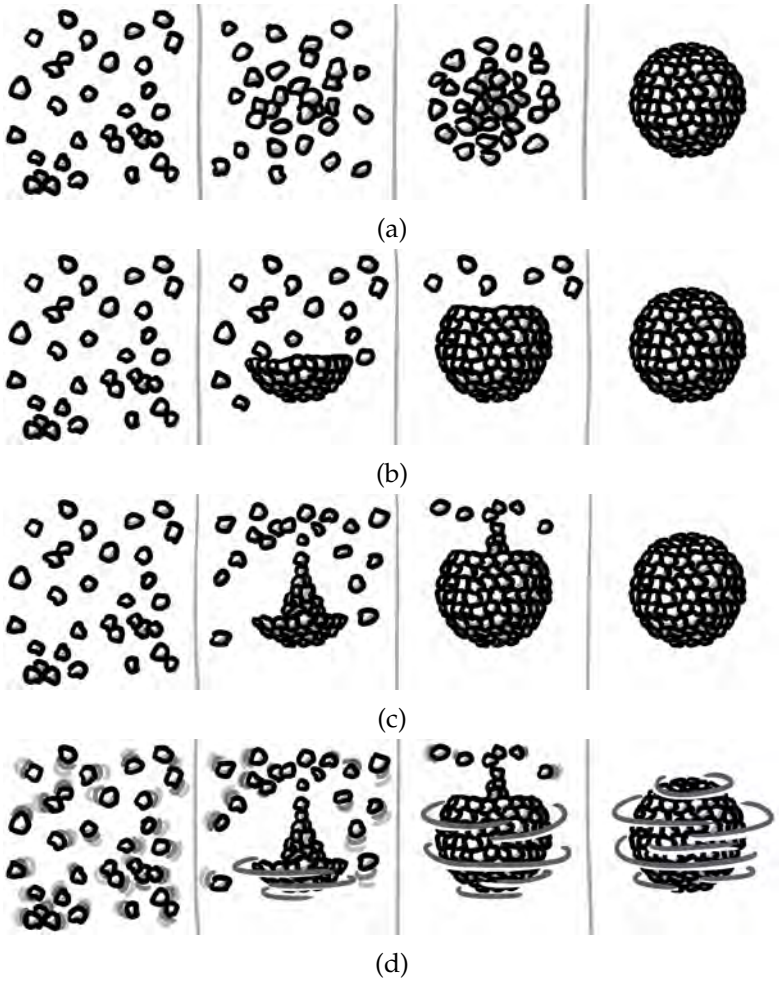


Figure 2.5: Four different animation approaches (a,b,c,d) illustrated with four time stamps [KVGMar]. (a) A keyframed animation, where the initial state (static randomly positioned molecules) is interpolated to the target state (molecules forming a sphere). (b) The same animation is enhanced through *instance-dependent timing*, which modifies the interpolation timings based on the individual instances. (c) An *instance-dependent transition function* transforms the instance paths. This can be used to create a specific building process of the model. (d) The same situation as in (c), but here both initial and target state are dynamic.

The use of IDT, IDTF, and TVK simplifies the creation of keyframe animations in multi-scale, instance-based environments.

We have created a procedural animation that shows the process of the dynamic instability of microtubules. Microtubules are fibrous structures and one of the building blocks of cells. They support cells in maintaining their shapes. To create procedural microtubule models, we propose a pipeline with several components, which is illustrated

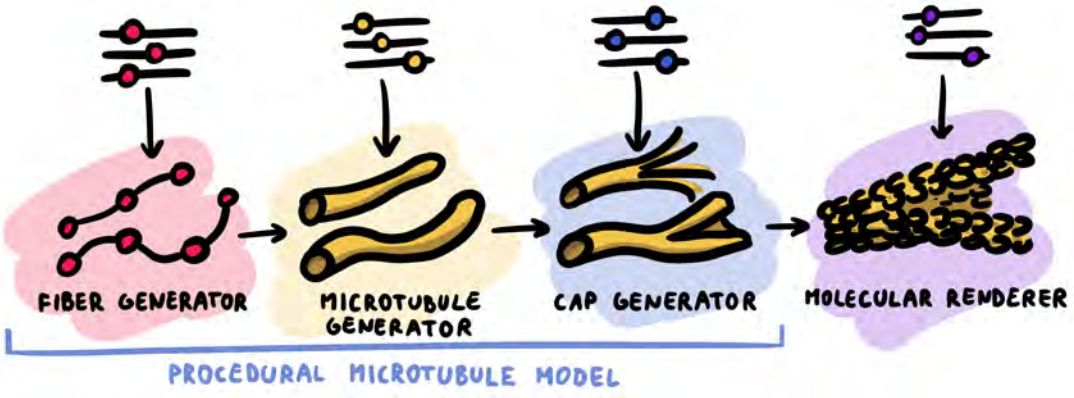


Figure 2.6: Pipeline for the procedural microtubule model [KVGMar]. First, the fiber generator produces a discrete curve that represents the spatial orientation of the microtubule. Subsequently, the microtubule generator assembles the structure of the microtubule. The cap generator displaces certain molecules of the microtubule so that a cap is modeled. Finally, the model is visualized by a molecular renderer.

in Figure 2.6. The first component is the fiber generator and produces a cubic spline that reflects the spatial arrangement of the microtubules. In the second component, the structure of the microtubules is formed through placing molecules along each fiber. The cap, which plays an important part in the dynamic instability, is generated in the third component. Finally, the molecular renderer visualizes the model and plays the procedural animation. With this system, we were able to evaluate the effectiveness of our approach through informal feedback sessions with several domain experts.

2.6 Impact - Applications, Animations, and Illustrations

While the main focus of this thesis is to advance research concerning the construction and visualization of integrative biology, it has also impacted the communities of visualization and structural biology through several applications, animations, and illustrations. Examples include thereof the demonstration of the multi-scale procedural animation approach to showcase the dynamic instability of microtubles (shown in Figure 2.7). The animation and illustration results of the approach have won the PacificVis 2019 Storytelling Contest (`https://vimeo.com/330528972`), as well as the jury award of the image contest at the VCBM 2019 conference (`https://conferences.eg.org/vcbm2019/awards/`). In order to advance the exploration of crowded biological environments, we have extended the visualization of mesoscale models to immersive virtual environments. A snapshot from the generated visualizations is depicted in Figure 2.8. Additionally, we have showcased the individual rendering steps of a virtual cell in a storytelling project, which proceeded to the finals of the Pacific Vis 2017 Storytelling Contest (shown in Figure 2.9). Furthermore, the research of this thesis has built the foundation of a company called Nanographics

Figure 2.7: Procedural animation of the dynamic instability of microtubules [KVGMar].

GmbH, which currently works on the creation of 3D stereoscopic movies about molecular processes. The movies are supposed to be shown in several 360° cinedome theaters in Sweden.

2.7 Conclusion and Future Work

The work of this thesis has contributed to many different areas of science. First, our visualization techniques serve as a catalyst to natural sciences. Watson and Crick have discovered the structure of DNA after experimenting with a physical model. Today, visualization provides a more efficient approach to help hypothesis formation and idea generation. Our instant construction approach has already been incorporated into a web-based tool called Mesoscope (`https://mesoscope.scripps.edu/`). With Mesoscope, the community of structural biologists can create and share structural models of complex molecular environments to facilitate new analyses and simulations thereof. Also tools like cellPAINT [GAB+18] and cellPAINTVR (`http://cellpaint.scripps.edu/`) make partial use of our techniques to create and paint mesoscale models as desktop illustrations or in virtual environments. Collaborating domain scientists plan to apply the approach of instantaneously constructing bacterial genomes to recapitulate the results from experimental data with the synthetic results from our approach.

The contributions of this thesis go beyond the involved application domain. We believe that our techniques already have influenced and will in the future further

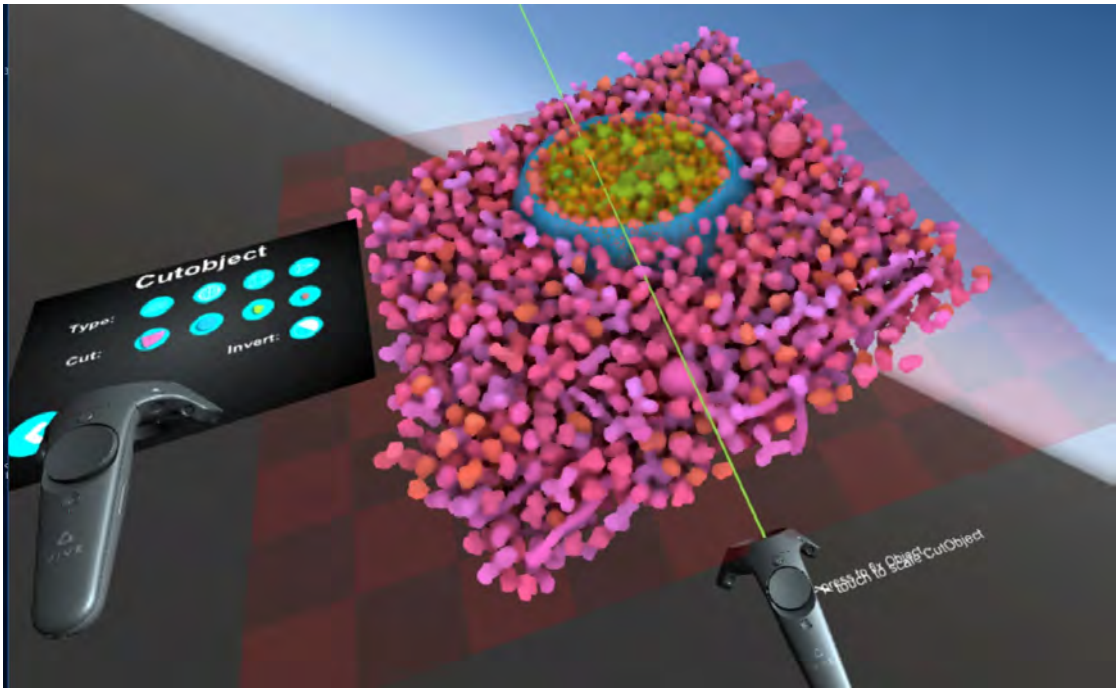


Figure 2.8: Snapshot of our virtual reality application for the interactive and immersive exploration of crowded biological structures.

advance the scientific state of visualizing biological environments and other areas with related characteristics. The task of animating environments with multi-scale behavior is omnipresent in visualization. Many natural phenomena, such as water, weather, climate, light, space, etc. feature multi-scale behavior, where dissemination could benefit from the presented visualization techniques.

Even though this work contributes to the field of biological construction and visualization, there are still many limitations that complicate the dissemination of large-scale and virtually endless worlds of biological environments. We have presented techniques that generate structural models and animations of their behavior instantaneously. The question is still open to generate only those substructures that are currently visible to the observer. The future will show how construction techniques can be incorporated into view-dependent approaches that build and depict biological structures. Only when these tasks are accomplished, endless and detailed worlds of living systems can be visualized. Furthermore, our approaches focus on the mesoscale, which limits the visualization and modeling techniques to atoms, molecules, and environments like viruses and bacteria. New ideas and advances have to be developed to incorporate data that go beyond these scales.

In future work, we plan to enhance the presented techniques in several ways. First, the incorporation of fast and intuitive methods to model interactions between the

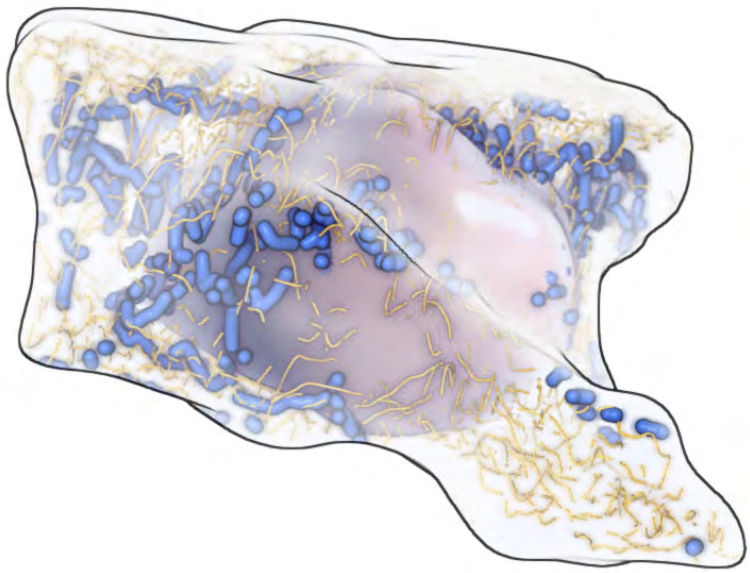


Figure 2.9: Interactive visualization of the birth of a virtual cell [MKS+18].

ingredients of the environments need to be researched. For instance, modeling the interactions of ribosomes with messenger RNA to form a polysome, or the crosslinking of DNA by DNA-binding proteins. State-of-the-art particle simulation methods might provide efficient solution for these challenges. Another research direction is the exploration of formulations that describe the behaviour of molecular machines. Behavioral animation approaches that use agent-based systems might enhance our multi-scale procedural animation technique. With increasing accuracy and number of features of our approaches, it will be possible to apply the techniques not only to the dissemination of knowledge but also in the scientific analysis process. A further research direction is the development of level-of-detail concepts for the construction approach. In case a complex structure is barely visible to the camera, it might not be necessary to model the structure in full detail. This concept also goes hand in hand with new level-of-detail visualization approaches. The question concerns how a structure can be visually compressed if it occupies only a small space on the screen. Furthermore, together with our domain scientists, we plan to use the parallel genome construction approach to compare a multitude of generated genomes with experimental data from Hi-C, a method to study the three-dimensional architecture of genomes.

CHAPTER 3

Contributions and Authorship Statement

In this chapter, we state the contributions of this thesis and provide a statement of the authorship.

3.1 Scope and Contributions

The central vision of this thesis is to develop new techniques and methods that help to disseminate the biological mesoscale, as well as to support the process of hypothesis generation. The scope of this endeavor focuses on biological systems that range from atoms to viruses, bacteria, and eukaryotic cells. The work presents techniques within the scope of computational sciences, computer graphics, and construction and visualization in the context of multi-scale, multi-instance environments.

This cumulative thesis is based on the following publications:

A **Tobias Klein**, Peter Mindek, Ludovic Autin, David Goodsell, Arthur Olson, Eduard Gröller, Ivan Viola: *Parallel Generation and Visualization of Bacterial Genome Structures*. Computer Graphics Forum (Proceedings of Pacific Graphics 2019), 38: 57-68, 2019. doi:10.1111/cgf.13816
[Best Student Presentation Award]

B **Tobias Klein**, Ludovic Autin, Barbora Kozlíková, David Goodsell, Arthur Olson, Eduard Gröller, Ivan Viola: *Instant Construction and Visualization of Crowded Biological Environments*. IEEE Transactions on Visualization and Computer Graphics (Proceedings of IEEE VIS 2017), 24(1):862-872, 2017. doi:10.1109/TVCG.2017.2744258
[SciVis Best Paper Honorable Mention]

C **Tobias Klein**, Ivan Viola, Eduard Gröller, Peter Mindek: *Multi-Scale Procedural Animations of Microtubule Dynamics Based on Measured Data*. IEEE Transactions on Visualization and Computer Graphics (Proceedings of IEEE VIS 2019), 2019. doi: 10.1109/TVCG.2019.2934612 (to appear)

The following articles are also published in the scope of the thesis:

I Johannes Sorger, Peter Mindek, **Tobias Klein**, Graham Johnson, Ivan Viola: *Illustrative Transitions in Molecular Visualization via Forward and Inverse Abstraction Transform*. In Eurographics Workshop on Visual Computing for Biology and Medicine (Proceedings of VCBM 2016), 21-30, 2016. doi: 10.2312/vcbm.20161267

II Thomas Bernhard Koch, David Kouřil, **Tobias Klein**, Peter Mindek, Ivan Viola: *Semantic Screen-Space Occlusion for Multiscale Molecular Visualization*. In Eurographics Workshop on Visual Computing for Biology and Medicine (Proceedings of VCBM 2018), 197-201, 2018. doi: 10.2312/vcbm.20181245

III Haichao Miao, **Tobias Klein**, David Kouřil, Peter Mindek, Karsten Schatz, Eduard Gröller, Barbora Kozlíková, Tobias Isenberg, Ivan Viola: *Multiscale Molecular Visualization*. In Journal of Molecular Biology, 431(6):1049-1070, 2019. doi: 10.1016/j.jmb.2018.09.004

The scientific contributions of **Paper A** include:

- A problem characterization of generating large-scale structures of bacterial genomes from the perspective of the visualization community.
- Novel techniques that facilitate the parallel generation of large-scale genomes and related linear polymers.
- The integration of a parallel construction process into a visual environment that allows users to quickly experiment with different DNA structures to perform visual hypothesis generation.
- The foundation for computational hypothesis generation and verification that requires the generation of a multitude of bacterial genome models.

The scientific contributions of **Paper B** include:

- A novel set of algorithms to populate soluble proteins into compartments and to generate membrane structures.
- The integration of construction into visualization in a common environment where different parameter settings can be interactively explored.

- Enhancing the construction process through a visual environment that instantaneously generates different instances of the same biological model.

The scientific contributions of **Paper C** include:

- A procedural model to visualize complex characteristics of biological processes with distinct behavior on different scales.
- Designing a procedural animation of the dynamic instability of microtubules.
- The integration of the procedural animation in a real-time molecular visualization system.

Within the duration of this thesis, the feedback to our contributions already indicate a noticeable impact on the community of visualization and structural biology. The instant construction techniques are actively used by our domain scientists for their tools and research. They have incorporated instant construction techniques into a web-based structural construction tool called Mesoscope (`https://mesoscope.scripps.edu/`). The approaches to instantly generate genome structures has build the basis for future research, where several instantly constructed genome models are directly compared to experimental data. With the help of the presented construction and animation approaches, our research team has already won several storytelling and image contests. This includes, the Pacific Vis 2017 Storytelling Contest (Finalist), Pacific Vis 2019 Storytelling Contest (Winner), VCBM Image Contest Award (Jury Award). Additionally, the research has also shown commercial potential. The results of this thesis have contributed to the foundation of a company, called Nanographics GmbH [KMV+], which creates interactive molecular animations. Nanographics is currently contracted to create an interactive rendering platform for a next-generation science show depicting nano-to-micro scale ranges.

3.2 Authorship Statement

The manuscripts constituting this thesis were published in the course of the doctoral studies. The ideas, implementations, and publications mainly originate from the author of this thesis, but were developed in collaboration with multiple co-authors. The core concepts of the three main publications derive from a multitude of discussions with the two supervisors of this thesis, namely Ivan Viola and Eduard Gröller. They have supported the progress of the thesis, encouraged innovative approaches, and also assisted in the writing process with valuable feedback and reviewing. The required knowledge in structural and computational integrative biology was provided mainly by the researches from the Scripps Research Institute, namely David Goodsell, Arthur Olson, and Ludovic Autin. They have laid the domain-specific foundation for the success of this work. Additionally, the research was partially supported and evaluated

by domain scientists from the Allen Institute of Cell Science. In the following, we list a more detailed authorship statement concerning the individual publications:

- **Paper A**: The high-level research idea of this publication was developed by the author together with the help of the two supervisors. The supervisors have also contributed through ongoing feedback sessions. While Eduard Gröller pointed out the applicability of the Midpoint Displacement algorithm, the adaption for the bacterial genome construction and parallel realization were developed by the author. Peter Mindek has contributed to the design of the of the parallel implementation of constructing bacterial genomes through numerous discussion. David Goodsell, Arthur Olson, and Ludovic Autin have evaluated and discussed our results in their role as domain experts.

- **Paper B**: The high-level research idea was mainly sketched by the author together with the two supervisors. The progression of the research was driven by the author, as well as the detailed realization of the research idea. The domain scientists David Goodsell and Arthur Olson have contributed with feedback related to structural biology and provided domain data. The implementation of the instant construction approach was primarily implemented by the author, but especially in domain aspects supported by Ludovic Autin. He has been a developer of cellPACK [JAAA+15] and possesses a detailed knowledge about the biological constraints and properties of the constructed structures. Barbora Kozlíková has contributed to the paper with her comprehensive knowledge on the analysis and visualization of biomolecules. The publication was generally written by the author, yet Barbora Kozlíková and the supervisors have supported the writing process in high-level reviewing and indication of related visualization research. The domain scientist David Goodsell also contributed to the writing process of the related work and the evaluation of the generated models.

- **Paper C**: This paper builds up on the foundations that were laid by **Paper A and B**. The high-level idea was developed by the author together with Peter Mindek. The author has progressed the research of the previous work and extended the instant construction process with a multi-scale animation approach. The structural building process of the spatial orientation of the microtubule models was developed by the by the author of this thesis. Overall, the work was co-implemented with Peter Mindek, who also contributed to the writing of the description of the animation approach. The results were evaluated by the author with the feedback of several experts in the area of structural biology, computational biology, as well as professional illustrators. The supervisors have contributed through idea discussions and reviews of the writing.

Bibliography

[Baa19] Marc Baaden. Visualizing biological membrane organization and dynamics. *Journal of molecular biology*, 2019.

[BWF+00] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, and Philip E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

[DA14] Jacob D. Durrant and Rommie E. Amaro. LipidWrapper: an algorithm for generating large-scale membrane models of arbitrary geometry. *PLOS Computational Biology*, 10(7):e1003720, 2014.

[DSHG08] Ishaan L. Dalal, Deian Stefan, and Jared Harwayne-Gidansky. Low discrepancy sequences for Monte Carlo simulations on reconfigurable platforms. In *2008 International Conference on Application-Specific Systems, Architectures and Processors*, pages 108–113. IEEE, 2008.

[DSV+19] William Dearnaley, Beatrice Schleupner, A Cameron Varano, Nick Alden, Floricel Gonzalez, Michael Casasanta, Birgit E Scharf, Madeline J Dukes, and Deb F Kelly. Liquid-cell electron tomography of biological systems. *Nano Letters*, 2019.

[EM03] David S. Ebert and F. Kenton Musgrave. *Texturing & modeling: a procedural approach*. Morgan Kaufmann, 2003.

[FFC82] Alain Fournier, Don Fussell, and Loren Carpenter. Computer rendering of stochastic models. *Commun. ACM*, 25(6):371–384, 1982.

[FGKR16] Martin Falk, Sebastian Grottel, Michael Krone, and Guido Reina. Interactive GPU-based visualization of large dynamic particle data. *Synthesis Lectures on Visualization*, 4(3):1–121, 2016.

[FKE13] Martin Falk, Michael Krone, and Thomas Ertl. Atomistic visualization of mesoscopic whole-cell simulations using ray-casted instancing. *Computer Graphics Forum*, 32:195–206, 2013.

[FKRE10] Martin Falk, Michael Klann, Matthias Reuss, and Thomas Ertl. 3D visualization of concentrations from stochastic agent-based signal transduction simulations. In *2010 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 1301–1304. IEEE, 2010.

[FL05] Chi-Wing Fu and Man-Kang Leung. Texture tiling on arbitrary topological surfaces using Wang tiles. In *Rendering Techniques*, pages 99–104, 2005.

[FOE+11] Martin Falk, Michael Ott, Thomas Ertl, Michael Klann, and Heinz Koeppl. Parallelized agent-based simulation on cpu and graphics hardware for spatial and stochastic models in biology. In *Proceedings of the 9th International Conference on Computational Methods in Systems Biology*, pages 73–82. ACM, 2011.

[GAB+18] Adam Gardner, Ludovic Autin, Brett Barbaro, Arthur J. Olson, and David S. Goodsell. Cellpaint: Interactive illustration of dynamic mesoscale cellular environments. *IEEE Computer Graphics and Applications*, 38(6):51–66, 2018.

[GAO18] David S. Goodsell, Ludovic Autin, and Arthur J. Olson. Lattice models of bacterial nucleoids. *The Journal of Physical Chemistry B*, 122(21):5441–5447, 2018.

[GFH18] David S. Goodsell, Margaret A. Franzen, and Tim Herman. From atoms to cells: Using mesoscale landscapes to construct visual narratives. *Journal of Molecular Biology*, 430(21):3954 – 3968, 2018.

[GKM+14] Sebastian Grottel, Michael Krone, Christoph Müller, Guido Reina, and Thomas Ertl. Megamol—a prototyping framework for particle-based visualization. *IEEE transactions on visualization and computer graphics*, 21(2):201–214, 2014.

[Goo91] David S. Goodsell. Inside a living cell. *Trends in Biochemical Sciences*, 16:203–206, 1991.

[Hal60] John H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2(1):84–90, 1960.

[HLLF13] Samuel Hornus, Bruno Lévy, Damien Lariviére, and Eric Fourmentin. Easy DNA modeling and more with GraphiteLifeExplorer. *PLoS ONE*, 8:1–12, 2013.

[HMK+19] Sarkis Halladjian, Haichao Miao, David Kouřil, M Eduard Gröller, Ivan Viola, and Tobias Isenberg. Scaletrotter: Illustrative visual travels across negative scales. *arXiv preprint arXiv:1907.12352*, 2019.

[HNC02] Damien Hinsinger, Fabrice Neyret, and Marie-Paule Cani. Interactive animation of ocean waves. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, pages 161–166. ACM, 2002.

[ILO+16] Wonpil Im, Jie Liang, Arthur Olson, Huan-Xiang Zhou, Sandor Vajda, and Ilya A Vakser. Challenges in structural approaches to cell modeling. *Journal of molecular biology*, 428(15):2943–2964, 2016.

[ILU+14] Helgi I. Ingólfsson, Cesar A. Lopez, Jaakko J. Uusitalo, Djurre H. de Jong, Srinivasa M. Gopal, Xavier Periole, and Siewert J. Marrink. The power of coarse graining in biomolecular simulations. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 4(3):225–248, 2014.

[IMAW15] Martin Ilčík, Przemyslaw Musialski, Thomas Auzinger, and Michael Wimmer. Layer-based procedural design of facades. *Computer Graphics Forum*, 34(2):205–216, 2015.

[Iwa14] Janet Iwasa. Crafting a career in molecular animation. *Molecular biology of the cell*, 25(19):2891–2893, 2014.

[Iwa15] Janet H Iwasa. Bringing macromolecular machinery to life using 3D animation. *Current opinion in structural biology*, 31:84–88, 2015.

[IWR+17] Mohamed Ibrahim, Patrick Wickenhäuser, Peter Rautek, Guido Reina, and Markus Hadwiger. Screen-space normal distribution function caching for consistent multi-resolution rendering of large particle data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):944–953, 2017.

[JAAA+15] Graham T. Johnson, Ludovic Autin, Mostafa Al-Alusi, David S. Goodsell, Michel F. Sanner, and Arthur J. Olson. cellPACK: a virtual mesoscope to model and visualize structural systems biology. *Nature methods*, 12(1):85–91, 2015.

[JAG+11] Graham Johnson, Ludovic Autin, David Goodsell, Michel Sanner, and Arthur Olson. ePMV embeds molecular modeling into professional animation software environments. *Structure (London, England : 1993)*, 19:293–303, 2011.

[JGA+14] Graham T. Johnson, David S. Goodsell, Ludovic Autin, Stefano Forli, Michel F. Sanner, and Arthur J. Olson. 3D molecular models of whole HIV-1 virions generated with cellpack. *Faraday discussions*, 169:23–44, 2014.

[JH14] Graham T Johnson and Samuel Hertig. A guide to the visual analysis and communication of biomolecular structural data. *Nature Reviews Molecular Cell Biology*, 15(10):690–698, 2014.

[JLKI09] Sunhwan Jo, Joseph B. Lim, Jeffery B. Klauda, and Wonpil Im. CHARMM-GUI membrane builder for mixed bilayers and its application to yeast membranes. *Biophysical Journal*, 97(1):50–58, 2009.

[JM12] Jodie Jenkinson and Gaël Mcgill. Visualizing protein interactions and dynamics: Evolving a visual language for molecular animation. *CBE life sciences education*, 11:103–10, 2012.

[KAK+18] Tobias Klein, Ludovic Autin, Barbora Kozlíková, David. S. Goodsell, Arthur Olson, M. Eduard Gröller, and Ivan Viola. Instant construction and visualization of crowded biological environments. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):862–872, 2018.

[KBE09] Michael Krone, Katrin Bidmon, and Thomas Ertl. Interactive visualization of molecular surface dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 15, 2009.

[KČK+18] David Kouřil, Ladislav Čmolík, Barbora Kozlíková, Hsian-Yun Wu, Graham Johnson, David S. Goodsell, Arthur Olson, Eduard Gröller, and Ivan Viola. Labels on levels: labeling of multi-scale multi-instance and crowded 3D biological environments. *IEEE transactions on visualization and computer graphics*, 25(1):977–986, 2018.

[KIW15] Gerald Karp, Janet Iwasa, and Marshall Wallace. *Karp's Cell and Molecular Biology: Concepts and Experiments, 8th Edition*. John Wiley & Sons, 2015.

[KKF+17] Barbora Kozlíková, Michael Krone, Martin Falk, Norbert Lindow, Marc Baaden, Daniel Baum, Ivan Viola, Julius Parulek, and H-C Hege. Visualization of biomolecular structures: State of the art revisited. In *Computer Graphics Forum*, volume 36, pages 178–204. Wiley Online Library, 2017.

[KMA+19] Tobias Klein, Peter Mindek, Ludovic Autin, David Goodsell, Arthur Olson, Eduard Gröller, and Ivan Viola. Parallel generation and visualization of bacterial genome structures. In *Computer Graphics Forum*, volume 38, pages 57–68. Wiley Online Library, 2019.

[KMV+] Tobias Klein, Peter Mindek, Ivan Viola, Eduard Gröller, Barbora Kozlíková, and Werner Purgathofer. Nanographics GmbH. `http://www.nanographics.at`.

[KVGMar] Tobias Klein, Ivan Viola, Eduard Gröller, and Peter Mindek. Multi-scale procedural animations of microtubule dynamics based on measured data. *IEEE Transactions on Visualization and Computer Graphics*, 2019 (to appear).

[LBH12] Norbert Lindow, Daniel Baum, and H-C Hege. Interactive rendering of materials and biological structures on atomic and nanoscopic scale. In *Computer Graphics Forum*, volume 31, pages 1325–1334. Wiley Online Library, 2012.

[LBLH19] Norbert Lindow, Daniel Baum, Morgan Leborgne, and Hans-Christian Hege. Interactive visualization of rna and dna structures. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):967 – 976, 2019.

[LMMS+16] Mathieu Le Muzic, Peter Mindek, Johannes Sorger, Ludovic Autin, David S. Goodsell, and Ivan Viola. Visibility equalizer cutaway visualization of mesoscopic biological models. In *Computer Graphics Forum*, volume 35, pages 161–170. Wiley Online Library, 2016.

[LMPSV14] Mathieu Le Muzic, Julius Parulek, Anne-Kristin Stavrum, and Ivan Viola. Illustrative visualization of molecular reactions using omniscient intelligence and passive agents. *Computer Graphics Forum*, 33, 2014.

[LSM+19] Joanna Leng, Massa Shoura, Tom CB McLeish, Alan N Real, Mariann Hardey, James McCafferty, Neil A Ranson, and Sarah A Harris. Securing the future of research computing in the biosciences. *PLoS computational biology*, 15(5):e1006958, 2019.

[LVRH07] Ove Lampe, Ivan Viola, Nathalie Reuter, and Helwig Hauser. Two-level approach to efficient visualization of protein dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 13:1616–23, 2007.

[MAPV15] Mathieu Le Muzic, Ludovic Autin, Julius Parulek, and Ivan Viola. cellVIEW: a tool for illustrative and multi-scale rendering of large biomolecular datasets. In *Eurographics Workshop on Visual Computing for Biology and Medicine*, pages 61–70. EG Digital Library, The Eurographics Association, 2015.

[McG08] Gaël McGill. Molecular movies… coming to a lecture near you. *Cell*, 133(7):1127 – 1132, 2008.

[MKA+19] Xavier Martinez, Michael Krone, Naif Alharbi, Alexander S. Rose, Robert S. Laramee, Sean O'Donoghue, Marc Baaden, and Matthieu Chavent. Molecular graphics: Bridging structural biologists and computer scientists. *Structure*, 27(11):1617 – 1623, 2019.

[MKK+19] Haichao Miao, Tobias Klein, David Kouřil, Peter Mindek, Karsten Schatz, M Eduard Gröller, Barbora Kozlíková, Tobias Isenberg, and Ivan Viola. Multiscale molecular visualization. *Journal of molecular biology*, 431(6):1049–1070, 2019.

[MKM89] Forest K. Musgrave, Craig E. Kolb, and Robert S. Mace. The synthesis and rendering of eroded fractal terrains. *SIGGRAPH Computer Graphics*, 23(3):41–50, 1989.

[MKS+18] Peter Mindek, David Kouřil, Johannes Sorger, David Toloudis, Blair Lyons, Graham Johnson, Eduard Gröller, and Ivan Viola. Visualization multi-pipeline for communicating biology. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):883–892, 2018.

[PJR+14] Julius Parulek, Daniel Jönsson, Timo Ropinski, Stefan Bruckner, Anders Ynnerman, and Ivan Viola. Continuous levels-of-detail and visual abstraction for seamless molecular visualization. In *Computer Graphics Forum*, volume 33, pages 276–287. Wiley Online Library, 2014.

[PM01] Yoav I. H. Parish and Pascal Müller. Procedural modeling of cities. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 301–308. ACM, 2001.

[RMV16] Matthias Reisacher, Mathieu Le Muzic, and Ivan Viola. Cellpathway: A simulation tool for illustrative visualization of biochemical networks. In *Proceedings of WSCG*, 2016.

[RPC+10] Damien Rohmer, Tiberiu Popa, Marie-Paule Cani, Stefanie Hahmann, and Alla Sheffer. Animation wrinkling: Augmenting coarse cloth simulations with realistic-looking wrinkles. *ACM Transactions on Graphics*, 29(6):157:1–157:8, 2010.

[RS16] Tyler Reddy and Mark S.P. Sansom. Computational virology: From the inside out. *Biochimica et Biophysica Acta (BBA) - Biomembranes*, 1858(7, Part B):1610–1618, 2016.

[RZ14] Angelo Rosa and Christophe Zimmer. Computational models of large-scale genome architecture. *International review of cell and molecular biology*, 307:275–349, 2014.

[SB+01] Joel R Stiles, Thomas M Bartol, et al. Monte carlo methods for simulating realistic synaptic microphysiology using mcell. *Computational neuroscience: realistic modeling for experimentalists*, pages 87–127, 2001.

[SCS+16] Devika Sirohi, Zhenguo Chen, Lei Sun, Thomas Klose, Theodore C. Pierson, Michael G. Rossmann, and Richard J. Kuhn. The 3.8 Å resolution cryo-EM structure of Zika virus. *Science*, 352(6284):467–470, 2016.

[SKK+14a] Markus Steinberger, Michael Kenzel, Bernhard Kainz, Jörg Müller, Peter Wonka, and Dieter Schmalstieg. Parallel generation of architecture on the GPU. *Computer Graphics Forum*, 33(2):73–82, 2014.

[SKK+14b] Markus Steinberger, Michael Kenzel, Bernhard Kainz, Peter Wonka, and Dieter Schmalstieg. On-the-fly generation and rendering of infinite cities on the GPU. *Computer Graphics Forum*, 33(2):105–114, 2014.

[SKPE19] Karsten Schatz, Michael Krone, Jürgen Pleiss, and Thomas Ertl. Interactive visualization of biomolecules' dynamic and complex properties. *The European Physical Journal Special Topics*, 227(14):1725–1739, 2019.

[SMK+16] Karsten Schatz, Christoph Müller, Michael Krone, Jens Schneider, Guido Reina, and Thomas Ertl. Interactive visual exploration of a trillion particles. In *2016 IEEE 6th Symposium on Large Data Analysis and Visualization (LDAV)*, pages 56–64. IEEE, 2016.

[SMR+17] Johannes Sorger, Peter Mindek, Peter Rautek, M. Eduard Gröller, Graham Johnson, and Ivan Viola. Metamorphers: Storytelling templates for illustrative animated transitions in molecular visualization. In *Proceedings of the 33rd Spring Conference on Computer Graphics*, SCCG '17, pages 2:1–2:10. ACM, 2017.

[SS10] Michael Schwarz and Hans-Peter Seidel. Fast parallel surface and solid voxelization on GPUs. *ACM Transactions on Graphics (TOG)*, 29(6):179, 2010.

[SSEH03] Joshua Schpok, Joseph Simons, David S. Ebert, and Charles Hansen. A real-time cloud modeling, rendering, and animation system. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 160–166. Eurographics Association, 2003.

[Sti04] Bradley J. Stith. Use of animation in teaching cell biology. *Cell Biology Education*, 3(3):181–188, 2004.

[Tom01] Masaru Tomita. Whole-cell simulation: a grand challenge of the 21st century. *TRENDS in Biotechnology*, 19(6):205–210, 2001.

[VDZLBI11] Matthew Van Der Zwan, Wouter Lueks, Henk Bekker, and Tobias Isenberg. Illustrative molecular visualization with continuous abstraction. In *Computer Graphics Forum*, volume 30, pages 683–690. Wiley Online Library, 2011.

[VLF11] Agnès Vendeville, Damien Lariviére, and Eric Fourmentin. An inventory of the bacterial macromolecular components and their spatial organization. *FEMS Microbiology Reviews*, 35(2):395–414, 2011.

[Wan61] Hao Wang. Proving theorems by pattern recognition-II. *Bell Labs Technical Journal*, 40(1):1–41, 1961.

[WMT+14] Benjamin G. Wilhelm, Sunit Mandad, Sven Truckenbrodt, Katharina Kröhnert, Christina Schäfer, Burkhard Rammner, Seong Joo Koo, Gala A. Claßen, Michael Krauss, Volker Haucke, Henning Urlaub, and Silvio O. Rizzoli. Composition of isolated synaptic boutons reveals the amounts of vesicle trafficking proteins. *Science*, 344(6187):1023–1028, 2014.

[WMW+08] Benjamin Watson, Pascal Müller, Peter Wonka, Chris Sexton, Oleg Veryovka, and Andy Fuller. Procedural urban modeling in practice. *Computer Graphics and Applications, IEEE*, 28:18 – 26, 2008.

[WS04] Chaoli Wang and Han-Wei Shen. A framework for rendering large time-varying data using wavelet-based time-space partitioning (wtsp) tree. 2004.

[WTHT14] 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):334, 2014.

[WWLM+19] Nicholas Waldin, Manuela Waldner, Mathieu Le Muzic, Eduard Gröller, David S. Goodsell, Ludovic Autin, Arthur J. Olson, and Ivan Viola. Cuttlefish: Color mapping for dynamic multi-scale visualizations. *Computer Graphics Forum*, 2019. Preprint.

Part II

Publications

CHAPTER 4

Paper A - Parallel Generation and Visualization of Bacterial Genome Structures

DOI: 10.1111/cgf.13816
Pacific Graphics 2019
C. Theobalt, J. Lee, and G. Wetzstein
(Guest Editors)

Volume 38 (2019), Number 7

Parallel Generation and Visualization of Bacterial Genome Structures

T. Klein[1] P. Mindek[1] L. Autin[2] D. S. Goodsell[2,3] A. J. Olson[2] E. M. Gröller[1,4] I. Viola[5]

[1] TU Wien, Austria [2] The Scripps Research Institute, USA [3] RCSB Protein Data Bank, Rutgers State University of New Jersey, USA
[4] VRVis Research Center, Austria [5] KAUST, Kingdom of Saudi Arabia

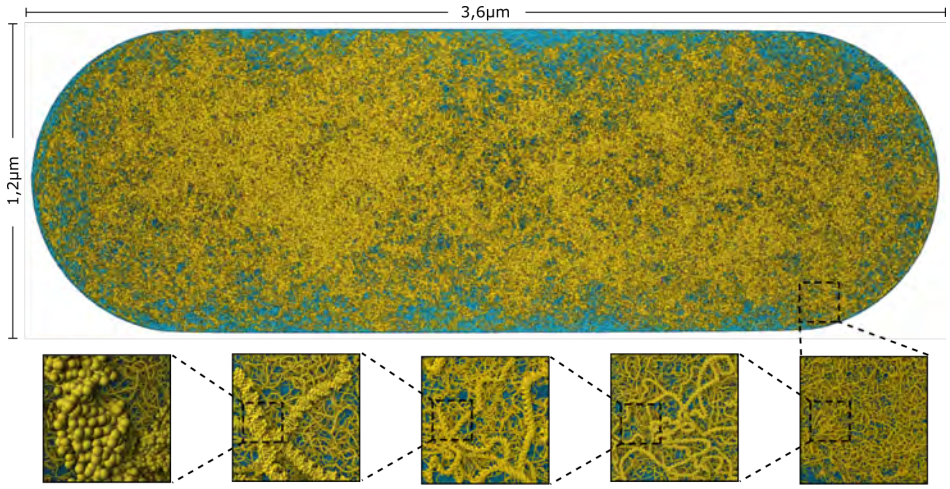


Figure 1: *Atomistic DNA rendering of the Sorangium cellulosum bacterium enclosed in a rod-shaped lipid membrane. The DNA structure is generated instantaneously using our new parallel modeling approach. Sorangium cellulosum currently holds the record for the largest known bacterial genome consisting of about 13 million base pairs.*

Abstract

Visualization of biological mesoscale models provides a glimpse at the inner workings of living cells. One of the most complex components of these models is DNA, which is of fundamental importance for all forms of life. Modeling the 3D structure of genomes has previously only been attempted by sequential approaches. We present the first parallel approach for the instant construction of DNA structures. Traditionally, such structures are generated with algorithms like random walk, which have inherent sequential constraints. These algorithms result in the desired structure, are easy to control, and simple to formulate. Their execution, however, is very time-consuming, as they are not designed to exploit parallelism. We propose an approach to parallelize the process, facilitating an implementation on the GPU.

1. Introduction

Visualization and computer graphics have shown much success in procedural modeling of nature, mostly through simulating images of objects at familiar, everyday scales. Non-biological natural phenomena, such as mountains, rock types, oceans, gases, clouds, or plasma, have been procedurally modeled for several decades [HRRG08]. Organic phenomena like vegetation, large forests [DN04], or dense jungles are modeled by procedural approaches and provide rich detail down to a single tree, or even leaves. The procedural generation of man-made structures is another example. A procedure can direct the generation of single room interior arrangements and can be scaled up to entire urban landscapes [SKK*14a; SKK*14b].

While in computer graphics, the goal is to generate and render visually plausible sceneries, in scientific visualization, the model generation and visualization has to be scientifically accurate, preserving key properties of the studied phenomenon. In biology, the generation and rendering of models of the biological mesoscale (the level between the nanoscale of atoms and the microscale of

cells) turned out to be a particular challenge. Such models are highly complex and heterogenous, and nanoscale objects cannot be directly observed.

The process of modelling mesoscale phenomena starts with understanding the hierarchical structure of living systems. At the finest level, atoms are specifically bonded to form large molecules, such as proteins and nucleic acids. They have characteristic structures and properties, and perform specific tasks. These molecules then interact and associate to form the microscale ultrastructure of the cell, which includes membrane-bounded compartments, and a variety of molecular infrastructure for regulation, support, transport, and communication. Reproducing the distinct shapes and interactions of these components, at scale ranges from nanometers to microns, is essential for creating accurate models of the structure and function of microorganisms.

Deoxyribonucleic acid (DNA) plays a central role in this hierarchy of structure and function. It is one of the most omnipresent structures representing and enabling life. DNA is a long polymer of four types of nucleotides, which carries the genetic information in the specific sequence of the nucleotides. Diverse experimental studies reveal many aspects of the DNA structure. This includes the detailed genetic sequence, topology and packing of the DNA into cells, and its interaction with the other molecular machinery of the cell. These results are making it possible to generate data-driven models of whole bacterial genomes, as shown in Figure 1.

Previous approaches generate such structures sequentially often by concatenating building blocks in many iterations. The main drawback of such procedures is the computation time, requiring minutes to hours to generate models of large-scale bacterial genomes. This limits the applicability for many visualization purposes, where parameters or constraints of the DNA structure are computed on the fly. Furthermore, biological studies that require numerous instances of such structures are limited to coarse-grained models, where the building blocks consist of several thousands of nucleotides. Our approach overcomes these limitations by generating DNA models instantaneously in a parallel fashion, similar to most real-time content generation procedures. Due to the complexity and constraints of the DNA structure, this is a non-trivial task.

Our central idea is to use a divide-and-conquer strategy together with a midpoint displacement algorithm for growing a DNA polymer. Initially, this procedure disregards any potential overlaps of the DNA structure to ignore sequential constraints for the moment. Subsequently, overlaps are efficiently detected and eliminated by a force-based system, similar to the ones used in fluid simulations. Our main contributions include:

- A problem characterization of generating large-scale DNA structures from the perspective of scientifically-accurate visualization
- Novel parallel algorithms for the instant generation of bacterial DNA structures
- An interactive visual environment allowing users to quickly experiment with different DNA generation parameters and constraints to perform *visual* hypothesis generation
- The foundation for *computational* hypothesis generation and verification that requires the generation of a multitude of bacterial genome models

2. Background and Related Work

In this section, we characterize architectural basics of bacterial genomes and review literature related to the generation and visualization of their structures.

Genome Architecture

The atomic details of DNA double helices were revealed in the classic work of Watson and Crick [DH53] and the bacterial nucleoid (the region containing most of the genetic material) was already described more than 60 years ago [Kel58]. However, structures of entire DNA genomes are still subject of intense study and only recently detailed information has become available [Dor13] through various experimental studies [DT16; KFS*14]. Bacterial cells typically have circular genomes, which solves an intrinsic problem with DNA replication: The enzyme DNA polymerase cannot replicate linear DNA to the end, so repeated rounds of replication lead to progressive shortening of linear DNA. Bacterial genomes are also typically highly supercoiled, leading to formation of a compactly-folded and functionally-organized structure [DT16]. Topoisomerase enzymes underwind the two strands of the DNA helix in an energy-dependent process. The underwound superhelical stress then leads to the formation of locally supercoiled loops termed 'plectonemes'. Supercoiling is generally thought to assist with the processes of DNA replication and transcription, which involve unwinding of the double helix. Supercoils have been characterized in plasmids (small circles of DNA) by electron microscopy [BWC90]. Higher-order structure of DNA is often probed using methods such as Hi-C (a technique to study the three-dimensional architecture of genomes). Hi-C uses selective crosslinking to identify regions of a genome that are in proximity. The flexibility of DNA remains a matter of conjecture. It is generally seen to be relatively rigid but there are abundant examples of kinking and bending under the influence of proteins. In general, the flexibility of polymers, like DNA, is quantified with a property called 'persistence length' defining how orientational correlations decay along the polymer chain. Studies indicate that DNA has a persistence length of around 500 Å(ngström) [Hag88]. Ångström is the unit in natural sciences for expressing the size of atoms, where 1 Å is equal to 10^{-10} m [PM10].

Generation and Visualization of DNA Structures

Early modeling and visualization of DNA structures used models built entirely from scratch. An example is the ground-breaking animated zoom from atoms to chromosomes by Max [Max85]. More recently, there have been several attempts at creating user-friendly tools, such as NAB [SJJ95] and GraphiteLifeExplorer [HF13]. These tools successfully model local details of DNA and its interaction with proteins, but are not scalable to model entire genomes. Modeling of entire genomes has been attempted by several groups. Coarse-grained techniques are often used, where segments of the DNA are represented by single beads. For bacteria these beads may represent single turns of the DNA helix [GAO18] or larger interaction domains [TYM*17]. For larger eukaryotic genomes, these beads often represent locally-compact domains of many thousand base pairs [RZ14]. Modeling of entire genomes at the atomic level has recently been attempted for bacteria, e.g., using a stepwise process with progressively-detailed coarse-graining [HLE17].

Many polymer-modeling methods produce only static models. In contrast, Kolesar et al. [KPV*14] propose a technique for modeling

the polymerization process itself. The work of Klein et al. [KAK*18] presents a novel technique for real-time generation and visualization of biological mesoscale models including fibrous DNA structures. The drawback of this method is its sequential implementation, which constitutes a severe performance bottleneck. Recent work by Lindow et al. [LBLH19] interactively visualizes RNA and DNA structures, but focuses on small-scale models with less than 10,000 nucleotides. The cellVIEW [MAPV15] system implements a novel approach for the assembling of DNA on the GPU. However, it requires an existing curve, approximated by discrete points, which are subsequently interpolated on the GPU.

Random Walk Methods

A random walk is a stochastic process used to model various real-world phenomena. Random walk algorithms often simulate movement and growth, or generate models of linear polymers like nucleic acids. A random walk is a special type of a Markov chain, where each generated point is dependent on its predecessor. We define the result of a random walk similar to Altendorf and Jeulin [AJ11] as a sequence P of points: $P = \{p_0, ..., p_n\}$ with $p_i = (x_i, \omega_i) \in \mathbb{R}^3 \times S^2$. Every point p_i consists of a location x_i and an orientation ω_i.

To model bacterial genomes, a correlated random walk with barriers is typically applied, where each point represents one bead of the genome model. The barrier is defined by a compartment, e.g. the cell nucleoid, that encapsulates the genome. The correlation is defined by the dependency of the orientations among successive steps. The random walk starts with an initial point $p_0 = (x_0, \omega_0)$, located inside of the compartment, and a random orientation. The position x_{i+1} of the next point p_{i+1} is generated through a random walk step, starting from the current position x_i along a new random orientation ω_{i+1}. It is calculated as follows:

$$x_{i+1} = x_i + s\omega_{i+1} \tag{1}$$

where $s \in \mathbb{R}^+$ is the step size of the random walk.

For more detail, we refer to the work of Spitzer [Spi01], which covers the theoretical foundation of random walks. Additionally, Codling et al. [CJB08] provide an extensive overview of random walks for modeling biological processes.

Midpoint Displacement Approaches

The power of parallel processors facilitates the generation of large models of bacterial genomes in real time. However, current genome modeling approaches, like random walks, have intrinsic sequential constraints and thus do not map well to parallel processors. Another approach to approximate various natural processes is midpoint displacement. In contrast to the random walk algorithm, it is well suited for parallelization. The midpoint displacement method has first been introduced by Mandelbrot [B M83] in the context of fractals. It became widely popular after Fournier et al. [FFC82] presented its extension to the diamond-square algorithm. The algorithm generates random heightmaps that are frequently used for terrain models. In the simple 1D case, the midpoint displacement algorithm starts with a line between two points. In the first step, the midpoint of the two initial points is displaced perpendicular to the line segment by a random amount. This process is repeated on the resulting new line segments until the desired level of detail is attained. The displacement magnitude is reduced in each iteration.

Midpoint displacement has been used and extended in various ways. Musgrave et al. [MM89] propose a midpoint displacement method for the generation of locally-controllable fractal terrains. They modify a standard midpoint displacement generation method by simulating erosion features to increase the realism of the generated terrains. Jilesen et al. [JKL12] expand the typical two-dimensional case of midpoint displacement to produce periodic, three-dimensional models of porous media. The method is suitable to create realistic models of rocks, as demonstrated through a comparison with two-dimensional cross-sections of real geological material. While most often used to model geological processes, the application of midpoint displacement in other areas is also well established. For instance, midpoint displacement has been used to simulate Brownian motion, which has various applications in biology [NML00].

3. Overview

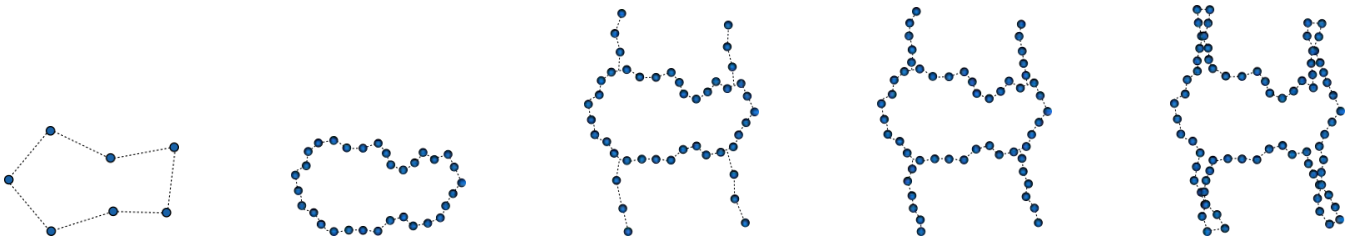
The complex hierarchical structure of DNA, ranging from specific nucleotide sequences at the atomic level to supercoiled topologies at the whole-genome level, poses challenges for modeling. Random walks and similar sequential algorithms are particularly suitable for generating models with shape and flexibility constraints, but are limited in their performance. This is especially problematic for large genome structures, like the genome of Sorangium cellulosum with about 13 million DNA base pairs. We introduce a divide-and-conquer approach, which makes it possible to parallelize the modeling process. Additionally, the real-time GPU implementation opens up the possibility of displaying models that are generally too large to fit into the memory. In the following we describe a typical sequential genome modeling approach and compare it to our parallel one.

Sequential Approach

Coarse-grained models of DNA, where a chain of beads is used to model the linear or circular polymer, may be constructed using a random walk. At each step the random walk chooses a random direction depending on the flexibility and shape of the surrounding compartment. The random walk starts with an initial seed point located inside of the compartment and walks along the first step. If a random walk step chooses a point outside of the given compartment or an occupied area, a new random direction is chosen. This process is repeated until a valid position is found. If no valid position can be found, the random walk retreats to an earlier state and tries again. When the coarse-grained chain of beads is computed, each bead may be exchanged with a more-detailed structural building block to assemble the final structure. Supercoils can be added by starting subsequent random walks from certain points of the generated structure. Other approaches [GAO18] start with a small square of points on a lattice and enlarge and migrate them to build the genome model. However, all of these sequential approaches require minutes up to hours to generate larger genome models.

Parallel Approach

Our parallel approach to generate bacterial genome models is illustrated in Figure 2. In order to map the computation of the model onto several threads of a parallel processor, an initial set of beads is required. For this reason, we first generate a rough backbone of the genome model. In theory, a randomly-placed circular set of beads could be used. However, in order to make this approach applicable to a variety of constraints and enclosing shapes, we choose to use



(a) *coarse-grained backbone* **(b)** *detailed backbone* **(c)** *coarse-grained supercoil axes* **(d)** *detailed supercoil axes* **(e)** *detailed supercoiled hairpins*

Figure 2: *Parallel approach for the generation of bacterial genomes. First, a sequential random walk with a large step size is applied to generated the main backbone (a), a coarse circular structure. The generated beads build the foundation for the subsequent parallel processing. In the next step, the circular structure is enhanced in detail (b) through our parallel midpoint displacement method. Subsequently coarse-grained supercoil axes (c) are generated with random walks branching from the backbone. Again, detail is added to the supercoils (d) with our parallel midpoint displacement method. Finally, superhelical hairpins (e) are generated by splitting the beads of the supercoil axes in two and rotating them resulting in the final model of the bacterial genome.*

a random walk to create the initial backbone (Figure 2a). Subsequently, the backbone is filled with detail (Figure 2b) through a parallel implementation of the midpoint displacement algorithm. For each pair of initial beads, the midpoint displacement algorithm computes a new midpoint and displaces it by a random amount, restricted by the constraints of the model. The midpoint displacement results in a sequence of beads with varying spacing. In order to fit equally-spaced building blocks of the polymer onto the beads, we need to perform a uniform resampling. Uniform resampling of a sequence is a sequential process, potentially slowing down the whole generation process. Therefore, we substitute uniform resampling with an approximation that can be executed in parallel. After the backbone is generated, supercoils are added to the structure. First, the coarse-grained versions of the supercoil axes are computed (Figure 2c), also using the random walk algorithm. Again, detail is added with the parallel midpoint displacement (Figure 2d) and the bead sequence is resampled. In the final step, the supercoiled hairpins are generated by splitting the beads of the supercoil axes into two chains and rotating them about the axes (Figure 2e). This is also computed in a parallel fashion. In order to generate an atomistic model of the bacterial genome, we calculate the orientation of structural units along the chain of beads in parallel and assemble them with an optimized version of the cellVIEW approach [MAPV15].

Applications

We expect that our parallel approach will be useful both in education/outreach settings and in research. For education, the interactive nature of the method is crucial. The ability to construct new models on the fly allows users to gain a more intuitive understanding of the hierarchical relationships between atoms and nucleotides, genes, entire genomes, and cells. The ability to explore different levels of superhelicity will help users to comprehend the role of DNA topology in the packing and function of the genome.

In research, there is a growing interest in the structure of bacterial genomes. It is becoming clear that supercoiling of DNA and subsequent condensation of chromosome interaction domains play a role in the regulation of gene expression [MLS17]. This has lead to the examination of multiple species using techniques such as Hi-C and fluorescence microscopy to quantify the location and interactions of domains within the genomic DNA. These techniques typically provide only a coarse-grained view, with resolutions of just thousands of base pairs. Therefore, modeling is employed to develop hypotheses for structural features at the finer scales.

Given the complexity of the system, most current modeling studies employ one of two simplifications. Coarse-grained modeling techniques support the creation and testing of multiple instances of multiple models, but at the cost of providing reduced-resolution results. Most often, these approaches treat individual chromosome interaction domains as the coarse-grained units, and provide useful information on the overall shape and packing of these domains in the interior space of the cell. More recently, several groups have created full atomic models of bacterial genomes, constrained by available biochemical observations. These models are laborious to produce, and thus typically only a handful of instances are generated. Nevertheless, they open up the ability to calculate detailed properties of the genome that will be important for understanding its interactions with regulatory proteins and the transcription machinery.

The ability to create detailed models in real time reduces the limitations of previous methods. For example, the simulation of Hi-C data requires the evaluation of contact information from many individuals in a population, recapitulating the experimental process of low-probability crosslinking in a culture of many cells. Rapid generation also facilitates the evaluation of multiple hypotheses. For example, the nature of each chromosome interaction domain is currently not known: It could be composed of a single supercoiled plectoneme, several tandem plectonemes, or a complex branched structure. The ability to rapidly build different models exploring multiple hypotheses, and for each, to build multiple stochastic instances of the model, will allow ready comparison to explore the nature of consistent structures. There are also direct connections to medical science that are becoming possible. Several classes of antibiotics, such as quinolones, attack topoisomerases and supercoiling. Studying the cellular consequences of these drugs in multiple bacterial physiological states (rapid growth, starvation, etc.) will help to identify aspects of the process that are amenable for continued antibiotic development.

4. Pipeline for Parallel Genome Generation

In the following, we describe the design of our parallel pipeline for the interactive generation of bacterial-genome models. Each step is designed to incorporate known characteristics and constraints of bacterial genomes.

4.1. Backbone Construction

In the initial step of the pipeline, we construct a rough backbone of the desired polymer structure with a random walk algorithm. The structural rigidity of DNA polymers is typically quantified with the persistence length. It defines the length over which the correlations of the tangents are lost. We use the following well-known equation [KP49] to incorporate the persistence length into the modeling process:

$$\langle \cos\theta \rangle = e^{-L/P} \tag{2}$$

The formula describes that the expectation value of the cosine of the angle θ between two tangents of the polymer chain falls off exponentially with distance L, where P denotes the persistence length. We use Equation 2 to approximate the orientations for the correlated random walk so that it reflects a given persistence length. First an angle θ_{max} is computed that indicates the maximum angle difference between the previous and the new orientation. Then we choose a new orientation using a random angle difference between 0° and θ_{max}; essentially, the computation of a new orientation is sampled from a spherical cap. The angle θ_{max} is defined as $acos(2e^{-L/P} - 1)$.

With the new orientation and Equation 1, the location of the next bead is computed. If a bead lies outside of a barrier, the algorithm retreats and uses a new orientation. This process is repeated until a valid position is found. The random walk algorithm stops when the given number of steps is reached. In order to support closed, circular DNA structures, we introduce an additional bias to the random walk with a direction towards the first initial bead. The bias grows with the length of the generated structure so that is has small to no influence at the beginning of the process.

To save computation time, the backbone is computed with a low resolution, which introduces gaps in between beads. Filling of the gaps increases the number of beads 2^n times, where n is the number of midpoint displacement steps. In practice, we use a maximum of ten midpoint displacement steps, which makes the backbone resolution 1024 times smaller than the resolution of the final sequence. However, using ten midpoint displacement steps is not a theoretical limit of the approach. It corresponds to a performance optimization of the algorithm that uses the shared memory of the GPU, which is limited in size.

4.2. Detail Insertion

Our version of the midpoint displacement approximates a given persistence length by utilizing a greedy approach. In every midpoint displacement step, we insert new beads into the chain and constrain the displacement using the given persistence. We show that the persistence length is efficiently approximated in Section 6.

For every pair of beads p_l and p_r with tangents $\vec{t_l}$ and $\vec{t_r}$, we introduce a new bead p_m using an adapted version of the midpoint displacement algorithm, as illustrated in Figure 3a. The amount of displacement is affected by the tangents and the persistence length. We define the midpoint displacement similar to the random walk algorithm. We start from bead p_l and walk in a direction $\vec{s_r}$ resulting in the displaced midpoint p_m. In order to choose an appropriate direction $\vec{s_r}$, first, a general direction $\vec{s_a}$ is calculated. The direction $\vec{s_a}$ represents a smooth continuation of the bead chain and is determined with the tangents $\vec{t_l}$ and $\vec{t_r}$. Then, the general direction is slightly varied resulting in the random direction $\vec{s_r}$.

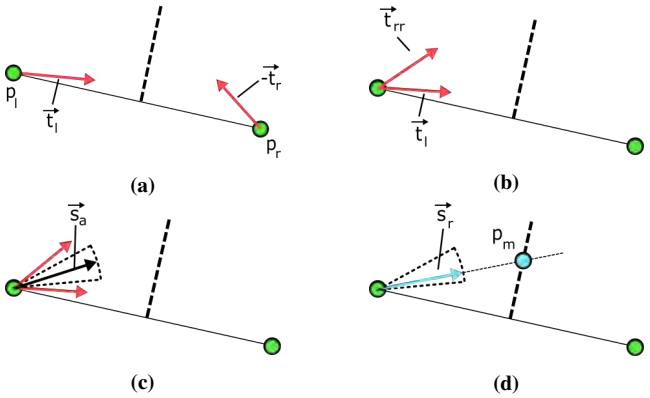


Figure 3: *Adapted midpoint displacement: (a) segment of the generated backbone with beads p_l and p_r and tangents $\vec{t_l}$ and $-\vec{t_r}$. (b) preparation step, where the tangent $-\vec{t_r}$ is reflected at the mid plane (dashed line). (c) depicts the bisector $\vec{s_a}$ of the two vectors $\vec{t_l}$ and $\vec{t_{rr}}$, which is used to find the new midpoint. (d) the midpoint p_m is computed through the intersection of a ray along the new direction $\vec{s_r}$ and the mid plane. The vector $\vec{s_r}$ is chosen randomly from a spherical cap surrounding $\vec{s_a}$.*

For the calculation of $\vec{s_a}$, we originate the direction at bead p_l and adjust the tangents accordingly. This means, we reflect $-\vec{t_r}$ across the mid plane (indicated with dashed lines) resulting in $\vec{t_{rr}}$, as shown in Figure 3b. The mid plane is defined by the center between p_l and p_r, and the normal $\vec{n} = \overrightarrow{p_l p_r} / ||\overrightarrow{p_l p_r}||$. With this the general direction is defined as $\vec{s_a} = (\overrightarrow{t_l} + \overrightarrow{t_{rr}}) / ||\overrightarrow{t_l} + \overrightarrow{t_{rr}}||$ (see Figure 3c). Subsequently, we choose a random direction $\vec{s_r}$ from the spherical cap surrounding $\vec{s_a}$, as depicted in Figure 3d.

Finally, the new midpoint p_m is computed through the intersection of a ray, starting at p_l with direction s_r, and the mid plane. In case the location of the new midpoint is outside of the compartment, we repeat the process with a new random direction s_r.

Uniform Resampling

The parallel midpoint displacement greatly improves the computation time, however, it does generate beads with non-uniform distances in between. To be able to fit equally sized building blocks on the beads, the bead chain needs to be uniformly resampled. Resampling is a sequential process, where each sampling step depends on its predecessor. To improve the performance of this process, we approximate uniform resampling so that it is suitable for parallel processing.

We utilize the idea of the divide and conquer design paradigm and first subdivide the sequence of beads into subsequences P_j. We de-

fine the subsequences as $P_j = \{p_{j(s-1)}, ..., p_{(j+1)(s-1)}\}$, where each subsequence contains s beads. Adjacent subsequences P_j, P_{j+1} have common border beads. Even though the uniform resampling within a subsequence is still a sequential process, the separate subsequences can now be processed in parallel.

Through the resampling, each subsequence P_j results in a resampled subsequence R_j. Within each resampled subsequence, the distances between the beads are uniform. However, the distances of adjacent resampled subsequences are still non-uniform. We quantify the offset between the desired uniform distance and the actual distance between adjacent resampled subsequences with the error vector $\vec{e_j}$. The vector $\vec{e_j}$ is defined as the distance between the first bead of a resampled subsequence R_j and the last bead of the non-resampled subsequence P_{j-1}. The error vector of the first subsequence is undefined, if the structure is not circular. If $\vec{r_j}$ is the last bead of the resampled subsequence R_j, then $\vec{e_j} = p_{j(s-1)} - r_{j-1}$. The maximum length of the error vector is always smaller than the sample distance. In order to minimize potential artifacts, we equalize the error among all beads of a subsqeuence. In detail, for a resampled subsequence R_j with indices $i = \{0, ..., n\}$ for its beads, we move each bead by $\frac{n-i}{n+1}\vec{e_j}$. Consequently, the last bead of a subsquence is not moved with the purpose of retaining the distance to the following subsequence.

Genome Length Estimation
In order to generate genome models with specific lengths, we utilize a heuristic approach. Due to the parallel midpoint displacement and the parallel uniform resampling the exact number of generated beads cannot be efficiently determined beforehand. For this reason, a heuristic is computed that estimates the resulting number of beads for a given parameter pair (random walk step size, persistence length). Experiments have shown that after several iterations of the algorithm, the number of generated beads for a specific parameter pair converges to a stable mean value. For example, an experiment with 500 repetitions of different seed values using a persistence length of 500 and 10,000 random walking steps results in a mean of $\mu = 327,863.95$ beads with a minimum of $324,550$, maximum of $329,669$, and standard deviation of $\sigma = 738.06$. For many visualization purposes the influence of such a variance is insignificant to the overall perception of the generated genome structure proofing the mean value to be a reliable estimator.

However, in some modeling scenarios the exact sequence of the genome is known and should be reflected by the model. Here, we slightly overestimate the length of the model and subsequently apply a trimming, to shorten it to the exact length by removing surplus beads. The removed beads create gaps in the structure, since its neighbors now have a larger distance. These gaps are automatically repaired in the subsequent relaxation step and the proper distances are restored.

4.3. Supercoiling

We model supercoils in three steps, as depicted in Figure 4. In the first step (a), a new sequence of beads (yellow) is generated that branches from the main backbone (blue) representing the supercoil axis. In the second step (b), the beads are replicated and shifted apart from each other. Finally, the beads are rotated around the supercoil axis (c) to generate the twist of the supercoil. The actual values for the twisting and shifting should be chosen corresponding to the characteristic of the desired genome structure.

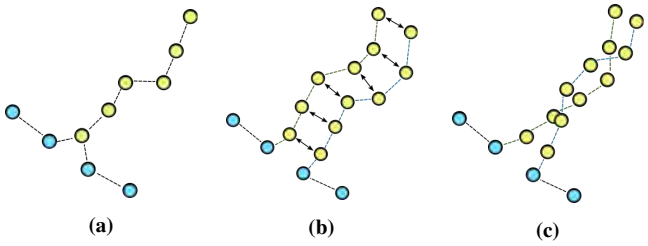


Figure 4: *In the initial step (a) of the supercoil generation, a sequence of beads (yellow) is generated that branches from the backbone (blue). Then (b) the generated beads are replicated and shifted apart from each other. In the last step (c), the beads are rotated to form the supercoil.*

4.4. Overlap Detection and Relaxation

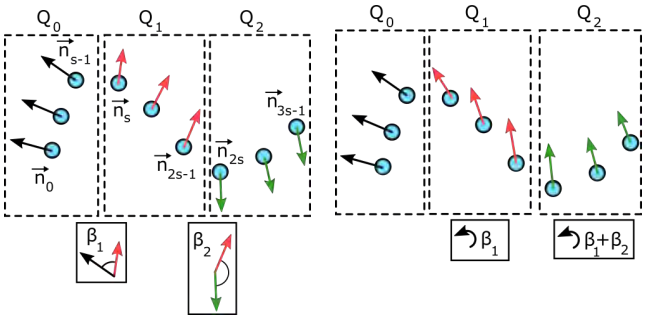
Due to performance reasons, the generation of the backbone and the subsequent detail insertion do not take into account if space is already occupied. Performing space occupancy checks requires synchronization, and would thus introduce sequential constraints in the parallel process. Therefore, we efficiently resolve overlapping through a subsequent relaxation process. The relaxation is based on a force-based system. First, overlaps are detected through a fixed radius nearest neighbor search, as described by Hoetzlein [Hoe14]. This approach is capable of simulating millions of beads in real-time and is often used in fluid simulations. Overlaps are resolved with a combination of a *repulsion* and a *recover force*, based on the work of Altendorf and Jeulin [AJ11]. The *repulsion force* drives overlapping beads away from each other, whereas the *recover force* models a springlike force and recovers structural constraints. The constraints comprise the correct uniform distances between beads and a maximum amount of kinking in the structure.

The application of forces can potentially move parts of the structure outside of its enclosing compartment. For this reason, we apply an additional force that moves parts that now lie outside back into the compartment. The forces are applied until all collisions are resolved or a stop criterion is reached. This criterion can be set by the user and specifies a certain number of acceptable remaining collisions.

4.5. Genome Visualization

Coarse-grained models of bacterial genomes are sufficient for exploring hypotheses about overall packing and topology. However, to explore sequence-specific properties, such as the interaction with proteins, more detailed atomic information is needed, which is provided by finer models. To visualize the atomic structure of the genome, a model is required that shows the individual DNA base pairs and their proper orientations.

Normal Computation
In order to coherently orient the building blocks along the bead



(a) (b)

Figure 5: *Overview of the parallel normal computation. In the first step (a), the computation of the normals is parallelized by dividing the computation into several subsequences (indicated with dashed lines) to facilitate parallel processing. This leads to discontinuities at the borders between subsequences, which is quantified with the error angle β_i. Discontinuities are corrected by rotating all normals of a subsequence by the accumulated error angles (b).*

sequence, the orientation of the beads must be computed. This is typically done by computing the orthonormal frame for each bead.

In differential geometry there are many simple methods, which compute a moving frame. However, they often exhibit discontinuities or strong torsion causing visible artifacts. For this reason, we approximate a rotation minimizing frame. The computation of the rotation minimizing frame is sequential in nature since the computation of each frame is dependent on its predecessor. Similar to our parallel resampling approach, we apply the divide-and-conquer scheme and break the computation down into multiple subsequences (Figure 5). We define the subsequences as $Q_j = \{p_{js}, ..., p_{(j+1)s-1}\}$, where each subsequence contains s beads.

For each bead x_i, we compute a corresponding normal n_i. To start the process, we choose an initial random normal for each first bead of every subsequence and compute the remaining normals according to the rules of rotation minimizing frames. Since the normal computation is only continuous within a subsequences, the process leads to discontinuities in between subsequences. We quantify the discontinuity for a subsequence Q_j with the error angle β_j. In detail, the angle β_j is the angle between the last normal of subsequence Q_{j-1} and the first of subsequence Q_j. It is defined as $\beta_j = acos(n_{js} \cdot n_{js-1})$. In order to restore continuity between subsequences, we first apply a prefix sum to accumulate the error angles. Then, we rotate the normals by the accumulated error angles around their corresponding tangents. This means, the continuity is restored by rotating all normals of a subsequence Q_j by the accumulated error angle, which is defined as $\beta_j = \beta_{j-1} + ... + \beta_0$.

Genome Assembling
Once the normals are computed, the final polymer strand can be constructed. In this step, the beads are exchanged with corresponding building blocks and the final image is rendered. In the visualizations of this work, we use the standard model of the B-DNA double helix, which consists of individual base pairs with an angular offset of

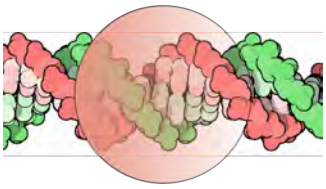


Figure 6: *Structural model of B-DNA overlayed with a bead representing a building block.*

34.3° and a spacing of 3.4 Å between each base, as depicted in Figure 6.

5. Implementation

For the reproducibility of the approach, we provide implementation details of our approach below.

Backbone Construction
In order to uniformly draw samples from a spherical cap we use the following equation: $a = \cos(\theta_{max})$, $z = X(1-a)+a$, $\psi = 2X\pi$, $r = \sqrt[2]{1-z^2}$, $\vec{\omega} = \{r\cos(\psi), r\sin(\psi), z\}$, where $X \sim U([0,1))$ is a random variable uniformly distributed between [0,1). This provides a random orientation ω with the maximal angle difference θ_{max} to the pole.

Detail Insertion
The midpoint displacement algorithm naturally lends itself to parallel processing since each level of repetition is independent from each other. Figure 7 illustrates the parallel implementation of the midpoint displacement algorithm. On the left side, each block represents a thread in the execution. Idle threads are depicted in blue and active ones in red. The right side depicts the beads, where the newly generated ones are shown in red, corresponding to the active threads. Horizontal lines indicate synchronization points between the individual iterations of the parallel midpoint displacement. In the first level, only one thread is active since only one midpoint can be created simultaneously. The number of active threads and generated beads doubles after each execution.

Uniform Resampling
To efficiently implement the parallel resampling, the resampled

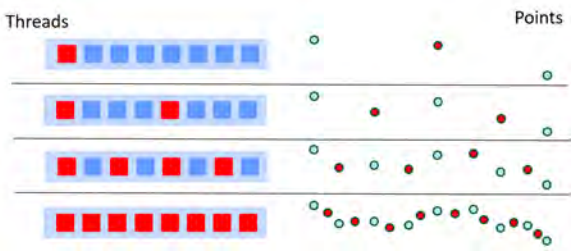


Figure 7: *Illustration of four iterations of our parallel midpoint displacement implementation. Every block (left) represents a thread and is depicted in red, if active, and blue, if inactive. The corresponding generated beads are also shown in red (right). The horizontal lines indicate the synchronization between consecutive iterations.*

(a) (b)

Figure 8: *Illustration of the trimming process. The beads (red) are overlayed with DNA building blocks. (a) shows the DNA structure with a trimmed bead (faded) before the relaxation. (b) shows the repaired DNA structure after the relaxation.*

beads are written in one continuous buffer of GPU memory. For this, we need to know beforehand the memory positions to which each thread has to write to. Therefore, we first compute the number of beads that each thread of the resampling will produce. Subsequently, we apply a prefix sum to the resulting values using the parallel scan algorithm [SHGO11]. With this information, we can resample the bead sequence in parallel.

Length Trimming

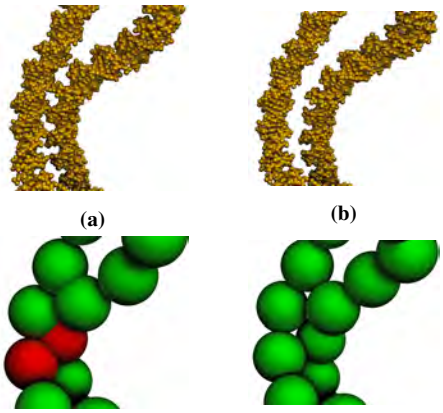
In case the exact length of the nucleic acid is known, we use a heuristic that slightly overestimates the resulting number of beads and trim surplus subsequently. To minimize the effect on the structure, we distribute the selection of the surplus beads uniformly along the structure. Figure 8 demonstrates one instance of the trimming process. Figure 8a shows a piece of a DNA structure, where one bead is trimmed. The illustrations show that the DNA structure would be stretched at the place where the bead is missing. In Figure 8b the hole in the structure is repaired due to the relaxation step and proper uniform distances are restored. In order to build a heuristic for various input parameters, we modeled a nonlinear regression using a set of sample results with a persistence length between 100 Å and 5000 Å, and 20 to 5000 random walk steps, whereas the remaining parameters were fixed. The nonlinear regression yields a goodness-of-fit of $R^2 = .998$ indicating a low discrepancy between observed and expected length.

Supercoiling

The supercoiling is parameterized with two values, the distance of the duplicated beads and the twisting factor that represents the level of supercoiling. These values are specified by the user and reflect

(a) (b) (c)

Figure 9: *Comparison of DNA with (a) low, (b) medium, and (c) high levels of supercoiling.*



(a) (b) (c) (d)

Figure 10: *Illustration of the overlap detection. (a) and (c) depict an occurrence of overlapping in atomistic detail and represented with beads, respectively. Beads highlighted in red indicate overlapping regions. (b) and (d) show the structure after the overlaps are resolved.*

results from experimental studies. In Figure 9, a visual comparison of DNA with different levels of supercoiling is shown.

Overlap Detection and Relaxation

The overlap detection and relaxation is processed on the basis of the beads. Each bead is modelled as a bounding sphere with the radius corresponding to its enclosing building block. The bead slightly overestimates the size of the DNA, as shown in Figure 6. However, the overestimation is a desired approximation from a perceptual point of view, since the elements are easier to distinguish if they are not in direct contact. Figure 10a and Figure 10c illustrate an instance of overlapping, shown with atomistic detail and shown with the corresponding beads, respectively. After the relaxation, shown in Figure 10b and Figure 10d, the overlapping is resolved and the two DNA strands have a distinct but small distance between them.

Normal Computation

Implementing the parallel computation of the normals consists of the following steps. In the first step, the tangents are computed for every bead. Then, the rotation minimizing frames are approximated. In our implementation, we utilize the concept proposed by Wang et al. [WJZL08]. In order to restore the continuity between neighboring subsequences, we first calculate the error angles and then accumulate them. To process the accumulation in parallel, we compute the prefix sum of the error angles using the parallel scan algorithm [SHGO11]. With the result of the prefix sum, we can also restore the continuity of the normals in parallel.

Genome Assembling

In order to assemble the DNA structure, we follow the approach of Le Muzic et al. [MAPV15], where the final assembling in atomistic detail is part of the rendering step. First, the bead sequence is subdivided into smaller beads, until each bead corresponds to a single DNA base pair. Afterwards, the normals for the newly introduced beads are interpolated on basis of the existing normals. In the same

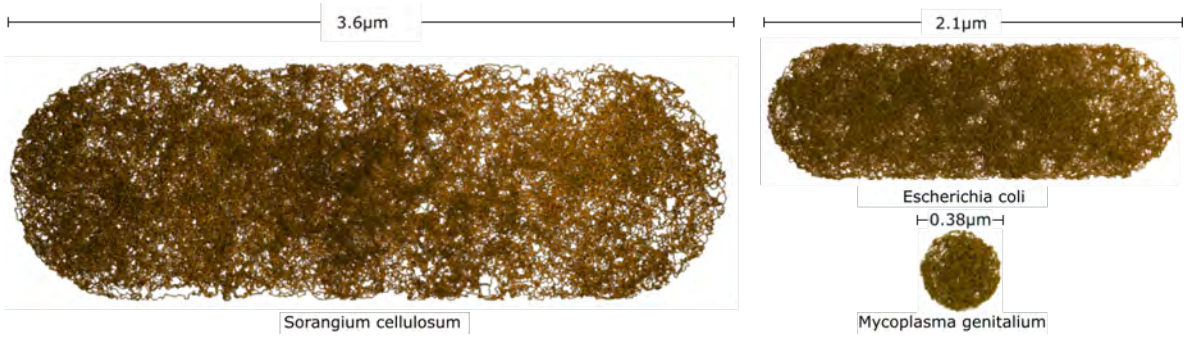


Figure 11: *Structural visualization of bacterial genomes. Left: Genome of Sorangium cellulosum, consisting of more than 13 million base pairs. Top right: Genome of Escherichia coli consisting of more than 4.6 million base pairs. Bottom right: Genome of a Mycoplasma genitalium consisting of more than 580,000 base pairs.*

step, every normal is rotated according to a given angular offset, which corresponds to the winding of the DNA. In the final step, each bead is substituted with the geometric model of the base pair.

6. Results and Evaluation

To evaluate the effectiveness of the method, we have created detailed data-driven models of the genomes of three bacteria. The performance was measured using a computer with an Intel Core i7-6700K CPU 4.00 GHz and a NVIDIA GeForce GTX 1080 graphics card with 8 GB memory. Since the final assembling of the genome model is part of the rendering process that was already presented in previous work [MAPV15], we do not include this step in the performance measurements.

Mycoplasma genitalium
The genome of Mycoplasma genitalium, with **580,000** base pairs (bp), is among the smallest of bacterial genomes. Figure 11 (bottom right) shows the rendering of the genome model, generated in 218 ms. Previous work [KAK*18] required about 100 seconds to generate the genome without including any complex supercoiling structures. Recent work [GAO18] that includes supercoils generate an equivalent model in about 9.7 minutes, thus we outperform the result with the same scientifically-accurate complexity 2600-fold. Figure 12 reveals a closer view of the structure of the genome enclosed in a spherical lipid membrane. The left side (a) shows the atomistic rendering where the right side (b) displays the model with beads. Individual genes are displayed with different colors.

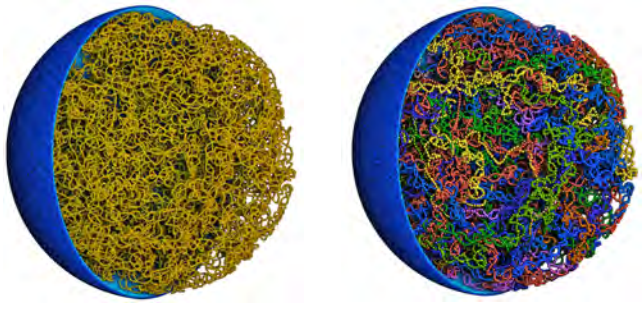


Figure 12: *Model of Mycoplasma genitalium. (a) shows the atomistic rendering of the DNA enclosed in a lipid membrane. (b) shows the individual genes of the model indicated with different colors.*

Escherichia coli
Escherichia coli is one of the best characterized organisms, with

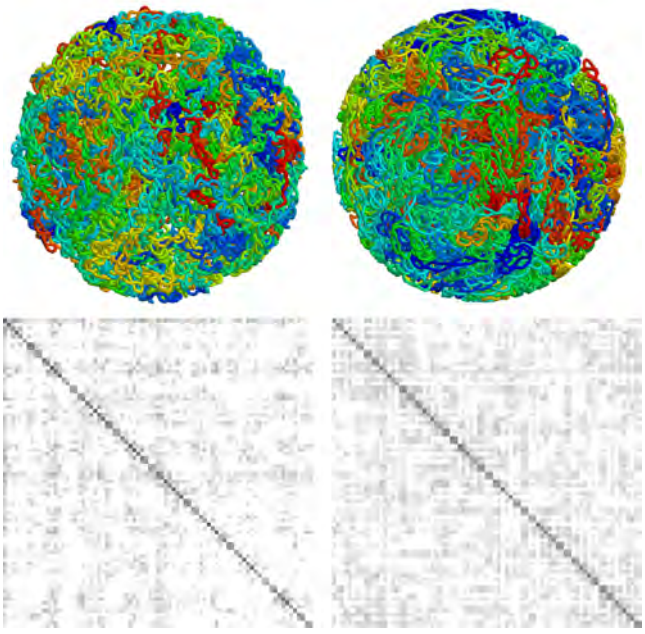


Figure 13: *Models of the Mycoplasma pneumonia genome, generated with (left) our parallel generation approach and (right) the lattice-based method [GAO18]. In the distance maps at the bottom, the entire genome is arrayed on the horizontal and vertical axes with a resolution of 10 kpb, and average distances are shown with a linear ramp from black at 0 nm to white at 200 nm and above. A representative genome model for each approach is shown at the top.*

Table 1: *Performance measurements of the generation of several bacterial genomes, separated into the different steps of the generation process.*

Model	Backbone	Detail	Supercoiling	Resampling	Trimming	Relaxation	Normals	Total
Mycoplasma genitalium	1 ms	11 ms	5 ms	35 ms	6 ms	150 ms	10 ms	218 ms
Escherichia coli	6 ms	39 ms	15 ms	136 ms	36 ms	112 ms	28 ms	372 ms
Sorangium cellulosum	19 ms	86 ms	26 ms	282 ms	88 ms	360 ms	81 ms	942 ms

a genome of about **4.6 million** bp. As in previous work [GAO18], we have separated the genome into 169 units. Each unit consists of a 11,000 bp long plectoneme and an unsupercoiled connecting segment of 16,000 bp. The membrane of the cell is rod-shaped with an approximate diameter of 0.7 micron and length of 2.1 micron, shown in Figure 11 (top right). The model was generated in 372 ms.

Sorangium cellulosum
Sorangium cellulosum currently holds the record for the largest known bacterial genome at about **13 million** bp [SPK*07], shown in Figure 11 (left). The genome is split into 482 units similar to Escherichia coli. Each unit consists of a 10,000 bp plectoneme and a 16,000 bp unsupercoiled connecting segment. It is enclosed by a rod-shaped membrane, with a diameter of 1.2 micron and 3.6 microns in length. The model was generated in 942 ms.

A more detailed investigation of the runtime, shown in Table 1, displays the individual timings for each generation step. It is interesting to note that the relaxation step for the genome model of the Mycoplasma genitalium requires more time than the one for the larger Escherichia coli. This is due to the fact that the Mycoplasma genitalium model is more crowded, thus making the relaxation more complex. The bottleneck of the performance resides in the relaxation and resampling steps. In order to resolve overlapping beads, the relaxation requires several iterations. For instance, resolving the overlaps for Mycoplasma genitalium models takes around 70 iterations, for Escherichia coli models around 32 iterations, and for Sorangium cellulosum models around 21 iterations. However, it is likely that a thorough parameter fine tuning of the force-based system potentially leads to even faster resolving timings. Our implementation also offers the possibility to run the relaxation progressively during the visualization to mitigate its performance impact. The other bottleneck is the resampling. We have reduced the sequential constraints of the resampling through a divide-and-conquer scheme, where we divide the sequence of beads into subsequences. However, the computation of each subsequence is still processed sequentially, making this the slowest step of the pipeline.

A direct comparison of our parallel generation approach with the previous lattice-based method [GAO18] is shown in Figure 13. The comparison is based on a genome model of Mycoplasma pneumonia. Ten instances of a single circular genome with 44 repeating units of a 17,000 bp plectoneme and a 1000 bp connecting region where modeled, within a spherical space with diameter 380 nm. Distance maps were computed based on the average distance between 10 kbp segments. For both, our parallel approach and the lattice-based method, these maps show the characteristic diamond-shaped features along the diagonal that are observed in Hi-C experiments [TYM*17]. They are a consequence of the proximity of chains within superhelical plectonemes. The weak signal in the off-diagonal area shows that our parallel approach produces models, where the DNA is distributed randomly through space, rather than in local regions.

In order to evaluate the stiffness of our fiber generation approach we compare it to the idealistic model given by Equation 2. We have generated and measured several genome structures of different persistence lengths. Figure 14 shows the corresponding curves for the expected correlation (orange) and the measured correlation (blue).

The constantly changing domain knowledge that influences the generation process of bacterial genomes renders it difficult to compare our parallel approach more explicitly to existing sequential approaches. For this reason, we conducted a simplified comparison of the sequential generation of polymers with our parallel approach, as shown in Figure 15. In the comparison, we generated simple linear polymer structures without supercoils, which were only constrained by the persistence length and the shape of an enclosing compartment. Our parallel approach shows a significant performance improvement starting with 100,000 beads rendering it useful for many applications of instant polymer modeling.

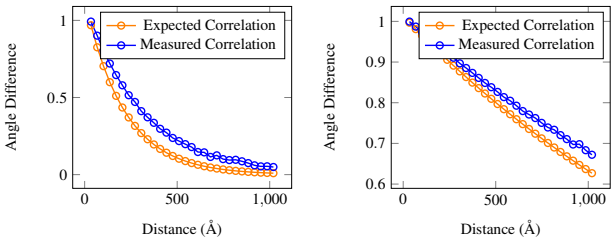


Figure 14: *Comparison of the expected stiffness (orange) and the measured stiffness (blue) of the genome generation approach. The plot on the left shows the curve for a persistence length of* $1,000$ *Å, whereas the one on the right side uses* $10,000$ *Å.*

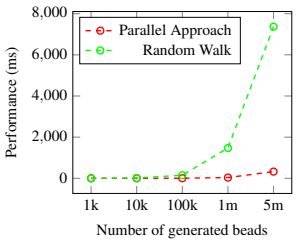


Figure 15: *Performance measurements for random walk and parallel midpoint displacement. Starting at 100,000 beads, the parallel approach indicates a significant performance speedup.*

7. Conclusion and Future Work

We have presented a new method for interactively constructing models of entire bacterial genomes, and validated its application on three bacteria that span the range of natural genome complexity. Looking to the future, there are many enhancements that will need to be approached. Modeling of transcription complexes (RNA polymerase, RNA, co-transcriptional ribosomes) will allow to study in detail the consequences of the genome structure on gene expression. As we move to more complex bacteria and to eukaryotes, methods for including DNA-binding proteins will be essential, since they play a central role in compacting, organizing, and regulating the genetic information. Detailed atomic structures and physicochemical properties are known for most of these additional molecules, so we are optimistic about future enhancement of the method.

ACKNOWLEDGMENTS

This work was funded under the ILLVISATION grant by WWTF (VRG11-010). It is based upon work supported by the King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR) under Award No. OSR-2019-CPF-4108 and BAS/1/1680-01-01. The Scripps Research Institute researchers acknowledge support from the National Institutes of Health under the grant R01-GM120604. This paper was partly written in collaboration with the VRVis Competence Center. VRVis is funded by BMVIT, BMWFW, Styria, SFG and Vienna Business Agency in the scope of COMET - Competence Centers for Excellent Technologies (854174), which is managed by FFG. The authors would like to thank Nanographics GmbH (nanographics.at) for providing the Marion Software Framework.

References

[AJ11] ALTENDORF, HELLEN and JEULIN, DOMINIQUE. "Random-walk-based stochastic modeling of three-dimensional fiber systems". *Phys. Rev. E* 83 (4 2011), 041804 3, 6.

[B M83] B. MANDELBROT, BENOIT. "The Fractal Geometry of Nature". Vol. 51. 1983, 468 p. ISBN: 0716711869 3.

[BWC90] BOLES, T. CHRISTIAN, WHITE, JAMES H., and COZZARELLI, NICHOLAS R. "Structure of plectonemically supercoiled DNA". *Journal of Molecular Biology* 213.4 (1990), 931–951. ISSN: 0022-2836 2.

[CJB08] CODLING, EDWARD, J PLANK, MICHAEL, and BENHAMOU, SIMON. "Random walks in biology". *Journal of the Royal Society, Interface / the Royal Society* 5 (2008), 813–34 3.

[DH53] D WATSON, JAMES and H CRICK, FRANCIS. "Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid". *The American journal of psychiatry* 160 (1953), 623–4 2.

[DN04] DECAUDIN, PHILIPPE and NEYRET, FABRICE. "Rendering Forest Scenes in Real-time". *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques*. EGSR'04. Eurographics Association, 2004, 93–102. ISBN: 3-905673-12-6 1.

[Dor13] DORMAN, CHARLES. "Genome architecture and global gene regulation in bacteria: Making progress towards a unified model?": *Nature reviews. Microbiology* 11 (2013) 2.

[DT16] DAME, REMUS T and TARK-DAME, MARILIIS. "Bacterial chromatin: converging views at different scales". *Current Opinion in Cell Biology* 40 (2016), 60–65. ISSN: 0955-0674 2.

[FFC82] FOURNIER, ALAIN, FUSSELL, DON, and CARPENTER, LOREN. "Computer Rendering of Stochastic Models". *Commun. ACM* 25.6 (1982), 371–384. ISSN: 0001-0782 3.

[GAO18] GOODSELL, DAVID S., AUTIN, LUDOVIC, and OLSON, ARTHUR J. "Lattice Models of Bacterial Nucleoids". *The Journal of Physical Chemistry B* 122.21 (2018), 5441–5447 2, 3, 9, 10.

[Hag88] HAGERMAN, PAUL J. "Flexibility of DNA". *Annual Review of Biophysics and Biophysical Chemistry* 17.1 (1988), 265–286 2.

[HF13] HORNUS, SAMUEL and FOURMENTIN, ERIC. "Easy DNA Modeling and More with GraphiteLifeExplorer, in "PLoS ONE". *doi : 10.1371/JOURNAL.PONE.0053609], http://hal. inria.fr/hal-00924190*. 2013 2.

[HLE17] HACKER, WILLIAM C., LI, SHUXIANG, and ELCOCK, ADRIAN H. "Features of genomic organization in a nucleotide-resolution molecular model of the Escherichia coli chromosome". *Nucleic Acids Research* 45.13 (2017), 7541–7554 2.

[Hoe14] HOETZLEIN, RAMA. "Fast Fixed-Radius Nearest Neighbors: Interactive Million-Particle Fluids". *GPU Technology Conference (GTC) 2014*. 2014 6.

[HRRG08] HAN, CHARLES, RISSER, ERIC, RAMAMOORTHI, RAVI, and GRINSPUN, EITAN. "Multiscale Texture Synthesis". *ACM Trans. Graph.* 27.3 (2008), 51:1–51:8. ISSN: 0730-0301 1.

[JKL12] JILESEN, JONATHAN, KUO, JIM, and LIEN, FUE-SANG. "Three-dimensional midpoint displacement algorithm for the generation of fractal porous media". *Computers & Geosciences* 46 (2012), 164–173. ISSN: 0098-3004 3.

[KAK*18] KLEIN, TOBIAS, AUTIN, LUDOVIC, KOZLÍKOVÁ, BARBORA, et al. "Instant Construction and Visualization of Crowded Biological Environments". *IEEE Transactions on Visualization and Computer Graphics* 24.1 (2018), 862–872. ISSN: 1077-2626 3, 9.

[Kel58] KELLENBERGER, EDOUARD. "Electron Microscope Study of DNA-Containing Plasms: II. Vegetative and Mature Phage DNA as Compared with Normal Bacterial Nucleoids in Different Physiological States". *The Journal of Cell Biology* 4 (1958), 671–678 2.

[KFS*14] KLECKNER, NANCY, FISHER, JAY K., STOUF, MATHIEU, et al. "The Bacterial Nucleoid: Nature, Dynamics and Sister Segregation". *Current Opinion in Microbiology* 22 (2014) 2.

[KP49] KRATKY, OTTO and POROD, GÜNTHER. "Röntgenuntersuchung gelöster Fadenmoleküle". *Recueil des Travaux Chimiques des Pays-Bas* 68.12 (1949), 1106–1122. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/recl.19490681203 5.

[KPV*14] KOLESAR, IVAN, PARULEK, JULIUS, VIOLA, IVAN, et al. "Interactively illustrating polymerization using three-level model fusion". *BMC Bioinformatics* 15.1 (2014), 345. ISSN: 1471-2105 2.

[LBLH19] LINDOW, NORBERT, BAUM, DANIEL, LEBORGNE, MORGAN, and HEGE, HANS-CHRISTIAN. "Interactive Visualization of RNA and DNA Structures". *IEEE Transactions on Visualization and Computer Graphics* 25.1 (2019), 967–976. ISSN: 1077-2626 3.

[MAPV15] MUZIC, MATHIEU LE, AUTIN, LUDOVIC, PARULEK, JULIUS, and VIOLA, IVAN. "cellVIEW: a Tool for Illustrative and Multi-Scale Rendering of Large Biomolecular Datasets". *Eurographics Workshop on Visual Computing for Biology and Medicine*. EG Digital Library. The Eurographics Association, 2015, 61–70. ISBN: 978-3-905674-82-8 3, 4, 8, 9.

[Max85] MAX, NELSON. "DNA Animation from atom to chromosome". *Journal of Molecular Graphics* 3.2 (1985), 69–71. ISSN: 0263-7855 2.

[MLS17] MIRAVET-VERDE, SAMUEL, LLORÉNS-RICO, VERÓNICA, and SERRANO, LUIS. "Alternative transcriptional regulation in genome-reduced bacteria". *Current Opinion in Microbiology* 39 (2017), 89–95. ISSN: 1369-5274 4.

[MM89] MUSGRAVE Forest K.and Kolb, CRAIG E. and MACE, ROBERT S. "The Synthesis and Rendering of Eroded Fractal Terrains". *SIGGRAPH Comput. Graph.* 23.3 (1989), 41–50. ISSN: 0097-8930 3.

[NML00] NORROS, ILKKA, MANNERSALO, PETTERI, and L. WANG, JONATHAN. "Simulation of Fractional Brownian Motion with Conditionalized Random Midpoint Displacement". *Advances in Performance Analysis* 2 (2000) 3.

[PM10] PETERS, JUSTIN P. and MAHER, L. JAMES. "DNA curvature and flexibility in vitro and in vivo". *Quarterly reviews of biophysics* 43 (2010), 23–63 2.

[RZ14] ROSA, ANGELO and ZIMMER, CHRISTOPHE. "Chapter Nine - Computational Models of Large-Scale Genome Architecture". *New Models of the Cell Nucleus: Crowding, Entropic Forces, Phase Separation, and Fractals*. Ed. by HANCOCK, RONALD and JEON, KWANG W. Vol. 307. International Review of Cell and Molecular Biology. Academic Press, 2014, 275–349 2.

[SHGO11] SENGUPTA, SHUBHABRATA, HARRIS, MARK, GARLAND, MICHAEL, and OWENS, JOHN. "Efficient Parallel Scan Algorithms for GPUs". 2011, 413–442. DOI: 10.1201/b10376-29 8.

[SJJ95] S. DUNCAN, BRUCE, J. MACKE, TOM, and J. OLSON, ARTHUR. "Biomolecular visualization using AVS". *Journal of molecular graphics* 13 (1995), 271–82, 299 2.

[SKK*14a] STEINBERGER, MARKUS, KENZEL, MICHAEL, KAINZ, BERNHARD, et al. "On-the-fly Generation and Rendering of Infinite Cities on the GPU". *Computer Graphics Forum* 33.2 (2014), 105–114. ISSN: 0167-7055 1.

[SKK*14b] STEINBERGER, MARKUS, KENZEL, MICHAEL, KAINZ, BERNHARD, et al. "Parallel Generation of Architecture on the GPU". *Comput. Graph. Forum* 33.2 (2014), 73–82. ISSN: 0167-7055 1.

[Spi01] SPITZER, FRANK. *Principles of Random Walk*. Graduate texts in mathematics. Springer, 2001. ISBN: 9780387951546 3.

[SPK*07] SCHNEIKER, SUSANNE, PERLOVA, OLENA, KAISER, OLAF, et al. "Complete genome sequence of the myxobacterium Sorangium cellulosum". *Nature biotechnology* 25 (2007), 1281–9 10.

[TYM*17] TRUSSART, MARIE, YUS, EVA, MARTINEZ, SIBOLESRA, et al. "Defined chromosome structure in the genome-reduced bacterium Mycoplasma pneumoniae". English (US). *Nature Communications* 8 (2017). ISSN: 2041-1723 2, 10.

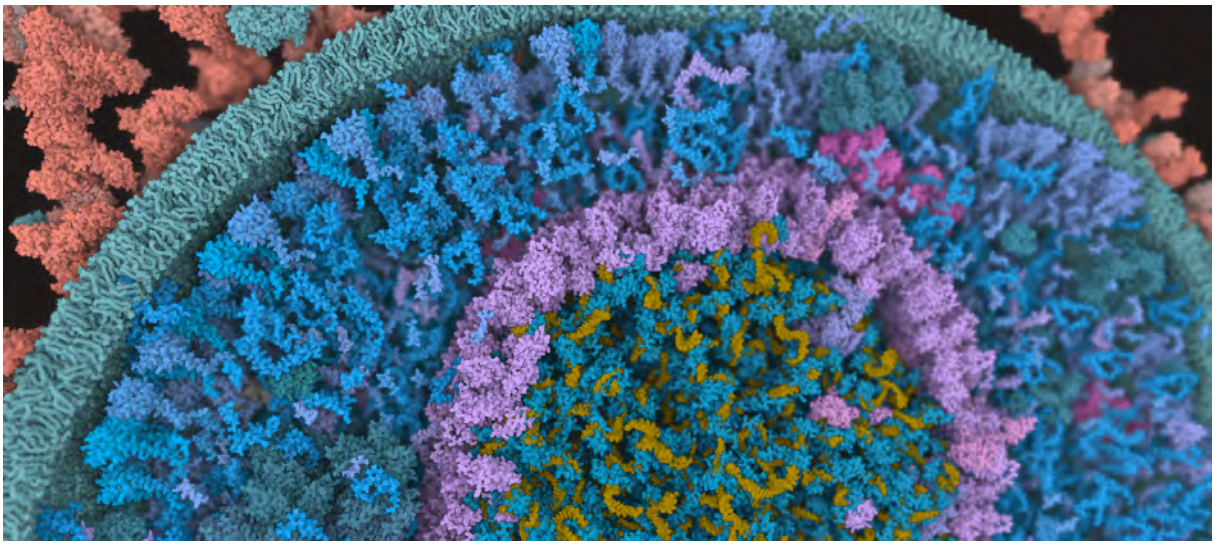
[WJZL08] WANG, WENPING, JÜTTLER, BERT, ZHENG, DAYUE, and LIU, YANG. "Computation of Rotation Minimizing Frames". *ACM Trans. Graph.* 27.1 (2008), 2:1–2:18. ISSN: 0730-0301 8.

CHAPTER 5

Paper B - Instant Construction and Visualization of Crowded Biological Environments

Instant Construction and Visualization of Crowded Biological Environments

Tobias Klein, Ludovic Autin, Barbora Kozlíková, David S. Goodsell, Arthur Olson, M. Eduard Gröller, and Ivan Viola



Abstract— We present the first approach to integrative structural modeling of the biological mesoscale within an interactive visual environment. These complex models can comprise up to millions of molecules with defined atomic structures, locations, and interactions. Their construction has previously been attempted only within a non-visual and non-interactive environment. Our solution unites the modeling and visualization aspect, enabling interactive construction of atomic resolution mesoscale models of large portions of a cell. We present a novel set of GPU algorithms that build the basis for the rapid construction of complex biological structures. These structures consist of multiple membrane-enclosed compartments including both soluble molecules and fibrous structures. The compartments are defined using volume voxelization of triangulated meshes. For membranes, we present an extension of the Wang Tile concept that populates the bilayer with individual lipids. Soluble molecules are populated within compartments distributed according to a Halton sequence. Fibrous structures, such as RNA or actin filaments, are created by self-avoiding random walks. Resulting overlaps of molecules are resolved by a forced-based system. Our approach opens new possibilities to the world of interactive construction of cellular compartments. We demonstrate its effectiveness by showcasing scenes of different scale and complexity that comprise blood plasma, mycoplasma, and HIV.

Index Terms—Interactive modeling, population, biological data, interactive visualization

1 Introduction

Technological advances in structural biology, proteomics, and biophysics, combined with the rapid advance of computational capabilities, have opened the door to studying increasingly large and complex biological systems. Recently, we have seen a shift from studying individual proteins to modeling and analyzing functional protein assemblies and even larger systems, such as viruses, bacteria, and portions of eukaryotic cells. These are often denoted as biological mesoscale structures, representing an intermediate scale between molecular and cellular biology. On the molecular (nanoscale) level, cells are built of proteins, nucleic acids, lipids, and polysaccharides. The mesoscale level reveals how these molecules are assembled into more complex subcellular environments that orchestrate the processes of life. Given the complexity of these massive mesoscale environments, modeling must be tightly coupled with a proper visual representation. This provides intuitive feedback for their validation, tools for exploration of the models, and visual materials for public dissemination and to communicate findings to peers. In typical scenarios, these visualizations are created by scientific illustrators who traditionally produce images by hand drawing or by using 3D modeling tools. Examples include our illustrations of cellular environments [15], and a detailed 3D model of a synaptic bouton created in Maya [47]. This is a laborious process with limited interaction and exploration capabilities, which also requires non trivial domain knowledge about the depicted biological structures.

A more general solution is to describe the scene on a higher level of abstraction and use this description to automatically create the resulting scene. This approach has recently been implemented in cellPACK [24]

- *Tobias Klein and Ivan Viola are with TU Wien, Austria. E-mails: tklein@cg.tuwien.ac.at, viola@cg.tuwien.ac.at.*
- *M. Eduard Gröller is with TU Wien and VRVis Research Center, Austria. E-mail: groeller@cg.tuwien.ac.at.*
- *Ludovic Autin, David S. Goodsell, and Arthur Olson are with The Scripps Research Institute, California, USA. E-mails: ludovic.autin@gmail.com, goodsell@scripps.edu, olson@scripps.edu.*
- *Barbora Kozlíková is with Masaryk University, Brno, Czech Republic. E-mail: kozlikova@fi.muni.cz.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

to generate and visualize 3D models of complex biological environments. CellPACK integrates data from multiple sources into a 'recipe' that describes the constitution of the desired environment. The environment is partitioned into membrane-bounded compartments, often arranged hierarchically. The compartments are defined by mesh surfaces that represent the lipid-bilayer membranes. Each compartment, including membranes, can be populated by a different set of ingredients, which include soluble molecules inside the compartment and molecules embedded in the membranes. These ingredients can be fibrous components, such as RNA or carbohydrate chains, soluble components such as proteins and metabolites, or lipids in membranes. CellPACK combines the knowledge about membranes, compartments, and ingredients, and then assembles a mesoscale model with respect to mutual biologically-relevant interactions between them.

The main drawback of this approach is the computation time for assembling the 3D mesoscale model from the input data and the recipe. Depending on the desired quality of the distribution of ingredients, this stage spans from minutes to hours. Therefore, it is often a lengthy process to, for example, change parameters like the number of ingredients (molarity) or their interactions and observe the impact of the change. Our aim is to overcome this limitation and to instantly provide distributions of various ingredients within biological compartments and on membranes. This will open new possibilities for biologists to gather domain-specific knowledge about mesoscale-level structures, integrate it into a model, and explore the consequences of gaps in knowledge or ranges of experimentally-observed parameters.

Another irreplaceable part of the whole process is the visual representation of the scene and its content. Molecular visualization techniques have been developed in the last decades, which predominantly focus on visual depictions of molecules captured by X-ray crystallography, NMR spectroscopy, and more recently, electron microscopy. Due to the nature of this structural data, the molecules may be visualized at atomic resolution. However, the mesoscale level poses new challenges. Mesoscale models are larger, with millions of atoms, taxing existing visualization hardware and software. They are also more complex, often integrating multiple types of data at different resolutions, and with complex hierarchical relationships that span from individual molecules to entire cells. Visual representations must provide the user with an overview of the structures as well as with means to explore their details. To address this challenge, our state-of-the art tool cellVIEW [36] enables a seamless transition between visual representations of structures on different levels of detail. CellVIEW integrates the latest GPU-based algorithms from computer graphics and visualization to interactively render large biological scenes.

The approach described here incorporates instant modeling into cellVIEW to provide a unified visual framework for modeling and for visualization. The main contributions include:

- New algorithms for instant modeling of different types of compartments and membranes of biological structures on the mesoscale level.
- An interactive visual environment that allows users to change the population parameters and instantly explore the resulting scene on both nanoscale and mesoscale levels.
- Enhancing the process of creating models of mesoscale structures by shifting it from a non-visual and non-interactive environment to a fully interactive and visually-supported one.

We demonstrate the approach with mesoscale models of different scale and complexity, including blood plasma, HIV, and an entire mycoplasma bacterium.

2 Related Work

Experimental methods to determine mesoscale structures are extending to larger and larger subjects, but currently fall short of being able to determine the atomic structure of living cells. For instance, cryo-electron microscopy (EM) can give a detailed structure of the regular envelope of the Zika virus, but the more randomly arranged genome inside is still largely inaccessible to experiments [40]. Similarly, cryo-EM tomography is currently yielding detailed views of complex enveloped viruses, bacteria, and eukaryotic cells, but are typically able to resolve only large molecules such as ribosomes, cytoskeletal elements, and membranes [2].

Many groups are attempting the structural modeling of the mesoscale with atomic detail, using computation to combine information from mesoscale experimental techniques with atomic structural information on the components [22, 38]. Much of the work in structural mesoscale modeling has focused on the soluble components of bacterial cytoplasm, using models to explore diffusion in crowded environments [35]. More recently, several groups are building tools for full structural models that include cellular infrastructure, including our own work with cellPACK [24] and LIFEEXPLORER [21, 43].

Procedural Modeling. The mesoscale structure of cells is similar to other natural phenomena with high geometric and visual complexity as well as with requirements for explicit control of details. Other examples include the modeling of clouds and meadows covered by grass. The general approach to model these types of phenomena is to first define the overall shape of the environment, then fill it in with detailed objects using a stochastic process. This approach, called procedural modeling, has a long tradition in computer graphics, for example, for on-the-fly generation of infinite cities [16] or forests [7] within rich game environments. Currently the focus is rather on parallelization and scheduling optimizations, so that the procedural content generation is efficiently realized on graphics hardware [4, 41]. Wonka et al. [49] presented a method for the automatic modeling of architecture. Recently, a novel technique was proposed for viewpoint-guided generation of high geometric details on facades [30]. These techniques typically use certain plans or grammars, which provide structural rules for the generation of shapes. In our work, we focus rather on the distribution of objects where the space is already constrained by membranes and divided into compartments.

Visualization of Large Biological Scenes. Driven by the wealth of structural information available through sources like the Protein Data Bank [3], highly effective software is currently available for the visualization of biomolecules and assemblies. For recent reviews, see Kozlikova et al. [29] and Johnson and Hertig [27]. These methods typically fail when faced with the large number of mesoscale molecules. Mesoscale structures require specialized, highly-optimized solutions. Currently, there are two publicly-available open-source systems where large numbers of atoms can be rendered efficiently by leveraging graphics hardware capabilities. One system is MegaMol [17], which is useful for various atomic representations, not only biomolecules. The other one is cellVIEW [36], a tool for illustrative multiscale visualization of large biomolecular datasets. An alternative approach, published by Waltermate et al. [45], introduces an interactive tool combining mesoscopic and molecular scale visualization. They reach this by using a magnifier tool that enables the user to select a region on the cell membrane and map a pre-computed membrane patch with atomistic resolution onto it. This approach is able to populate only a very limited area of the cell surface.

Mesoscale scenes are also often produced by scientific illustrators using general-purpose 3D modeling tools. To ease their tasks, several extensions specific to molecular modeling have been introduced. For example, our ePMV plugin [25] allows users to run molecular modeling software directly inside of professional 3D animation applications. SketchBio [44] is another example of a 3D interface for molecular modeling and animation. Lv et al. [33] explored the possibility to use game technology in biomolecular visualization, using the Unity3D game engine to develop and prototype a molecular visualization application for subsequent use in research or education.

Texture Synthesis. Our approach to populate large biological membranes with lipids is based on a tiling technique commonly used in seamless 2D texture synthesis. We adapt Wang Tiles [46], which utilize square tiles to generate a plane tiling. The positioning of tiles on the plane is done according to their face-color rules to compose a desired pattern. Recently, several extensions of this basic approach have been published. Most of them were designed for synthesizing 2D textures

that are mapped onto 3D surfaces. Fu and Leung [13] applied Wang Tiles to arbitrary topological surfaces. Our case differs in the necessity to synthesize tiles containing 3D objects, such as lipids. Therefore, the detection and handling of overlaps has to take these objects into account. Here we were inspired by the solution provided by LipidWrapper [9]. LipidWrapper solves the collisions for each triangle on the 3D mesh, which makes it unsuitable for large membranes. Neyret and Cani [37] presented another strategy to tiling, using triangles with homogeneous textures to tile surfaces. Culik and Kari [5] introduced Wang Cubes, which extend the Wang Tiles approach and are able to create non-periodic illustrative 3D patterns and textures. Lu et al. [32] presented a framework for volume illustration utilizing Wang Cubes. They extend the original idea and modify it for multipurpose tiling. The Wang Cubes are not applicable in our case. Although we deal with 3D tiles representing the lipid bilayer, the tiles are still defined by a 2D plane. Fleischer et al. [12] presented an approach to texturing surfaces with so called cellular textures. These are textures represented by more complex geometry, which can cover the surface with small-scale features, such as feathers or thorns. This is very close to our situation with lipid bilayers. However, cellular textures cannot be applied in our case because of the regularity of the produced surfaces and the algorithm speed.

Population with Fibrous Structures. Algorithms for self-avoiding walks may be used to populate a compartment with a fibrous structure. Fan et al. [10] published an extension of a Manhattan lattice to generate fibrous structures by going from 2D to 3D space. In our approach we execute a procedural building of fibrous structures, which is similar to the polymerization process addressed by Kolesar et al. [28]. Their illustrative approach is a fusion of three different modeling techniques, i.e., L-systems, agent-based systems, and systems of densities. Adding new building blocks to the existing polymer is based on attraction forces. Gruenert et al. [18] presented an approach to rule-based spatial modeling of chemical reaction systems. The technique is also suitable for simulating the polymerization process, showcased on the growth of filaments. The growth is driven by parameters, such as torsion and bending, that control the pathways and structures of formed complexes.

The problem of procedurally constructing paths is common in other areas as well. As an example, Galin et al. [14] presented an automatic method for generating roads between two given points, based on a weighted anisotropic shortest path algorithm. The result path minimizes a cost function influenced by different features of the scene, such as terrain shape and obstacles.

3 Overview

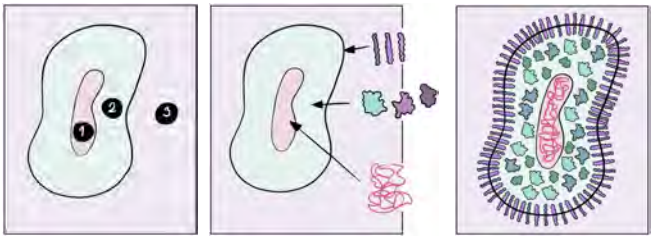
When modeling scenes containing complex mesoscale biological structures, the workflow consists of the following basic steps:

1. Scene organization, to define the shape of each membrane-bounded compartment.

2. Recipe definition, to determine the molecular composition of each compartment and membrane.

3. Population of the model, to fill the soluble spaces and membranes with ingredients defined in the recipe.

The organization of the scene is defined by closed 3D meshes that represent the lipid bilayer membranes. The membranes specify the separate compartments of the scene. For example, Figure 1 (a) shows a scene with two concentric closed membranes defining two intracellular environments (labeled 1 and 2) and a surrounding environment (labeled 3). The following step is to compile a recipe that defines the molecular composition of these compartments and membranes (Figure 1 (b)). This recipe includes the structures of the molecules, their concentration (molarity), and constraints related to their position in the membranes, specific interactions with other molecules, etc. The last step takes the recipe and populates the compartments and membranes accordingly (see Figure 1 (c)). This population step is typically stochastic, so it may be repeated to provide an ensemble of similar models that are each consistent with the scene organization and recipe.



(a) (b) (c)

Fig. 1: (a) The hierarchy is provided by compartments that organize the scene (here marked as 1, 2, and 3). (b) For each individual compartment, their composition is defined including the type and number of ingredients. (c) Our algorithms automatically populate the compartments and their surfaces with the information about the ingredients.

With currently available approaches, such as cellPACK [24], the process of transforming the scene definition and recipe to the final model can take from minutes to hours, depending on the complexity of the scene. This makes the modeling process non-interactive as the user cannot immediately see the consequences of parameter changes. We overcome this by applying parallel processing, to improve computational bottlenecks of the workflow.

As can be seen in Figure 1 (b), the individual compartments and membranes can be populated in several ways. We can distinguish three basic cases:

- **Membranes** represent a semi-permeable barrier between two compartments. They often consist of a lipid bilayer with embedded proteins used in the communication and transport of molecules and ions across the membrane. In order to model membranes, we have implemented a solution based on texture tiling.

- **Soluble components** occupy the free space in each compartment. They consist of ingredients (molecules) of different shape and complexity, which influences the difficulty of the population step inside of each predefined space. Our approach for positioning ingredients in compartments consists of two steps. The first step populates the spaces with ingredients without considering overlaps between them. Collisions are resolved in the second step.

- **Fibrous structures** such as DNA, RNA, or actin filaments, are long strands folded inside specific compartments. We use a self-avoiding random walk to construct fibers in the allowable regions of the compartments.

Algorithms for populating spaces with each of these types of ingredients, and the advantages of populating them in a specific order, are described in detail in the following sections.

4 Membrane Population

Biological membranes are built around a continuous fluid sheet containing two opposed layers of lipids. Lipids are small molecules consisting of several dozen atoms, with a characteristic chemical character. They are composed of a head group that is soluble in water, and one or more tails that are insoluble. In water, they spontaneously associate to hide the insoluble tails, forming some of the largest structures in cells. These membranes form the semi-permeable boundary that surrounds cells, as well as the boundary of inner compartments such as the nucleus or mitochondria. Different types of lipids may be distributed unequally to create an asymmetry between the outer and inner surfaces, which is important for the cell function. For each biological structure, such as the HIV virion or the red blood cell, the bilayer has a unique lipid composition. The common property is that lipids are densely packed on the membrane and are oriented roughly perpendicularly to it.

The modeling of lipid bilayers is a complex and laborious task, because of the dense packing and the large number of lipids compared to other surrounding molecules. The main challenge, in terms of modeling, is to incorporate the known structural information and to avoid repetitive patterns on the surface. Several bilayer-modeling programs are available, such as CHARMM-GUI [23], Packmol [34], or cellPACK [24]. These tools position individual lipids one-by-one on the membrane, which leads to unacceptable computational costs. A more sophisticated solution is provided by LipidWrapper [9]. It extracts whole patches from a pre-equilibrated planar membrane model. These patches contain the detailed structural information about the lipids. In order to produce the membrane surface, LipidWrapper uses a triangulated mesh as input, where each triangle patch is randomly cut out from the provided membrane model. However, the edges of adjacent patches on the surface do not fit together and lipids potentially overlap in these regions. LipidWrapper deletes overlapping lipids and fills the resulting holes with new lipids. This is an expensive process since it must be done for every triangle edge on the surface. Because of this, the computational cost of LipidWrapper is significant and not applicable to instant modeling.

For the interactive population of membranes, our solution uses a tiling approach from texture synthesis to cover the membrane mesh with lipids. In our approach, we require the mesh to be defined as a quad-based surface map. In general, any method can be used that resamples a given mesh into a quad-based surface if the resulting surface map is conformal and has low area distortions. An example can be PolyCubeMaps introduced by Tarini et al. [42]. The mesh generated by this approach is only marginally affected by distortions, which makes it suitable for our tiling approach.

To cover the mesh with lipids, we use only a small number of tiles. These tiles are pre-populated with lipids and here we use the basic principle of LipidWrapper. Our solution is significantly faster because we generate the content for only a limited set of tiles instead of applying it onto the whole mesh geometry. The tiles are subsequently used for covering the membrane. They have to be placed in such a way that their edges fit together and that the tiling will not result in periodic patterns. As the repetition pattern should not be visible, we use Wang Tiles [46].

In the following, we describe individual steps of our approach in detail. We start with the description of the concept and principle of Wang Tiles. Then we present our adaptation of this concept to membrane meshes.

4.1 Wang Tiles and their Extension to Membranes

The Wang tiling concept was introduced in the early '60s and is well-known in texture synthesis. A Wang Tile is defined as a square with color-encoded edges. A set of Wang Tiles can be used to cover a 2D plane without periodic patterns. The colors of edges restrict how the tiles can be placed during the tiling process. The tiling is valid only if shared edges have the same color. At least four colors are required to tile a plane non-periodically.

There are several approaches that extend the concept of Wang Tiles to 3D space. In our solution, we follow the approach of Fu and Leun [13], which applies the tiling concept of Wang Tiles to arbitrary topological surfaces. They generate each tile from four different diamond-shaped input patches that are positioned west, east, north, and south around the tile center, respectively (Figure 2). These patches are combined to generate a set of tiles with different colors on the edges. We adapt this approach for use in membrane modeling by synthesizing a collection of lipid-textured tiles and then mapping them onto the membrane mesh.

4.1.1 Tile Synthesis for Lipid Bilayer Models

Synthesis of the tiles is the major challenge to generating 3D models of biological membranes. First, we have to extract the content for the tiles from an input texture. For biological membranes, this input texture is a planar representation of the lipid bilayer containing non-overlapping, tightly-packed lipid molecules. Atomic coordinates for the lipid bilayer can be generated, for example, by a molecular dynamics simulation using the Amber force field for lipids [8]. The tiles are generated in several steps. First, four small diamond-shaped patches

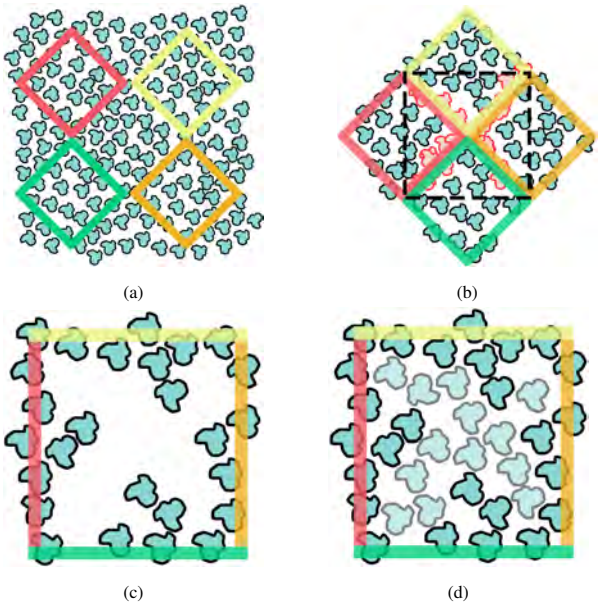


Fig. 2: Overview of the Wang Tile synthesis. (a) Four small patches are extracted from the input texture. (b) A large patch is assembled from the four small patches, and then the tile is extracted from the center of the patch. (c) Bordering lipids with collisions are removed, leaving holes in the tile. (d) Holes are filled by positioning new lipids.

are extracted from the large bilayer texture. When extracting these patches (Figure 2 (a)), we include only lipids whose center is inside the bounds of the specified patch, where the center of a lipid is defined as the center of its bounding box. A large diamond-shaped patch is then created by positioning four of these tiles together. Finally, we extract a rectangular tile from the large diamond-shaped patch (see Figure 2 (b)). The different tiles needed for Wang Tiling are generated by different arrangements of the small patches into the large patches. The tile has to be further processed to resolve overlaps and fill gaps.

Detection of Overlapping Lipids The assembly of a tile from multiple input patches inherently leads to overlapping lipids in areas where the input patches meet. Collisions are evaluated in a two step process to improve performance. First the overlap between bounding boxes is tested, then if necessary individual lipid atoms are checked. The lipid with the highest number of collisions is removed from the tile. If multiple lipids have the same number of collisions, one is chosen randomly for removal. The number of collisions is then updated for all neighbors of the removed lipid, and the procedure is repeated until all collisions are eliminated. This leaves the tile with undesirable holes (Figure 2 (c)), which are filled in the final step.

Hole Filling The holes in a tile are filled by non-colliding lipids (Figure 2 (d)). Individual lipids are extracted from the same input membrane bilayer that was used to generate patches for tiles.

For each side of the bilayer, a plane is defined that intersects the hydrophilic head groups of the lipids in the tile. Trial lipids are randomly chosen from the input texture and placed at regular intervals along the diagonals of the tile, aligning their head groups within the planes. If there are no collisions, the lipid is added to the tile, otherwise, the lipid is rotated around an axis perpendicular to the plane and tested for collisions. If no acceptable orientations are found, the process continues with another lipid from the input texture. Additionally, we reduce the number of possible candidates by rejecting those that already intersect with the lipids from the tile.

The result of the hole filling process depends on three parameters: how many lipids do we have in the testing set, how many rotations of lipids we test, and how many times we repeat the entire hole filling procedure. These parameters can be adjusted according to user needs.

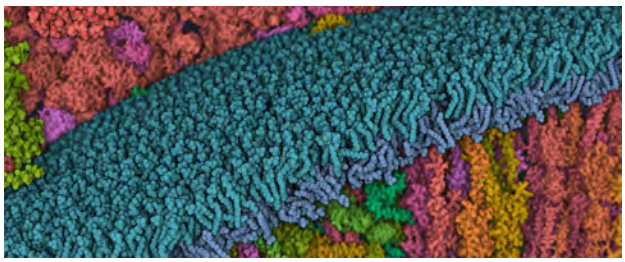


Fig. 3: Example of lipid-bilayer membrane populated by lipid tiles. This cross-sectional view is visualized using a cutting plane.

As we apply the process only on the tiles and not on every quad of the mesh, the parameter setting is not a critical aspect.

4.2 Tile Mapping

Finally, the generated tiles are mapped onto the membrane mesh consisting of quads. Since we are using data from a bilayer lipid membrane instead of textures, the 3D coordinates are mapped onto the quads instead of 2D texture coordinates. For this reason, we use quadrilateral coordinates to map lipid centers (λ, μ) to the quad with the vertices (v_0, v_1, v_2, v_3) through the following equation, where p is the resulting vertex in 3D space.

$$p = (1-\lambda)(1-\mu)v_0 + \lambda(1-\mu)v_1 + \lambda\mu v_2 + (1-\lambda)\mu v_3 \quad (1)$$

At each resulting position, the corresponding lipid is instantiated. The orientation of the lipids on the quad has to be adjusted as well. As the main axis of the lipids is always perpendicular to the plane of the tile, we determine their orientation by computing the rotation angle between the original tile and its position on the quad. This rotation is applied to all lipids of a given tile. Figure 3 shows the result after positioning the tiles with lipids on the membrane mesh.

5 Soluble and Membrane-Bounded Components

Cells are typically crowded with proteins, nucleic acids, and other molecules. Soluble components and assemblies fill the compartments. Membrane-bounded components are embedded in the lipid bilayers. The recipe for a mesoscale scene specifies the quantity of these ingredients, their locations and orientations (if appropriate), and their mutual interactions. The task of population boils down to spatially distribute these ingredients in the corresponding space avoiding overlaps. Currently available techniques are capable of positioning soluble ingredients inside compartments, however, only in a sequential manner. This means that they position ingredients one by one. After positioning each ingredient the surrounding space is updated so it is aware of the distance to the closest ingredient. The next ingredient can only be positioned at locations with a sufficient distance to the nearby ingredients. The computational cost of the sequential approach is not acceptable for interactive environments. Therefore, our solution uses parallel processing to increase the performance.

To make the procedure applicable to GPU implementation, we divide our approach into three consecutive steps (see Figure 4):

1. Compartment space organization using voxelization.
2. Populating the space with the given ingredients.
3. Detecting and resolving collisions between ingredients.

The major advantage of this approach is that the processing of the individual ingredients becomes mostly independent from the processing of other ingredients. In the rest of this section we describe the steps in more detail.

(a) (b) (c)

Fig. 4: The three steps of our approach to populate compartments with soluble components. (a) Voxelization of the space and definition of inner grids (purple and grey), surface grids (dark purple and dark blue), and outer grid (light blue). (b) Positioning of individual ingredients inside the inner compartments. Collisions between ingredients are highlighted. (c) Scene after resolving collisions using a force-based approach.

5.1 Step 1 – Space Partitioning using Voxelization

The first step is to partition the scene into a set of compartments, delimited by membrane meshes. Our approach is based on the state-of-the-art method of GPU voxelization by Schwarz and Seidel [39]. The voxelization process starts with the classification of voxels that intersect with the compartments. In one of the grid axes, a scanline algorithm is applied to all voxels to determine if they belong to cellular compartments, membranes, or to the surrounding environment. We assign a negative value to cellular compartment voxels, a positive value to surface voxels, and zero to outside voxels. We repeat this process for all compartments in the scene. The resulting structure, which we call a **compartment grid**, identifies the compartment or membrane associated with each voxel, for the rapid population in the following step. Moreover, we can sort the voxels per compartment and estimate the volume for each of them. Thus, given a molarity we can calculate the proper count for a specific protein.

In addition to the compartment grid, we define an **occupancy grid** that helps to resolve some of the overlaps between ingredients. The occupancy grid is of the same size and resolution as the compartment grid. It holds negative values for empty voxels or the corresponding ID of the ingredient if the voxel is occupied. This is especially important if the size of ingredients differs significantly, e.g., if surface ingredients protrude from the compartment surface. For this reason, we start the population process by distributing the large surface ingredients and then use the remaining space for the interior ones by employing the updated occupancy grid. This divides the population process into two passes, but prevents small ingredients from being completely overlapped by larger ingredients coming from a different compartment.

5.2 Step 2 – Population

In this step we populate a given compartment with individual ingredients defined in the recipe, using the compartment grid and occupancy grid generated in the previous step. Here we distinguish between two types of ingredients to be populated. The first type are ingredients that occupy the inner part of a compartment but are bounded to the compartment membrane. The second type of ingredients populates the soluble space of the compartment. In order to avoid additional overlaps, we first populate the surface of the compartment and then its inner part.

5.2.1 Population of Compartment Surface

Cell membranes typically include a diverse collection of membrane-bounded proteins, which may interact just on one side of the membrane or extend through the membrane. This poses several challenges: the proteins must be oriented correctly, must face the proper direction, and their membrane-spanning portions must be embedded in the membrane. These proteins often have large portions extending from one or both sides of the membrane, which must be evaluated for collisions with other membrane-bounded proteins and soluble components. In order to spatially distribute ingredients on a given membrane, we need

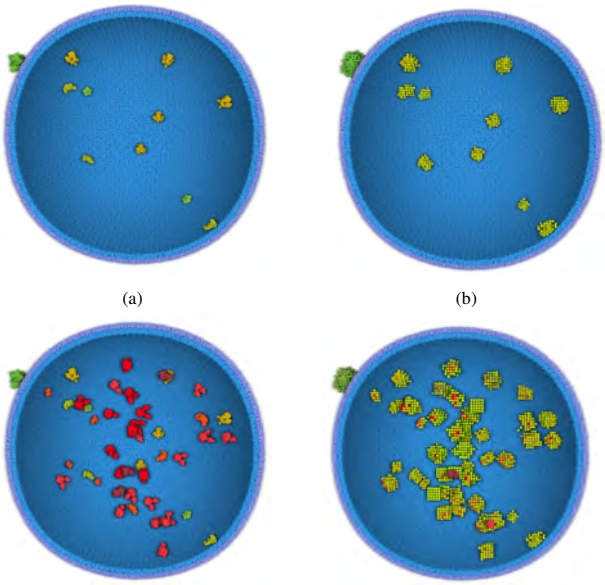


Fig. 5: (a) Ingredients distributed on the surface of a compartment. (b) Occupancy grid containing information about the voxels occupied by these ingredients (green for outside, yellow for inside). (c) Positioning ingredients inside the compartment (new ingredients have red and orange color). (d) Occupancy grid update.

the information about the ingredient type, its molarity, the ID of the membrane, and the principal vector and offset of the ingredients (Figure 5 (a)). Then we update the information stored in the occupancy grid (Figure 5 (b)).

The ingredients should be randomly distributed and should not overlap. To solve this issue, Willmott previously used the property of a Halton sequence to distribute in real time non-overlapping objects in a plane [48]. Similar to his approach, we use a Halton sequence to select positions on the mesh of the membrane to place the ingredients.

In general, a Halton sequence [19] provides point sets with low discrepancy, i.e., it produces well-spaced samples. The sequence is constructed using a prime number that defines the number of divisions of a unit interval into sub-intervals. These are subsequently divided using the same prime number until the desired length of the sequence is reached. The sequence is then ordered in such a way that it produces subsets that evenly cover the entire domain. Halton sequences are often used for numeric methods like Monte Carlo simulations [6]. Another advantage of a Halton sequence is that it never contains the same number twice so that we do not choose the same position for multiple ingredients. This approach provides an efficient and rapid way of distributing ingredients on the surface of the membrane with sufficient randomness while minimizing potential overlaps.

The orientation of the ingredient is then described as the principal vector defining the orientation of the ingredient with respect to the membrane, and an offset vector that places the ingredient at the proper position relative to the surface of the membrane (Figure 6). This information is usually computed using the OPM webserver [31].

5.2.2 Population of Compartment Inner Area

For populating the soluble space of compartments, we use the information stored in the compartment grid and the occupancy grid to find appropriate positions for the ingredients. The membrane-bounded ingredients positioned in the previous step are considered as non-moving obstacles, i.e., they are handled as static objects when resolving collisions.

The population procedure works as follows. For each instance of the ingredient defined in the recipe, we place it at a voxel marked

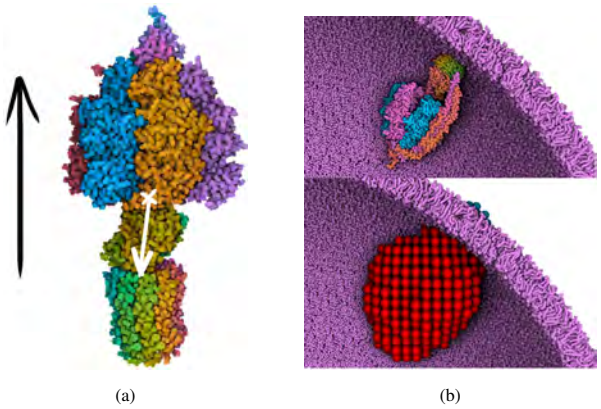


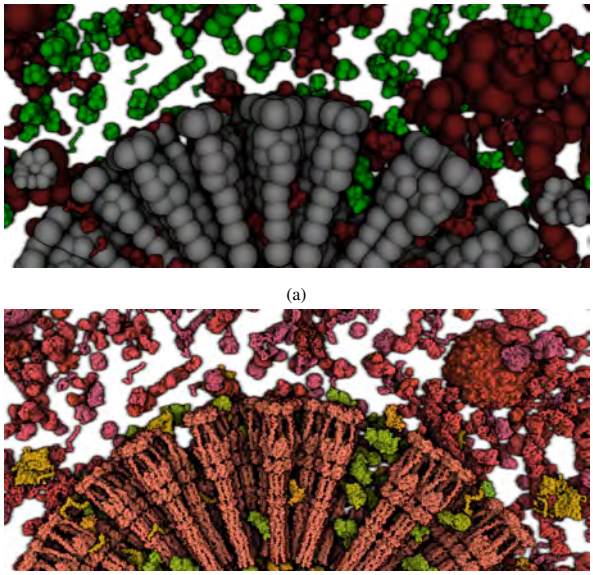
Fig. 6: (a) Example of surface ingredient with proper principal vector (black), and offset vector (white), pointing from the center of mass to the anchor position at the surface. (b) Positioning of the ingredient on the surface using the correct orientation and offset aligned to the surface normal (top). Grid occupancy update (bottom).

with the appropriate value in the compartment grid, and with an empty value in the occupancy grid. The occupancy grid is then updated with the identity of the ingredient, and these steps are repeated until all ingredients are positioned (Figure 5 (c), (d)). The occupancy grid is updated even after the positioning of all ingredients is completed, so that the grid may be used in populating this compartment with fibrous structures (see Section 6).

As the size of the ingredients can be substantially larger than the voxel size, this will ultimately result in intersecting ingredients if they are placed in a close proximity. The population step does not necessarily need to take overlap into consideration, although the less overlap is produced, the less computation is required. The following resolves the overlaps. A straightforward approach to avoid overlaps is to uniformly distribute ingredients. However, this leads to an undesired visual appearance since the grid structure becomes clearly visible. In order to introduce a certain degree of randomness and still reduce overlaps, we again utilize a Halton sequence to choose potential grid points for population. In practice a compartment populated using a Halton sequence can still suffer from visible regularities. Therefore, we additionally introduce jittering by a random vector in each grid position and we also apply a random rotation to each ingredient. With the Halton sequence, we can significantly reduce the number of overlapping molecules but it does not completely avoid them. The following step detects and resolves these remaining overlaps.

5.3 Step 3 – Detecting and Resolving Collisions

Ideally, we would want to detect collisions by evaluating contacts between all atoms in the molecules. However, this is not practical in terms of performance, so for every molecule type, we compute a two-level proxy geometry that approximates its shape. The higher-level proxy is simply the bounding sphere of the molecule. The lower-level proxy approximates the molecule with a small number of spheres calculated with a GPU-based K-means clustering algorithm [11]. As preset we use 16 spheres. The clustering provides the centers of the spheres and the radii which correspond to the cluster sizes. This ensures that all atoms are covered by the proxy geometry. The collision-detection process first uses the higher level proxy geometry to detect potential collisions between two molecules, which are then verified with the lower-level proxy geometries using the finer approximation of the shape. Figure 7 shows the lower-level of the proxy geometries (a) and the scene populated by ingredients (b). The proxy geometry slightly overestimates the actual shape of the molecule to ensure that a certain distance is created between molecules after the collision is resolved. From a perceptual point of view, the molecules are easier to distinguish if their shape boundaries are not in direct contact.



(a)

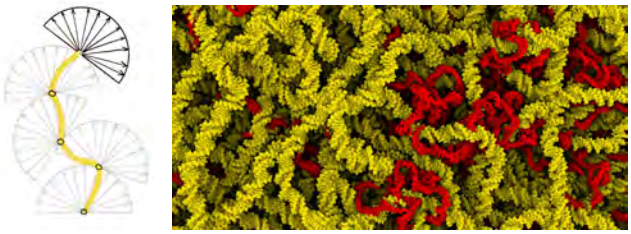
(b)

Fig. 7: (a): Proxy geometry of ingredients. Overlapping geometries are marked in red, non-overlapping ones in green, and static ones in grey. (b) The same scene populated by ingredients.

The resolving of collisions is loosely based on standard rigid body dynamics [1]. Usually the exact point of contact between two bodies is detected and then corresponding forces are computed. Since our molecules are already in a colliding state, we simply compute forces that resolve these collisions. We use the information about collisions from the lower-level of the proxy geometry as a basis for the computation. For each intersecting pair of molecules we determine the overlapping sphere pairs and consider each as one collision. Each molecule accumulates a linear and angular force that is updated for every collision. The linear force is defined by a direction and strength. The direction is derived from the vector between the centers of the two spheres. The distance between the two sphere centers determines the strength of the force. As in rigid body dynamics, we compute the angular force by taking into account the vector between the center of the molecule and the collision, as well as the direction vector. After the forces for all collisions are computed, they determine the new position and rotation of molecules, as in a standard physics-based system. This defines one integration step. During the collision-resolving process, a molecule might temporarily leave its compartment. In this case we apply a force to the molecule, that will steer it back into the compartment. This process is repeated for a certain number of integration steps until all collisions are resolved or a stop criterion is reached. The criterion can be set by the user and specifies a certain number of acceptable remaining collisions. This makes the system flexible for less powerful hardware systems as well. Both the collision detection and the subsequent resolving is computed on the GPU.

6 Fibrous Structures

Fibrous ingredients are linear or branched polymers of repeated units, found in many places in cells. They include protein filaments that form infrastructure, polysaccharides that provide protection or store energy, and nucleic acids that encode genetic information. They are usually modeled using a procedural growing algorithm, such as self-avoiding random walks. Angle and length are constrained to maintain the persistent length of the processed fiber. The process of building the fibrous structure is sequential by nature as we have to be aware of the previous state of the fiber. However, this process can be greatly improved by using the compartment and occupancy grid information. For every incremental step of the walk, the corresponding grid element can be checked if it belongs to the compartment (compartment grid) and if it is occupied by some other protein (occupancy grid). This improves the computational cost by restraining the growing fiber to allowed space and minimizing intersections with previously positioned ingredients, including self-intersections. For large systems, such as genomes, the fiber can contain millions of subunits, so the complete fiber-growing process will not be interactive. Here, we provide the possibility for the user to observe the current progress of the fiber growth, to remain interactive.



(a) (b)

Fig. 8: Fibrous structures are modeled using a self-avoiding random walk. (a) The walk starts with a random point inside of a certain compartment. A random direction in the forward-hemisphere at this point is then chosen. If there is no intersection with existing structures or compartment surfaces, the direction is selected and the segment is added to the fiber. This process is repeated until the desired length of the fiber is reached. (b) A very long DNA (yellow) with long persistence length is combined with shorter RNA (red) with short persistence length.

The self-avoiding random walk in our approach works as follows. In the target compartment, a starting point is randomly chosen and the ensuing protocol is iterated until the required length of the fiber is reached. In each step, a random direction is selected from the hemisphere that is oriented and positioned according to the previous direction, as illustrated in Figure 8 (a). Possible intersections of the random walk with membranes, soluble molecules or itself are tested by using the occupancy and compartment grids. If there are no intersections, the segment is added and the grids are updated. Otherwise, a different random direction is chosen. A dead-end is reached if the walk cannot be continued after a given number of attempts. In our experience, the order in which molecules are distributed strongly influences the success of the population step. We have obtained the best results by placing membrane-bounded proteins first, followed by fibers, and finishing with soluble ingredients. This sequence addresses two significant potential problems: (i) membrane-bounded proteins often protrude in the compartments, so the fiber must avoid them, and (ii) typical soluble components are highly crowded into compartments, lead to few free grid points, and leave no options for fiber growth. Figure 8 (b) shows result of positioning DNA and RNA fibrous structures.

The method provides interactive performance for small fibers, such as the lipopolysaccharides extending from the mycoplasma surface or small RNA molecules in the cytoplasm. Long genomic DNA molecules, however, require several minutes to generate. We provide two methods to allow the user more interaction for these large structures: a premade DNA model may be preloaded, or we grow the walk in real-time to show the progress. This enables the user to stop and restart if the growing model shows problems.

7 Results and Discussion

The approach described here is flexible, allowing the user to model a variety of biological systems. This ability is due, in part, to the relationship of all living organisms on Earth. Similar membranes surround all cells and define their inner organelles, and similar proteins orchestrate the traffic of molecules and information across the membranes. Similar proteins and nucleic acids are used for the many metabolic and genetic tasks required for energy production, biosynthesis, and reproduction, which are packaged in very similar compartments within cells. Since

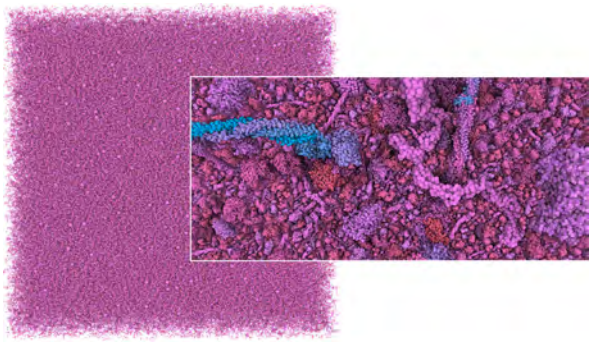


Fig. 9: A cubic space, two cubic microns wide, filled with blood plasma, with over 4.6 million soluble protein ingredients, populated in ∼0.6 sec.

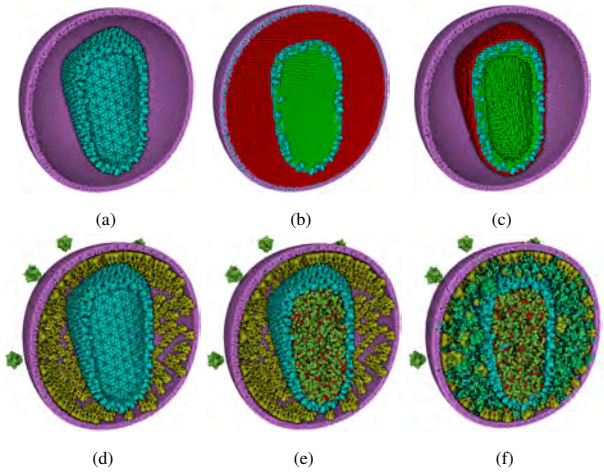


Fig. 10: Overview of the interactive modeling of the mature HIV virion based on the cellPACK recipe. (a) Building the membrane (magenta) based on a spherical mesh, and preloading the capsid (blue) from a cryo-EM structure (PDB entry 3J3Q). (b) Solid voxelization of compartments. (c) Updating the occupancy grid. (d) Populating the surface ingredients (envelope protein in green, matrix protein in yellow). (e) Building two copies of the RNA (red) using the fiber growth method and attaching the HIV NC protein at all control points. (f) Resulting model.

viruses rely on cells for reproduction, they are also built from a subset of this machinery. With a limited set of structural modeling tools, we are able to build a variety of different cellular scenes.

Building mesoscale models is always an iterative process, where the recipe, ultrastructure and other parameters are tuned to reflect on and lend insight to the available experimental results. This is a laborious process with systems such as cellPACK, where single models require from minutes to hours to generate, followed by a significant effort to visualize the results. Our approach tightens this research cycle, allowing researchers to generate models or ensembles of models rapidly in an intrinsically visual environment. This allows the users to interactively optimize recipes, identify and correct any errors or bugs, and incorporate new data rapidly as it becomes available.

We demonstrate the capabilities and advantages of this new approach in three test systems. In a mesoscale model of blood plasma, the efficiency of the current method allows us to create a very large scene containing millions of molecules, which required a prohibitive computational effort with previous methods. With HIV, the interactive capability allows us to generate multiple models from a single recipe, and explore stochastic variations of a structurally complex subject. In a model of an entire mycoplasma bacterium, we are able to tune the recipe and modeling parameters interactively to achieve a particular goal: creating a 3D version of an existing 2D rendering of the cell. In the following sections, we describe these cases as well as the achieved results. The performance was measured using a computer with an Intel Core i7-6700K CPU 4.00 GHz and NVIDIA GeForce GTX 1080 graphics card with 8 GB memory.

Blood Plasma. Blood plasma is largely homogeneous and relatively sparsely occupied by proteins and short fibrous components. These proteins have a variety of functions, including molecules of the immune system, blood clotting factors, transport molecules, and hormones. It is a perfect system for early tests of methods for populating mesoscale scenes, because the proteins have a variety of sizes and shapes, including several with very large aspect ratios. When generating the model using cellPACK, a scene containing nearly 30,000 molecules was created in 116 seconds. With our approach we have tested a scene with blood plasma containing more than 33,000 molecules. The population step takes on average 100 ms and the consecutive overlap resolving takes on average 2000 ms. Our system is also capable of populating significantly larger scenes. As shown in Figure 9, we have populated a scene with a volume of two cubic microns with blood plasma. Given the exact molarity, our approach produces over 4.6 million soluble protein ingredients in 0.6 seconds. In our current implementation, this is beyond the capabilities of our force-based system to resolve potential overlaps. The main bottleneck here is the collision detection. However, this is not a conceptual limitation. Current research [20] has shown that it is already possible to interactively solve the collisions of millions of particles.

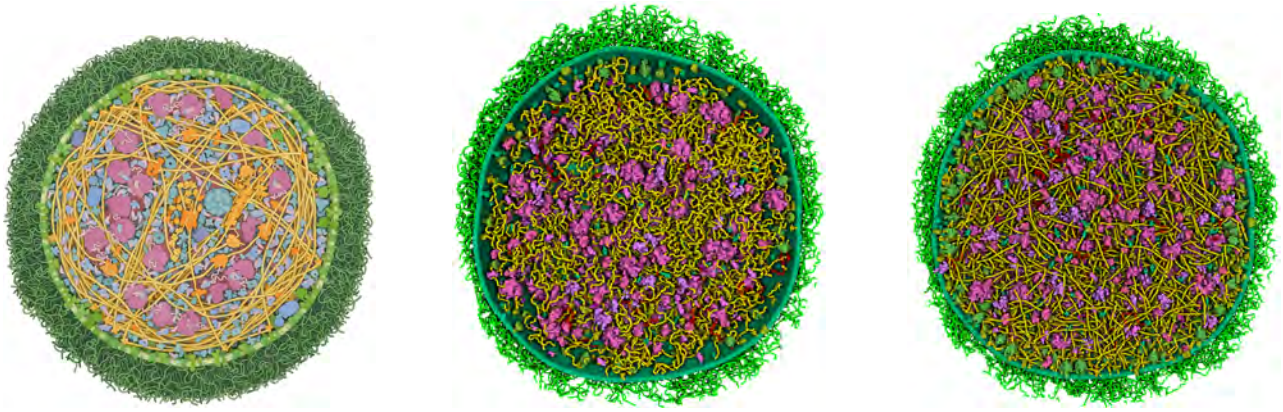
HIV. We have used HIV as a convenient test system for mesoscale modeling since the inception of the cellPACK project. It has several advantages: it is relatively small, but still taxes most methods and it is one of the most-studied organisms, so there is abundant information available. However, there are still significant gray areas in the available data, so this leaves room for experimentation in the parameters of the recipe.

The recipe used here was described by Johnson et al. [26], and comprises five types of surface proteins, 19 types of interior proteins and two RNA copies (9,200 bases, roughly 30,000 Ångströms long). A membrane envelope surrounds the virion, with several copies of the viral envelope protein embedded in the membrane and extending outwards. Inside, a protein-enclosed capsid protects two copies of the viral genome. A variety of proteins, including the viral enzymes and several accessory proteins, are found in the space between the membrane and the capsid.

Figure 10 depicts the different steps of our procedural approach to model HIV. The final model consists of 11,790 protein ingredients. The entire model requires 0.07 seconds to generate without the RNA, and requires ∼3 seconds to generate the two copies of the RNA. It takes up to an hour to generate this model using cellPACK.

Mycoplasma Bacterium. The ultimate goal of this work is to create computational models of entire cells, to interpret experimental results, and model structure/function relationships in healthy and diseased states. To address this challenge, we have focused on one of the simplest cells, a mycoplasma bacterium. This bacterium has a very simple structure, with a single cellular membrane and a small genome. The membrane is filled with pumps and channels, and decorated on the outer surface with lipopolysaccharides that form a protective coat around the cell. The detailed organization of molecules inside the cell is only now beginning to be explored by cryo-EM microscopy, given the small size of the cells. We have modeled a random tangle of DNA surrounded by the soluble protein and RNA components.

The draft recipe for the current model (Figure 11) includes the membrane, one strand of DNA (1,211,703 base pairs), 1500 lipopolysaccharides, and a collection of 18 of the most prevalent proteins and nucleic acids. The model required ∼100 sec for the generation of all the fibers in the model (DNA, mRNA, peptides, lipopolysaccharides) and 0.1 sec to fill the remaining space with soluble components. Using cellPACK, the creation of the mycoplasma model takes several hours. We employed the interactive capabilities of the method to tune the recipe,



(a) (b) (c)

Fig. 11: Cross-sections of Mycoplasma mycoides. (a) A hand-drawn illustration. (b) 3D model generated with the draft recipe. (c) 3D model generated by interactively tuning the recipe to match general characteristics of the illustration. In the 3D models, the membrane is in green, with membrane-embedded proteins in orange and lipopolysaccharides in green. Inside, DNA is in yellow, soluble proteins in blue and magenta, and RNA in red.

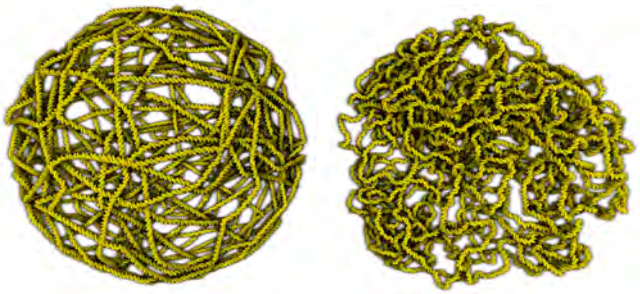
based on previous research used to create a hand-drawn illustration of the cell. Figure 11 includes the illustration, a 3D model using the draft recipe, and a 3D model generated after interactively tuning the recipe to better match the illustration.

Performance and Interactive Capabilities

Creation of the presented types of models has previously been a laborious process, forming a bottleneck in their study. For instance, the micron-sized blood plasma scene previously required hours of computation time, and is modeled with the new method in less than a second. Currently, the method is able to populate soluble compartments interactively, allowing the user on-the-fly experimentation with recipes and testing their effects on the resultant scenes. The performance of modeling fibrous and membrane components has also been significantly improved when comparing with previous approaches. It takes seconds to generate a cell-sized membrane and about a minute to calculate a genome-sized object.

One of the great advantages of our presented methods is the ability to test hypotheses interactively. Biological research is an ongoing process, and there are many gray areas in the body of knowledge about the structure and function of cells at the mesoscale level. In many cases, it is useful to test the emergent mesoscale consequences of different properties at the molecular level. In the current implementation, many useful parameters are tunable at runtime. The user imports meshes that represent compartments, descriptions of ingredients, and then creates and visualizes the model based on the recipe on the fly. Tunable parameters in the recipe currently include the location, the molarity, the orientation (as defined by principal vector and offset), and fiber-generation parameters. Once the basic geometry is loaded, everything may be changed interactively.

For example, in Figure 12, we tested two assumptions on DNA persistence length in a model of a small genome packed into a spherical bacterium. A longer persistence length leads to DNA strands that wrap around the periphery of the cell, whereas a shorter persistence length leads to a trajectory that is more reminiscent of a 3D random walk. The reality is probably a combination of these two assumptions, since DNA-binding proteins often form kinks that mimic the short persistence length, while free strands of DNA show a longer persistence length.



(a) (b)

Fig. 12: Two models of DNA in a bacterial cell, using (a) a long persistence length and (b) a short persistence length (7 sec to generate).

8 Conclusion and Future Work

In this work we have presented the first interactive approach to an integrative structural modeling of the biological mesocale. The interactive performance of the method provides several significant advantages in scientific and educational applications. With the possibility to create interactive models, domain experts are now able to tune parameters and explore the mesoscale properties that emerge. In addition it is now possible to create an ensemble of models from a single recipe, allowing researchers to explore the stochastic variations in mesoscale properties and compare them with experiments. In education, interactive capabilities can allow curriculum developers to better tune their models and representations to optimize the clarity of the resultant imagery. A long tradition in scientific illustration has proven the pedagogical utility of introducing small artistic changes to a subject to improve the interpretability of what is shown.

The next major enhancement of the method will be the incorporation of a faster and more intuitive approach to model interactions between the ingredients. For instance, modeling the interaction of ribosomes with messenger RNA to form a polysome, or crosslinking of DNA by DNA-binding proteins, is currently not feasible. We are currently exploring particle simulation methods such as NVidia Flex to add this capability. In addition, we will continue to optimize the generation of fibrous components, which is currently the rate-limiting step of the model generation with the new approach.

Acknowledgments

This project has been funded by the Vienna Science and Technology Fund (WWTF) through project VRG11-010 and also supported by EC Marie Curie Career Integration Grant through project PCIG13-GA-2013-618680. The Scripps Research Institute researchers acknowledge support from NIH P41-GM103426 and R01-GM120604, and this manuscript has TSRI reference number 29494.

REFERENCES

[1] D. Baraff. An introduction to physically based modeling: rigid body simulation I-unconstrained rigid body dynamics. *SIGGRAPH Course Notes*, pp. D31–D68, 1997.

[2] M. Beck and W. Baumeister. Cryo-electron tomography: Can it reveal the molecular sociology of cells in atomic detail? *Trends in Cell Biology*, 26(11):825–837, 2016. Special Issue: Future of Cell Biology. doi: 10.1016/j.tcb.2016.08.006

[3] 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.

[4] P. Boechat, M. Dokter, M. Kenzel, H.-P. Seidel, D. Schmalstieg, and M. Steinberger. Representing and scheduling procedural generation using operator graphs. *ACM Transactions on Graphics*, 35(6):183:1–183:12, 2016. doi: 10.1145/2980179.2980227

[5] K. Culik and J. Kari. An aperiodic set of Wang Cubes. *Journal of Universal Computer Science*, 1(10):675–686, 1996.

[6] I. L. Dalal, D. Stefan, and J. Harwayne-Gidansky. Low discrepancy sequences for monte carlo simulations on reconfigurable platforms. In *Application-Specific Systems, Architectures and Processors, 2008. ASAP 2008. International Conference on*, pp. 108–113. IEEE, 2008.

[7] P. Decaudin and F. Neyret. Rendering forest scenes in real-time. In *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques*, EGSR'04, pp. 93–102. Eurographics Association, 2004. doi: 10.2312/EGWR/EGSR04/093-102

[8] C. J. Dickson, B. D. Madej, A. A. Skjevik, R. M. Betz, K. Teigen, I. R. Gould, and R. C. Walker. Lipid14: The Amber lipid force field. *Journal of Chemical Theory and Computation*, 10(2):865–879, 2014. PMID: 24803855. doi: 10.1021/ct4010307

[9] J. D. Durrant and R. E. Amaro. LipidWrapper: an algorithm for generating large-scale membrane models of arbitrary geometry. *PLOS Computational Biology*, 10(7):e1003720, 2014.

[10] K. Fan, X.-Y. Li, and D.-C. Wu. Self-avoiding walk on a three-dimensional Manhattan lattice. *Journal of Physics A: Mathematical and General*, 33(22):3971, 2000.

[11] R. Farivar, D. Rebolledo, E. Chan, and R. H. Campbell. A Parallel Implementation of K-Means Clustering on GPUs. In *PDPTA'08 - The 2008 International Conference on Parallel and Distributed Processing Techniques and Applications*, pp. 340–345, 2008.

[12] K. W. Fleischer, D. H. Laidlaw, B. L. Currin, and A. H. Barr. Cellular texture generation. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 239–248. ACM, New York, NY, USA, 1995. doi: 10.1145/218380.218447

[13] C.-W. Fu and M.-K. Leung. Texture tiling on arbitrary topological surfaces using wang tiles. In *Rendering Techniques*, pp. 99–104, 2005.

[14] E. Galin, A. Peytavie, N. Maréchal, and E. Guérin. Procedural Generation of Roads. *Computer Graphics Forum (Proceedings of Eurographics)*, 29(2):429–438, 2010.

[15] D. S. Goodsell. Inside a living cell. *Trends in Biochemical Sciences*, 16:203–206, 1991. doi: 10.1016/0968-0004(91)90083-8

[16] S. Greuter, J. Parker, N. Stewart, and G. Leach. Real-time procedural generation of 'pseudo infinite' cities. In *Proceedings of the 1st International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, GRAPHITE '03, pp. 87–ff. ACM, New York, NY, USA, 2003. doi: 10.1145/604471.604490

[17] S. Grottel, M. Krone, C. Müller, G. Reina, and T. Ertl. MegaMol – a prototyping framework for particle-based visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 21(2):201–214, 2015. doi: 10.1109/TVCG.2014.2350479

[18] G. Gruenert, B. Ibrahim, T. Lenser, M. Lohel, T. Hinze, and P. Dittrich. Rule-based spatial modeling with diffusing, geometrically constrained molecules. *BMC Bioinformatics*, 11(1):307, 2010. doi: 10.1186/1471-2105-11-307

[19] J. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2:84–90, 1960.

[20] R. Hoetzlein. Fast fixed-radius nearest neighbors: interactive million-particle fluids. In *GPU Technology Conference*, p. 18, 2014.

[21] S. Hornus, B. Lévy, D. Lariviére, and E. Fourmentin. Easy DNA modeling and more with GraphiteLifeExplorer. *PLoS ONE*, 8(1):1–12, 01 2013. doi: 10.1371/journal.pone.0053609

[22] W. Im, J. Liang, A. Olson, H.-X. Zhou, S. Vajda, and I. A. Vakser. Challenges in structural approaches to cell modeling. *Journal of Molecular Biology*, 428(15):2943–2964, 2016. doi: 10.1016/j.jmb.2016.05.024

[23] S. Jo, J. B. Lim, J. B. Klauda, and W. Im. CHARMM-GUI membrane builder for mixed bilayers and its application to yeast membranes. *Biophysical Journal*, 97(1):50–58, 2009. doi: 10.1016/j.bpj.2009.04.013

[24] G. T. Johnson, L. Autin, M. Al-Alusi, D. S. Goodsell, M. F. Sanner, and A. J. Olson. cellPACK: a virtual mesoscope to model and visualize structural systems biology. *Nature methods*, 12(1):85–91, 2015.

[25] G. T. Johnson, L. Autin, D. S. Goodsell, M. F. Sanner, and A. J. Olson. ePMV embeds molecular modeling into professional animation software environments. *Structure*, 19(3):293–303, 2011.

[26] G. T. Johnson, D. S. Goodsell, L. Autin, S. Forli, M. F. Sanner, and A. J. Olson. 3D molecular models of whole HIV-1 virions generated with cellPACK. *Faraday Discussions*, 169:23–44, 2014.

[27] G. T. Johnson and S. Hertig. A guide to the visual analysis and communication of biomolecular structural data. *Nature Reviews Molecular Cell Biology*, 15(10):690–698, 10 2014.

[28] I. Kolesár, J. Parulek, I. Viola, S. Bruckner, A.-K. Stavrum, and H. Hauser. Interactively illustrating polymerization using three-level model fusion. *BMC Bioinformatics*, 15(1):345, 2014. doi: 10.1186/1471-2105-15-345

[29] B. Kozlíková, M. Krone, M. Falk, N. Lindow, M. Baaden, D. Baum, I. Viola, J. Parulek, and H.-C. Hege. Visualization of biomolecular structures: State of the art revisited. *Computer Graphics Forum*, 2016. doi: 10.1111/cgf.13072

[30] L. Krecklau, J. Born, and L. Kobbelt. View-dependent realtime rendering of procedural facades with high geometric detail. *Computer Graphics Forum*, 32(2pt4):479–488, 2013. doi: 10.1111/cgf.12068

[31] M. A. Lomize, A. L. Lomize, I. D. Pogozheva, and H. I. Mosberg. OPM: orientations of proteins in membranes database. *Bioinformatics*, 22(5):623–625, 2006.

[32] A. Lu, D. S. Ebert, W. Qiao, M. Kraus, and B. Mora. Volume illustration using Wang Cubes. *ACM Transactions on Graphics*, 26(2):11, 2007. doi: 10.1145/1243980.1243985

[33] Z. Lv, A. Tek, F. Da Silva, C. Empereur-mot, M. Chavent, and M. Baaden. Game on, science - how video game technology may help biologists tackle visualization challenges. *PLoS ONE*, 8(3):e57990, 2013.

[34] L. Martínez, R. Andrade, E. G. Birgin, and J. M. Martínez. PACKMOL: A package for building initial configurations for molecular dynamics simulations. *Journal of Computational Chemistry*, 30(13):2157–2164, 2009. doi: 10.1002/jcc.21224

[35] S. R. McGuffee and A. H. Elcock. Diffusion, crowding & protein stability in a dynamic molecular model of the bacterial cytoplasm. *PLOS Computational Biology*, 6(3):1–18, 2010. doi: 10.1371/journal.pcbi.1000694

[36] M. L. Muzic, L. Autin, J. Parulek, and I. Viola. cellVIEW: a tool for illustrative and multi-scale rendering of large biomolecular datasets. In *Eurographics Workshop on Visual Computing for Biology and Medicine*, pp. 61–70. EG Digital Library, The Eurographics Association, 2015.

[37] F. Neyret and M.-P. Cani. Pattern-based texturing revisited. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pp. 235–242. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999. doi: 10.1145/311535.311561

[38] T. Reddy and M. S. Sansom. Computational virology: From the inside out. *Biochimica et Biophysica Acta (BBA) - Biomembranes*, 1858(7, Part B):1610–1618, 2016. doi: 10.1016/j.bbamem.2016.02.007

[39] M. Schwarz and H.-P. Seidel. Fast parallel surface and solid voxelization on GPUs. *ACM Transactions on Graphics*, 29(6):179:1–179:10, 2010. doi: 10.1145/1882261.1866201

[40] D. Sirohi, Z. Chen, L. Sun, T. Klose, T. C. Pierson, M. G. Rossmann, and R. J. Kuhn. The 3.8 Å resolution cryo-EM structure of Zika virus. *Science*, 352(6284):467–470, 2016. doi: 10.1126/science.aaf5316

[41] M. Steinberger, M. Kenzel, B. Kainz, P. Wonka, and D. Schmalstieg. On-the-fly generation and rendering of infinite cities on the GPU. *Computer Graphics Forum*, 33(2):105–114, 2014. doi: 10.1111/cgf.12315

[42] M. Tarini, K. Hormann, P. Cignoni, and C. Montani. Polycube-maps. *ACM Transactions on Graphics*, 23(3):853–860, 2004. doi: 10.1145/1015706.1015810

[43] A. Vendeville, D. Lariviére, and E. Fourmentin. An inventory of the bacterial macromolecular components and their spatial organization. *FEMS Microbiology Reviews*, 35(2):395–414, 2011. doi: 10.1111/j.1574-6976.2010.00254.x

[44] S. M. Waldon, P. M. Thompson, P. J. Hahn, and R. M. Taylor. SketchBio: a scientist's 3D interface for molecular modeling and animation. *BMC Bioinformatics*, 15(1):334, 2014. doi: 10.1186/1471-2105-15-334

[45] T. Waltemate, B. Sommer, and M. Botsch. Membrane Mapping: Combining Mesoscopic and Molecular Cell Visualization. In *Eurographics Workshop on Visual Computing for Biology and Medicine*. The Eurographics Association, 2014. doi: 10.2312/vcbm.20141187

[46] H. Wang. Proving theorems by pattern recognition-II. *Bell Labs Technical Journal*, 40(1):1–41, 1961.

[47] B. G. Wilhelm, S. Mandad, S. Truckenbrodt, K. Kröhnert, C. Schäfer, B. Rammner, S. J. Koo, G. A. Claßen, M. Krauss, V. Haucke, H. Urlaub, and S. O. Rizzoli. Composition of isolated synaptic boutons reveals the amounts of vesicle trafficking proteins. *Science*, 344(6187):1023–1028, 2014. doi: 10.1126/science.1252884

[48] A. Willmott. Fast object distribution. *In SIGGRAPH sketches*, p. 80, 2007.

[49] P. Wonka, M. Wimmer, F. Sillion, and W. Ribarsky. Instant architecture. *ACM Transactions on Graphics*, 22(3):669–677, 2003. doi: 10.1145/882262.882324

CHAPTER 6

Paper C - Multi-Scale Procedural Animations of Microtubule Dynamics Based on Measured Data

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics

Multi-Scale Procedural Animations of Microtubule Dynamics Based on Measured Data

Tobias Klein, Ivan Viola, Eduard Gröller, and Peter Mindek

Fig. 1: A procedurally generated, real-time rendered model of a microtubule assembly inside a cell cytoplasm.

Abstract—Biologists often use computer graphics to visualize structures, which due to physical limitations are not possible to image with a microscope. One example for such structures are microtubules, which are present in every eukaryotic cell. They are part of the cytoskeleton maintaining the shape of the cell and playing a key role in the cell division. In this paper, we propose a scientifically-accurate multi-scale procedural model of microtubule dynamics as a novel application scenario for procedural animation, which can generate visualizations of their overall shape, molecular structure, as well as animations of the dynamic behaviour of their growth and disassembly. The model is spanning from tens of micrometers down to atomic resolution. All the aspects of the model are driven by scientific data. The advantage over a traditional, manual animation approach is that when the underlying data change, for instance due to new evidence, the model can be recreated immediately. The procedural animation concept is presented in its generic form, with several novel extensions, facilitating an easy translation to other domains with emergent multi-scale behavior.

Index Terms—Procedural modeling, molecular visualization, animation, microtubules.

1 Introduction

Molecular biology studies various complex structures composed of macromolecules, such as proteins or nucleic acids. These structures are highly dynamic and they carry out various tasks important for the function of a cell. However, as these structures are often smaller than the wavelength of light, their dynamics cannot be directly observed with a microscope. One way to display these structures, either for education, interdisciplinary communication, or hypothesis generation and testing, is through scientific illustration.

Biologists often display their findings about nanoscopic structures by means of computer graphics. However, 3D models of these structures have to be created first. This is a time-consuming and expensive task carried out by scientific animators who carefully study the underlying biological data and findings to manually create expressive visual representations of these phenomena. Only after the models are produced and rigged for animation, the animators can work on the overall look and feel of the animation and the storytelling aspects.

One of the structures of high interest to biologists are microtubules. They are long tubes spanning across eukaryotic cells, assembled from molecules of a protein called *tubulin*. As part of the cytoskeleton, microtubules carry out various functions, such as maintaining the cell shape, acting as channels for material transportation and as signal broadcasting system, and providing support in cell division. All of these functions are possible through a process called *dynamic instability*. It is a property of microtubules, where they constantly and in cycles grow, disassemble, and start growing again. In this way, the entire cytoskeleton is able to reshape according to the current needs of the cell.

Dynamic instability is extensively studied in biology. There are still aspects of it that are not fully understood. One of these aspects is the sequence in which the individual tubulin molecules bind together to assemble a microtubule, and the way the microtubule is disassembled. Various studies provide evidence for several theories on how this process functions. To communicate these findings, it is necessary to create illustrations, as the process itself cannot be observed with a microscope due to physical limits.

Nowadays, scientific animators manually generate dynamic 3D models representing the way how tubulins are added to a growing microtubule, or how they are removed when the microtubule shrinks. These models are created based on evidence biologists gather through various observations and experiments.

Building such models is a very time-consuming, and therefore expensive task. Since the data about the dynamic instability are not yet conclusive, it is reasonable to assume new evidence will be found and

- *Tobias Klein is with TU Wien. E-mail: tklein@cg.tuwien.ac.at.*
- *Ivan Viola is with KAUST. E-mail: ivan.viola@kaust.edu.sa.*
- *Eduard Gröller is with TU Wien and VRVis. E-mail: groeller@cg.tuwien.ac.at.*
- *Peter Mindek is with TU Wien and Nanographics GmbH. E-mail: mindek@cg.tuwien.ac.at.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics

our understanding of the process will change. In that case, the model would have to be recreated from scratch.

To alleviate scientific animators from manually recreating microtubule models whenever new data about their growth and shrinkage are gathered, we propose a fully procedural model parameterized by measurement data. We identified several key parameters, which can be adjusted in our model according to the given theory of the microtubule dynamics, to generate a realistic visualization representing the respective theory.

To make our work extensible to other situations, where multi-scale animations of complex molecular environments are needed, we design our model within a framework for multi-scale procedural animations. The framework can be used in different application domains, where dynamic behaviour has to be modelled, and where it is essential to examine various parameter settings for the generated animations. This could be necessary, because the precise nature of the dynamics is not yet known, and hence the parameters might have to be readjusted once new evidence about the dynamics is gathered. In another example the nature of the dynamics changes according to the given context, for instance, microtubule growth might behave differently in vivo and in vitro. The ability to capture all these different characteristics within a single procedural model gives animators and biologists a powerful tool for visualizing the complex processes they are studying.

The contributions of this paper are two-fold: We introduce a novel framework for designing procedural animations in environments, where distinct behaviours on different scales interact and influence each other, for instance molecular biological processes. Subsequently, we apply this framework to design a procedural animation model on the dynamic instability of microtubules. This is an important problem studied in biology, where 3D visualizations are essential to understand the scientific data and to put them into perspective. Our procedural model can be integrated with existing real-time molecular visualization systems, so that the microtubules can be depicted in complex molecular environments, as shown in Fig. 1.

2 Related Work

Animations of molecular data are related with various techniques and processes that are fundamental to build and communicate visual narratives in a meaningful way. In a typical case, a biological model builds the foundation of the narrative. This model is assembled with a combination of data from structural biology, proteomics, and microscopy. The assembling is often realized with techniques that originate from the field of procedural modeling. Additionally, the rendering of such models implies challenges that are distinct from conventional mesh rendering. In order the explain the functionality of the showcased model, dynamics are added to the scene, which result from either animations or even simulations. In this section, we review literature corresponding to the three involved areas - procedural modeling and animation, molecular visualization, and molecular animation and storytelling.

Procedural Modeling and Animation

Models of structural biology often reveal patterns similar to other natural phenomena or man-made structures that show a high degree of geometric complexity. In visualization and computer graphics, structures that comprise a substantial amount of repetition are typically modeled in a procedural way. Early approaches procedurally generate natural phenomena, such as forests [32] and man-made structures like cities [33]. There are various ways to formulate the procedure, such as declarative modeling approaches [38], shape grammars [31], or L-systems [45]. Many approaches are even able to generate whole urban landscapes [47], or infinite cities in real-time [41, 42], a technique often used in computer games. The computational effort increases with the size and complexity of the modeled structure. A frequently used way to reduce the computation is to only generate what is currently visible in the viewing frustum of the camera [23].

Computational models of the biological meso- and nanoscale, like models of bacteria, viruses, and nucleic acids, use similar procedures for the generation. They enable users to observe the structure in atomic detail by combining data from many sources. For instance, cellPACK [18] utilizes a packing algorithm to assemble complex biological models. A more recent approach [21] extends this concept to real-time and combines it directly with the visualization.

Procedural methods for generating animations have long been used in the video-game industry for character animation, clothing [35], ocean waves [15], or clouds [36]. They simplify the process of animators to control the behavior of a model. Even for models with complex parameters, example-based approaches [2] exist that support the user in finding suitable settings. However, none of the previous approaches addresses the specifics of multi-scale models consisting of many instances, such as molecular scenes.

Molecular Visualization

Molecular visualization has a long history compared to some other data visualization branches, starting with early hand drawings of cellular environments [9] or digital drawings [48] made by computer-animation and modeling software [19]. Digital approaches often use protein building blocks from the comprehensive Protein Data Bank [1] to assemble larger biological models. There are several systems capable of rendering large datasets with up to billions of atoms [5, 27]. For instance, the cellview [25] system is a tool for illustrative multi-scale visualization of large biomolecular datasets. MegaMol [11] is capable of rendering various atomic representations not limited to biomolecules.

The structure of molecular data is complex and challenging to understand even through visualization. Its shape should be perceived without ambiguities to derive the correct information. To improve the spatial shape perception and aesthetics of molecular visualization, various rendering techniques from computer graphics have been adapted to the special characteristics of molecular data. Tarini et al. [43] have shown that the user's understanding of three-dimensional structures strongly increases with ambient occlusion. Further methods to enhance the perception include halos [12], translucency [13], depth darkening [28], and line drawings algorithms [24]. Waldin et al. [46] have shown that adjustments of the color scheme depending on the current scale level improve the capability to distinguish between different structures. Recent work by Hermosilla et al. [14] presents an illumination model that can be applied across various atom-based molecular representations.

In this paper, we utilize a framework called *Marion* [30], which incorporates a multi-pipeline approach for communicating biology. We have incorporated many essential effects to enhance the perception of the visualized structures, such as ambient occlusion, halos, depth of field, fog, and a variant of the chameleon coloring approach.

In many cases, molecular visualization shows single instances of a bacteria or viruses, where the information in between development stages is often not clear. The work of Sorger et al. [39] presents a novel approach that uses illustrative transitions and abstractions to visually transform between such stages. Molecular systems are typically visualized with various degrees of abstraction. Continuous abstraction methods [44] facilitate the creation of seamless transitions between different levels of abstraction. An extensive review of molecular visualization is presented by Kozlíková et al. [22] and Johnson and Hertig [20].

Molecular Animation and Storytelling

There is a recent shift from studying individual proteins to modeling and analyzing complex assemblies, like microtubules, up to larger systems like viruses and bacteria. These structures reside in the cellular mesoscale, which is the intermediate scale between molecular and cellular biology. To a large part, it is still an area of interpretations and assumptions. Modeling and visualization of the mesoscale is building a bridge between the molecular and the cellular scale. Animations and carefully crafted stories support the goal of building and communicating this bridge.

Goodsell et al. [10] present how mesoscale landscapes are used to construct a visual narrative. Since the process of creating an animation is tedious and time-consuming, there are first approaches [40] to reuse animation in the form of templates. The work of McGill [29] reveals the power of visualization and animation to communicate cellular and molecular structures and dynamics. It advocates to assemble visualizations into a seamless whole, the "Visual Cell". CellPAINT [7] provides a playful way for generating and exploring models of the mesoscale, which is even suitable for non-expert users. Le Muzic et al. [26] focus on temporal molecular data and propose the use of a passive agent

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics

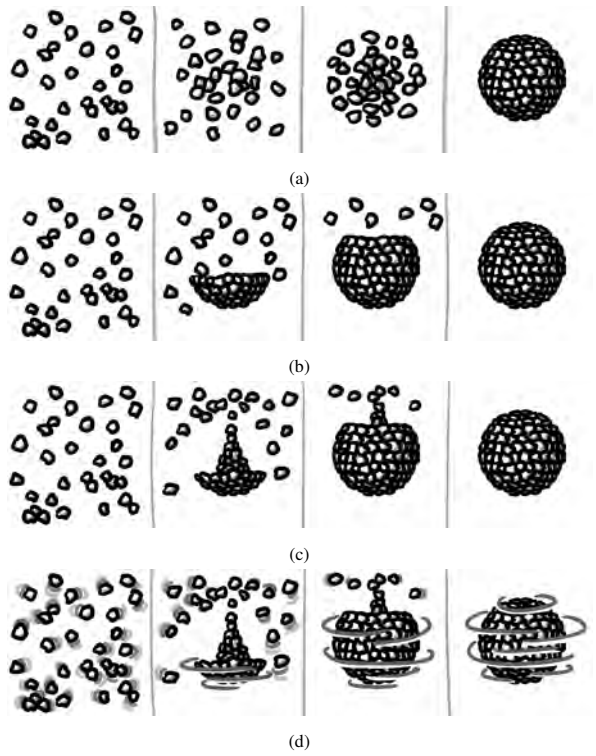


Fig. 2: (a) Four steps of a keyframed animation, where the initial state (static randomly positioned molecules) is interpolated to the target state (molecules forming a sphere). The interpolation function is independent of the molecule instance. (b) The same animation is enhanced by introducing *instance-dependent timing*. By modifying the timing of the interpolation separately for each instance, the animation is now staged. (c) An *instance-dependent transition function* is introduced. The interpolated path of each individual instance is transformed by the transition function, which can be used to model specific sequences in which the final object is built. In this case, the sphere is formed by the molecules forming a pillar, which continuously fills the sphere. (d) The same situation as in (c), but here both initial and target state are dynamic. The molecules randomly move around, while the sphere rotates around its axis.

system that controls molecular interactions according to a visual story. Recently, Iwasa [17] has reviewed different ways on how 3D animation software plays a valuable role in visualizing and communicating macromolecular structures and their dynamics.

The work of this paper goes beyond the presented approaches by tackling the problem of procedurally animating multi-scale environments.

3 Multi-scale Procedural Animations

In this paper, we describe a procedural model, which can be used to visualize complex characteristics of a biological process concerning microtubule dynamics. However, to make this approach generalizable to other applications, we first describe a conceptual framework within which our procedural dynamic model will be designed. As we show in this paper, this conceptual framework can be used to create procedural models of the dynamic behaviour in biological systems. This is an important step towards utilizing multi-scale molecular visualization not only to convey *structure*, but also *function*.

There are various approaches to create animations in molecular visualization. From the animation point of view, the most simple one is *simulation*. Sometimes, it is possible to simulate a certain environment, and the outcome of the simulation is a time-series of positions and rotations of the involved objects. The animation is created by simply displaying the time-series. However, this is not always possible, as the intrinsic properties of the environment might be unknown, or too complex, to the level where the simulation becomes infeasible. Another problem with simulations is that they cannot be easily steered by the animator to reveal certain specific aspects of the environment. Therefore, simulation might not be the most optimal choice for storytelling.

An alternative to simulation for creating molecular animations is *keyframing*. It refers to creating snapshots of the state of the environment in discrete time steps, and subsequently interpolating between them to reconstruct the state of the environment in all the missing points in time.

When using keyframing, the transformation (position and rotation) $x_{m,t}$ of an instance (in our case, a molecule) m at time t is given by:

$$x_{m,t} = mix(k_n(m), k_{n+1}(m), i) \tag{1}$$

where $k_n(m)$ is the transformation of m in the n-th keyframe and mix is the interpolation function parametrized with the interpolation parameter i. In case of simple keyframing, $i = t$, where t is the current time in the animation.

An example of a keyframed animation is shown in Fig. 2a. The figure depicts four steps of an animation. An initial keyframe (left) consisting of randomly scattered molecules and the interpolation (middle) with the target keyframe (right), where the molecules are organized into a sphere. By interpolating these keyframes, all the molecules are moved and rotated to their destination. While the animation is continuous and results in the desired outcome, it might be unrealistic in certain situations. For instance, in biology, the assemblies of larger structures are often not synchronized in a way that all molecules arrive to their destination at the same time, but rather they attach to the structure one by one.

The problem is that in the sketched situation, there are inherently two distinct scales - the behaviour of molecules is independent from the behaviour of the assembled sphere. The keyframed animation captures the environment correctly, but it cannot distinguish between the scales. During the interpolation, it applies the same rules across all the scales, thus, yielding unexpected results when applied to environments that are inherently multi-scale.

In order to support keyframe animation in multi-scale, multi-instance environments, which are ubiquitous in molecular biology, we propose the following three new concepts:

Instance-Dependent Timing A keyframed animation can be enhanced to support complex behaviour of molecular structures by modifying the interpolation function used to transition between two subsequent keyframes. The interpolation function can be made instance-dependent, as described by Sorger at al. [40]. Our approach is to assign timestamp values $t_{0,m}$ and $t_{1,m}$ to each instance m. The timestamp $t_{0,m}$ determines when the interpolation for the instance m should start, while the timestamp $t_{1,m}$ determines when it should end. Subsequently, the interpolation formula from Equation 1 is replaced by:

$$x_{m,t} = mix(k_n(m), k_{n+1}(m), i_m) \tag{2}$$

where

$$i_m = (t - t_{0,m})/(t_{1,m} - t_{0,m}) \tag{3}$$

with i_m being an instance-specific interpolation parameter.

In this way, it is possible to stage the animation for individual molecules, without the necessity of creating more keyframes than the initial and the target state. The timestamps $t_{0,m}$ and $t_{1,m}$ can be calculated procedurally for every instance. In this paper, we refer to this concept as *instance-dependent timing* (IDT) (Fig. 2b).

Instance-Dependent Transition Function While the IDT ensures that the molecules arrive to their destinations in the right order, their trajectory might still be wrong. Therefore, we introduce an *instance-dependent transition function* (IDTF) $f(x,m)$, which modifies the interpolation trajectory individually for each instance (Fig. 2c). The interpolation formula from Equation 2 becomes:

$$x_{m,t} = f(mix(k_n(m), k_{n+1}(m), i_m), m) \tag{4}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics

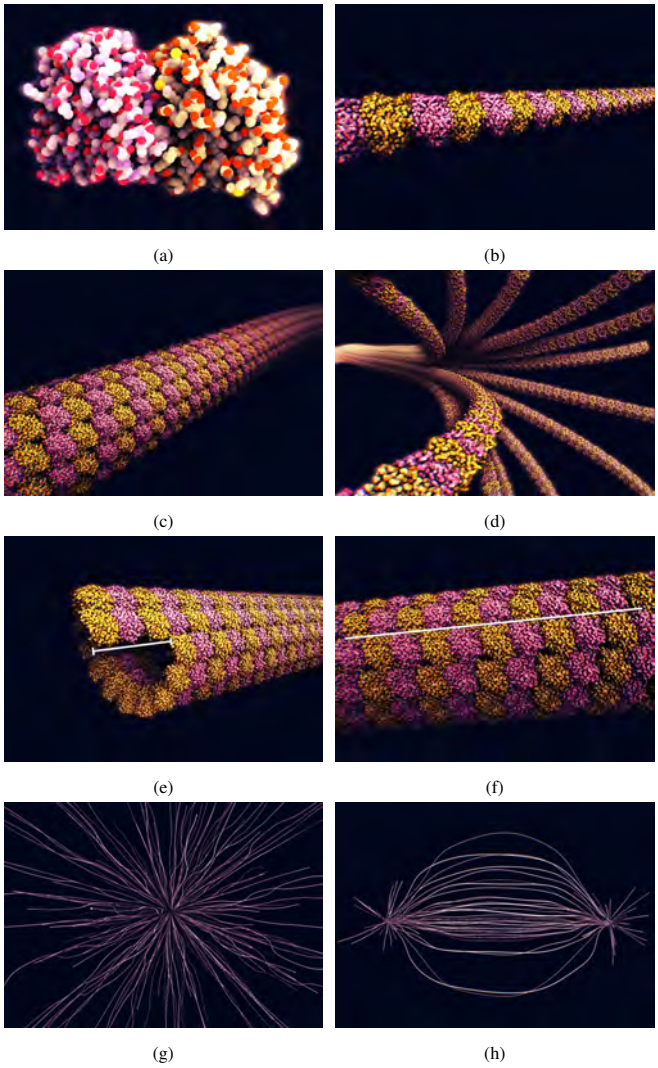


Fig. 3: (a) A tubulin dimer, consisting of two molecules - α-tubulin (purple) and β-tubulin (yellow). (b) Tubulins polymerized into a protofilament. (c) Protofilaments forming a microtubule. (d) Assembled microtubule. (e) Protofilament offset, marked with a line, forming a pseudo-helical structure of the microtubule. (f) The seam on the microtubule (white line). It is visible when the dimers are misaligned, which is caused by odd-numbered offsets. (g) A model of microtubules growing in-vitro from an isolated centrosome. (h) A model of a spindle apparatus.

Time-Varying Keyframes Fig. 2a - 2c show situations where we create an animated transition between two static states. Each of these states can be described by a simple keyframe. Now, we want to create an animated transition between two *dynamic states*. As an example, we assume an animated transition between a set of *moving* molecules and molecules organized into a rotating sphere (Fig. 2d). In this case, instead of static keyframes, we introduce *time-varying keyframes* (TVKs). A TVK is a time-dependent function $v_n(m,t)$, which specifies the transformation of the instance m at time t, in the n-th dynamic state describing our environment. In this case, v_0 describes the Brownian motion of the free-floating molecules (the initial state), and v_1 describes the molecules assembled into the sphere rotating around its axis.

To apply time-varying keyframes, the interpolation formula from Equation 4 becomes:

$$x_{m,t} = f(mix(v_n(m,t), v_{n+1}(m,t), i_m), m) \quad (5)$$

In this way, the two dynamic states are interpolated in the same way, as shown in Fig. 2d, while the dynamic behaviour on the two distinct scales is preserved. TVKs can be realized either as simulations, procedural descriptions of the dynamic states, or separate keyframe animations.

Using IDT, IDTF, and TVK, we can create keyframe animations of multi-scale, instance-based environments. This is otherwise extremely tedious with classic keyframe animation. Since we are applying interpolation to dynamic states, the velocities of the currently interpolated molecules can be distorted. In this type of animation, we are trading precision for control (as opposed to simulation, where we trade control for precision). Therefore, this type of animation is suitable for storytelling.

4 Biological Background

In this paper, we describe a multi-scale procedural model of microtubules. This particular molecular structure, found in all eukaryotic cells, is heavily researched in biology. The name microtubule was initially introduced by Slautterback [37] and their ubiquity has been revealed by Porter [34]. To communicate the structure and the function of microtubules, biologists typically use diagrams and computer generated animations. In this section, we describe the biological background necessary to understand how a procedural model of microtubules can be designed and built.

4.1 Microtubule Architecture

Microtubules are one of the building blocks of cells. They are part of the *cytoskeleton* - a complex system of protein filaments, that are carrying out various functions essential for the life of the cell. Microtubules help maintain the shape of the cell, and they act as links along which material can be transported and communication signals can be sent between different parts of cell. In eukaryotic cells, one of the crucial tasks of microtubules is to form a scaffold that helps to separate the genetic information into the daughter cells, when the cell divides. This scaffold is referred to as spindle apparatus and it only exists during the cell division.

Microtubules consist of molecules of a protein called *tubulin* (Fig. 3a). Tubulin is a dimer consisting of two tightly bound molecules - α-tubulin and β-tubulin. Tubulin molecules are dissolved in the intracellular fluid, or *cytosol*. They move around through Brownian motion caused by collisions with fast-moving fluid molecules. Upon coming into contact with each other, tubulin molecules polymerize into long chains called *protofilaments* (Fig. 3b).

Protofilaments, pushed around by the Brownian motion, tend to join into hollow tubes, which are referred to as microtubules (Fig. 3c). In eukaryotic cells, (e.g., human cells), microtubules consist of 13 protofilaments (Fig. 3d). In the most common configuration, the protofilaments

Fig. 4: Dynamic instability of microtubules. GTP-bound tubulins are added to the growing (+) end of a microtubule. GTP in the microtubule hydrolyzes into GDP. When the GDP tubulins reach all the way to the end of the microtubule, the microtubule starts to break down (catastrophe). When enough GTP tubulins are added to the shrinking end of the microtubule, the disassembly stops (rescue) and the cycle repeats. Adapted from Calligaris et al. [4].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics

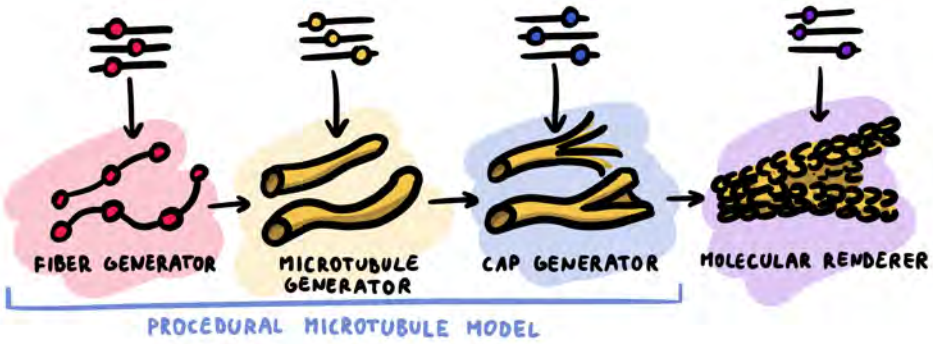


Fig. 5: Components of the pipeline for generating multi-scale visualizations of microtubules and their dynamics. First, the *fiber generator* lays out the spatial arrangement of the microtubules. This is done by generating a set of control points for each microtubule, which are then interpolated by a cubic spline. Afterwards, models of tubulin molecules are placed along each fiber in the *microtubule generator*. Subsequently, the *cap generator* displaces the tubulin molecules so that a microtubule cap is created. Finally, the model is displayed by a *molecular renderer*. Each of the modules has its own set of parameters. The blue box marks the modules belonging to the procedural microtubule model.

associate with a longitudinal offset of three tubulin monomers, forming a pseudo-helical structure of the microtubule (Fig. 3e). Since the offset length is three monomers, and a single tubulin dimer consists of two monomers, there is a seam on the microtubule. At the seam, α-tubulin and β-tubulin molecules laterally interact (Fig. 3f), as opposed to the rest of the microtubule, where α-tubulins and β-tubulins of the neighboring protofilaments are aligned.

4.2 Dynamic Instability

In order to be able to carry out all of their functions, microtubules have to be highly dynamic. They are continuously assembled and disassembled according to the current needs of the cell. The process of continuous growth and shrinkage of the microtubules is referred to as *dynamic instability*. Despite intensive study, this process has yet to be fully understood.

When assembling into a microtubule, α-tubulins are bound to β-tubulins already integrated into the microtubule. Therefore, one end of the microtubule has always the α-tubulins exposed, while the other end has the β-tubulins exposed. The end where α-tubulins are exposed is designated (-), while the one where β-tubulins are exposed is designated (+). In living eukaryotic cells, the (-) ends are anchored to *microtubule organizing centers* (*centrosomes*), while the (+) ends are continuously growing and shrinking, and in this way, exploring the space. *In vivo*, in the presence of a single, isolated centrosome, tubulin molecules polymerize into microtubules radiating away from the centrosome in all directions (Fig. 3g).

A diagram of dynamic instability is shown in Fig. 4. The free tubulin molecules dissolved in the intracellular fluid are bound to a molecule of *guanosine triphosphate* (GTP). The GTP-bound tubulins are added to the (+) end of a microtubule through their stochastic movement. Some time after joining the microtubule, the GTP molecule bound to the β-tubulin breaks down into *guanosine diphosphate* (GDP), in a process called *hydrolysis*. Protofilaments consisting of GDP-bound tubulins tend to bend outwards, as shown in Fig. 3d. It is believed that the GTP-bound tubulins at the (+) end of the microtubule hold the structure together. When the continuing hydrolysis catches up with the (+) end, the growth of the microtubule stops. This event is called *catastrophe*.

After the catastrophe, the microtubule separates into individual protofilaments, which then continue to break down into individual tubulin dimers. This rapid disassembly can be stopped when a sufficient number of GTP-bound tubulins are added to the (+) end, forming a GTP cap that will stabilize the microtubule. This event is called a *rescue*, and after it the microtubule starts to grow again.

Dynamic instability allows the microtubules to be reassembled into structures, which are needed in the given life phase of the cell. One example of this is the process of cell division. In this phase, the genetic information stored in the nucleus of the cell is copied, and the two copies need to be spatially separated. For this purpose, microtubules reassemble into a structure called *spindle apparatus* (Fig. 3h). This is achieved by moving two microtubule organizing centers into opposite sides of the cell, from where two sets of microtubules grow towards the copied genetic material. Each copy is then bound to one set of microtubules, which then act as railways for transporting each copy towards different sides of the cell. Subsequently, the cell can be divided into two new cells.

5 Multi-scale Visualization of Microtubule Dynamics

Microtubules form an inherently multi-scale structure. They span across the entire cell and can grow up to 50 micrometers. On this cellular scale, they form complex filament networks. On the molecular scale, they consist of polymerized molecules of tubulin, organized in a pseudo-helical way. Finally, on the atomic scale, processes such as hydrolysis of GTP into GDP modify the geometry of the molecular bonds, which causes the dynamic behaviour of microtubules.

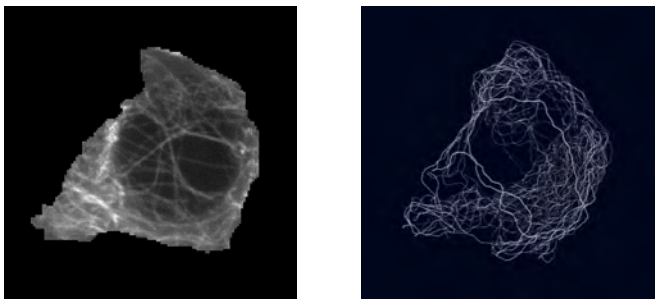
Biologists study processes happening on all of these scales. To provide a robust tool for creating visualizations of their findings and theories about these processes, we propose a multi-scale procedural model of microtubules. We integrated this model into a pipeline, which allows the users to create visualizations of microtubules, either by providing the measured data as input parameters, or by manually exploring the parameter space. This makes the model useful for both domain experts and laypersons interested in microtubules. The pipeline is illustrated in Fig. 5.

5.1 Fiber Generator

The first module in the pipeline for generating the microtubule visualizations is the fiber generator. This module generates spatial curves representing individual microtubules on the cellular scale. The curves are bound to a 3D volume, representing the cytoplasm of the cell. The fiber generator creates the points of the curve by a random walk. It originates from a single point representing the microtubule organizing center. A random walk is a stochastic process, used to model polymer chains, such as microtubules, as well as other real-world phenomena. Random walks are often applied to model growth or movement, like Brownian motion.

The flexibility of polymer chains is typically described with the persistence length. It describes how orientational correlations decay along the chain. Microtubules are tubulin polymers and considered to be rigid over cellular dimensions [8]. Random walks inherently lend themselves to incorporate flexibility constraints into the generation process. At each step, the random walk chooses a new direction. To map a random walk to a given persistence length, the new direction of a step is constrained by the direction of the previous step. In the case of an unconstrained random walk, the new direction is chosen from a sphere surrounding the current point. In the constrained case, only a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics



(a) (b)

Fig. 6: (a) A fluorescence microscopy image of microtubules in a human-induced pluripotent stem cell. (b) Microtubule fibers generated using a segmentation mask from the scan of the same cell.

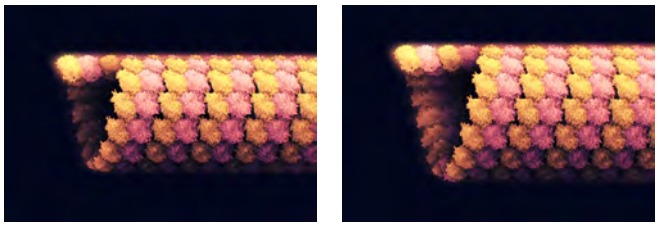
cap of the sphere is used. The center of the cap is computed from the previous direction and its size results from the given persistence length. Additional barriers limit the growth of the random walk with the result that the fiber is generated only inside of a compartment, e.g. a cell.

In Fig. 6, we show an application of the fiber generation approach that produces the curves of the microtubules. To generate a microtubule network inside a cell, we use a segmented 3D volume of a human-induced pluripotent stem cell. It has been scanned with fluorescence microscopy to restrain the generated curves within the cell cytoplasm. The segmentation mask provides the barrier for the random walk by a binary definition that reveals if the location of a step is valid, i.e. inside of the compartment, or invalid, i.e. outside of the compartment. The compartment is defined by the shape of the cell, but also by the including organelles like the nucleus. In case a random walk step is outside, the algorithm backtracks and tries again until a valid location is found.

5.2 Microtubule Generator

The microtubule generator creates the details of the microtubules on the molecular scale. It takes the curves generated by the fiber generator, and populates them with 3D models of the tubulin molecule from the PDB [1]. This is achieved by resampling the curve into uniformly spaced control points, and by generating a ring of tubulins around each point. The tubulins are aligned with the tangent of the curve at a given point. The number of tubulins along a ring and its longitudinal offset can be specified by the user. In this way, it is possible to generate microtubules of various architectures. Fig. 7 shows examples of two different microtubule architectures.

After the tubulin positions are generated, they are perturbed by a low-frequency, low-amplitude noise function to achieve a more natural look. Both frequency and amplitude should be chosen carefully, so that the solid structure of the microtubule is maintained. Fig. 8a shows a microtubule without the low-frequency noise pertubation, while Fig. 8b shows a microtubule with the noise applied.



(a) (b)

Fig. 7: Generated microtubules with two different architectures. (a) 13 protofilaments with the longitudinal offset of three monomers. (b) 16 protofilaments with the longitudinal offset of four monomers.

We use a two-dimensional *value noise* (linearly interpolated random points), where the first dimension is mapped to the length of the microtubule, while the second dimension can be mapped to time. This creates an animation with slight movements of the microtubule. Such movements naturally exist as a result of interactions with the fast-moving molecules of the surrounding medium, such as the cytosol. The advantage of value noise is that it can be easily calculated and its frequency can be specified.

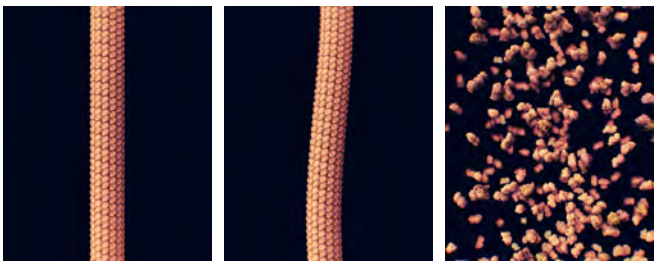
5.3 Cap Generator

The fiber generator and the microtubule generator are responsible for creating the structure of the microtubule network on the cellular and molecular scale, respectively. The third module of the pipeline, the cap generator, is responsible for creating the dynamic behaviour of individual microtubules. It generates the transition between assembled and disassembled microtubules. To make this transition realistic, we designed the cap generator according to the concept of multi-scale procedural animations introduced in Section 3. The cap generator constitutes the core contribution of the procedural microtubule model.

A microtubule is assembled from tubulin molecules, which are quickly moving around through Brownian motion. When tubulin molecules are associated to the growing microtubule, they form a structure on the growing end which we refer to as cap. One of the popular theories of how the cap looks like states that the tubulins form a flat sheet of protofilaments. The sheet quickly zips up into the tubular shape, while new tubulins are associated to the end of the cap. In this way, the microtubule grows. Similarly, when the microtubule shrinks, the tubulins are dissociated from the cap and move to the surrounding space. First, the individual protofilaments bend away from the microtubule, and then they dissolve into individual tubulins. Both growing and shrinking caps are shown in Fig. 9.

The environment of the microtubules consists of two dynamic states. One state is the Brownian motion of the free tubulin molecules. The other state is the assembled microtubule, which moves slightly around due to the collisions with the surrounding molecules. We describe the two dynamic states of the environment in a procedural way to model the assembly and disassembly process. The model of the microtubule growth and shrinking is built as interpolation between these two dynamic states, or TVKs. However, as the tubulins are associating to the microtubule and dissociating from it in a specific way, we have to use IDT and IDTF, as illustrated in Fig. 2d.

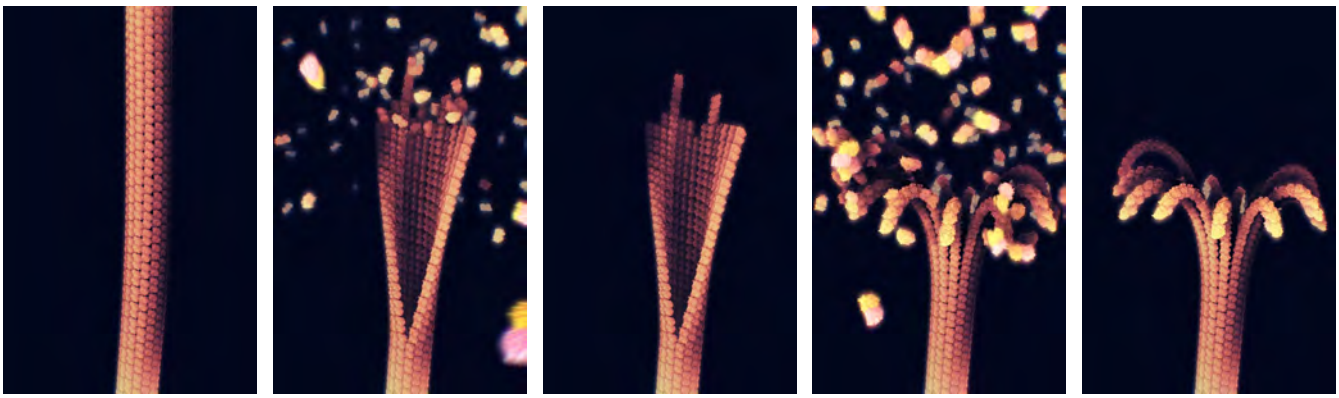
The procedural description of the assembled microtubule comes directly from the microtubule generator. Let us refer to it as v_1. We utilize it as our target TVK. The initial TVK, v_0, generates the fast stochastic movement of free tubulin molecules. Here, we perturb the positions and rotations of the molecules produced by the microtubule generator with a noise function. Similarly to the perturbation with the low-frequency noise, shown in Fig. 8b, we again use a two-dimensional value noise. While the first dimension is mapped to the tubulin position along the microtubule, the second dimension is mapped to time. The frequency in the first dimension is high enough so that the microtubule structure



(a) (b) (c)

Fig. 8: (a) A microtubule generated along a straight line. (b) A low-frequency noise applied to the microtubule to produce a slight random bending. (c) A high-frequency noise applied to the microtubule to produce the state before the microtubule assembles.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics



(a) (b) (c) (d) (e)

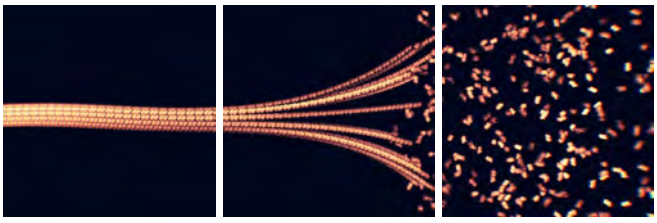
Fig. 9: (a) A microtubule produced by the microtubule generator. (b) A growing microtubule cap. (c) The growing cap with the free tubulins removed. (d) A shrinking microtubule cap. (e) The shrinking cap with the free tubulins removed.

is completely disintegrated, and the moving tubulins appear as being dissolved in the surrounding medium. The same noise perturbation is applied to the rotation of the molecules as well. The result of this transformation is shown in Fig. 8c.

In order to interpolate between v_0 and v_1, correctly visualizing the microtubule growth, we need to split a microtubule into three parts - an assembled part, a cap, a disassembled part (Fig. 10). In the assembled part, all the molecules are transformed by v_1, while in the disassembled part they are transformed by v_0. The molecules in the middle part, or the cap, currently undergo interpolation. The splitting is achieved by defining IDT for all the molecules according to their position along the microtubule. The timestamps $t_{0,m}$ and $t_{1,m}$, as defined in Equation 3, are procedurally set in a way that in the assembled part $t_{1,m} \leq t$, in the cap part $t_{0,m} \leq t < t_{1,m}$, and in the disassembled part $t_{1,m} > t$, where t is the current time. The differences between $t_{0,m}$ and $t_{1,m}$ depend on the input parameter defining the length of the cap.

The next step is to define the trajectory along which the molecules are associated to the microtubule. We do this by defining an IDTF for both the growing and the shrinking cap (Fig. 9). This is done as a geometrical transformation based on the molecule position p along the microtubule and the ID r of the protofilament it belongs to.

For the growing cap, the molecules are positioned on a flat sheet where the longitudinal position within the sheet is proportional to p, while the lateral position is proportional to r. In this way, the protofilaments are laid out next to each other. The sheet is slightly bent away from the central line of the microtubule by rotating the molecules along an axis perpendicular to the microtubule and parallel to the sheet. The axis is positioned where the cap joins the microtubule. The bending is perturbed by a low-frequency noise function to create the appearance that the growing cap is flapping due to the molecular collisions. Finally, the molecule positions within the sheet are interpolated to the positions in the assembled microtubule, to create a continuous zipping up effect.



(a) (b) (c)

Fig. 10: The interpolation between the (a) assembled and (c) disassembled states of a microtubule. The cap's middle part (b) is formed by the molecules currently undergoing the interpolation due to a specific IDT.

The IDTF for the shrinking cap is defined in a way that each molecule in the cap is rotated around the axis perpendicular to the microtubule center line and parallel to the protofilament r. This means the molecules in the same protofilament will rotate around the same axis. The amount of rotation is proportional to p. This creates the effect of individual protofilaments bending away from the microtubule - an effect characteristic for GDP-bound tubulins.

To achieve a realistic animation, it is important that the molecules are always associated only to the end of the individual protofilaments. This is achieved by selecting IDT timestamps proportionally to the position of the given molecule within the microtubule. This means that the growing or shrinking end of the microtubule is achieved by the synchronized timings of their IDTF.

The IDT timestamps define the duration of the interpolation for each molecule. Since the molecules are forced to move a certain distance in this duration, as defined by the IDTF, the speed of the molecule undergoing the interpolation might differ from the diffusion movement defined in v_0. These visual artifacts are minimized by the fact that the v_0 positions (dissolved molecules) are created by preturbing the positions in v_1 (assembled microtubule) by a noise function. This means the dissolved molecules, despite appearing randomly scattered throughout the environment, are at all times relatively close to their target position within the assembled microtubule. The molecules do not have to travel large distances during the interpolation, thus reducing visual artifacts caused by it.

With the IDT, IDTF, and TVKs defined, we can animate the growing and the shrinking of a microtubule by remapping the current time t to the microtubule length (Fig. 12). The IDTF for the growing or shrinking cap is based on whether the graph of the length over time is ascending or descending at that point. To avoid sudden changes in the shape of the cap, we can interpolate between both IDTFs when growing changes to shrinking and vice versa, as shown in Fig. 11.

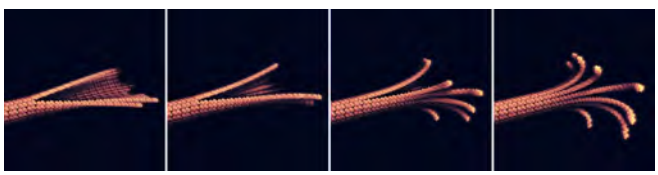


Fig. 11: Four frames of the transition between the growing and the shrinking cap of the microtubule. This is achieved by interpolating two distinct IDTFs.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics

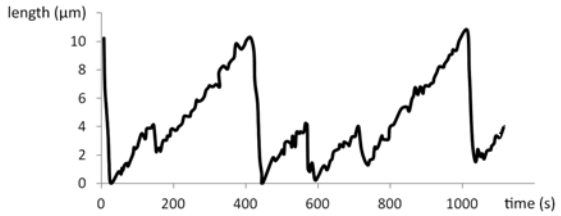


Fig. 12: Microtubule length over time. In our animation, this data is used to drive the dynamic instability. Adapted from Fygenson et al. [6].

5.4 Molecular Renderer

The last module in the pipeline is a molecular renderer. In this work, we use the molecular visualization framework Marion [30]. It supports a real-time visualization of large molecular scenes thanks to its level-of-detail scheme. It includes various graphics effects that can be used to enhance the perception of the displayed structures. In particular, we make use of screen-space ambient occlusion to enhance the perception of the macromolecular shapes of microtubules. We also employ depth-of-field and fog to improve the depth perception, motion blur for conveying the speed of the free floating molecules, and a bloom effect to highlight specific subunits of the microtubule for storytelling purposes.

Marion also implements an automatic multi-scale coloring scheme, which we use to distinguish α-tubulin and β-tubulin molecules with adaptive colors. The colors of individual tubulins are only different in close-ups, thus preventing a noisy appearance of the microtubule when rendered from a larger distance.

Marion supports a free exploration of the generated 3D scene, but also rendering the scene with pre-programmed camera paths. All the parameters of the microtubule model as well as the visualization can be keyframed. Therefore, our implementation can be used interactively to explore individual parameter settings, or non-interactively to produce animations about microtubules.

6 Results

Fig. 13 depicts visualizations generated by our pipeline. In Fig. 13a, the growing end of a microtubule is displayed. The image shows how the flat protofilament sheet closes up into the tubular shape of a microtubule. To distinguish the monomers, α-tubulins are colored in pink, while GTP-bound β-tubulins are colored in yellow.

The user can select how far along the microtubule GTP bound to β-tubulins has already hydrolysed into GDP. In Fig. 13b, the GDP-bound β-tubulins are colored in blue. The user can also define the length of the region, where the hydrolysis already started, but not all of the GTP is hydrolysed yet. Internally, the cap generator marks the GDP-bound tubulins, while the molecular renderer can assign different visual properties to them, such as different colors. Fig. 13c displays the disassembly of a microtubule. The GDP-bound β-tubulins are colored in blue. Since there is no GTP cap at the end of the microtubule, the protofilaments curl outwards and the microtubule collapses.

The application was tested using a computer with Intel Core i7-6700K CPU 4.00 GHz and NVIDIA GeForce GTX 1080 graphics card with 8 GB memory and consistently achieved interactive frame rates. This includes the computation of the model as well as the rendering.

7 Evaluation

To evaluate and discuss our procedural approach, we have collected informal feedback from several domain experts. One of the experts (P1) has more than 30 years of experience in structural biology and is skilled in scientific drawing and illustration. The second expert (P2) is a computational biologist and a certified medical illustrator with about 20 years of professional experience. This expert is specialized in the visual communication of molecular and cellular biology. The third expert (P3) studied biochemistry and biomedical visualization and has a professional background in medical animation and molecular simulation.

During the feedback session we have presented a sandbox system of the procedurally generated microtubules. In this system, a continuous animation of a growing or shrinking microtubule is shown, which is parameterizable through a graphical interface (see supplementary material).

The experts could freely explore the application and experiment with the parameters. They were asked several questions during the process. We first asked in which areas they would utilize such an application. We also inquired if they see the usefulness and effectiveness of the application for the biologists to communicate their findings. We posed questions concerning the general efficacy of the system to communicate the current knowledge about the structure of microtubules and the dynamic instability. Moreover, we wanted to know which potential they see for scientific hypothesis generation and verification.

After watching the microtuble animation, expert P1 immediately mentioned that microscopy images indicate a stronger bending of the cap than in the default setting of the application. Since the application exposes the intrinsic parameters, the bending factor could be directly changed to match the current knowledge of the expert. The expert appreciated the graphical effects that create an almost “mesmerizing” animation, however, P1 mentioned that scientists communicate their findings to peers in a more simplified way in order to emphasize only the most essential features. Furthermore, the expert pointed out that the current model of the microtubule does not facilitate the generation and validation of scientific hypotheses. More vital interactions are happening that are not yet fully supported by the model. However, P1 can clearly see the usefulness of the system for communicating the microtubule processes not only to layman, but also to scientific peers. Expert P2 was “very impressed with the realtime visualization” and thinks that the “visual effects greatly contribute to the clarity of the story”. The expert acknowledged that “this is the clearest and most informative description of the protofilaments merging theory and the overall process, and how the overall process relates to the catastrophe/rescue cycle, that I have seen”. Expert P3 emphasized the importance of interactivity in education. He mentioned that the possibility to directly change parameters definitely improves the understanding of the displayed structures. The expert also indicated the importance of showing “reality” in education instead of very abstracted illustrations. The expert said, “this is what animators should be doing instead of animations”, since it is“super tedious to build even one version of this animation”. The expert would like to see further developments of the tool, so that arbitrary proteins can be added to the scene, and to have the time scale exposed to the user. For scientific publications the expert indicated that usually more schematic illustrations are preferred instead of detailed renderings.

In addition to the expert feedback, we have used the presented application to create an educational video that we submitted to the *PacificVis Visual Data Storytelling Contest*. It was evaluated by visualization and data storytelling experts and it received the award for the best storytelling video.

8 Discussion

The expert feedback demonstrates that our approach is capable of generating measurement-driven procedural animations that illustrate clear and informative aspects of the underlying biological data. The feedback made clear that the system does not fully support all required features for scientific hypothesis testing. However, the system received positive feedback concerning communicating existing knowledge and was considered superior to existing animation approaches in this aspect.

In biology, various fibrous structures play a crucial role in living cells. These structures have multiple characteristics in common, which are addressed by our approach. These structures are often synthesized by polymerization through random Brownian motion, which means that only the growing end is affected during the synthesis while the rest of the structure remains relatively static. The collisions of the building elements do not occur as frequently as when entire structures

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2934612, IEEE Transactions on Visualization and Computer Graphics

(a) (b) (c)

Fig. 13: Visualizations generated by our pipeline. (a) shows the growing end of a microtubule, (b) microtubule with GDP-bound β-tubulins highligthed in blue, and (c) shows the disassembly of a microtubule.

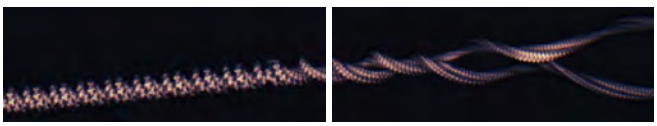
are assembled at the same time. Therefore, assigning IDT and IDTF to the building elements is a powerful strategy for creating procedural animations of these processes. While we present an application of this method to model the dynamic instability of microtubules, the method could be applied to other fibrous structures as well.

Fig. 14 displays a prototype implementation of our procedural animation approach for such a polymer (DNA). In this example we show the difference between DNA in its typical helix form (Fig. 14a) and DNA in a state where part of it is unwound (Fig. 14b). The transition between the two states serves purely illustrative purposes in order to reveal the atomic structure of the individual DNA base pairs.

The goal of our procedural animation approach and most molecular animations is not to achieve the same accuracy as simulations, but rather focus on the scientific accuracy that is needed for the phenomenon to be communicated. For instance, in the implementation of our system, we have disregarded the resolving of collision for overlapping molecules. Proper collision handling would only result in minor changes in the final visualization and have insignificant influence on the particular story. Similarly, neither water nor other small molecules are present in the visualization. If a story requires proper collision handling, related work from fluid simulation [16] has demonstrated that millions of collisions of uniformly sized particles (like atoms) can be solved in real-time.

In the short term, we imagine our multi-scale procedural animation approach to be useful for generating scientifically accurate animations of chain-structured models that consist of repeating subunits. We also see our approach suitable to enhance existing simulations. Here, either coarse-grained simulations are augmented with animations of smaller structures, or vice versa, a procedural animation is enhanced with simulations of smaller structures. The approach, however, is limited to educational purposes where the result of the simulation is supposed to be communicated instead of analyzed. For the analysis of simulations more specialized tools [3] are required. In the long term, we envision our approach as a foundation for various kinds of animations of multi-scale models, such as animations of molecular machines.

As future research work, we would like to explore the capabilities of visual interfaces in order to simplify the process of describing IDTs, IDTFs, and TVKs to generate procedural animations. The work of Sorger et al. [40] already made first steps in order to reuse and simplify the description of illustrative animations with visual interfaces in the context of molecular biology. We imagine that with such an interface our approach can be used by biological illustrators and biologist with basic programming experience.



(a) (b)

Fig. 14: Interpolation between the (a) wound and (b) unwound states of DNA.

Procedural animation in multi-scale biological environments has the potential to become a widely used rapid modeling approach for authoring dynamic phenomena. As a next step, we are investigating the possible application to processes, such as DNA replication, where DNA is read in a complex sequence of operations and copied into a new strand. Another research direction is the design of visual authoring metaphors for procedural animations. Artists and biologists should be able to easily set up animations without any programming skills.

9 Conclusion

In this paper, we present a novel concept to approach the animation of dynamic processes in multi-scale, multi-instance molecular scenes. We propose an approach to design a procedural model of microtubule dynamics. This model can be used to create scientifically-accurate 3D animations of growth and shrinking of microtubules, a biological process that is intensively studied. The model can be parameterized by measured data. Examples are association and dissociation rates of tubulins when microtubules form and break down, or the length of the microtubule cap, which changes during the growing and shrinking process. In this way, biologists can create accurate visual representations of their theories and findings of how microtubule dynamic instability evolves. The model can be used for communicating scientific results, but also for education in biology.

We have evaluated the model by creating a user interface, where the parameters can be modified. We showed the software to several domain experts in biology and biological illustration and animation. The experts complimented the visual quality of the generated microtubule visualizations, as well as their suitability as both communication and education tool. Nowadays, microtubule research relies on schematic drawings and hand-made illustrations to describe the theories of this not yet completely understood process. With our procedural model, biologists can visually inspect the 3D structure of microtubules, and how different parameters would influence it. While for hypothesis testing a simulation would be needed, our procedural model gives biologists an initial idea about the implications of the given parameter values. Since the microtubule model is generated in real-time, various parameter settings can be tried quickly. While the procedural animation is presented in the context of microtubule research, it can be effectively translated into other scenarios of multi-scale emergent biological behavior.

Acknowledgments

This work was funded under the ILLVISATION grant by WWTF (VRG11-010). It is based upon work supported by the King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR) under Award No. OSR-2019-CPF-4108 and BAS/1/1680-01-01. The paper was partly written in collaboration with the VRVis Competence Center in the scope of COMET (854174). Authors would like to thank Nanographics GmbH (nanographics.at) for providing the Marion Software Framework. Additionally, the authors wish to thank Graham Johnson and David Kouřil for the help with the implementation of the static microtubule model, and Theresia Gschwandtner for the feedback on the design of the microtubule graphics.

REFERENCES

[1] 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. doi: 10.1093/nar/28.1.235

[2] E. Brochu, T. Brochu, and N. de Freitas. A Bayesian interactive optimization approach to procedural animation design. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pp. 103–112. Eurographics Association, 2010.

[3] J. Byška, T. Trautner, S. M. Marques, J. Damborský, B. Kozlíková, and M. Waldner. Analysis of long molecular dynamics simulations using interactive focus+context visualization. *Computer Graphics Forum*, 2019. doi: 10.1111/cgf.13701

[4] D. Calligaris, P. Verdier-Pinard, F. Devred, C. Villard, D. Braguer, and D. Lafitte. Microtubule targeting agents: from biophysics to proteomics. *Cellular and Molecular Life Sciences*, 67(7):1089–1104, 2010. doi: 10.1007/s00018-009-0245-6

[5] M. Falk, M. Krone, and T. Ertl. Atomistic visualization of mesoscopic whole-cell simulations using ray-casted instancing. *Computer Graphics Forum*, 32(8):195–206, 2013. doi: 10.1111/cgf.12197

[6] D. K. Fygenson, E. Braun, and A. Libchaber. Phase diagram of microtubules. *Phys. Rev. E*, 50:1579–1588, 1994. doi: 10.1103/PhysRevE.50.1579

[7] A. Gardner, L. Autin, B. Barbaro, A. J. Olson, and D. S. Goodsell. Cellpaint: Interactive illustration of dynamic mesoscale cellular environments. *IEEE Computer Graphics and Applications*, 38(6):51–66, 2018. doi: 10.1109/MCG.2018.2877076

[8] F. Gittes, B. Mickey, J. Nettleton, and J. Howard. Flexural rigidity of microtubules and actin filaments measured from thermal fluctuations in shape. *The Journal of cell biology*, 120:923–34, 1993.

[9] D. S. Goodsell. Inside a living cell. *Trends in Biochemical Sciences*, 16:203–206, 1991. doi: 10.1016/0968-0004(91)90083-8

[10] D. S. Goodsell, M. A. Franzen, and T. Herman. From atoms to cells: Using mesoscale landscapes to construct visual narratives. *Journal of Molecular Biology*, 430(21):3954 – 3968, 2018. doi: 10.1016/j.jmb.2018.06.009

[11] S. Grottel, M. Krone, C. Müller, G. Reina, and T. Ertl. MegaMol – a prototyping framework for particle-based visualization. *IEEE Transactions on Visualization and Computer Graphics*, 21(2):201–214, 2015. doi: 10.1109/TVCG.2014.2350479

[12] P. Hermosilla, V. Guallar, A. Vinacua, and P. Vázquez. High quality illustrative effects for molecular rendering. *Computers & Graphics*, 54:113 – 120, 2016. Special Issue on CAD/Graphics 2015. doi: 10.1016/j.cag.2015.07.017

[13] P. Hermosilla, S. Maisch, P.-P. Vzquez, and T. Ropinski. Improving perception of molecular surface visualizations by incorporating translucency effects. In *Eurographics Workshop on Visual Computing for Biology and Medicine*. The Eurographics Association, 2018. doi: 10.2312/vcbm.20181244

[14] P. Hermosilla, P.-P. Vázquez Alcocer, A. Vinacua, and T. Ropinski. A general illumination model for molecular visualization. *Computer Graphics Forum*, 37:367–378, 2018. doi: 10.1111/cgf.13426

[15] D. Hinsinger, F. Neyret, and M.-P. Cani. Interactive animation of ocean waves. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, pp. 161–166. ACM, New York, NY, USA, 2002. doi: 10.1145/545261.545288

[16] R. Hoetzlein. Fast fixed-radius nearest neighbors: Interactive million-particle fluids. In *GPU Technology Conference (GTC)*, 2014.

[17] J. H. Iwasa. Bringing macromolecular machinery to life using 3d animation. *Current Opinion in Structural Biology*, 31:84 – 88, 2015. Theory and simulation/Macromolecular machines and assemblies. doi: 10.1016/j.sbi.2015.03.015

[18] G. T. Johnson, L. Autin, M. Al-Alusi, D. S. Goodsell, M. F. Sanner, and A. J. Olson. cellPACK: a virtual mesoscope to model and visualize structural systems biology. *Nature methods*, 12(1):85–91, 2015.

[19] G. T. Johnson, L. Autin, D. S. Goodsell, M. F. Sanner, and A. J. Olson. ePMV embeds molecular modeling into professional animation software environments. *Structure*, 19(3):293–303, 2011.

[20] G. T. Johnson and S. Hertig. A guide to the visual analysis and communication of biomolecular structural data. *Nature Reviews Molecular Cell Biology*, 15(10):690–698, 2014.

[21] T. Klein, L. Autin, B. Kozlíková, D. S. Goodsell, A. Olson, M. E. Gröller, and I. Viola. Instant construction and visualization of crowded biological environments. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):862–872, 2018. doi: 10.1109/TVCG.2017.2744258

[22] B. Kozlíková, M. Krone, M. Falk, N. Lindow, M. Baaden, D. Baum, I. Viola, J. Parulek, and H.-C. Hege. Visualization of biomolecular structures: State of the art revisited. *Computer Graphics Forum*, 2016. doi: 10.1111/cgf.13072

[23] L. Krecklau, J. Born, and L. Kobbelt. View-dependent realtime rendering of procedural facades with high geometric detail. *Computer Graphics Forum*, 32(2pt4):479–488, 2013. doi: 10.1111/cgf.12068

[24] K. Lawonn, M. Krone, T. Ertl, and B. Preim. Line integral convolution for real-time illustration of molecular surface shape and salient regions. *Computer Graphics Forum*, 2014. doi: 10.1111/cgf.12374

[25] M. Le Muzic, L. Autin, J. Parulek, and I. Viola. cellVIEW: a tool for illustrative and multi-scale rendering of large biomolecular datasets. In *Eurographics Workshop on Visual Computing for Biology and Medicine*, pp. 61–70. EG Digital Library, The Eurographics Association, 2015.

[26] M. Le Muzic, J. Parulek, A.-K. Stavrum, and I. Viola. Illustrative visualization of molecular reactions using omniscient intelligence and passive agents. *Computer Graphics Forum*, 33, 2014. doi: 10.1111/cgf.12370

[27] N. Lindow, D. Baum, and H.-C. Hege. Interactive rendering of materials and biological structures on atomic and nanoscopic scale. *Computer Graphics Forum*, 31(3pt4):1325–1334, 2012. doi: 10.1111/j.1467-8659.2012.03128.x

[28] T. Luft, C. Colditz, and O. Deussen. Image enhancement by unsharp masking the depth buffer. *ACM Trans. Graph.*, 25(3):1206–1213, 2006. doi: 10.1145/1141911.1142016

[29] G. McGill. Molecular movies coming to a lecture near you. *Cell*, 133(7):1127 – 1132, 2008. doi: 10.1016/j.cell.2008.06.013

[30] P. Mindek, D. Kouřil, J. Sorger, D. Toloudis, B. Lyons, G. Johnson, M. E. Gröller, and I. Viola. Visualization multi-pipeline for communicating biology. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):883–892, 2018. doi: 10.1109/TVCG.2017.2744518

[31] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool. Procedural modeling of buildings. *ACM Trans. Graph.*, 25(3):614–623, 2006. doi: 10.1145/1141911.1141931

[32] F. K. Musgrave, C. E. Kolb, and R. S. Mace. The synthesis and rendering of eroded fractal terrains. *SIGGRAPH Comput. Graph.*, 23(3):41–50, 1989. doi: 10.1145/74334.74337

[33] Y. I. H. Parish and P. Müller. Procedural modeling of cities. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pp. 301–308. ACM, New York, NY, USA, 2001. doi: 10.1145/383259.383292

[34] K. R. Porter. *Cytoplasmic Microtubules and Their Functions*, chap. 13, pp. 308–356. John Wiley & Sons, Ltd, 2008. doi: 10.1002/9780470719442.ch13

[35] D. Rohmer, T. Popa, M.-P. Cani, S. Hahmann, and A. Sheffer. Animation wrinkling: Augmenting coarse cloth simulations with realistic-looking wrinkles. *ACM Trans. Graph.*, 29(6):157:1–157:8, 2010. doi: 10.1145/1882261.1866183

[36] J. Schpok, J. Simons, D. S. Ebert, and C. Hansen. A real-time cloud modeling, rendering, and animation system. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pp. 160–166. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2003.

[37] D. B. Slautterback. Cytoplasmic microtubules. *The Journal of Cell Biology*, 18(2):367–388, 1963. doi: 10.1083/jcb.18.2.367

[38] R. Smelik, T. Tutenel, K. J. de Kraker, and R. Bidarra. A declarative approach to procedural modeling of virtual worlds. *Computers & Graphics*, 35:352–363, 04 2011. doi: 10.1016/j.cag.2010.11.011

[39] J. Sorger, P. Mindek, T. Klein, G. Johnson, and I. Viola. Illustrative Transitions in Molecular Visualization via Forward and Inverse Abstraction Transform. In S. Bruckner, B. Preim, A. Vilanova, H. Hauser, A. Hennemuth, and A. Lundervold, eds., *Eurographics Workshop on Visual Computing for Biology and Medicine*. The Eurographics Association, 2016. doi: 10.2312/vcbm.20161267

[40] J. Sorger, P. Mindek, P. Rautek, M. E. Gröller, G. Johnson, and I. Viola. Metamorphers: Storytelling templates for illustrative animated transitions in molecular visualization. In *Proceedings of the 33rd Spring Conference on Computer Graphics*, SCCG '17, pp. 2:1–2:10. ACM, New York, NY, USA, 2017. doi: 10.1145/3154353.3154364

[41] M. Steinberger, M. Kenzel, B. Kainz, J. Müller, W. Peter, and D. Schmalstieg. Parallel generation of architecture on the GPU. *Computer Graphics Forum*, 33(2):73–82, 2014. doi: 10.1111/cgf.12312

[42] M. Steinberger, M. Kenzel, B. Kainz, P. Wonka, and D. Schmalstieg. On-

the-fly generation and rendering of infinite cities on the GPU. *Computer Graphics Forum*, 33(2):105–114, 2014. doi: 10.1111/cgf.12315

[43] M. Tarini, P. Cignoni, and C. Montani. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1237–1244, 2006. doi: 10.1109/TVCG.2006.115

[44] M. van der Zwan, W. Lueks, H. Bekker, and T. Isenberg. Illustrative molecular visualization with continuous abstraction. *Computer Graphics Forum*, 30(3):683–690, 2011. doi: 10.1111/j.1467-8659.2011.01917.x

[45] O. Šťava, B. Beneš, R. Měch, D. G. Aliaga, and P. Krištof. Inverse procedural modeling by automatic generation of L-systems. *Computer Graphics Forum*, 29(2):665–674, 2010. doi: 10.1111/j.1467-8659.2009.01636.x

[46] N. Waldin, M. Waldner, M. Le Muzic, M. E. Gröller, D. S. Goodsell, L. Autin, A. J. Olson, and I. Viola. Cuttlefish: Color mapping for dynamic multi-scale visualizations. *Computer Graphics Forum*, 2019. Preprint. doi: 10.1111/cgf.13611

[47] B. Watson, P. Müller, O. Veryovka, A. Fuller, P. Wonka, and C. Sexton. Procedural urban modeling in practice. *IEEE Computer Graphics and Applications*, 28(3):18–26, 2008. doi: 10.1109/MCG.2008.58

[48] B. G. Wilhelm, S. Mandad, S. Truckenbrodt, K. Kröhnert, C. Schäfer, B. Rammner, S. J. Koo, G. A. Claßen, M. Krauss, V. Haucke, H. Urlaub, and S. O. Rizzoli. Composition of isolated synaptic boutons reveals the amounts of vesicle trafficking proteins. *Science*, 344(6187):1023–1028, 2014. doi: 10.1126/science.1252884

Curriculum Vitae

Tobias Klein

Education

05/2016 – Present **Ph.D. Candidate in Visual Computing**, *Thesis: Instant Construction of Atomistic Models for Visualization in Integrative Cell Biology*, TU Wien, Austria.

03/2012 – 11/2015 **M.Sc. in Visual Computing**, *Thesis: Towards Interactive Visual Exploration of Parallel Programs using a Domain-specific Language*, TU Wien, Austria.

10/2006 – 07/2011 **B.Sc. in Computer Science, emphasis in Media Studies**, *Thesis: Gesture Recognition for Virtual Reality Applications*, University of Siegen, Germany.

Experience

Vocational

11/2018 – Present **CEO**, NANOGRAPHICS, Vienna, Austria.

11/2017 – Present **Co-Founder**, NANOGRAPHICS, Vienna, Austria.
Manage operations and implementation of technology for 3D visualization for life sciences.

08/2016 – Present **University Assistant**, TU WIEN, Vienna, Austria.
Research in molecular visualization.

05/2016 – 07/2016 **Project Assistant**, TU WIEN, Vienna, Austria.
Research in molecular visualization.

04/2016 – 05/2016 **Remote Research Consultant**, KAUST, Vienna, Austria.
Preparing research prototype for open release.

10/2015 – 04/2016 **Research Visit**, KAUST, Thuwal, Saudi Arabia.
Work towards publication and talk at NVIDA GPU Technology Conference.

02/2015 – 06/2015 **Visiting Student**, KAUST, Thuwal, Saudi Arabia.
Work on master's thesis. (C++, OpenCL, Java Script, D3, Qt)

09/2014 – 01/2015 **Research Intern**, VRVIS, Vienna, Austria, Biomedical Visualization Group.
Work within a research team in the field of Visual Computing. Responsible for the analysis of web-based solutions for multi-channel volume rendering. (WebGL)

06/2011 – 02/2012 **Research Intern**, SIEMENS CORPORATE RESEARCH, Princeton, USA.
Responsible for the extension of a tool for rapid and extensible software development in medical imaging. (C++, OpenInventor, Qt)

Lindengasse 31-33/1/8 – Vienna, Austria 1070
+43 676 467 9510 • tklein@cg.tuwien.ac.at
** 24.01.1986, Siegen, Germany*

01/2009 – 01/2011 **Student Assistant**, University of Siegen, Siegen, Germany.
Assistance in the SiDiff Project, a meta model-independent research approach for model comparison. (Java, Eclipse Modeling Framework)

Teaching Assistant (TU Wien)

10/2014 – 02/2015 Introduction to Pattern Recognition (MATLAB)
10/2014 – 02/2015 Introduction to Digital Imaging (MATLAB)
03/2014 – 07/2014 Introduction to Visual Computing (MATLAB, Blender)
10/2013 – 02/2014 Introduction to Computer Graphics (Java)
10/2013 – 02/2014 Introduction to Digital Imaging (MATLAB)
03/2013 – 07/2013 Introduction to Visual Computing (MATLAB, Blender)

Teaching Assistant (University of Siegen)

10/2010 – 01/2011 Computer Graphics II
04/2010 – 07/2010 Practical course of Computer Graphics (C++, OpenGL, GLSL)
10/2009 – 01/2010 Database Management Systems I (XML, SQL)

Volunteer Work

06/2018 Volunteer at the EuroVis 2018 conference in Brno, Czech Republic
10/2010 Student volunteer at the Vision, Modeling and Visualization Workshop 2010 (VMV 2010) in Siegen, Germany

Selected Talks

2019 **KAUST Cell Summit**, *Instant Construction for Visualization in Integrative Cell Biology*, Saudia Arabia.
2019 **Pacific Graphics**, *Parallel Generation and Visualization of Bacterial Genome Structures*, Korea.
2019 **VCBM**, *Multiscale MolecularVisualization*, Czech Republic.
2017 **IEEE VIS**, *Instant Construction and Visualization of Crowded Biological Environments*, USA.
2016 **NVIDIA GTC**, *Towards Interactive Visual Exploration of Massively Parallel Programs Using a Domain-Specific Language*, USA.

Teaching

2017/18/19 Seminar aus Visualisierung (student supervision)
2017/18/19 Wissenschaftliches Arbeiten (student supervision)
2017/18/19 Visualisierung 1 (exercise)
2017/18/19 Computergraphics (exams)
2018 Real-time Visualization (lecture)

Student Supervision

Bachelor Thesis

2019 Profiling and Optimization of Large Biomolecular Scenes, Felix Kugler

2017 Explorable Semiconductors: Interacting with Crystal Structures in Virtual Reality, Jakob Knapp

2017 Generation and Visualisation of Crystal Structures, Gerald Michl

2017 Modeling Microorganisms, Lukas Mitterhofer

2017 Statistical Modelling of Microorganisms, Theresa Wihann

2017 Visual Evaluation of Computational Models of the Biological Mesoscale, Guillermo Garcia-Escribano

Awards and Scholarships

2019 Pacific Graphics – Best Student Presentation

2019 VCBM – Image Contest Award

2019 Pacific Vis – Storytelling Contest (Winner)

2019 TU Wien Mobility Grant: Funding for PhD students participating in conferences

2017 SciVis – Best Paper Honorable Mention, IEEE Transactions on Visualization and Computer Graphics

2017 Pacific Vis – Storytelling Contest (Finalist)

2017 Best Overall Concept Award, in context of the Bootcamp for Sciencepreneurs. TU Wien, Innovation Incubation Center

2016 Austrian Computer Graphics Award 2016 for Best Technical Solution

2016 Best Poster, Conference on Computational Imaging and Visualization (CIV)

2015 Short-term grant for scientific work abroad. TU Wien, International Office

Publications

2019 Tobias Klein, Ivan Viola, Eduard Gröller, Peter Mindek: *Multi-Scale Procedural Animations of Microtubule Dynamics Based on Measured Data*. IEEE Transactions on Visualization and Computer Graphics (Proceedings of IEEE Vis 2019).

2019 Tobias Klein, Peter Mindek, Ludovic Autin, David Goodsell, Arthur Olson, Eduard Gröller, Ivan Viola: *Parallel Generation and Visualization of Bacterial Genome Structures*. Pacific Conference on Computer Graphics and Applications (PG 2019).

2019 Tobias Klein, Ivan Viola, Peter Mindek: *Towards a Novel Approach for Multi-Scale Procedural Animations of Fibrous Biological Structures*. Poster in Proceedings of EuroVis 2019.

2018 Haichao Miao, Tobias Klein, David Kouřil, Peter Mindek, Karsten Schatz, Eduard Gröller, Barbora Kozlíková, Tobias Isenberg, Ivan Viola: *Multiscale Molecular Visualization*. In Journal of Molecular Biology.

2018 Thomas Bernhard Koch, David Kouřil, Tobias Klein, Peter Mindek, Ivan Viola: *Semantic Screen-Space Occlusion for Multiscale Molecular Visualization*. In Eurographics Workshop on Visual Computing for Biology and Medicine (VCBM 2018).

2017 Tobias Klein, Ludovic Autin, Barbora Kozlikova, David Goodsell, Arthur Olson, Eduard Gröller, Ivan Viola: *Instant Construction and Visualization of Crowded Biological Environments*. IEEE Transactions on Visualization and Computer Graphics IEEE Vis 2017.

2016 Johannes Sorger, Peter Mindek, Tobias Klein, Graham Johnson, Ivan Viola: *Illustrative Transitions in Molecular Visualization via Forward and Inverse Abstraction Transform*. In Eurographics Workshop on Visual Computing for Biology and Medicine (VCBM 2016), pages 21-30.

2016 Tobias Klein, Stefan Bruckner, Eduard Gröller, Markus Hadwiger, Peter Rautek: *Towards Interactive Visual Exploration of Parallel Programs using a Domain-Specific Language*. Poster shown at 4th International Workshop on OpenCL (IWOCL 2016).

Languages

- German: Native
- English: Fluent

Lindengasse 31-33/1/8 – Vienna, Austria 1070
+43 676 467 9510 • tklein@cg.tuwien.ac.at
** 24.01.1986, Siegen, Germany*