

VISUALIZING B CELL DEVELOPMENT:  
CREATING AN IMMUNOLOGY VIDEO GAME

By  
Emily Lunhui Ling

A thesis submitted to Johns Hopkins University in conformity with  
the requirements for the degree of Master of Arts

Baltimore, Maryland  
March, 2016

© 2016 Emily L. Ling  
All Rights Reserved

## ABSTRACT

The foundational immunology concepts of lymphocyte development are important for beginning science students to comprehend. Video games offer the potential for a novel approach to teaching this complex subject matter by more effectively engaging students in this material. However, currently available educational video games intended to teach immunology have distinct limitations such as a lack of explicit demonstrations of the stages of lymphocyte development and clonal selection.

This project identifies the content focus and gameplay mechanics of currently available immunology video games. Using this as a basis, a novel approach for developing an immunology video game was outlined with the primary goal of improving integration of educational content. A proof of concept was developed for the B lymphocyte development portion of the game content and a partial prototype was developed in Unity 5 3D.

The important contribution of this thesis was the development of a new approach to designing a more effective educational video game specifically for immunology. Outcomes of this research will serve to inform future biomedical communicators on how to develop content for active learning games in immunology and provide a guide for designing full length educational video games featuring novel gameplay mechanics such as those identified through this project.

Emily L. Ling

## CHAIRPERSONS OF THE SUPERVISORY COMMITTEE

### *Thesis Preceptor*

**Mark J. Soloski, Ph.D.**, Professor of Medicine

Departments of Medicine, Pathology, Molecular Biology and Genetics,  
and Molecular Microbiology and Immunology

Director, Immunology Training Program

The Johns Hopkins University School of Medicine

### *Departmental Advisor*

**David A. Rini, M.F.A., C.M.I., F.A.M.I.**, Associate professor

Department of Art as Applied to Medicine

The Johns Hopkins University School of Medicine

## ACKNOWLEDGMENTS

I am grateful to have been able to work on a thesis project that combines many of my interests. This thesis would not have been possible if not for the support and encouragement of many individuals.

First and foremost, I would like to thank, **Dr. Mark Soloski**, for agreeing to be my thesis preceptor. I am grateful for his support, expertise, and enthusiastic feedback, which helped guide the development of my project.

Thank you to my faculty advisor, **David A. Rini**. His guidance and feedback helped me organize the focus and creative objectives of my project. His insights were invaluable and I thank him for his responsiveness and availability.

Thank you to my department director, **Corinne Sandone**, for helping me create, and for approving, my project proposal .

I want to thank the **Class of 2016** for their support, feedback, and openness to exchanging creative ideas. Special thanks to **Dacia Balch** for being the caring, understanding, and amazing individual that she is.

I want to thank the immunology graduate students, **Andriana Lebid**, **Brandom Lam**, **Brian Christmas**, and **Ali Ghasemzadeh**, for their feedback and help in developing the project's game content.

Thank you to **Alex Rosensweet**, for lending his amazing vocal talents for the narrative portions of my proof of concept.

Finally, a lifetime of thanks and gratitude to my mother, **Mei Yu Lee**, for her loving patience and encouragement to pursue my goals and dreams, and to my sister, **Kimberly L. Ling**, for setting examples and being my lifelong friend.

This thesis is dedicated to my late father,

**Morgan H. Ling**

His belief in the pursuit of knowledge,  
and that nothing is too difficult as long you try harder,  
created the core of who I am today.

## TABLE OF CONTENTS

ABSTRACT.....	ii
CHAIRPERSONS OF THE SUPERVISORY COMMITTEE .....	iii
ACKNOWLEDGMENTS.....	iv
INTRODUCTION.....	1
Video Games and Learning .....	2
Definition of “Educational Video Games” as “Games for Learning”.....	2
Current Serious Games with a Focus on Immunology .....	4
The Immune System Defender .....	5
Immune System Responses .....	7
Immune Defense.....	9
Immune Attack .....	11
ImmuneQuest™ .....	13
Problems with Gameplay of Currently Available Titles.....	15
Unintuitive Game Mechanics .....	15
Speed of Game Content Presentation: Too Fast or Too Slow.....	16
Distracting Heads-Up-Display elements (HUD).....	19
An Opportunity to Redesign Content Focus and Gameplay for an Educational Immunology Video Game .....	20
Changing the Narrative Perspective To Focus on the Development of Lymphocytes.....	21
Constructivism Learning Theory and Exploring Genres of Gameplay Mechanics used in Off-the-Shelf Video Games.....	22
The Stanley Parable - Narrative.....	23
Journey - Adventure, Exploration, Atmospheric.....	24

Never Alone (Kisima Ingitchuna) - Atmospheric, Narrative.....	25
Goals and Objectives.....	27
Audience.....	27
MATERIALS AND METHODS.....	28
Developing the Game Content.....	28
Textbook References and Content Expert.....	28
Flowcharting the Game Content.....	28
Gameplay Flowchart of B and T Lymphocytes.....	28
Developing the Narrative Script of the Game Content.....	29
Gameplay Design.....	33
Developing Main Gameplay Design.....	33
Developing Mini Game Gameplay Design.....	34
Visual Development of the Game.....	35
Developing the Bone Marrow Microenvironment.....	35
Developing the Characters.....	36
Developing the HUD.....	38
Proof of Concept Storyboards.....	39
Narration Recording.....	39
Building the 3D Prototypes of Proof of Concept.....	40
Modeling Software.....	40
Modeling the Multipotent Stem Cell.....	40
Modeling the Bone Marrow Environment.....	43
Modeling Central Bone Trabeculae in ZBrush.....	46

Modeling Vessels in C4D.....	47
Optimizing Vertex Count and Surface Meshes in ZBrush.....	48
Creating UV Textures for Models in ZBrush.....	49
Reimporting into C4D and Exporting as FBX for use in Unity 5 3D.....	49
Unity 5 3D Game Development Platform Software: Assembling Components.....	50
Free Edition of Unity 5 3D's Game Engine.....	50
Creating the Main Menu and Exit Screens in Unity 5 3D.....	51
Custom Particle Effect Mesh For Main Menu Screen.....	51
Setting Up Multiple Scenes.....	52
Atmospheric Rendering Effects.....	53
RESULTS.....	54
Models Created from Concept Designs.....	55
Multipotent Stem Cell.....	55
Bone Marrow Environment.....	55
Blood Vessels and Stromal Cells.....	56
Still Images for Interface and Gameplay Mockups.....	57
Main Game and Mini Game Mockups.....	57
Prototyped Scenes in Unity 5 3D.....	58
Main Menu and Exit Interfaces.....	58
Setup for Cutscene Sequence.....	60
Scene 1 Gameplay Stage - Multipotent Stem Cell.....	61
Access to Assets Resulting from this Thesis.....	62



DISCUSSION .....	63
Rethinking Content and Gameplay for Immunology Video Games.....	63
Unity 5 3D.....	64
Process of Modeling Assets for Unity 5 3D.....	65
Learning Scripting, and Prototyping for Unity 5 3D in Segments.....	65
Challenges of Scripting Character Control Physics.....	66
Future Goals and Directions for Game Prototype Development .....	67
CONCLUSION .....	68
APPENDICES .....	69
Visual Development Sketches .....	69
Blood Vessel Capillary Anatomy Studies .....	69
Bone Marrow, Micro Environment Studies and Concepts.....	70
Character and Movement Concepts .....	74
Character Design Sheet .....	76
Storyboards.....	80
Main Menu Layout Concepts .....	80
Narration Script.....	84
C# Scripts .....	104
REFERENCES .....	116
General References .....	119
Online Educational Video Game Directories .....	119
Games Referenced.....	120
VITA .....	121

## INDEX OF FIGURES

Figure 1	The Immune System Defender in-browser launch menu link. . . . .	5
Figure 2	The Immune System Defender in-browser screen capture of gameplay introduction. . . . .	6
Figure 3	The Immune System Defender in-browser screen capture of the first level. . . . .	6
Figure 4	Immune System Responses in-browser launch menu link. . . . .	7
Figure 5	Immune System Responses in-browser screen capture of the introduction scene (cropped). . . . .	8
Figure 6	Immune System Responses in-browser screen capture of one of the learning modules. . . . .	8
Figure 7	Immune Defense’s in-browser launch and main menu . . . . .	9
Figure 8	Immune Defense in-game screen capture of the Level Selection menu. . . . .	10
Figure 9	Immune Defense in-game screen capture of gameplay tutorial . . . . .	10
Figure 10	Immune Attack in-game screen capture. . . . .	11
Figure 11	Immune Attack’s in-game screen capture of an on-screen information window. . . . .	12
Figure 12	ImmuneQuest™ in-game screen capture of the main menu. . . . .	13
Figure 13	ImmuneQuest™ in-game screen capture of the macrophage character . . . . .	14
Figure 14	ImmuneQuest™ in-game screen capture of the complement character . . . . .	14
Figure 15	Immune System Defender’s in-game screen capture of the gameplay mechanics. . . . .	16
Figure 16	Immune Defense’s in-game screen capture of the gameplay tutorial’s on-screen prompts . . . . .	17
Figure 17	Immune Defense’s in-game screen capture of the gameplay tutorial’s on-screen prompts . . . . .	17

Figure 18	Immune Defense’s top screen HUD elements . . . . .	19
Figure 19	Immune Defense’s bottom screen HUD elements. . . . .	19
Figure 20	The Stanley Parable’s in-game screen capture of a player choice scene . . . . .	23
Figure 21	The Stanley Parable’s in-game screen capture of a player choice scene . . . . .	23
Figure 22	Journey’s in-game screen capture of on-screen gameplay control prompts. . . . .	24
Figure 23	Journey’s in-game screen capture of the game environment. . . . .	25
Figure 24	Never Alone (Kisima Ingitchuna) in-game screen capture of a cutscene. . . . .	25
Figure 25	Never Alone (Kisima Ingitchuna)’s in-game screen capture of the unlocked video content screen . . . . .	26
Figure 26	Flowchart of the main menu organization of the project. . . . .	30
Figure 27	Flowchart of the general event tree for both the B and T lymphocyte development and clonal selection portions of the game content . . . . .	31
Figure 28	Flowchart of the proof of concept B lymphocyte branch of the project . . . . .	32
Figure 29	Conceptual designs of the CXCL12 particle system . . . . .	33
Figure 30	Concept render and mock up of health gauge feature . . . . .	34
Figure 31	Telltale Games, Wolf Among Us in-game screen captures of real-time gameplay design feature. . . . .	34
Figure 32	Concept render and mock up of mini-game gameplay design and on-screen prompts . . . . .	35
Figure 33	Conceptual design of game environment and elevation change. . . . .	36
Figure 34	ImmuneQuest™ in-game anthropomorphic macrophage character. . . . .	37
Figure 35	C4D Multipotent Stem Cell character with Joints . . . . .	40

Figure 36	C4D Joint Tool location . . . . .	41
Figure 37	Controller parenting to grouped Joints in C4D . . . . .	42
Figure 38	C4D mesh weighting adjustments using the Weighting Tool . . . . .	42
Figure 39	Surface receptor alignment with a Target Effector in C4D. . . . .	43
Figure 40	Compact bone and bone marrow anatomy concept designs . . . . .	44
Figure 41	C4D Proc3Durale plugin use on a Parametric object without Simulate Cloth Surface . . . . .	45
Figure 42	Relative Position on Source object . . . . .	45
Figure 43	C4D viewport of the Proc3Durale object in the World Coordinate Position 0x, 0y, and 0z. . . . .	45
Figure 44	ZSphere use for modeling individual trabeculae . . . . .	46
Figure 45	Adaptive skin to convert the ZSphere into a meshed surface . . . . .	46
Figure 46	ZBrush surface difference with use of DynaMesh . . . . .	48
Figure 47	Unity 5 3D mesh splitting of a model into multiple sub-65k vertex count parts . . . . .	48
Figure 48	C4D CV-SmartExport batch exporter settings . . . . .	49
Figure 49	Unity 5 3D Particle System Renderer options . . . . .	52
Figure 50	Unity 5 3D's Build Settings menu . . . . .	52
Figure 51	Unity 5 3D's Build Settings and added scene hierarchy numbers . . . . .	53
Figure 52	C4D test renders of the Multipotent Stem Cell character with various surface receptor variations . . . . .	55
Figure 53	ZBrush optimized bone marrow environment model . . . . .	56
Figure 54	Bone marrow model imported into Unity 5 3D. . . . .	56
Figure 55	Optimized vessel models and trabeculae imported into Unity 5 3D . . . . .	57
Figure 56	Example mock up of main gameplay's health gauge system . . . . .	58

Figure 57	Unity 5 3D in-game demo screen capture of the prototyped main menu .....	59
Figure 58	Unity 5 3D in-game demo screen capture of the prototyped exit menu.....	59
Figure 59	Unity 5 3D in-game demo screen capture of a scene stage for subsequent addition of a cutscene .....	60
Figure 60	Unity 5 3D in-game demo screen capture of the loading screen after “space bar” key down is triggered on the keyboard .....	60
Figure 61	Unity 5 3D prototype build of Scene 1.....	61
Figure 62	Unity 5 3D prototype build of Scene 1 with Particle Systems created to enhance atmosphere turned on .....	61
Figure 63	Unity 5 3D in-game demo screen capture of Scene 1 with placeholder sphere for the character controller .....	62

# INTRODUCTION

Immunology, the study of the body's defense against infection (Murphy et al. 2012), influences disciplines such as clinical medicine, public health, and cancer research (Kaufmann 2007). The clonal selection theory introduced in 1959 (Burnet 1959) led to research on T and B cell (or lymphocyte) receptor specificity, antigen recognition, and immunological memory (Coutinho 1989; Mackay 1991). Further research subsequently clarified the acquisition of self-tolerance in developing lymphocytes (Hoffmann 1975; Janeway 1989; Jerne 1971; Lederberg 1959; Schwartz 1989). These paradigms inform many aspects of present-day immunology (Coutinho 1989; Janeway 1989).

Many of these concepts are hard to comprehend for beginning science students, due to the multiple stages involved in lymphocyte development. Most resources such as lectures, 2D illustrations, and animations provide only passive learning opportunities. While new programs and technologies allow for the creation of interactive media, which presents a promising new platform for active learning (Faust and Paulson 1998; Freitas and Neumann 2009); current directories of educational video games (see General References: Online Educational Game Directories) list more resources in the genres of history and math than in science. Of those games listed, only a fraction were specific to immunology, and none contained explicit demonstration of the stages of lymphocyte development and clonal selection.

Therefore, the goal of this project is to identify the focus of content and gameplay mechanics of currently available immunology video games. Then, using this as a basis, a novel approach to developing an immunology video game with a focus on lymphocyte

development will be outlined with the goal of improving integration of educational content. It will serve as an improved model for integrating narrative gameplay and educational content presentation.

## **Video Games and Learning**

### **Definition of “Educational Video Games” as “Games for Learning”**

To effectively discuss educational video games, it is valuable to establish a definition of what constitutes a “game”. From establishing the definition of a “game”, the purpose of an “educational video game” can be clarified.

Juul et al. (2003) described six common features that are necessary and sufficient for defining a game:

A game is a rule-based formal system with a variable and quantifiable outcome, where different outcomes are assigned different values, the player exerts effort in order to influence the outcome, the player feels attached to the outcome, and the consequences of the activity are optional and negotiable.

To refine this definition for “educational video games” as games with an educational purpose, Young et al. (2012) quotes Klopfer et. al. (2009) to describe digital-learning games as:

...those that “target the acquisition of knowledge as its own end and foster habits of mind and understanding that are generally useful or useful within an academic context. Learning Games may be associated with formal educational environments (schools and universities, online or offline), places of informal learning (e.g. museums), or self-learners interested in acquiring new knowledge or understanding” (63-4).

Therefore, for this thesis, an educational video game successfully utilizing narrative gameplay to present educational content will include the following:

1. An educational content driven rule-based system
2. Contain various and quantifiable outcomes
3. Outcomes will be based on a positive or negative value relative to each other
4. Challenges the player to achieve a certain outcome
5. Emotional investment by participants
6. Consequences of the game activity that are optional and variable based on the player's interaction with the defined system.

Games for learning can also be categorized as a “serious game”. Serious games are those designed for an audience with an objective beyond pure entertainment (Bellotti et al. 2013). Besides being fun, entertaining, attractive, and appealing to a target audience, a serious game aims to fulfill an educational goal (Bellotti et al. 2013).

Finally, in the context of this project, it is important to distinguish the difference between “games for learning” and the term “gamification”. Most sources define “gamification” as “the use of game design elements in non-game contexts” to enhance “services with (motivational) affordances in order to invoke gameful experiences and further behavioral outcomes” (Hamari et al. 2014). For instance, “Gamification” design may be applied to programs that teach management skills, or applications such as health and wellness trackers. Therefore, “gamification” will be considered a design process encouraging habit formation, which is outside the scope of this project's focus.



## Current Serious Games with a Focus on Immunology

To determine the availability of serious games for immunology, various online serious game databases were utilized to search for video games categorized by key words such as science, biology, and cell biology (see REFERENCES: Online Educational Video Game Directories). While these databases revealed a range of serious games for science, such as physics, chemistry, evolutionary biology, and genetics, only a fraction focused on educational content related to immunology. For instance, of an approximate 95 educational video games listed on the Science Game Center's website (<http://www.sciencegamecenter.org/games>), a database of math and science games, only three titles were found that focused on concepts in immunology: Immune Attack, ImmuneQuest™, and Immune Defense. Two additional flash-based online games, The Immune System Defender, and Immune System Responses, were identified from the Nobel Prize Educational Medicine database listings.

Games focusing more heavily on concepts of virulence or immunization were excluded from the search since such games focused their educational content on concepts in public health and epidemiology.

These five games were the only immunology based video games identified using online search engines and databases (see REFERENCES: Online Educational Video Game Directories) and to the author's best knowledge are the only such games available.

## The Immune System Defender



Figure 1 The Immune System Defender in-browser launch menu link.

Immune System Defender is an online flash-based interactive game. It can be found through the Nobel Prize online education database (<http://www.nobelprize.org/educational/medicine/immUnity53D/>). The game uses flat 16-bit pixel-like graphical game elements. The game is presented as a mission briefing and the goals center around defending the host from a possible splinter induced infection. The player begins by controlling immune cells and launching them towards bacteria. As the player completes each level, the difficulty of clearing the infection increases. In addition to increasing difficulty, new immune cells with different gameplay abilities are introduced with each level.

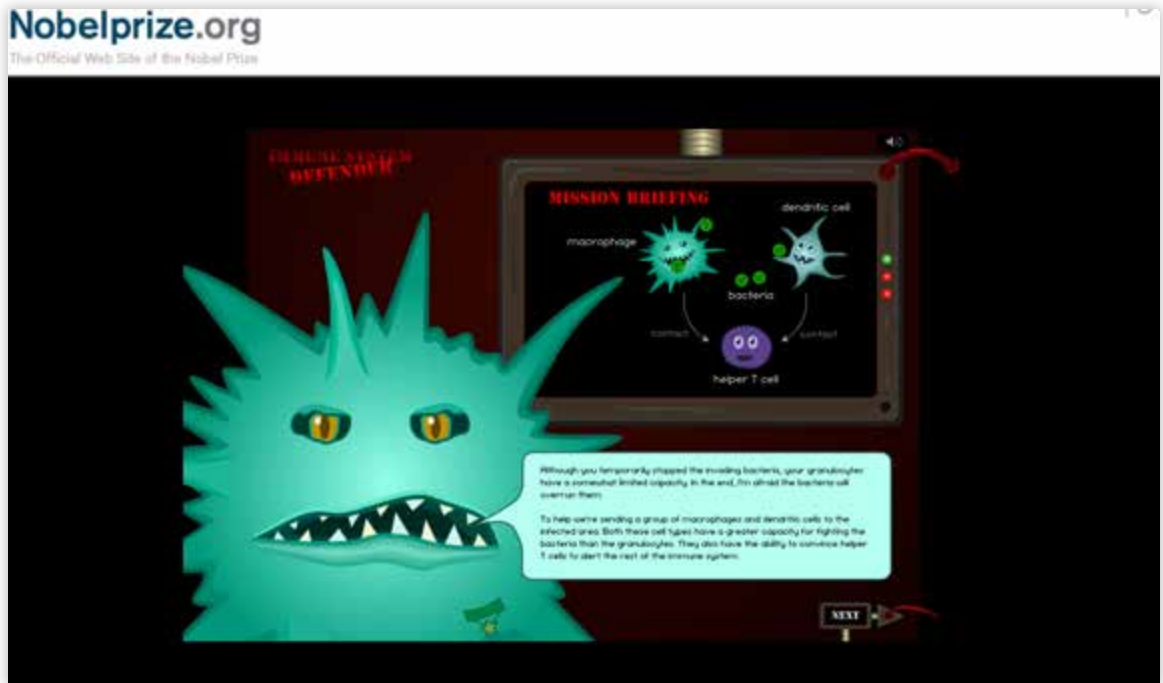


Figure 2 The Immune System Defender in-browser screen capture of gameplay introduction.



Figure 3 The Immune System Defender in-browser screen capture of the first level.

## Immune System Responses



Figure 4 Immune System Responses in-browser launch menu link.

An online flash-based interactive, Immune System Responses can be found through the Nobel Prize online educational database (<http://www.nobelprize.org/educational/medicine/immuneresponses/>). This game utilizes a cartoon stylized representation of 2D animated graphic characters and immune cell elements. Gameplay mechanics are centered around clicking and dragging to progress through the game content separated into a series of 10 modules. Each playable module explains in on-screen text windows how the immune system and its various cells respond to pathogens. The game also aims to explain to players how the immune system functions to fight off illnesses. Players may also access more information about the immune cells featured in each module through expanding nested menu items or videos.

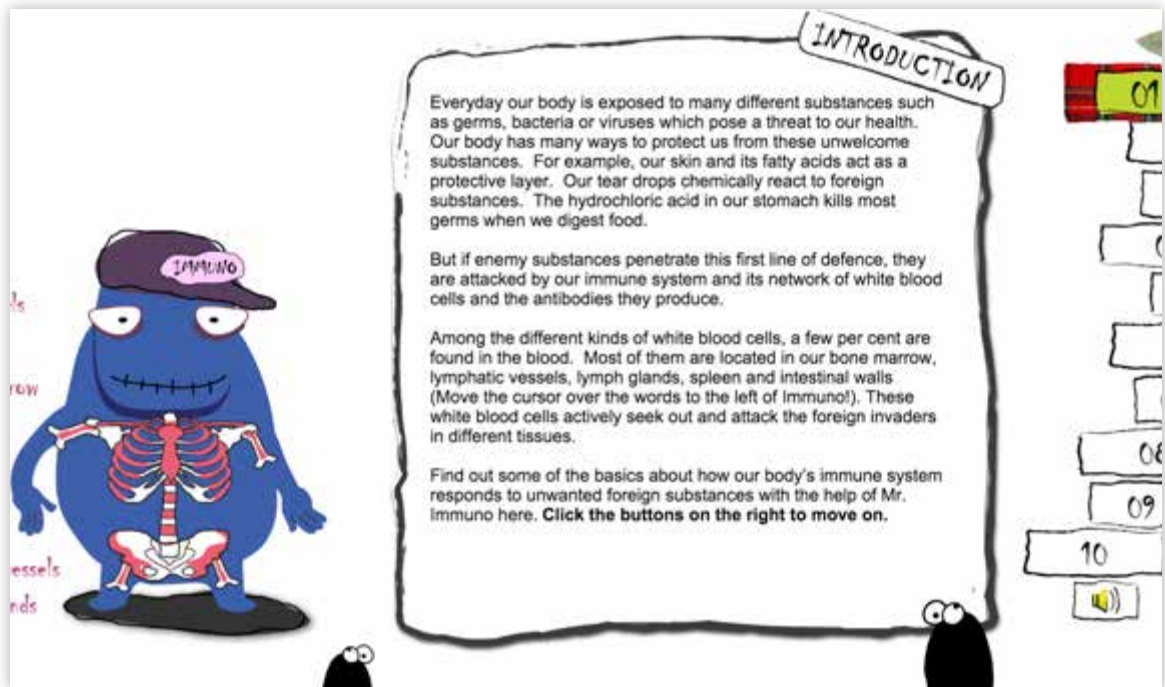


Figure 5 Immune System Responses in-browser screen capture of the introduction scene (cropped).



Figure 6 Immune System Responses in-browser screen capture of one of the learning modules. Text in this image is not intended to be read.

## Immune Defense



Figure 7 Immune Defense's in-browser launch and main menu.  
Text in this image is not intended to be read.

Created by Molecular Jig Games in Unity 5 3D and published to run in online web browsers, Immune Defense is a 2D based immunology game (<http://www.molecularjig.com>). Gameplay mechanics involve clicking and dragging game elements on-screen to complete levels in the game. The educational goal of Immune Defense is to visualize the abstract concept of protein driven and mediated immune cell activities and responses to pathogens. On-screen windows presents the users with game objectives and prompts. Additional scientific information about components of the game can be accessed through expandable windows.

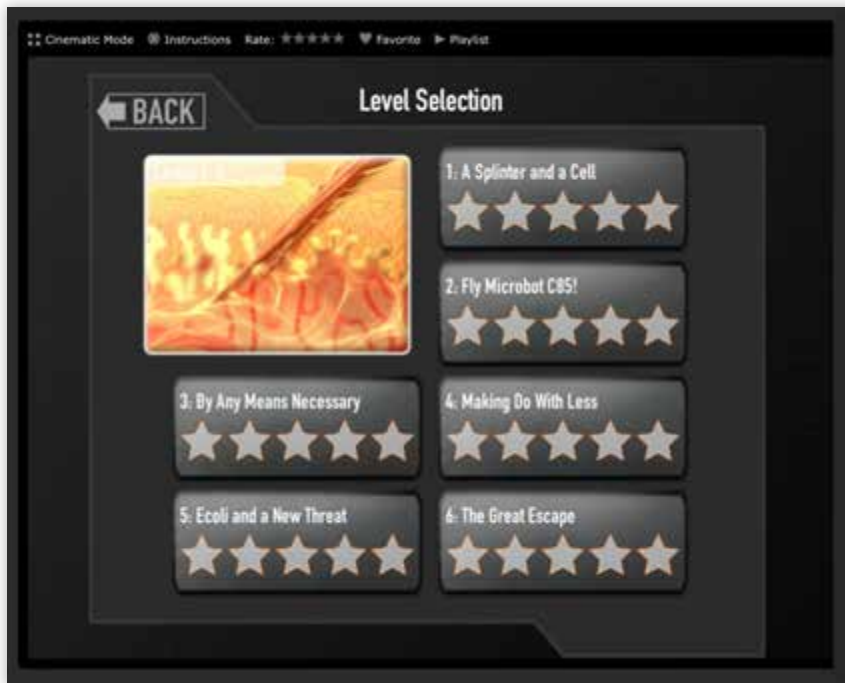


Figure 8 Immune Defense in-game screen capture of the Level Selection menu. Text in this image is not intended to be read.



Figure 9 Immune Defense in-game screen capture of gameplay tutorial. Text in this image is not intended to be read.

## Immune Attack

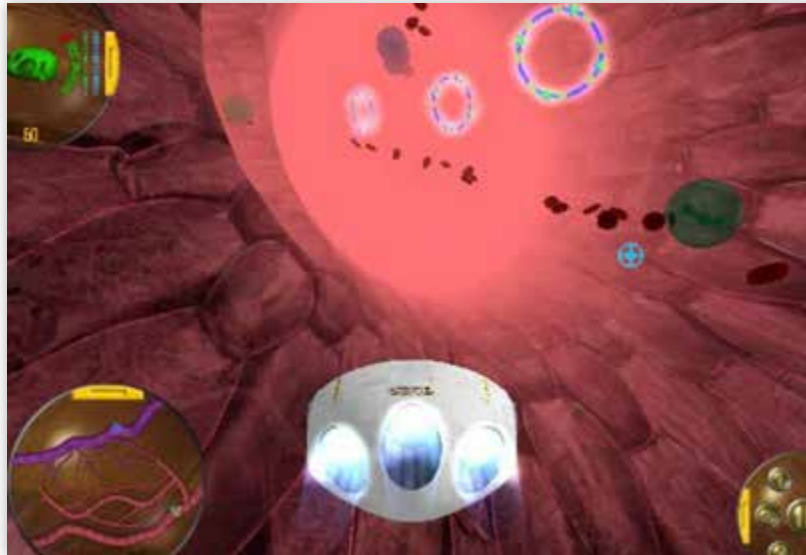


Figure 10 Immune Attack in-game screen capture.

Available for download online (<http://immuneattack.org/>), Immune Attack is a 3D action adventure game available only for PC Microsoft platforms. Drawing from science fiction narrative elements, the player controls a nano-robot to direct immune cells to inflamed or infected areas of the body. The game is a narrative driven game, where players embody the goals of an immunocompromised researcher who has injected himself or herself with a nano-robot to “re-train” their immune cells to function normally. Various non-player characters (NPCs) instruct the player on how to navigate and what the game level objectives are.





Figure 11 Immune Attack's in-game screen capture of an on-screen information window. Text in this image is not intended to be read.

## ImmuneQuest™



Figure 12 ImmuneQuest™ in-game screen capture of the main menu. Text in this image is not intended to be read.

Developed by the Federation of American Scientists, ImmuneQuest™ is a turn-based strategy video game created by Syandus, Inc. (<http://ImmuneQuest.com/>). The story begins when bacteria are introduced from a splinter. Players control a set of anthropomorphic macrophages on a hexagonal 2.5D grid. A narrator instructs the user on how to control the immune cells and complete objectives, while more detailed information on gameplay are also presented as on-screen prompts. Additional information regarding the cells of the immune system (the characters) being represented in the game can be accessed through expandable secondary windows and menus. User comprehension is tested through occasional test questions that are triggered by accessing in-game bonus items. Answering questions correctly rewards the player with advantages for use in completing objectives in the game.



Figure 13 ImmuneQuest™ in-game screen capture of the macrophage character. Text in this image is not intended to be read.



Figure 14 ImmuneQuest™ in-game screen capture of the complement character. Text in this image is not intended to be read.

## **Problems with Gameplay of Currently Available Titles**

In Young et al.'s review of trends in serious gaming for education, content findings for science video games noted a lack of coherence with the educational concepts presented to players and the gameplay environment in which they are applying the information (Young et al. 2012). It was suggested that the application of factual information presented to the game user was not as effective as the gaming narratives utilized in another serious games created for content such as history (Young et al. 2012).

In addition to the limitations listed above, the serious games mentioned have gameplay mechanics and elements that can distract from the educational content they are intended to enhance.

### **Unintuitive Game Mechanics**

Game, or gameplay, mechanics are “actions invoked by an agent”, such as the player, “to interact with a game world, as constrained by the game rules” (Sicart 2008). Easy to understand and use controls for interacting with a game world allows a player to focus on the tasks presented. This also allows the player to properly control their experience and access to the game content. An inability to control the outcomes, or direction of a game, due to poor gameplay mechanics and design, can cause frustration and aggression, leading to a player quitting gameplay (Lopes 2014). Another possible outcome is performance anxiety, since difficult controls can lead to limited attention to gameplay prompts and gameplay progression.

For instance, in *The Immune System Defender*, players are required to click-and-

drag the game element they wish to move. To propel the cells in the desired direction, players must launch the game element by pulling counter to or opposite of the desired direction. This type of control would require a learning period to understand. However, players are only allowed to see a fraction of where the game element is being launched, and in most cases can only interpret where the desired direction will be since the rest of the game field not within view is represented in a small navigation window (Fig. 15).



Figure 15 Immune System Defender's in-game screen capture of the gameplay mechanics. The player control area (A) only allows visibility of a small portion of the map represented in the mini map (B).

### Speed of Game Content Presentation: Too Fast or Too Slow

Speed of content presentation can cause frustration depending on how it is used with the game mechanics. This is generally less problematic in a turn-based system such as ImmuneQuest™. If too much is occurring for the player to follow the events on-screen,

then the gameplay mechanics or content can become confusing.

Immune Defense, for instance, creates a feeling of quick gameplay which inadvertently distracts from the game content. For example, during the gameplay tutorial, arrows and on-screen text prompts are presented as flashing graphical elements directing a player's attention to actionable triggers (Fig. 16). At the same time, during gameplay instruction, the player's focus is divided by the quick movements of pathogens and cells within the main view of the game environment (Fig.16, Fig. 17). These two features can combine to create a feeling of urgency for the player, which can then contribute to player frustration due to the lack of control over gameplay (Lopes 2014).

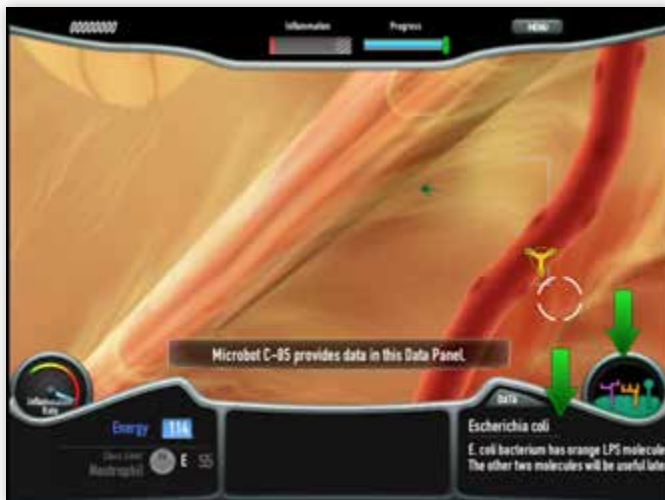


Figure 16 Immune Defense's in-game screen capture of the gameplay tutorial's on-screen prompts. Text in this image is not intended to be read.



Figure 17 Immune Defense's in-game screen capture of the gameplay tutorial's on-screen prompts. Text in this image is not intended to be read.

One way to minimize urgency is to allow players to pause and process important details of gameplay. Long lines of texts requiring player attention should be presented in controlled pieces, one at a time if possible or as bullet points. In addition to this, moving gameplay sequences in the background should either be stationary or paused for the player to comprehend the next set of gameplay details.

In other instances gameplay can be frustrating if the content is presented too slowly. This can be largely resolved if players are allowed the option to control information speed, skip narrations, or still interact with the environment while narration or rules are being presented. In addition, the content can be further elucidated if the narration has subtitles.

For instance, in Immune Attack the game objectives are narrated to the player, as well as specific scientific content that would establish the parameters of the game objectives. While the pace may be appropriate with players learning the content of the game, the lack of on-screen subtitles and options to skip narration could lead to irritation in players accustomed to faster material presentation. In addition to the slow narration pace, the player controls are inactive during narrative sequences without indications to the player that the gameplay has been paused. To retain player interest, slow narration would benefit from allowing the player to continue moving within the game environment while listening to the game objectives.

## Distracting Heads-Up-Display elements (HUD)

Heads-Up-Displays are persistent on-screen elements that allow the player to track their progress, game conditions, or other features set forth by the game design (Wilson 2006). These may include player health status, goals achieved and may also house sets of actions a player can trigger for gameplay.



Figure 18 Immune Defense's top screen HUD elements. Text in this image is not intended to be read.



Figure 19 Immune Defense's bottom screen HUD elements. Text in this image is not intended to be read.

Some of the identified immunology games have HUD components featuring various gameplay actions and details for the player to focus on in addition to the immediate activities being presented. However, most of these features can “distract the player from the environment in which he or she is immersed” when combined with other factors such as the speed at which gameplay information is presented (Wilson 2006). For instance,



in the game Immune Defense “Inflammation” is tracked in a top screen meter (Fig. 18) in addition to an “Inflammation Rate” throttle-like gauge (Fig. 19) in the bottom left of the screen. Both add visual complexity to the scene. However, these elements hinder the main view of the game’s environment and can divide the player’s attention from areas of gameplay.

## **An Opportunity to Redesign Content Focus and Gameplay for an Educational Immunology Video Game**

One theme common among the currently available titles is their narrative point-of-view. These games involve controlling a robot to interact with the game’s immune system cells and environment, or they involve controlling the immune cells from a third-person perspective. What has not been attempted and therefore would be a novel approach, is to create a game from a second-person narrative, third-person camera perspective of controlling and existing in a game as a developing immune cell.

In a second-person narrative, the player is referred to by second-person pronouns, such as “you”. In the context of this game design, this narrative style would connect the player with the actions of the cell. The third-person camera perspective would allow players to follow the in-game character from behind while maintaining a view of the character within the environment.

## **Changing the Narrative Perspective To Focus on the Development of Lymphocytes**

Problems with gameplay and the narrative based approaches of currently available titles offer the opportunity to redesign an immunology video game with regards to both gameplay content presentation and gameplay mechanics. Thus, while currently available titles allow the player to control mature lymphocytes in the body, the thesis aims to go a step further by focusing on the development of lymphocytes: where they originate, how they differentiate into mature lymphocytes, and what they interact with in their environment during development.

Feedback on challenges of learning lymphocyte development and clonal selection was collected from a focus group of first to fourth year graduate students in immunology. B cell development was identified as more foundational for an understanding of lymphocyte development than the development of T cells. Content on lymphocyte development and selection was collected from a graduate student focus group, content experts, as well as standard immunology textbooks.

The innovative gameplay feature of this project is the focus on the user's exploratory experience of the game environment. The game is designed to incorporate third-person camera perspective, exploration, and adventure game genre designs. Minimal in-game user-interface menus coupled with narrations of user controlled actions will enhance the user's immersive gameplay experience of the game content.

## **Constructivism Learning Theory and Exploring Genres of Gameplay Mechanics used in Off-the-Shelf Video Games**

The project was designed following principles of constructivist learning theory. Constructivist philosophy on the nature of learning is based on the principle that individuals form or construct their own learning and understanding based on their experiences from situations (Schunk 2012).

Various gameplay mechanics were considered for their contribution in achieving constructivist principles in gameplay. Again, game, or gameplay, mechanics are “actions invoked by an agent”, such as the player, “to interact with a game world, as constrained by the game rules” (Sicart 2008).

Determining the optimal game mechanics for the proposed immunology game was necessary to create and enhance the player’s understanding of the educational content of the program. Therefore, to better understand how commercially available entertainment based games integrate gameplay narrative with gameplay mechanics, the following genres of gameplay and titles were considered. Selected game genres featuring qualities that would allow players to learn from self-directed exploration of a game environment are: narrative, adventure, exploration, and atmospheric.

## The Stanley Parable - Narrative

Created in 2011 by Davey Wreden of Galactic Cafe, The Stanley Parable is a first-person narration game where players explore an office environment as the character Stanley. A narrator describes the actions of the player, or narrates the desired instructions the player should adhere to. The player can explore the narrative by either following a choice described by the narrator or pursuing another choice offered to the player in visual cues, such as open doors or hallways (Fig. 20, Fig. 21).



**Figure 20** The Stanley Parable's in-game screen capture of a player choice scene.



**Figure 21** The Stanley Parable's in-game screen capture of a player choice scene.

Hearing the narrator reiterate the actions of the player reinforces what the player's are doing or how they are interacting with the presented environment. The multiple choices also encourage the player to explore the limits and possibilities of the environment. Another benefit of multiple choices for gameplay interaction is the integration of multiple narrative outcomes. This would reward replaying the game to discover additional options in the story, thereby increasing the player's exposure to the game content.

### **Journey - Adventure, Exploration, Atmospheric**

Released in 2012, Journey is an artistic adventure game created by thatgamecompany™. Players are introduced to the narrative of a robed figure wandering a desert of ruins and spaces of artistically crafted environments. No spoken words or texts are used to drive the story and players are instructed on the gameplay controls through minimally designed on-screen graphical prompts (Fig. 22).



**Figure 22** Journey's in-game screen capture of on-screen gameplay control prompts.



**Figure 23** Journey's in-game screen capture of the game environment.

The game uses color and contrasts in the environment to encourage players to explore regions of interest (Fig. 23). Interaction with elements, solutions to puzzles, and progression through the story are driven by the player. This game's visual aesthetics, atmospheric considerations, and minimal interface designs creates an immersive gaming experience for players.

#### **Never Alone (Kisima Ingitchuna) - Atmospheric, Narrative**



**Figure 24** Never Alone (Kisima Ingitchuna) in-game screen capture of a cutscene.

Developed by Upper One Games, Never Alone (Kisima Ingitchuna) is a 2014 game that explores the folklores and traditions of the Alaskan indigenous populations. Player

control switches between an Iñupiaq girl and her arctic fox to solve puzzles and progress through the game. As players continue through levels of mythological narratives, video content is unlocked and can be viewed through a live on-screen prompt, or accessed later in the main game menu.



Figure 25 Never Alone (Kisima Ingitchuna)'s in-game screen capture of the unlocked video content screen. Text in this image is not intended to be read.

One of the powerful features of this narrative game is the integration of video, which allows players to explore more information about the game's content (Fig. 25). Players are given the option to access unlocked video content during gameplay, which allows players immediate access to viewing the video. If they choose not to view the video, they can return to playing the game and access the videos at a later time. This method of transitioning between embedded content subsequently minimizes the disruption to immersive gameplay.

## Goals and Objectives

User engagement through active learning improves comprehension of new material (Faust and Paulson 1998; Schunk 2012), and video games support the creation of visually engaging and immersive interactive environments (Freitas and Neumann 2009). While there exists some educational video games for immunology, most lack intuitive gameplay mechanics and engaging visuals.

Therefore, this project will outline an interactive video game showing the steps of lymphocyte development and clonal selection. The goal of this project is to create a video game demonstration of B lymphocyte development and clonal selection using game mechanics not previously demonstrated in the serious video games intended to teach immunology. This project will serve as a feasibility study and proof of concept for a full length immunology video game.

Design objectives:

- To design intuitive game mechanics, similar to previously described commercially available entertainment video games, to optimize immersive gameplay in the context of this project's immunology video game.
- To use narrative commentary to drive user progress through the video game.

## Audience

The primary intended audience is high school students taking biology or advanced biology courses. Secondary audiences are undergraduate biology and pre-medical students. Other audiences may include members of the general public and video gamers, particularly those interested in the subject of lymphocyte development and immunology.



## **MATERIALS AND METHODS**

### **Developing the Game Content**

#### **Textbook References and Content Expert**

Janeway's Immunobiology 8th edition by Kenneth Murray (Murphy et al. 2012) was the primary reference used for researching the stages of B lymphocyte development. This textbook was recommended and provided by the content expert Dr. Mark Soloski, Director of the Johns Hopkins University Immunology Graduate Program. Janeway's Immunology is the main textbook used by immunology doctoral candidates at Johns Hopkins University for foundational courses in the Immunology Graduate Program. Feedback and consultation on the outlines and proposed game content were reviewed by Dr. Soloski at weekly meetings.

### **Flowcharting the Game Content**

#### **Gameplay Flowchart of B and T Lymphocytes**

The overall game content was flowcharted in Axure RP Pro 7.0.0.3189, an interactive wireframe, mock-up, and prototyping software. Basic organization of main menu features were developed in addition to a general event tree outlining the stages of both B and T lymphocyte development and clonal selection (Fig. 26, Fig. 27). This served to organize the gameplay content. The general gameplay event tree constitutes the focused scope of the immunology concepts proposed for the educational video game. From the general

gameplay event tree, a more detailed flowchart was developed for the B lymphocyte development branch of the game content. This portion of the game served as the outline for demonstrating the proof of concept (Fig. 28).

### **Developing the Narrative Script of the Game Content**

Narration for the script was organized as a screenplay in Adobe Story CC 2015. The screenplay format allowed for the organization of scene events in sequence with narrative dialogue developed from the researched game content. (see APPENDICES: Narration Script).

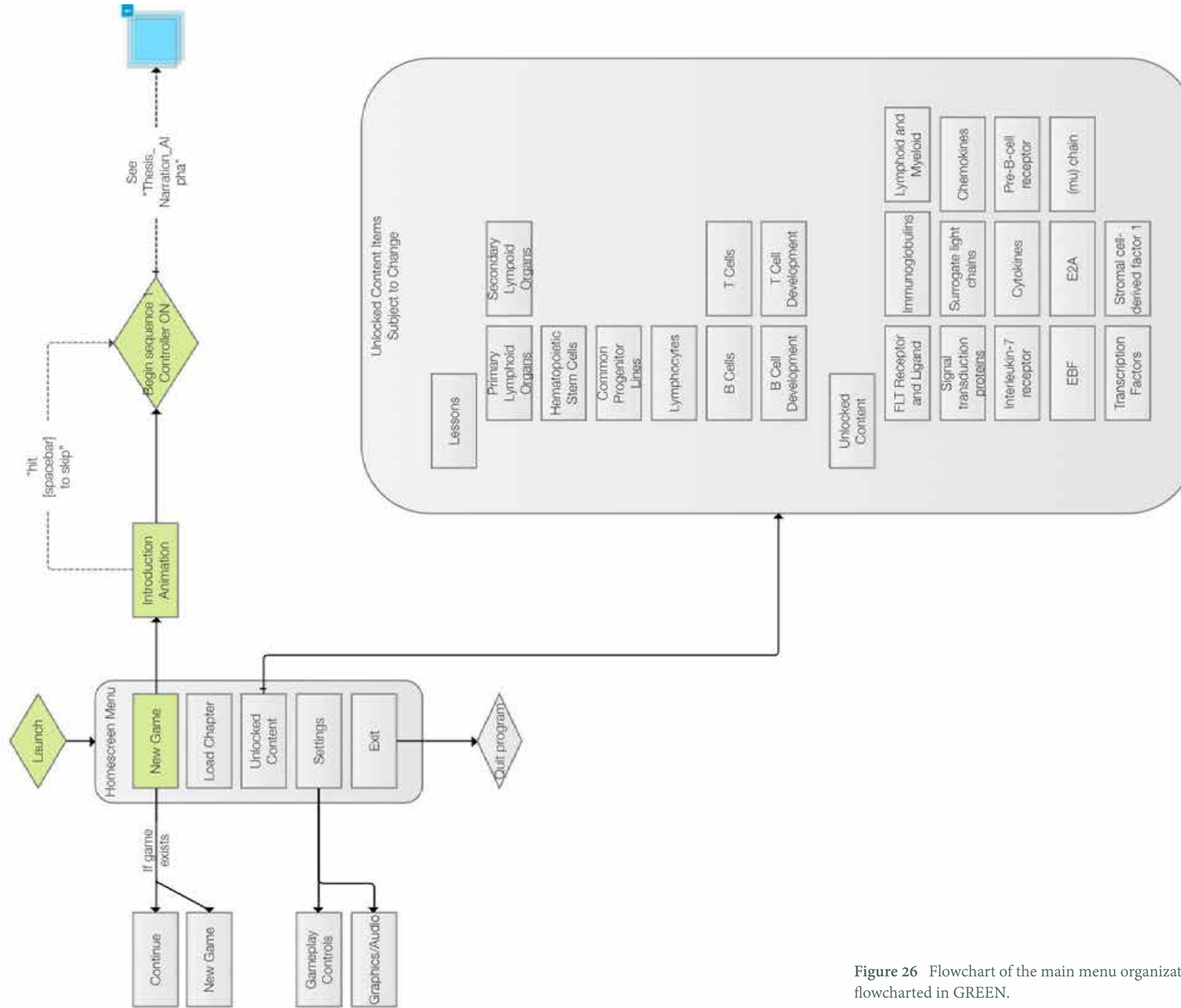


Figure 26 Flowchart of the main menu organization of the project. Main sequence of events is flowcharted in GREEN.



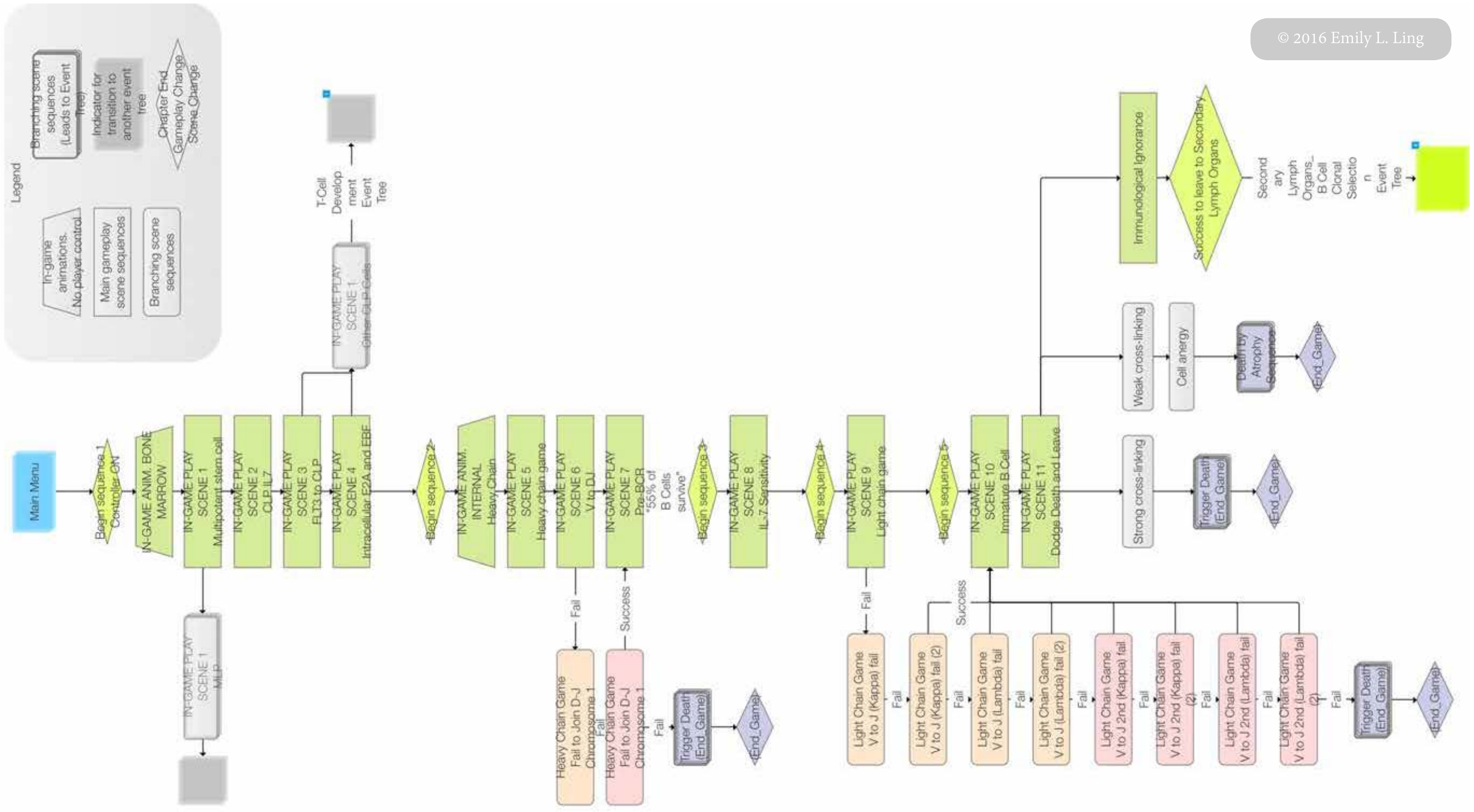


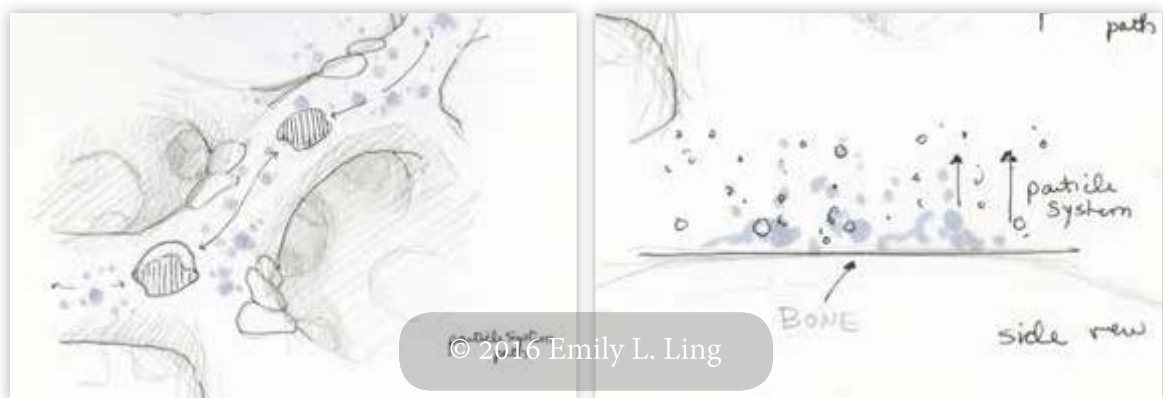
Figure 28 Flowchart of the proof of concept B lymphocyte branch of the project. Main sequence of events is flowcharted in GREEN.

# Gameplay Design

## Developing Main Gameplay Design

Main gameplay focused on allowing the player to explore the environment and trigger narrative events in the game. Inspired by *The Stanley Parable*, narrative sequences of gameplay were divided into scene sections, with each scene's narration describing the current actions of the player and the cell. The narrative script was designed to be placed along a desired path of exploration in order to trigger the narration in sequential order.

To encourage players to follow a correct path to trigger events, a particle system was created using Adobe Photoshop CC 2015 over a C4D render. Players would be instructed through the narrative content to follow the particles. These particles would be revealed in the main gameplay after being named and described as a chemokine, CXCL12, which in life guides developing lymphocytes to proper areas of the bone marrow.



**Figure 29** Conceptual designs of the CXCL12 particle system. Text in this image is not intended to be read.

A health gauge was a unique feature added to better engage player focus on moving through the environment. The environment was mocked up in C4D and the health gauge was created in Adobe Illustrator CC 2015 (Fig. 30).

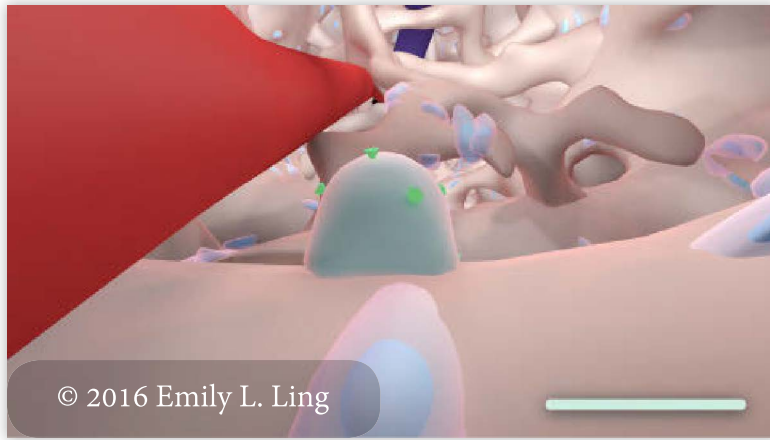


Figure 30 Concept render and mock up of health gauge feature.

### Developing Mini Game Gameplay Design

A mini game was conceptualized for the DNA rearrangement portion of the game. Borrowing from various off-the-shelf games, such as *Wolf Among Us* by Telltale Games (Fig. 31), DNA folding is controlled by the player until the orientation is correct, followed by a sequence of keystrokes to trigger a successful rearrangement. These concepts were rendered in C4D and graphic user interface (GUI) elements created in Adobe Illustrator (Fig. 32).



Figure 31 Telltale Games, *Wolf Among Us* in-game screen captures of real-time gameplay design feature.

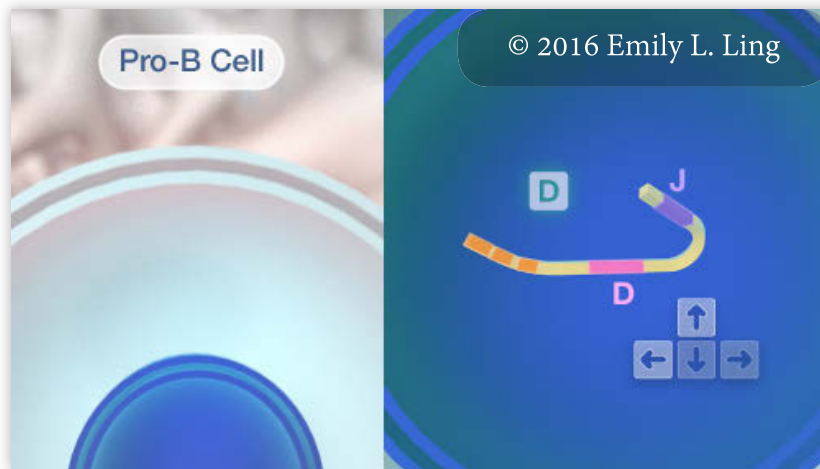


Figure 32 Concept render and mock up of mini-game gameplay design and on-screen prompts.

## Visual Development of the Game

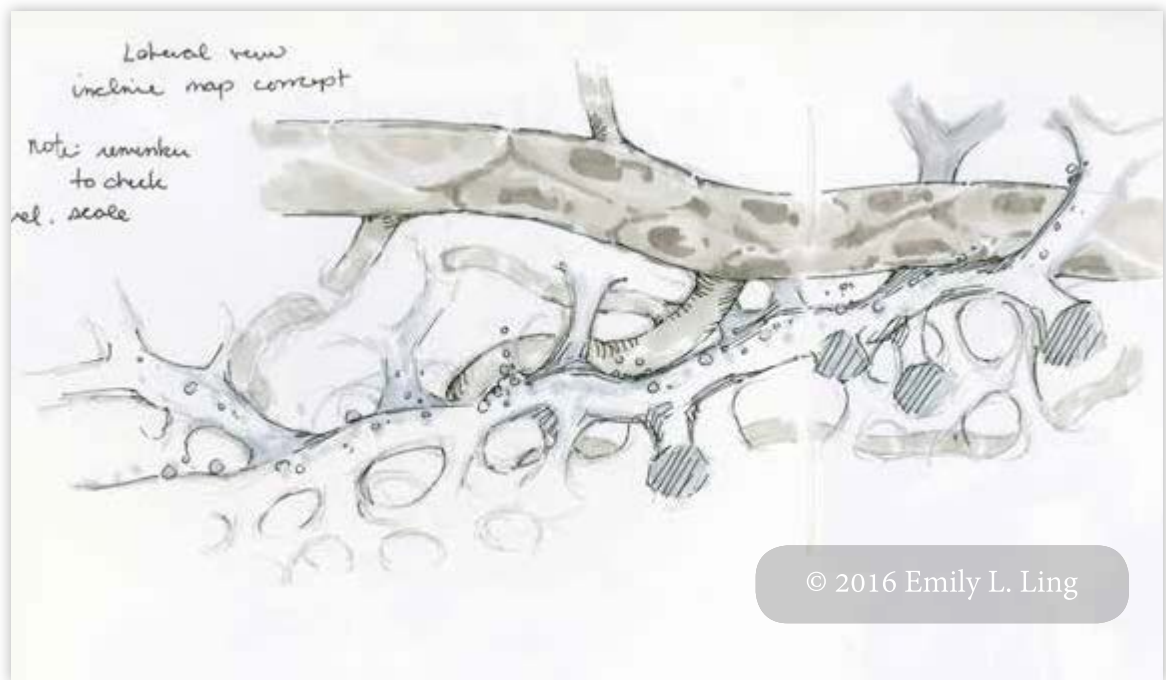
### Developing the Bone Marrow Microenvironment

Development and design of the bone marrow game level environment required research of the bone matrix and marrow anatomy as well as the microenvironmental organization of tissues within bone and bone marrow (Nagasawa 2006)

Player movement through the bone marrow was also conceptualized (Fig. 33). Janeway's Immunobiology describes developing B-lineage cells as initially making contact with reticular stromal cells in the trabecular spaces (Murphy et al. 2012). As the cells continue to mature, they eventually move towards the central sinus (Murphy et al. 2012) to join with blood flow leaving the bone marrow cavity (Nagasawa 2006).

Therefore, initial player interaction and navigation in the environment was designed to start at a lower point in the environment. As the player progresses through the game, the navigation would allow the player to move on paths higher in elevation and closer to the core, where the central sinus and nutrient artery would be situated (Fig. 33).





**Figure 33** Conceptual design of game environment and elevation change. Text in this image is not intended to be read.

### Developing the Characters

Main characters were developed based on the sequence of gameplay events outlined in the narrative script and flowchart developed for the B lymphocyte branch of the game (Fig 28). When designing characters, one of the important factors to consider is the silhouette design of a character. Silhouettes offer a way for a user to quickly distinguish the differences between forms and is another element that enhances a memorable character design.

However, the gameplay content focuses the player interaction with the environment from the point-of-view of a developing cell. In order to associate the player's interpretation of his or her role as a developing cell within the educational scope of the game, anthropomorphic characterizations were minimized. The reasoning was to create a more direct analogy of a cell's movements through the bone marrow microenvironment.

For example, in Immune Quest, macrophages are depicted as brutes that engulf bacteria (Fig. 34).

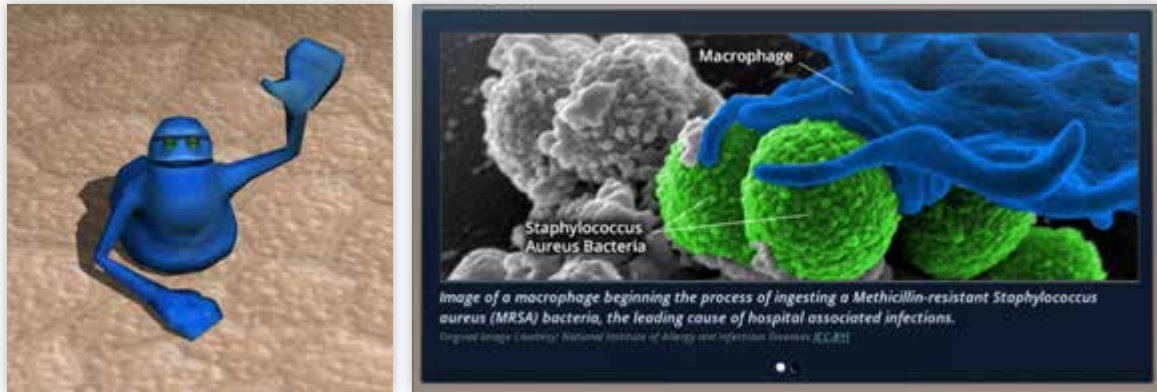


Figure 34 ImmuneQuest™ in-game anthropomorphic macrophage character (left image) and a scanning electron microscope image of a macrophage (right image).

An on-screen menu can be accessed for more information related to macrophages and their functions in the body. Colors are used to associate the character avatar with a scanning electron microscope image of a macrophage engulfing bacteria (Fig. 34).

While these analogies may be beneficial for comprehension of the material, it is the intention of this proposal to create new analogies through visualizations of cell movement and behavior more accurate to those described in the literature, recorded videos, and microscopy images.

However, this presented a problem in that most cells are spherical in form. Thus, quick differentiations between characters that could traditionally be done through silhouettes, in addition to other qualities such as color, were constrained. Furthermore, the game content focuses on the early stem cell differentiation to a mature B cell. Unlike neutrophils which have a unique 3-lobed nucleus, many of these stages lack visible surface or nuclear shapes that would distinguish them. Therefore, it was necessary to

develop characteristics to add personalities to cells that reflect their stage of development and interaction with the environment.

In addition to this, the Unity 5 3D game development engine does not support point level animation on meshes. Character animation and control features require that animated character models, or assets, be rigged with a joint system.

With these constraints in mind, incorporating aspects of anthropomorphic forms that viewers could relate to were developed and incorporated. Since a cell can be considered moving on its “belly”, similar to a crawling quadruped, front and back forms were added to give characters a visible indication of mobility and a method of animating interactions with the game environment.

### **Developing the HUD**

It was necessary to evaluate the importance of HUD elements the player has access to. As noted in the section, Problems with Gameplay of Currently Available Titles, if a HUD contains too much information, it can “distract the player from the environment in which he or she is immersed” (Wilson 2006).

Based on the outline of the gameplay features, the most consistent information players would be required to monitor involves their character’s exposure to survival signals. Similar to a health bar, this concept was mocked up in Adobe Illustrator and placed in the lower corner of the screen to avoid obscuring the gameplay field.

Necessary elements for the HUD and any additional on-screen GUIs for the mini game’s gameplay were identified to be: the current name of the character in its stage of

maturation, on-screen gameplay commands or real time actions, on-screen text labels of new models or segments of the DNA being rearranged, and narration subtitles.

### **Proof of Concept Storyboards**

Storyboards of the opening sequence were created to evaluate the composition and initial presentation of the player into the game environment. In addition, storyboards were developed for transition sequences between the introduction scene and the initial player interaction with a game asset (see APPENDICES: Storyboards).

Other storyboards and mock-ups were created as necessary using a combination of Adobe Photoshop, and Adobe Illustrator. Cinema4D was used to visualize organization of design elements, such as the main menu GUI elements or on-screen user interface (UI) menus.

### **Narration Recording**

Narration was recorded on a Zoom H4N Handy Portable Digital Recorder through the freeware program Audacity. Initial processing of the audio was achieved in Audacity, with subsequent adjustments and separation of narrated phrases into .mp3 clips in Adobe Audition for import and use in Unity 5 3D.

## Building the 3D Prototypes of Proof of Concept

### Modeling Software

Cinema 4D (C4D) version R16 by Maxon and ZBrush version 4R7 by Pixologic were used for modeling the 3D game prototype assets.

### Modeling the Multipotent Stem Cell

Tests of the Multipotent Stem Cell character were first modeled in C4D R16. The initial goal was to animate the character animation cycles, for import into the Unity 5 3D game engine, using C4D Deformers, Splines, Pose Morphs, or other Primitives. However, these were abandoned since Unity 5 3D does not support point level vertex animation. Animation of the character was then developed through a joint system. Using a bound joint skeleton to a model's mesh allowed the asset to be animated with the native Mecanim features in Unity 5 3D for future development (Fig. 35).

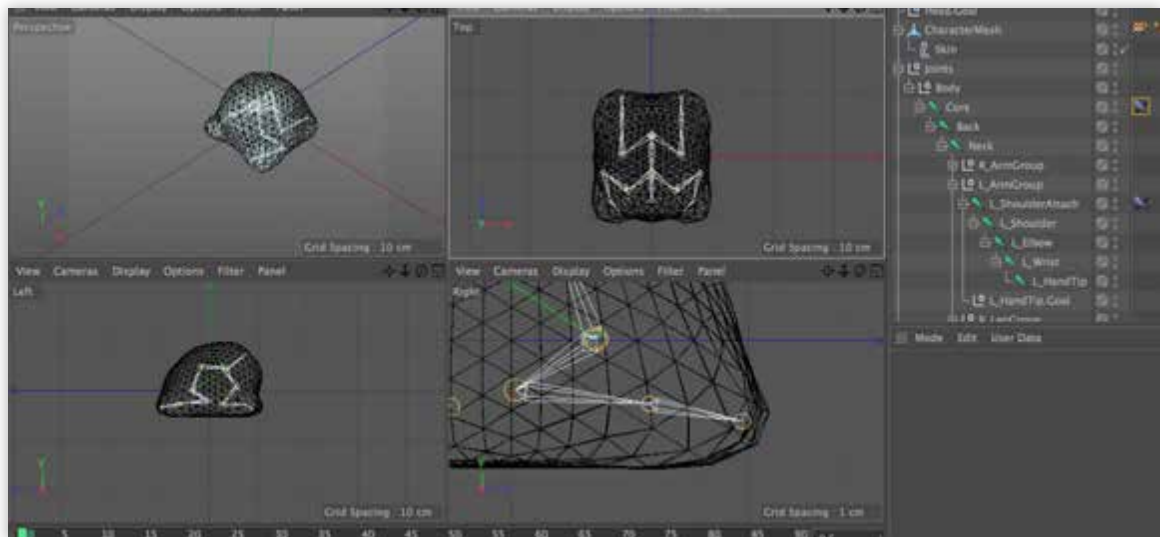


Figure 35 C4D Multipotent Stem Cell character with Joints. Text in this image is not intended to be read.

Since the character design lacks the usual features and anticipated movements of a humanoid character, the joint system had to be designed to fit the Multipotent Stem Cell. Joints were drawn using the Joint Tool under the Character menu (Fig. 36). The Joints placed into the model needed to encompass the character's mesh, while providing enough Joint regions for desired movement qualities such as stretching and crawling.

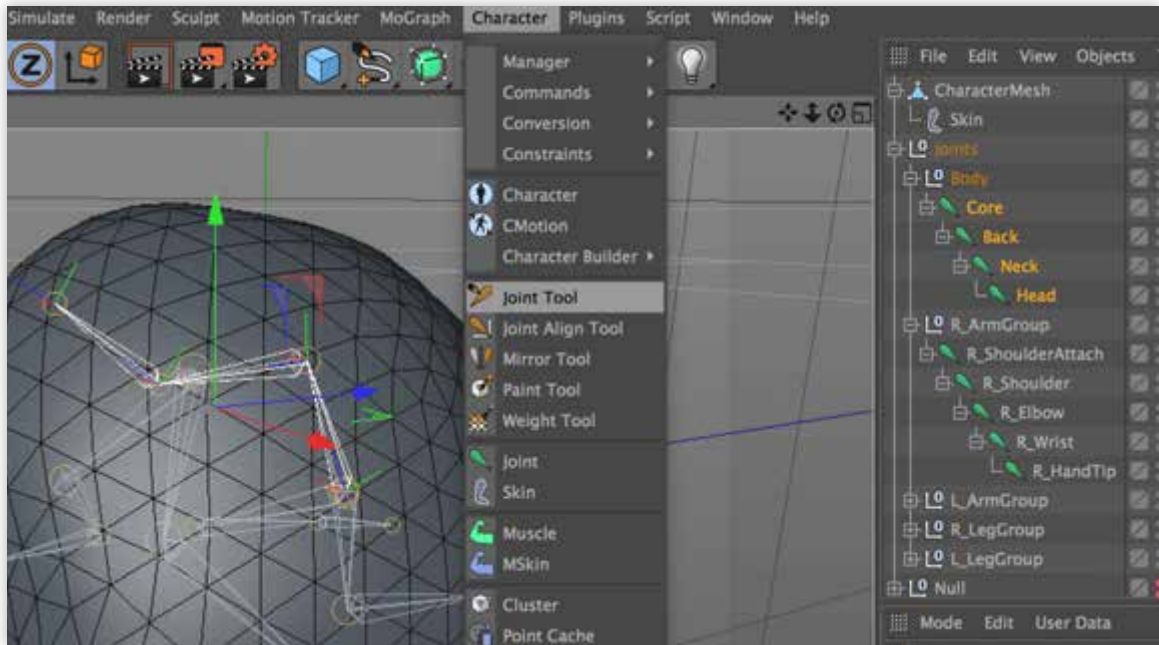


Figure 36 C4D Joint Tool location.

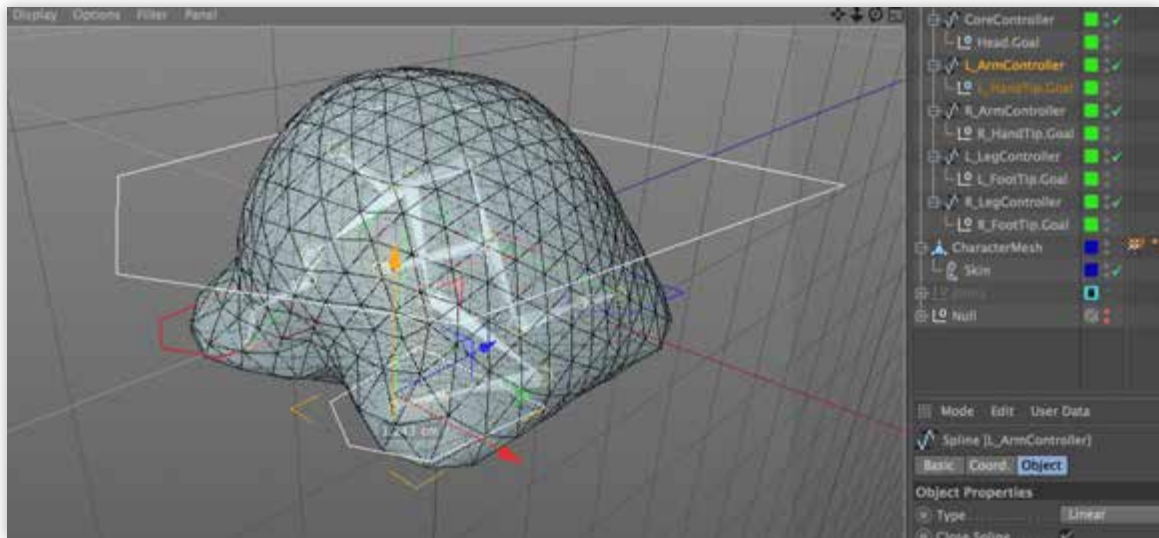


Figure 37 Controller parenting to grouped Joints in C4D. Text in this image is not intended to be read.

Spline tools were used to draw controllers. These controllers were then added to limb-like Joint bound regions of the model, as well as the main body of the character, to enable easier key framing and animation movement. This was done by grouping the Joints and childing them under the desired parent controller Spline (Fig. 37). Weighting of the bound character mesh and Joints were also refined to minimize mesh pinching during animation or movement (Fig. 38).

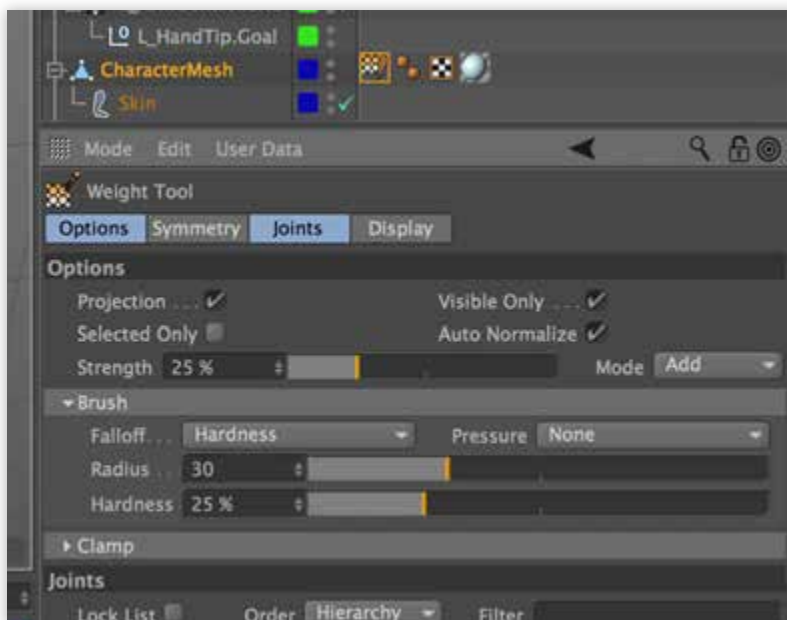


Figure 38 C4D mesh weighting adjustments using the Weighting Tool.

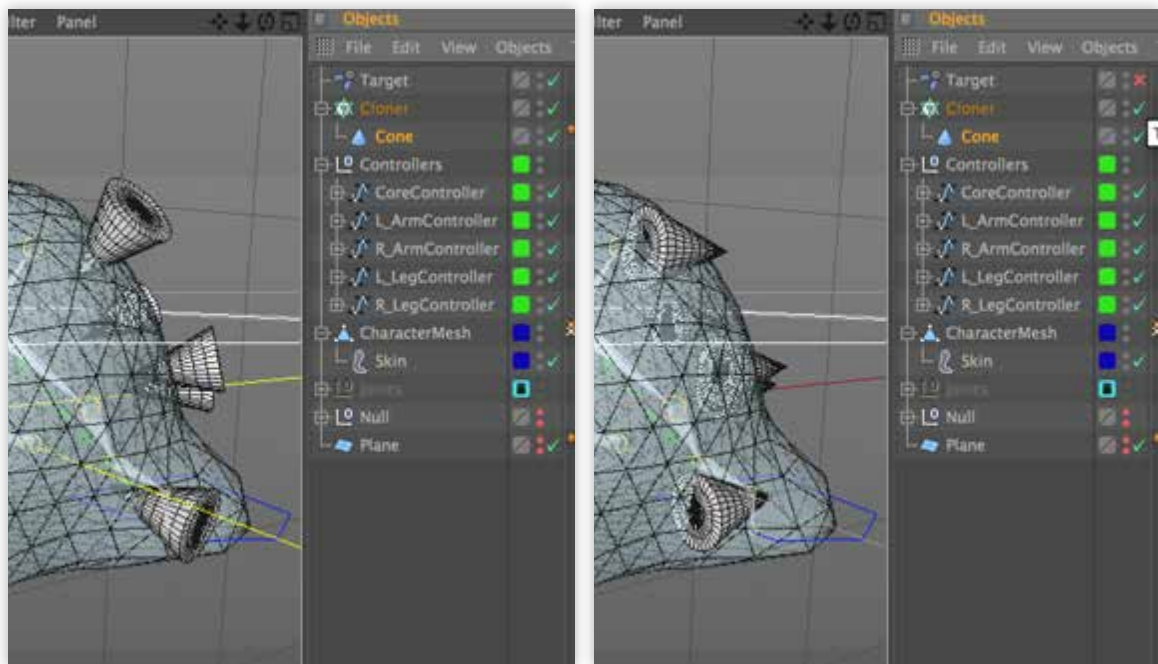


Figure 39 Surface receptor alignment with a Target Effector in C4D (left image) in comparison surface receptor to alignment without a Target Effector (right image).

Surface receptor objects were added to the surface of the main character model using a MoGraph Cloner and setting the Mode to Object, and Distribution to Surface. The rotation of the cloned object was further refined by adjusting the Transform Rotation parameters in the MoGraph Cloner.

To ensure the cloned objects were orientated properly on the surface, an additional MoGraph Target Effector was added to the cloner (Fig. 39). The Effector's Target Mode was set to Object Target, and the Target Object was set as the character model.

### Modeling the Bone Marrow Environment

The game environment was conceptualized as a cylindrical world. Similar to the organization of long bones and red bone marrow, the density of trabecular spaces would decrease towards the center of the bone. As the bone marrow extends from the center



outwards, the spongy bone would transition into compact bone. Thus the design needed a dense trabecular like wall with protruding trabeculae extending into a hollow and navigable center.

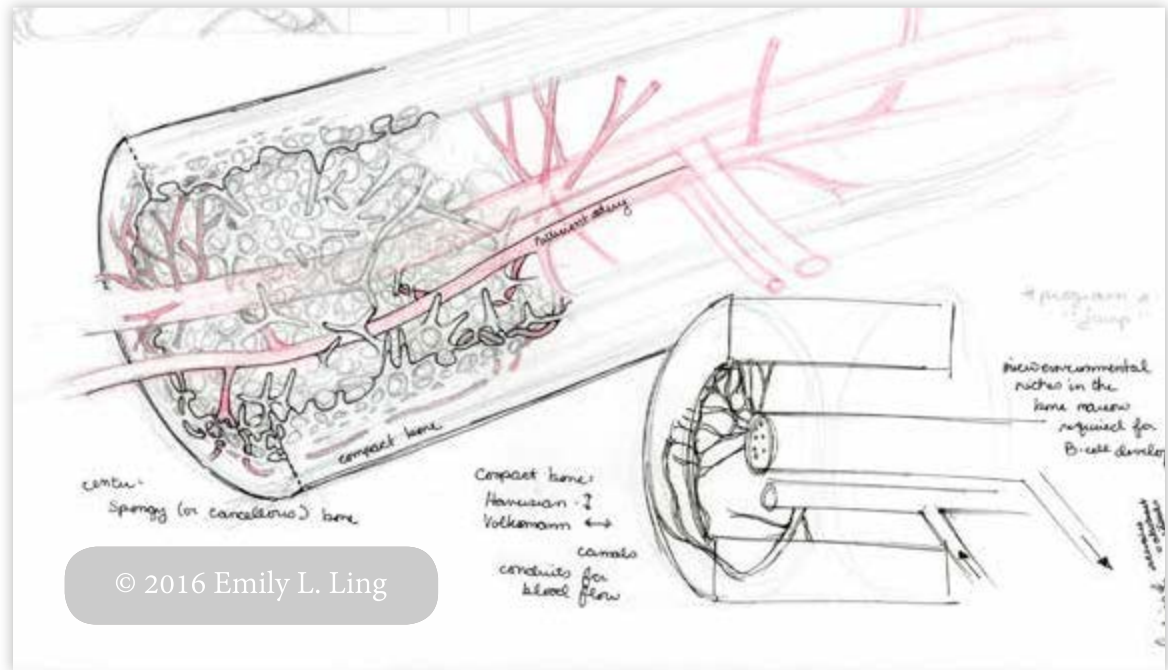


Figure 40 Compact bone and bone marrow anatomy concept designs. Text in this image is not intended to be read.

To achieve this design in a 3D model, the free plugin Proc3Durale v 0.41 beta was used to create the base bone marrow world. The plugin creates negative space cut away from a black and white texture channel applied to a Source Object in C4D. The plugin works on Parametric as well as Editable objects.

A base Parametric Capsule was used as the Source Object for the plugin. However, if no thickness was applied to the Source Object, the plugin treats the Source Object as a solid form and created the negative spaces throughout (Fig. 41).

The best results for a bone marrow environment while preserving a hollowed center utilized Simulate Cloth Surface on a base Parametric Capsule. Also, when Source Objects

were used with Pro3Durale, the plugin created a duplicate of the Source Object (Fig. 41). Therefore, it was best to toggle the visibility of the Source Object to minimize cluttering

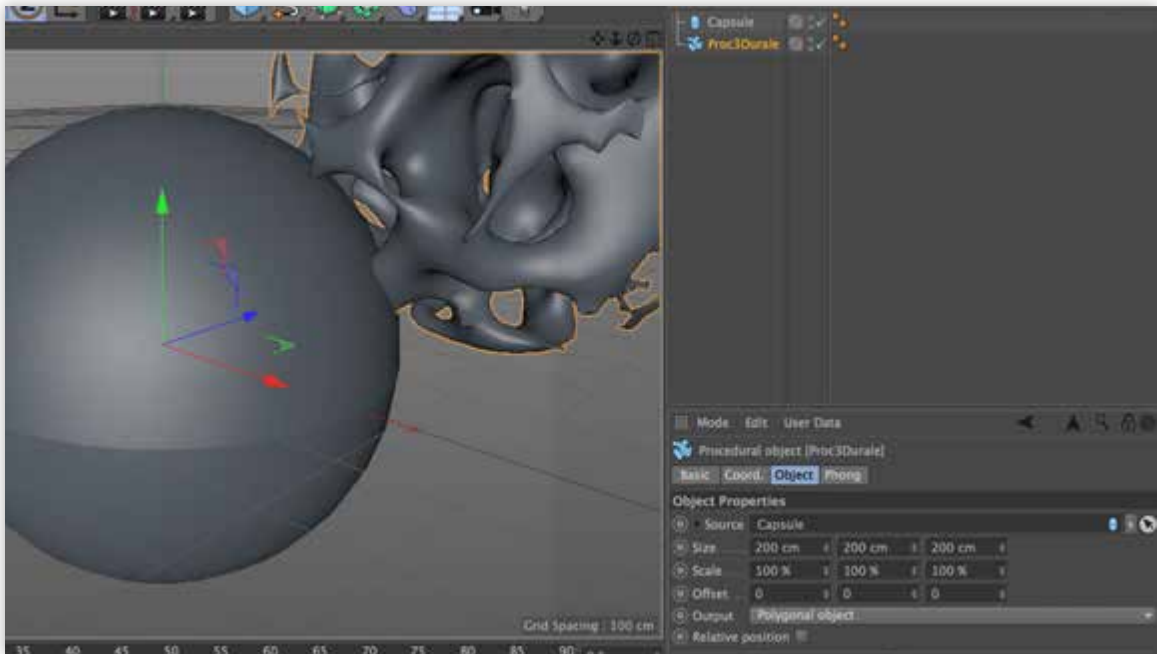


Figure 41 C4D Pro3Durale plugin use on a Parametric object without Simulate Cloth Surface creates negative spaces throughout the object's volume.

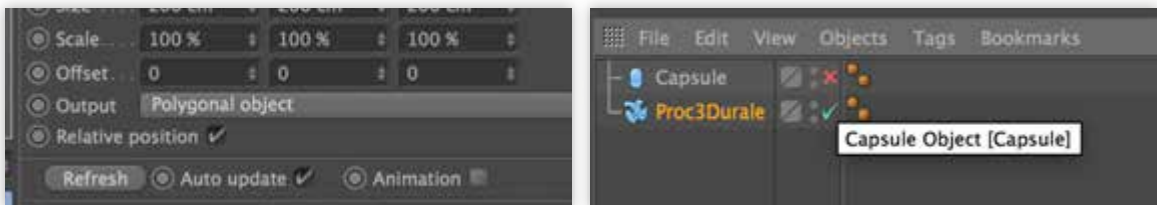


Figure 42 Relative Position (left image) was toggled off. Source object was also toggled off (right image).

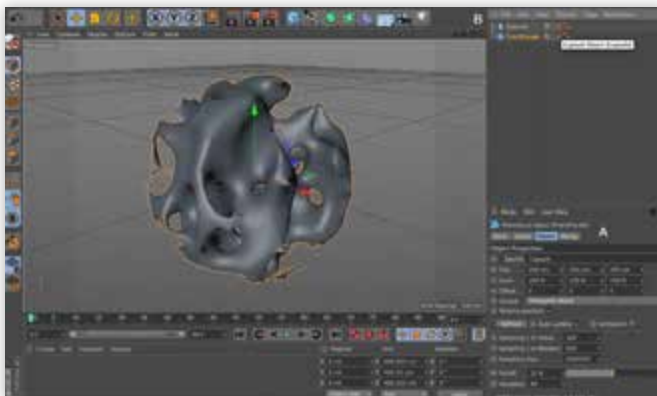


Figure 43 C4D viewport of the Proc3Durale object in the World Coordinate Position 0x, 0y, and 0z.

(Fig. 42) the viewport. The checkbox “Relative Position”, in the plugin’s Object Properties (Fig. 42) was toggled on to preserve the newly created object’s Coordinate Position at the World Coordinate Position of 0 x, 0 y, and 0 z (Fig. 43).

Once the plugin settings for the bone marrow environment were adjusted, the object was converted to an Polygonal Object using “Current State to Object”.

### Modeling Central Bone Trabeculae in ZBrush



Figure 44 ZSphere use for modeling individual trabeculae.



Figure 45 Adaptive skin (left image) was used to convert the ZSphere into a meshed surface (right image) for further modeling.

Trabeculae was modeled on the bone marrow environment using appended ZSpheres (Fig. 44). Once the general shapes and positioning of the ZSpheres were drawn, each ZSphere trabeculae was turned into a meshed surface using Adaptive Skin (Fig. 45). Further modeling was done to smooth out the generated ZSphere surfaces.

### **Modeling Vessels in C4D**

Once the trabeculae was modeled onto the bone marrow environment in ZBrush, the ZBrush Subtools were exported as individual Wavefront OBJ Files. It is important to note that Subtools exported with ZBrush Polygroups retain their separation as Children Objects in C4D. Once imported into C4D, the Polygroup Children Objects were collected under a C4D Parent Group.

Using the imported OBJs of the bone marrow environment and the central bone trabeculae from ZBrush, splines were drawn and fitted into open spaces through the model. Vessels were mimicked by using MoGraph Sweep to create a cylindrical surface along the drawn spline path. Under the Sweep Object Caps settings, Start and End Caps were set to “None”. Next, Simulate Cloth Surface was used on the MoGraph swept object to add thickness to the vessel model wall. Isoparm Subdivisions were increased to minimize awkwardly angular curves.

This process was used for both the artery and vein objects, with the vein being modeled larger in diameter and thinner in wall thickness.

## Optimizing Vertex Count and Surface Meshes in ZBrush

Model surface vertex counts were reduced in ZBrush to optimize rendering in Unity 5 3D as well as distribute and smooth areas of unusual surface geometry (Fig. 46). First, DynaMesh was used on each Subtool to create evenly distributed quad topology over each model. The settings for DynaMesh polycounts were adjusted to retain a degree of angular surface topology without compromising detailed shapes.

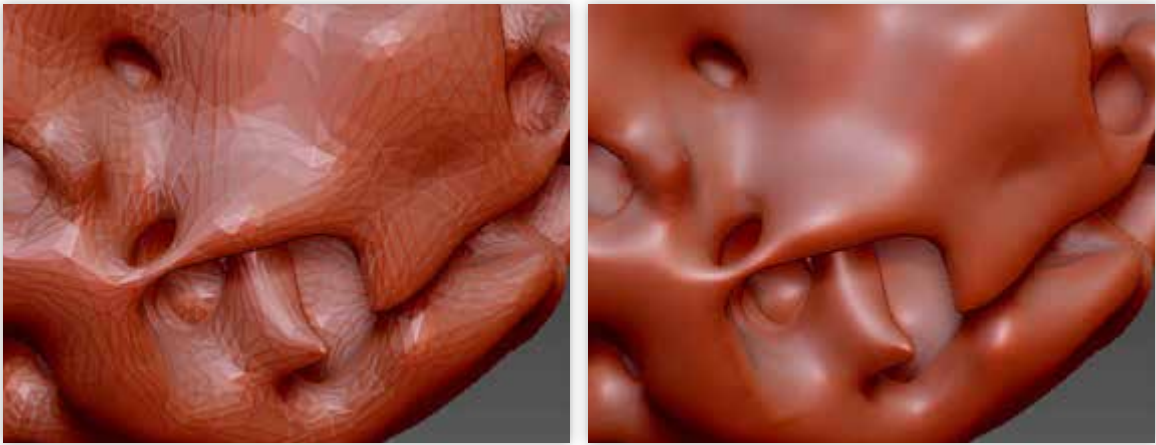


Figure 46 ZBrush surface difference before (left image) and after (right image) use of DynaMesh.

Once the models were DynaMeshed, Zplugin Decimation Master was used to further reduce polycounts and create a lower resolution copy of decimated Subtools. The objective was to create models with polycounts below 100k, or 60k when possible, since

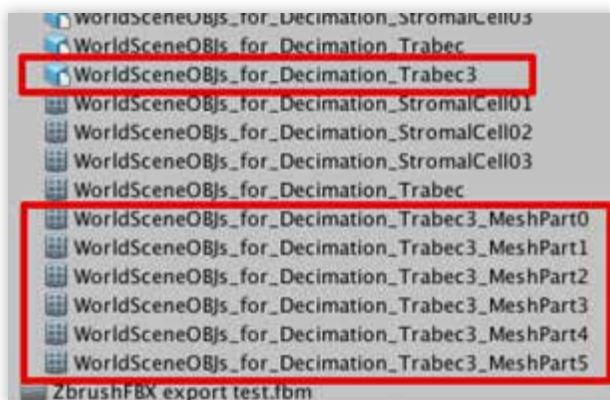


Figure 47 Unity 5 3D mesh splitting of a model into multiple sub-65k vertex count parts.

Unity 5 3D splits an imported object's mesh if the vertex count is greater than 65k (Fig. 47).

To bring back the details lost during decimation, the Project function, which can be found in the Tools palette as a subcategory of Subtools, was used to project the higher resolution model's details to the low poly versions of models.

### Creating UV Textures for Models in ZBrush

UV textures were created for models in ZBrush using the UV Master function found under Zplugins.

### Reimporting into C4D and Exporting as FBX for use in Unity 5 3D

Once models were optimized in ZBrush, Subtools were exported as OBJs for reimport into C4D. When imported into C4D, the Subtools, now called objects, retain their relative sizes and relationships to each other. From this file, objects were renamed and grouped into nulls to organize them for export into FBX files for use in Unity 5 3D.

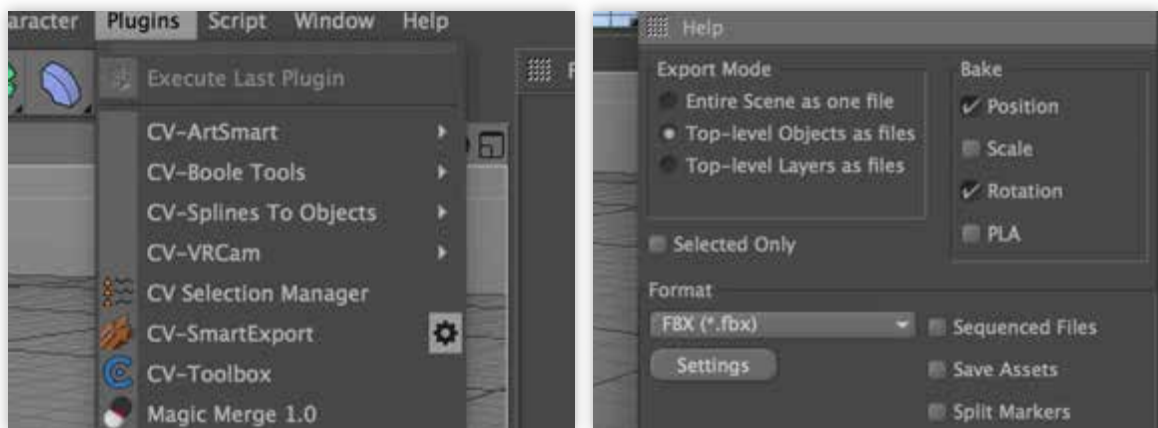


Figure 48 C4D CV-SmartExport batch exporter settings.

Since C4D does not support separation of models into multiple FBX files when exporting all assets in a scene, the Cineversity Pro plugin CV-SmartExport for batch exporting was used.

“Top-level Objects as files” option was selected for export (Fig. 48). Export preferences were adjusted under the plugin window’s settings. Lights, Cameras, and Splines were also unchecked since Lights and Splines are not supported in the carry over into Unity 5 3D. Cameras were unnecessary since Unity 5 3D has native cameras. Also, “Textures and Materials” as well as “Embed Textures” options were checked to embed the file settings into the exported FBX files for Unity 5 3D to automatically extract as assets during import.

## **Unity 5 3D Game Development Platform Software: Assembling Components**

### **Free Edition of Unity 5 3D’s Game Engine**

Unity 5 3D is a game development platform and engine that is available in both a free and professional edition. Initial prototyping of the proof of concept was developed in Unity 5 3D free version 3.1f1. One limitation of the free edition is that it does not support movie textures, therefore, cutscenes must be animated within the game engine since the free edition does not currently allow for the import and use of video files.

Prototyped scripts were created using C# (pronounced “C sharp”) and Unity 5 3D’s native coding software MonoDevelop version 5.9.6. C# was chosen as the code language since C# offered more flexibility for developing potentially complex custom operations.

Scripts developed for the initial prototypes were developed using the online Unity Manual (<http://docs.unity3d.com/Manual/index.html>) documentation, Gibson's *Introduction to Game Design, Prototyping, and Development* (2015), and various online tutorials on Lynda.com and YouTube.

### **Creating the Main Menu and Exit Screens in Unity 5 3D**

The Unity 5 3D's online User Interface Tutorials (<https://unity3d.com/learn/tutorials/topics/user-interface-ui>) were utilized for creating the prototype main menu and exit screens. Menu assets were created in Adobe Illustrator, exported as PNG-24 images, and imported for use into Unity 5 3D as sprite (2D and UI) textures. Converting the image assets into sprite (2D and UI) textures allowed the images to be used for Sprite Swap GUI animated transitions. Each menu option had 4 states: the default state called the target graphic, highlighted sprite, pressed sprite, and disabled sprite.

An exit menu was created using Unity 5 3D's native GUI options. A script was created in C# to test activation of the quit menu, which was designed as an overlaid on-screen prompt when activated.

### **Custom Particle Effect Mesh For Main Menu Screen**

Custom Particle System effects were designed as part of the main menu's interactive feature. The default Mesh particle shape was altered by changing the Particle System's Renderer's Render Mode to Mesh and choosing another mesh shape from the project assets imported into Unity 5 3D (Fig. 49). Script was written in C# to allow for the



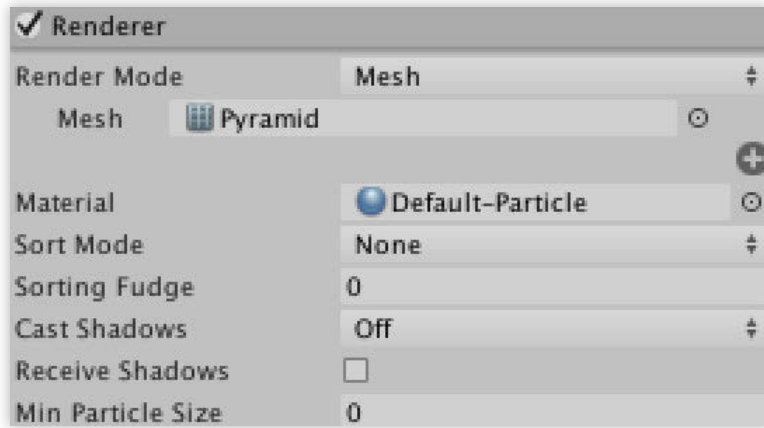


Figure 49 Unity 5 3D Particle System Renderer options

particle systems to turn on when the game cursor is detected over buttons created for the main menu options (see APPENDICES: C# Scripts).

### Setting Up Multiple Scenes

New scene assets were created and added to the project's asset folder. A script was created in C# to allow for switching between scenes when defined parameters were met (see APPENDICES: C# Scripts). The script utilized Build Settings hierarchical numbers assigned to each scene compiled into a singular project (Fig. 51).

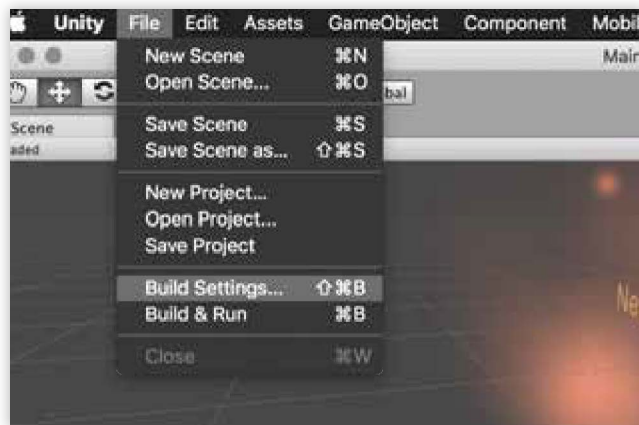


Figure 50 Unity 5 3D's Build Settings menu.

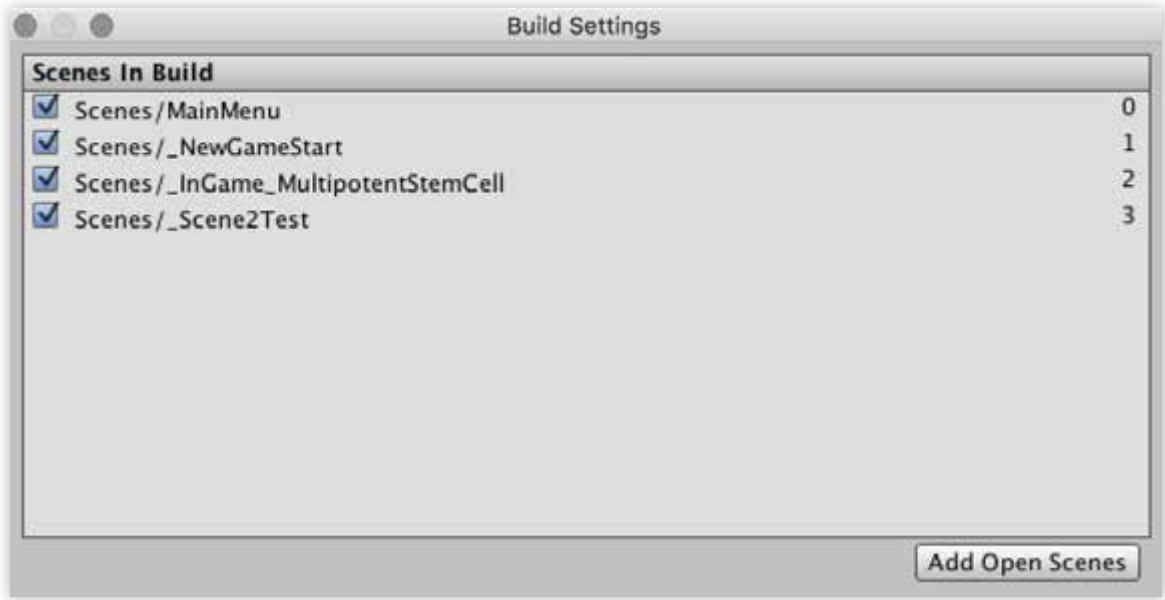


Figure 51 Unity 5 3D's Build Settings and added scene hierarchy numbers.

An additional script was created for the scene following the main menu as a placeholder for an explanatory in-game cutscene sequence for players to learn the premise of the game narrative. The script coded for switching between the cutscene to the first gameplay scene when a down key input, such as the “space bar” on the keyboard, is pressed.

### Atmospheric Rendering Effects

Initial tests of atmospheric fog was simulated by adding particles system effects in Unity 5 3D. This was tested since Unity 5 3D free edition does not, currently, allow for the procedural generation of atmospheric fog in game environments.

## RESULTS

Available titles were identified and evaluated for content focus and problematic gameplay design. Designs incorporating features of gameplay used in commercially available video games were created as a solution to achieving an engaging educational video game. Models and interface mock-ups of gameplay features were created based on the considerations for the game content, B lymphocyte development, and improvements to educational gameplay design.

Due to time constraints, development of model assets were prioritized based on the sequence of events outlined in both the B lymphocyte development flowchart and the narrative script. These assets were imported into Unity 5 3D to begin creating and testing a functional prototype of the proof of concept.

## Models Created from Concept Designs

### Multipotent Stem Cell

Visual character development sketches were produced based on the constraints imposed by the use of character silhouettes (see APPENDICES: Visual Development Sketches). From the sketches, a 3D model was produced in C4D with joint rigging for import to Unity 5 3D (Fig. 52).

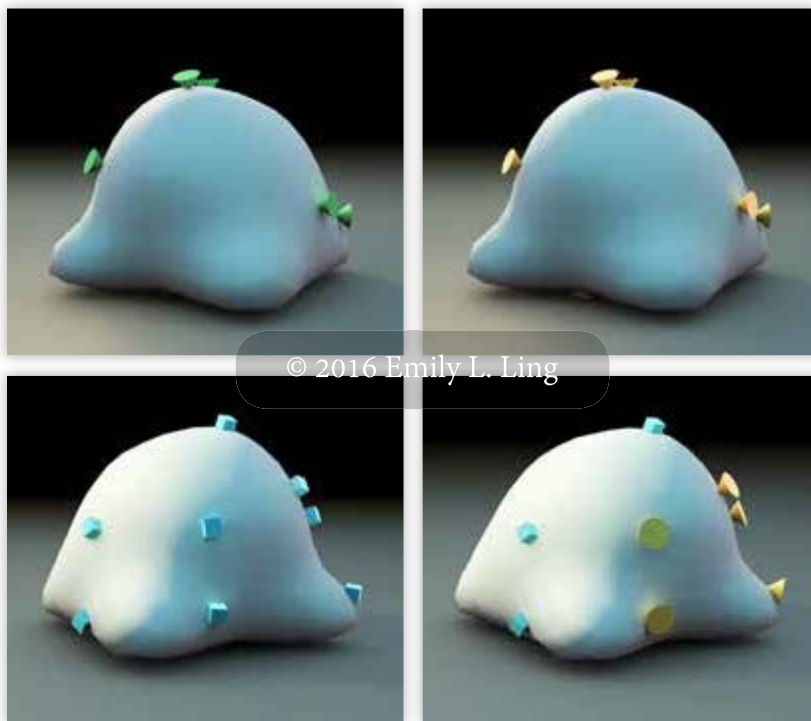


Figure 52 C4D test renders of the Multipotent Stem Cell character with various surface receptor variations.

### Bone Marrow Environment

Sketches were created based on images and research collected on the microenvironment and organization of compact bone and red bone marrow. Based on these early concept sketches, a cylindrical environment representing bone marrow was

modeled in C4D and optimized for import into Unity 5 3D using ZBrush (Fig. 53, Fig. 54).

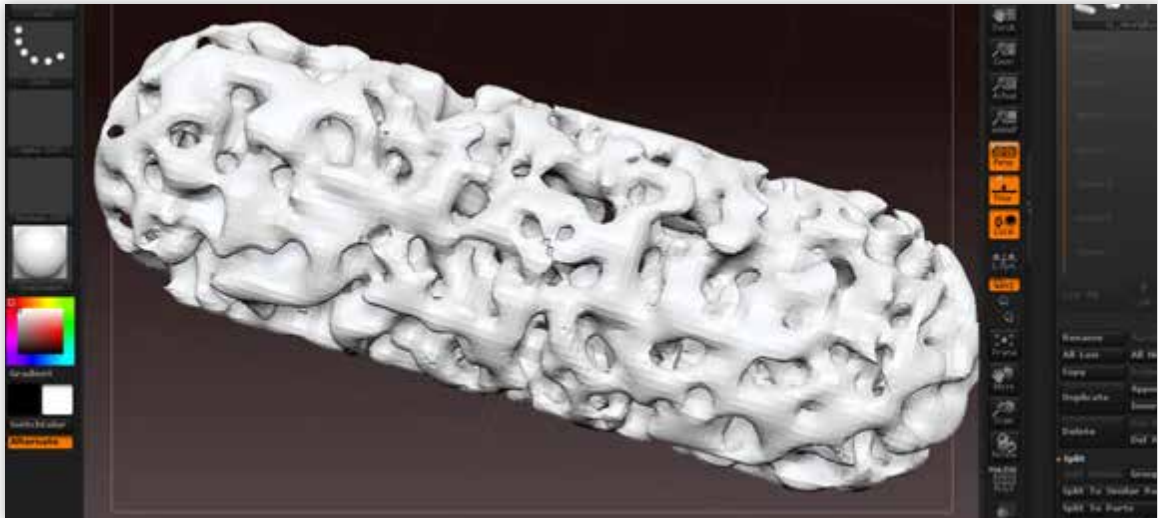


Figure 53 ZBrush optimized bone marrow environment model. Text in this image is not intended to be read.

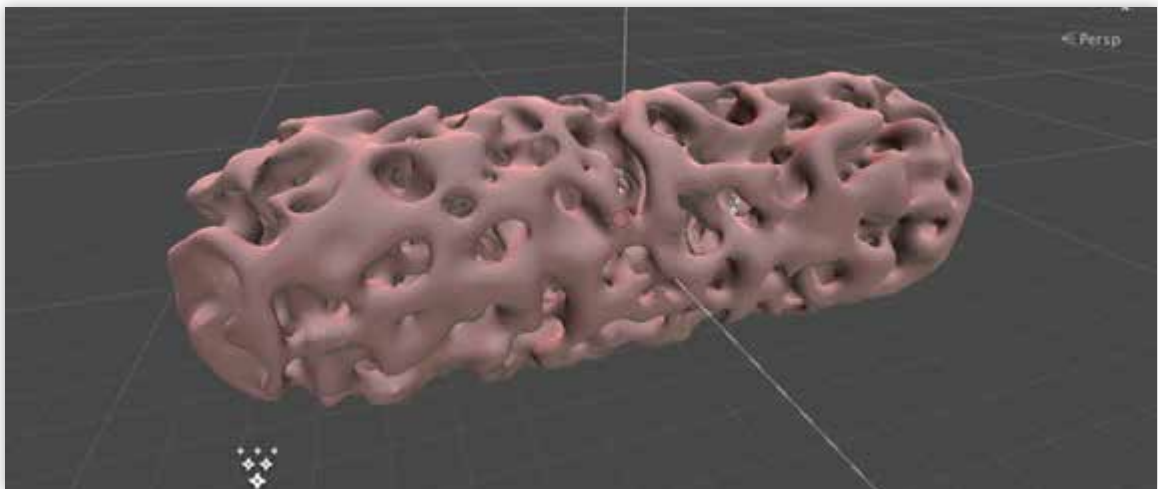


Figure 54 Bone marrow model imported into Unity 5 3D. Text in this image is not intended to be read.

### Blood Vessels and Stromal Cells

Blood vessels, including the central sinus vein and nutrient artery, as well as the stromal cells were modeled in C4D and optimized in ZBrush. These models were then

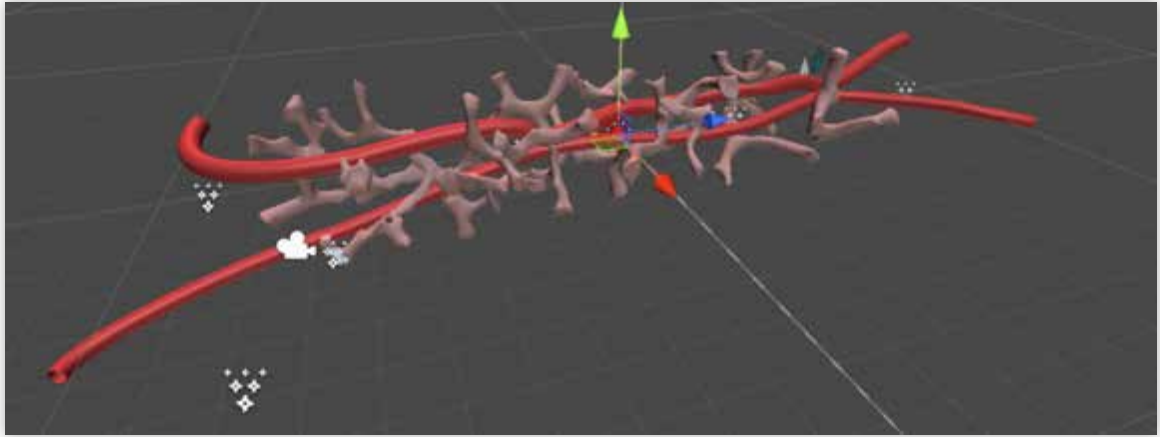


Figure 55 Optimized vessel models and trabeculae imported into Unity 5 3D.

exported from C4D as FBX files for use in Unity 5 3D (Fig. 55).

## Still Images for Interface and Gameplay Mockups

Storyboards were created for the introductory cutscene as well as for sequences involving gameplay interactions (see APPENDICES: Storyboards).

### Main Game and Mini Game Mockups

Using the designed storyboard drafts, mock-ups were developed for proof of concept examples of both the main gameplay as well as mini game features. HUD display elements and on-screen prompts were designed and created as vector elements as part of the mock-ups as well as for prototyping use in Unity 5 3D.

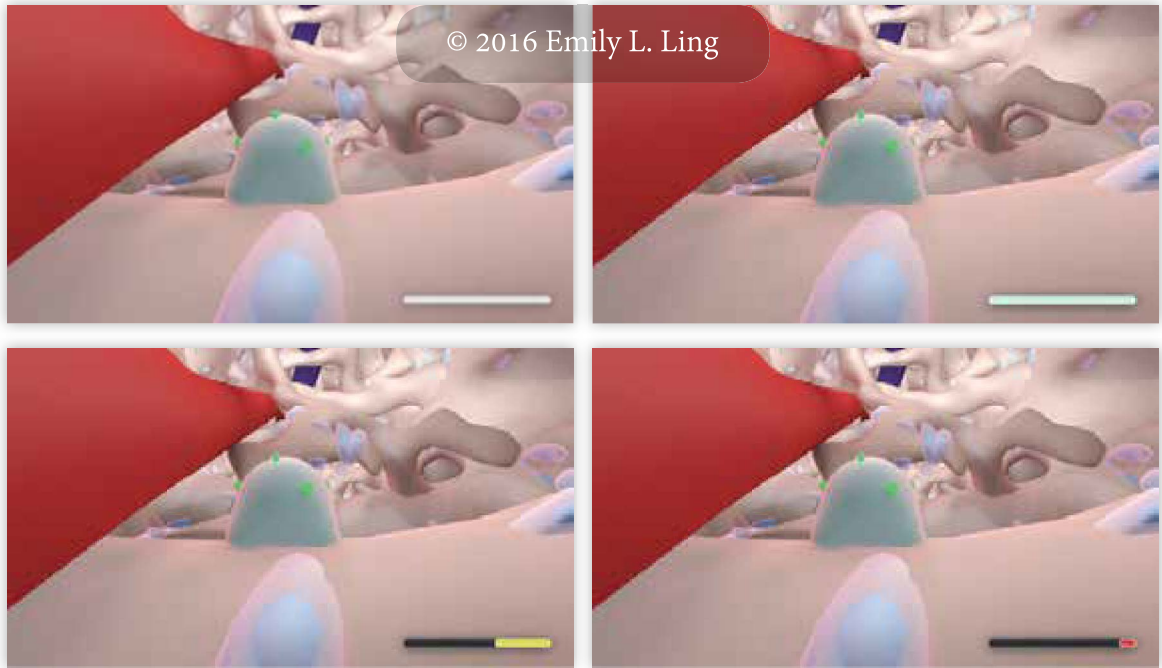


Figure 56 Example mock up of main gameplay's health gauge system.

## Prototyped Scenes in Unity 5 3D

Assets were imported into Unity 5 3D to prototype the proof of concept within the game engine. The following scenes were organized and scripts were created and tested for achieving the outlined gameplay design.

### Main Menu and Exit Interfaces

Main menu GUI assets were created and exported as PNG-24s for use in the Unity 5 3D main menu prototype. The main menu's base scripts for achieving interactive visibility of particle systems was achieved (Fig. 57). C# script for loading of a specified Unity 5 3D scene when the menu item "New Game" button is selected was successfully created.

In addition, script for triggering the on-screen Exit Interface by clicking on the “Exit” button, or by pressing the keyboard down stroke for “esc”, was also achieved (Fig. 58).



Figure 57 Unity 5 3D in-game demo screen capture of the prototyped main menu.

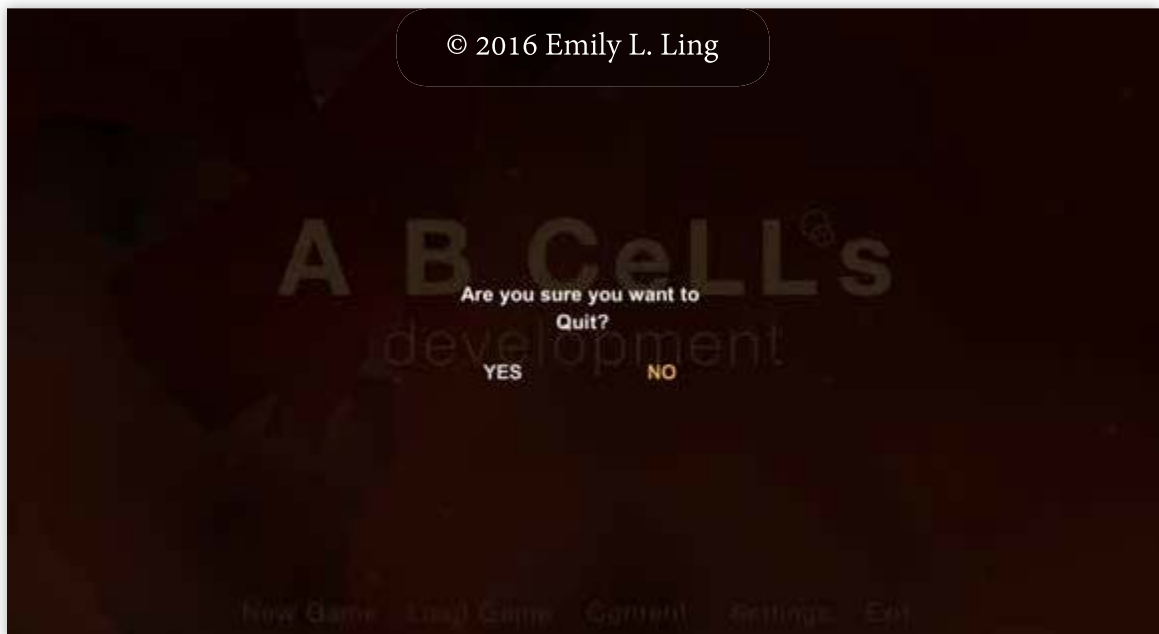
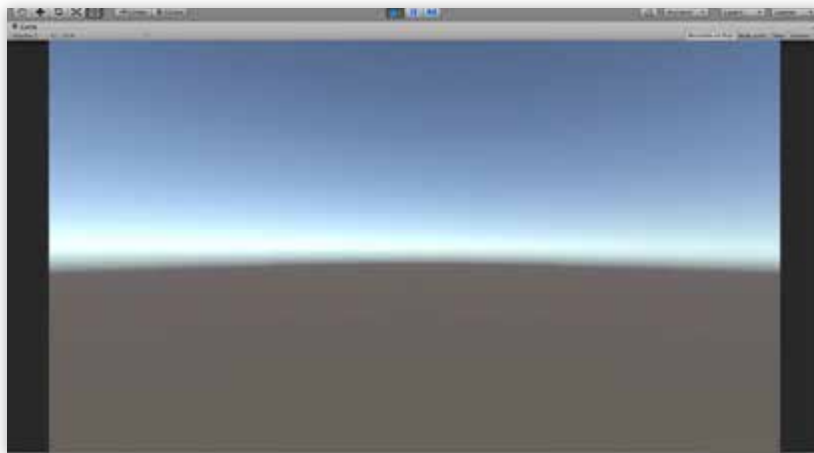


Figure 58 Unity 5 3D in-game demo screen capture of the prototyped exit menu.

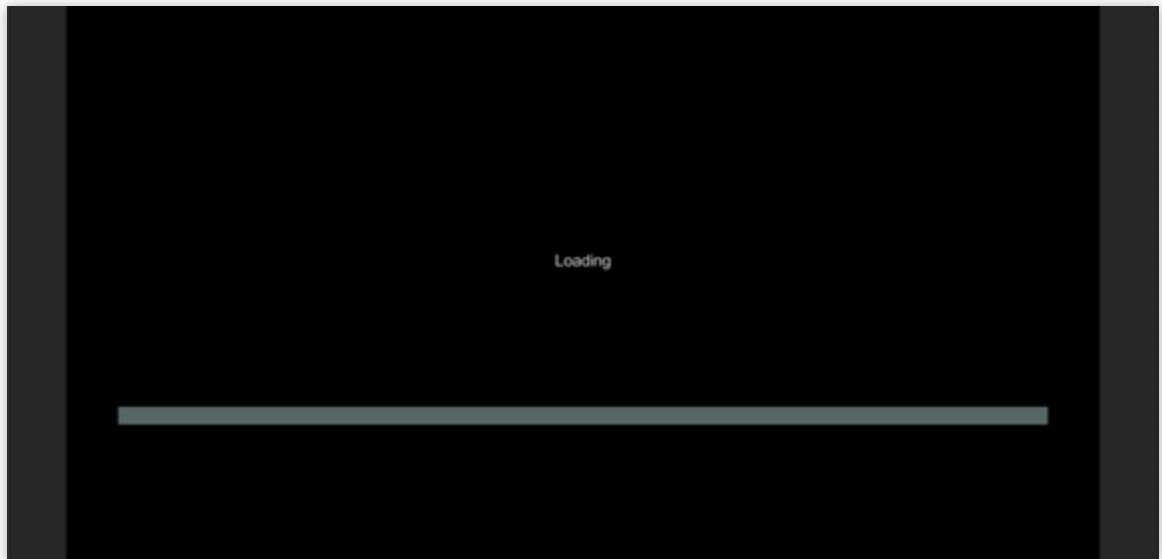


## Setup for Cutscene Sequence

When cutscenes are loaded it is customary for games to allow for users to skip introductory gameplay dialogues. A script was successfully created for triggering the loading of the next game scene after the keyboard hotkey “space bar” is pressed (Fig. 59). In addition, the script also accounted for loading time, incorporating a load screen transition if the subsequent scene does not load immediately (Fig. 60).



**Figure 59** Unity 5 3D in-game demo screen capture of a scene stage for subsequent addition of a cutscene.



**Figure 60** Unity 5 3D in-game demo screen capture of the loading screen after “space bar” key down is triggered on the keyboard.

## Scene 1 Gameplay Stage - Multipotent Stem Cell

Setup of modeled assets for the first gameplay stage focusing on the Multipotent Stem Cell character was organized in Unity 5 3D (Fig. 61). A placeholder sphere model was created in Unity 5 3D for the character model (Fig. 63).

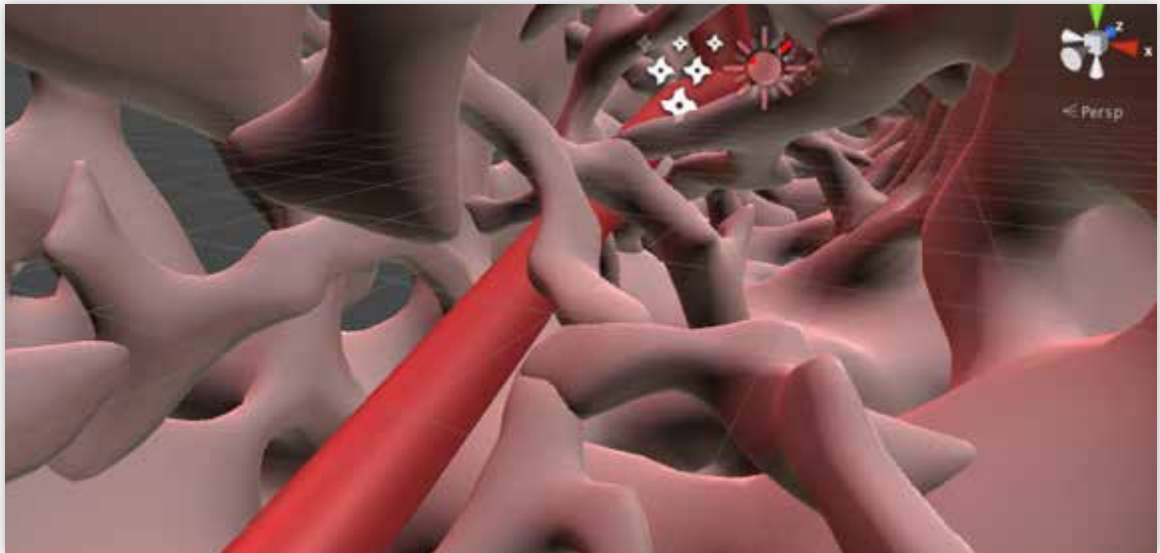


Figure 61 Unity 5 3D prototype build of Scene 1. Text in this image is not intended to be read.

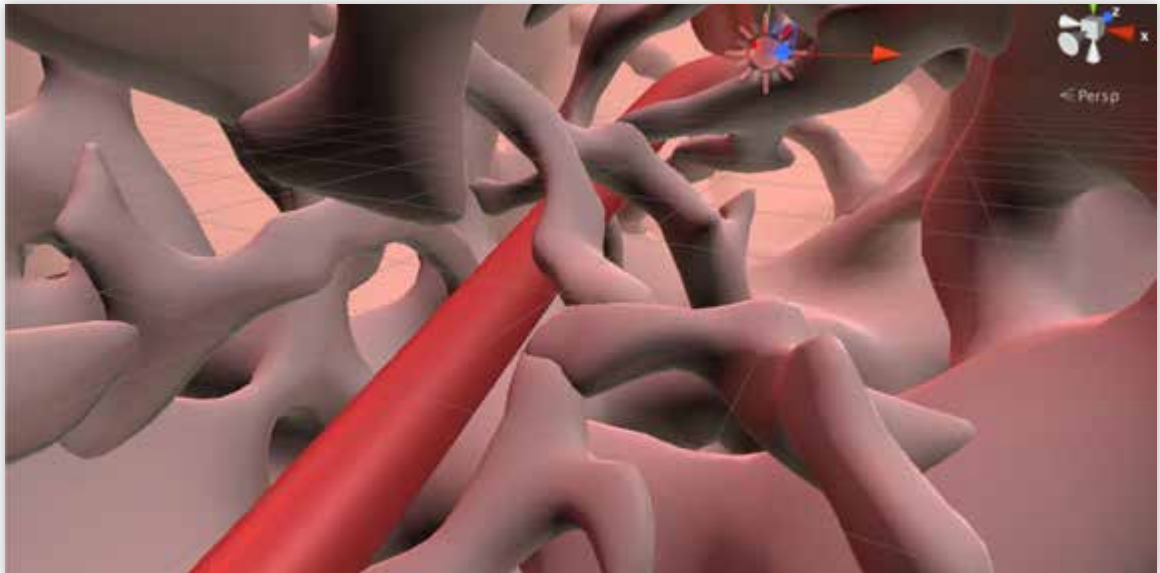
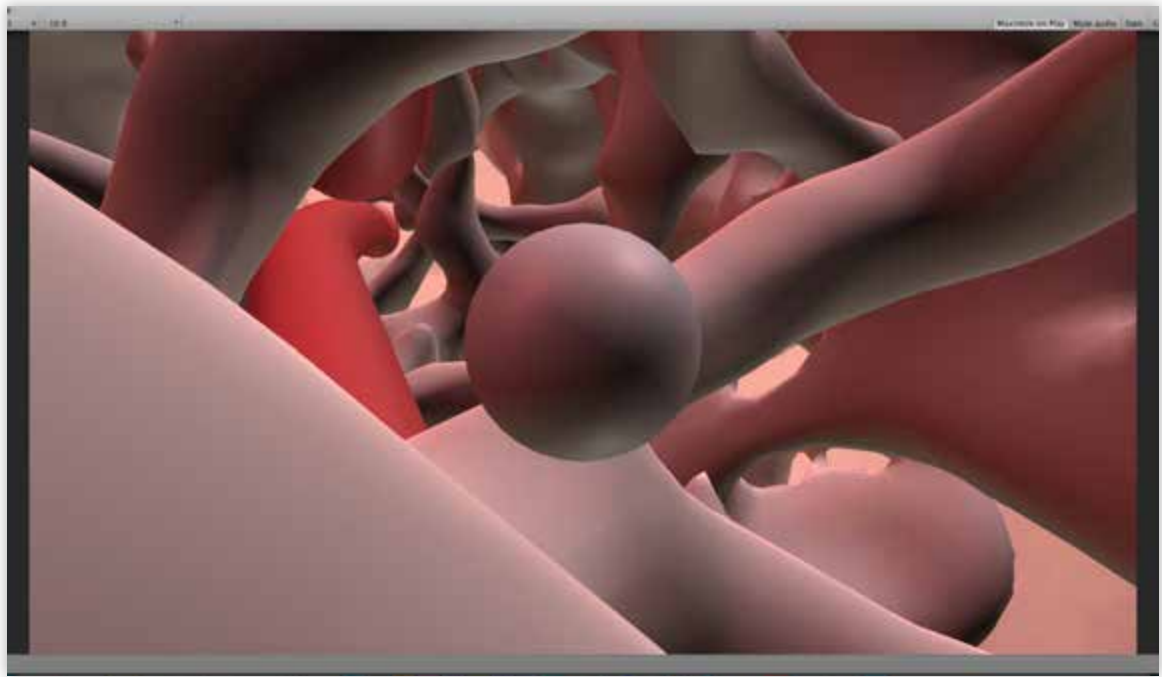


Figure 62 Unity 5 3D prototype build of Scene 1 with Particle Systems created to enhance atmosphere turned on. Text in this image is not intended to be read.

Particle system effects to enhance the scene's atmospheric quality were successfully created (Fig. 62). Mesh emitter particles were also created for a stromal cell, however, triggers and scripts for activating the particles and on-screen HUD elements have yet to be created and tested in the prototype scene.



**Figure 63** Unity 5 3D in-game demo screen capture of Scene 1 with placeholder sphere for the character controller. Text in this image is not intended to be read.

### **Access to Assets Resulting from this Thesis**

Access to visual development, digital 3D models, and Unity 5 3D prototype files resulting from this thesis can be viewed at the author's website [www.emilyling.com](http://www.emilyling.com), or by contacting the author at [contact.emilyling@gmail.com](mailto:contact.emilyling@gmail.com). The author may also be reached through the Department of Art as Applied to Medicine via the website [www.medicalart.johnshopkins.edu](http://www.medicalart.johnshopkins.edu).

## DISCUSSION

The purpose of this project was to create a video game demonstrating B lymphocyte development. A combination of novel gameplay design together with various elements from off-the-shelf entertainment video games allowed for the creation of a unique proof of concept. Limitations due to software proficiency and time constraints presented challenges to developing a playable demo. These issues will be addressed at a future phase of this project.

### **Rethinking Content and Gameplay for Immunology Video Games**

The evaluation of currently available video games for immunology allowed for the identification of certain trends in gameplay design. Many of the titles presented the immunology content from a more removed perspective. Players control actions of immune cells from either an omnipresent point-of-view or as a non-cellular, 3rd-person, character point-of-view. While these more removed experiences are based on an educational goal of demonstrating how multiple cells of the immune system operate together, the opportunity to gain a deeper foundational understanding of how each immune cell matures and develops is lost.

This led to the consideration of presenting educational material to the player by allowing him, or her, to embody the role of an immune cell. To experience the immune system from the perspective of an immune cell would facilitate a player's conceptualization of interactions between a developing immune cell in their

microenvironment and other cellular elements.

Subsequently, immersive gameplay in commercially available video games was considered for gameplay mechanics that would encourage players to experience a game world from the perspective of a defined character. Such features were: in-game narration of the player's actions as they control the defined character, minimal game HUDs or UI elements in the game field during gameplay, integrated access to extra educational content, and real-time prompts for gameplay challenges.

Overall, the new gameplay design considerations were chosen to better expose the players to educational materials as subtly as possible. If players are allowed to encounter the educational material in an integrated narrative, without disruptive presentation of educational content throughout the game experience, they may be more likely to enjoy the experience and continue playing.

## **Unity 5 3D**

Unity 5 3D offers a promising development tool for educational video game creation due to its generous online community support and documentation. However, there are some limitations that required the redesigning of characters for prototyping. Unity 5 3D does not, currently, support point level animation. The engine relies on joint system character rigs for in-game animations. Thus, the cellular character designs had to incorporate features of movement that could be modeled and bound to a joint system in C4D for export and use in Unity 5 3D. While the initial intention was to have a more spherical, squash-and-stretch rolling character movement, the joint system limited the

developed model's movement to a four-legged quadruped crawl. The eventual prototype model design made allowances for these constraints while still retaining a character personality through movement, which was originally designed using the initial squash-and-stretch concept.

## **Process of Modeling Assets for Unity 5 3D**

The modeling environment of C4D was ideal for creating 3D assets using numerical parameters. In addition, the C4D batch export plugin, CV-SmartExport, simplified exports of FBX files for individual objects in larger C4D project scenes. However, asset optimization was more efficient using ZBrush's Dynamesh retopology functions, and Decimation. These proved most useful with model meshes containing more than 65,000 vertices, which Unity 5 3D will automatically split into a single object with multiple sub-65,000 meshes. Other reasons for reducing polycounts and remeshing object surfaces in ZBrush was to reduce the potential computational strain in Unity 5 3D.

## **Learning Scripting, and Prototyping for Unity 5 3D in Segments**

One limitation to achieving a working prototype for the proof of concept was scripting in Unity 5 3D. Many of the gameplay mechanics features had to be developed in steps. Each script was tested for the desired outcome in the Unity 5 3D engine, which offered a manageable way to progress through the prototype's design. However, while Unity 5 3D offers a large online support commUnity 5 3D, as well as documentation, for developing C# script, many features required a higher level of coding to achieve the

desired in-game function.

The scripting was organized based on the main actions a player would follow for the B lymphocyte development branch of the game. In addition to the flowcharts, the narrative script was utilized as a guide to creating a list of scripts that had to be addressed for the desired game design. Due to time constraints and the author's limited C# coding proficiency, only early flowcharted features of the proof of concept were successfully developed.

### **Challenges of Scripting Character Control Physics**

The major challenge that delayed the prototype development involved the character controller script. The intended gameplay mechanics for movement was to have the player character stay parallel to the surface it was moving on without falling off the surface it was against. While various pre-developed controller scripts were found online, none restricted the character model's movement to the ground surface. This led to the conclusion that either an original script would have to be coded for this particular character movement, or the game's environment would have to be redesigned. Neither of these solutions were reasonable given the time frame of this project.

## **Future Goals and Directions for Game Prototype Development**

Initial feedback on the game design from the content expert, Dr. Mark Soloski, was encouraging. Due to the challenges of creating original script and coding for the prototype gameplay mechanics, the next steps for further development of the prototype will require the assistance of a more experienced C# coder who can develop script for Unity 5 3D. Once a coder is involved in the development of the game prototype, further asset modeling and optimization will be possible to complete a playable proof of concept.

With the completion of the playable demo, initial user audience testing can proceed. Feedback from user groups will be important in determining user experience, efficiency of content integration, and overall user response to the game design developed through this project.



## CONCLUSION

The foundational immunology concepts of lymphocyte development are important for beginning science students to comprehend. Video games offer the potential for a novel approach to teaching this complex subject matter by more effectively engaging students in this material. However, currently available educational video games intended to teach immunology have distinct limitations such as a lack of explicit demonstrations of the stages of lymphocyte development and clonal selection.

This project identified the content focus and gameplay mechanics of currently available immunology video games. Using this as a basis, a novel approach for developing an immunology video game with a focus on lymphocyte development was outlined with the primary goal of improving integration of educational content. The resulting design will serve as an improved model for integrating narrative gameplay and educational content presentation.

While the prototype was not developed to a playable demo, the important contribution of this thesis was the development of a new approach to designing a more effective educational video game specifically for immunology. Outcomes of this research will serve to inform future biomedical communicators on how to develop content for active learning games in immunology and provide a guide for designing full length educational video games featuring novel gameplay mechanics such as those identified through this project.

# APPENDICES

## Visual Development Sketches

### Blood Vessel Capillary Anatomy Studies

3 TYPES  
 BASAL LAMINA; BASEMENT MEMBRANE  
 Endothelial layer (continuous)

• Lipids types - through cell membranes  
 • ions - intercellular clefts  
 • water  
 • Tight junctions

• pores (60-80nm diameter)  
 • allows small molecules - limited proteins - diffuse

• Bone marrow primarily (Lymph nodes) + (adrenal gland)  
 • discontinuous capillary layer openings 90-40µm  
 • allows RBCs + WBCs and various serumproteins  
 • aided by discent. basal lamina  
 • gaps present in cell to cell junctions few transfer btw endothelial and across the membrane  
 • pinocytotic vesicles lacking

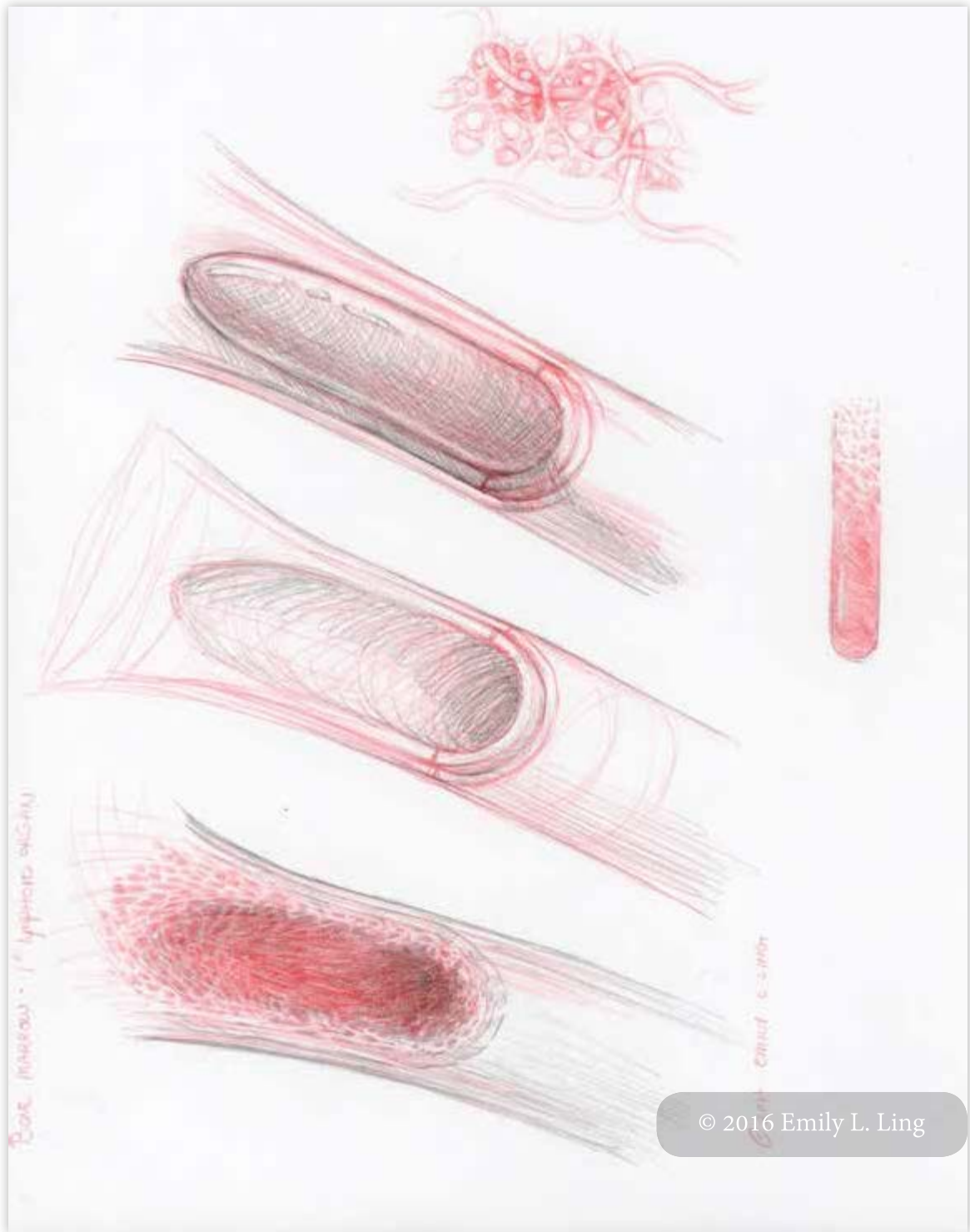
• some that lack tight junctions btw cells  
 • liver + spleen = discent. sin capillaries where greater movement of cells is necessary  
 • \* capillary wall is only 1 cell thick - simple squamous

Continuous  
 Fenestrated  
 Sinusoid

© 2016 Emily L. Ling

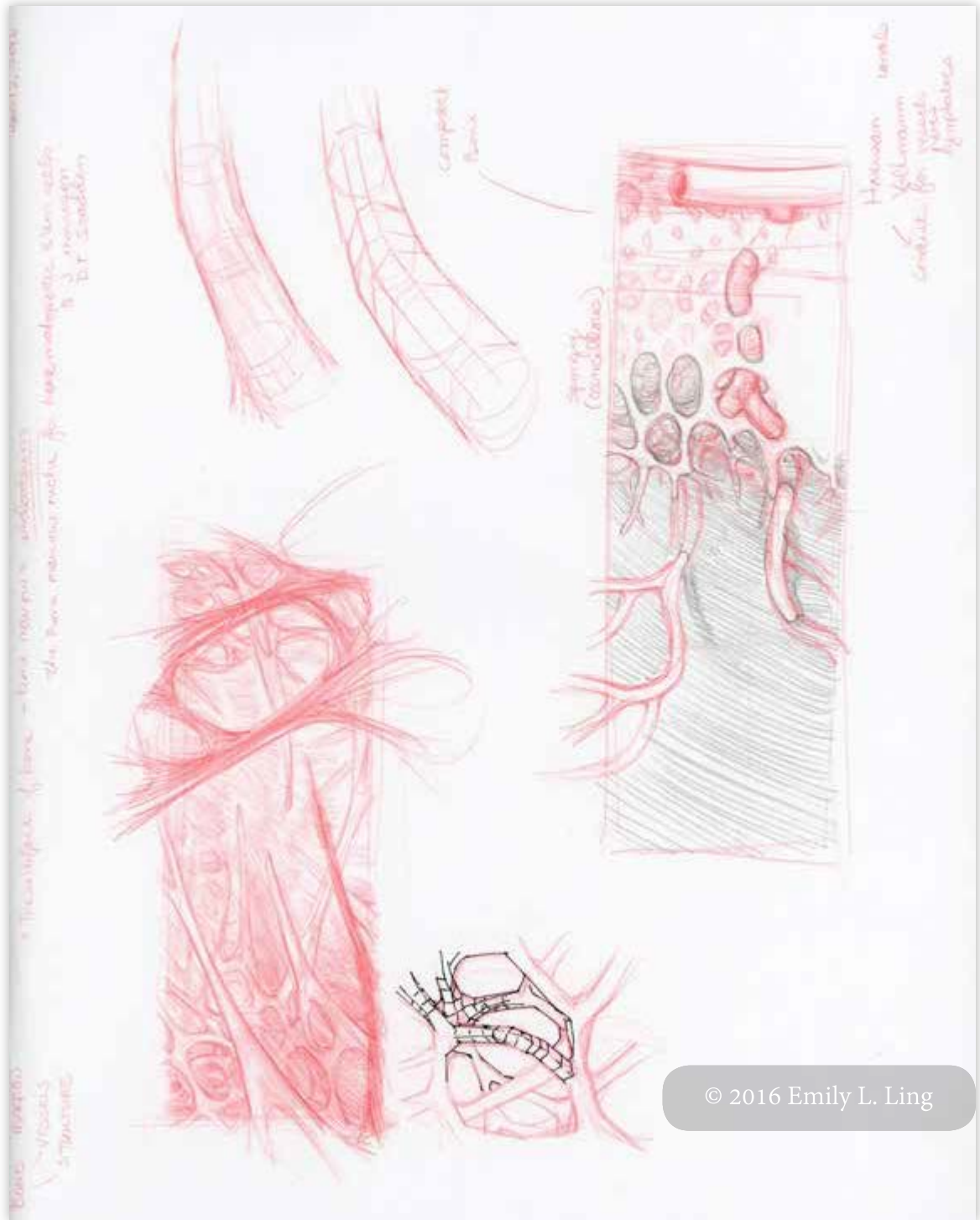
# Visual Development Sketches

## Bone Marrow, Micro Environment Studies and Concepts



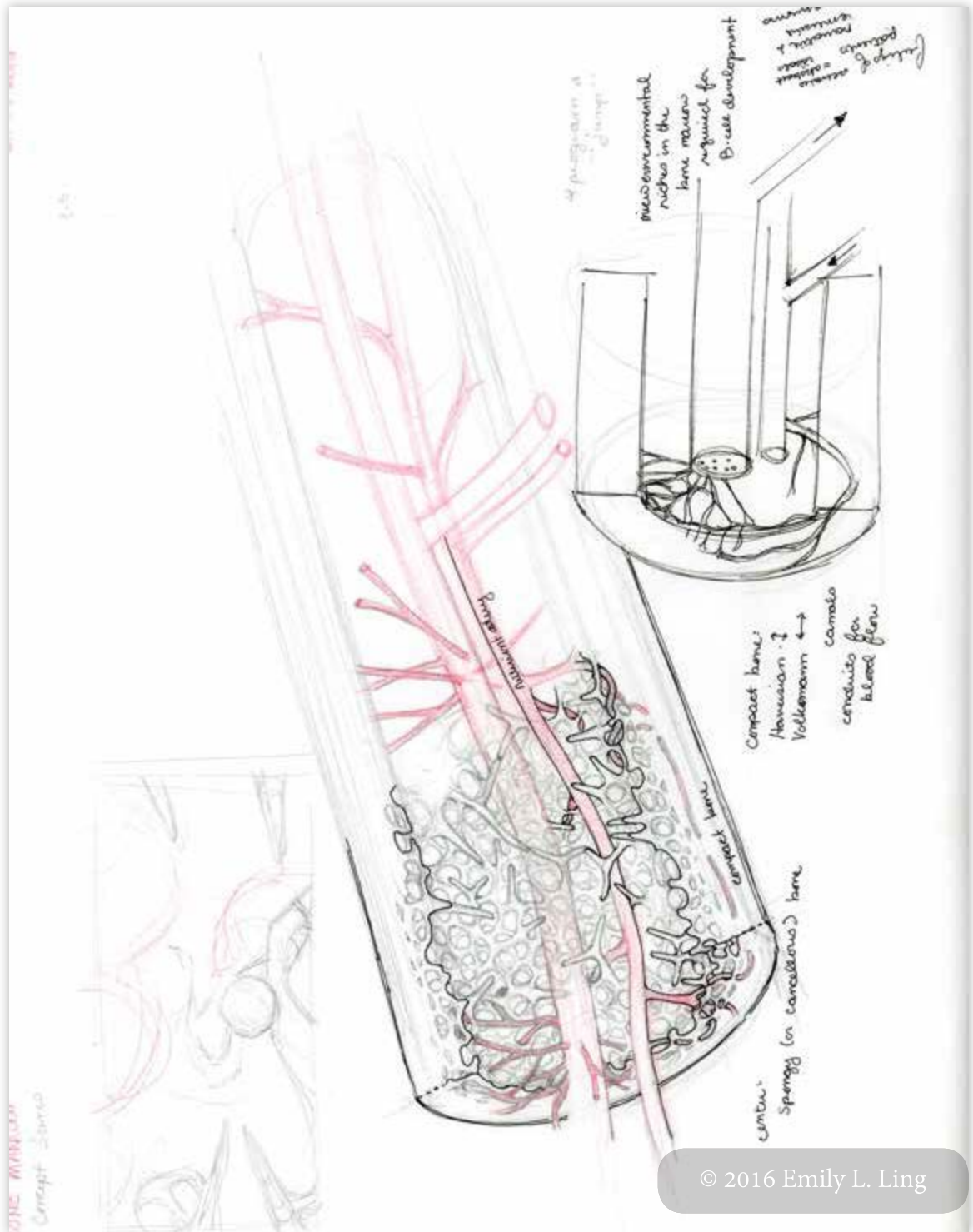
# Visual Development Sketches

## Bone Marrow, Micro Environment Studies and Concepts



# Visual Development Sketches

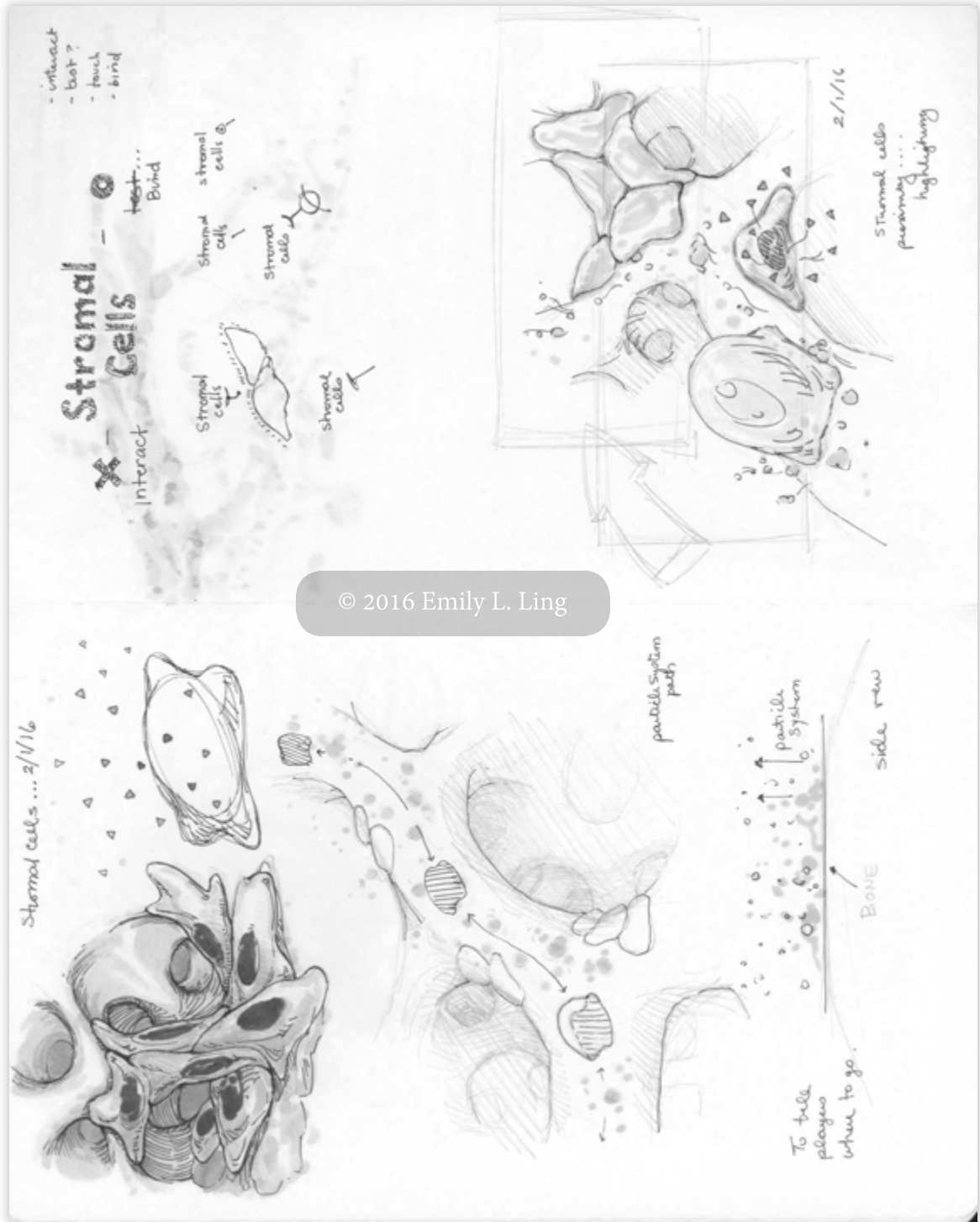
## Bone Marrow, Micro Environment Studies and Concepts



© 2016 Emily L. Ling

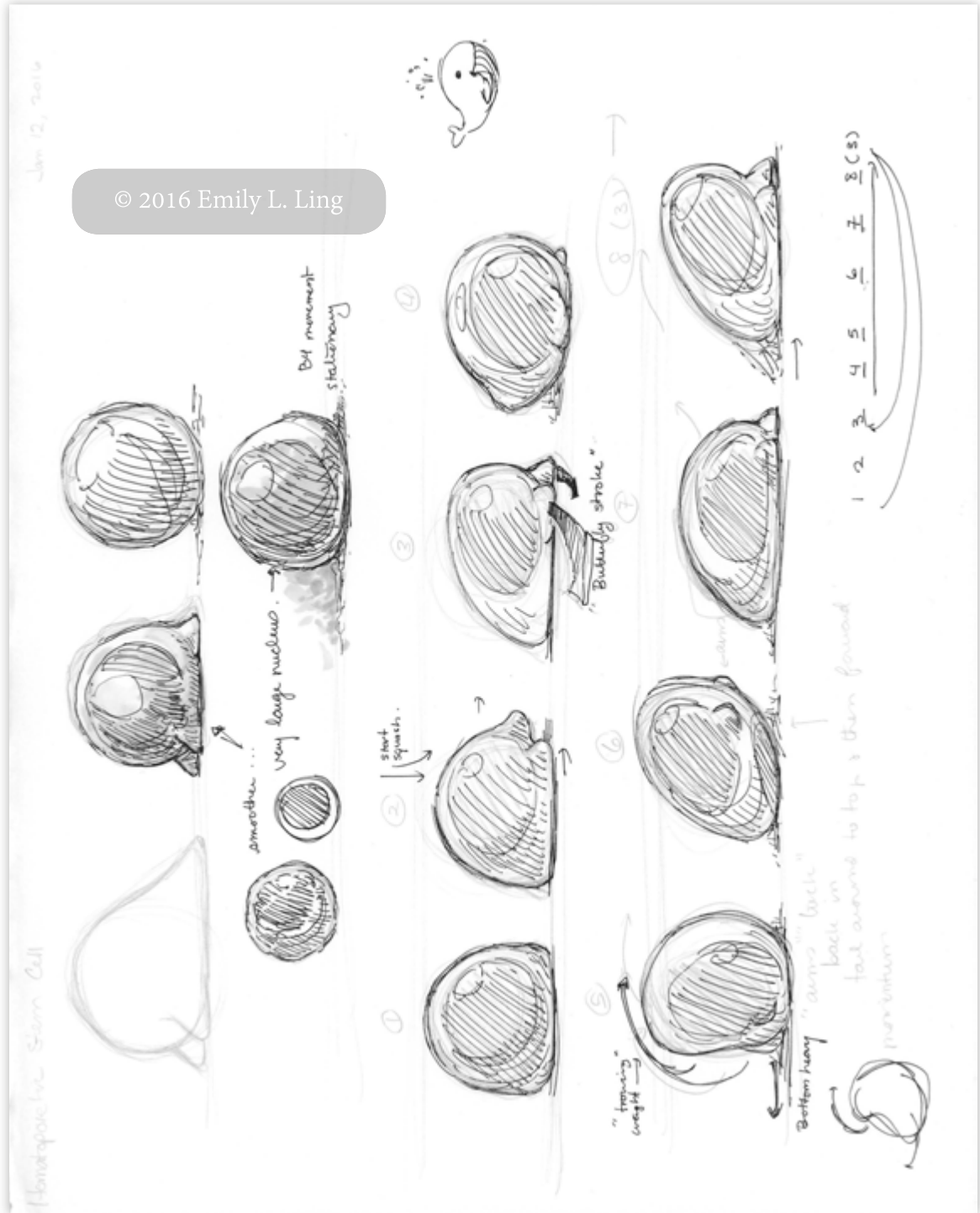
# Visual Development Sketches

## Bone Marrow, Micro Environment Studies and Concepts



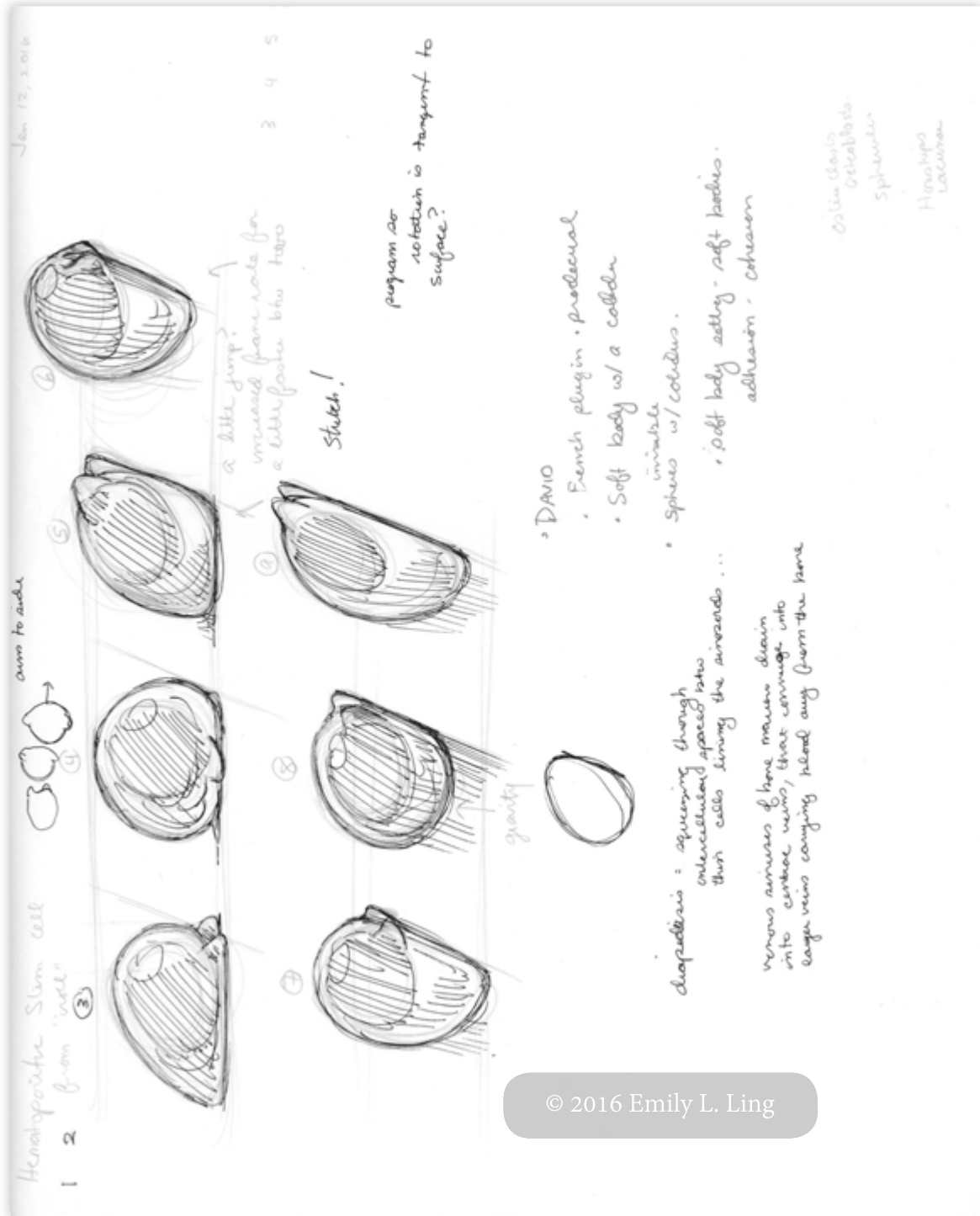
# Visual Development Sketches

## Character and Movement Concepts



# Visual Development Sketches

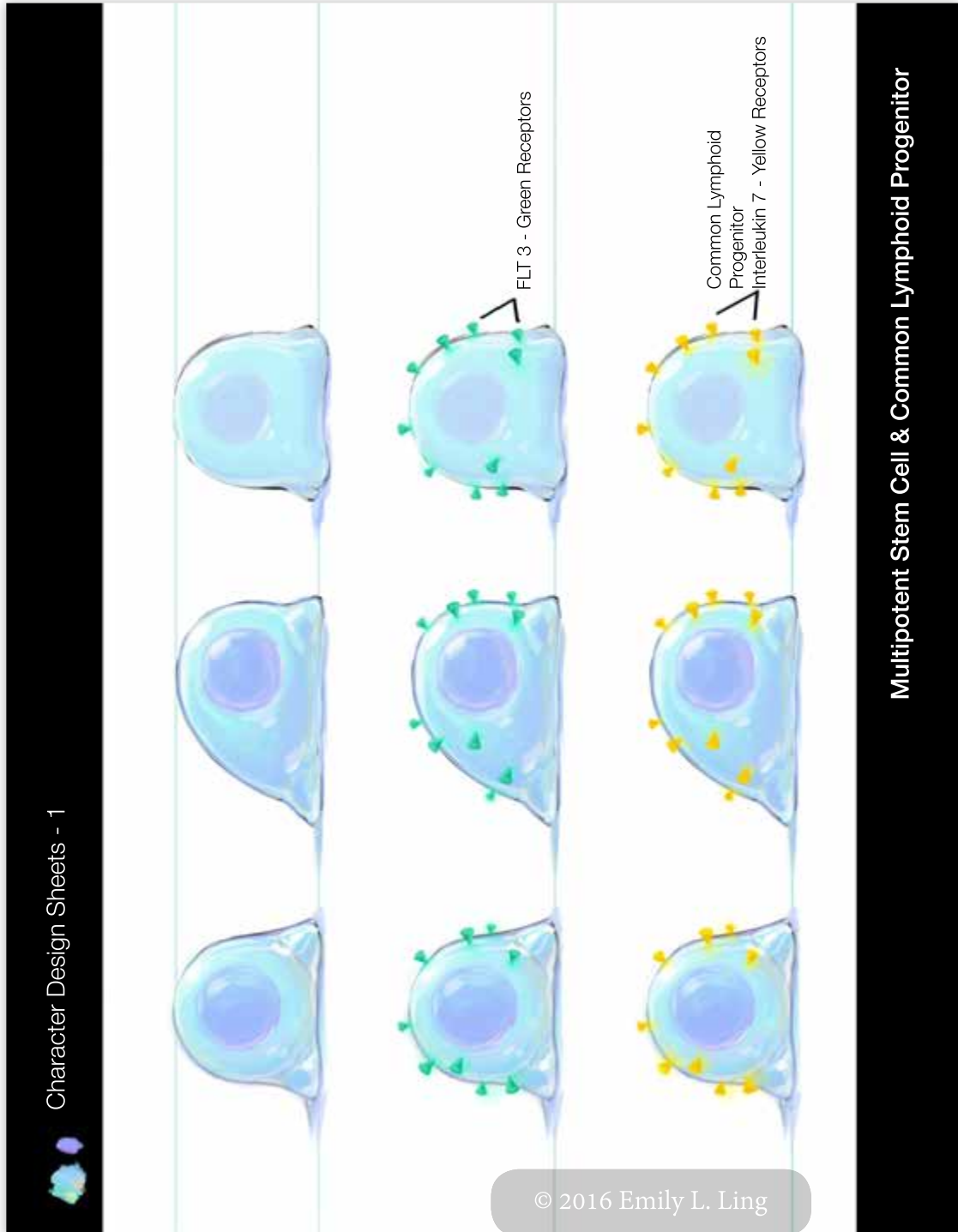
## Character and Movement Concepts





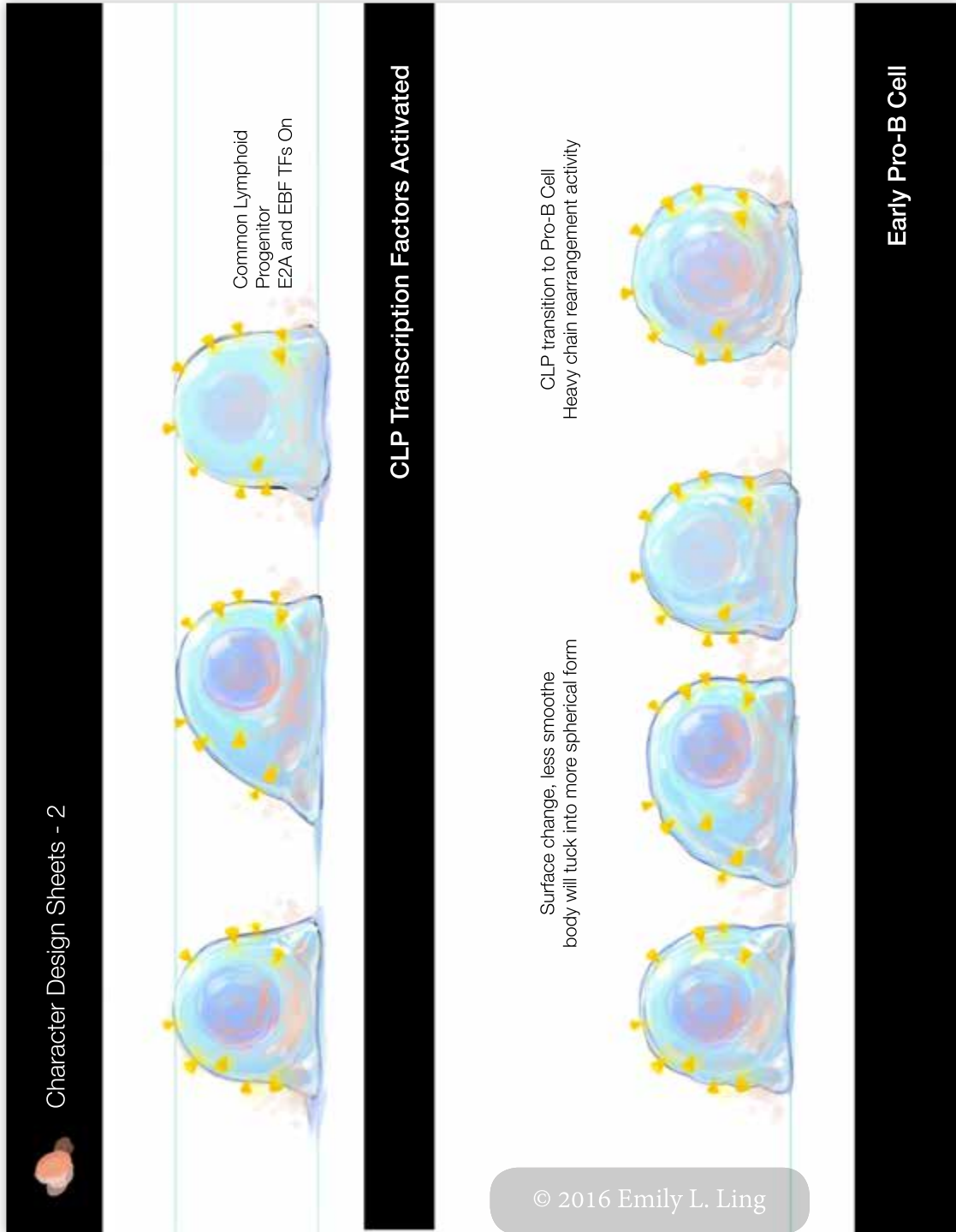
# Visual Development Sketches

## Character Design Sheet



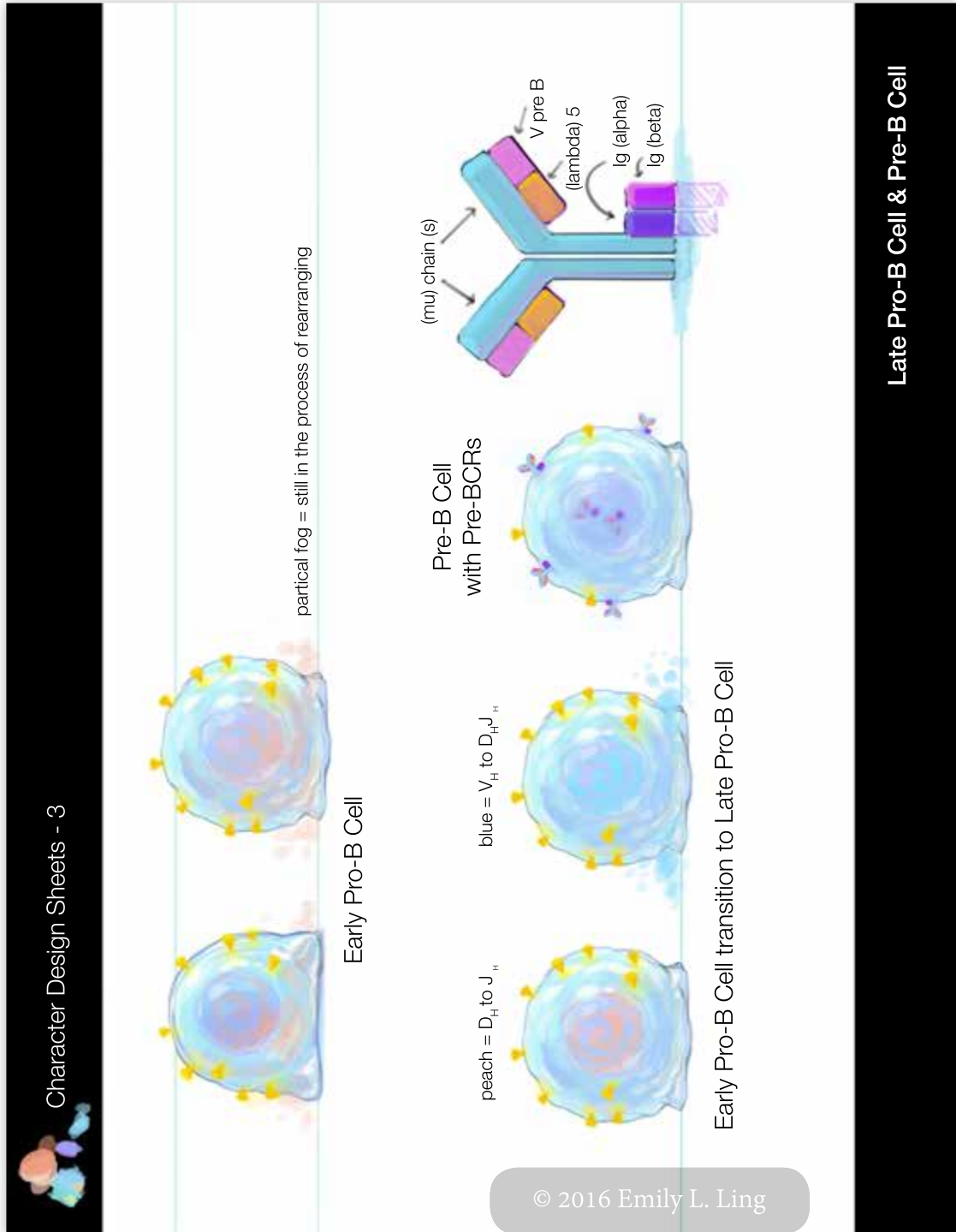
# Visual Development Sketches

## Character Design Sheet



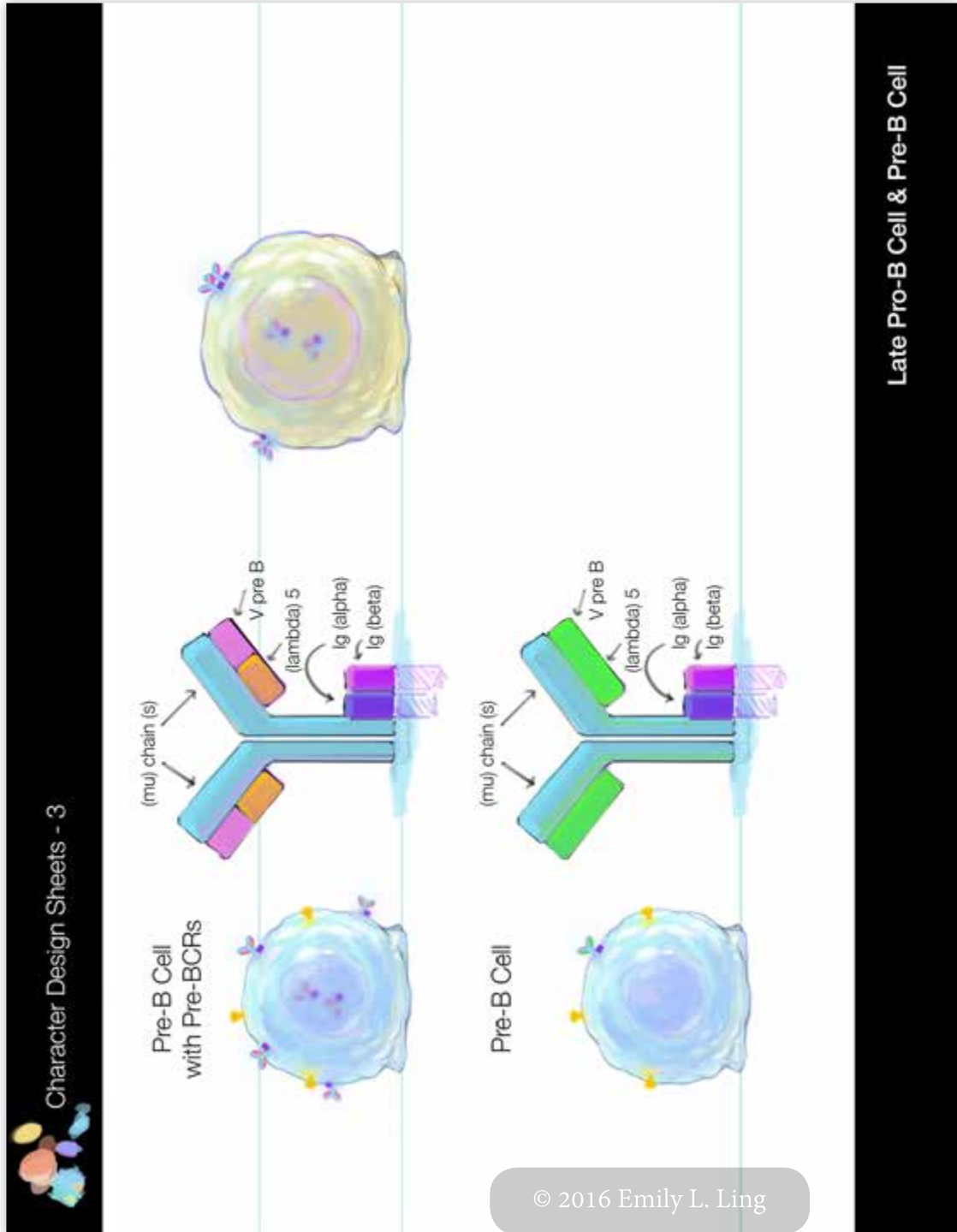
# Visual Development Sketches

## Character Design Sheet



# Visual Development Sketches

## Character Design Sheet



# Storyboards

## Main Menu Layout Concepts

Project Thesis Alpha Sequence Main Date 1/29/16 Client  
(Main Menu) Menu (A) ideas Rgt

Notes:  
Video: for prototyping  
seasons, keep  
BG as single image  
simple loop  
or matte films?

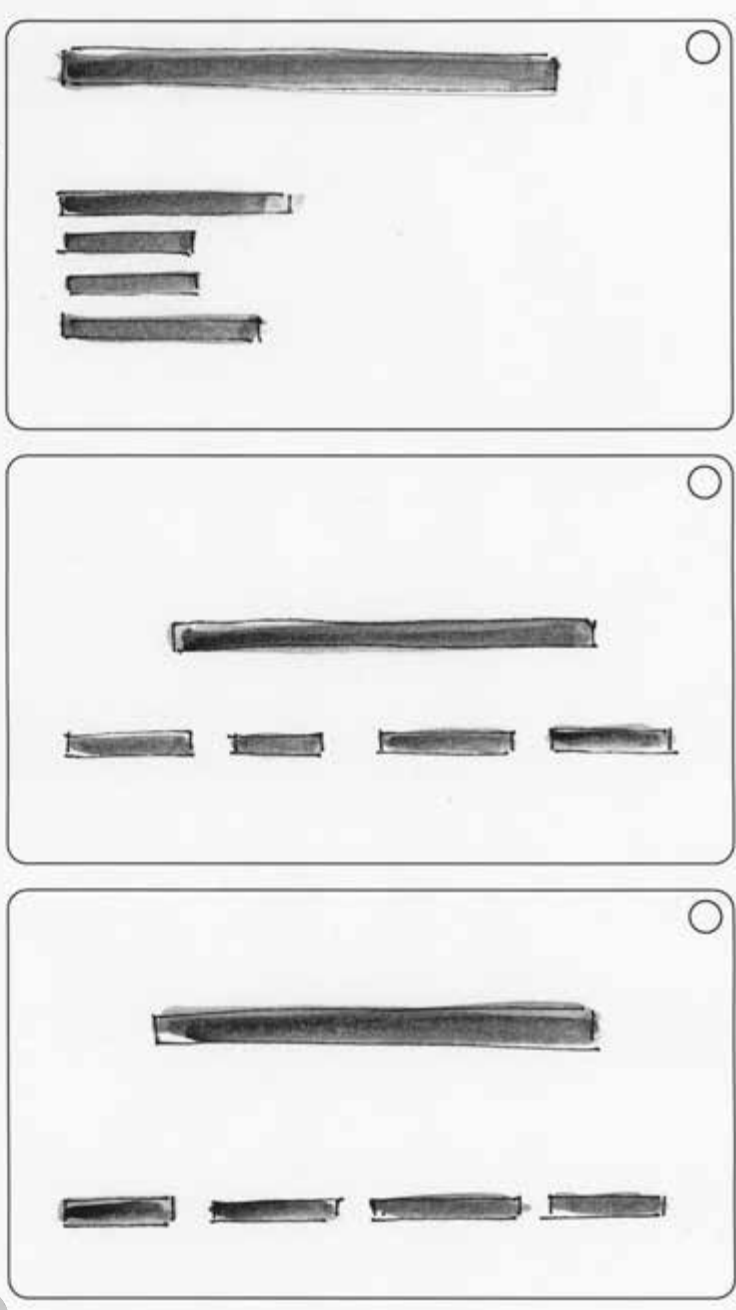
Audio: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

Video: \_\_\_\_\_  
\_\_\_\_\_

Audio: \_\_\_\_\_  
\_\_\_\_\_

Video: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

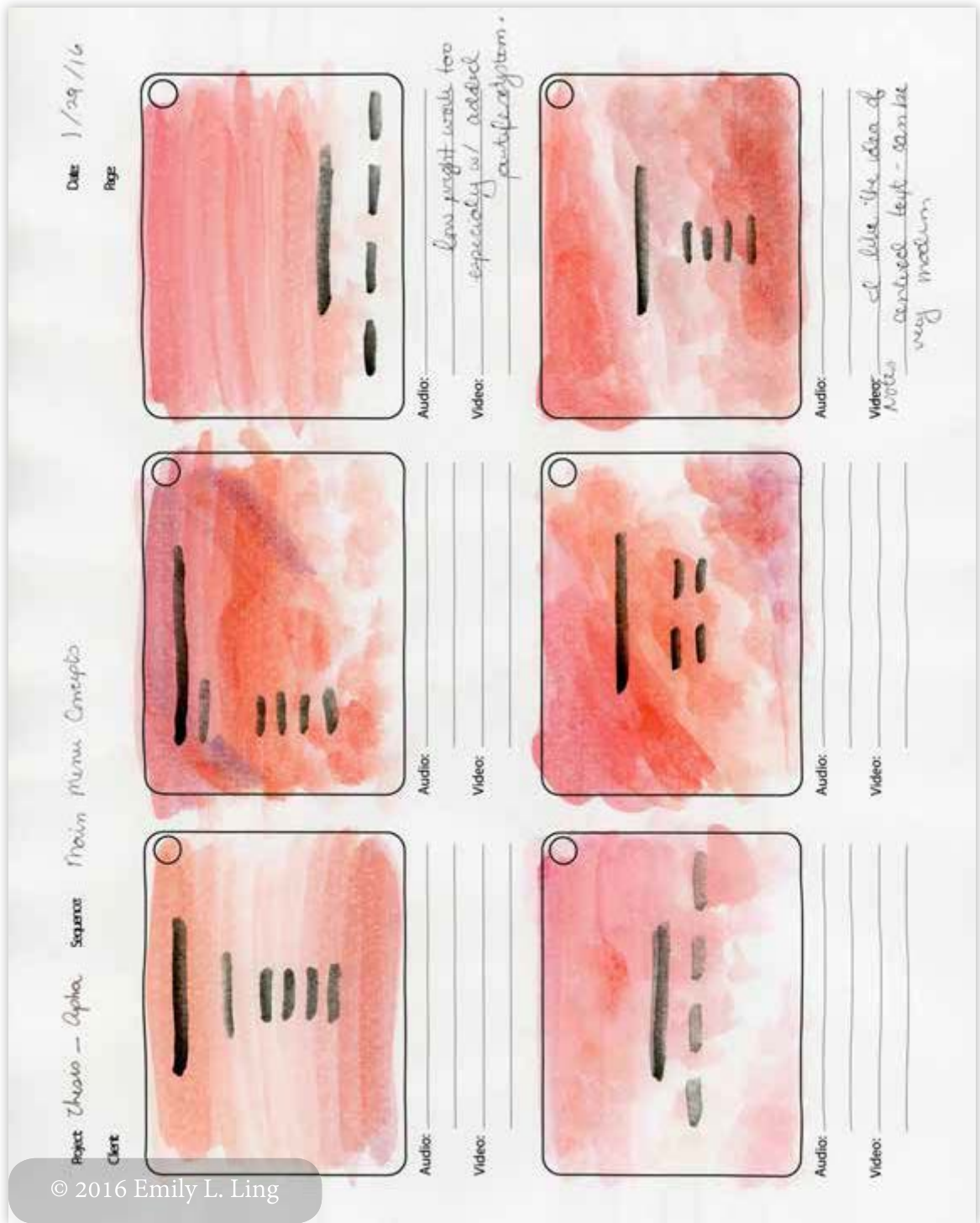
Audio: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_



© 2016 Emily L. Ling

# Storyboards

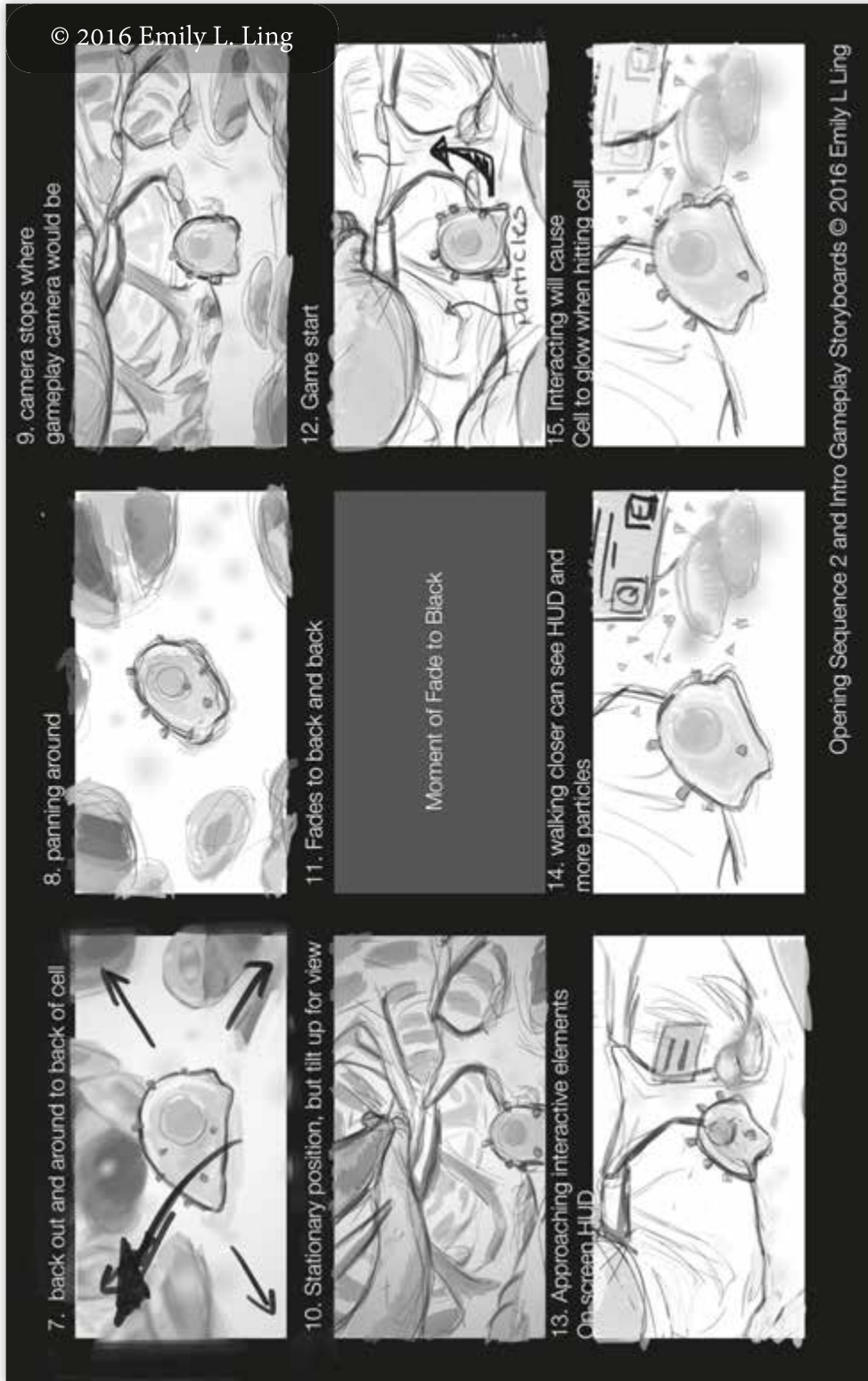
## Main Menu Layout Concepts



# Storyboards



# Storyboards





# Narration Script

Thesis Narration Alpha

Written By

Emily Lunhui Ling  
Masters Candidate, class of 2016

Based on, Janeway's Immunobiology - 8th edition

Johns Hopkins University School of Medicine  
Department of Art as Applied to Medicine  
1830 E. Monument St. Suite 7000  
Baltimore, MD, 21287  
410-955-3213

© 2016 Emily L. Ling

# Narration Script

1

FADE IN:

ANIM. B-CELL IN SPACE

Model of a B-cell and its antibody shooting capabilities. Background still image or animation of environment of the bone marrow.

Particle system: Stromal cell-derived factor 1 (SDF-1) (aka. CXCL12) around areas that are accessible to player.

NARRATOR: Tenor-voiced male, or mature female. Tone is conversational.

NARRATOR (V.O.)

B cells, or B lymphocytes are amazing. They are one of many kinds of lymphocytes, or white blood cells, that are part of the human body's adaptive immune response. B cells are the ones that use antibodies to target invading pathogens.

Now, ever wonder where these specialized white blood cells come from?

DISSOLVE TO:

**BEGIN SEQUENCE 1**

IN-GAME ANIM. BONE MARROW

In-game pan through of the microenvironment of the bone marrow. Highlighting features: trabeculae, stromal cells, floaties and ligand release.

NARRATOR (V.O.)

Everything starts in primary lymphoid organs, like bones. Within the bone is the bone marrow, where all the blood cells of the human body, be it red blood cells, or white blood cells, are produced from self-renewing cells called hematopoietic stem cells.

So, then, how do hematopoietic stem cells give rise to the specialized B cell?

© 2016 Emily L. Ling

# Narration Script

2

DISSOLVE TO:

IN-GAME PLAY. BONE MARROW

Camera runs through the bone marrow and centers onto the  
PLAYER character.

DISSOLVE TO:

IN-GAME PLAY - SCENE 1 - MULTIPOTENT STEM CELL

PLAYER begins to move forward and explore initial opening of  
gameplay. Functions are programmed default navigation and  
camera control settings.

MULTIPOTENT STEM CELL, is pale in color, and has a shape on  
it's surface or particle system that demonstrates that it  
has a cell surface receptor: FLT3

FLT3 RECEPTOR: TRIANGLES / GREEN. Particle system: cloud

NARRATOR (V.O.)

The MULTIPOTENT PROGENITOR CELL has  
arisen from a hematopoietic stem  
cell.

No specific surface features are  
present on it that can distinguish  
it as a specialized type of cell...

PLAYER moves further forward. Ensure scene does not allow  
PLAYER to progress far enough to begin interacting with  
stromal cells.

NARRATOR (V.O.) (CONT'D)

...except for the presence of a  
surface receptor known as FLT3  
(FLT3 surface receptor  
form is highlighted)  
(NEW CONTENT UNLOCKED:  
"FLT3 Receptor and  
Ligand")

These cells can produce both  
lymphoid and myeloid lineage cells,  
but are no-longer self renewing.  
(NEW CONTENT UNLOCKED:  
"Lymphoid and Myeloid")

UNLOCKED CONTENT SEQUENCE

© 2016 Emily L. Ling

# Narration Script

3

With the first "NEW CONTENT UNLOCKED", ON SCREEN TEXT PROMPT: "PRESS the "\" key on your keyboard to see content about "Lymphoid and Myeloid". Also, at any time, you can navigate to your UNLOCKED CONTENT from the HOME SCREEN".

IF they press "\" HUD appears and plays or shows content. ON SCREEN TEXT PROMPT: "To RETURN TO GAME, PRESS "Esc".

This applies to all instances of "NEW CONTENT UNLOCKED" except after the first prompt, "Also, at any time, you can navigate to your UNLOCKED CONTENT from the HOME SCREEN" no longer appears.

END UNLOCKED CONTENT SEQUENCE - RETURN SCENE 1

NARRATOR (V.O.) (CONT'D)

The bone marrow is a specialized microscopic environment with supportive stromal cells that aid in the development of hematopoietic stem cell derivatives.

(Stromal cells highlighted with a quick glow then glow disappears.)

Stromal cell-derived factor 1 (SDF-1), also known as C-X-C motif chemokine 12, is a chemokine that helps the developing cell stay on the path to success. It keeps the cell from migrating out of the bone too early.

(NEW CONTENT UNLOCKED: "Chemokines" and "Stromal cell-derived factor 1")

Following spoken narration, stromal cells that are highlighted with UI elements for interaction.

UI element for discovered structures reveals name and actions [key]s for interacting with GameObject appear based on a closer proximity distance to the GameObject.

NARRATOR (V.O.)

Bone marrow stromal cells have, on their surfaces, FLT3 ligands.

(stromal cells that become highlighted are highlighted with same

© 2016 Emily L. Ling

# Narration Script

4

color , GREEN, as FLT3  
receptor convention.  
TRIANGLE-shaped particles  
particle system around  
them are turned on when  
action [key]s HUD  
activated due to  
proximity trigger.)

IN-GAME PLAY - SCENE 2 - FLT3 TO CLP

Wait for PLAYER to move towards stromal cells. UI proximity  
detection brings up action functions. [key] for "interact  
with stromal cell".

After action [key] is triggered:  
(Particle system: cloud.  
Particle system  
representing FLT3  
receptor around  
MULTIPOTENT STEM CELL  
grows brighter.)

NARRATOR (V.O.)  
Binding of the FLT3 receptor with  
FLT3 ligands through contact with  
the stromal cell surfaces signals  
the MULTIPOTENT STEM CELL to...  
(color for FLT3 receptor  
particle system cloud  
color change to ORANGE.)

... differentiate to the next stage  
of development: the COMMON LYMPHOID  
PROGENITOR.

COMMON LYMPHOID PROGENITOR, represented with ORANGE color  
scheme shifts, including convention for particle system:  
cloud. Main PLAYER model is still pale in local color.

IN-GAME PLAY - SCENE 3 - CLP IL-7

Wait for PLAYER to move again 2 seconds following end of  
SCENE 2 narration.

Trabeculae and reticular stromal cell regions are available  
to walk with areas of stromal cells ahead in pockets and  
regions. Some with interaction UI, others just animated  
environment.

© 2016 Emily L. Ling

# Narration Script

5

NARRATOR (V.O.)

With signaling from FLT3 and the transition to a COMMON LYMPHOID PROGENITOR cell, a new surface receptor is expressed.

(IL-7 receptor appears:  
INVERSE CONES and  
ORANGE.)

The interleukin-7 (IL-7) receptor.  
(NEW CONTENT UNLOCKED:  
"Interleukin-7 Receptor")

INTERLEUKIN-7: INVERSE CONES / ORANGE. Particle system:  
cloud.

PLAYER can continue to move. Create scene so that the next set of interactive stromal cells are in the distance and surrounded by a slightly different particle fog system. Color for fog is ORANGE.

Upon approaching the orange fogged stromal cells, UI name appears and at a specific proximity triggers particle system to release cone-shaped orange particles.

(Action [key] to interact  
with action HUD appears  
when proximity distance  
within parameters.)

Trigger action [key] or no trigger leads to narration via proximity.

NARRATOR (V.O.)

Cytokine interleukin-7 is secreted by the bone marrow stromal cells and essential for the growth and survival of developing B cells.

(NEW CONTENT UNLOCKED:  
"Cytokines")

If triggered during narration, timer bar will appear following interaction with UI HUD for IL-7.

(FLT3 particle cloud  
changes colors to ORANGE  
and glows bright)

TIMER BAR for IL-7: 2 minute. 1 minute timer, alpha is 75%;  
30 sec timer, alpha is 20%.

At 30 sec, 20% alpha

© 2016 Emily L. Ling

# Narration Script

6

**\*\* BRANCHING SEQUENCES - 0 \*\***

**\*\* END BRANCHING SEQUENCE - 0 \*\***

NARRATOR (SUBTITLE)  
Revitalize the B cell with IL-7  
signaling.

Most stromal cells from this point on will be action HUD  
types for IL-7.

IN-GAME PLAY - SCENE 4 - INTRACELLULAR E2A AND EBF

Current environment mostly stromal cells with fog particle  
system ORANGE for IL-7.

(When 1 minute timer is  
left on IL-7 signaling,  
allow UI HUD IL-7 to  
appear on these stromal  
cell structures. )

Trabecular spaces that are part of navigated spaces with  
CXCL12 particle system highlighting or suggesting route  
paths that can be explored.

At this stage, path is mostly in the endosteum region, close  
to the more peripheral trabecular spaces.

PLAYER continues moving through bone marrow environment.  
Navigation now opens further towards a more centralized path  
away from the peripheral endosteum spaces; CXCL12.

NARRATOR (V.O.)  
At this point, the COMMON LYMPHOID  
PROGENITOR begins to express  
internal transcription factors that  
contribute to its development into  
the EARLY PRO-B CELL.

(Insert a transcription  
factor particle system  
that follows the mesh:  
E2A and EBF) internally.)  
(NEW CONTENT UNLOCKED:  
"Transcription Factors")

Notable transcription factors are  
E2A and EBF. These transcription  
factors induce the expression of  
important proteins that allow the  
(MORE)

© 2016 Emily L. Ling

# Narration Script

7

NARRATOR (V.O.) (CONT'D)  
cell to begin rearrangement and  
expression of immunoglobulin genes.  
(NEW CONTENT UNLOCKED:  
"E2A"; "EBF";  
"Immunoglobulins")

FADE TO:

## BEGIN SEQUENCE 2

IN-GAME ANIM. INTERNAL - HEAVY CHAIN

In game animation sequence transition. Cell rolls on the surface of a trabecular bone and near stromal cell.

NARRATOR (V.O.)  
B cell development proceeds through  
stages of rearrangement and  
expression of immunoglobulin  
genes...

Close up of the cell, then opens a split screen window HUD.

NARRATOR (V.O.)  
...The first of which is  
rearrangement in the heavy-chain  
gene locus, or location, for  
production of a functional heavy  
chain, called a (mu) chain.  
(NEW CONTENT UNLOCKED:  
"(Mu) Chain")

MATCH CUT TO:

IN-GAME PLAY - SCENE 5 - HEAVY CHAIN GAME

NARRATOR (V.O.)  
The first stage is the PRO-B CELL  
stage, which can be divided into an  
EARLY or LATE PRO-B CELL stage.  
(HUD states the stage of  
the B-cell; Here, it  
displays EARLY PRO-B CELL  
following the narration.)

In the EARLY PRO-B CELL stage, the  
heavy-chain gene rearrangement  
begins with the joining of the  
diversity (D) segment to the  
joining (J) segment.  
(On screen prompt appears

© 2016 Emily L. Ling



# Narration Script

8

for controls. "Use [input  
1] and [input 2] to bring  
the two segments  
together")

PLAYER CONTROL returns. Dual [input] control system to match  
sides. When aligned properly the two sides will bring up a  
UI element for an action [key] trigger. Event following the  
action [key]:

(segment joins together  
with a particle system  
effect smoke to  
transition change to  
shorter gene segment.  
Removed INTRON  
dissolves.)

**\*\* BRANCHING SEQUENCES - 1 \*\***

**\*\* END BRANCHING SEQUENCE - 1 \***

IN-GAME PLAY - SCENE 6 - V TO DJ

Following D to J rearrangement.

NARRATOR (V.O.)

Now that the D and J segments of  
the heavy chain have successfully  
joined, the EARLY PRO-B CELL enters  
the LATE PRO-B CELL stage.

(HUD states the stage of  
development. Changes from  
EARLY PRO-B CELL to LATE  
PRO-B CELL)

The variable (V) gene segment now  
begins rearrangement to join to the  
DJ gene segment.

PLAYER CONTROL returns. Dual [input] control system to match  
sides. When aligned properly the two sides will bring up a  
UI element for an action [key] trigger. Event following the  
action [key]:

(segment joins together  
with a particle system  
effect smoke to  
transition change to  
shorter gene segment.  
Removed INTRON

© 2016 Emily L. Ling

# Narration Script

9

dissolves.)

**\*\* BRANCHING SEQUENCES - 2 \*\***

**\*\* END BRANCHING SEQUENCE - 2 \*\***

IN-GAME PLAY - SCENE 7 - PRE-BCR

Still in split screen mode.

NARRATOR (V.O.)

A (mu) chain is now formed from the successful rearrangement of the VDJ heavy chain gene segments. The cell becomes a PRE-B CELL.

(HUD stage display changes to reflect it is now a PRE-B CELL.)

SPLIT SCREEN

Side with the cell model shows a (mu) chain at it's core. Adjust alpha on texture for transparency affect.

Side with model close ups in split screen HUD shows the (mu) chain in intracellular environment. Shows the structures in space to highlight forms.

NARRATOR (V.O.)

To test that a functional (mu) chain has been created, some (mu) chains are brought to the surface by incorporation into a surface receptor known as a PRE-B-CELL RECEPTOR (PRE-BCR).

(NEW CONTENT UNLOCKED:  
"Pre-B-Cell Receptor")

PLAYER CONTROL. Directional orientation to be adjusted until in a proper range. Then action [key] trigger. Upon triggering the action [key] the PRE-BCR will animate migrating to the surface of the model.

Transparency on the cell model is returned to opaque as the (mu) chain combines with other components and appears on the cell's surface. Ig(alpha) and Ig(beta) are not on the surface yet.

© 2016 Emily L. Ling

# Narration Script

10

NARRATOR (V.O.)

The PRE-BCR is comprised of the  
(mu) chain...  
(mu) chain highlighted  
though momentary color  
shift)

...two surrogate light chains  
(alpha)5 and VpreB...  
(named structures are  
highlighted after  
narration spoken by means  
of a momentary color  
shift)  
(NEW CONTENT UNLOCKED:  
"Surrogate Light Chains")

...and the signal transduction  
proteins Ig(alpha) and Ig(beta).  
(Ig(alpha) and Ig(beta)  
appear next to the (mu)  
chain and surrogate chain  
structures after being  
named. They appear on the  
cell surface as well.)  
(NEW CONTENT UNLOCKED:  
"Signal Transduction  
Proteins")

These PRE-BCRs are present at low  
levels on the cell surface...  
(3 more PRE-BCRs slowly  
appear on the surface as  
narration completes.)

PLAYER CONTROL. Button mash [key] to cause PRE-BCRs to begin  
gathering to each other

NARRATOR (V.O.)

It signals the PRO-B CELL to  
transition to the PRE-B CELL stage  
through formation of PRE-BCR dimers  
or oligomers on the cell surface.

PLAYER CONTROL. Once two dimers form, the Ig(alpha) and  
Ig(beta) will begin to glow. This will prompt the player to  
hit a sequence of action [key]s. Triggering the next event  
by completing the action [key]s will cause the Ig(alpha) and  
Ig(beta) to glow brighter and then trigger a particle  
system.

© 2016 Emily L. Ling

# Narration Script

11

(particle system is a downward, towards the center of the cell model, cloud that will fade away. Loop a few times. Particle system represents signal transduction.)

MATCH CUT TO:

## BEGIN SEQUENCE 3

### IN-GAME PLAY - SCENE 8 - IL-7 SENSITIVITY

Return to bone marrow exploratory game view (i.e. not split screen close up and HUD). Stromal cells in the environment still visible with ORANGE fog particle systems.

PLAYER CONTROL. Return to free explore and interaction with stromal cells or movement along CXCL12 particle system highlighted areas. Allow paths for highlighted areas to move a bit closer towards central sinus model region.

Once PLAYER begins to move closer to stromal cells, triggered by proximity:

NARRATOR (V.O.)

The PRE-B CELL is now particularly sensitive to IL-7 ligands released by bone marrow stromal cells. Whereas before, the IL-7 interaction was a signal for checking the B cell's proper development and survival, IL-7 now also...

Wait for PLAYER to begin moving into and interacting with UI action HUD action [key] for stromal cells releasing IL-7 particle systems.

NARRATOR (V.O.)

...induces the PRE-B CELL to proliferate, initiating the developing B cell's transition to the LARGE PRE-B CELL.

PLAYER may interact with stromal cells. UI proximity feature still applies, but action [key] will also include a text "Proliferate". Following each action [key] trigger the cell model will divide.

© 2016 Emily L. Ling

# Narration Script

12

(New model stays relatively still with some pre-programmed animations.)

ON SCREEN TEXT: "It is not well understood what signals the LARGE PRE-B CELL to stop dividing uncontrollable, but LARGE PRE-B CELLS divide rapidly for a period of time before its natural transition into the SMALL PRE-B CELL stage." Text will fade once PLAYER triggers the 5 divisions into the next scene (SCENE 9).

Once PLAYER triggers division 5 times.

FADE TO:

IN-GAME ANIM. PROLIFERATION

Animation will add a few more proliferated cells into the scene, and then pan to center on the main PLAYER model.

MATCH CUT TO:

**BEGIN SEQUENCE 4**

IN-GAME PLAY - SCENE 9 - LIGHT CHAIN GAME

SPLIT SCREEN

Following pan will be the centering and closer framing of the main controllable model. Return the UI HUD split screen display for the mini game.

(HUD states LARGE PRE-B CELL.)

NARRATOR (V.O.)

Once the PRE-B CELL has proliferated, proliferation stops and the cell becomes a SMALL PRE-B CELL.

(HUD states LARGE PRE-B CELL and changes to SMALL PRE-B CELL and gets smaller.)

Light-chain gene locus rearrangement takes place one allele at a time. Light-chain gene loci lack diversity (D) segments and, therefore, rearrangement occurs between variable (V) gene

(MORE)

© 2016 Emily L. Ling

# Narration Script

13

NARRATOR (V.O.) (CONT'D)

segments to joining (J) segments.  
(On screen prompt appears  
for controls. "Use [input  
1] and [input 2] to bring  
the two segments  
together")

PLAYER CONTROL returns. First movement brings (kappa) V  
segment together with the first J segment.

NARRATOR (V.O.)

Each chromosome has a few attempts  
to successfully rearrange a V to J  
gene segment.

PLAYER CONTROL returns. Dual [input] control system to match  
sides. When aligned properly the two sides will bring up a  
UI element for an action [key] trigger. Event following the  
action [key]:

(segment joins together  
with a particle system  
effect smoke to  
transition change to  
shorter gene segment.  
Removed INTRON  
dissolves.)

NARRATOR (V.O.)

After a successful rearrangement  
and joining of the V to J, the  
light chain is paired with a (mu)  
chain...

(Light chain on the split  
screen HUD side is shown  
complete and then joined  
to the intracellular (mu)  
chains. Main model side  
has a lower alpha to  
reveal the light chain  
joining the intracellular  
(mu) chains.)

...and IgMs can be expressed on the  
surface, surface immunoglobulin  
(mu)s.

(Main model side shows  
the IgM)

PLAYER CONTROL returns, and PLAYER must button mash an  
action [key] and align the IgM model to bring it to the  
surface. After a certain distance from the surface, it will

© 2016 Emily L. Ling

# Narration Script

14

trigger an animation that finishes and shows the IgM on the surface. A few more appear as well. Split side shows the model move into the surface. Ig(alpha) and Ig(beta) come to the surface as well.

NARRATOR (V.O.)

The SMALL PRE-B CELL has now become an IMMATURE B CELL.

(HUD name changes from SMALL PRE-B CELL to IMMATURE B CELL.)

**\*\* BRANCHING SEQUENCES - 3 \*\***

**\*\* END BRANCHING SEQUENCE - 3 \*\***

MATCH CUT TO:

**BEGIN SEQUENCE 5**

IN-GAME PLAY - SCENE 10 - IMMATURE B CELL

Split screen ends. Return to overview scene among proliferated SMALL PRE-B CELLS.

IMMATURE B CELL, surface is studded with IgMs. Particle system of a particular COLOR.

PLAYER CONTROL resumes. Player must push through the other cells and continue moving along CXCL12 path. Initial player movement triggers narration.

NARRATOR (V.O.)

With the cell having achieved its IMMATURE B CELL developmental stage, the cell must now check its new sIgM for self-reactivity.

IN-GAME PLAY - SCENE 11 - DODGE DEATH AND LEAVE

Ahead in the bone marrow are stromal cells, but after the narration they have various colors highlighting them with HUD appearing based on proximity. UI header for these cells shows "stromal cell", however the actionable [key] description: "test self-reactivity"

(SPLIT SCREEN HUD close up of the IgM closing in on a surface antigen. Button mash action [key] to get the IgM to stay

© 2016 Emily L. Ling

# Narration Script

15

away from the antigen.)

Button mashing must be within a certain distance away to successfully avoid a:

(1. Strong cross-linking  
self reactivity or 2.  
Weak cross-linking  
self-reactivity)

**\*\* BRANCHING SEQUENCES - 4 \*\***

**\*\* END BRANCHING SEQUENCE - 4 \*\***

© 2016 Emily L. Ling



# Narration Script

Branching Sequences Narration Alpha

Written By

Emily Lunhui Ling  
Masters Candidate, class of 2016

Based on, Janeway's Immunobiology - 8th edition

Johns Hopkins University School of Medicine  
Department of Art as Applied to Medicine  
1830 E. Monument St. Suite 7000  
Baltimore, MD, 21287  
Phone Number

© 2016 Emily L. Ling

# Narration Script

1

## \*\* BRANCHING SEQUENCES - 1 \*\*

HEAVY CHAIN GAME - FAIL TO JOIN D-J

Sequence in the instance the player fails the first attempt at joining the diversity (D) and joining (J) segments.

NARRATOR (V.O.)

The D segments has been exhausted on this chromosome. The D-J segment may be prove successful on the second chromosome.

PLAYER CONTROL returns. Dual input game repeats with new, sequence from the 2nd chromosome's D to J segment.

RESET SCENE HUD:

HEAVY CHAIN GAME - 2ND CHROMOSOME D-J

Second attempt at D-J rearrangement on the heavy chain mini-game. HUD display resets to beginning of SCENE 5 with new title "2nd Chromosome" in header and narration sequence is as follows:

NARRATOR (V.O.)

Rearrangement of the D-J segment occurs simultaneously on both chromosomes. In this instance, it happens that the D-J rearrangement on the first chromosome has failed to produce a proper D-J joining. It falls to the second chromosome then...

PLAYER CONTROL returns. Dual [input] control system to match sides. When aligned properly the two sides will bring up a UI element for a [key] trigger. Event following the [key]:  
(segment joins together with a particle system effect smoke to transition change to shorter gene segment. Removed INTRON dissolves.)

HEAVY CHAIN GAME - 2ND CHROMOSOME - IF SUCCESSFUL

If rearrangement is achieved by player on the second chromosome, continue to "In-game play - scene 6 - v to DJ"

© 2016 Emily L. Ling

# Narration Script

2

of main narration tree.

HEAVY CHAIN GAME - 2ND CHROMOSOME D-J FAIL TO JOIN (2)

Failed segment disappears and game proceeds with shorter chain on the 2nd chromosome.

NARRATOR (V.O.)

A functional heavy-chain has failed to be produced on this EARLY PRO-B CELL...

## END GAME SEQUENCE

Screen goes to to greyscale. Words "Your cell has entered apoptosis, or programmed cell death."

FADE TO BLACK.

(Restart Last Chapter)  
(Return to Main Menu)  
(Quit Game)

**\*\* END BRANCHING SEQUENCE - 1 \*\***

**\*\* BRANCHING SEQUENCES - 2 \*\***

**\*\* END BRANCHING SEQUENCE - 2 \*\***

**\*\* BRANCHING SEQUENCES - 3 \*\***

LIGHT CHAIN GAME - V TO J (KAPPA) FAIL

LIGHT CHAIN GAME - V TO J (KAPPA) FAIL (2)

LIGHT CHAIN GAME - V TO J (LAMBDA) FAIL

LIGHT CHAIN GAME - V TO J (LAMBDA) FAIL (2)

LIGHT CHAIN GAME - 2ND(KAPPA) FAIL

LIGHT CHAIN GAME - 2ND(KAPPA) FAIL (2)

LIGHT CHAIN GAME - 2ND(LAMBDA) FAIL

LIGHT CHAIN GAME - 2ND(LAMBDA) FAIL (2)

© 2016 Emily L. Ling

# Narration Script

3

END GAME SEQUENCE

FADE TO BLACK.

\*\* END BRANCHING SEQUENCE - 3\*\*

\*\* BRANCHING SEQUENCES - 4 \*\*

STRONG CROSS-LINKING

WEAK CROSS-LINKING  
(Anergy)

IMMUNOLOGICAL IGNORANCED

\*\* END BRANCHING SEQUENCE - 4 \*\*

© 2016 Emily L. Ling

## C# Scripts

### Script for adding sound to button interactions

```
using UnityEngine;
using System.Collections;
using UnityEngine.UI;

// this makes sure there is a button attached to this script
[RequireComponent(typeof(Button))]

public class ButtonSound : MonoBehaviour {
    // where we pass in the sound we want to make
    public AudioClip sound;
    // get accessor returns button component on the object
    private Button button { get { return GetComponent<Button>
    (); } }
    // audio source with get access on the audio source component on
    the object
    private AudioSource source { get { return
    GetComponent<AudioSource> (); } }

    void Start () {
        // add an audiosource component to the game object and now
        that it has one we can call the audiosource
        gameObject.AddComponent<AudioSource> ();
        // set the clip to sound and turn off play on awake
        source.clip = sound;
        source.playOnAwake = false;
        // add a listener to play the sound and we call the
        function below which we have named playsound
        button.onClick.AddListener (() => PlaySound ());
    }

    void PlaySound () {
        //
        source.PlayOneShot (sound);
    }
}
```

## C# Scripts

**Script for changing music when a specified scene is loaded**

```
using UnityEngine;
using System.Collections;

public class ChangeMusic : MonoBehaviour {

    public AudioClip level2Music;
    private AudioSource source;

    void Awake ()
    {
        source = GetComponent<AudioSource> ();
    }

    void OnLevelWasLoaded (int level)
    {
        if (level == 2)
        {
            source.clip = level2Music;
            source.Play ();
        }
    }
}
```

## C# Scripts

### Scripts for hovering interaction on main menu buttons

```
using UnityEngine;
using System.Collections;

public class NewGameBtnHover : MonoBehaviour {

    public GameObject particleSystem;

    void Awake (){
        Transform particleSystemTrans = transform.Find
("NewGameBtnActiveParticles");
        particleSystem = particleSystemTrans.gameObject;
        particleSystem.SetActive (false);
    }

    void OnMouseEnter () {
        print ("Button:OnMouseEnter");
        particleSystem.SetActive (true);
    }

    void OnMouseExit () {
        print ("Button:OnMouseExit");
        particleSystem.SetActive (false);
    }

}
```

---

```
using UnityEngine;
using System.Collections;

public class LoadGameBtnHover : MonoBehaviour {

    public GameObject particleSystem;

    void Awake (){
        Transform particleSystemTrans = transform.Find
("LoadGameBtnActiveParticles");
        particleSystem = particleSystemTrans.gameObject;
        particleSystem.SetActive (false);
    }

    void OnMouseEnter () {
        print ("Button:OnMouseEnter");
        particleSystem.SetActive (true);
    }

}
```

## C# Scripts

```
    }  
    void OnMouseExit () {  
        print ("Button:OnMouseExit");  
        particleSystem.SetActive (false);  
    }  
}  
  
-----  
  
using UnityEngine;  
using System.Collections;  
  
public class SettingsGameBtnHover : MonoBehaviour {  
    public GameObject particleSystem;  
    void Awake () {  
        Transform particleSystemTrans = transform.Find  
("SettingsGameBtnActiveParticles");  
        particleSystem = particleSystemTrans.gameObject;  
        particleSystem.SetActive (false);  
    }  
    void OnMouseEnter () {  
        print ("Button:OnMouseEnter");  
        particleSystem.SetActive (true);  
    }  
    void OnMouseExit () {  
        print ("Button:OnMouseExit");  
        particleSystem.SetActive (false);  
    }  
}  
  
-----  
  
using UnityEngine;  
using System.Collections;  
  
public class ContentGameBtnHover : MonoBehaviour {  
    public GameObject particleSystem;  
    void Awake () {  
        Transform particleSystemTrans = transform.Find
```



## C# Scripts

```
    ("ContentGameBtnActiveParticles");
        particleSystem = particleSystemTrans.gameObject;
        particleSystem.SetActive (false);
    }

    void OnMouseEnter () {
        print ("Button:OnMouseEnter");
        particleSystem.SetActive (true);
    }

    void OnMouseExit () {
        print ("Button:OnMouseExit");
        particleSystem.SetActive (false);
    }
}
```

---

```
using UnityEngine;
using System.Collections;

public class ExitGameBtnHover : MonoBehaviour {

    public GameObject particleSystem;

    void Awake (){
        Transform particleSystemTrans = transform.Find
("ExitGameBtnActiveParticles");
        particleSystem = particleSystemTrans.gameObject;
        particleSystem.SetActive (false);
    }

    void OnMouseEnter () {
        print ("Button:OnMouseEnter");
        particleSystem.SetActive (true);
    }

    void OnMouseExit () {
        print ("Button:OnMouseExit");
        particleSystem.SetActive (false);
    }
}
```

## C# Scripts

### Scripts for calling new scene to load

```
using UnityEngine;
using System.Collections;

public class FadeOnClick : MonoBehaviour {

    public void LoadLevel (int sceneBuildIndex) {
        Application.LoadLevel (sceneBuildIndex);
    }
}
```

---

```
using UnityEngine;
using System.Collections;

public class DontDestroy : MonoBehaviour {

    void Awake () {
        DontDestroyOnLoad (gameObject);
    }

}
```

---

```
using UnityEngine;
using System.Collections;

public class LevelToLoad : MonoBehaviour {
    // not for the main menu level loader script
    public string levelToLoad;

    public GameObject background;
    public GameObject text;
    public GameObject progressBar;

    // this will track load progress
    private int loadingProgress = 0;

    // Use this for initialization
    void Start () {
        background.SetActive (false);
        text.SetActive (false);
        progressBar.SetActive (false);
    }
}
```

## C# Scripts

```
    }

    // Update is called once per frame
    void Update ()
    {
        if (Input.GetKeyDown ("space")) {
            StartCoroutine (DisplayLoadingScreen (levelToLoad));
        }
    }
    // this coroutine will accept the string, our level
    /* all coroutines need a yield return statement, it
    will run it until it hits the null and then it passes it back
    to where it was called and then back down until
    it doesnt run anymore*/

    IEnumerator DisplayLoadingScreen(string level){
        background.SetActive (true);
        text.SetActive (true);
        progressBar.SetActive (true);

        progressBar.transform.localScale = new Vector3
        (loadingProgress, progressBar.transform.localScale.y,
        progressBar.transform.localScale.z);

        // concatenate
        // text.guiText = "Loading Progress" + loadingProgress +
        "%";

        // this is what allows the scene to only come in after it
        is done
        AsyncOperation async = Application.LoadLevelAsync (level);

        // to keep updating the loading screen
        while (!async.isDone){
            loadingProgress = (int)(async.progress * 100);
            // text.guiText= "Loading Progress" + loadingProgress
            + "%";

            progressBar.transform.localScale = new Vector3
            (async.progress, progressBar.transform.localScale.y,
            progressBar.transform.localScale.z);

            yield return null;
        }
    }
}
```

7

## C# Scripts

### Script for quit menu on "esc" key down

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections;

public class QuitMenuScript : MonoBehaviour {

    public Canvas quitMenu;
    public Button newText;
    public Button loadText;
    public Button contentText;
    public Button settingsText;
    public Button exitText;

    // Use this for initialization
    void Start () {

        quitMenu = quitMenu.GetComponent<Canvas> ();
        newText = newText.GetComponent<Button> ();
        loadText = loadText.GetComponent<Button> ();
        contentText = contentText.GetComponent<Button> ();
        settingsText = settingsText.GetComponent<Button> ();
        exitText = exitText.GetComponent<Button> ();

        quitMenu.enabled = false;
    }

    void Update () {

        if (Input.GetKeyDown ("escape")) {

            quitMenu.enabled = true;
            newText.enabled = false;
            loadText.enabled = false;
            contentText.enabled = false;
            settingsText.enabled = false;
            exitText.enabled = false;

        }

    }

    public void ExitPress () {

        quitMenu.enabled = true;
    }
}
```

## C# Scripts

```
        newText.enabled = false;
        loadText.enabled = false;
        contentText.enabled = false;
        settingsText.enabled = false;
        exitText.enabled = false;
    }

    public void NoPress(){

        quitMenu.enabled = false;
        newText.enabled = true;
        loadText.enabled = true;
        contentText.enabled = true;
        settingsText.enabled = true;
        exitText.enabled = true;

    }

    public void ExitGame(){

        Application.Quit ();

    }
}
```

## C# Scripts

### Script for scene build manager

```
using UnityEngine;
using System.Collections;
    // this is for code on getting scene switch to
    preserve something you dont want destroyed
public class SceneManager : MonoBehaviour {

    // protect the SceneManager script from being
    destroyed when the new scene loads.
    static SceneManager Instance;

    void Start(){
        // first check if instance is not equal to null,
        else, do something in the else.
        // if the instance exists we want to destroy it
        because we dont want duplicates of the game manager.
        // if the instance is existing currently we need
        to destroy it.
        if (Instance != null) {

            GameObject.Destroy (gameObject);

        } else {
            // if the instance doesnt exist we dont want
            to destroy it on load.
            GameObject.DontDestroyOnLoad (gameObject);
            Instance = this;
            // we are setting it equal to current
            instance.
        }
    }
    void Update(){

        if (Input.GetKeyUp (KeyCode.Keypad1)) {
            // if this key is registered to be coming
            up...
            Application.LoadLevel("scene2");
        }
        if (Input.GetKeyUp (KeyCode.Keypad2)) {
            Application.LoadLevel("scene1");
        }
    }
}
```

10

## C# Scripts

### Script for camera following of player object

```
using UnityEngine;
using System.Collections;

public class TransformFollower : MonoBehaviour
{
    [SerializeField]
    private Transform target;

    [SerializeField]
    private Vector3 offsetPosition;

    [SerializeField]
    private Space offsetPositionSpace = Space.Self;

    [SerializeField]
    private bool lookAt = true;

    private void Update()
    {
        Refresh();
    }

    public void Refresh()
    {
        if(target == null)
        {
            Debug.LogWarning("Missing target ref !", this);

            return;
        }

        // compute position
        if(offsetPositionSpace == Space.Self)
        {
            transform.position =
target.TransformPoint(offsetPosition);
        }
        else
        {
            transform.position = target.position + offsetPosition;
        }

        // compute rotation
        if(lookAt)
        {
            transform.LookAt(target);
        }
    }
}
```

11

## C# Scripts

```
        }  
        else  
        {  
            transform.rotation = target.rotation;  
        }  
    }  
}
```



## REFERENCES

- Bellotti, Francesco, Bill Kapralos, Kiju Lee, Pablo Moreno-Ger, and Riccardo Berta. 2013. Assessment in and of serious games: An overview. Vol. 2013.
- Burnet, F. M. 1959. The Clonal Selection Theory of Acquired Immunity 3D. Cambridge University Press. London.
- Cockburn, Andy. 2004. Revisiting 2D vs 3D implications on spatial memory. Paper presented at Proceedings of the Fifth Conference on Australasian User Interface - Volume 28, Dunedin, New Zealand.
- Coutinho, Antonio. 1989. Beyond clonal selection and network. *Immunological Reviews* 110 (1): 63-88.
- Faust, J. L., and Paulson, D. R. 1998. Active learning in the college classroom. *Journal on Excellence in College Teaching* 9 (2): 3-24.
- Freitas, Sara de, and Tim Neumann. 2009. The use of 'exploratory learning' for supporting immersive learning in virtual environments. *Computers & Education* 52 (2) (2): 343-52.
- Hamari, J., J. Koivisto, and H. Sarsa. 2014. Does gamification work? - A literature review of empirical studies on gamification. Paper presented at 2014 47th Hawaii International Conference on System Science, Hawaii, USA.
- Hoffmann, G. W. 1975. A theory of regulation and self-nonsel discrimination in an immune network. *European Journal of Immunology* 5 (9): 638-47.
- Janeway, C. A. 1989. Approaching the asymptote? evolution and revolution in

- immunology. Cold Spring Harbor Symposia on Quantitative Biology 54 (01/01): 1-13.
- Jerne, N. K. 1971. The somatic generation of immune recognition. *European Journal of Immunology* 1 (1): 1-9.
- Juul, Jesper, Marinka Copier, and Joost Raessens. 2003. The game, the player, the world: Looking for a heart of gameness . Paper presented at In Level Up: Digital Games Research Conference Proceedings, edited by Marinka Copier and Joost Raessens, Utrecht, Utrecht University, <http://www.jesperjuul.net/text/gameplayerworld/>.
- Kaufmann, Stefan H. E. 2007. The contribution of immunology to the rational design of novel antibacterial vaccines. *Nat Rev Micro* 5 (7) (print): 491-504.
- Klopfers, E., Osterweil, S., and Salen, K. 2009. Moving learning games forward. Cambridge, MA: Education Arcade.
- Lederberg, Joshua. 1959. Genes and antibodies. *Science* 129 (3364): 1649-53.
- Lopes, Melissa G. Video gamers' aggression linked to frustration, not violent content. in University of Rochester: Newscenter [database online]. 2014 [cited March 7 2016]. Available from <http://www.rochester.edu/newscenter/frustration-in-mastering-video-games-linked-to-aggression/> (accessed March 7, 2016).
- Mackay, Ian R. 1991. The "Burnet era" of immunology: Origins and influence. *Immunology and Cell Biology* 69 (5) (print): 301-5.
- Murphy, Kenneth, Paul Travers, Mark Walport, and Charles Janeway. 2012. Janeway's immunobiology. 8th ed. New York: Garland Science.
- Nagasawa, Takashi. 2006. Microenvironmental niches in the bone marrow required for B-cell development. *Nat Rev Immunol* 6 (2) (print): 107-16.

- Schunk, Dale H. 2012. Learning theories : An educational perspective. 6th ed. Boston: Pearson.
- Schwartz, Ronald H. 1989. Acquisition of immunologic self-tolerance. *Cell* 57 (7) (6/30): 1073-81.
- Sicart, Miguel. 2008. Defining game mechanics. *The International Journal of Computer Game Research* 8, (2), <http://gamestudies.org/0802/articles/sicart>. (accessed February 25, 2016.)
- Wilson, Greg. February 3, 2006. Off with their HUDs!: Rethinking the heads-up display in console game design *Gamasutra: The Art & Business of Making Games*, [http://www.gamasutra.com/view/feature/130948/off\\_with\\_their\\_huds\\_rethinking\\_.php](http://www.gamasutra.com/view/feature/130948/off_with_their_huds_rethinking_.php) (accessed February 29, 2016).
- Young, Michael F., Stephen Slota, Andrew B. Cutter, Gerard Jalette, Greg Mullin, Benedict Lai, Zeus Simeoni, Matthew Tran, and Mariya Yukhymenko. 2012. Our princess is in another castle a review of trends in serious gaming for education. *Review of Educational Research* 82 (1): 61-89.

## General References

Gibson, Jeremy. 2015. Introduction to game design, prototyping, and development.

Upper Saddle River, NJ: Addison-Wesley.

Kessel, Richard G., and Randy H. Kardon. 1979. Tissues and organs : A text-atlas of scanning electron microscopy. San Francisco: W. H. Freeman.

Lynda.com. [www.lynda.com](http://www.lynda.com)

Maxon. Cineversity. <http://www.cineversity.com/>

Unity 5 3D Technologies. Unity 5 3D Documentation. <http://docs.unity3d.com/Manual/index.html>

Unity3d.com. Tutorials. Creating a Scene Selection Menu. <https://unity3d.com/learn/tutorials/modules/beginner/live-training-archive/creating-a-scene-menu>

Unity3d.com. Tutorials. User Interface (UI). <https://unity3d.com/learn/tutorials/topics/user-interface-ui>

ZBrush Documentation. <http://docs.pixologic.com/>

Axure Software Solutions. <http://www.axure.com/> (accessed March 2014).

Proc3Durale. <http://code.vonc.fr/?a=30> (accessed January 3, 2016).

## Online Educational Video Game Directories

Molecular Jig Games. Find games. ScienceGameCenter. <http://www.sciencegamecenter.org/games> (accessed December 20, 2015).

Nobel Media AB 2016. Nobelprize.org. <http://www.nobelprize.org/educational/medicine/>  
<http://www.nobelprize.org/educational/medicine/immUnity53D/> (accessed December

20, 2015).

Serious Game Classification. <http://serious.gameclassification.com/> (accessed December 20, 2015).

Serious Games Directory Beta. Education. <http://www.seriousgamesdirectory.com/proj/education/> (accessed December 20, 2015).

## Games Referenced

Immune Attack. <http://immuneattack.org/> (accessed December 20, 2015).

ImmuneQuest™. <http://ImmuneQuest.com/> (accessed December 20, 2015).

Immune Responses. <http://www.nobelprize.org/educational/medicine/immuneresponses/> (accessed December 20, 2015).

Never Alone. <http://neveralonegame.com/> (accessed January 1, 2016).

thatgamecompany™. Journey. <http://thatgamecompany.com/games/journey/> (accessed January 1, 2016).

The Immune Defense Game. <http://www.molecularjig.com> (accessed December 20, 2015).

The Immune System. <http://www.nobelprize.org/educational/medicine/immUnity53D/> (accessed December 20, 2015).

The Stanley Parable: a galactic cafe game. <http://www.stanleyparable.com/> (accessed January 1, 2016).

## VITA

Emily Ling was born in West Covina, California on June 5, 1989. She attended Duarte High School until her sophomore year when she transferred to the visual arts program at the Los Angeles County High School for the Arts.

In college, Emily majored in biological sciences at the University of California, Irvine. Although intending to pursue a career in medicine, Emily's interest in art persisted. This lead her to working at the university's campus newspaper as a layout intern, editor, and comic artist. She graduated *cum laude* and Phi Beta Kappa in 2011 with a Bachelor of Science in Biological Sciences and a Minor in Chinese Language and Literature.

She learned of biomedical illustration when an acquaintance commissioned her to create a scientific diagram. After honing her drafting skills at the Concept Design Academy in Pasadena, California, Emily applied to the Medical and Biological Illustration program at the Johns Hopkins University School of Medicine and matriculated in the fall of 2014.

During her graduate studies, Emily was supported by the William P. Didusch and Gwynne M. Gloege Scholarships. She earned a Salon Gold Award at the O. A. Parkes Symposium and Student Conference and was recognized by the Association of Medical Illustrators for her graduate work in pen and ink media.

Emily sees biomedical communication as a combination of her skills and interests, and as a means of contributing to society. She will receive her Master of Arts in Medical and Biological Illustration in May of 2016 and looks forward to a future creating engaging educational media for the public.