Eukaryo: An Agent-based, Interactive Simulation of a Eukaryotic Cell

Douglas Yuen[1] and Christian Jacob[1,2]

[1]Department of Computer Science, University of Calgary
[2]Department of Biochemistry & Molecular Biology, University of Calgary
dwkyuen@ucalgary.ca, cjacob@ucalgary.ca

Abstract

Eukaryo is a 3D, interactive simulation of a eukaryotic cell. In comparison to existing cell simulations, our model illustrates the structures and processes within a biological cell with increased fidelity and a higher degree of real-time interactivity using a virtual reality environment. Implemented in a game engine, *Eukaryo* is a hybrid model that combines agent-based and mathematical modelling.

Through the use of visual scripting, *Eukaryo* incorporates both agent-based modelling and mathematical representations to describe gene expression, energy production and waste removal within the cell in a highly visual, interactive simulation environment. With the help of virtual reality displays, users can be immersed in the crowded spaces of biomolecular worlds and observe metabolic reactions at a high level of detail. Compared to traditional media, such as illustrations and videos, *Eukaryo* offers superior representations of cellular architecture, its components and dynamics of the machineries of life.

Motivation

The concept of scaling humans to a molecular size to facilitate exploration of biological systems at the cellular level forms the premise for the 1966 film "Fantastic Voyage" (Fleischer, 1966; Asimov, 1988). A group of scientists explores the human body by means of a miniaturised submarine. If such journeys at molecular scale were feasible today, they would offer unparalleled experiences to explore the molecular universes of a biological cell. Illustrations of biomolecular worlds have been limited to a few prominent examples only. David Goodsell's "The Machinery of Life" is one such example (Goodsell, 2009). Meticulously constructed from X-ray crystallography, NMR and high-resolution electron micrographs, Goodsell's illustrations capture the densely packed environments inside a cell at the molecular level. Contrary to impressions from typical textbook illustrations, these cellular spaces are highly crowded. Accentuating the complexity of a cell, Goodsell suggests that every structure visible in his illustrations is likely supported and regulated by a myriad of other structures that are not visible.

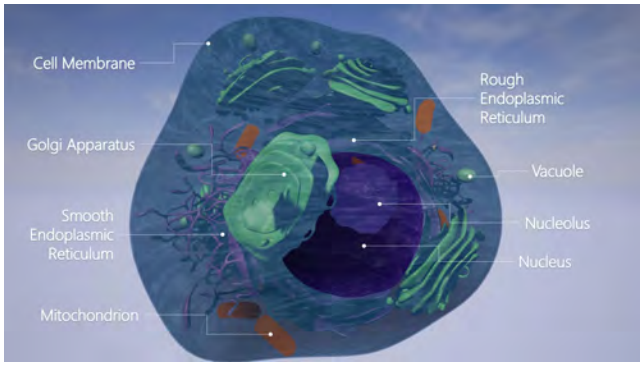


Figure 1: A Generic Eukaryotic Cell. The major organelles of a eukaryotic cell are replicated in the *Eukaryo* model.

Despite capturing the density of materials within a cell, renderings – even at the highly detailed level of Goodsell's illustrations – remain static and cannot depict how each of the different structures do interact with one another. As a result, textbook illustrations tend to be supplemented with videos that can portray the progression of sophisticated cellular biochemical reactions and pathways. The BioVisions video "Inner Life of a Cell" makes use of 3D computer animation to represent some of the key processes that occur in a eukaryotic cell, ranging from gene expression to cellular transport (Harvard BioVisions, 2007). Despite being more expressive, videos limit viewers to observing events from predetermined camera perspectives. Hence, videos do not permit exploration or interaction with the model.

In order to provide an interactive, exploratory environment that – to a certain degree of accuracy – captures the sense of complexity underlying the machinery of life, we have implemented *Eukaryo*, a virtual model of a generic eukaryotic cell (Fig. 1). Built using the game development software Unreal Engine (Epic Games, 2015), our model strives to capture the key structures and functioning units of a cell, similar to Goodsell's illustrations and the BioVisions animations. Furthermore, by utilizing virtual reality (VR) interfaces, *Eukaryo* enables users to interact with the simulation, navigate to different locations within the cell,

investigate its architecture, and explore molecular structures and metabolic pathways.

Using virtual reality visualizations and by combining the interactivity of a video game and artistic renderings of cellular structures with accurate representations of proteins (downloaded from the Protein Database (Westbrook and Fitzgerald, 2003)), *Eukaryo* can provide a sense of dynamics and immersion reminiscent of "The Fantastic Voyage" (Fig. 2). Not only is this a novel and engaging learning approach, this experience also provides a more effective learning method through computational models (Laha et al., 2014).

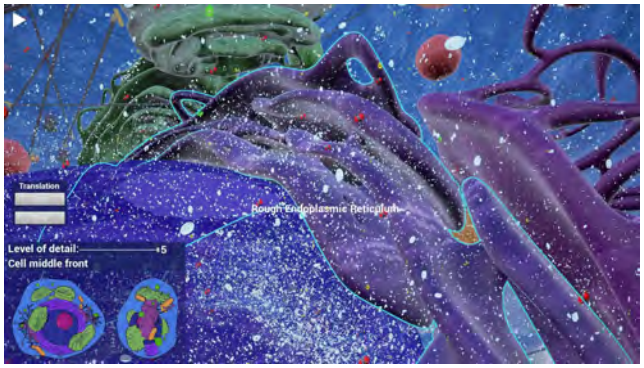


Figure 2: The cellular space with control elements to adjust the level of detail and a minimap for location reference.

Related Work

While biological systems are still mostly studied *in vivo*, many processes often cannot occur in isolation from living systems or are too difficult to explore. Computer models overcome this constraint, allowing for individual processes to be observed in greater detail and providing solutions towards virtual experiments – ideally starting at the cellular level.

Cells Constituting the foundational building block of any biological system, simulations of cells are of particular interest. Projects such as *E-CELL* (Tomita et al., 1999) and *Virtual Cell* (Loew and Schaff, 2001) provide the frameworks for modelling interconnected processes inside a biological cell. *E-CELL* can simulate signaling, cellular reactions and gene regulation, while *Virtual Cell* is intended to act as a more general platform for simulations of micro-biological systems.

Both *E-CELL* and *Virtual Cell* produce numerical outputs only. Actual visualization or illustration of how the system evolves over time is left to other tools and the user's imagination. Moreover, these models are limited in their interactivity: while they are running, users cannot visually observe a process or pause a simulation to inspect its current state.

More recent cell models have been built as sophisticated mathematical models for predictive simulations. For example, Karr et al.'s whole-cell model captures all known processes in the bacterium *M. genitalium* (Karr et al., 2012). While such models are powerful, they are also complex to set up. For instance, the whole-cell model requires twenty-eight separate modules running concurrently to represent one cell and is controlled by $1,900$ empirically-determined values as input parameters. In order to better manage such complex control structures, we utilize hierarchical visual scripting similar to pathway interaction diagrams.

Molecular Dynamics. UnityMol (Lv et al., 2013) and MolecularRift (Norrby, 2015) have been developed using the Unity game engine to visualize molecules (Unity Technologies, 2012). MolecularRift works on the Oculus Rift (Oculus, 2015), whereas UnityMol has some virtual reality support, such as for immersive CAVE visualizations. These applications further demonstrate the opportunities offered by game engines for biomolecular visualization and simulation. These systems also show the benefit of virtual reality for molecular visualization. However, they are focused on molecular dynamics and drug design applications rather than full cell simulations.

Agent-based Modeling. Agent-based methods have been used as simulation techniques complementary to purely mathematical models (Haefner, 2005). For example, a gene regulatory model of the λ-switch has been recreated in a 3D, purely agent-driven simulation (Jacob et al., 2006). A classic gene regulation model studied in *E. coli* bacteria, the lactose operon, has been implemented to illustrate the protein interactions that determine gene expression (Jacob and Burleigh, 2004, 2006). In comparison to other simulators, *LINDSAY Composer* offers a 3D, interactive environment where models can be directly constructed in virtual, 3-dimensional spaces (Jacob et al., 2012). The feasibility of such agent-based models in game engine-inspired environments has been demonstrated in an immune system simulation, purely based on interacting components (Sarpe and Jacob, 2013) and gene expression in *E. coli*, including transcription and translation (Esmaeili et al., 2015) as well as chemotaxis.

Implementating a Eukaryotic cell

Eukaryo is implemented in Unreal Engine (Epic Games, 2015) as a hybrid, interactive 3D model that combines agent-based modeling with mathematical techniques (using differential equations). Agent-based models (ABMs) simulate behaviours by defining a set of interactions between *agents* in a system. We adopt a general approach to define an agent as (1) a set of *situations* an agent may be in, (2) its set of *actions*, (3) all of the possible combinations of its *internal data* and (4) a *decision function* that triggers an action based on the situation and internal data (Afsharch et al., 2006). An

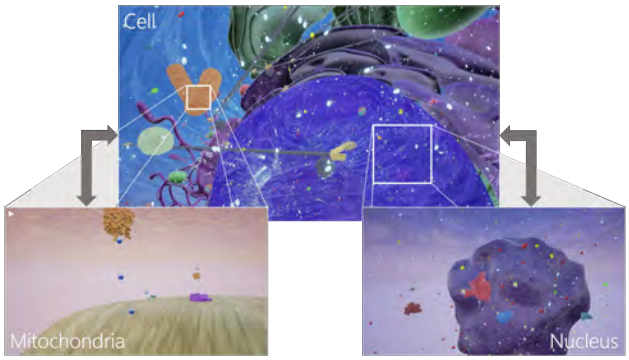


Figure 3: Level setup in *Eukaryo*. The simulation begins in the cytosol (Cell level) from which users may explore organelles such as Mitochondria and the Nucleus in greater detail, which are implemented as separate levels.

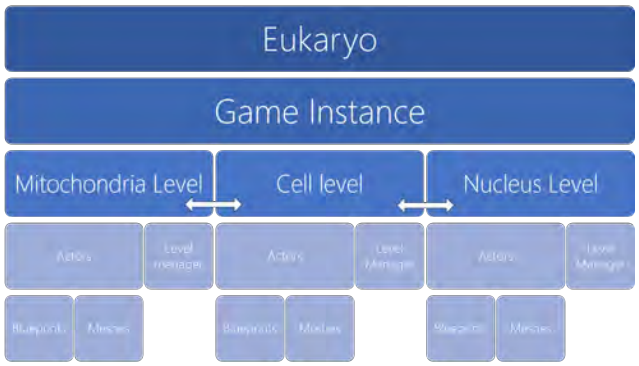


Figure 4: Modular Architecture of *Eukaryo*. The game instance stores the cell's state. Different levels receive information from the game instance to determine their local environment. Each level consists of a collection of Actors. Some Actors have meshes for graphical representation and Blueprints to specify their actions.

ABM constitutes the set of all agents, where the total activity of the agents in an environment forms the ABM's overall behaviour. As such, ABMs are suitable for describing biological systems because observations in biology can be translated into rules for agents within an ABM. Moreover, new information – such as higher levels of detail regarding biomolecular processes – can be incorporated into an ABM simply by adding new agents and/or rules (Jacob et al., 2012). Compared to mathematical models, where equations define system behaviour, ABMs are more difficult to validate and require more computational power to run. However, mathematical models are specific to the processes they formalize and tend to be harder to adapt for describing other systems (Haefner, 2005; Edelstein-Keshet, 1988). A hybrid model confers benefits of both agent-based and mathematical modelling (Esmaeili et al., 2015), complementing one another to produce more powerful, flexible and extensible simulations, as we are about to demonstrate with the *Eukaryo* model.

Cell Universes as Game Levels

In comparison to prokaryotic (i.e., bacterial) cells, eukaryotic cells are more complex, consisting of organelles that carry out specific functions (Figs. 1 and 2). Compartmentalization allows the cell to regulate distinct environments (Jékely, 2007). In *Eukaryo*, the simulation consists of three interconnected environments implemented as (game) levels (Fig. 3). Each level contains a set of Actors, to represent the cell as a whole (Cell) and two organelles (Mitochondria and Nucleus). In Unreal Engine, an Actor[1] is a collection of components that define its location, size and appearance. An Actor is inert, unless their *event graph* is implemented to specify its attributes. The permissible values for these attributes is the set of possible actions available to the Actor and the conditions that result in a particular action.

Actors can be affected by and affect their environment (Fig. 4): information applying to the entire level is stored within persistent data structures known as Game Instances, which enables users to transition between different levels to view cellular processes at different levels of detail (Fig. 3). Blueprint Actors can communicate with the Game Instances in a similar manner as they would with other Actors. Event graphs for each Blueprint Actor (Fig. 6[2]) make use of events, special conditions tracked by the game engine, such as collisions and frame updates that serve as triggers for interactions among agents.

Visual Scripting with Blueprints

Biochemical reactions proceed after collisions that bring molecules and enzymes in close proximity to each other. This forms the basis for all reactions within a cell (Gold, 2014). In a game engine, a biological reaction corresponds to a collision between two Actors that results in a state change in at least one Actor. While this makes it straightforward to describe reactions in terms of collision events, actual biological reactions occur at very high rates determined (and measured) by the densities of, e.g., substrate, enzyme and signal molecules in the cytosol. The concentrations of these reactants are high enough in the cell such that the probability of two reactants colliding and reacting is also high (in humans, each cell may have upwards of 1.7 billion proteins) (Milo, 2013). However, using a large number of Actors to reproduce such a high-density, reactive environment in a game engine is (currently) computationally unfeasible, especially if we want to execute our model on commodity computing devices.

[1]We use the term Actor, instead of agent, to refer to an actor entity within the Unreal Engine game programming system.

[2]Details of this script are explained in a later section.

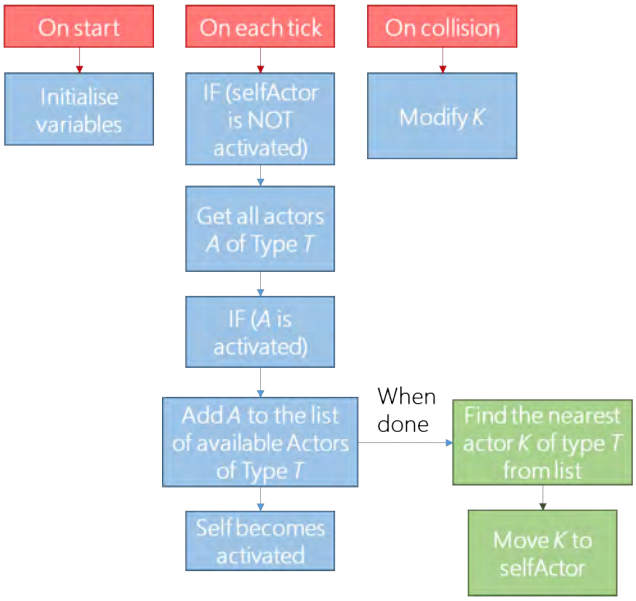


Figure 5: Layout for a Generic Blueprint. Every Blueprint in *Eukaryo* that an Actor uses follows a similar layout. After its variables are initialised, on every step in the simulation, the Actor will look for the nearest Actor A of type T. If ready for a reaction, both Actors will move towards each other. On a collision, they become activated and partake in the reaction.

As such, *Eukaryo* makes use of an alternative mechanism to facilitate an accurate yet illustrative representation of biochemical reactions. In Unreal Engine, each Actor has a *transform component* that keeps track of its location within a level. Other Actors may access this value, allowing for an Actor to compute its distance relative to any other Actor. Using the Blueprint Function Library common to all Actors, one can calculate an Actor's distance to all Actors of a certain type and determine the closest agent (Fig. 5). We use this information in conjunction with the *iTween* animation plugin to smoothly translate Actors from one point to another (Therriault, 2016). Together, this allows for different Actors to be moved rapidly towards other Actors and initiate a reaction without having to wait for molecule agents to collide with one another at random. While this is not a true representation of how chemical reactions proceed *in vivo*, it allows for reactions to be replicated more readily for illustrative purposes. The actual densities of highlighted proteins and other molecules are controlled by mathematical equations and visualized by particle systems as described below.

Case Studies

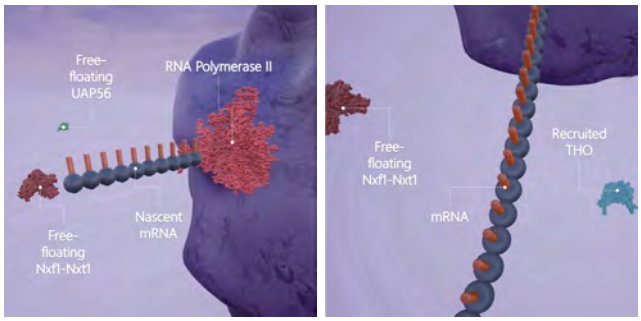
Using a game level approach in combination with visual scripting turns out to be highly flexible in composing pathways, including 3D visualizations, and enabled us to implement a number of key cell metabolism processes within *Eukaryo*. In this paper, we only describe mRNA transport, transcription, translation, glycogenolysis, and the effect of pH on carbonic anhydrase in greater detail. Yet, the underlying mechanisms have been applied to represent other processes within *Eukaryo* at similar levels of detail, such as peroxide decomposition and microtubule assembly.

Figure 6: Event Graph for Glycogen Phosphorylase implemented as a Blueprint visual script. While all Actors use similar blueprints, their behaviours may be further specified: for instance, in glycogen phosphorylase, the enzyme will only continue breaking down glycogen into G1P provided that the glycogen chain has not been consumed. After a certain period of inactivity, the enzyme will be broken down.

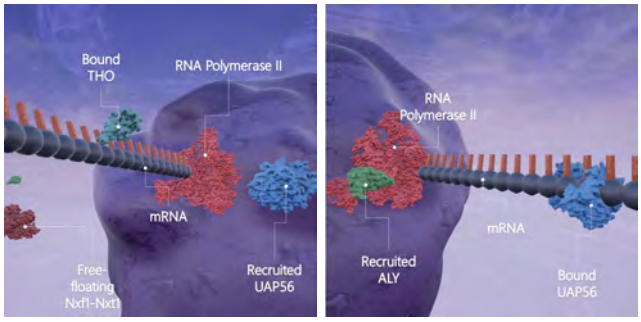
Case Study 1: Transcription and Translation

Within the nucleus game level (Fig. 3), mRNA is represented by a single Actor with a preset number of static mesh components. An mRNA Actor spawned into the level makes its segments visible, one at a time, to mimic the synthesis of individual mRNA segments during transcription (Fig. 7a). Once all segments are visible, the entire strand detaches from RNA polymerase. Beginning with the THO complex (Fig. 7b), transport signals can now identify and bind to mRNA (Carmody and Wente, 2009). The THO Actor continuously looks through the level for any nascent mRNA to bind to. When a binding partner is found, the actor is moved to the mRNA, where, on collision, THO is attached to it. Now with bound THO, mRNA is ready for a UAP56 signal protein to bind (Fig. 7c). Once UAP56 is bound, ALY binds in a similar fashion (Fig. 7d). When Nxf1-Nxt1 binds, the other Actors are detached from the mRNA, which is then ready to be transported out of the nucleus and undergo additional processing for translation (Fig. 7e). Once an mRNA actor is transported into the cytosol and tagged by Gle1 and Dbp5, it is ready for translation (Fig. 7f).

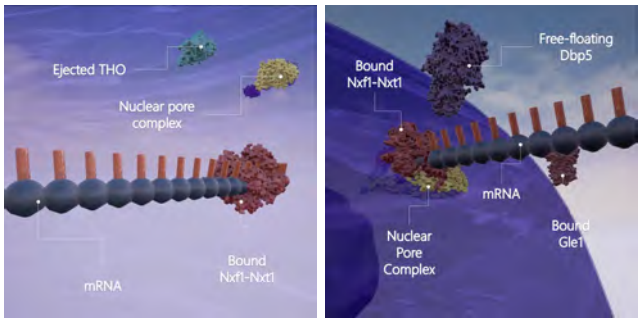
Ribosomes carry out translation in the cell level. After a ribosome Actor finds an mRNA strand, it follows a spline along the mRNA and synthesizes nucleic acid chains that "fold" into a protein once the end of the spline is reached. The mRNA Actors are animated to improve their visual im-



(a) mRNA transcription **(b)** THO binds

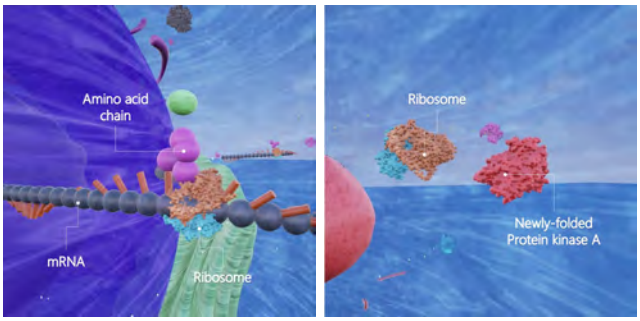


(c) UAP56 recruited **(d)** ALY recruited

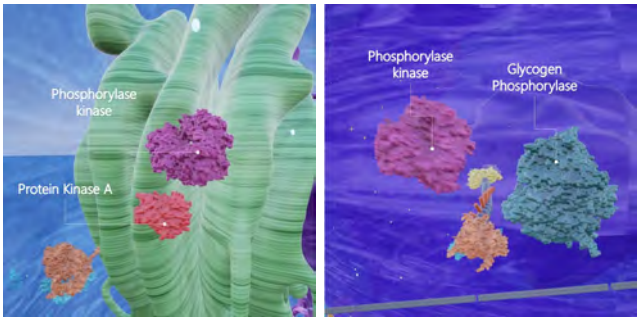


(e) Nxf1-Nxt1 signals export **(f)** Gle1, Dbp5 binds

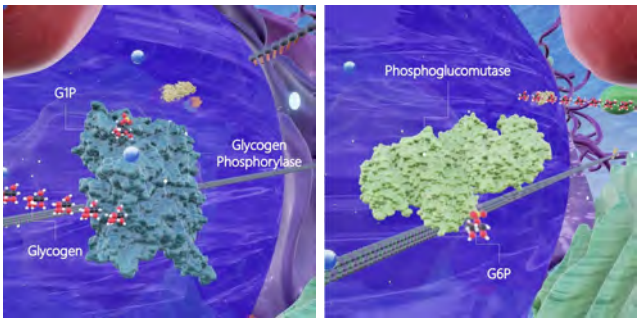
Figure 7: A step-by-step illustration of mRNA export.



(a) mRNA translation **(b)** Amino acids fold into PKA



(c) PKA activates PK **(d)** PK activates GP



(e) GP breaks glycogen to G1P **(f)** Conversion of G1P to G6P

Figure 8: Translation and Glycogenolysis.

pact within the simulation; its children mesh components are positioned along an animated spline. This spline also forms the path for the ribosome Actor to move along during translation (Fig. 8a). Following translation, exosome complex Actors "degrade" the mRNA so that mRNA does not accumulate within the cell (Fig. 9). Exosome complexes are responsible for mRNA degradation, removing inactive mRNA as a part of mRNA turnover (Makino et al., 2013).

Each mRNA Actor possesses an attribute that specifies which protein the ribosome produces during translation. In the current version of *Eukaryo*, mRNA transported into the cytosol ecodes for adenylase cyclase, protein kinase A (PKA) and phosphorylase kinase, and signal proteins for glycogenolysis. Adenlyase cyclase converts adenosine monosphosphate (AMP) into its cyclyic form, cAMP, activating PKA, which, in turn, activates phosphorylase kinase (Alberts et al., 2015). mRNA also codes for the glycogen phosphorylase enzyme, which is activated by phosphorylase kinase and carries out glycogen breakdown, as described in the next section.

Case Study 2: Glycogenolysis

As a second case study, we present the replication of the glycogenolysis pathway. After translation (Fig. 8a) and protein folding (Fig. 8b), the signal proteins adenylase cyclase, protein kinase A, phosphorylase kinase and glycogen phosphorylase participate in the regulatory pathway that leads to the release of glucose-1-phosphate (G1P). We have implemented this pathway and describe it here in detail (Venkataraman and Luck, 1949).

Protein kinase A is activated by cyclic AMP (cAMP, a second messenger synthesised by adenylase cyclase) (Lodish et al., 2012). Once activated, protein kinase A phosphorylates phosphorylase kinase to stabilise it (Fig. 8c). In

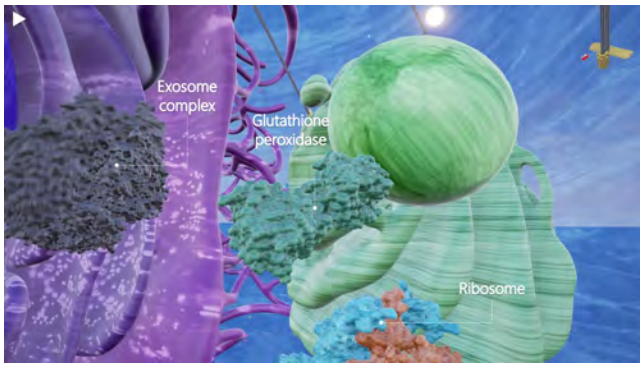


Figure 9: Glutathione Peroxidase within the Cell Level. An exosome complex can be seen (in grey) left of glutathione peroxidase.

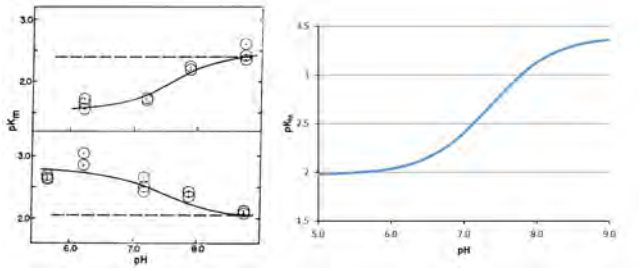
turn, phosphorylase kinase activates glycogen phosphorylase (Fig. 8d), which can begin cleaving glucose units from glycogen: G1P is formed (Fig. 8e). Phosphoglucomutase converts G1P into glucose-6-phosphate (G6P), a reactant in glycolysis (Fig. 8f). The visual scripts to control these reactions are depicted in Figure 6.

Case Study 3: pH Effect on Carbonic Anhydrase

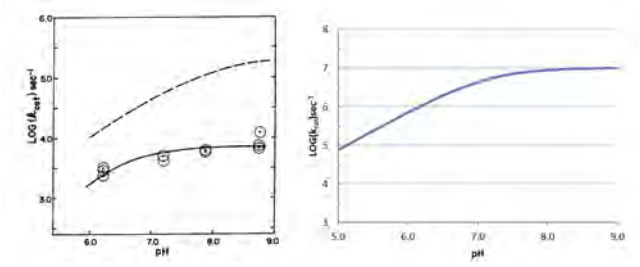
It has been found that varying the pH level in a eukaryotic cell's environment alters its enzymatic activity. This was also replicated in a mathematical model by Khalifah and Edsall (1972). We followed their reaction schema and implemented a C++ Blueprint Function that calculates the resulting Michaelis-Menten (pK_m) and catalytic (k_{cat}) constants based on the pH value (Fig. 10). The Michaelis-Menten constant pK_m is the substrate concentration that results in a reaction rate half of the maximum rate; inversely, pK_m measures a substrate's affinity for an enzyme. A smaller pK_m value corresponds to a reaction that approaches the maximum rate of reaction more quickly. The catalytic constant, k_{cat}, describes how fast an enzyme may react with substrates to form the product. Specific values for pK_m and k_{cat} can be calculated using rate constants as described in (DeVoe and Kistiakowsky, 1961).

The Blueprint Function is attached to a level manager actor that takes the environment pH to calculate pK_m and k_{cat} (Fig. 4). The values are sent to all of the carbonic anhydrase Actors in the scene, thus controlling their rate of reaction. Each carbonic anhydrase Actor has an attached particle emitter which releases bicarbonate ions at a rate corresponding to the calculated reaction rate.

As illustrated in Figure 10, we found that our model qualitatively aligns with the values reported in (Khalifah and Edsall, 1972). The pK_m and k_{cat} values in our model suggest higher affinities and reaction turnovers. Further adjustments to the experimental parameters will need to be made to provide more accurate representations.



(a) pK_m values in response to varying pH



(b) k_{cat} values in response to varying pH

Figure 10: pH affecting carbonic anhydrase. Left: Khalifah and Edsall model; right: replicated *Eukaryo* model. This deterministic model controls the visual effects and average carbonic anhydrase agent concentrations (i.e., activity) in the simulated cell.

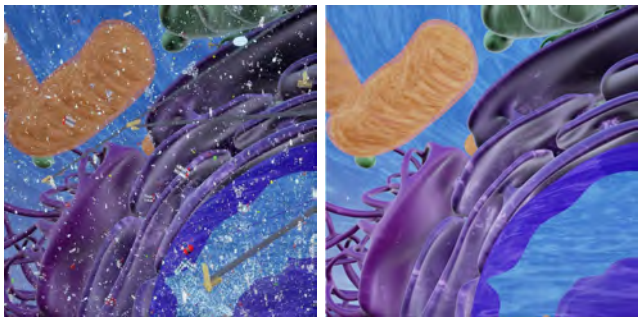
The inclusion of a Michaelis-Menten model in *Eukaryo* illustrates how mathematical models can be incorporated into agent-driven models: functions that keep track of reaction rates and concentration changes are used to manipulate attributes of individual agents, rather than directly influencing their decision functions. While our model is relatively simple, for now, this approach can be applied to more detailed models. For example, C++ libraries can be imported to construct functions that solve differential equations and generate biologically relevant outputs for predictive modelling, similar to those seen in programs such as Virtual Cell (Loew and Schaff, 2001) and other studies (Esmaeili et al., 2015; Sarpe and Jacob, 2013).

Virtual Cell Spaces

In a gaming environment with immersive visualization and real-time interaction, one has to think about how to visualize the "cellular universes", how to navigate through them and create convincing, yet scientifically justified effects to highlight the implemented biomolecular processes.

Visualisations

Similar to illustrations found in textbooks, organelles in *Eukaryo* have been given specific colours to ensure that they are distinguishable from one another, while simultaneously providing some indication of their function (Fig. 1). The nucleus and endoplasmic reticula are purple for proximity



(a) High detail **(b)** Low detail

Figure 11: Side-by-side comparison of the simulation at different levels of detail. (a) All of the elements are visible to capture the crowded cell environment. (b) All game objects and particle systems disabled, allowing users to focus on the cell's structures.

to the cell's core and involvement in biomolecule synthesis. The Golgi apparatus and vacuoles are green: both are involved in storage and transport. Mitochondria are orange, standing out from the other organelles. Together, they provide the context for the processes described in this paper.

Particle Systems. To convey that cellular spaces are highly packed, multiple particle systems are incorporated into *Eukaryo*. Typically used for representing transient entities like smoke and flames, particle systems can be configured to emit billboards, which are entities that always face the camera (Reeves, 1983). These billboards are used in *Eukaryo* to represent water molecules, sodium, potassium and chlorine ions, as well as albumin, fatty acids and carbon dioxide (Fig. 11). By utilizing the graphics processing unit (GPU) for rendering, more than a million particles can be displayed on screen without compromising performance. This means that the simulation remains responsive to user input at any point. Each particle has a random velocity, angular motion and lifespan to convey a sense of stochasticity to mirror Brownian motion.

Protein Data. In order to accurately represent each protein in *Eukaryo* we have imported their 3D structure from the Protein Databank (PDB) repository (Westbrook and Fitzgerald, 2003). A PDB file contains the coordinates, rotation and amino acid sequence in a protein. The resulting structural information can be converted into surface meshes and imported into 3D modeling software (such as Autodesk(Autodesk)). Before import into the game engine, we apply textures to the mesh to improve a structure's visibility in the cell. Hence, we can present protein structures consistent with accurate, empirical data.

Navigation and Level of Detail Controls

In comparison with other virtual environments that depict entities such as buildings and terrain, traditional notions of up and down are not apparent within a cell as familiar reference points are absent. This makes it more difficult for users entering the virtual cell spaces to orient themselves using visual cues present in the cell alone. Therefore, additional navigational aids were implemented. A minimap is displayed in the lower left-hand corner, where a cursor indicates the current camera position (Fig. 2). A slider allows the user to hide or show varying levels of detail (Fig. 11). At the maximum level, all proteins and molecules are visible. A lower level of detail reduces the number of visible elements. Actors that are invisible still continue running in the background, so the simulation itself is never interrupted. Lastly, users can highlight and bring up information about any entity they are looking at or approaching within the cell.

Conclusion

We have described an implementation of *Eukaryo*, an interactive, hybrid 3D model of a eukaryotic cell created in the Unreal Engine game environment. Using an agent-based approach to describe biological processes as a series of shared interactions between Actors, our model becomes highly versatile. We have highlighted some of the pathways and processes implemented in *Eukaryo*: gene expression with transcription, translation, and mRNA transport; glycogenolysis and pH effects on carbonic anhydrase activity. Built on a game engine architecture, the *Eukaryo* system incorporates mathematical models as additional components, thus facilitating and further extending the expressiveness and accuracy of the simulations. By combining state-of-the-art biological modelling with 3D visualization and real-time interactivity, *Eukaryo* provides an immersive cell model that can serve as a learning tool as well as an environment to perform virtual experiments.

The *Eukaryo* software, videos and virtual experiments are available on the Lindsay Virtual Human website.

References

Afsharch, M. et al. (2006). Ontology-Guided Learning to Improve Communication. In *AAMAS 2006*, pages 923–930, Hakodate, Hokkaido, Japan.

Alberts, B. et al. (2015). *Molecular Biology of the Cell*. Garland Science, Taylor & Francis Group, LLC, sixth edition.

Asimov, I. (1988). *Fantastic Voyage*. Bantam.

Autodesk. Maya 3D Animation Software.

Carmody, S. R. and Wente, S. R. (2009). mRNA nuclear export at a glance. *Journal of Cell Science*, 122(12):1933–7.

DeVoe, H. and Kistiakowsky, G. B. (1961). The enzymic kinetics of carbonic anhydrase from bovine and human

erythrocytes. *Journal of the American Chemical Society*, 83(2):274–280.

Edelstein-Keshet, L. (1988). *Mathematical models in biology*. McGraw-Hill.

Epic Games (2015). Unreal Engine 4 Documentation.

Esmaeili, A. et al. (2015). PROKARYO: an illustrative and interactive computational model of the lactose operon in the bacterium Escherichia coli. *BMC Bioinformatics*, 16(1):311.

Fleischer, R. (1966). *Fantastic Voyage*. Twentieth Century Fox Film Corporation.

Gold, V. (2014). International Union of Pure and Applied Chemistry Compendium of Chemical Terminology. *Iupac*, page 1670.

Goodsell, D. S. (2009). *The Machinery of Life.* Springer Science, New York, USA, second edition.

Haefner, J. W. (2005). *Modeling Biological Systems:: Principles and Applications*. Springer Science & Business Media.

Harvard BioVisions (2007). The Inner Life of the Cell (Video).

Jacob, C. and Burleigh, I. (2004). Biomolecular swarms: an agent-based model of the lactose operon. *Natural Computing*, 3(4):361–376.

Jacob, C. and Burleigh, I. (2006). Genetic programming inside a cell. *Genetic Programming Theory and Practice III*, 9:191–206.

Jacob, C. et al. (2006). Swarms and genes: Exploring λ-switch gene regulation through swarm intelligence. In *IEEE Congress on Evolutionary Computation*.

Jacob, C. et al. (2012). LINDSAY Virtual Human: Multiscale, Agent-based, and Interactive. In Kolodziej, J., Ullah Khan, S., and Burczynski, T., editors, *Advances in Intelligent Modelling and Simulation*, pages 327–349. Springer, Berlin.

Jékely, G. (2007). *Eukaryotic Membranes and Cytoskeleton*, volume 607. Springer, New York, 1st edition.

Karr, J. R. et al. (2012). A whole-cell computational model predicts phenotype from genotype. *Cell*, 150(2):389–401.

Khalifah, R. G. and Edsall, J. T. (1972). Carbon dioxide hydration activity of carbonic anhydrase: kinetics of alkylated anhydrases B and C from humans (metalloenzymes-isoenzymes-active sites-mechanism). *PNAS*, 69(1):172–6.

Laha, B. et al. (2014). Effects of VR system fidelity on analyzing isosurface visualization of volume datasets. *IEEE Transactions on Visualization and Computer Graphics*, 20:513–522.

Lodish, H. et al. (2012). *Molecular Cell Biology*. W.H. Freeman and Company, seventh edition.

Loew, L. M. and Schaff, J. C. (2001). The Virtual Cell: a software environment for computational cell biology. *Trends in Biotechnology*, 19(10):401–6.

Lv, Z. et al. (2013). Game On, Science - How video game technology may help biologists tackle visualization challenges. *PLoS ONE*, 8(3):e57990.

Makino, D. L., Halbach, F., and Conti, E. (2013). The RNA exosome and proteasome: common principles of degradation control. *Nature Reviews. Molecular Cell Biology*, 14(10):654–60.

Milo, R. (2013). What is the total number of protein molecules per cell volume? A call to rethink some published values. *BioEssays*, 35(12):1050–1055.

Norrby, M. (2015). MolecularRift, a Gesture Based Interaction Tool for Controlling Molecules in 3-D. Master's thesis, Department of Design Sciences, Lund University, Sweden.

Oculus (2015). Oculus Rift VR Headset.

Reeves, W. T. (1983). Particle systems—a technique for modeling a class of fuzzy objects. *ACM SIGGRAPH Computer Graphics*, 17(3):359–375.

Sarpe, V. and Jacob, C. (2013). Simulating the decentralized processes of the human immune system in a virtual anatomy model. *BMC bioinformatics*, 14(Suppl 6):1–26.

Therriault, J. (2016). iTween for Unreal Engine.

Tomita, M. et al. (1999). E-cell: software environment for whole-cell simulation. *Bioinformatics*, 15(1):72–84.

Unity Technologies (2012). Unity Manual.

Venkataraman, J. and Luck, J. M. (1949). Phosphoglucomutase: II. Mechanism of Action. *Journal of Biological Chemistry*, 179(2):569–75.

Westbrook, J. D. and Fitzgerald, P. M. D. (2003). The PDB format, mmCIF formats, and other data formats. In Bourne, P. E. and Weissig, H., editors, *Structural Bioinformatics*, chapter 8, pages 161–179. Wiley-Liss Inc., 2nd edition.