Meta!Blast computer game: a pipeline from science to 3D art to education

William Schneller  
Iowa State University

P.J. Campbell  
Iowa State University

Diane Bassham  
Iowa State University, bassham@iastate.edu

Eve S. Wurtele  
Iowa State University, mash@iastate.edu

Follow this and additional works at: http://lib.dr.iastate.edu/gdcb_las_conf

Part of the Bioinformatics Commons, Cell and Developmental Biology Commons, and the Computational Biology Commons

Recommended Citation
Schneller, William; Campbell, P.J.; Bassham, Diane; and Wurtele, Eve S., "Meta!Blast computer game: a pipeline from science to 3D art to education" (2012). Genetics, Development and Cell Biology Presentations, Posters and Proceedings. 4.  
http://lib.dr.iastate.edu/gdcb_las_conf/4

This Conference Proceeding is brought to you for free and open access by the Genetics, Development and Cell Biology at Iowa State University Digital Repository. It has been accepted for inclusion in Genetics, Development and Cell Biology Presentations, Posters and Proceedings by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
Meta!Blast computer game: a pipeline from science to 3D art to education

William Schnellera, P.J. Campbella, Diane Basshamb, Eve Syrkin Wurteleb*

aVirtual Reality Applications Center, Iowa State Univ., 1620 Howe Hall, Ames, IA, USA 50011; bDept. of Genetics, Development, and Cell Biology, Iowa State Univ., 1210 Molecular Biology Building, Ames, IA, USA 50011

ABSTRACT

Meta!Blast (http://www.metablast.org) is designed to address the challenges students often encounter in understanding cell and metabolic biology. Developed by faculty and students in biology, biochemistry, computer science, game design, pedagogy, art and story, Meta!Blast is being created using Maya (http://usa.autodesk.com/maya/) and the Unity 3D (http://unity3d.com/) game engine, for Macs and PCs in classrooms; it has also been exhibited in an immersive environment. Here, we describe the pipeline from protein structural data and holographic information to art to the three-dimensional (3D) environment to the game engine, by which we provide a publicly-available interactive 3D cellular world that mimics a photosynthetic plant cell.

Keywords: Cellular biology, metabolic biology, serious games, education, 3D art, computer graphics, path-planning

1. INTRODUCTION

A cell is a complex unit, made up of many types of structures, each of which functions differently and interacts with the others in particular ways. Traditional methods of teaching cell biology, via textbooks and illustrated diagrams, do not always adequately show how the cell functions or how its parts interact in real time and space. In a 3D computer game, players are immersed into a virtual environment that requires them to recognize, not only where in the virtual space they are located, but also how the environment reacts to their actions. By presenting the cell as an interactive environment that responds to the players actions in real-time, students will not only take more active interest in what they are being taught, but also will gain a better sense of how the cell actually operates by being able to see it function as it would in real life.

Meta!Blast is intended to provide a gameplay experience on par with those of professionally developed commercial computer games. Among the developers working on Meta!Blast are artists, programmers, sound programmers, and writers, all dedicated to high quality game design. Their goals are creating aesthetically pleasing environments that will catch students’ attention and motivate them to explore; a storyline that is practical, believable, and will hold the attention and interest of high school and undergraduate students; and gameplay mechanics that are not only educational, but also simply fun. In addition, by using a computer game as a vehicle to teach, students who typically do not care for science may be taught basic biological concepts through play. Meta!Blast will be able to reach students who do not learn well from text or diagrams, who learn best from interactive, first-hand experience.

By playing Meta!Blast, students will be introduced to the concepts of respiration, photosynthesis, and the functions of various organelles in a manner complementary to the textbook. Students will also be introduced to how the living cell actually appears through a biologically accurate virtual replica of a cell environment. Students will be presented with a series of puzzles and goals, such as having to create oxygen, which will require them to think critically in order to come to the solutions. By actively participating in these biological processes, students will “learn by discovery.”

2. RESEARCH

The world of the cell is unique. Strange structures, odd environments, and fascinating processes make it an interesting place for students to explore. However, creating an accurate representation of the cell has been an incredible challenge. The Meta!Blast team has had the privilege of working with some talented young developers, all contributing their skills and imagination to create this scientific adventure. Through the use of primary research literature, various microscopic imaging techniques, and other scientific data, our developers have worked with leading biologists to make Meta!Blast as authentic as technology will allow.

To ensure our information is accurate, valid, and current, we use a wide variety of original research papers in the areas of cell biology, chemistry, and metabolic biology to shape the details of the cell environment and the game itself. The primary foundation of the artists’ research arsenal includes multiple textbooks. The world of Meta!Blast is set inside of a soybean leaf spongy mesophyll cell, so our resources and references are always filtered by relevance and similarity to the soybean species.

2.1 Micrographs

To gain an accurate, unaltered vision of the cell’s structures we refer to various forms of microscopy. Transmission electron micrographs (TEMs), scanning electron micrographs (SEMs) (Figure 1) and confocal micrographs are the types that we use most often for structural information. In addition, we refer to content generated by means of electron tomography to model the overall 3D structures of organelles. This data is located through connections with Iowa State researchers, and from the primary literature.

![Figure 1. Scanning electron micrograph of a soybean mesophyll cell with chloroplasts by Hilal Ilarslan.](image)

2.2 PDB

The RCSB Protein Data Bank (PDB) (www.rcsb.org) is a global database for molecular structures. It is the source of structural data of proteins and molecules for the Meta!Blast team, and has provided great value in helping us optimize the accuracy of representations of cellular components. PDB files contain specific information such as the 3D coordinates of atoms, b-factor data, and domain information. The Meta!Blast artists download PDB files after the biologists have approved the structure, and load them into Maya or Cinema 4D (C4D) (http://www.maxon.net/) for visualization (Figure 2). Two plugins that we currently use are Molecular Maya (mMaya) (http://www.molecularmovies.com/) and the embedded python molecular viewer (ePMV) (http://epmv.scripps.edu/). Both tools offer different methods of viewing the molecular structure, similar to PyMol (http://www.pymol.org/) or Chimera (http://www.cgl.ucsf.edu/chimera/), but instead using the digital content creation application familiar to most 3D artists.
2.3 Artistic Interpretations and Scientific Visualizations

Since the time of Robert Hooke in the 1600s, visualizations of the cellular world have existed. For Meta!Blast, we create an original take on this fascinating science. Therefore, we need to know what has previously been done. Two-dimensional (2D) images from artists such as Frank Armitage and animations from studios such as Harvard BioVisions and Hybrid Medical Animation have served as inspiration, and are a good reminder of established styles of scientific visualization. Furthermore, researchers are constantly demonstrating new research with 3D renderings, such as new findings on the architecture of thylakoid membranes.

3. CONCEPT ART

Before anything is created in 3D, the artists fully explore all designs in the 2D realm. “Concept art” is a fundamental part of the pre-production phase in game development and the earliest artwork that is created. It is by this work that the entire tone and visual style of the project is defined. Although the 3D cell is supposed to remain close to the source material the artists still needed to apply their imagination to produce creative and inspiring visuals.

3.1 Initial Sketches

All concept art begins with small quick sketches called thumbnails. The primary purpose is to quickly explore multiple ideas before moving on to larger, more refined drawings. Usually thumbnails are simply line drawing, but sometimes will use value and color in an attempt to organize space (Figure 3). The latter is especially helpful when planning a production painting, which could take hours or days to complete. A variation on the thumbnail sketch is a silhouette. This is a simple black and white image that is meant to explore the overall shape of an object. This shape is what will help players immediately recognize the asset in the game and should convey key characteristics, otherwise referred to as being “readable”.

Figure 3 and Figure 4. Meta!Blast concept art. Left: “The Cell” by Trevor Brown. Right: “The Cytosol” by Amy Dixon.
3.2 Production Paintings

Envisioning a dynamic, ever-changing environment is no easy task. Production paintings are important for defining the atmosphere and lighting of a scene, and demonstrating what is possible (Figure 4). We wanted to make something that depicts the cell as a lively place, rather than being a static diagram or simplified cartoon. Due to the minimal amount of pigment inside most structures of the mesophyll cell our artists had to be a bit more creative, yet still remain conservative with the color palette. We chose something less saturated to avoid looking cartoonish or too stylized. Another challenge was determining the effects of light at such a small scale. Doing so allowed us to develop our own unique view of the microscopic world.

3.3 Style Sheets

The organelles were designed based on actual micrograph images, and data regarding protein composition (Figure 5). Since only the vacuole and chloroplast contain pigment, the artists relied on the idea of reflected color for the rest of the organelles. “Style sheets” are primarily used for modelers and texture artists. By laying out concept art in this fashion, other artists on the team are able to see the primary characteristics of each structure. A typical style sheet will also include orthographic views (top, side, three-quarter) of the assets, and may be brought directly into the modeling application to be used as a guide.

4. 3D ART

Once the pre-production phase is complete, the project enters full production. The production phase is when the models are built and textured, materials are created, and animation is refined, thereby bringing the project to life. Artistic visions that once existed only as drawings and paintings are transformed into full 3D scenes. The production for a real-time application is similar to that of film, but the artists need to be more mindful about creating art assets that are both good-looking and efficient for in-game use. There are simply more restrictions, thus giving the artists a larger challenge.

4.1 Modeling

Since the advent of 3D gaming in the mid-1990s, people have come to expect higher fidelity graphics with each passing year. To keep up with the demand, the Meta!Blast artists continue to utilize industry standard software. We primarily use Maya, which is also used in the film industry for high quality visual effects, to create our 3D content. However, due to recent advances in interchange data formats such as FBX (http://usa.autodesk.com/), we can also use programs such as Cinema 4D. The assets shown in Figures 6 and 7 were modeled and rendered with C4D.

Using the reference images gathered by the biologists and the concept art created by the Art Director and Concept Artist, modelers use polygonal meshes to establish the forms. It is generally best to work from general to specific, taking great care to establish the proper proportions. A common method of modeling, called box modeling, involves using a
geometric primitive such as a cube for the base shape, then gradually adding subdivisions as more detail is necessary. In some instances, it is actually easier to model the object polygon by polygon, however this is usually only in extreme cases where edge flow is critical. The goal of this process is to come away with a low polygon, organized mesh that is fit for animation. The artist will attempt to maintain a model that is all quad faces in order to provide a well divided surface and to make editing the mesh easier. Once the model is rendered, the faces will then become triangulated.

For highly detailed organic forms such as the surface of cellular organelles or proteins, a specialized application is used. Digital sculpting is a process of taking a standard polygonal mesh and using digital brushes to establish shape, form, and fine details. It is desirable for the artist to use a graphics tablet, such as the Wacom Intuos (http://www.wacom.com/) in order to achieve an analog feel. This is in sharp contrast with the aforementioned modeling programs that rely on pushing and pulling faces and vertices with a mouse. Applications such as ZBrush (http://www.pixologic.com/) or Mudbox (http://usa.autodesk.com/) are widely used to provide this high level functionality. While standard modeling programs can handle meshes up to a few million faces, digital sculptures commonly have tens of millions of faces, all while allowing the artist to smoothly interact with the brushes. All of this detail cannot be placed directly into the game though. We use a combination of the traditional 3D modeling techniques, and advanced texture maps to simulate the high frequency details.

Figure 6 and Figure 7. Screen shots from within the Meta!Blast cell. Left: Chloroplast thylakoid network. Right: Approaching the vacuole. Rendered with C4D.

4.2 UV Layout

After a model is finished, the U and V surface coordinates need to be “flattened” to prepare the model for its surface definitions, or texture maps. UV mapping is similar to pelting an animal hide; the model is analyzed and projections are made from various axes. The goal of the UV layout is to provide a flat, uniform, distortion free canvas on top of which an artist can paint in an application such as Photoshop (http://www.adobe.com/). For efficiency in the game engine, the UV map must be laid out in the first quadrant of a Cartesian coordinate system. The different projections, or UV shells, of the UV map are scaled in size proportionate to how important they are on the model. For example, if we “unwrap” a human body, the face will be given the largest space, or pixel ratio, on the UV layout to allow for more detail. Figure 8 shows the model of the Meta!Blast nanobot in Maya with the corresponding UV texture editor to the left. The nanobot body, which has the largest volume on the model, is given the most space on the canvas.
4.3 Texture Mapping

Texture maps are special images that help define the different attributes that make up the model’s surface quality. These can range from diffuse color (Figure 9), to specular information, to height information (Figure 10). Texture maps allow the artist maximum control over the look and feel of the asset. The process of texturing traditionally begins with importing the UV layout from the 3D application into a 2D image editor. In recent years digital sculpture applications are now supporting 3D painting, and several dedicated programs have been developed, such as Mari (http://www.thefoundry.co.uk/) and Body Paint 3D (http://www.maxon.net/). This does make some painting tasks easier for the artist, but when painting things like straight lines, a 2D canvas is still preferred.

As a rule of thumb, today’s game engines seem to prefer texture maps with square pixel ratios in multiples of 2, 4, 8, 16, and up. The overall texture resolution is determined by how important an asset is for gameplay, how large the asset is in game units, and if any tiling can be used. For example on object that is fairly prominent might need a 1024x1024 pixel map, while a terrain model can make use of a 64x64 pixel map that repeats any number of times in the U and V directions. Ultimately the repeating texture will be more efficient, but the player could start to notice the tiling. Like all aspects of game development, we use a combination of the techniques depending on the situation.

4.4 Materials and Shaders

A “material” is nothing more than a container node that holds attributes that define a surface. A “shader” is a program that does all of the calculations to determine what attributes a material can possess, as well as direct what shading model to use when rendering. Many different shading models exist thanks to the work of early computer graphics scientists such as James Blinn. Artists rarely author (program) the shaders, but they do provide the material set up to which the
shaders are applied. This involves finding a material type that most closely represents the original concept art, plugging in the appropriate texture map to the correct attribute, and endlessly fiddling with sliders and numbers until the material looks good. Often we save this step until the lighting is decided upon as lighting can have a drastic impact on the look of both the material and textures.

Shaders can be very simple for basic effects, but using these shaders we can utilize advanced features that may be otherwise difficult to implement. For example, in the Meta!Blast cell scene we use a special vertex displacement shader (Figure 11). Doing so allows us to achieve a sense of motion and fluidity at almost no computational cost as the movement is done on the GPU. However, we have to be mindful to not use shaders for effects integral to gameplay, as lower end hardware may not be sufficient enough to handle these shader instructions. At this time, our vertex displacement shaders require OpenGL 3.0 or above.

4.5 Scene Assembly

Once all of the individual assets have been created, artists combine these assets into something the user can experience. The level, or scene, must be interesting and detailed, but fall within the overall polygon count budget to achieve fast performance for the user. This step involves a great deal of thinking about how to intelligently reuse assets, and figuring out how a complex model can be broken down into repeating, seamless modules. For example, in the cell scene shown in Figure 12, artists used a combination of these techniques. The vacuole and nucleus are unique models, while the endoplasmic reticulum is a network of intelligently designed modules that can be rotated 90° along any axis and still fit together. It does not matter what methods are used, as long as the end result looks aesthetically pleasing and fits the technical specifications.

Figure 11. Meta!Blast vertex displacement shader with parameters in Unity 3D.

Figure 12. Screen shot from within the Meta!Blast cell: Cytosol level with the HUD on view, using Unity 3D.
4.6 Lighting

Much like a movie or a photo shoot, each scene has to be lit. Considerations at this step include providing enough illumination for the player to understand what is happening, creating a certain mood or atmosphere as defined in the production paintings, and always remaining efficient for in-game performance. In Meta!Blast we use pixel shaders for the higher quality graphics settings. However, because each pixel on the screen is affected by each light, having four lights may be fine for performance, but having over one hundred individual lights would prove disastrous in terms of speed of rendering. We achieve a high quality aesthetic while remaining efficient by the process of light mapping, or “baking” all lighting information into a set of texture maps. The advantage to this is that we can simulate the effect of bounced and reflected light, thus making our game look much more realistic. However, there is a trade-off. Generally, only static geometry, i.e., game objects that do not move, have baked light maps due to the corresponding baked shadow data. It would look very strange and confusing to see shadows stuck to a moving object. There are emerging technologies that allow for real-time dynamic global illumination, but as of now they are not used in Meta!Blast due to high cost and low performance on hardware available to our target demographic.

4.7 Animation

An asset in a computer game that needs to act or move needs to be animated. For the development of Meta!Blast we use a combination of “bones” and “blend shapes”. The most common way to animate objects in a game is through a bone system, in which a hierarchy of joints in created. This skeleton is then bound to the mesh’s vertices through a process called skinning, and each vertex is weighted. When a mesh is weighted, each joint in the skeleton has influence over a specific set of vertices. A bone system generally gives optimal performance, based on the number of bones used and how many vertices each bone may affect. We have considered using bones to drive motion of the endoplasmic reticulum model, and any characters that may be in the game. Other times, bones are not the best option. For example, in Meta!Blast we have many surfaces such as those of the organelles that exhibit fluid characteristics. These could be rigged with bones, but the resulting animation may be rough, or the level of fidelity may not be as high as the artist would like. Thus, for amorphous shapes, alternate rigging methods, such as blend shapes are used.

Blend shapes, also known as morph targets, are multiple variations of a model that have their vertex information recorded and baked into a neutral pose. This type of animation is generally used for facial animation, or anything needing extremely smooth motion. The only major downside is that blend shapes can be very expensive in-game if they are applied to a higher resolution polygon mesh. We have considered using this approach for animations of the organelle membranes. When the animation is played, the system compares the position of the models vertices and calculates the movement.

Whichever method is used, the animation still must work in an interactive world where the player could potentially play any animation in any amount of repetitions. Thus, we use animation cycles. An animator will break down a complex animation into a set of key poses and set the positional and rotational values of each joint or blendshape. Figure 13 shows the working environment of an animator, with various control objects to manipulate the animation rig. When played back, the engine will interpolate between these poses and make the motion seem believable. For example, many of the cellular structures in Meta!Blast, such as the ATP synthase protein complex, have been looped to play continuously. The beginning of the animation cycle must line up with the end to prevent any jarring motions or jerking. If we had several alternate cycles, they would be played based on certain trigger events, and be seamlessly integrated into the main animation cycle. What we end up with is a consistent, seemingly interactive environment.
5. ARTIFICIAL INTELLIGENCE

5.1 Virtual Agents and Motion Planning

Virtual agents (or autonomous game characters) and virtual worlds are the hallmark of computer games and simulations and are important in robotics, remote collaboration, simulation and training, and educational projects. Virtual agents stimulate believability, contribute to a sense of immersion, enhance user experience and engender a sense of realism and interaction in the virtual world in which they exist. An extremely important feature of virtual agents is path-planning, or the ability to independently traverse the cell without colliding with other agents or objects. Many consider path-planning as an essential requirement for virtual agents to move around a virtual world in a life-like manner. In a multifaceted 3D world such as the cell scene in Meta!Blast, this is a daunting computational task. A well-designed path-planning mechanism can significantly enhance the overall performance of the game. If path-planning mechanisms are not implemented efficiently and correctly in the game, the player is distracted and the overall game play experience is reduced. For instance, in Meta!Blast, the believability of the game would be reduced if one of the proteasomes (here considered the virtual agent) is allowed to “pass through” a chloroplast or the vacuole.

With technological advances in computer graphics, 3D virtual worlds are becoming increasingly detailed and complex. In most computer games, path-planning is the lowest level of intelligence. Despite this, many challenges arise in developing an effective path-planning system. Meta!Blast represents a particular challenge, because unlike most other games, the gameplay itself occurs in three dimensions. In Meta!Blast, we consider factors including the virtual agent’s degree of freedom and the complexity of the virtual cell.

5.2 Octrees and Steering Behaviors

In a 3D virtual environment such as Meta!Blast, the world needs to be represented in a way that enables a good perception of its navigational properties so that path-planning algorithms may be executed more efficiently. One of the techniques we use to partition the cell scene is an octree. An octree is a graph in which each node has either exactly eight nodes or no nodes. Each node represents a position in the cell scene that an agent can occupy at any time. We use an A* search algorithm to generate a path between a start and goal location. This algorithm traverses the nodes of the octree and returns a path for the agent to follow if one exists.
As shown in Figure 14, octrees can be used to closely approximate complex irregular models, which helps identify tight free spaces for the virtual agents to travel. Octrees are more efficient than a uniform grid because they can greatly reduce the number of nodes to search. However, there are also two major disadvantages to octrees that can lead to reduced in-game efficiency: first, the time in-game that it takes to construct an octree increases exponentially with the number of polygons in the scene; also, the octree cannot remain up to date during gameplay without reconstruction. To remedy this, we use steering behaviors to supplement the octree. A steering behavior produces motion based on “physical” properties in the game world, such as the Meta!Blast bioship’s velocity and position. As shown in Figure 15, we use the pursue behavior to steer the ubiquitin molecules to fasten to the bioship. The advantages to using steering behaviors are that they remain up-to-date in each frame and that they can also be combined and programmed to generate new behaviors for virtual agents, such as flocking or crowd simulation. By incorporating both octree and steering behaviors into Meta!Blast, we benefit from the advantages of both methods.

![Figure 14. Octree for the cell.](image1)

![Figure 15. Screen shot from within the Meta!Blast cell: ubiquitin molecules being attached to the bioship (upper left) and targeting it for destruction. The pursue steering behavior is used to achieve this result.](image2)

6. CONCLUSIONS

Making a game is difficult. Making a scientifically accurate game about the complex systems of the cell is even more difficult. By working together, biologists, artists, and programmers hope to provide educators another tool for teaching science and to inspire a new generation to understand the structure and function of living cells. Using scientific data such as micrographs and PDB files, industry standard tools like Maya, Photoshop, and the Unity 3D game engine, as well as
techniques such as digital sculpting, light mapping, octtrees and steering behaviors, the Meta!Blast developers have
developed the most robust and accurate virtual cell game to-date.

With our ability to show this dynamic and complex cell with visuals that have in the past been reserved for pre-rendered
animations, we believe that Meta!Blast will lead to a greater understanding of the cellular world, and the processes and
reactions contained within. This software will allow educators to supplement their traditional curriculum with an
immersive look at the spatial and time scale relationships that static textbooks and diagrams could never portray.

ACKNOWLEDGMENTS

We are grateful for support from National Institutes of Health, SEPA IR25RR025147 and National Science Foundation
MCB-0951170 to ESW.

Molecular graphic imagery was produced using the Molecular Maya Toolkit (mMaya),
www.molecularmovies.org/toolkit.

REFERENCES


(2002).


[7] Linde, Riccard, [Game Art: Creation, Direction, and Careers], Charles River Media, Inc., Hingham, Massachusetts
(2005).


(2009).

Indianapolis, Indiana (2007).


