



Electronics and Computer Engineering Dept.

Technical University of Crete

An Interactive 3D Lighting System for fMRI Rendering Fidelity Experiments

Christodoulou Ioannis

*A thesis submitted in fulfillment of the requirements for the degree of Master of Science in
Electronic and Computer Engineering.*

Department of Electronic and Computer Engineering
Laboratory of Distributed Multimedia Information Systems and Applications – *TUC/MUSIC*

CHANIA 2012

Abstract

Although improvements in basic computer graphics rendering hardware and lighting algorithms have produced some remarkable results, it is still computationally demanding to render a highly realistic Virtual Environment (VE) in real-time. The “sense of being there” or presence in the situation depicted by the VE display systems is, lately, linked with behavioural fidelity, or the extent to which someone acts in a simulation, such as in a flight simulator as if real. The ultimate goal of the presence research could be explained as finding the equation of presence that allows to trade off realistic rendering against render speed, while still maintaining the same level of presence.

This thesis presents a real-time synthetic lighting system incorporating sophisticated global illumination algorithms aiming to induce similar subjective lighting impressions as in the real world. The system will be used as the main platform for cognitive and neuroscientific experimental paradigms related to simulator fidelity explorations. The lighting system proposed is designed to render an interactive VE on an fMRI display, enabling the conduct of experiments, researching the effects of visual fidelity on feelings of presence and impressions of lighting. Feelings of presence were explored by responses to a questionnaire, as well as examining the brain activity and the heart rate during the navigation in the VEs. This study allowed the researchers to investigate the participants’ neurocorrelates of fidelity.

Ultimately, the goal of this project is to use this system to explore the effect of lighting variations (daylight vs forms of artificial light) in relation to a group of patients suffering from the ‘depersonalization’ syndrome. Anecdotal evidence has shown that lighting affects the patients’ mood in varied ways. Neurocorrelates of participants’ actions assess the fidelity of a simulation while being immersed in the synthetic scenes, instead of previously utilized self-report of fidelity or task performance, after the task has occurred. The ultimate goal of the experiments planned is to explore whether natural and artificial scenes of varied fidelity for training or for therapeutic purposes engage common perceptual or neuroscientific mechanisms. Such input is non-obtrusive and is derived at the same time as the experience occurs.

The system was developed in close collaboration with the Brighton and Sussex Medical School in the UK and the brain imaging experiments were conducted in the UK. The results from the participants’ responses during the experiment indicated that the exposure to the photorealistically-rendered indoors synthetic scenes generated a significantly higher feeling of presence in the VEs, rather than the exposure to the wireframe indoors synthetic scenes. Participants felt significantly more depersonalized while navigating the evening and afternoon indoors wireframe scenes, than the respective photorealistically-rendered ones, as well as after exposure to the morning outdoors wireframe scene, than after exposure to the equivalent photorealistic one.

The results indicated that participants felt the environment to be more comfortable after exposure to photorealistically-rendered scenes, especially during midday, afternoon and evening. Interestingly, there were no statistically significant differences in relation to perceived comfort after exposure to the morning photorealistically-rendered and wireframe indoors scenes.

Abstract

Preliminary results from the analysis on brain image data acquired during the first stage of the experiments revealed that there is a greater activation in regions previously implicated in “spatial navigation”, including navigation in VEs, after exposure to the indoors photorealistically-rendered scenes compared to the respective wireframe scenes. Examples of these regions, which are related to spatial navigation, are the superior frontal gyrus, the posterior cingulate and the cerebellum. This would suggest that in the realistic indoors virtual scenes, participants were practically “navigating” and exploring the environment.

Declaration

Declaration

The work in this thesis is original and no portion of the work referred to here has been submitted in support of an application for another degree or qualification of this or any other university or institution of learning.

Signed:

Date:

Christodoulou Ioannis

Acknowledgements

Σε αυτή τη σελίδα θα κάνω μια παρασπονδία και θα χρησιμοποιήσω την ελληνική γλώσσα, σε αντίθεση με την υπόλοιπη εργασία. Έτσι γλιτώνω και τις απανωτές διορθώσεις της κ. Μανιά (η οποία για έναν περίεργο λόγο μισεί τις παρενθέσεις, ούτε μία δεν μου άφησε).

Αρχικά οφείλω να ευχαριστήσω ιδιαίτερα την επιβλέπουσα καθηγήτριά μου κ. Κατερίνα Μανιά για την ευκαιρία που μου έδωσε να ασχοληθώ με ένα τόσο ενδιαφέρον και πρωτότυπο θέμα, τη συνεχή υποστήριξη, γενικότερη συμπαράσταση, καθοδήγηση, επίβλεψη και την άψογη συνεργασία. Εκτιμώ ιδιαίτερα την θέληση και την διάθεση που είχε να μου συμπαρασταθεί και να με βοηθήσει, ακόμα και εξ αποστάσεως, όταν ήμουν στο Brighton. Να 'ναι καλά το Skype ☺. Ζητώ συγνώμη για το άγχος και την κούραση που της προκάλεσα. Είμαι πραγματικά χαρούμενος που συνεργάστηκα με την κ. Μανιά και η στάση της θα αποτελεί για μένα πηγή έμπνευσης!

Θα ήθελα επίσης να ευχαριστήσω και τους συνεργάτες μας στην Αγγλία, τον Prof. Hugo Critchley, τον Dr. Nick Medford και, φυσικά, την Dr. Eugenia Radulescu. Χωρίς αυτούς δεν θα ήταν εφικτή αυτή η εργασία, με την πολύτιμη εμπειρία τους, τις συμβουλές τους (και τον εξοπλισμό τους). Τους ευχαριστώ για το ενδιαφέρον που επέδειξαν και για τον κόπο και χρόνο που κατέβαλλαν, ειδικά τον 1.5 μήνα που ήμουν στον Brighton, παρά το βαρύ πρόγραμμά τους, ώστε να ξεπεραστούν οι τεχνικές δυσκολίες των πειραμάτων σε σύντομο χρονικό διάστημα και να διεξαχθούν κανονικά.

Οφείλω να ευχαριστήσω και τον κ. Χριστοδουλάκη για την εμπειρία που μου προσέφερε όλα αυτά τα χρόνια κατά την διάρκεια της εργασίας μου στο MUSIC. Επίσης ευχαριστώ τον κ. Λαγουδάκη και τον κ. Σαμολαδά για τον χρόνο που θα αφιερώσουν στην ανάγνωση αυτής της εργασίας, παρά τα στενά χρονικά πλαίσια. Δεν μπορώ να μην ευχαριστήσω και όλα τα μέλη του εργαστηρίου, τον Αλέξανδρο, τον Γιώργο (που είναι και σειρά τώρα ☺), τον Μανώλη και τον άλλο Γιώργο για την αλληλεγγύη που υπήρχε (ιδιαίτερα στα πλαίσια του Pattern Recognition) και για τους καφέδες που πίναμε παρέα.

Ταυτόχρονα ευχαριστώ θερμά και όλους τους φίλους μου στα Χανιά (και όχι μόνο), για τις ωραίες στιγμές που περάσαμε μαζί, χωρίς τις οποίες δεν θα είχα κουράγιο να ολοκληρώσω αυτό το μεταπτυχιακό. Γιάννη (συγκάτοικος), Σοσάρα (Dirty Juicy), Κωστή (Γιόχαν), Ασημίνα και όλοι οι υπόλοιποι (ζητώ συγνώμη, αλλά αν έγραφα όλα τα ονόματα, οι ευχαριστίες θα έβγαιναν μεγαλύτερες από το master!) σας εύχομαι να είστε καλά!

Φυσικά πρέπει να ευχαριστήσω τους γονείς μου, Φίλιππο και Σοφία, για τη συμπαράσταση, πίστη (;), υπομονή και υποστήριξη όλο αυτό το διάστημα. Αφού άντεξα τέτοιο μεταπτυχιακό, το στρατό και τον Έβρο θα φοβηθώ; Επίσης, ευχαριστώ τον Δημήτρη (Big Bro) και την Ερσίλια (βρε κοπέλα μου, αυτό το όνομα γραπτώς είναι ακόμα πιο περίεργο από ό,τι προφορικώς) για την ηθική και ψυχολογική συμπαράστασή τους.

Για το τέλος κράτησα το καλύτερο, τη Μαρία ή το Kuku μου για την ψυχολογική υποστήριξη σε όλα τα προβλήματα που συνάντησα, γιατί ό,τι πρόβλημα κι αν είχα, είχε έτοιμη την απάντηση «μην αγχώνεσαι, όλα θα πάνε καλά». Σε ευχαριστώ που ήσουν μαζί μου στα ευχάριστα και μου συμπαραστάθηκες όποτε το χρειάστηκα! Εύχομαι τα καλύτερα να είναι ακόμα μπροστά μας! Αφιερωμένο στο Kuku λοιπόν...

Publications

- Mania, K., Medford, N., Christodoulou, I., Watten, P., Rivera, F., Radulescu, E., Critchley, H. (2011). Exploring Behavioural Fidelity of Synthetic Stimuli while Immersed in fMRI Displays. *VSGames 2011 (IEEE sponsored)*, 215-219.
- Christodoulou, I., Mania, K., Watten, P., Radulescu, E., Critchley, H., Medford, N. (invited as one of best papers of IEEE sponsored VSGames 2011, submitted, 2012). A 3D lighting system for fMRI rendering fidelity experiments. *International Journal of Interactive Worlds*, IBIMA Publishing.

Table of Contents

Abstract	2
Declaration	4
Acknowledgements	5
Publications	6
Table of Contents	7
List of Figures	10
List of Tables	15
1 Chapter 1 – Introduction.....	17
1.1 Contribution.....	20
1.2 Thesis Outline	22
2 Chapter 2 – Technical Background	23
2.1 Computer Graphics Rendering	23
2.1.1 The Physical Behavior of the Light.....	24
2.1.2 Computer Graphics Illumination Models.....	26
2.1.3 Ray Tracing	29
2.1.4 Radiosity	30
2.2 Virtual Reality Technology and Game Engines	34
2.2.1 Unity 3D.....	34
2.2.2 Torque 3D	35
2.2.3 Unreal Engine 3 – Unreal Development Kit (UDK).....	35
2.3 Subjective Impressions of Lighting and Color Effects.....	35
2.3.1 Introduction.....	35
2.3.2 Effects of Lighting Conditions on Feelings of Presence and Attention	37
2.3.3 Effects of Lighting Variations on Task Performance	37
2.3.4 Effects of Lighting on Mood and Emotion	38
2.3.5 Effects of Lighting on Mood of Depersonalization Syndrome Patients	39
2.3.6 Effects of Color on Space Perception and Subjective Impressions.....	40
2.4 Perceptual Fidelity and Presence in Virtual Environments	41
2.4.1 Fidelity Metrics for Computer Graphics Simulations.....	41
2.4.2 Perceived Presence in a Virtual Environment	42
3 Chapter 3 – Software Architecture and Development Framework	45
3.1 Game Engine – Unreal Development Kit (UDK)	45
3.1.1 Unreal Editor.....	45

Table of Contents

3.1.2	Sound Engine	50
3.1.3	Configuration Files	51
3.1.4	DLL Files	52
3.1.5	Input Manager	52
3.1.6	Lighting and Rendering Engine	53
3.1.7	Unreal Lightmass	54
3.2	UnrealScript	55
3.2.1	The Unreal Virtual Machine	55
3.2.2	Object Hierarchy	56
3.2.3	Timers	57
3.2.4	States	58
3.2.5	Delegates	58
3.2.6	Interfaces	58
3.2.7	UnrealScript Compiler	59
3.2.8	UnrealScript Programming Strategy	59
3.3	Flash Applications as User Interfaces (UI)	59
3.3.1	Authoring Environment for Interactive Content	60
3.3.2	ActionScript 2.0	61
3.3.3	Connection of the User Interface (UI) with the Application	62
4	Chapter 4 – Implementation	63
4.1	Creating the 3D Virtual Scenes	63
4.1.1	Creating the Visual Content	63
4.1.2	Setting up the Scene in UDK	67
4.2	Lighting Configurations	74
4.2.1	Morning Lighting Configuration	74
4.2.2	Midday Lighting Configuration	75
4.2.3	Afternoon Lighting Configuration	76
4.2.4	Evening Lighting Configuration	77
4.3	UnrealScript Classes	80
4.4	Handling User Input	83
4.5	Handling the Stages of the Experiment Using States	87
4.6	Logging of Events and Participants' Actions Mechanism	89
4.7	Time Limits Control	93
4.8	Emotional images slideshow	94
4.9	Manipulation of the indoors artificial light	96
4.10	Synchronization with the fMRI Scanner	98

5	Chapter 5 – UI Implementation	99
5.1	Application menus as Flash UIs	99
5.2	Questionnaires designed as Flash UIs	101
5.3	Light Control Panel as Flash UI	104
6	Chapter 6 – Experiments	105
6.1	Materials	105
6.1.1	Participants	105
6.1.2	Apparatus	105
6.1.3	Visual Content	106
6.2	Methods	106
6.2.1	Experimental Procedures	106
6.2.2	Experimental Setup	107
6.2.3	Training	108
6.2.4	First stage	110
6.2.5	Second stage	111
6.2.6	Third stage	112
6.3	Statistical Analysis	113
6.3.1	Repeated-Measures Generalized Linear Model (GLM)	113
6.3.2	Wilcoxon signed-rank test	114
6.3.3	Friedman test	114
6.3.4	Mann-Whitney Test	114
6.4	Results	114
6.4.1	First Stage Results	115
6.4.2	First Stage Brain Imaging Results	130
6.4.3	Second Stage Results	132
6.4.4	Third Stage Results	137
6.4.5	Third Stage Brain Imaging Results	150
6.5	Comments by Experiment Participants	157
6.6	Discussion	158
7	Chapter 7 – Conclusions and Future Work	161
7.1	Main contributions	161
7.2	Implications for Future Work	163
8	References – Bibliography	164

List of Figures

Figure 1: Sample of computer-generated photo-realistic images.	17
Figure 2: The goal of realistic image synthesis: an example from photography.	23
Figure 3: The visible portion of the electromagnetic spectrum.....	24
Figure 4: Light transmitted through a material.....	25
Figure 5: Light absorbed by a material.....	25
Figure 6: Light refracted through a material.....	25
Figure 7: Light reflected off a material in different ways. From left to right, specular, diffuse, mixed, retro-reflection and finally gloss (Katedros 2004).....	26
Figure 8: The differences between a simple computer generated polyhedral cone (left), linearly interpolated shading to give appearance of curvature (Gouraud Shading). Note Mach bands at edges of faces (middle) and a more complex shading calculation, interpolating curved surface normals (Phong Shading). This is necessary to eliminate Mach Bands (right).	27
Figure 9: Graphical Depiction of the rendering equation (Yee 2000).	28
Figure 10: Ray-tracing.	30
Figure 11: Radiosity (McNamara 2000).....	30
Figure 12: Relationship between two patches (Katedros 2004).	31
Figure 13: Nusselt's analog. The form factor from the differential area dA_i to element A_j is proportional to the area of the double projection onto the base of the hemisphere (Nusselt 1928).	32
Figure 14: The hemicube (Langbein 2004).	32
Figure 15: The difference in image quality between ray tracing (middle) and radiosity (right hand image).....	33
Figure 16: Architecture of a game engine.....	34
Figure 17: The Unreal Editor displaying a synthetic scene.	46
Figure 18: Static Mesh Editor for an imported object.....	47
Figure 19: Properties of a Static Mesh Actor instance.....	47
Figure 20: Light Actor properties: for a <i>PointLight Actor</i> , simulating the indoors artificial light (top) and for a <i>DominantDirectionalLight Actor</i> , simulating the midday sun (bottom).	48
Figure 21: Unreal Kismet set up for the first stage of the experiment.	49
Figure 22: The Material Editor with a sample material loaded.	50
Figure 23: UDK's Sound Editor and a sound imported into a scene.	51
Figure 24: Class Hierarchy Diagram for 3 user-created classes: <i>VRExperimentGame</i> , <i>VRExperimentPawn</i> and <i>VRExperimentPlayerController</i>	57
Figure 25: Flash Authoring environment displaying a Flash User Interface.	60
Figure 26: Initiation of a Flash User Interface Application through Unreal Kismet.	62
Figure 27: A 3D object representation of a bench. The 3D object was created in 3ds Max and its geometry exported to UDK.	64
Figure 28: The UVW mapping of an object in 3ds Max. This UV channel is used by UDK in order to apply a material correctly onto the object.	65
Figure 29: The UVW unwrapping of an object in 3ds Max. This UV channel is used by UDK in order to realistically illuminate an object, assigning more shading detail in the larger polygons of the object in the channel.	66

List of Figures

Figure 30: The final scene with all the 3D objects created in 3ds Max.....	66
Figure 31: The imported in UDK geometry of the bench 3D object, with a simple collision vector applied on it.	67
Figure 32: On the left is a grey matte material, with no specular color. On the right is the same grey material with white specular color.	68
Figure 33: A material for the surface of a wooden table with a normal map defined. As can be seen from the sample sphere with the material applied on it, the lighting effects seem as though falling on a rough surface.	68
Figure 34: A table surface object with the normal mapped material applied on it. Although the table's geometry is simple, with only 12 triangles, the lighting effects give the illusion of a rough (and complex) surface.	69
Figure 35: A translucent material representing the glass in a window.	70
Figure 36: An unlit scene with the modeled 3D objects in UDK. The created materials are applied to the respective objects.....	70
Figure 37: The wireframe version of the material for the wooden surface of a table.	71
Figure 38: The original realistic material applied to the tiled floor in the yard.	72
Figure 39: The non-realistic material applied on the tiled floor in the yard in the wireframe scenes. The diffuse texture is replaced by its average color. The normal maps are also removed.	73
Figure 40: The unlit wireframe version of the original scene in UDK.	73
Figure 41: Morning sun light configuration. The most important settings are <i>Indirect Lighting Scale</i> and <i>Saturation, Brightness</i> and <i>Light Color</i> (as an RGB value).	74
Figure 42: Screenshot of the Morning Outdoors Scene (left) and the respective wireframe version (right).....	75
Figure 43: Screenshot of the Morning Indoors Scene (left) and the respective wireframe version (right).....	75
Figure 44: Midday lighting configuration.....	75
Figure 45: Screenshot of the Midday Outdoors Scene (left) and the respective wireframe version (right).....	76
Figure 46: Screenshots of the Midday Indoors Scene (left) and the respective wireframe version (right).....	76
Figure 47: Afternoon sun configuration.....	76
Figure 48: Screenshots from the Afternoon Outdoors Scene (left) and the respective wireframe version (right).....	77
Figure 49: Screenshots from the Afternoon Indoors Scene (left) and the respective wireframe version (right).....	77
Figure 50: Evening Indoors Artificial light settings. The most important options are the <i>Indirect Lighting Scale, Indirect Lighting Saturation, Light Source Radius, Brightness</i> and <i>Light Color</i> (in RGB value).	78
Figure 51: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the <i>Standard Fluorescent</i> artificial light type applied.	79
Figure 52: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the <i>40 Watt Incandescent Tungsten</i> artificial light type applied.	79
Figure 53: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the <i>100 Watt Incandescent Tungsten</i> artificial light type applied.	79

List of Figures

Figure 54: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the <i>Halogen</i> artificial light type applied.	79
Figure 55: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the <i>Carbon Arc</i> artificial light type applied.....	80
Figure 56: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the <i>Cool White Fluorescent</i> artificial light type applied.	80
Figure 57: The zones into which the yard was divided. The triggers in the center of each zone are highlighted.	91
Figure 58: The trigger's "Touch" generated event evokes the TrackMovement method in the controller class.	91
Figure 59: When the index 1 of <i>ShowQuestionnaires</i> event is activated, its <i>Show Questionnaires</i> slot (the second) would be activated, activating the <i>Open Gfx Movie</i> action in turn.....	94
Figure 60: The Pleasant 0 index of the Change TV Screen event was connected to an action that changed the TV screen to the respective emotional image.	95
Figure 61: Sample part of the Unreal Kismet sequence for the emotional image slideshow in the 2nd stage of the experiment.	95
Figure 62: Screenshot of the Unreal Kismet sequence for the manipulation of the artificial indoors light.	96
Figure 63: The Kismet sequence responsible of changing the artificial indoors light type, during the third experimental stage. When the "Warm Fluorescent" index of the "Change Light Color" event was activated, the Matinee that was responsible to change the artificial indoors light color to "Warm Fluorescent" was activated in turn.	97
Figure 64: The Kismet sequence responsible to handle participant's input to change the displayed light's brightness. When the participant moved the cursor to alter the brightness, the <i>Lever Moved</i> event was activated, triggering the <i>Change Brightness</i> action, with the new brightness value passed as a parameter (the blue circle connected to both the event and the action).	97
Figure 65: A sample screenshot of the spike recording application. In the first line, each black bar signifies a new brain image acquired from the scanner. In the second line, each green bar is a sound sync pulse sent from the application. The heart pulse and the pulse oximetry signals were omitted.....	98
Figure 66: A Flash UI questionnaire being displayed on top of the currently rendered virtual scene.	99
Figure 67: The start Flash menu that was displayed when the experiment application was started.....	100
Figure 68: The start menu Flash UI is loaded and displayed, immediately after the virtual scene becomes visible.	101
Figure 69: The Flash UI questionnaire that was displayed during the second experimental stage.....	103
Figure 70: When the " <i>Got Ratings</i> " event is activated, it activates the Close Gfx Movie action, which stops UDK's Flash player from displaying the Flash UI.	104
Figure 71: The light control panel Flash UI.....	104
Figure 72: Photo of the Current Designs HHSC-2x4-C response pad used in the experiments.	106
Figure 73: The experimental setup.	107

List of Figures

Figure 74: The fMRI scanner inside which the experiments were conducted. The participants' heads were placed inside the coil that can be seen in the fMRI scanner. Through a mirror on top of the coil, the participants could see the VEs displayed on the projector screen behind them.	108
Figure 75: Screenshot of the training synthetic scene in relation to the first stage of the experiment.	109
Figure 76: Screenshot of the training synthetic scene in relation to the second stage of the experiment.	109
Figure 77: Screenshot of the training synthetic scene in relation to the third stage of the experiment.	110
Figure 78: Sample screenshot of the Midday Indoors scene with a pleasant emotional image displayed.	111
Figure 79: Sample screenshot of the Midday Indoors scene with an unpleasant emotional image displayed.	112
Figure 80: The first question of the questionnaire presented to participants after their free navigation in each virtual scene of the first stage of the experiment.	115
Figure 81: Estimated Marginal Means for Question 1 presented during the first stage of the experiments.	116
Figure 82: Configuration for the Wilcoxon test for the Indoors Locations.	117
Figure 83: The Wilcoxon test configuration for the participants' responses in the questions, after navigating the outdoor synthetic scenes.	118
Figure 84: The second question of the questionnaire presented to participants after their free navigation in each synthetic scene during the first stage of the experiment.	119
Figure 85: Estimated Marginal Means for Question 2 presented during the first stage of the experiments.	121
Figure 86: The third question of the questionnaire presented to participants after their free navigation in each synthetic scene during the first stage of the experiment.	123
Figure 87: Estimated Marginal Means for Question 3 presented during the first stage of the experiments.	124
Figure 88: The fourth question of the questionnaire presented to participants after their free navigation in each synthetic scene during the first stage of the experiment.	126
Figure 89: Estimated Marginal Means for Question 4 presented during the first stage of the experiments.	128
Figure 90: Brain images acquired from the fMRI scanner during the first stage of the experiment. This image shows midline regions like superior frontal gyrus and posterior cingulate.	131
Figure 91: Brain images acquired from the fMRI scanner during the first stage of the experiment. This image shows the cerebellum and the superior frontal gyrus cluster.	131
Figure 92: The question shown after each emotional image was displayed, probing for the participant's rating of the image.	132
Figure 93: Estimated Marginal Means of the emotional image ratings in the indoors virtual scenes.	133
Figure 94: Estimated Marginal Means of the emotional image ratings during exposure to the outdoors synthetic scenes.	134

List of Figures

Figure 95: Estimated Marginal Means for the first question during the third stage of the experiment. 138

Figure 96: Estimated Marginal Means for the third question during the third stage of the experiment. 141

Figure 97: Estimated Marginal Means for the fourth question during the third stage of the experiment. 144

Figure 98: Design Matrix for Brain Imaging. 151

Figure 99: SPM software, selection of contrast. 152

Figure 100: Contrast weights. 152

Figure 101: The 6x2x2 matrix used in the GLM test. 153

Figure 102: Brain Imaging Analysis results for the third experimental stage. This is an overview of the contrast. 154

Figure 103: Brain Imaging Analysis results for the third experimental stage. The brain regions are highlighted. 154

Figure 104: The 6x2 matrix used in the GLM test. The two factors are the "reality" (photorealistic vs. wireframe) and the available light types (6 light types). The Z scores of participants' responses to the presence question were used as a covariate and they are shown in the last six columns (in lighter gray). 155

Figure 105: Contrast based on GLM. 156

Figure 106: Brain Imaging Analysis Results. For this analysis, the scores of participants' responses to the fourth question presented to them during the third experimental stage, which investigated participants' perceived feeling of presence, were introduced as a covariate interacting with the light type applied in the synthetic scene. 157

List of Tables

Table 1: Objects existing in the indoors room and their respective counterparts in the yard. . .	64
Table 2: Artificial light types and their respective RGB color value.	78
Table 3: Configuration of the repeated-measures GLM test for the first question presented during the first stage of the experiment.....	115
Table 4: Descriptive Statistics of Question 1 presented during the first stage of the experiments.	116
Table 5: Descriptive Statistics for Question 1 presented during the indoors virtual scenes. WF stands for Wireframe.	117
Table 6: Wilcoxon test statistics for Question 1 presented during the indoors virtual scenes.	118
Table 7: Descriptive Statistics for Question 1 presented during the outdoors virtual scenes..	119
Table 8: Test Statistics for Question 1 presented during the outdoors virtual scenes.	119
Table 9: Configuration of the repeated-measures GLM test for the second question presented during the first stage of the experiment.....	120
Table 10: Descriptive Statistics for Question 2 presented during the first stage of the experiments.	120
Table 11: Descriptive Statistics for Question 2 presented during the indoors synthetic scenes.	122
Table 12: Test Statistics for Question 2 presented during the indoors synthetic scenes.	122
Table 13: Descriptive Statistics for Question 2 presented during the outdoors synthetic scenes.	122
Table 14: Test Statistics for Question 2 presented during the outdoors synthetic scenes.	123
Table 15: Configuration of the repeated-measures GLM test for the third question presented during the first stage of the experiment.....	123
Table 16: Descriptive Statistics of Question 3 presented during the first stage of the experiments.	124
Table 17: Descriptive Statistics for Question 3 presented during the indoors virtual scenes..	125
Table 18: Test Statistics for Question 3 presented during the indoors virtual scenes.	125
Table 19: Descriptive Statistics for Question 3 presented during the outdoors virtual scenes.	126
Table 20: Test Statistics for Question 3 presented during the outdoors virtual scenes.	126
Table 21: Configuration of the repeated-measures GLM test for the fourth question presented during the first stage of the experiment.....	127
Table 22: Descriptive Statistics of Question 4 presented during the first stage of the experiments.	127
Table 23: Descriptive Statistics for Question 4 presented during the indoors virtual scenes..	129
Table 24: Test Statistics for Question 4 presented during the indoors virtual scenes.	129
Table 25: Descriptive Statistics for Question 4 presented during the outdoors virtual scenes.	129
Table 26: Test Statistics for Question 4 presented during the outdoors virtual scenes.	130
Table 27: Descriptive statistics of the emotional image ratings in the indoors virtual scenes.	132
Table 28: Descriptive Statistics of the emotional image ratings after viewing the outdoors virtual scenes.....	133

List of Tables

Table 29: Descriptive Statistics for the Wilcoxon test on the emotional image ratings during exposure to the specified indoors synthetic scenes.	135
Table 30: Wilcoxon test statistics on the emotional image ratings during exposure to the specified indoors synthetic scenes.....	135
Table 31: Descriptive statistics of Wilcoxon test on the emotional image ratings during exposure to the specified outdoors scenes.	135
Table 32: Wilcoxon test statistics on the emotional image ratings during exposure to the specified outdoors synthetic scenes.	136
Table 33: Descriptive statistics of the Wilcoxon test on the unpleasant emotional image ratings during exposure to the morning, midday and afternoon outdoors synthetic scenes.	136
Table 34: Wilcoxon test statistics on the unpleasant emotional images during exposure to the morning, midday and afternoon outdoors synthetic scenes.....	136
Table 35: Descriptive statistics for the GLM test for the first question during the third stage of the experiment.....	137
Table 36: Wilcoxon test statistics for the first question during the third stage of the experiment.	138
Table 37: Descriptive statistics and Friedman test results for the first question after exposure to the photorealistic synthetic scene during the third stage of the experiment.....	139
Table 38: Descriptive statistics and Friedman test results for the first question after exposure to the wireframe synthetic scene during the third stage of the experiment.	140
Table 39: Descriptive statistics for the GLM test for the third question during the third stage of the experiment.....	140
Table 40: Descriptive statistics and Wilcoxon test results for the third question during the third stage of the experiment.	142
Table 41: Descriptive statistics and Friedman test results for the third question after exposure to the photorealistic synthetic scene during the third stage of the experiment.....	143
Table 42: Descriptive statistics and Friedman test results for the third question after exposure to the wireframe synthetic scene during the third stage of the experiment.	143
Table 43: Descriptive statistics for the fourth question during the third stage of the experiment.	144
Table 44: Descriptive statistics and Wilcoxon test results for the fourth question during the third stage of the experiment.	145
Table 45: Descriptive statistics and Friedman test results for the fourth question after exposure to the photorealistic synthetic scene during the third stage of the experiment.....	146
Table 46: Descriptive statistics and Friedman test results for the fourth question during the third stage of the experiment.	146
Table 47: Participants' responses for their most comfortable brightness value for each of the displayed artificial light types during the third stage of the experiment. WF stands for wireframe.....	147
Table 48: Behavioural results for the first and the third experimental stages.	149

1 Chapter 1 – Introduction

Virtual Environments (VEs) allow us to explore an ancient historical site, visit a new home led by a virtual estate agent, or fly through the twisting corridors of a space station in pursuit of alien prey. They simulate the visual experience of immersion in a 3D environment by rendering images of a computer model as seen from an observer viewpoint moving under interactive control by the user. If the rendered images are visually compelling, and they are refreshed quickly enough, the user feels a sense of presence in a virtual world, enabling applications in education, training simulation, computer-aided design, electronic commerce, entertainment and medicine.

The huge growth in computer hardware has in turn led to the rapid development of computer graphics (CG) in the last 50 years, as well as to the increased complexity and speed of software algorithms. Computer graphics can, nowadays, render highly realistic images of complex scenes in ever decreasing amounts of time. The realism of rendered scenes has been increased by the development of lighting algorithms, which can simulate the lighting effects produced in the real world. These methods ensure, to a high degree, the physical accuracy of the images produced and give what is known as a photo-realistic rendering (Ward Larson and Shakespeare 1997).

The images shown in Figure 1 help demonstrate the power of computer graphics to generate images that can mislead into believing they are actual photographs.



Figure 1: Sample of computer-generated photo-realistic images.

The greatest issue in computer graphics that needs to be addressed is how to achieve the rendering of these photo-realistic images at real-time frame rates. On modern computers, creating a single image, such as these, may take a huge amount of computational time. Although improvements in basic rendering hardware and lighting algorithms have produced some remarkable results, it is still computationally demanding to render a highly realistic VE in real-time, especially in relation to specular effects (Chalmers and Cater 2002). Rendering lighting propagation following the physics of the real-world as well as comparing the subjective feelings of lighting produced in the real-world and in simulations, is still a challenge for computer graphics.

A key issue in the design of VEs, then, is to tailor visual fidelity to fit the needs of the application. The term visual fidelity refers to the degree to which visual features in the VE conform to visual features in the real environment (Mania et al 2005). Visual fidelity mediates the mapping from the real world environment to a computer generated one (Waller et al 1998).

The concept of presence in a virtual environment (VE) naturally raises the question of the meaning of ‘presence’ with respect to real world experiences. The feeling of presence in a VE could be described as “the sense of being there” in the situation depicted by the VE display systems (Barfield & Weghorst 1993, Slater & Usoh 1998). Lately, the definition of presence is linked with behavioural fidelity or the extent to which someone acts in a simulation, such as in a flight simulator as if real (Slater, 2004). The ultimate goal of the presence research could be explained as finding the equation of presence that allows to trade off realistic rendering against render speed, while still maintaining the same level of presence.

It has been suggested that presence is important, because it is tantamount to a “common currency” for VE applications—the one result of a VE experience that may be universally independent of application and other aspects such as “task performance.” Hence, the idea of trying to discover how to “maximize presence” is thought to be a useful goal of VE research—especially since presence is likely to be associated with behavior that is appropriate to the situation. In this case it would be useful for training—where the experience in the VE should be as close as possible to the experience in the corresponding real world situation, so that whatever is learned in the VE is transferred positively to appropriate behavior in the real world (Slater 2004). Positive transfer means that the trainee conducts the learned tasks better in the real-world compared to task performance in a simulator.

In order to measure presence, the normal approach is to use questionnaires, with questions relating to the “sense of being there” in the displayed environment. However, it has been argued that sole reliance on questionnaire results cannot be used to verify that “presence” actually exists as a phenomenon (Slater 2004). In this study, Slater suggested that presence measurements with questionnaires should be accompanied with physiological or brain activity data.

There were studies in psychotherapy that used VEs, which demonstrated that presence exists, although there is no scientific evidence. These studies used anxiety as a surrogate for presence, such as fear of heights, fear of flying, arachnophobia, agoraphobia and burns treatment (Emmelkamp et al 2002; Rothbaum et al 1996; Carlin et al 1997; Botella et al 2007

and Hoffman et al 2008). Apart from indicating that presence exists, these studies also provided evidence that VEs can be used as an effective treatment in confronting the aforementioned fears and illnesses. Measuring ‘presence’ or better, behavioural fidelity of a simulation remains an open research question.

The work of this thesis was performed in collaboration with the Neuroscience & Psychiatry sectors of the Brighton and Sussex Medical School in the UK. The author of this thesis visited the collaborating Brighton and Sussex Medical School twice and spent a total of around three months in the UK in two separate timings. The first one was in order to gather requirements and test the primitive prototype of the system. During his last visit, he was responsible for installing the system, testing the UI and button boxes utilized inside the scanner and running the complete experimental study.

The general aim of the work presented was to explore changes in regional brain activity associated with changes in the sense of ‘presence’ and subjective feelings associated with lighting when immersed in simulations of varied lighting configurations and rendering quality and thus discover more about the neural circuitry that supports such feelings. In this manner, it is endeavored to devise behavioural fidelity metrics of simulations based on neural activity. Moreover, another goal was to examine the interaction between changes in the sense of presence and the neural response to, and subjective experience of, emotional material. This is important, because it is known that in mental states where people feel very disengaged from their environment (i.e. not ‘present’), emotional responsivity tends to be reduced (Medford et al. 2005, Sierra 2009).

The purpose of this study was to examine how certain aspects of the experience of being exposed in a synthetic simulation such as rendering of lighting propagation and extreme variations of rendering are modified by subtle changes in the environment. This is mostly relevant to a psychiatric syndrome called the ‘depersonalization syndrome’ (Medford et al. 2004). Depersonalization is a state in which a person feels that they and their surroundings are oddly unreal and dreamlike. Most people have experienced mild depersonalization at times e.g. when jetlagged or very tired, or when under stress. Usually it passes within a few hours. However in some people it can become intense and long-lasting, to a point where it becomes distressing and interferes with daily life. In these circumstances it may be considered as an illness (primary depersonalization disorder). It may also occur in association with other psychiatric illnesses such as depression. The work presented in this thesis was based on the observation that people suffering from the depersonalization syndrome often describe feeling slightly ‘altered’ under different lighting conditions. In particular, it has been noted that depersonalization occurs more commonly when people are in artificial lighting conditions than in natural light (Sierra 2009).

In healthy people, it is a common experience to feel that changes in lighting and other features of the environment can induce a sense of not being fully present, or of things not being fully real. This is essentially a mild, short-lived experience of depersonalization. Studying what is happening in the brain during these experiences can tell us more about how the brain creates the sense of reality, and the sense of being present or conscious in the world. This has relevance not only to understanding states of depersonalization, but also to understanding

states where a person's sense of reality has become radically altered, as happens in psychotic disorders such as schizophrenia.

In this thesis, a 3D interactive lighting framework is described, rendering interactive VEs in an fMRI scanner of varied physics-based daytime lighting as well as artificial configurations of lighting. The system supports the neuroscientific experimental protocols enabling experiments investigating the effect of lighting on neural activity related to presence and depersonalization. A complete study involving a group of healthy individuals was conducted at the Brighton and Sussex Medical School's fMRI scanner where the system was installed. Following a strict experimental protocol, the lighting system implemented involved interactive photorealistic synthetic scenes as well low quality wireframe scenes of varied lighting configurations presented to healthy volunteers who could interactively manipulate such scenes in the scanner and answer questions set by the psychiatrists collaborating in this project. The fMRI scanner acquired brain images at strictly specified timings while the participants followed the experimental protocol, performing the tasks assigned to them while being immersed in the VEs, such as navigating, or viewing emotional images. Meanwhile, participants' physiological measures were taken, such as heart rate and heart pulse oximetry.

The lighting framework presented was designed specifically render the VEs, with different lighting effects, being presented in the fMRI display, therefore, it was developed to maintain the imposed time limits and was completely synchronized with the fMRI scanner. It was designed to simulate both high-fidelity virtual scenes and low-fidelity wireframe scenes. Most importantly, it allowed the manipulation of the artificial light in real-time, providing a real-time simulation of the lighting effects. This study allowed the researchers to investigate the participants' neurocorrelates of fidelity at the same time as being immersed in the synthetic scenes, instead of self-report of fidelity or task performance, after the task has occurred.

Study participants lied in the fMRI scanner and navigated in the VE displayed on the fMRI display, depicting the inside of a house and a connected to it yard. By using button boxes, they were able to explore the VE, alter the lighting conditions, look at 'indoors' and 'outdoors' (a garden scene) parts of the environment, and interact with some objects in the scene. While doing this, they were at intervals asked to give ratings of how they are feeling and how 'real' things in the VE seem. While this was happening, the scanner was recording data on brain activity, so that the researchers were able to explore what was happening in the brain as participants felt more or less 'present' in the environment and experienced it as more or less 'real'.

1.1 Contribution

As safety and cost issues in relation to training in real world task situations are becoming increasingly complex, simulators have proven to be the most successful arena for production of synthetic environments. Within simulators, such as flight simulators, flight crew can train to deal with emergency situations, gain familiarity with new aircraft types and learn airfield specific procedures. It is argued that training in a simulator with maximum fidelity would result in transfer equivalent to real-world training, since the two environments would be indistinguishable. However, there is always a trade-off between visual/interaction fidelity and

computational complexity. The key in this trade-off is to maintain users' suspension of disbelief and presence in the VE, while keeping the rendering computations as simple as possible, so as to be performed by computers in real-time.

Therefore, it is essential to identify the factors that affect the feeling of presence or associated behavioural fidelity, so as to be able to create the least computation-demanding VEs that can still generate the sense of being in the displayed environment and promote positive transfer of training. Here, we propose to develop a real-time synthetic lighting system incorporating sophisticated global illumination algorithms aiming to induce similar subjective lighting impressions as in the real world. The system will be used as the main platform for cognitive and neuroscientific experimental paradigms related to simulator fidelity explorations.

Ultimately, the goal of this project is to use this system to explore the effect of lighting variations (daylight vs forms of artificial light) in relation to a group of patients suffering from the 'depersonalization' syndrome. Anecdotal evidence has shown that lighting affects the patients' mood in varied ways. The system implemented will host an interactive 3D graphics application which allows patients to set the light in an interior environment to their preferred state as well as rate images and events occurring in that space in relation to their emotional intensity. Brain scanning recorded through fMRI while such actions are taking place will identify neuro-correlates of brain disturbances towards therapy. Neurocorrelates of participants' actions assess the fidelity of a simulation while being immersed in the synthetic scenes, instead of previously utilized self-report of fidelity or task performance, after the task has occurred. The ultimate goal of the experiments planned is to explore whether natural and artificial scenes of varied fidelity for training or for therapeutic purposes engage common perceptual or neuroscientific mechanisms. Such input is non-obtrusive and is derived at the same time as the experience occurs.

The contribution of this thesis is the innovative application designed to render an interactive VE on an fMRI display, enabling the conduct of experiments, searching the effects of visual fidelity on feelings of presence and impressions of lighting. Feelings of presence were explored by responses to a questionnaire, as well as examining the brain activity and the heart rate during the navigation in the VEs. This study allowed the researchers to investigate the participants' neurocorrelates of fidelity.

Moreover, it is not straightforward to provide interactive synthetic stimuli to be displayed in fMRI displays due to the infrastructural and technical demands. fMRI experiments of this type currently employ simple display material, for example using photographs, video clips or simple computerized stimuli which are non-stereoscopic. Using VEs in fMRI has the advantage that it is possible to involve participants in interactive animated environments which more realistically reflect social and emotional situations. This seamless naturalism and interactivity is impossible to achieve with video clips. In addition, some experiments could be only conducted using synthetic stimuli for ethical reasons.

A broader aim of this work is to assess whether such powerful social-psychological studies could be usefully carried out within VEs advancing both cognitive neuroscience and computer graphics research.

1.2 Thesis Outline

This thesis is divided into a number of chapters, which will be outlined below.

Chapter 2 – Technical Background: This chapter introduces a set of fundamental terms in computer graphics starting with defining light and its properties, light energy, photometry and radiometry. Subsequently, computer graphics illumination models are analyzed. Furthermore, this chapter illustrates the complex variety of tools and equipment required to create, view and interact with immersive VEs. It provides background information regarding the key technologies used in order to implement the lighting system and experimental protocol put forward and an overview of the technologies necessary to display and interact with immersive VEs.

Moreover, in this chapter, previous research findings on subjective impressions of lighting effects, feelings of presence and visual fidelity metrics are presented. The concept of emotional images, which were used in the experiments, is also analyzed.

Chapter 3 – Software Architecture and Development Framework: In this chapter, the technical requirements of the interactive 3D lighting system are introduced. The architecture of the application developed for the experiments is presented, along with the inherent architecture of the Unreal Development Kit (UDK) used to develop it.

Chapter 4 – Implementation: Chapter 4 describes in detail the implementation of the interactive computer graphics framework. The technical issues that occurred are explained, as well as the decisions taken to address them. More specifically, there are examples and source code samples demonstrated, in relation to how the application met the requirements imposed by the expert psychiatrists in order to incorporate a formal neuroscientific protocol to run the fMRI scanner.

Chapter 5 – UI Implementation: In this chapter, the implementation of the User Interfaces (UI) as presented to the users in the fMRI scanner is described. The steps taken to create the individual UIs as Flash applications and embed them in the complete systems are presented. The challenges concerning the creation of interactive synthetic worlds and associated UIs as displayed in the fMRI scanner are presented and the solution to overcome them are explained.

Chapter 6 – Experiments: This chapter is concerned with the Experimental Methods employed when the actual experiments were conducted in the the fMRI scanner. The experimental procedure is presented, as well as the results of the experiments.

Chapter 7 – Conclusions: In the final chapter, the conclusions of this thesis are presented as well as hints about future work.

2 Chapter 2 – Technical Background

A Virtual Environment (VE) is a computer simulated scene which can be interactively manipulated by users. Typical scenes used in such environments generally comprise of geometric 3D models, shades, images and lights which are converted into final images through the rendering process. The rendering process must be conducted in real time in order to provide scenes which are updated reacting to user interaction. An Immersive Virtual Environment (IVE) perpetually surrounds the user within the VE.

The first and second section of this chapter presents the fundamental principles of photorealistic computer graphics rendering. The third section proceeds by describing previous research exploring subjective impressions of lighting and color effects in VEs, while the fourth section presents background information in relation to the so-called emotional images used in the experiments presented in Chapter 6 – Experiments.

2.1 Computer Graphics Rendering

There is a great need for realistic rendering of computer graphical images in real time. The term ‘realistic’ is used broadly to refer to an image that captures and displays the effects of light interacting with physical objects occurring in real environments looking perceptually plausible to the human eye, e.g. as a painting, a photograph or a computer generated image (Figure 2). In computer graphics, it is generally acceptable that if a synthetic image looks like computer graphics, it is not good computer graphics (Birn 2000).



Figure 2: The goal of realistic image synthesis: an example from photography.

There are no previously agreed-upon standards for measuring the actual realism of computer-generated images. In many cases, physical accuracy may be used as the standard to be achieved. Perceptual criteria are significant rather than physics based simulations to the point of undetermined ‘looks good’ evaluations. Ferwerda (2003) proposes three levels of realism requiring consideration when evaluating computer graphical images. These are physical realism, in which the synthetic scene is an accurate point-by-point representation of the spectral radiance values of the real scene; photorealism, in which the synthetic scene produces the same visual response as the real scene even if the physical energy depicted from the image is different compared to the real scene; and finally functional realism, in which the same information is transmitted in real and synthetic scenes while users perform visual tasks targeting transfer of training in the real world (Ferwerda 2003). Physical accuracy of light and

geometry does not guarantee that the displayed images will seem real. The challenge is to devise real time rendering methods which will produce perceptually accurate synthetic scenes, as well as realistic behaviour in real – time. Behavioural fidelity is defined as the degree one acts and reacts in a simulation as in the real-world task situation simulated. Behavioural fidelity could be related to behavioural responses communicated by self-report or task performance as well as physiological measures and neural activity compared as derived in the real world and in the simulation. The next section examines how light propagation is simulated with computer graphics technologies. We will analyze global illumination solutions to the rendering equation focusing on radiosity rendering employed in this work.

2.1.1 The Physical Behavior of the Light

Light is one form of *electromagnetic radiation*, a mode of propagation of energy through space that includes radio waves, radiant heat, gamma rays and X-rays. One way in which the nature of electromagnetic radiation can be pictured is as a pattern of waves propagated through an imaginary medium. The term ‘visible light’ is used to describe the subset of the spectrum of electromagnetic energy to which the human eye is sensitive. This subset, usually referred to as the *visual range* or the *visual band*, consists of electromagnetic energy with wavelengths in the range of 380 to 780 nanometers, although the human eye has very low sensitivity to a wider range of wavelengths, including the infrared and ultraviolet ranges. The range of visible light is shown in Figure 3. As shown, the wavelength at which the human eye is most sensitive is 555 nm.

In the field of computer graphics three types of light interaction are primarily considered: *absorption, reflection and transmission*. In the case of absorption, an incident photon is removed from the simulation with no further contribution to the illumination within the environment. Reflection considers incident light that is propagated from a surface back into the scene and transmission describes light that travels through the material upon which it is incident and can then return to the environment, often from another surface of the same physical object. Both reflection and transmission can be subdivided into three main types:

Specular: When the incident light is propagated without scattering as if reflected from a mirror or transmitted through glass.

Diffuse: When incident light is scattered in all directions.

Glossy: This is a weighted combination of diffuse and specular.

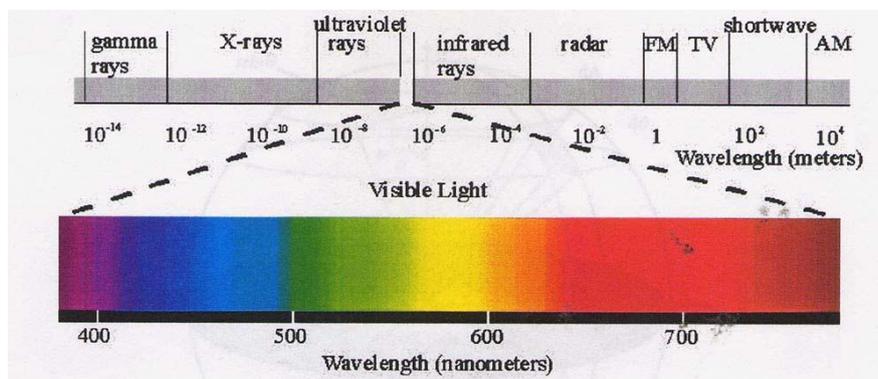


Figure 3: The visible portion of the electromagnetic spectrum.

Most materials do not fall exactly into one of the material categories described above but instead exhibit a combination of specular and diffuse characteristics.

In order to create shaded images of three dimensional objects, we should analyze in detail how the light energy interacts with a surface. Such processes may include emission, transmission, absorption, refraction, interference and reflection of light (Palmer 1999).

- **Emission** is when light is emitted from an object or surface, for example the sun or man-made sources, such as candles or light bulbs. Emitted light is composed of photons generated by the matter emitting the light; it is therefore an intrinsic source of light.
- **Transmission** describes a particular frequency of light that travels through a material returning into the environment unchanged as shown in Figure 4. As a result, the material will be transparent to that frequency of light. Most materials are transparent to some frequencies, but not to others. For example, high frequency light rays, such as gamma rays and X-rays, will pass through ordinary glass, but the lower frequencies of ultraviolet and infrared light will not.

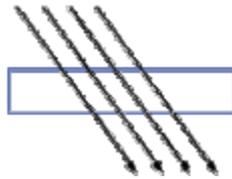


Figure 4: Light transmitted through a material.

- **Absorption** describes light as it passes through matter resulting in a decrease in its intensity as shown in Figure 5, i.e. some of the light has been absorbed by the object. An incident photon can be completely removed from the simulation with no further contribution to the illumination within the environment if the absorption is great enough.

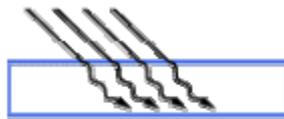


Figure 5: Light absorbed by a material.

- **Refraction** describes the bending of a light ray when it crosses the boundary between two different materials as shown in Figure 6. This change in direction is due to a change in speed. Light travels fastest in empty space and slows down upon entering matter. The refractive index of a substance is the ratio of the speed of light in space (or in air) to its speed in the substance. This ratio is always greater than one.

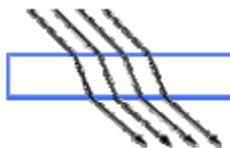


Figure 6: Light refracted through a material.

- **Interference** is an effect that occurs when two waves of equal frequency are superimposed. This often happens when light rays from a single source travel by different paths to the same point. If, at the point of meeting, the two waves are in phase (the crest of one coincides with the crest of the other), they will combine to form a new wave of the same frequency. However, the amplitude of this new wave is the sum of the amplitudes of the original waves. The process of forming this new wave is called *constructive interference* (NightLase 2004). If the two waves meet out of phase (a crest of one wave coincides with a trough of the other), the result is a wave whose amplitude is the difference of the original amplitudes. This process is called *destructive interference* (NightLase 2004). If the original waves have equal amplitudes, they may completely destroy each other, leaving no wave at all. Constructive interference results in a bright spot; destructive interference produces a dark spot.
- **Reflection** considers incident light that is propagated from a surface back into the scene. Reflection depends on the smoothness of the material's surface relative to the wavelength of the radiation (ME 2004). A rough surface will affect both the relative direction and the phase coherency of the reflected wave. Thus, this characteristic determines both the amount of radiation that is reflected back to the first medium and the purity of the information that is preserved in the reflected wave. A reflected wave that maintains the geometrical organization of the incident radiation and produces a mirror image of the wave is called a *specular reflection*, as can be seen in Figure 7.

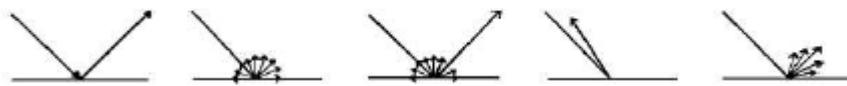


Figure 7: Light reflected off a material in different ways. From left to right, specular, diffuse, mixed, retro-reflection and finally gloss (Katedros 2004).

2.1.2 Computer Graphics Illumination Models

An illumination model computes the color at a point in terms of light directly emitted by the light source(s). A *local illumination model* calculates the distribution of light that comes directly from the light source(s). A *global illumination model* additionally calculates reflected light from all the surfaces in a scene which could receive light indirectly via interreflections from other surfaces. Global illumination models include, therefore, the complete light interaction in a scene, allowing for soft shadows and color bleeding that contribute towards a more photorealistic image. The rendering equation expresses the light being transferred from one point to another (Kajiya 1986). Most illumination computations are approximate solutions of the rendering equation:

$$I(x,y) = g(x,y) [\epsilon(x,y) + \int S p(x, y,z) I(y,z) dz]$$

where

x,y,z are points in the environment,

$I(x,y)$ is related to the intensity passing from y to x ,

$g(x,y)$ is a 'geometry' term that is 0 when x,y are occluded from each other and 1 otherwise,

$p(x,y,z)$ is related to the intensity of light reflected from z to x from the surface at y , the integral is over all points on all surfaces S .

$\epsilon(x,y)$ is related to the intensity of light that is emitted from y to x .

Thus, the rendering equation states that the light from y that reaches x consists of light emitted by y itself and light scattered by y to x from all other surfaces which themselves emit light and recursively scatter light from other surfaces. The distinction between *view-dependent* rendering algorithms and *view-independent* algorithms is a significant one. *View-dependent* algorithms discretise the view plane to determine points at which to evaluate the illumination equation, given the viewer's direction, such as ray-tracing (Glassner 2000). *View-independent* algorithms discretise the environment and process it in order to provide enough information to evaluate the illumination equation at any point and from any viewing direction, such as radiosity.

Bouknight (1970) introduced one of the first models for local illumination of a surface. This included two terms, a *diffuse* term and an *ambient* term. The diffuse term is based upon the Lambertian reflection model, which makes the value of the outgoing intensity equal in every direction and proportional to the cosine of the angle between the incoming light and the surface normal. The ambient term is constant and approximates diffuse inter-object reflection. Gouraud (1971) extended this model to calculate the shading across a curved surface approximated by a polygonal mesh. His method calculated the outgoing intensities at the polygon vertices and then interpolated these values across the polygon as shown in Figure 8 (middle).

Phong (1975) introduced a more sophisticated interpolation scheme where the surface normal is interpolated across a polygon and the shading calculation is performed at every visible point, as shown in Figure 8 (right). He also introduced a *specular* term. Specular reflection is when the reflection is stronger in one viewing direction, i.e. there is a bright spot called a *specular highlight*. This is readily apparent on shiny surfaces. For an ideal reflector, such as a mirror, the angle of incidence equals the angle of specular reflection. Although this model is not physically based, its simplicity and efficiency make it still the most commonly used local reflection model.

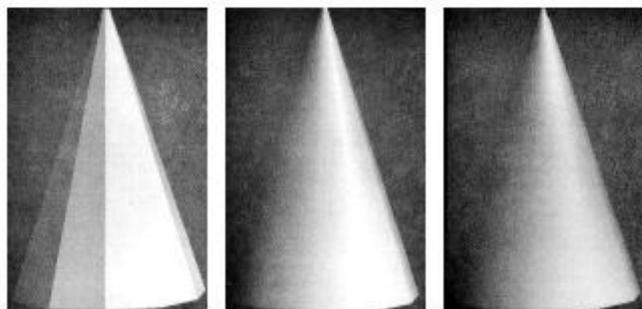


Figure 8: The differences between a simple computer generated polyhedral cone (left), linearly interpolated shading to give appearance of curvature (Gouraud Shading). Note Mach bands at edges of faces (middle) and a more complex shading calculation, interpolating curved surface normals (Phong Shading). This is necessary to eliminate Mach Bands (right).

A global illumination model adds to the local illumination model, the light that is reflected from other non-light surfaces to the current surface. A global illumination model is physically correct and produces realistic images resulting in effects such as color bleeding and soft shadows. When measured data is used for the geometry and surface properties of objects in a scene, the image produced should then be theoretically indistinguishable from reality. However, global illumination algorithms are also more computationally expensive.

Global illumination algorithms produce solutions of the rendering equation proposed by Kajiya (1986):

$$L_{out} = L_E + \int_{\Omega} L_{in} f_r \cos(\theta) d_{\omega_{\theta}}$$

where L_{out} is the radiance leaving a surface, L_E is the radiance emitted by the surface, L_{in} is the radiance of an incoming light ray arriving at the surface from light sources and other surfaces, f_r is the bi-directional reflection distribution function of the surface, θ is the angle between the surface normal and the incoming light ray and $d_{\omega_{\theta}}$ is the differential solid angle around the incoming light ray.

The rendering equation is graphically depicted in Figure 9. In this figure L_{in} is an example of a direct light source, such as the sun or a light bulb, L'_{in} is an example of an indirect light source i.e. light that is being reflected off another surface, R, to surface S. The light seen by the eye, L_{out} , is simply the integral of the indirect and direct light sources modulated by the reflectance function of the surface over the hemisphere Ω .

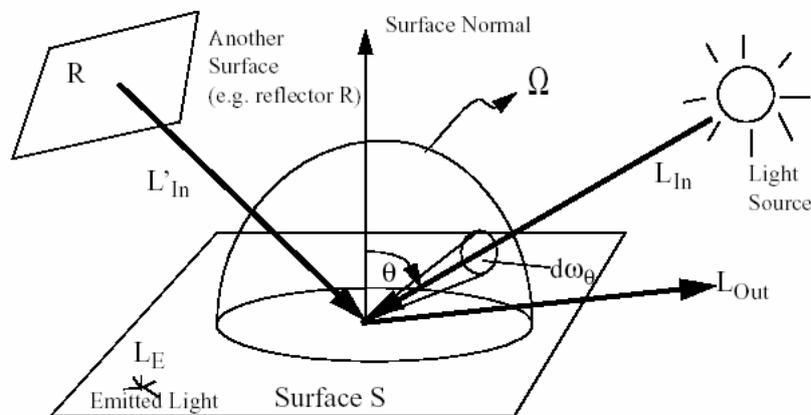


Figure 9: Graphical Depiction of the rendering equation (Yee 2000).

The problem of global illumination can be seen when you have to solve the rendering equation for each and every point in the environment. In all but the simplest case, there is no closed form solution for such an equation so it must be solved using numerical techniques and therefore this implies that there can only be an approximation of the solution (Lischinski 2004). For this reason most global illumination computations are approximate solutions to the rendering equation.

The two major types of graphics systems that use the global illumination model are *radiosity* and *ray tracing*. For the processes of this work, *Lightmass* was used, which is a global

illumination algorithm that combines the *radiosity* and *ray-tracing* algorithm techniques and is developed and supported by the Unreal Development Kit (UDK).

2.1.3 Ray Tracing

Ray tracing is a significant global illumination algorithm which calculates specular reflections (view dependent) and results in a rendered image. Rays of light are traced from the eye through the centre of each pixel of the image plane into the scene, these are called *primary rays*. When each of these rays hits a surface it spawns two child rays, one for the reflected light and one for the refracted light. This process continues recursively for each child ray until no object is hit, or the recursion reaches some specified maximum depth. Rays are also traced to each light source from the point of intersection. These are called *shadow rays* and they account for direct illumination of the surface, as shown in Figure 10. If a shadow ray hits an object before intersecting with the light source(s), then the point under consideration is in shadow. Otherwise, there must be clear path from the point of intersection of the primary ray to the light source and thus a local illumination model can be applied to calculate the contribution of the light source(s) to that surface point.

The simple ray tracing method outlined above has several problems. Due to the recursion involved and the possibly large number of rays that may be cast, the procedure is inherently expensive. Diffuse interaction is not modeled, nor is specular interaction, other than that by perfect mirrors and filters. Surfaces receiving no direct illumination appear black. In order to overcome this, an indirect illumination term, referred to as *ambient light*, is accounted for by a constant ambient term, which is usually assigned an arbitrary value (Glassner 2000). Shadows are hard-edged and the method is very prone to aliasing. The result of ray tracing is a single image rendered for a particular position of the viewing plane, resulting in a view –dependent technique.

In ray tracing each ray must be tested for intersection with every object in the scene. Thus, for a scene of significant complexity, the method rapidly becomes impracticable. Several acceleration techniques have been developed, which may be broadly categorized into two approaches: reducing the number of rays and reducing the number of intersection tests. Hall and Greenberg noted that the intensity of each ray is reduced by each surface it hits, thus the number of rays should be stopped before any unnecessary recursion to a great depth occurs (Hall and Greenberg 1983). Another approach, which attempts to minimize the number of ray object intersections, is *spatial subdivision*. This method encloses a scene in a cube that is then partitioned into discrete regions, each of which contains a subset of the objects in the scene. Each region may then be recursively subdivided until each sub-region (voxel or cell) contains no more than a preset maximum number of objects.

Several methods for subdividing space exist. Glassner (1984) proposes the use of an *octree*, e.g. a structure where the space is bisected in each dimension, resulting in eight child regions. This subdivision is repeated for each child region until the maximum tree depth is reached, or a region contains less than a certain number of objects. Using such a framework allows for *spatial coherence*, i.e. the theory that similar objects in a scene affect neighboring pixels. Rays are traced through individual voxels, with intersection tests performed only for the

objects contained within, rather than for all the objects in the scene. The ray is then processed through the voxels by determining the entry and exit points for each voxel traversed by the ray until an object is intersected or the scene boundary is reached.

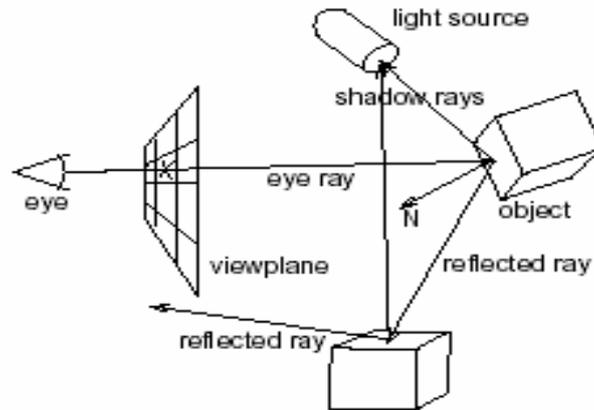


Figure 10: Ray-tracing.

2.1.4 Radiosity

Radiosity calculates diffuse reflections in a scene and results in a finally divided geometrical mesh. The scenes produced in this work have been rendered using radiosity calculations. The *radiosity* method of computer image generation has its basis in the field of thermal heat transfer (Goral et al. 1984). The heat transfer theory describes radiation as the transfer of energy from a surface when that surface has been thermally excited. This encompasses both surfaces that are basic emitters of energy, as with light sources and surfaces that receive energy from other surfaces and thus have energy to transfer. The thermal radiation theory can be used to describe the transfer of many kinds of energy between surfaces, including light energy.

As in thermal heat transfer, the basic radiosity method for computer image generation makes the assumption that surfaces are diffuse emitters and reflectors of energy, emitting and reflecting energy uniformly over their entire area. Thus, the radiosity of a surface is the rate at which energy leaves that surface (energy per unit time per unit area). This includes the energy emitted by the surface as well as the energy reflected from other surfaces in the scene. Light sources in the scene are treated as objects that have self emittance.

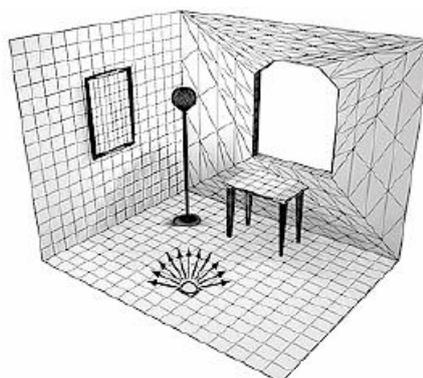


Figure 11: Radiosity (McNamara 2000).

The environment is divided into surface patches, Figure 11, each with a specified reflectivity and between each pair of patches there is a *form factor* that represents the proportion of light leaving one patch (*patch i*) that will arrive at the other (*patch j*) (Siegel and Howell 1992).

Thus the radiosity equation is:

$$B_i = E_i + r_i \sum_j B_j F_{ij}$$

Where:

B_i = Radiosity of *patch i*

E_i = Emissivity of *patch i*

r_i = Reflectivity of *patch i*

B_j = Radiosity of *patch j*

F_{ij} = Form factor of *patch j* relative to *patch i*

Where the form factor, F_{ij} , is the fraction of energy transferred from *patch i* to *patch j* and the reciprocity relationship (Siegel and Howell 1992) states:

$$A_j F_{ji} = A_i F_{ij}$$

Where A_j and A_i are the areas of patch j and i respectively, as shown in Figure 12.

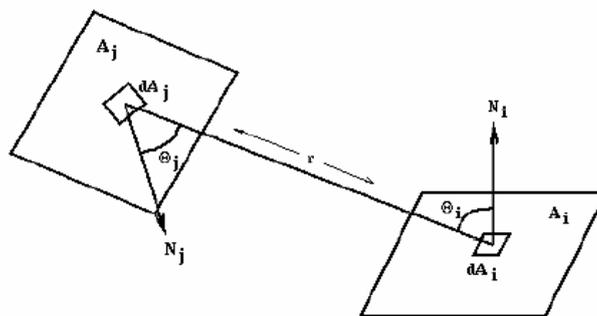


Figure 12: Relationship between two patches (Katedros 2004).

As the environment is closed, the emittance functions, reflectivity values and form factors form a system of simultaneous equations that can be solved to find the *radiosity* of each patch. The radiosity is then interpolated across each of the patches and finally the image can then be rendered.

The basic form factor equation is difficult even for simple surfaces. Nusselt (1928) developed a geometric analog that allows the simple and accurate calculation of the form factor between a surface and a point on a second surface. The *Nusselt Analog* involves placing a hemispherical projection body, with unit radius, at a point on the surface A_i . The second surface, A_j , is spherically projected onto the projection body, and then cylindrically projected onto the base of the hemisphere. The form factor then may be approximated by the area

projected on the base of the hemisphere divided by the area of the base of the hemisphere, as shown in Figure 13.

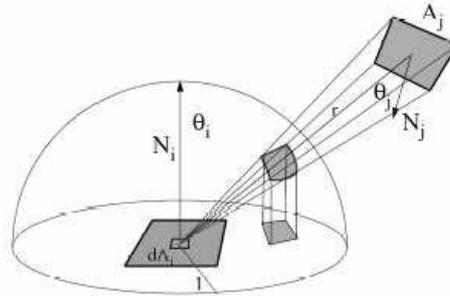


Figure 13: Nusselt's analog. The form factor from the differential area dA_i to element A_j is proportional to the area of the double projection onto the base of the hemisphere (Nusselt 1928).

Cohen and Greenberg (1985) proposed that the form factor between each pair of patches could also be calculated by placing a hemi-cube on each patch and projecting the environment on to it as defined by the Nusselt Analog. Each face of the hemicube is subdivided into a set of small, usually square ('discrete') areas, each of which has a precomputed delta form factor value, as shown in Figure 14. When a surface is projected onto the hemicube, the sum of the delta form factor values of the discrete areas of the hemicube faces which are covered by the projection of the surface is the form factor between the point on the first surface (about which the cube is placed) and the second surface (the one which was projected). The speed and accuracy of this method of form factor calculation can be affected by changing the size and number of discrete areas on the faces of the hemicube.

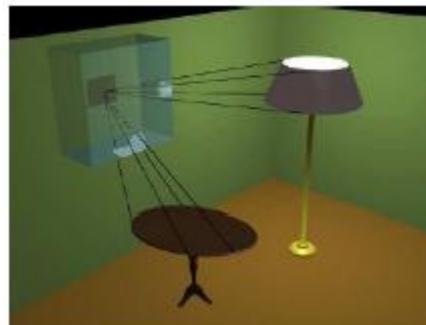
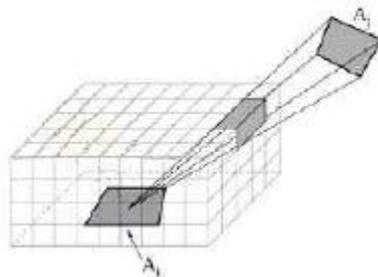


Figure 14: The hemicube (Langbein 2004).

Radiosity assumes that an equilibrium solution can be reached; that all of the energy in an environment is accounted for, through absorption and reflection. It should be noted that because of the assumption of only perfectly diffuse surfaces, the basic radiosity method is **viewpoint independent**, i.e. the solution will be the same regardless of the viewpoint of the image. The diffuse transfer of light energy between surfaces is unaffected by the position of the camera. This means that as long as the relative position of all objects and light sources remains unchanged, the radiosity values need not be recomputed for each frame. This has made the radiosity method particularly popular in architectural simulation, targeting high-quality walkthroughs of static environments. Figure 15 demonstrates the difference in image quality that can be achieved with radiosity compared to ray tracing.



Figure 15: The difference in image quality between ray tracing (middle) and radiosity (right hand image).

However, there are several problems with using the hemicube radiosity method. It can only model diffuse reflection in a closed environment; it is limited to polygonal environments; it is prone to aliasing and has excessive time and memory requirements. Also, only after all the radiosities have been computed in the scene is the resultant image displayed. There is a form factor between each pair of patches, so in an environment with N patches, N^2 form factors must be stored. For a scene of moderate complexity this will require a vast amount of storage and as the form factor calculation is non-trivial the time taken to produce a solution can be extensive. This means that the user is unable to alter any of the parameters of the environment until the entire computation is complete. Then once the alteration is made, the user must once again wait until the full solution is recomputed.

The visual quality of the rendered images in radiosity also strongly depends on the method employed for discretizing the scene into patches. A too fine discretization may give rise to artefacts, while with a coarse discretization, areas with high radiosity gradients may appear (Gibson and Hubbard 1997). To overcome these problems, the discretization should adapt to the scene. That is, the interaction between two patches should account for the distance between them as well as their surface area. In other words, surfaces that are far away are discretized less finely than surfaces that are nearby. These aspects are considered by the adaptive discretization method proposed by Langu  nou et al. (1992). It performs both discretization and system resolution at each iteration of the shooting process, which allows for interactivity. Gibson and Hubbard (1997) demonstrated another solution for this problem by presenting an oracle that stops patch refinement once the difference between successive levels of elements becomes perceptually unnoticeable.

Progressive refinement radiosity (Cohen et al. 1988) works by not attempting to solve the entire system simultaneously. Instead, the method proceeds in a number of passes and the result converges towards the correct solution. At each pass, the patch with the greatest unshot radiosity is selected and this energy is propagated to all other patches in the environment. This is repeated until the total unshot radiosity falls below some threshold. Progressive refinement radiosity generally yields a good approximation to the full solution in far less time and with lesser storage requirements, as the form factors do not all need to be stored throughout. Many other extensions to radiosity have been developed and a very comprehensive bibliography of these techniques can be found in (Ashdown 2004).

2.2 Virtual Reality Technology and Game Engines

There are several game engines offering a wide range of tools to create interactive photorealistic environments, primarily used for the development of a computer game. Game engines are also powerful platforms for the development of any 3D interactive environment. It was decided early on in this project that the implementation of the 3D interactive scene would be based on a game engine for this reason. Although the final application is not a computer game, a game engine offers the required tools and programming platform in order to create an interactive photorealistic environment.

A game engine provides the framework and the Application User Interface (API) for the developer to use and communicate with the hardware. It consists of separate autonomous systems, each handling a specific process, e.g. the graphics system, the sound system, the physics system, etc. Figure 16 shows the standard architecture of game engines.

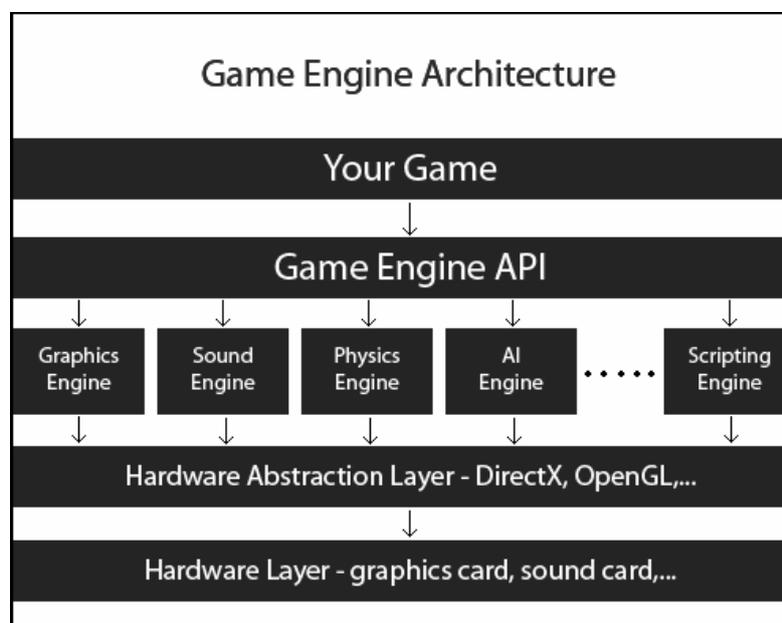


Figure 16: Architecture of a game engine.

In the following section, the game engines that were considered but rejected will be presented, as well as UDK, which is the game engine that was selected for the implementation of the 3D lighting system.

2.2.1 Unity 3D

Unity 3D is a fully featured application suite, providing tools to create 3D games or other applications, such as architectural visualization. It provides support for editing object geometry, surfaces, lights and sounds. It uses the Ageia physics engine, provided by nVidia. A lightmapping system, called Beast, is included.

In terms of programming, Unity 3D supports three programming languages: JavaScript, C# and Boo, which is a python variation. All three languages are fast and can be interconnected. The game’s logic runs in the open-source platform “Mono”, offering speed and flexibility. Required for the development process a debugger is also included, allowing pausing the game at any time and resuming it step-by-step.

Unity 3D is widely used, utilized by a large community offering help. It is free for non-commercial use and is targeted to all platforms, such as PC, MAC, Android, iOS and web.

This game engine targets at offering increased rendering speed, even on machines with low memory and computational power, such as iOS and Android smartphones, and not at creating interactive photorealistic environments, which need a lot of memory and very fast CPU and GPU to render at acceptable speed. Unity 3D was rejected, because it was necessary to have the ability to create photorealistic VEs and render them in real-time.

2.2.2 Torque 3D

Torque 3D is a sophisticated game engine for creating networked games. It includes advanced rendering technology, a Graphical User Interface (GUI) building tool and a World Editor, providing an entire suit of WYSIWYG (What-You-See-Is-What-You-Get) tools to create the game or simulation application.

The programming language used is “TorqueScript”, which resembles C/C++. It is targeted for both Windows and MacOS platforms, as well as the web. The main disadvantage of Torque 3D is that it is not free, but it needs to be licensed for \$100. For this reason, Torque 3D was, also, rejected.

2.2.3 Unreal Engine 3 – Unreal Development Kit (UDK)

Unreal Development Kit (UDK) is one of the leading game engines currently. It became free on November 2009 for non-commercial use and is used by the world’s largest development studios. The UDK community includes thousands of people from around the world, providing help and advice.

UDK’s core is written in C++, making it very fast. It offers the ability to use both “UnrealScript”, UDK’s object-oriented scripting language, and C/C++ programming languages. It provides many different tools for the creation and the rendering of a virtual scene. The Unreal Development Kit includes the Unreal Lightmass, which is an advanced global illumination solver. Unreal Lightmass supports the illumination with a single sun, giving off soft shadows and automatically computing the diffuse interreflection (color bleeding). It also offers a variety of options to optimize the illumination solution. It can provide detailed shadows by using directional light mapping, static shadowing and diffuse normal-mapped lighting. An unlimited number of lights can be pre-computed and stored in a single set of texture maps. The complete software architecture of UDK will be presented in Chapter 3 – Software Architecture and Development Framework.

2.3 Subjective Impressions of Lighting and Color Effects

2.3.1 Introduction

It is well known that lighting and color effects in a Virtual Environment (VE) have an impact on human perception of that scene, affecting the user’s behavior, emotions, feelings of presence and task performance. Relevant research in this field exploring the effect of lighting,

use a task-oriented approach, aiming to measure users' task performance under different lighting conditions. In some cases detailed below, relevant experimental studies explore users' subjective perception of lighting and/or color effects. This section aims to identify, describe and categorize the major findings of research exploring subjective impressions of lighting and color effects. The modern view about computer graphics technologies is that slavish simulation of physical propagation of light does not necessarily communicate the subjective impressions that lighting can evoke. A major challenge for the technical, as well as experimental work described in this thesis is to produce synthetic environments which would affect users' mood and subjective impressions similarly as in the real-world. It is of interest to examine whether lower fidelity environments can communicate such impressions.

In order to measure the subjective impressions of lighting effects, researchers often cite Flynn's work in the 1970's (see Zimmons, 2004 and Mania and Robinson, 2006), who published a series of articles (Flynn et al, 1973, 1977, 1979), introducing a methodology with which he quantifies the parameters that elicit a shared human behavioral response and subjective impression under specific lighting conditions. Flynn was not only able to demonstrate that there is a definite correlation between the measurable physical quantities of lighting (non-uniform, peripheral and bright lighting) and the subjective impressions lighting induces (visual clarity, spaciousness, and relaxation), but was able to quantify the amount of influence of each of the measurable dimensions on each subjective impression.

In particular, Flynn examined how non-uniform, peripheral, and bright lighting affects impressions of visual clarity, spaciousness, relaxation, and privacy. Flynn created six different light settings for a conference room and subjectively associated each lighting setting with a *uniform/non-uniform* value, referring to the articulation or modeling of the room and/or articulation of forms and objects in the room, an *overhead/peripheral* value, referring to a lighting emphasis of vertical surfaces, as distinguished from overhead luminaires that light central horizontal surfaces and a *bright/dim* value, referring to the perceived intensity of light on the horizontal activity plane, so that each setting corresponded to a point in a 3-dimensional space of lighting characteristics, such as overhead downlighting low intensity lighting. Flynn also associated a set of Semantic Differential (SD) rating scales (such as large-small, clear-hazy, pleasant-unpleasant and spacious-cramped) with each category of impression, such as impressions of visual clarity, spaciousness, spatial complexity, color tone, glare. Test subjects were then asked to make pair wise comparisons of the differences between each room accommodating a specific lighting setting from the set of SD rating scales where 0 meant no difference and 10 meant a large difference. The data gathered resulted in a 6x6 symmetric dissimilarity matrix comparing the 6 rooms for each subject tested and each SD comparison made. The results showed that brightness, non-uniformity, and peripheral lighting reinforce particular impressions: for instance, the brightest condition was reported as spacious, most clear and bright, overhead diffusing conditions (both bright and dim) were reported as more hostile, and monotonous. In addition, subjective impressions of lighting have proved to be similar when utilizing similar light settings in different rooms and with different object arrangements or activity settings indicating that the modifying effect of lighting is consistent across rooms.

2.3.2 Effects of Lighting Conditions on Feelings of Presence and Attention

The effect of varied lighting conditions in synthetic environments on the subjective impression of presence has been examined. Research results indicated that there is a positive correlation between presence and subjective impressions of lighting associated with a high-quality full-shadow accuracy rendered condition (Mania and Robinson, 2006). A high level of perceived presence resulted in a high rating of “comfort”, “warmth”, “spacious feeling” and “relaxing feeling” associated with subjective lighting impressions. Zimmons, 2004 has shown that subjects occupied synthetic spaces lit with higher intensities for longer periods of time and gazed longer at objects that were displayed under higher luminance lighting contrast conditions. Higher lighting contrast conditions occur when the illumination intensity of the object was higher than the overall illumination intensity of the scene. Zimmons’ work also showed that lighting conditions do not affect users’ feelings of presence in a high-stress environment.

2.3.3 Effects of Lighting Variations on Task Performance

Lighting effects have an impact on task performance while immersed in synthetic scenes. A high level of simulation fidelity in approximation to the real-world illumination has resulted in improvements of task performance, such as object recognition, object alignment or office work tasks and subjective impression benefits, such as enhanced feelings of presence and confidence and improved generalization of training to other tasks (Maida et al. 1997, Veitch et al. 2008, Izso et al. 2009). Photorealistic lighting conditions in synthetic environments result in improved performances and higher feeling of confidence, success and presence (Izso et al. 2009). Users in a Virtual Environment which includes photorealistic shadows, light colour and light intensity feel more confident that their fulfillment of a simple alignment task, similar to the alignment of the orbiter with the Mir docking target, will be accurate, that their training will generalize to other tasks and that it is generally more realistic (Maida et al. 1997). Literature suggests that lighting conditions that improve visibility make people feel the space more attractive and also improve their performance in specific tasks of office work, like responding to visual and auditory prompts or categorizing short articles (Veitch et al. 2008).

Illumination effects such as shadow edges affect visual memory and the ability of users to locate specific objects among others, as well as facilitate the user in identifying 3D objects with respect to object shape, rather than simply encode the cast shadows as an artifact within the image (Tarr et al. 1998). Also, realistic lighting quality and orientation improves the ability to perform the task and identify the object (Zimmons, 2004).

Short-term visual memory and longer-term visual memory are affected by changes in illumination. Several experiments carried out suggested that a change in illumination direction induces a decrement in recognition performance, both for short-term (over a few seconds) and longer-term (over a day) visual memory (Tarr et al. 1998). Second, much of the cost in response times arising from variation in lighting direction appears to be associated with the presence of cast shadows that may produce spurious edges or surfaces. Third, while it may be true that the effects of illumination are difficult to discount in early vision, it also seems that they serve a useful function—that of disambiguating three-dimensional shapes.

2.3.4 Effects of Lighting on Mood and Emotion

The mood of the people can be greatly affected by different lighting conditions, as people have reported more pleasant mood when they subjectively perceived their lighting conditions as being of higher quality, that is brighter and uniform (Veitch et al. 2008). There are systematic influences of lighting on mood influenced by the lighting parameters within the range of those encountered in everyday interior conditions: 266-270lx for low illuminance condition and 807-821lx for high illuminance conditions, 3000°K for warm Colour Temperature and 4000°K for cool Colour Temperature. The nature of the lighting effects is complex and is best summarized as initial effects and longer-term effects (McCloughan et al. 1999). Initial effects link illuminance with sensation seeking and Correlated Colour Temperature (CCT) with hostility. The *Correlated Color Temperature* (T_{cp}) is the temperature of an incandescent black-body radiator whose perceived colour most closely resembles that of a given stimulus at the same brightness. The main effect of illuminance is on the mood variable of Sensation seeking, which is significantly higher under lower (266-270lx) than under higher (807-821lx) illuminance condition. As for the effect of CCT, it relates to the negative mood of hostility, which was significantly higher under the warm (around 3000°K) than under the cool (around 4000°K) CCT condition. Longer-term effects involve complex interactions between gender, illuminance and CCT, with interactive effects between illuminance, CCT and gender to influence negative mood and general negative mood shifts with lighting.

Illuminance intensity (BRIGHTNESS) levels have an influence on the perception of interiors and color discrimination performance. One major factor determining the impression induced by the lighting was the illuminance intensity (Boyce and Cuttle, 1990). Increasing the illuminance intensity resulted in the lighting of the room appearing more pleasant, more comfortable, clearer, more stimulating, brighter, more colourful, more natural, friendlier, warmer, more uniform, less hazy, less oppressive, less dim and less hostile. The correlated colour temperature of the lamps used had virtually no effect on the observer's impression of the lighting of the room. The other major factor influencing the impression of the lighting of the room was the presence of natural colour. Introducing natural colour, in the form of fruit and flowers, enhances the positive impressions created by the lighting, particularly at the higher illuminances. This enhancement occurs regardless of the correlated colour temperature of the lamps being used. Also, higher illuminance levels make rooms to appear more spacious (Houser et al. 2002).

In a cross cultural study of indoor environments, the impact of light and color on psychological mood has been examined (Küller et al. 2006). The aim of the study was to determine whether indoor lighting and color would have any systematic impact on the mood of people indoors. The results showed that people's mood was at its lowest when the lighting was experienced as much too dark. The mood then improved and reached its highest level when the subjectively perceived lighting conditions were "just right - neither too dark, nor too bright" (these conditions varied a lot between countries and the illuminance intensity reported as "just right" was between 300lx-800lx); then declined again when it became too bright.

2.3.5 Effects of Lighting on Mood of Depersonalization Syndrome Patients

In relation to a group of patients suffering from the ‘depersonalization’ syndrome, anecdotal evidence has shown that various configurations of lighting affect the patients’ mood negatively. The depersonalization syndrome is an alteration in the perception of experience of the self so that one feels detached from experienced mental processes or own body as if operating as an outside observer, a spectator of life (Medford et al. 2005). Emotional responses of such populations are numbed or even non-existent. Such patients do not process emotionally salient material in the same way as healthy controls.

The purpose of this study is to examine how certain aspects of experience are modified by subtle changes in the environment.

The study is based on the observation that people often describe feeling slightly ‘altered’ under different lighting conditions. In particular, it has been noted that depersonalization occurs more commonly when people are in artificial lighting conditions than in natural light (Sierra 2009). Depersonalization is a state in which a person feels that they and their surroundings are oddly unreal and dreamlike. Most people have experienced mild depersonalization at times e.g. when jetlagged or very tired, or when under stress. Usually it passes within a few hours. However in some people it can become intense and long-lasting, to a point where it becomes distressing and interferes with daily life. In these circumstances it may be considered as an illness (primary depersonalization disorder). It may also occur in association with other psychiatric illnesses such as depression.

In healthy people, it is a common experience to feel that changes in lighting and other features of the environment can induce a sense of not being fully present, or of things not being fully real. This is essentially a mild, short-lived experience of depersonalization. Studying what is happening in the brain during these experiences can tell us more about how the brain creates the sense of reality, and the sense of being present in the world. This has relevance not only to understanding states of depersonalization, but also to understanding states where a person’s sense of reality has become radically altered, as happens in psychotic disorders such as schizophrenia.

To study this in the fMRI scanner we propose to use a virtual environment (VE) which can be projected into the scanner. This approach dovetails with a related interest in ‘feelings of presence’ – these are feelings that have been studied in virtual reality research, where the concept of ‘presence’ has been used to explore how realistic a VE feels to study participants. Standard rating scales for the measurement of ‘presence’ in VEs have been developed (Slater et al. 2009).

Study participants lie in the scanner and see a scene which is a computerized mock-up of the inside of a house. By using a joystick/button-box, they will be able to explore this scene, alter the lighting conditions, look at ‘indoors’ and ‘outdoors’ (a garden scene) parts of the environment, and interact with some objects in the scene. While doing this, they will at intervals be asked to give ratings of how they are feeling and how ‘real’ things in the VE seem. While this is happening, the scanner will be recording data on brain activity, so that we will be able to explore what is happening in the brain as people feel more or less ‘present’ in the environment and experience it as more or less ‘real’.

From a neuroscientific point of view, we wish to explore how the normal population, as well as patients suffering from the so-called depersonalization syndrome, respond to varied lighting configurations of degraded photorealistic quality while exposed to immersive VEs. Potentially, the results from such experiments could improve the life of the depersonalization patients involved by, for instance, making them aware of changes they should apply to the lighting of their every-day surroundings in order to improve their mental state.

The aims of the study detailed in Chapter 6 are:

- To explore changes in regional brain activity associated with changes in the sense of ‘presence’ and thus discover more about the neural circuitry that supports such feelings.
- To examine the interaction between changes in the sense of presence and the neural response to – and subjective experience of – emotional material. This is important, because it is known that in mental states where people feel very disengaged from their environment (i.e. not ‘present’), emotional responsivity tends to be reduced (Medford et al. 2005, Sierra 2009).

2.3.6 Effects of Color on Space Perception and Subjective Impressions

Previous research has investigated users’ perception of a room, painted with different color combinations (Stahre, 2006). The results showed that green rooms were experienced as open, tranquil, lacking cheerfulness, more formal, hard, relatively warmer than light green and more surrounding. The light green rooms were experienced to be cooler and more open. The blue-green room was experienced to be the coldest. Pink rooms were perceived to be neither formal, nor tranquil, giving a cheerful impression and a surrounding and lively feel and were the colors in which the observers reacted strongest to in the study. All pink rooms were described as warm, except for the bluish pink, which was described as cold. The light pink and the pink rooms offered a similar experience, while the light green and the green rooms differed. The more colorful samples were perceived as dynamic, striking, vivid, strong, soft and gaudy.

By studying the association of common terms to colours, Mahnke, 1996, concluded that there are cross-cultural similarities in relation to subjective feelings when exposed to specific colours. The results showed that there are associations of subjective feelings to colours, such as love being associated to red, hatred to black-red, peace/tranquility to green/blue (Mahnke, 1996). Also, for mourning/sorrow the colors chosen mostly were black and gray, for happy yellow and orange, for jovial orange and yellow, for life green and for luminous yellow. These hues chosen for those terms also conform to general selections made in color-psychology tests on their associative-symbolic content.

As for the subjective impressions of colour on specific locations (ceiling, wall, floor) in the interior space, research showed that a particular hue that is perfectly suitable on an indoors location (ceiling, wall, floor) may elicit an entirely different reaction when applied to a different location (Mahnke, 1996). For example, the brown colour was reported as steady and stable when placed on floor, but was reported as awkward and discomforting when placed on

the ceiling. Mahnke's work showed that the location (ceiling, walls, floor) of colour within the interior space can make a great deal of difference in influencing a room's character, the way it is perceived psychologically and subsequent reactions to it. In a basic overview, the results show that pink ceilings are delicate and comforting, while yellow ceilings are luminous and stimulating. Orange walls are warm and luminous and brown floors are steady and stable.

2.4 Perceptual Fidelity and Presence in Virtual Environments

In a Virtual Environment (VE), efficient techniques are often needed to economize on rendering computation without compromising the information transmitted. It is not computationally feasible to immerse an observer into an interactive artificial environment which mimics the panoply and complexity of sensory experiences associated with a real-world scene. For a start, it is technologically challenging to control all of the sensory modalities to render the exactly equivalent sensory array as that produced by real world interaction (Billinghurst et al. 2002; Mania et al. 2003; Biocca et al. 2002).

In this section, previous studies and experiments addressing the issue of visual fidelity perception, as well as the controversial feeling of presence in VEs is discussed.

2.4.1 Fidelity Metrics for Computer Graphics Simulations

The mapping from the real world environment to the computer graphics environment is mediated by environmental or visual fidelity (Waller et al. 1998). The term visual fidelity refers to the degree to which visual features in the Virtual Environment (VE) conform to visual features in the real environment. The different levels of fidelity can be described as:

- *Physical Realism*: in which the synthetic scene is an accurate point-by-point representation of the spectral radiance values of the real scene.
- *Photorealism*: in which the synthetic scene produces the same visual response as the real scene even if the physical energy depicted from the image is different compared to the real scene.
- *Functional Realism*: in which the same information is transmitted in real and synthetic scenes while users perform visual tasks targeting transfer of training in the real world (Ferwerda 2003). Flight simulators provide an example of synthetic scenes which, although not physically accurate, nor photorealistic, are functionally realistic, since they transmit the same visual information to users as if flying a real plane. The goal is to engineer a simulator of high functional realism so that positive transfer of training is achieved when training in the simulators, meaning that task performance is better in the real-world compared to training in the simulator.

Interface or interaction fidelity refers to the degree to which the simulator technology (visual and motor) is perceived by a trainee to duplicate the operational equipment and the actual task situation. It is argued that training, for instance, in a VE with maximum fidelity would result in transfer equivalent to real-world training since the two environments would be indistinguishable (Waller, Hunt & Knapp, 1998).

Robust metrics are essential in order to assess the fidelity of VE implementations comprising of computer graphics imagery, display technologies and 3D interaction metaphors across a range of application fields. Apart from optimization of technological characteristics such as resolution, Field-of-View (FoV), latency, etc., one common belief is that efficient task performance measures should serve as fidelity metrics for any application that mainly targets transfer of training in the real world (Bailey & Witmer, 1994, Waller, Hunt & Knapp, 1998, Lathrop & Kaiser, 2002). A commonly employed strategy, therefore, for assessing the simulation fidelity of a VE is to compare task performance in a VE to task performance in the real world scene represented in the VE. Another common approach is to employ a cross-application construct, such as the sense of ‘presence’ to assess the effectiveness of a VE or aspects of a VE according to its success in enhancing presence. There is a widespread belief that presence should somehow improve task performance, although this has yet to be verified or indeed reasons offered as to why this should be the case (Stanney et al., 1998).

It is tempting to replicate the real world as accurately as possible in order to provide equivalent experiences (Liu et al. 2005). Whilst arguably ideal, it is not yet computationally feasible for this to occur. Trade-offs between visual/interaction fidelity and computational complexity should be applied to a simulation system without detracting from its training effectiveness (Mania et al. 2003, 2005; Wagner 1987 and Mourkoussis et al. 2010). There is, therefore, a call for efficient techniques assessing the fidelity of a VE and determine its relationship with performance in order to economize on rendering computation without compromising the level of information transmitted (functional realism) (Ferwerda 2003). Whilst existing techniques aim to address this issue, it can be argued that no one technique provides a comprehensive approach.

The first effort to compare real and simulated computer graphics static scenes side by side was attempted by Meyer et al. 1986. Radiometric values predicted using a radiosity rendering of a basic scene were compared to physical measurements of radiant flux densities in the real scene both of which were viewed through the back of a view camera. In a more recent approach, McNamara et al. 2000 described a method for measuring the perceptual equivalence between a real scene and static computer simulations of the same scene based on human judgements of lightness. Results showed that rendering solutions such as tone-mapping were of the same perceptual quality as a photograph of the real scene.

Perceptual fidelity is not necessarily equivalent to physical simulation. The ultimate goal, as often argued, is to create synthetic spaces that are going to induce a sense of ‘presence’ similar to the real world. This goal would not necessarily be achieved by accurately simulating real-world spaces and illumination. Building a VE system to match the human perceptual and motor systems is essential. Generally, for any given task or for any application that requires a high level of simulation fidelity and mainly targets, for instance, transfer of training in the real world, the ability to induce spatial awareness and impressions as in the real world could be significant for any task situation.

2.4.2 Perceived Presence in a Virtual Environment

What sets VE technology apart from its ancestors is that in VE systems users can receive a number of distinct multi-sensory stimuli (i.e., visual, auditory, haptic) which are

intended to provide a sensation of ‘natural’ interaction with the virtual world and, consequently, an illusion of being ‘present’ in a VE. ‘Presence’ generally, refers to the sense of being present in time or space in a particular location (Webster’s II Dictionary, 1984). In the world of media and emergent technologies such as video conferencing, high definition television and home theatre, presence is defined as the perceptual illusion of non-mediation (Lombard & Ditton 1997). An ‘illusion of non-mediation’ occurs when the user fails to perceive the existence of a medium in his/her communication environment and reacts as he/she would if the medium were not there. Presence in VEs can be explained as the participant’s sense of ‘being there’ in a VE; the degree to which the users feel that they are somewhere other than they physically are while experiencing a computer generated simulation (Shloerb 1995).

Varied perceived presence measurement ‘devices’ have been employed in literature. Loomis 1992 observed human response to events that in the natural world would provoke ‘reflex’ reactions. For example, if one is sitting in front of a screen and experiences a scene of a car moving towards him/her very fast, then he/she might be ‘forced’ to turn to the right or left, in order to avoid ‘collision’ responding to the moving image as if it was occurring in reality. Another way of measuring presence introducing a quantitative strategy was proposed by Schloerb 1995. This method is based on a user’s inability to discriminate between a real and a VE and proposed the addition of certain types of ‘noise’ to a real image until it is impossible to be distinguished from the virtual image. Slater et al. introduced a measure of presence based on self-report of ‘Breaks in Presence’ while a participant experiences a VE simulation (Slater & Steed 1998). Also, physiological measures as blood pressure and heart rate have been employed (Meehan 2001). According to Frederick Brooks, one of the “hot, open challenges” is to measure the degree of presence and its operational effectiveness (Brooks 1999).

The most common method for measuring presence is post-experiment self-reports, with questions such as those included in the Slater et al. questionnaire (Slater et al. 1998). These questions are associated with the notion of presence itself and not with any characteristics of the technology. Hence, it could be applied to the real world as well as to the fMRI display used in the experiment.

The feeling of presence in Virtual Environments (VE) could be described as the sense of being ‘there’; the degree to which the users feel that they are somewhere other than they physically are while experiencing a computer generated simulation (Barfield & Weghorst 1993, Slater & Usoh 1998). The ultimate goal of the presence research could be explained as finding the equation of presence that allows to trade off factors against each other, while still maintaining the same level of presence.

The inspiration for this study was given by Slater’s suggestion that feelings of presence in VEs could be checked as responding equally to “virtual” events as to real events at many levels – from physiological responses through to behavioural and cognitive responses, including what can be picked from EEG and fMRI (Slater 2004). Although there is no scientific evidence, it has been demonstrated that feelings of presence exist – a powerful application is in psychotherapy. There have been a number of studies to reflect this, using anxiety as a surrogate for presence, such as *fear of heights*, *fear of flying*, *arachnophobia*, *agoraphobia* and *burns treatment*.

Emmelkamp et al.'s study found evidence that Virtual Reality Exposure Therapy (VRET) is effective for participants with fear of heights and of flying (Emmelkamp et al. 2002). Also, Meehan et al.'s research found that participants had a higher self-reported sense of presence and a statistically higher change of heart rate while being in a stressful environment (Meehan et al. 2002).

There was a study designed to find differences between Virtual Reality Exposure (VRE) and Standard Exposure (SE) in the treatment of fear of flying. The results indicated that VRE and SE were essentially equivalent on standardized questionnaires, willingness to fly, anxiety ratings during the flight, self-ratings of improvement, and patient satisfaction with treatment and the treatment gains were maintained after a 6 and 12 months follow-up assessments (Rothbaum et al. 1996)

The first case report to demonstrate the efficacy of immersive computer-generated virtual reality (VR) and mixed reality (touching real objects which patients also saw in VR) for the treatment of spider phobia was made from Carlin et al. The outcome was assessed on measures of anxiety, avoidance, and changes in behavior toward real spiders. VR graded exposure therapy was successful for reducing fear of spiders, providing converging evidence for a growing literature showing the effectiveness of VR as a new medium for exposure therapy. (Carlin et al. 1997)

Data about the efficacy of virtual reality exposure (VRE) in the treatment of panic disorder with or without agoraphobia (PDA) were gathered by Botella et al. 2007. The study was a between-subject design with three experimental conditions (VRE group, in vivo exposure [IVE] group and waiting-list [WL] group) and repeated measures (pre-treatment, post treatment and 12 month follow-up). Thirty-seven patients meeting DSM-IV criteria for PDA participated in this study. The improvement achieved using virtual exposure was superior to a WL condition and similar to that achieved using IVE. The results supported the efficacy of VRE in the treatment of PDA at short and long term (Botella et al. 2007)

In a study performed by Hoffman et al, the results provide the first available evidence from a controlled study that immersive VR can be an effective non-pharmacologic pain reduction technique for burn patients experiencing severe to excruciating pain during wound care (Hoffman et al. 2008).

In this thesis we propose a truly interdisciplinary approach of measuring the simulation fidelity and subjective 'sense of being there' or presence of a simulation system by utilizing *neuroscientific data through fMRI imaging*. Experiments exploring the effect of lighting types on subjective impressions and spatial performance will indicate the relative 'fidelity' of a synthetic scene in relation to whether *cognitive (attention modeling)* or *neuroscientific information* acquired when exposed to non-photorealistic scenes resemble the cognitive/neuro patterns of behaviour identified when exposed to globally illuminated synthetic scenes.

3 Chapter 3 – Software Architecture and Development Framework

In this chapter, the software architecture and the framework used in the development process will be described in detail. The tools used to build several parts of this project's application, such as the Flash authoring environment, will also be analyzed.

3.1 Game Engine – Unreal Development Kit (UDK)

For the purposes of this project, the power of building and extending upon a framework was preferred to building from scratch. As already discussed, UDK is a powerful framework used mostly in creating computer games and visualization. Its built-in lighting system was a requisite feature, which addressed the needs for realistic lighting effects of this project.

UDK consists of different parts, making it act both like a game engine and a 3D authoring environment. It provides the necessary tools to import 3D objects, create and assign materials on objects that affect the lighting distribution, precompute lighting effects and import and use sounds and sound effects. It, also, allows the designed application to seemingly attach to Flash User Interfaces (UI). UDK can also be used to render the created VEs, as well as create and respond to events while navigating the synthetic scenes. UDK offers the ability to use both C/C++ and UnrealScript, which provides the developers with a built-in object-oriented programming language that maps the needs of game programming and allows easy manipulation of the actors in a synthetic scene.

The main components inside UDK are the Unreal Editor, which is used to create and edit VEs, handling all the actors and their properties located in the VEs, the Unreal Kismet, which allows for the creation of sequences of events and corresponding actions and Unreal Matinee, responsible for the animation of actors or real-time changes in the actors' properties.

3.1.1 Unreal Editor

The Unreal Editor is the tool inside UDK used to create and edit VEs. From within Unreal Editor, 3D objects, sounds, videos, textures and images can be imported to the Content Browser library and inserted in the VEs. Also, the Unreal Editor can create and assign materials to 3D objects, as well as alter lighting and rendering configurations (Figure 17).

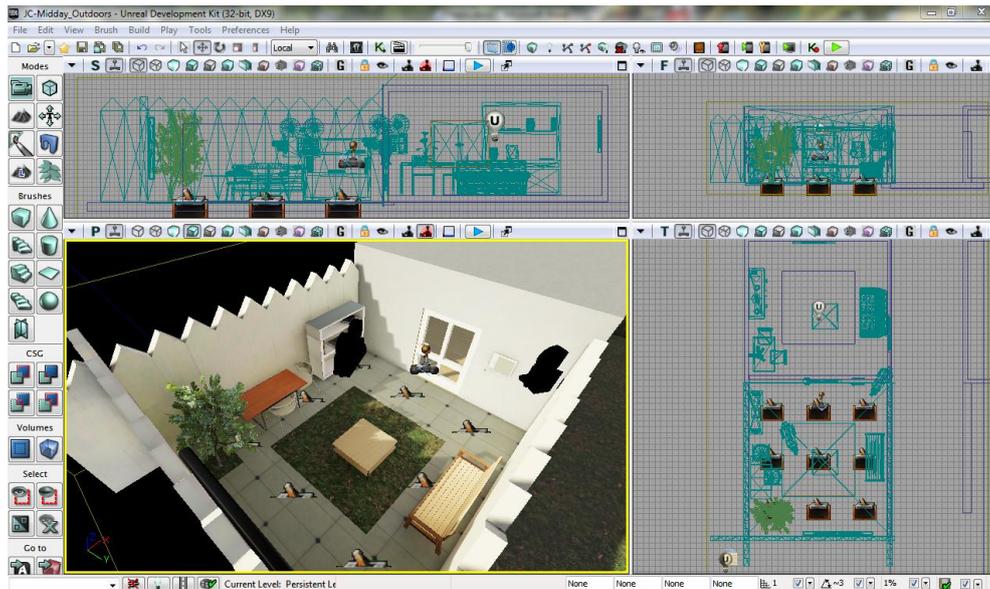


Figure 17: The Unreal Editor displaying a synthetic scene.

Actors, Lights and properties

Everything inside the virtual scene created in the Unreal Editor is considered by UDK to be an *Actor*, ranging from 3D objects to lights. This is in accordance with Unreal Script (see below), which is an Object-Oriented Programming language and every object is assigned to a class that extends from *Actor*. So, 3D objects are assigned to the *StaticMeshActor* class, lights can be variedly assigned to the *PointLight*, *PointLightToggleable*, *DominantDirectionalLight* classes according to their function, sounds are assigned to the *Sound* class, while all these classes extend from the *Actor* class.

The 3D objects imported into Unreal Editor can be assigned to *Static Mesh*, used for static objects, or *Skeletal Mesh*, used for character bodies. The 3D objects in this project were *Static Meshes*, as there weren't any characters used. After an object is imported through the Content Browser, we can change its main attributes, such as its collision box, materials, light map UVs and polygon count within the Static Mesh Editor. These changes will affect the instances of this object that will be inserted in the virtual scene, unless they are overridden. Figure 18 displays the Static Mesh Editor, displaying a 3D object. A representation of the 3D object, which depicts a bench, can be seen on the left, with the green bounding box indicating the collision vector assigned to it. On the right there are the properties of the object, such as the *light map resolution* and the *LODInfo*.

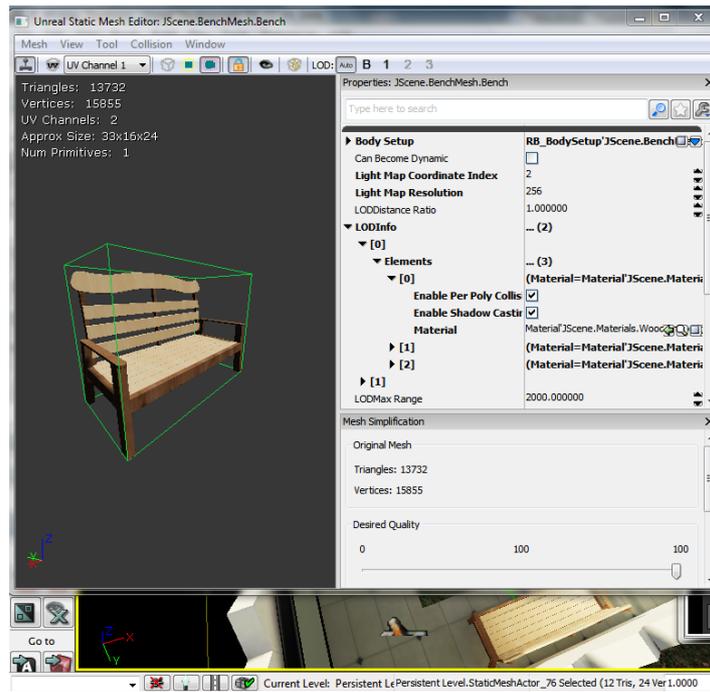


Figure 18: Static Mesh Editor for an imported object.

Once an object is inserted in the editor from the Content Browser library, an instance of its predefined *Actor* class is created and the editor offers the option to change the configuration of the specific instance, without affecting the other instances. This is true for all kinds of actors inserted in a scene, either being a light or any other possible Actor. The options that can be changed include the object’s position, draw scale, properties for the lighting system, materials, collision, components, etc. Figure 19 depicts the properties for the bench 3D object instance that was inserted in the VE. It shows that several properties can be changed, such as *Physics* and *Lightmass* properties, or the *Location* and *Rotation* of the object.

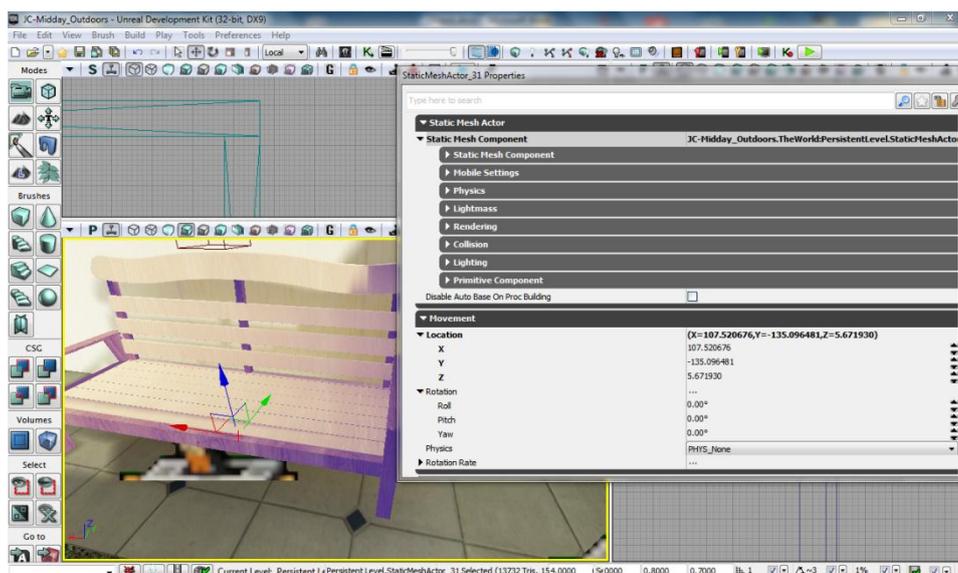


Figure 19: Properties of a Static Mesh Actor instance.

Light Actors

Lights are also considered to be *Actors* in the Unreal Editor. The type of *Actor* each light belongs to is different according to the light's function. In this project, two different light types were used, the *Dominant Directional Light* used to simulate the sun and a *Point Light Toggleable* used to simulate the indoors artificial light. Every light actor's properties can be changed as well, like a *Static Mesh Actor's*. Figure 20 displays the properties that can be changed for each *Light Actor*. The top part of the figure shows the properties of a *PointLightToggleable* light actor, used to simulate the indoors artificial light, which include the *Light Source Radius*, the *Indirect Lighting Scale* and *Saturation*, the light's *brightness*, and *colour*, etc. The bottom part of the figure shows the properties of a *DominantDirectionalLight*, used to simulate the sun light. These properties are the same with the ones for the *PointLightToggleable* actor, but they also include the option to change the direction of the light.

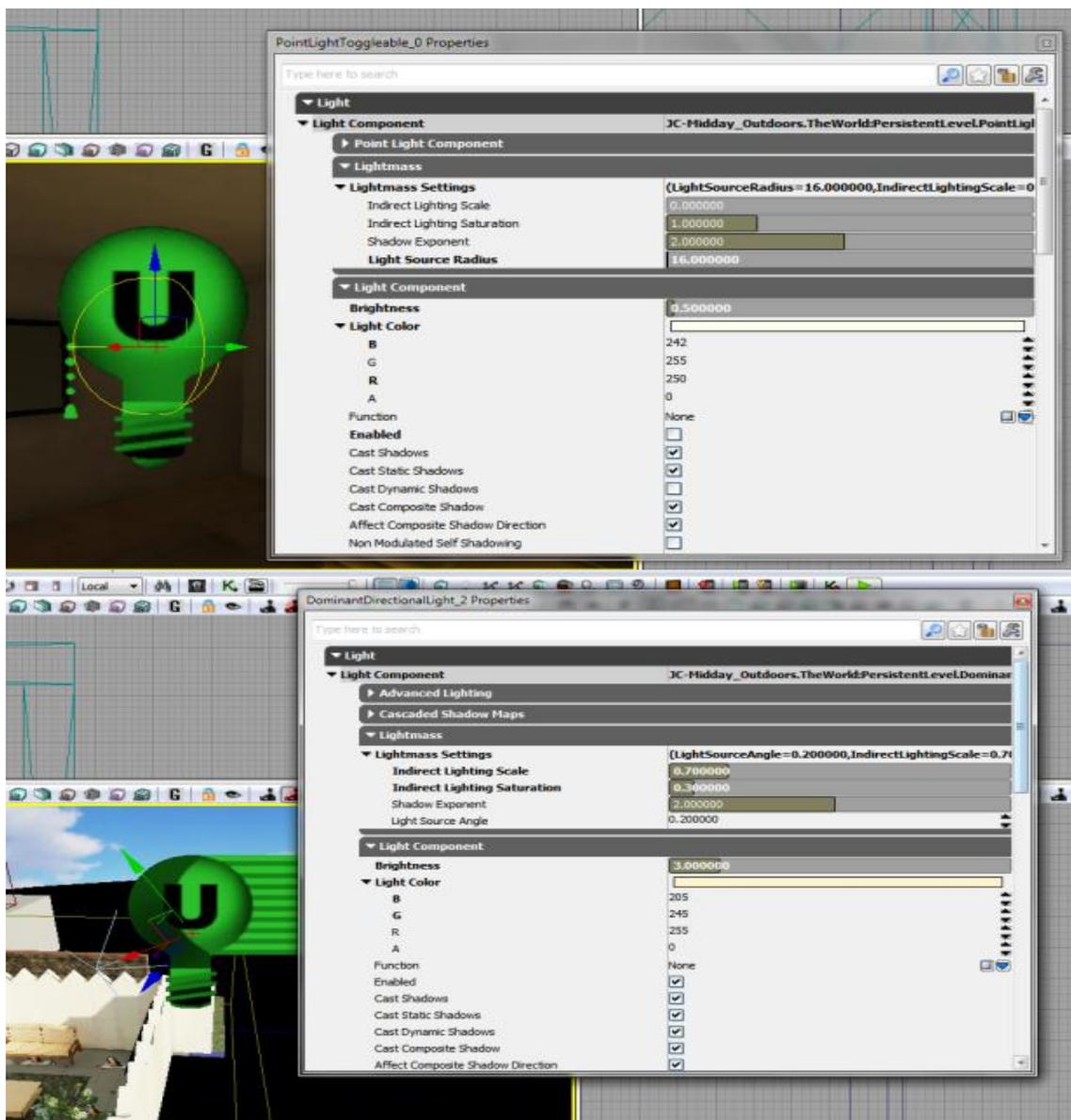


Figure 20: Light Actor properties: for a *PointLight Actor*, simulating the indoors artificial light (top) and for a *DominantDirectionalLight Actor*, simulating the midday sun (bottom).

There are many other actor classes that simulate the functions of a variety of other objects, such as triggers, sounds, dynamic meshes that can be moved or rotated, etc. All of these can be inserted and manipulated through the Unreal Editor.

Kismet

The Unreal Kismet is a tool inside the Unreal Editor and can be described as a graphical system that connects specific events to specific actions. It is node-based and properties of different nodes can be connected with arrows. There are some predefined events and actions, however, more can be created through Unreal Script, as described further below.

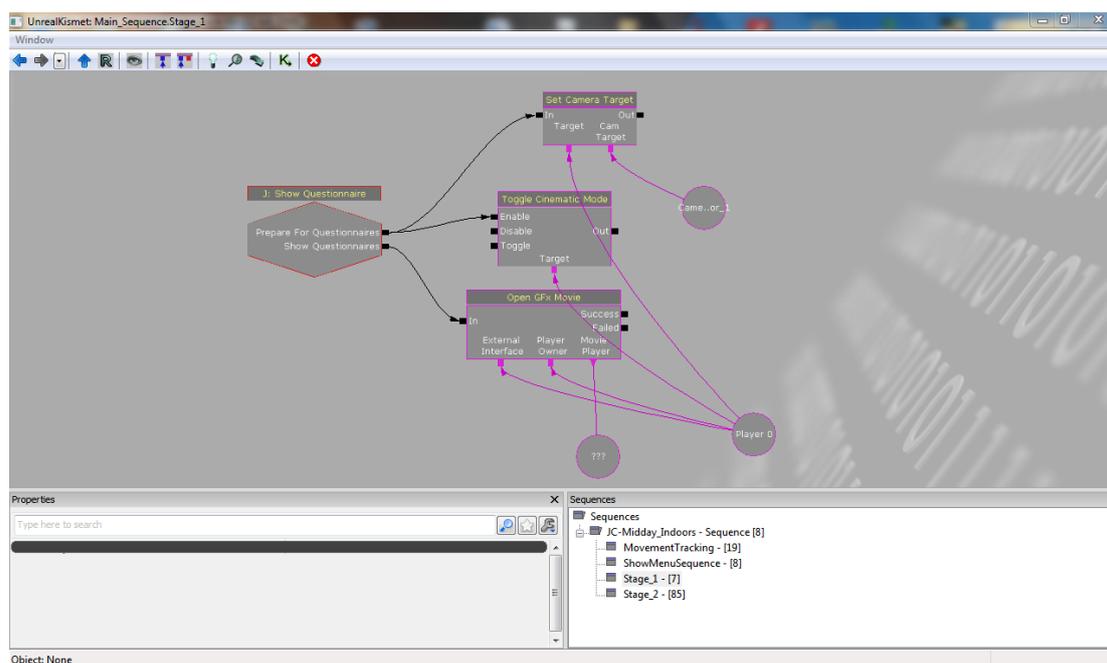


Figure 21: Unreal Kismet set up for the first stage of the experiment.

An example is shown in Figure 21, which depicts an event node containing two Unreal Script generated events, concerning the questionnaires that will be shown at the end of each scene during the first stage of the experiment, as detailed in Chapter 6. The first event is generated when the application should prepare for the questionnaires, therefore, it is linked to the actions necessary to move the participant’s viewpoint to a predefined location and restrict their movement. Afterwards, the next event is generated to signal the start of the administration of the questionnaires, therefore, it is linked to the actions needed to display the questionnaires on the screen inside the fMRI scanner. Variables can be created and assigned to events or actions, either primitive or actor instance variables, drawn as pink circles.

It is significant to note is that the complete sequence of events and actions applies only to the currently loaded scene and not to other scenes. So, Unreal Kismet is not efficient in creating general rules that apply to a complete game or application. In such cases, Unreal Script is recommended. Rather than that, Unreal Kismet is useful in connecting scene-specific events and actions.

Material Editor

A useful tool within Unreal Editor which is necessary in order to create realistic environments is the Material Editor. This tool handles the creation and editing of different materials that can be assigned to objects inside the scenes created. Materials affect the objects they are assigned to in a variety of ways, mainly in terms of their texture and their interaction with the light.

The Material Editor is node-based, much like the Unreal Kismet. However, its nodes do not represent events or actions, but textures, colors and several processing filters, such as addition of two different textures. The Material Editor provides the main node which has all the supported properties of the material, such as the diffuse, emissive and specular properties and each property can receive the output from a node or from a sequence of nodes. Figure 22 displays the Material Editor displaying a wooden material. It can be seen that a texture node, containing a wood texture, is connected to the *diffuse* and *specular* properties of the main node and another one is connected to the *normal map* property.

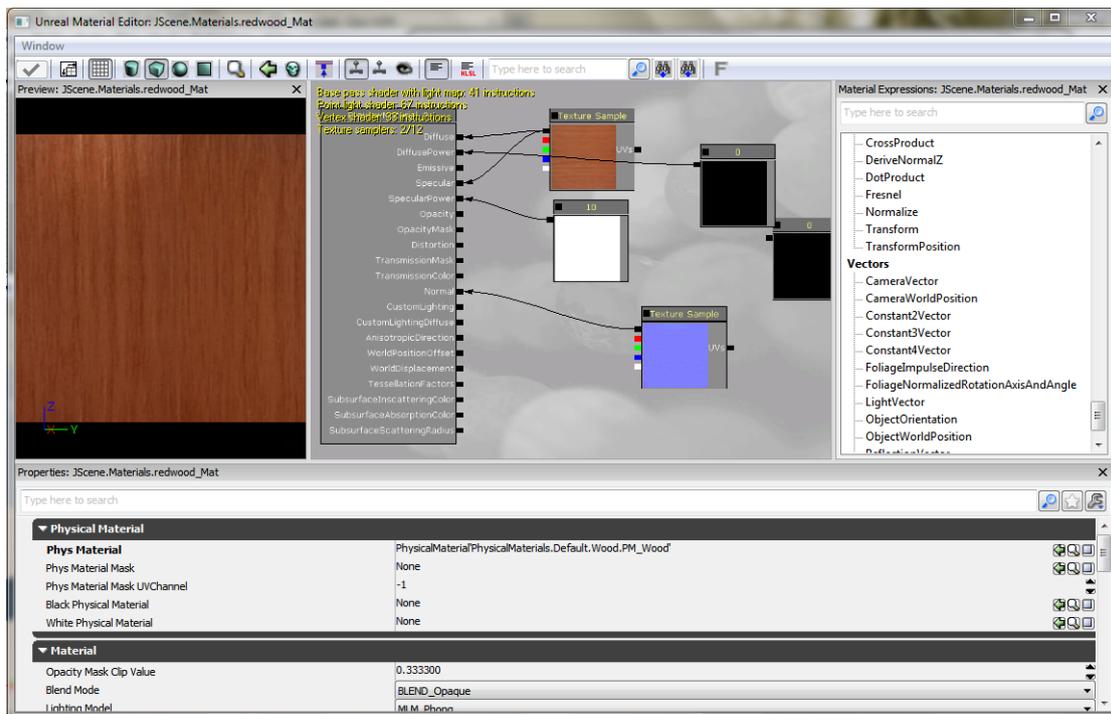


Figure 22: The Material Editor with a sample material loaded.

3.1.2 Sound Engine

The Unreal Editor supports its own sound engine and a Sound Editor provides the necessary tools, in order to create various sound effects. It supports immersive 3D location-based sounds and gives complete control over pitch, levels, looping, filtering, modulation and randomization.

As the rest of the Unreal Editor’s tools, the Sound Editor provides a node-based User Interface to import and use several sound cues, change their properties, mix them together

and channel the resulting output as a new sound effect. An example of the Sound Editor is shown in Figure 23, which depicts a *SoundCue* inserted in a synthetic scene. In the left part of the figure, the *SoundCue* residing in the asset library is shown. In the lower right the sound inserted in the synthetic scene is displayed. The Sound Editor window is shown in upper right part of the figure, which shows two *Sound* nodes connected to a *Mixer* node, which in turn is connected to the output channel.

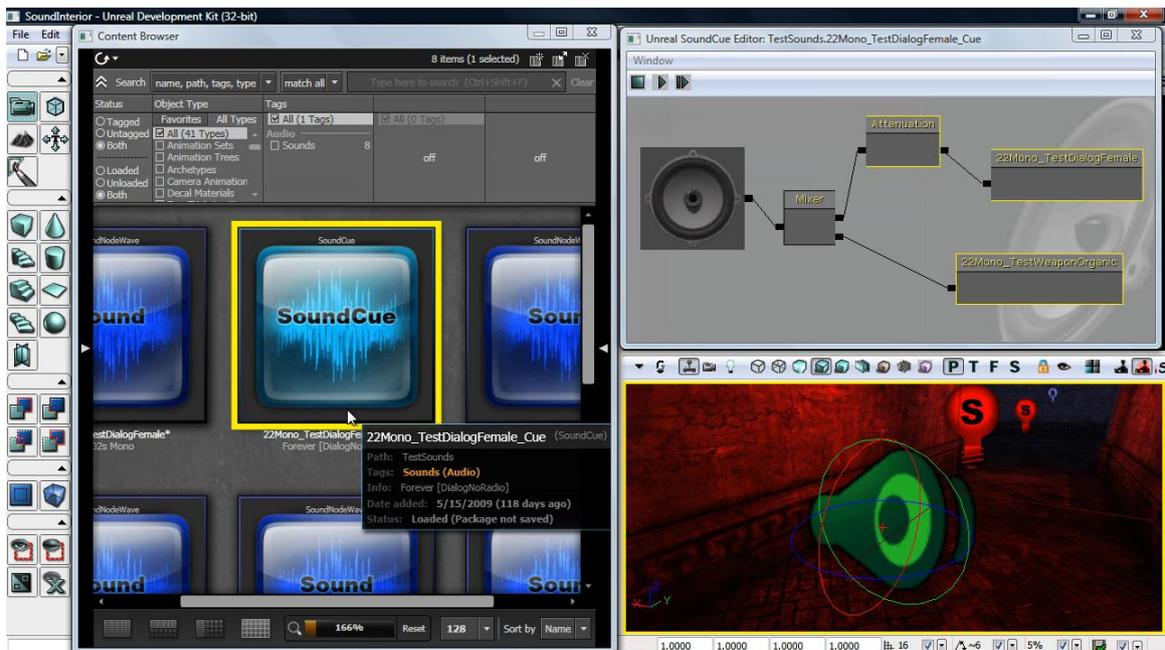


Figure 23: UDK's Sound Editor and a sound imported into a scene.

3.1.3 Configuration Files

Almost every property of the Unreal Editor and its components can be changed through the configuration files, which are bound to their respective UnrealScript class that implements each component. There are two versions of the configuration files, the first is the default, which is used to initialize the component and the other is the compiled and altered version. If, for any reason, the compiled version of a configuration file is corrupted or deleted, a new one is created based on the default one.

Every UnrealScript class that binds to a configuration file can define which of its properties should be saved and / or edited in the respective configuration file. So, when the Unreal Editor loads up and initializes its components, it uses the configuration files to recall their properties. The UnrealScript class bound to a configuration file can save the current state of its properties to the file, by calling the *SaveConfig* function.

An UnrealScript can bind to a configuration file by declaring it in its definition in the UnrealScript file. For example, the following class definition declares that it will bind to the configuration file "ExampleConfigFile" and the class properties that are defined to be set through configuration will be saved and loaded through that file:

```
class ExampleClass extends GameInfo config (ExampleConfigFile);
```

Each configuration file stores the information as key – value pairs, with each pair being the declared variables as keys and their respective values as the value of the pair. So, assuming that the ExampleClass included a variable definition and code as follows:

```
var config int myVariable;  
...  
myVariable = 3;  
SaveConfig();
```

Then, UDK would produce the following statement in the *ExampleConfigFile*:

```
myVariable = 3
```

3.1.4 DLL Files

UDK provides the option for an UnrealScript class to bind to an external DLL file, by declaring it in the class definition. For example, the following line declares that *ExampleClass* binds to the *ExampleDLL*:

```
class ExampleClass extends GameInfo DLLBind (ExampleDLL);
```

By binding to a DLL file, an UnrealScript class can call the declared in that DLL file methods or functions, which are written in C/C++. This proves to be an easy and efficient way to implement functions that either UDK does not support at all, such as I/O operations, or it would slow down the application, due to the fact that UnrealScript is slow.

A function residing inside the DLL must be declared in the UnrealScript class file and then it can be called exactly like it would be if it was an original UnrealScript function. Following the previous example and assuming that the *ExampleDLL* contained a function called *ExampleDLLFunction*, the code inside the UnrealScript class would be:

```
dllimport final function ExampleDLLFunction(); //function declaration  
...  
ExampleDLLFunction (); //function call
```

3.1.5 Input Manager

The input manager is responsible to handle the communication between the input hardware, such as keyboard, mouse, joystick or button boxes and the application. The input manager examines a configuration file based on DefaultInput.ini and according to it binds each input action, such as the joystick/mouse movement or key press to a specific method

designated to perform the selected action. The Unreal Editor comes with a default configuration file including a limited set of predefined bindings between buttons and methods, however, this file can be altered to match the needs of each application.

In order to create a new binding between a button press and the performed action or to change an already defined binding, this change must be reflected in the configuration file. Also, the method defined in the configuration file must exist in the UnrealScript code of the new application being developed.

For example, if we wanted to add or change the action performed when the Left Mouse Button is pressed, we would add or change these lines in the “DefaultInput.ini” configuration file:

```
.Bindings = (Name="LeftMouseButton", Command="GBA_Fire")  
.Bindings = (Name="GBA_Fire", Command="JProcessItem");
```

In the first line, we define that the press of the left mouse button corresponds to the game bindable action named *GBA_Fire*. In the next line, we define that if a game bindable action named *GBA_Fire* occurs, then the Input Manager should start the *JProcessItem* method, located in the application’s UnrealScript file. Inside that method we could effectively develop the application to perform whatever action is necessary to correspond to the press of the left mouse button.

3.1.6 Lighting and Rendering Engine

The Unreal Development Kit comes along with Gemini, a flexible and highly optimized multi-threaded rendering system, which creates lush computer graphics scenes and provides the power necessary for photorealistic simulations. UDK features a 64-bit color High Dynamic Range (HDR) rendering pipeline. The gamma-corrected, linear color space renderer provides for immaculate color precision while supporting a wide range of post-processing effects such as motion blur, depth of field, bloom, ambient occlusion and user-defined materials.

UDK supports all modern per-pixel lighting and rendering techniques, including normal mapped, parameterized Phong lighting, custom user-controlled per material lighting models including anisotropic effects, virtual displacement mapping, light attenuation functions, pre-computed shadow masks and directional light maps. UDK provides volumetric environmental effects that integrate seamlessly into any environment. Camera, volume and opaque object interactions are all handled per-pixel. Worlds created with UDK can easily feature multi-layered, global fog height and fog volumes of multiple densities.

It also supports a high-performance texture streaming system. Additionally, UDK’s scalability settings ensure that the application will run on a wide range of PC configurations, supporting both Direct3D 9 and Direct3D 11.

The Unreal Development Kit includes the Unreal Lightmass, which is an advanced global illumination solver. Unreal Lightmass supports the illumination with a single sun, giving off soft shadows and automatically computing the diffuse interreflection (color bleeding). It also offers a variety of options to optimize the illumination solution. It can provide detailed shadows by using directional light mapping, static shadowing and diffuse normal-mapped lighting. An unlimited number of lights can be pre-computed and stored in a single set of texture maps.

3.1.7 Unreal Lightmass

Unreal Lightmass is an advanced global illumination solver. It uses a refined version of the radiosity algorithm, storing the information in each illuminated 3D object's light map, while providing ray-tracing capabilities, by supporting Billboard reflections, which allows complex reflections even with static and dynamic shadows with minimal CPU overhead.

Unreal Lightmass is provided as part of the Unreal Development Kit (UDK) and it can only work on scenes created through it. Its performance is dependent on the complexity of the scenes created and the types of light emitting sources that exist in the scene. It is optimized to increase the renderer's performance.

Its main features include:

- *Area lights and shadows:* Within Lightmass, all lights are area lights by default. The shape used by Point and Spot light sources is a sphere. The radius of the sphere is defined by *LightSourceRadius* under *LightmassSettings*. Directional light sources use a disk, positioned at the edge of the scene. *Light source size* is one of the two factors controlling shadow softness, as larger light sources will create softer shadows. The other factor is distance from the receiving location to the shadow caster. Area shadows get softer as this distance increases, just like in real life.
- *Diffuse interreflection:* Diffuse Interreflection is by far the most visually important global illumination lighting effect. Light bounces by default with Lightmass, and the *Diffuse* term of each material controls how much light (and what color) bounces in all directions. This effect is sometimes called color bleeding. It's important to remember that diffuse interreflection is incoming light reflecting equally in all directions, which means that it is not affected by the viewing direction or position.
- *Mesh Area Lights from Emissive:* The *emissive* input of any material applied to a static object can be used to create mesh area lights. Mesh area lights are similar to point lights, but they can have arbitrary shape and intensity across the surface of the light. Each positive emissive texel emits light in the hemisphere around the texel's normal based on the intensity of that texel. Each neighboring group of emissive texels will be treated as one mesh area light that emits one color.
- *Translucent shadows:* Light passing through a translucent material that is applied to a static shadow casting mesh will lose some energy, resulting in a translucent shadow.

In this thesis, we will present a complete system based on the Unreal Development Kit which allows participants to interactively manipulate 3D scenes while in a fMRI scanner, able

to set the light of an indoors and outdoors scene in their preferred state offering choices between daylight (morning, mid-day and afternoon) and artificial light (fluorescent, incandescent, coloured) while immersed in a fMRI scanner detailed in Chapter 4 and 5. A complete neuroscientific experimental scenario has been implemented detailed in Chapter 6.

3.2 UnrealScript

UnrealScript was designed to provide the developers with a powerful, built-in programming language that maps the needs of game programming. The major design goals of UnrealScript are:

- Enabling time, state and network programming, which traditional programming languages do not address but are needed in game programming. C/C++ deals with AI and game logic programming, through events which are dependent on aspects of the object's state. This results in long-length code that is hard to maintain and debug. UnrealScript includes native support for time state and network programming which not only simplifies game programming, but also results in low execution time, due to the native code written in C/C++;
- Programming simplicity, object-orientation and compile-time error checking, helpful attributes met in Java are also met in UnrealScript. More specifically, deriving from Java UnrealScript offers:
 - A pointerless environment with automatic garbage collection;
 - A simple single-inheritance class graph;
 - Strong compile-time type checking;
 - A safe client-side execution "sandbox";
 - The familiar look and feel of C/C++/Java code.

Often during the implementation, design trade-offs had to be made, choosing between execution speed and development simplicity. Execution speed was then sacrificed, since all the native code in UnrealScript is written in C/C++ and resulted performance outweighs the added complexity. UnrealScript has very slow execution speed compared to C, but since a large portion of the engine's native code is in C only the 10%-20% of code in UnrealScript that is executed when called has low performance.

3.2.1 The Unreal Virtual Machine

The Unreal Virtual Machine consists of several components: The server, the client, the rendering engine, and the engine's support code.

The Unreal server controls all the gameplay and interaction between players and actors (3D objects, lights or sounds that can be inserted in a synthetic scene). A listen server is able to host both a game and a client on the same computer, whereas the dedicated server allows a host to run on the computer without a client. All players connect to this machine and are considered clients.

The gameplay takes place inside a level, containing geometry actors and players. Many levels can be running simultaneously, each being independent and shielded from the other.

This helps in cases where pre-rendered levels need to be fast-loaded one after another. Every actor on a map can be either player-controlled or script-controlled. The script controls the actor's movement, behavior and interaction with other actors. Actor's control can change in game from player to script and vice versa.

Time management is done by dividing each time second of gameplay into *Ticks*. Each *Tick* is only limited by CPU power, and typically lasts 1/100th of a second. Functions that manage time are really helpful for gameplay design. Latent functions, such as *Sleep*, *MoveTo* and more cannot be called from within a function but only within a state.

When latent functions are executing in an actor, the actor's state execution does not continue until the latent functions are completed. However, other actors may call functions from the specific actor that handles the latent function. The result is that all functions can be called, even with latent functions pending.

In UnrealScript, every actor is as if executed on its own thread. Windows threads are not efficient in handling thousands at once, so UnrealScript simulates threads instead. This means that 100 spawned actors will be executed independently of each other each *Tick*.

3.2.2 Object Hierarchy

UnrealScript is purely object-oriented and comprises of a well-defined object model with support for high level object-oriented concepts such as serialization and polymorphism. This design differs from the monolithic one that classic games adopted having their major functionality hardcoded and being non-expandable at the object level. Before working with UnrealScript, understanding the object's hierarchy within Unreal is crucial in relation to the programming part.

The main gain from this design type is that object types can be added to Unreal at runtime. This form of extensibility is extremely powerful, as it encourages the Unreal community to create Unreal enhancements that all interoperate. The five main classes one should start with are *Object*, *Actor*, *Pawn*, *Controller* and *Info*.

Object is the parent class of all objects in Unreal. All of the functions in the *Object* class are accessible everywhere, because everything derives from *Object*. *Object* is an abstract base class, in that it doesn't do anything useful. All functionality is provided by subclasses, such as *Texture* (a texture map), *TextBuffer* (a chunk of text), and *Class* (which describes the class of other objects).

Actor (extends *Object*) is the parent class of all standalone game objects in Unreal. The *Actor* class contains all of the functionality needed for an actor to be placed inside a scene, move around, interact with other actors, affect the environment, and complete other useful game-related actions.

Pawn (extends *Actor*) is the parent class of all creatures and players in Unreal which are capable of high-level AI and player controls.

Controller (extends *Actor*) is the class that defines the logic of the pawn. If *Pawn* resembles the body, *Controller* is the brain commanding the body. Timers and executable functions can be called from this type of class.

Info (extends *Actor*) is the class that sets the rules of gameplay. Players joining will be handled in this class, which decides which *Pawn* will be created for the player in the scene and which *Controller* will handle the behavior of the pawn.

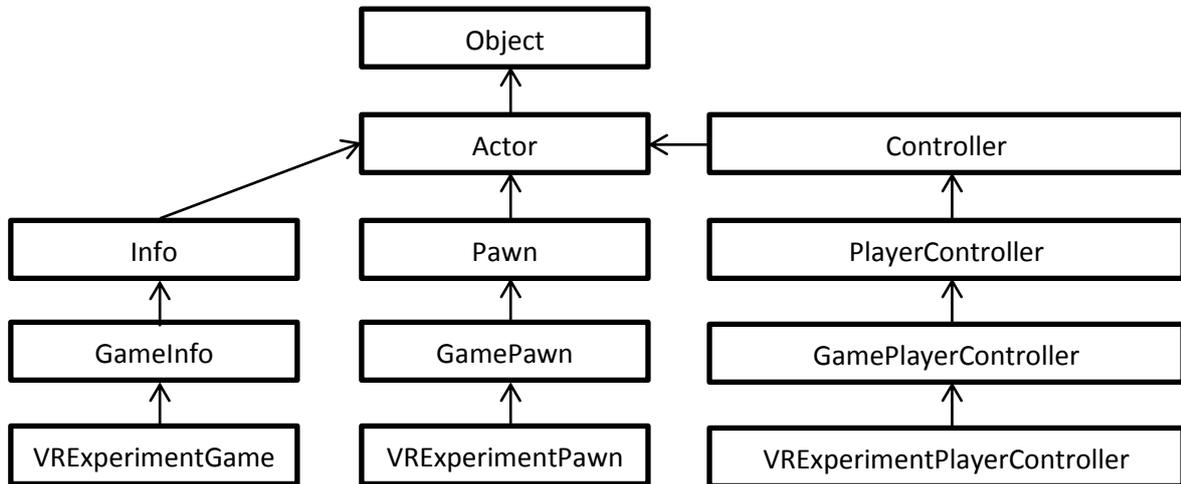


Figure 24: Class Hierarchy Diagram for 3 user-created classes: *VRExperimentGame*, *VRExperimentPawn* and *VRExperimentPlayerController*.

In the example shown in Figure 24, the class hierarchy for the three most important classes that control the application flow is depicted. As already described, the *VRExperimentGame* class decides how new players entering the scene are treated. The *VRExperimentPawn* class describes the properties and the behavior of a *Pawn* entering a scene that is handled by *VRExperimentGame*. Finally, the *VRExperimentPlayerController* class handles the *Pawn* in the scene, according to user input.

3.2.3 Timers

Timers are a mechanism used for scheduling an event to occur. Time management is important both for gameplay issues and for programming tricks. All Actors can have more than one timers implemented as an array of structs. The native code involving timers is written in C++, so using many timers per tick is safe, unless hundreds expire simultaneously. This would require the execution of UnrealScript code for each one of the timers and the heavy computational demands would lead to unwanted delay to the handling of the timers.

The following function starts a timer counting the time that the participant has available to navigate the scene and then the questionnaires show up in the screen:

```
SetTimer (FIRST_STAGE_TIME_SECONDS, false, 'showQuestionnaires');
```

This line of code defines that after `FIRST_STAGE_TIME_SECONDS`, signifying a specific amount of time, the function `showQuestionnaires` should be called. The false value passed as an argument means that this timer should not repeat the counting; it will only work once.

3.2.4 States

States are known from Hardware engineering where it is common to see finite state machines managing the behaviour of a complex object. The same management is needed in game programming, allowing each actor to behave differently, according to its state. Usually, when implementing states in C/C++ there are many switch cases used, based on the object's state. This method, however, is not efficient, since most applications require many states, resulting to difficulties in developing and maintaining the application.

UnrealScript supports states at the language level. Each actor can include many different states, however, only one can be active at any time. The state the actor is in reflects the actions it wants to perform. *Attacking*, *Wandering*, *Dying* are potential states a *Pawn* may acquire. Each state can have several functions, which can be the same as another state's functions. However, only the functions in the active state can be called. For example, if an application dictates that an action should only be performed in a specific stage, then this stage could be encapsulated in a different state that implements the function corresponding to that action differently than other states.

States provide a simple way to write state-specific functions, so that the same function can be handled in different ways depending on which state the actor is in when the function is called. Within a state, one can write special "state code", using the regular UnrealScript commands plus several special functions known as "latent functions". A latent function is a function that executes slowly (i.e. non-blocking), and may return after a certain amount of "game time" has passed. Time-based programming is enabled which is a major benefit that neither C/C++, nor Java offer. Namely, code can be written in the same way it is conceptualized. For example, a script can support the action of "turn the TV on; show video for 2 seconds; turn the TV off". This can be done with simple, linear code, and the Unreal engine takes care of the details of managing the time-based execution of the code.

3.2.5 Delegates

Delegates are a reference to a function within an instance. Delegates are a combination of two programming concepts, e.g. functions and variables. In a way, delegates are like variables in that they hold a value and can be changed during runtime. In the case of delegates, though, that value is another function declared within a class. Delegates also behave like functions, because they can be executed. It is this combination of variables and functions that makes delegates such a powerful tool under the right circumstances.

3.2.6 Interfaces

UnrealEngine3's UnrealScript has support for interface classes that resembles much of the Java implementation. As with other programming languages, interfaces can only contain function declarations and no function bodies. The implementation for these declared methods must be conducted in the class that actually implements the interface. All function types, as well as events, are allowed. Even delegates can be defined in interfaces.

An interface can only contain declarations which do not affect the memory layout of the class: enums, structs and constants can be declared. Variables cannot be declared for this reason.

3.2.7 UnrealScript Compiler

The UnrealScript compiler is three-pass. Unlike C++, UnrealScript is compiled in three distinct passes. In the first pass, variable, struct, enum, const, state and function declarations are parsed and remembered, e.g. the skeleton of each class is built. In the second pass, the script code is compiled to byte codes. This enables complex script hierarchies with circular dependencies to be completely compiled and linked in two passes, without a separate link phase. The third phase parses and imports default properties for the class using the values specified in the default properties block in the .uc file.

3.2.8 UnrealScript Programming Strategy

UnrealScript is a slow programming language when compared to C/C++. A program in UnrealScript runs about 20x slower than C. However, script programs written are executed only 5-10% of the time with the rest of the 95% being handled in the native code written in C/C++. This means that only the 'interesting' events will be handled in UnrealScript. For example, when writing a projectile script, you typically write a *HitWall*, *Bounce*, and *Touch* function describing what to do when key events happen. Thus, 95% of the time, a projectile script isn't executing any code and is just waiting for the physics code to notify it of an event. This is inherently very efficient.

The Unreal log may provide useful information while testing scripts. The UnrealScript runtime often generates warnings in the log that notify the programmer of non-fatal problems that may have occurred.

UnrealScript's object-oriented capabilities should be exploited as much as possible. Creating new functionality by overriding existing functions and states leads to clean code that is easy to modify and easy to integrate with other peoples' work. Traditional C techniques should be avoided, such as writing a switch statement based on the class of an actor or the state because code like this tends to clutter as new classes and states are added or modified.

3.3 Flash Applications as User Interfaces (UI)

The Unreal Development Kit supports Heads-Up Displays (HUD) that can be constructed in UnrealScript. In order to create a Graphical User Interface to be displayed on top of the regular viewport such as a questionnaire that displays a question and requires user response to it, however, this is inefficient. For this reason, UDK provides support to integrate a Flash application inside a scene and project it on top of the surface of a 3D object in the scene, or in the center of the screen.

A Flash application can be ideally used as a User Interface in UDK, because it incorporates the ability to display animated graphics, text or buttons on the screen on top of the scene being rendered. It can also receive user input and provide feedback according to it.

A Flash application consists of many frames, placed in the main timeline. The flow of the frames being displayed can be changed through ActionScript - Flash's scripting language, thus allowing to control which frame will be displayed next and when that will happen. Each frame can have its own set of graphics, texts, movie clips and buttons and a script controlling the behavior of the frame's components.

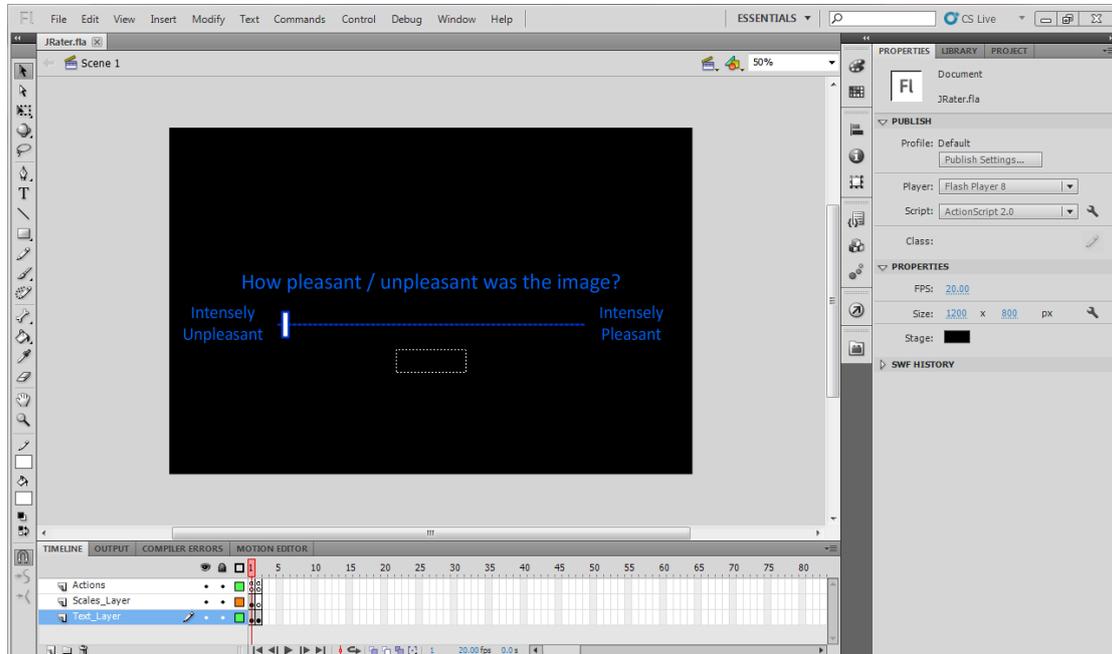


Figure 25: Flash Authoring environment displaying a Flash User Interface.

In the example shown in Figure 25, the Flash authoring environment is presented with a completed Flash User Interface loaded. The graphics and the movie clips inserted into the stage are shown. The scale bar shown can be adjusted by user input through ActionScript, which is a scripting programming language with a superset of the syntax and semantics of the more widely known JavaScript and is suited to the development of Flash applications. More details about Action Script can be found in section 3.3.2.

3.3.1 Authoring Environment for Interactive Content

In order to create a Flash application, a Flash authoring environment is necessary. There are many different Flash authoring environments available, however, the most powerful is Adobe Flash Professional CS5.5. Although it is not free, a 30-day trial version is available for download.

A freshly created Flash application is equipped with an empty stage and an empty timeline. Objects, such as movie clips, graphics, buttons, text, sounds or other Flash components, can be inserted into the application's library, or directly into the scene in the currently selected frame. Various different frames can be created, carrying different components inserted into each frame and the control of the application can be handled through ActionScript.

When the Flash application is fully developed and working, the authoring environment can compile the assets and the ActionScript comprising the application into an executable file in SWF format. Such files can be directly executed by a Flash player and also this is the format that UDK supports.

3.3.2 ActionScript 2.0

Although Flash Professional provides the tools to create applications running in all versions of ActionScript (up to 3.0) and of Flash Player (up to 10.3), UDK currently only supports the integration of Flash applications with ActionScript 2.0 (AS2) and Flash Player 8.

ActionScript is a scripting programming language and it is a dialect of ECMAScript, meaning it has a superset of the syntax and semantics of the more widely known JavaScript. It is suited to the development of Flash applications.

The language itself is open-source in that its specification is offered free of charge and both an open source compiler and open source virtual machine are available. It is often possible to save time by scripting something rather than animating it, which usually also enables a higher level of flexibility when editing.

ActionScript 2.0 primitive data types

The primitive data types supported by ActionScript 2.0 are:

- *String*: A list of characters such as "Hello World".
- *Number*: Any Numeric value.
- *Boolean*: A simple binary storage that can only be "true" or "false".
- *Object*: Object is the data type all complex data types inherit from. It allows for the grouping of methods, functions, parameters, and other objects.

ActionScript 2.0 complex data types

There are additional "complex" data types. These are more processor and memory intensive and consist of many "simple" data types. For AS2, some of these data types are:

- *MovieClip* - An ActionScript creation that allows easy usage of visible objects.
- *TextField* - A simple dynamic or input text field. Inherits the MovieClip type.
- *Button* - A simple button with 4 frames (states): Up, Over, Down and Hit. Inherits the MovieClip type.
- *Date* - Allows access to information about a specific point in time.
- *Array* - Allows linear storage of data.
- *XML* - An XML object
- *XMLNode* - An XML node
- *LoadVars* - A Load Variables object allows for the storing and send of HTTP POST and HTTP GET variables.
- *Sound*
- *NetStream*
- *NetConnection*

- *MovieClipLoader*
- *EventListener*

3.3.3 Connection of the User Interface (UI) with the Application

The integration of a Flash application inside a scene in UDK requires that it should first be compiled into an SWF file and imported inside the UDK asset library. Afterwards, either UnrealScript or Unreal Kismet can initiate the Flash application, interact with it, hide it or instruct it to stop playing.

While a Flash application is playing inside a scene, UnrealScript can initiate a call of an ActionScript function and vice versa. This feature allows full interaction between the Flash interface and the application. Consequently, it is easy and efficient to create an application that initiates a Flash interface whenever it is required and then receive the user's response and order it to stop playing.

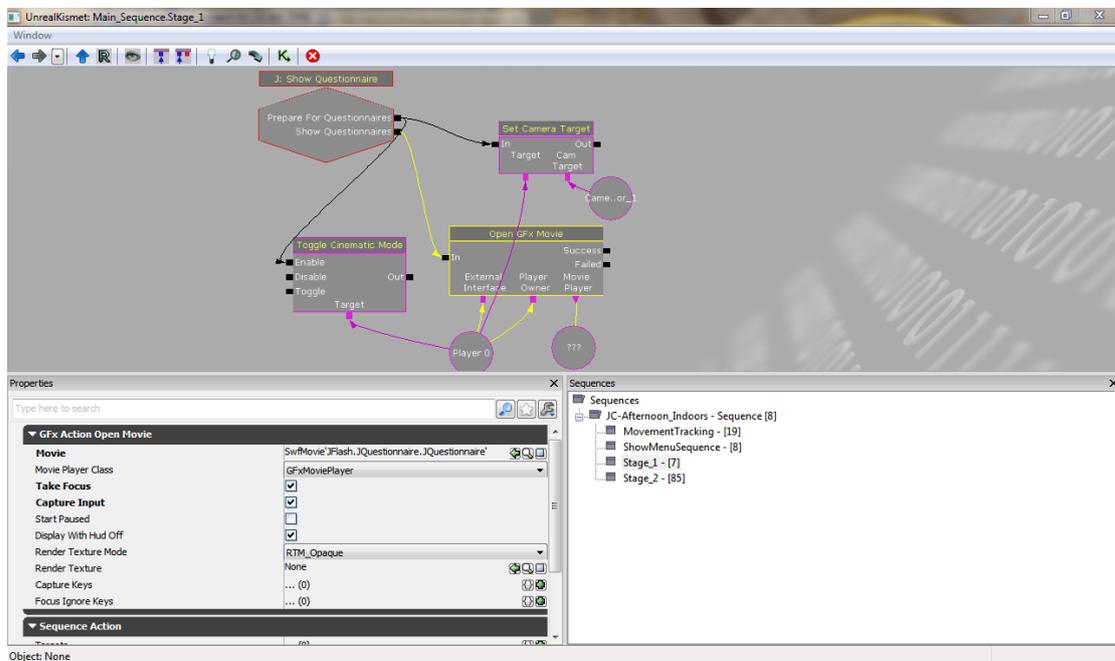


Figure 26: Initiation of a Flash User Interface Application through Unreal Kismet.

An example can be seen in Figure 26, which shows the action *Open Gfx Movie* firing up when it receives the *Show Questionnaires* event. This action performs the required operations to start the Flash Movie that is inserted as an argument in its properties. Additional setting may include whether a movie can capture user input, as well as where to project this movie, either on the screen or on an Actor's surface.

Also, Unreal Kismet provides the action *Close Gfx Movie* which handles the termination of the selected Flash application.

4 Chapter 4 – Implementation

In this chapter, the implementation and development of the complete application developed to be utilized in order to explore the effect of lighting and rendering fidelity on users' subjective impressions of lighting as communicated by self-report and fMRI clinical imaging will be described. More specifically, the way the interactive 3D virtual scenes were designed in order to simulate both outdoor and indoor artificial lighting will be analyzed. Experimental design issues, which were pre-requisite in order to conduct fMRI experiments enabling the analysis of brain data, will be presented.

The lighting system was implemented in order to conduct rendering fidelity experiments inside an fMRI scanner, comparing user responses and brain data utilizing photorealistic as well as wireframe scenes. The main features required were the ability to render a photorealistic or wireframe simulation of a virtual scene on an fMRI display, navigate it and interact with it while immersed inside an fMRI scanner. It was essential that the user placed in the fMRI scanner would have the ability to manipulate the lights of the scene in real-time. Every operation had to be fine-tuned and synchronized with the fMRI scanner in order for it to happen at a specific time, simultaneously acquiring brain imaging information.

4.1 Creating the 3D Virtual Scenes

In order to create the 3D scenes that the application would render, several steps were required, from creating the actual 3D objects placed in the scenes to importing them inside UDK and placing them in a synthetic scene, as well as creating and assigning the appropriate materials to these objects. These steps will be further explained below.

4.1.1 Creating the Visual Content

A five by five meters room connected through a door to an equal sized yard was chosen as the rendered displayed environment, allowing for the simulated light effects of a sun and of an artificial light inside the room to be seen when the viewpoint is set indoors and outdoors respectively. The indoor room and the outdoor yard were designed to be as similar as possible, in terms of the 3D objects placed inside them. Wireframe versions of the photorealistic scenes were created by using wireframe-specific materials.

The 3D models populating the scenes were created or downloaded from 3D models' repositories and were placed in a scene with the help of an industry-standard 3D modeling software (3D Studio Max 2011). The final result can be seen in Figure 30, which depicts the completed scene including the objects placed in it.

Before the 3D objects could be exported, two UV channels had to be set for each one of them. The first channel was designed to reflect the way a texture wraps up the 3D model and the second channel laid down the surfaces of the object for the computation of the lighting effects on the object later in UDK.

The room and the yard had to be similar in terms of the objects placed in them in order to fairly compare user responses when navigating either the indoor or the outdoor space, so every 3D object placed inside the room had its respective counterpart in the yard, as can be seen in Table 1.

Indoors Room Objects	Outdoors Yard Objects
Walls with ceiling	Fence
Wooden floor	Tiled floor
Carpet	Grass
Sofa	Bench
Television	Television
Coffee Table	Coffee Table
Entry phone	Entry phone
Bookcase with books	Bookcase with books
Table with chairs	Table with chairs
Plant in pot	Tree

Table 1: Objects existing in the indoors room and their respective counterparts in the yard.

Figure 27 shows the 3D representation of a bench object, taken from different viewpoints in 3ds Max. The object has only a basic gray material applied on it, since the actual material used for this object was assigned when the scene was imported in UDK. This was necessary, as UDK does not support the import of materials created in 3ds Max. The geometry of the object was exported to the UDK and could then be used as an actor.

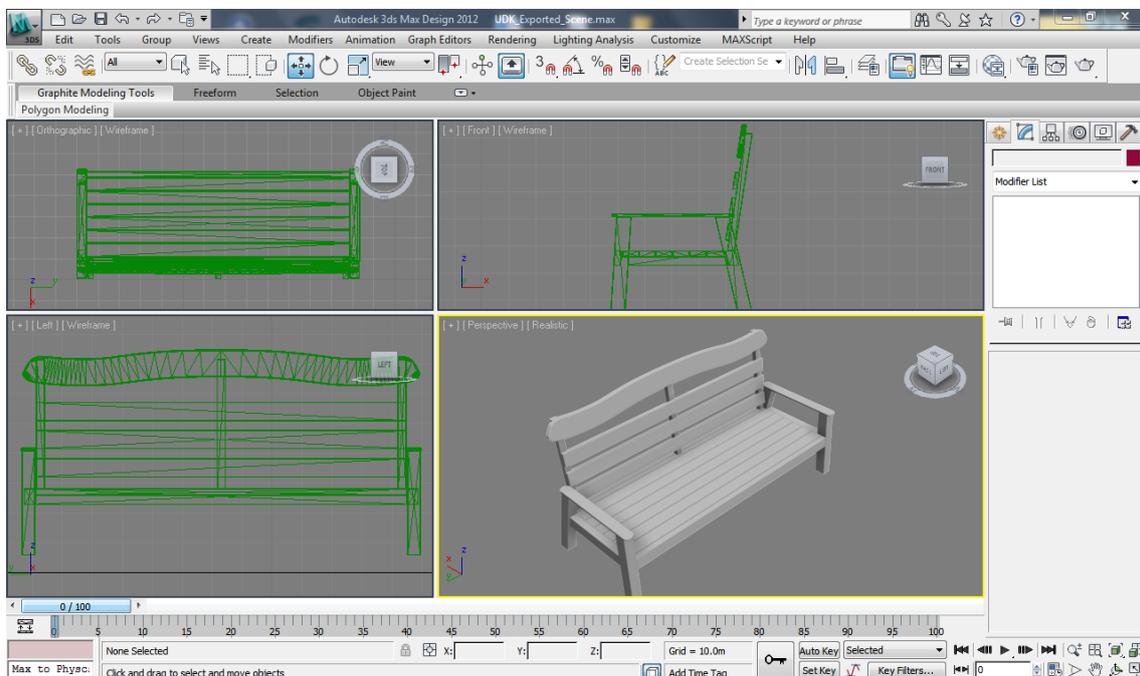


Figure 27: A 3D object representation of a bench. The 3D object was created in 3ds Max and its geometry exported to UDK.

In order to create and assign a material to an object with correctly assigned textures, as well as realistically illuminate the object imported in UDK, there must be two UV channels defined for each 3D object in 3ds Max. The first UV channel will be used to apply and align a material onto the object and can be created by applying a UVW map modifier to the editable mesh representing the object, as seen in Figure 28.

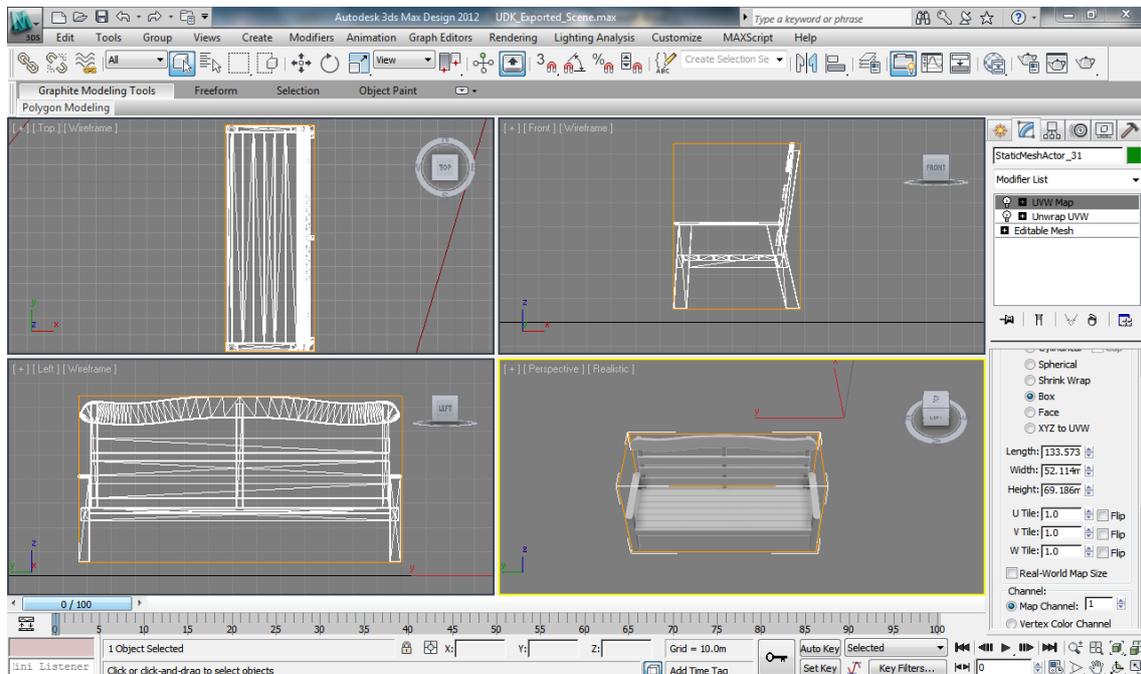


Figure 28: The UVW mapping of an object in 3ds Max. This UV channel is used by UDK in order to apply a material correctly onto the object.

The second UV channel is needed by UDK in order to correctly apply illumination and shading effects on the object, assigning more detail on the larger polygons of the channel. This channel can be created in 3ds Max by applying a UVW unwrap modifier on the editable mesh that will be exported and then, in the respective editor, select to automatically flatten the UVW mapping. An example of this can be seen in Figure 29.

Chapter 4 – Implementation

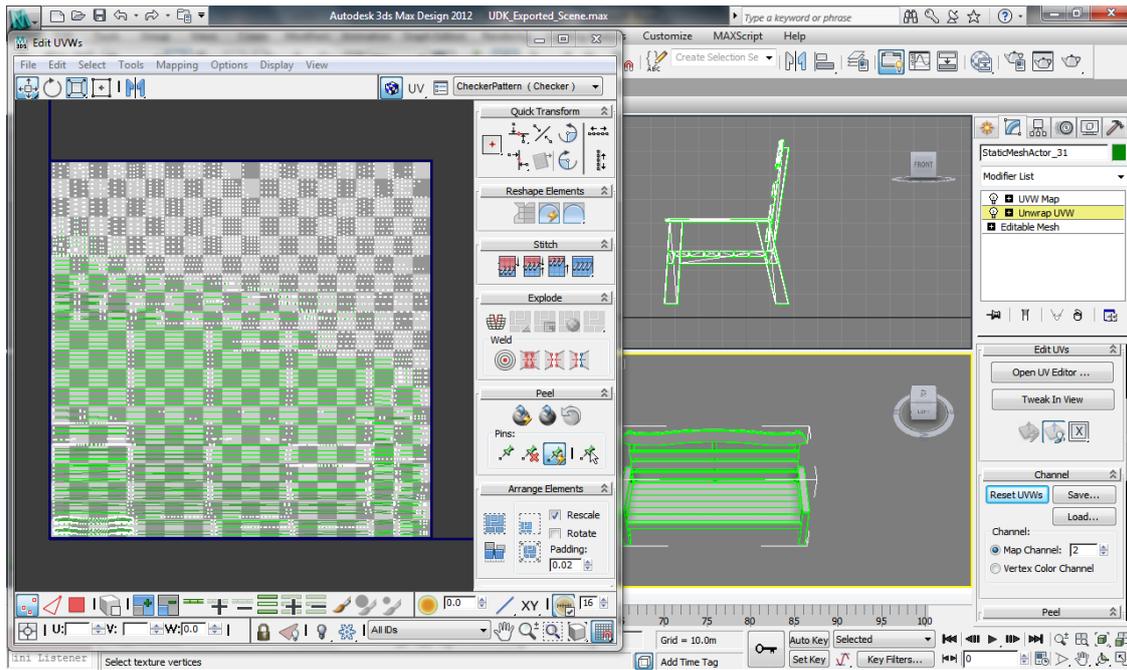


Figure 29: The UVW unwrapping of an object in 3ds Max. This UV channel is used by UDK in order to realistically illuminate an object, assigning more shading detail in the larger polygons of the object in the channel.

Figure 30 displays the final scene, including the 3D objects created in 3ds Max. The lower right part of the figure shows the 3D representation of all the objects, as seen from above. The rest of the screens display a 2D wireframe representation of the objects, taken from different viewing angles. Each object is painted with a different color, in order to make them more distinguishable.

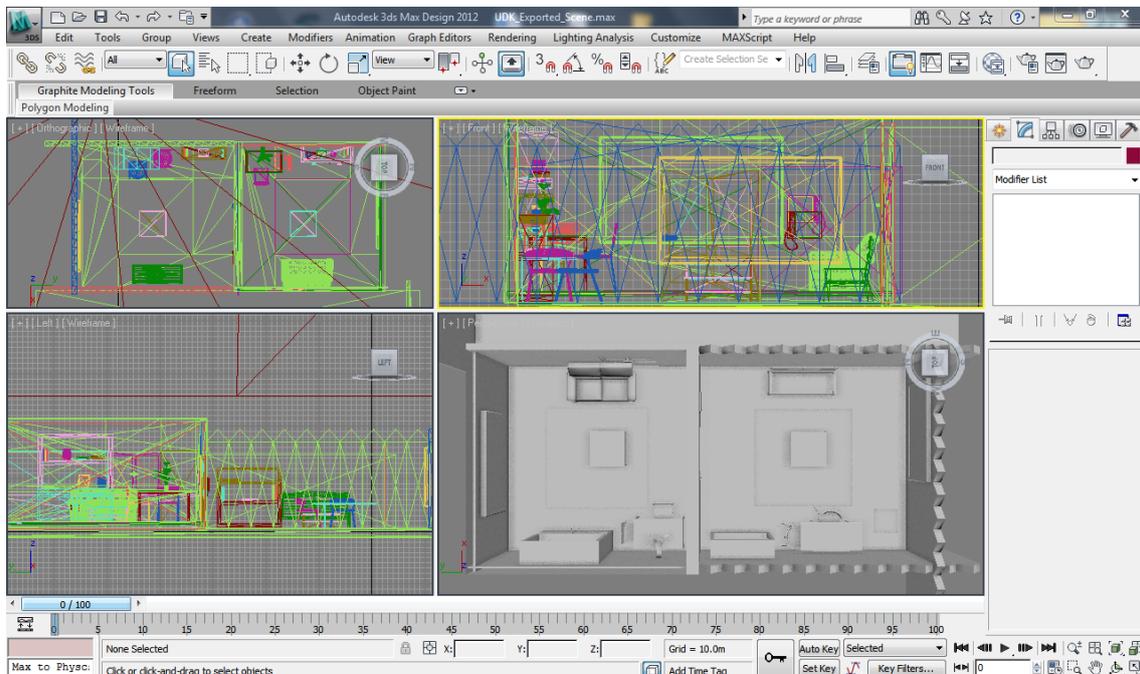


Figure 30: The final scene with all the 3D objects created in 3ds Max.

4.1.2 Setting up the Scene in UDK

Inside UDK, the created 3D objects were imported, by selecting the import option in the asset library of the Unreal Editor. UDK reads the file containing the exported 3D objects and recreates the geometry of the objects. It also creates the different material slots for each part of the 3D object and initializes the UV channels of the object.

The collision detection requirements of the experiments are automatically performed by UDK, after assigning a collision vector to each 3D object, according to its geometry. UDK provides several different ways of creating a collision vector for a given geometry. The more complex collision vectors produce more realistic collision detection, however, they suffer from increased need for computation. For the purposes of the experiments, the collision detection mechanism was required to simply prevent the participant from intersecting other objects, or passing through walls. So, simple collision vectors were chosen, thus, decreasing the computational needs.

An example of the creation of a collision vector is presented in Figure 31, which shows the – imported in the UDK – bench 3D object opened in the Unreal Editor. The green box surrounding the geometry of the object is the collision vector that was created for that object. Although the geometry of the actual 3D object is quite complex in terms of polygon count, the collision vector is extremely simple, represented by a bounding box. This does not offer great accuracy in collision detection, however, it serves the purpose of the experiment being computationally efficient.

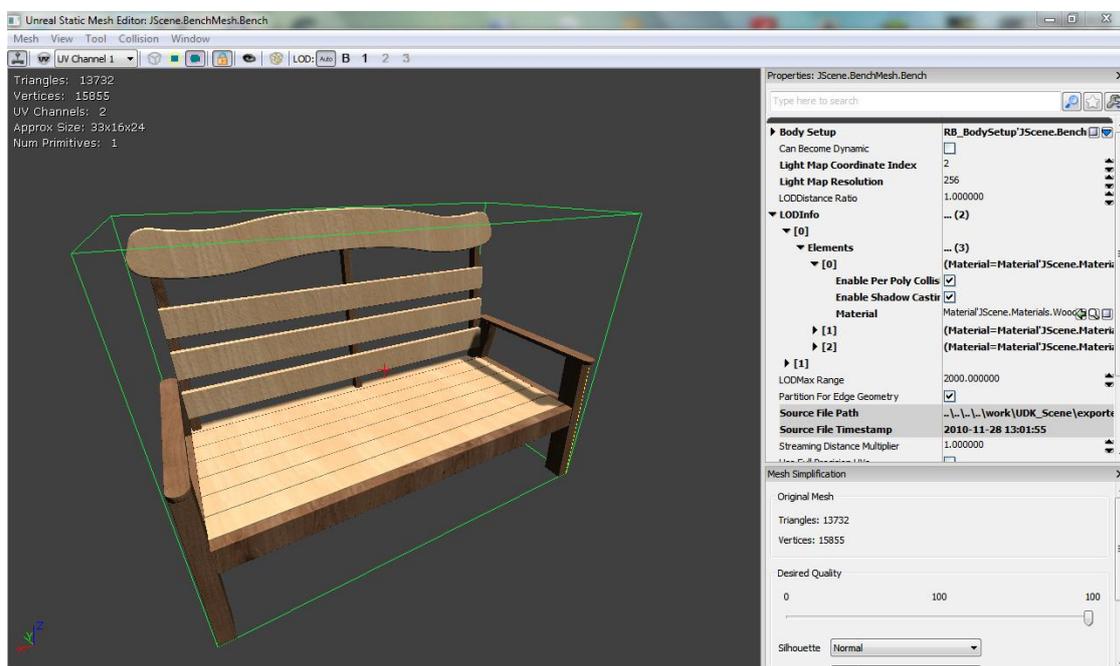


Figure 31: The imported in UDK geometry of the bench 3D object, with a simple collision vector applied on it.

The imported models were placed inside a new virtual scene. Then the texture images for each object were imported and materials were created and assigned to the objects in the scene. As already mentioned, UDK’s Material Editor offers the ability to not only set a diffuse texture or expression for a material, but also alter its properties, such as setting the specular

Chapter 4 – Implementation

color and power of the material, or defining a normal map, which can realistically differentiate the lighting of a rough (not smooth) surface.

For example, Figure 32 shows 2 different materials in the Material Editor, applied on a sample sphere object. The first material on the left is a matte grey one, with no specular properties and consequently does not produce any lighting effects on the object's surface. The material on the right has a white color connected to its specular property, with high specular power (20.0), so it produces a specular effect due to the light that the object receives.

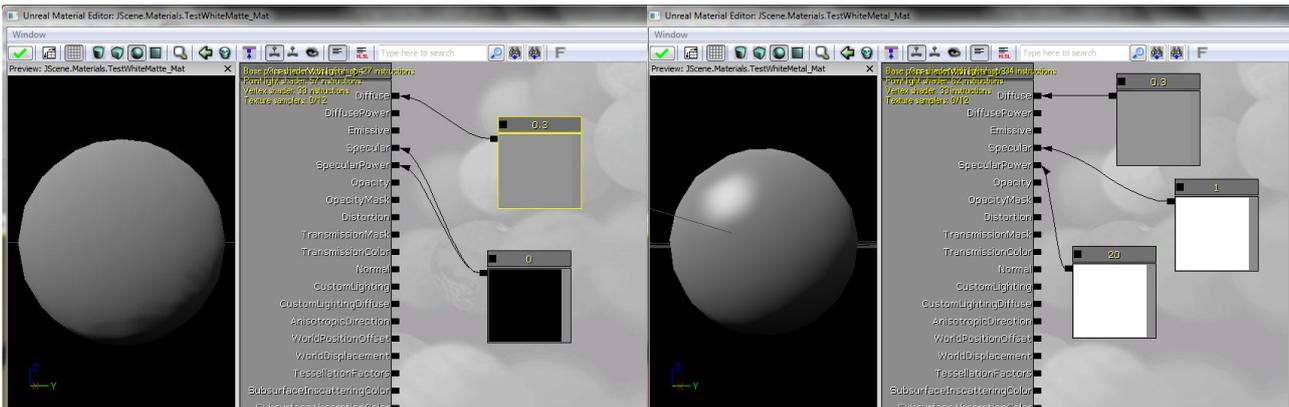


Figure 32: On the left is a grey matte material, with no specular color. On the right is the same grey material with white specular color.

There is, also, the option provided to set up a normal map for a material, which helps Lightmass to calculate the light that bounces on the object when each specific material is applied and scatter it according to the supplied normal map. This is very helpful in order to create the illusion of rough surfaces on objects with smooth geometry, such as the wooden surface of a table, or a carpet.

The geometry of a wooden table is extremely complex. When creating a normal map for the wooden material, although the geometry's surface is smooth, the light should scatter as though bouncing on a rough surface, as can be seen in Figure 33.

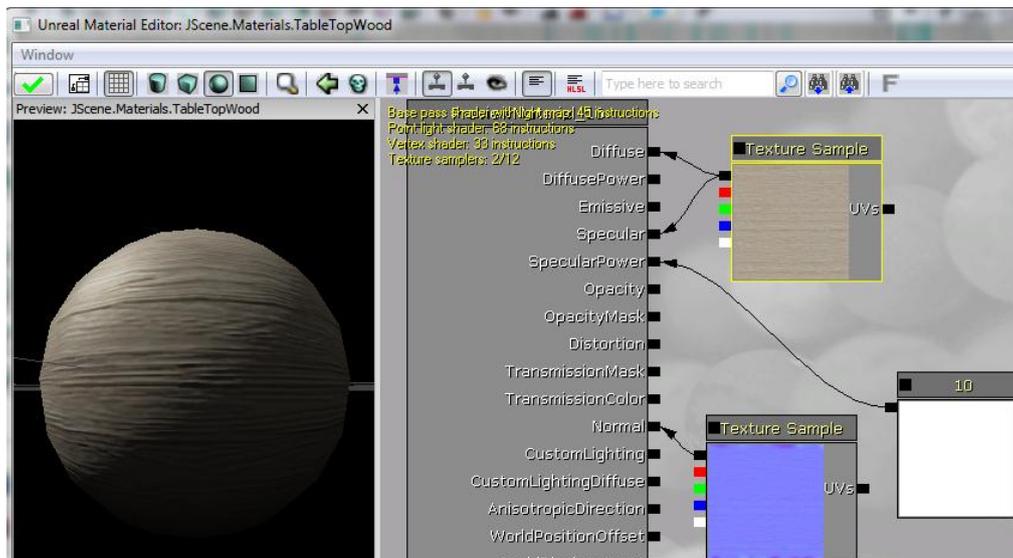


Figure 33: A material for the surface of a wooden table with a normal map defined. As can be seen from the sample sphere with the material applied on it, the lighting effects seem as though falling on a rough surface.

This is a very efficient way of creating the illusion of rough surfaces and realistic lighting effects while keeping the triangle count of the 3D objects at low levels. This is demonstrated in Figure 34, where the normal-mapped wooden material was applied on a simple box object, of only 12 triangles, representing the surface of a table. As can be seen, the lighting effects seem very realistic, just as a very complex – in terms of triangle count – geometry would produce.



Figure 34: A table surface object with the normal mapped material applied on it. Although the table's geometry is simple, with only 12 triangles, the lighting effects give the illusion of a rough (and complex) surface.

For the experiments, it was required that the lighting effects of the sun light could be visible from the indoors room through a door with glass windows. This means that the materials assigned to the windows should be transparent, representing a glass material. Also, they should allow the light to come through and illuminate the room. UDK supports translucent materials by offering the option to set a material as such, thus allowing the light to pass through it.

The diffuse and emissive properties of the translucent material affect the color of the light that passes through the material. There are different levels of translucency supported defined by the *opacity* property of the material; the higher the opacity value of a material, the less light it allows to pass through. There is an example displayed in Figure 35, showing the glass material used on the glass window.

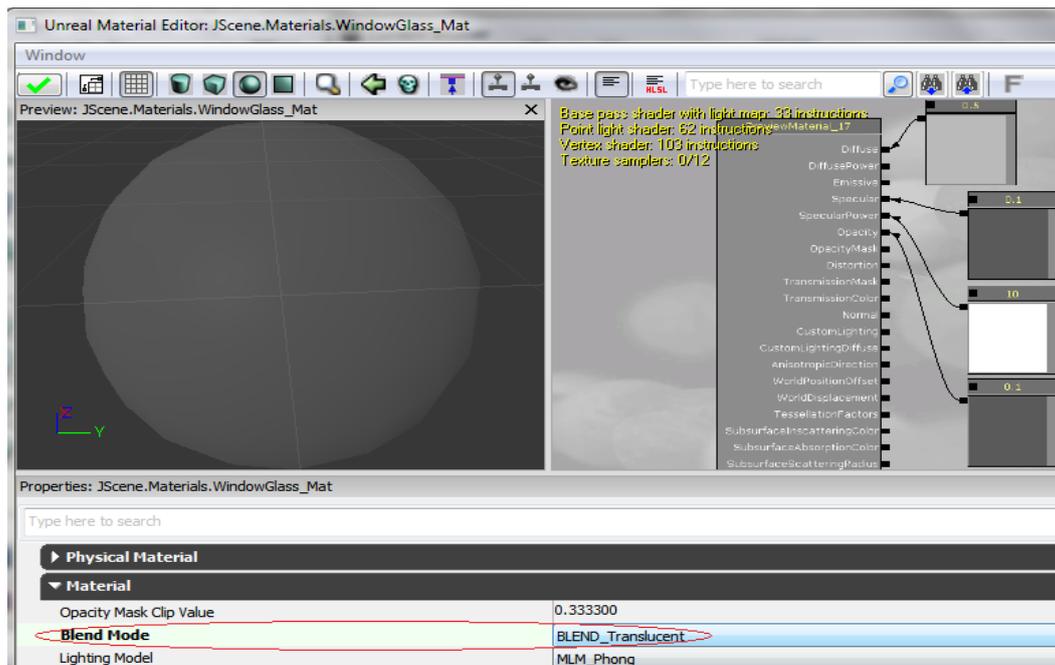


Figure 35: A translucent material representing the glass in a window.

The resulting scene including the 3D objects placed inside it is shown in Figure 36. The synthetic scene is still far from being considered as photorealistic, since there are no lighting effects present. For this reason, the lighting of the scene must be configured by Lightmass which allows executes the lighting computation.



Figure 36: An unlit scene with the modeled 3D objects in UDK. The created materials are applied to the respective objects.

In order to create the wireframe version of the scene, new materials were created similarly colored as the original ones for each object. The scene was rendered as wireframe and the color set for the wireframe rendering of each object was acquired from the diffuse

property of each material. The walls, the floor and the ceiling were not rendered as wireframe, but colored with the average color of the original diffuse property of the respective material.

A wireframe object is the 3D object for which only its edges are painted and displayed, while its surface is not rendered. For every object that was displayed as wireframe in the experiments, its material had its wireframe option enabled in the Material Editor, while the rest of the object was set to be translucent. An example wireframe material can be seen in Figure 37, showing the wireframe material applied on the top of a table which was otherwise photorealistically rendered to simulate a wooden surface.

The wireframe option of the material forces the renderer to paint only the edges of the displayed object, while the translucent option of the material makes the rest of the surface of the object translucent. So, it is only possible to see the edges of the object just as seeing an invisible object wrapped up with wires. Also, it is important to note that the color of the edges is the exact same color as specified in the original realistic material of the object, since it is connected to the diffuse property, as well as the specular property of the original material.

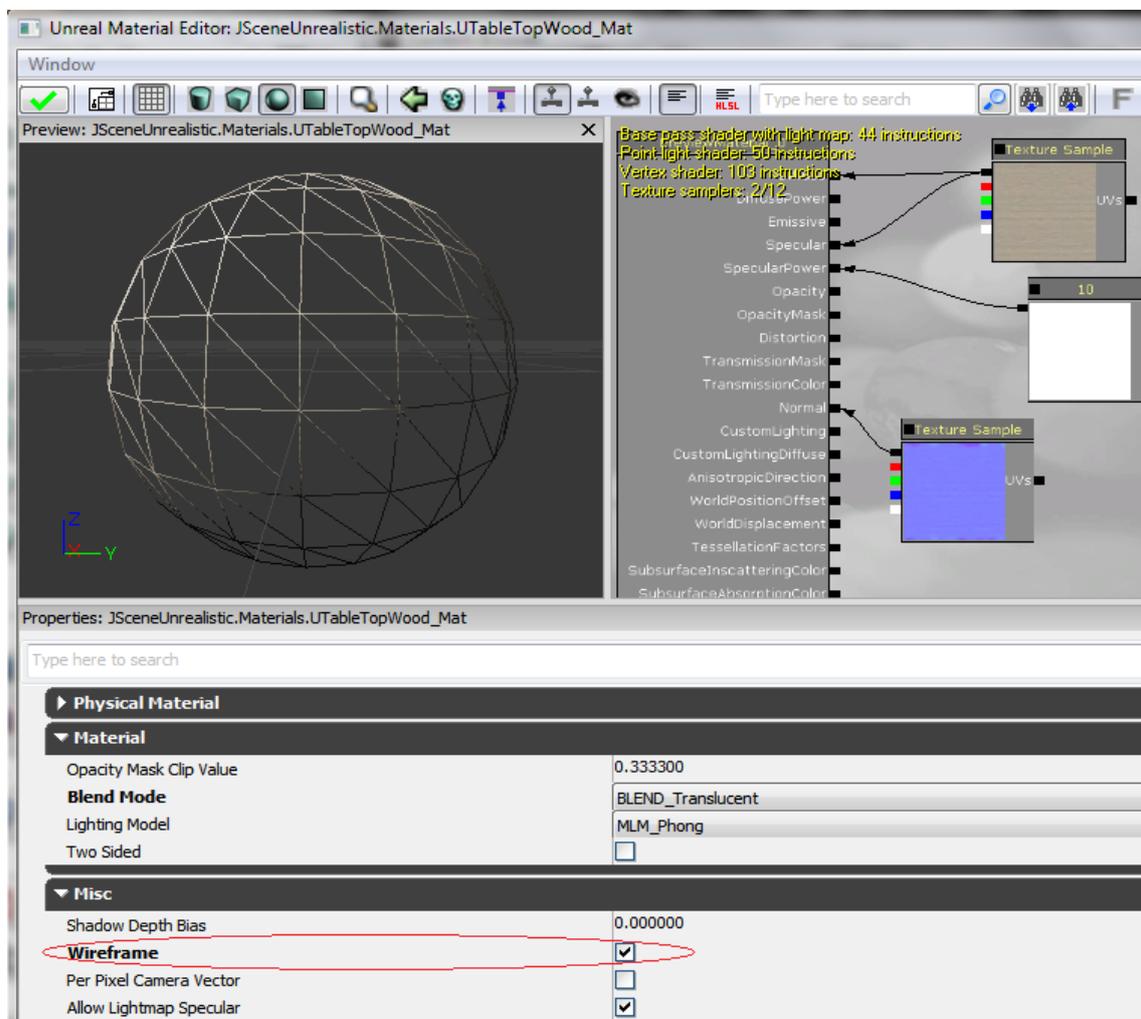


Figure 37: The wireframe version of the material for the wooden surface of a table.

Similarly, all the objects were transformed to wireframe objects for the wireframe version of the scenes required for the experiments. The issue that came up by transforming the objects to wireframe was that there weren't any surfaces rendered in the wireframe scenes for the lighting effects to still be visible. The edges that were displayed were not communicating adequate visible information in order to create subjective impression of lighting effects, thus, not making it possible to distinguish between the different lighting situations due to different time of day.

It was necessary to make the lighting effects visible in both the realistic scenes and the wireframe ones, for comparison purposes when conducting the experiments. In order to address this issue, the surface of the floor, the ceiling, the walls and the fences were not turned to wireframe. Instead, a different material was applied on them removing any textures they had and setting the diffuse property to the average color of the original diffuse expression.

An example of the materials created for the wireframe scenes and the difference between the original material and the material intended for the wireframe scenes can be seen in Figure 38(realistic material) and Figure 39 (non-realistic material).

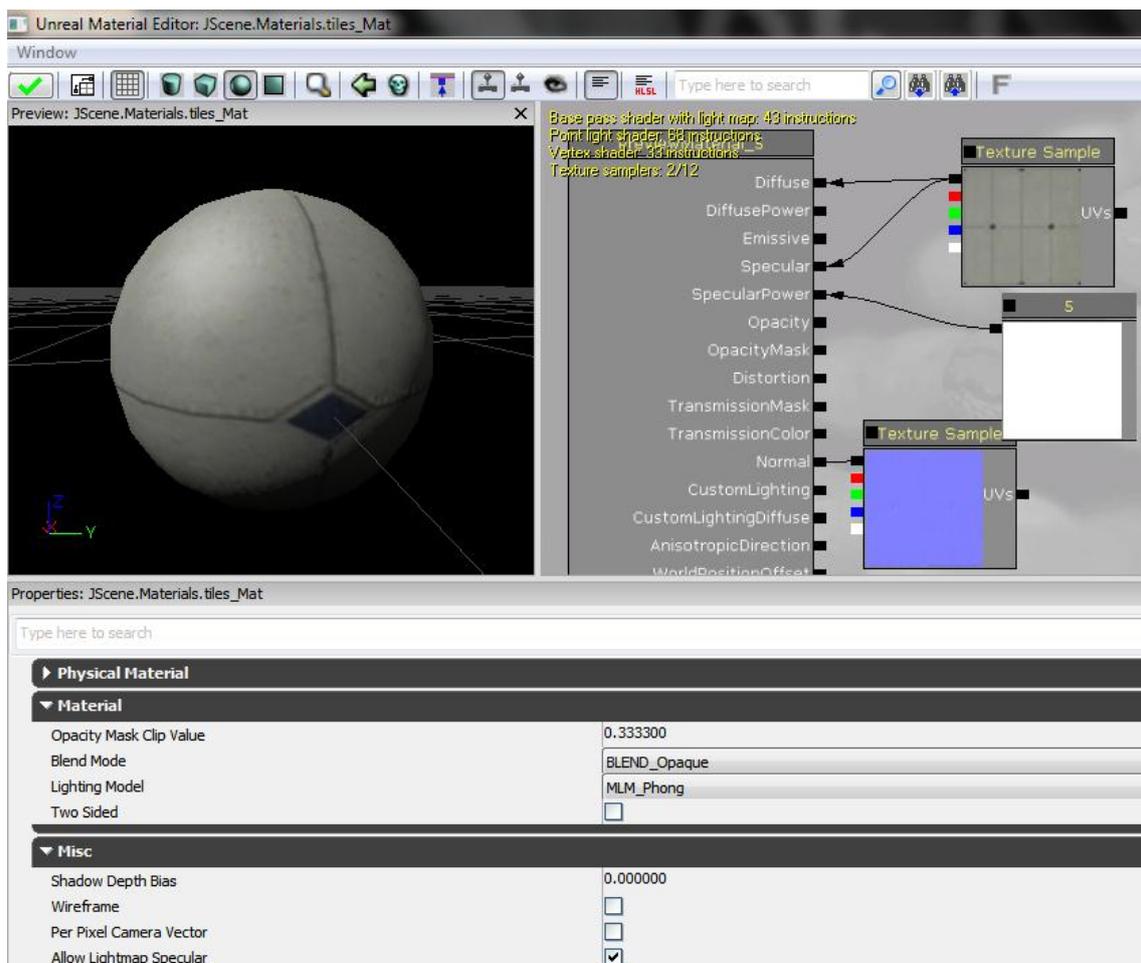


Figure 38: The original realistic material applied to the tiled floor in the yard.

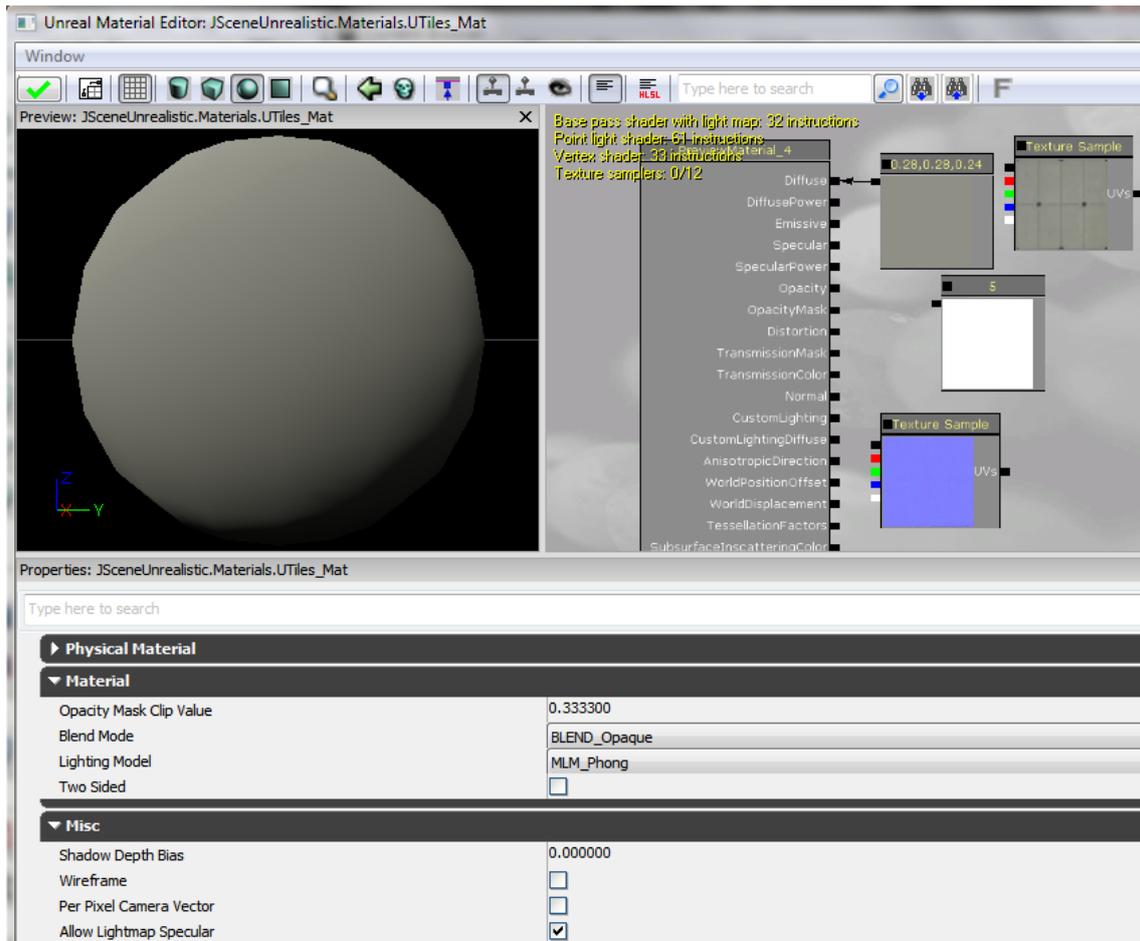


Figure 39: The non-realistic material applied on the tiled floor in the yard in the wireframe scenes. The diffuse texture is replaced by its average color. The normal maps are also removed.

In this way, every material was turned to either wireframe as applied to the 3D objects and to flat colour as applied to the surfaces on floors and walls. Then these new materials were applied to the respective 3D objects as placed in the original scene. The final *unlit* wireframe version of the scene is displayed in Figure 40.



Figure 40: The unlit wireframe version of the original scene in UDK.

4.2 Lighting Configurations

The virtual scenes constructed during the previous step are unlit and thus they are not photorealistic. Lighting effects greatly enhance the photorealism of the synthetic scenes. UDK's Lightmass was used to pre-compute photorealistic lighting effects for outdoor day-time as well as indoor artificial lighting.

The experimental methodology described in Chapter 6 required computing the lighting effects of four lighting conditions simulating a different time of day, i.e. morning, midday, afternoon and evening. Moreover, an artificial light was placed in the center of the indoors room's ceiling. Its illumination effect around the room both in terms of brightness, colour and type of lighting was programmed to change in real-time based on user interaction, when immersed in the fMRI scanner.

These requirements were accomplished by creating four versions for each of the unlit scenes, based on lighting configurations required as detailed above. The time of day affected the lighting for both indoors and outdoors space of the scene. The light actor simulating the sun was configured to be different in each condition in terms of color, brightness and direction.

4.2.1 Morning Lighting Configuration

In order to create the lighting effects simulating the morning sun, there was a *DominantDirectionalLight* Actor inserted into the UDK indoors and outdoors scenes, with settings configured as shown in Figure 41. Lightmass was instructed to simulate three indirect lighting bounces. Figure 41 shows the lighting settings, e.g., the *Indirect Lighting Scale*, set to be 1.5, the *Indirect Lighting Saturation*, set as 0.7, the *Brightness*, which was set to be 2.0 and the *RGB Color*, initialized as Red=255, Green=255 and Blue=235. The aforementioned settings, as well as the location and direction of the *DominantDirectionalLight* representing the sun, were based on 3ds Max 2011's representation of a *Daylight System*, set to simulate the morning sun, at 9:00 in the morning.

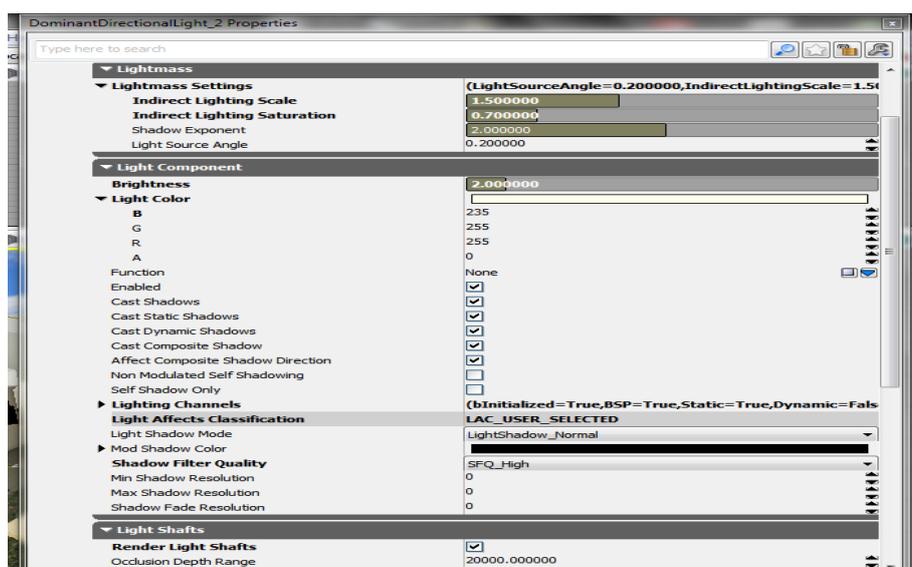


Figure 41: Morning sun light configuration. The most important settings are *Indirect Lighting Scale* and *Saturation*, *Brightness* and *Light Color* (as an RGB value).

The final scenes, following the computation of the lighting effects by Lightmass are shown in Figure 42 and Figure 43.



Figure 42: Screenshot of the Morning Outdoors Scene (left) and the respective wireframe version (right).



Figure 43: Screenshot of the Morning Indoors Scene (left) and the respective wireframe version (right).

4.2.2 Midday Lighting Configuration

The *DominantDirectionalLight* actor simulating the sun for the midday indoors and outdoors scenes was configured to be brighter than the morning sun, while Lightmass was instructed to simulate three bounces of indirect lighting. The whole configuration settings can be viewed in Figure 44. Figure 44 shows the lighting settings, e.g. the *Indirect Lighting Scale*, set to be 0.7, the *Indirect Lighting Saturation*, set as 0.3, the *Brightness*, which was set to be 3.0 and the *RGB Color*, initialized as Red=255, Green=245 and Blue=205. These settings, as well as the location and direction of the *DominantDirectionalLight* representing the sun, were based on 3ds Max 2011’s representation of a *Daylight System*, set to simulate the midday sun, at 12:00.

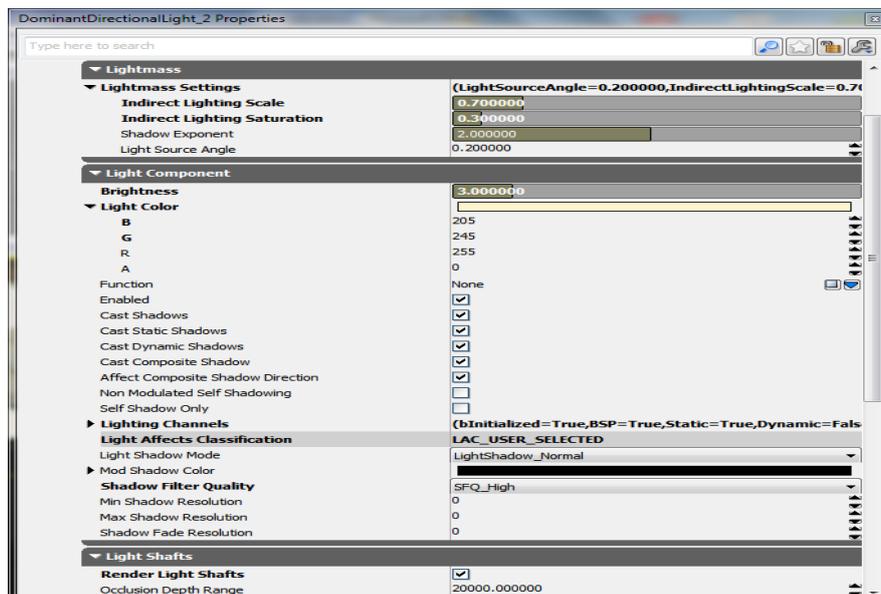


Figure 44: Midday lighting configuration.

The resulting scenes of the Lightmass algorithm computation are shown in Figure 45 and Figure 46.



Figure 45: Screenshot of the Midday Outdoors Scene (left) and the respective wireframe version (right).



Figure 46: Screenshots of the Midday Indoors Scene (left) and the respective wireframe version (right)

4.2.3 Afternoon Lighting Configuration

For the afternoon scenes, the sun was simulated to be less bright than in Morning and Midday scenes, as well as have a different color and affect the scene more with indirect lighting than with direct. The settings for the *DominantDirectionalLight* actor that simulated the sun can be seen in Figure 47. Figure 47 shows the lighting settings, e.g., the *Indirect Lighting Scale*, set to be 1.5, the *Indirect Lighting Saturation*, set as 0.3, the *Brightness*, which was set to be 0.5 and the *RGB Color*, initialized as Red=200, Green=180 and Blue=130. The aforementioned settings, as well as the location and direction of the *DominantDirectionalLight* representing the sun, were based on 3ds Max 2011’s representation of a *Daylight System*, set to simulate the afternoon sun, at 19:00 in the afternoon.

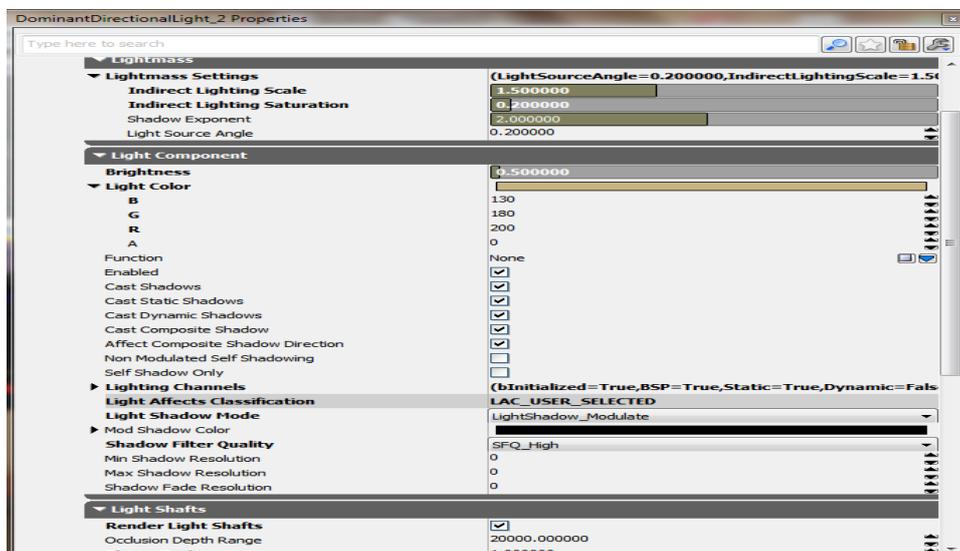


Figure 47: Afternoon sun configuration.

The scenes that were produced after running the Lightmass lighting pre-computation algorithm with the aforementioned settings can be seen in Figure 48 and Figure 49.



Figure 48: Screenshots from the Afternoon Outdoors Scene (left) and the respective wireframe version (right)



Figure 49: Screenshots from the Afternoon Indoors Scene (left) and the respective wireframe version (right)

4.2.4 Evening Lighting Configuration

In relation to the evening scenes, there was no sun simulated. A *PointLightToggleable* was utilized in the center of the indoors room, simulating an artificial light. This light was initiated to simulate a *Standard Fluorescent* light type, however its color and brightness were required to change in real-time. Therefore, Lightmass pre-computed the lighting effects, which would later affect the color or brightness changes.

There were no evening outdoors scenes created, because the absence of a simulated sun would make them completely dark and pointless. The configuration settings for the light actor can be seen in Figure 50. The most important settings include the *Indirect Lighting Scale*, which was set to be 0.5, the *Indirect Lighting Saturation*, which was set to be 1.0 and the *Light Source Radius*, which was set to be 16.0.

Various artificial lighting configurations for indoors were simulated. UDK provides the option to create a Light Actor with a custom set *Light Color*, as an RGB value, as well as the ability to change this value in real-time. In order to make the indoors light actor to simulate the required artificial light types, their RGB value was approximated, according to Hastings 2011 and 3ds Max 2011's representation of artificial light types. Table 2 describes the RGB color value of the light types that were used during the third stage of the experiment. The RGB value associated with the *Standard Fluorescent* light type, which was the initial light value, was used to define the *Light Color* option of the *PointLightToggleable* actor properties, as seen in Figure 50.

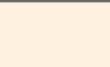
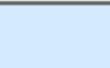
Light Source	R G B Values	Color
40W Tungsten	255, 197, 143	
100W Tungsten	255, 214, 170	
Halogen	255, 241, 224	
Carbon Arc	255, 250, 244	
Standard Fluorescent	244, 255, 250	
Cool White Fluorescent	212, 235, 255	

Table 2: Artificial light types and their respective RGB color value.

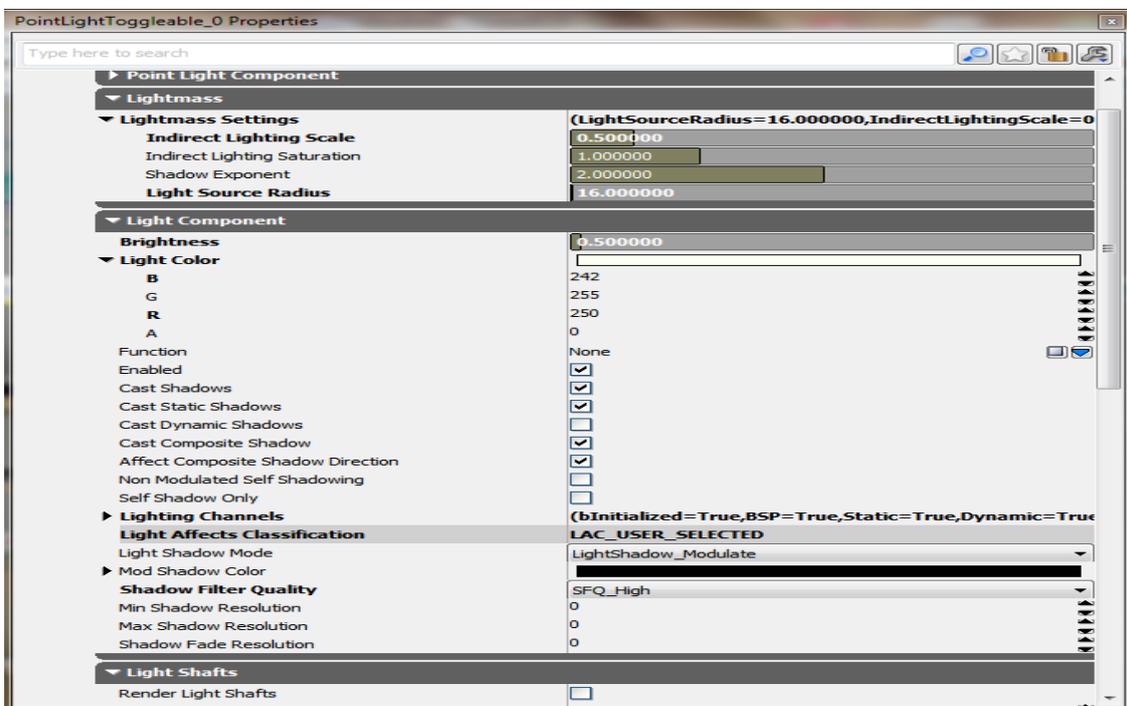


Figure 50: Evening Indoors Artificial light settings. The most important options are the *Indirect Lighting Scale*, *Indirect Lighting Saturation*, *Light Source Radius*, *Brightness* and *Light Color* (in RGB value).

Screenshots from the evening indoors scene and the respective wireframe version are depicted in Figure 51, which shows the synthetic scenes lit by the *Standard Fluorescent* light type applied. Although the lighting effects of the artificial indoors light are still visible in the wireframe scene, the wireframe objects do not offer any impression of realism. The remaining available lighting configurations, which are the *40 Watt Incandescent Tungsten*, the *100 Watt Incandescent Tungsten*, the *Halogen*, the *Carbon Arc* and the *Cool White Fluorescent* light types, are showcased in Figure 52, Figure 53, Figure 54, Figure 55 and Figure 56 respectively.

Chapter 4 – Implementation



Figure 51: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the *Standard Fluorescent* artificial light type applied.



Figure 52: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the *40 Watt Incandescent Tungsten* artificial light type applied.



Figure 53: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the *100 Watt Incandescent Tungsten* artificial light type applied.



Figure 54: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the *Halogen* artificial light type applied.



Figure 55: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the *Carbon Arc* artificial light type applied.



Figure 56: Screenshots from the Evening Indoors scene (left) and the respective wireframe version (right) with the *Cool White Fluorescent* artificial light type applied.

4.3 UnrealScript Classes

The core of the development of the complete application required for the fMRI experiments was implemented in UnrealScript. Several classes were created in UnrealScript which would handle the aspects of the application’s rendering, navigation and interaction with the synthetic scenes. The most important classes created will be presented here.

VRExperimentGame: This was the class of the application that defined the main properties, such as the *Pawn* that would be used in the application and the *Controller* that would handle the *Pawn*. This class extended the *GameInfo* class, as can be seen in its declaration:

```
class VRExperimentGame extends GameInfo;
```

This class was only needed to define the default *Pawn* and the *Controller* for the *Pawn* that would be used in the experiments, when a new 3D scene was loaded. It defines the game being played: the game rules, scoring, which actors are allowed to exist in this game type, and who may enter the game. By ‘game’ here, we mean the experimental scenario implemented. While this class is the public interface, much of its functionality is delegated to several classes to allow easy modification of specific game components. A *VRExperimentGame* actor is instantiated when the level is initialized for gameplay (in C++ *UGameEngine::LoadMap()*). The class of this actor is determined by the *DefaultGame* entry in the game's .ini file (in the *Engine.Engine* section), which was set to be *VRExperimentGame*. The *GameType* used can be

overridden in the class's script event *SetGameType()*, called on the game class picked by the above process.

The experiment flow was controlled from the *Controller* of the *Pawn*, so the only code needed in this class was simply to define these two classes – the default *Pawn* and the default *Controller* of the *Pawn* – in the default properties block of the class, as follows:

```
defaultproperties {  
//The default pawn and the default controller for the VRExperimentGame are defined here  
  DefaultPawnClass=class'VRExperiment.VRExperimentPawn'; // The pawn class is located in the  
  VRExperiment folder.  
  PlayerControllerClass=class'VRExperiment.VRExperimentPlayerController'; //The controller class is  
  located in the VRExperiment folder.  
}
```

VRExperimentPawn: This class defined the *Pawn* that would be used in the application. When a scene was started, a new *VRExperimentPawn* was instantiated, as instructed from the *VRExperimentGame* class. It extended from the *GamePawn* class and defined the main properties of the used *Pawn*, such as the height, the speed and the collision radius of the *Pawn*. The class declaration was the following:

```
class VRExperimentPawn extends GamePawn;
```

The only pawn used in the experiments was that belonging to the participant and it was assumed to be a simple camera navigating inside the synthetic scenes. Also, UDK supports the ability for *Pawns* to jump, swim, fly, climb ladders, etc. Such features were not required, therefore, these abilities were disabled in the *default properties* of the *Pawn* class. One more important aspect that was configured through the *Pawn* class was the *collision cylinder* of the *Pawn*, which was taken into account by UDK to detect collisions between the *Pawn* and the objects in the synthetic scenes.

The settings used for the *Pawn* in its *default properties* block can be seen below:

```
defaultproperties  
{  
  bCanBeDamaged=false //this pawn cannot receive any damage.  
  bCanCrouch=false //this pawn cannot crouch  
  bCanFly=false //this pawn cannot fly  
  bCanJump=false //this pawn cannot jump  
  bJumpCapable=false //this pawn is not capable of jumping  
  bCanSwim=false //this pawn cannot swim  
  bCanTeleport=false //this pawn cannot teleport  
  bCanWalk=true //this pawn can only walk  
  
  AccelRate=+0512.000000 //the acceleration rate of this pawn.  
  
  AirSpeed=+00000.000000 //No speed in the air.  
  GroundSpeed=+0032.000000 //The walking speed the pawn  
  JumpZ=+00000.000000 //The jumping ability of the pawn is set to 0.  
  OutofWaterZ=+000.0 //If the pawn was swimming, this would set how high out of the water it  
  could get.
```

```

LadderSpeed=+000.0 //Speed of climbing ladders.
WaterSpeed=+00000.000000 //Swimming speed

// Sound
bLOSHearing=true
HearingThreshold=+2800.0
SoundDampening=+00001.000000
noise1time=-00010.000000
noise2time=-00010.000000

// FOV / Sight
ViewPitchMin=-16384
ViewPitchMax=16383
RotationRate=(Pitch=20000,Yaw=20000,Roll=20000)
MaxPitchLimit=3072

SightRadius=+05000.000000

// Collision
BaseEyeHeight=+00044.000000
EyeHeight=+00044.000000

CrouchHeight=+34.0
CrouchRadius=+34.0

MaxStepHeight=35.0
MaxJumpHeight=96.0
WalkableFloorZ=0.7 // 0.7 ~= 45 degree angle for floor
LedgeCheckThreshold=4.0f

MaxOutOfWaterStepHeight=40.0
AllowedYawError=2000
Mass=+00100.000000

bCollideActors=true //these options set that this pawn collides with other actors, or the world
bCollideWorld=true
bBlockActors=true

Begin Object Name=CollisionCylinder //the collision cylinder of the pawn
    CollisionRadius=+0020.000000
    CollisionHeight=+0044.000000
    BlockNonZeroExtent=true
    BlockZeroExtent=true
    BlockActors=true
    CollideActors=true
End Object
}

```

VRExperimentPlayerController: This class was used as the *Controller* of the *VRExperimentPawn* and took control of the created *Pawn* when a scene was started, as instructed by *VRExperimentGame*. It was the class that handled all aspects of the application and in which all computations were taking place, such as navigation, interactions, or moving on to the next stage of the experiment. It extended from *GamePlayerController* and it was bound to *VRExperimentConfig* for saving its properties between different scenes and to

VRExperimentDLL, in order to create entries in the log file. The declaration of the class was the following:

```
class VRExperimentPlayerController extends GamePlayerController DLLBind(VRExperimentDLL)
config(VRExperimentConfig);
```

Unlike the previous classes, the *VRExperimentPlayerController* class did not consist only of the *default properties* block, since it was responsible for the flow of the experiments. It contained all the functions and code necessary to control the state of the experiment and the events that should occur at specific time points. All of these will be described in different parts in the following subsections. In the *default properties* block of this class, the default *Player Input* class, which is responsible for the translation between button presses from the user and actions inside the virtual scene, was defined, as follows:

```
defaultproperties
{
    InputClass=class 'VRExperiment.VRExperimentPlayerInput' ;
}
```

4.4 Handling User Input

One of the main requirements of the application was that the synthetic scenes were to be interactively navigated and that the application was required to react to user input, ranging from navigation of the synthetic scenes to interaction with the questionnaires. In order to achieve this, the buttons corresponding to the physical button boxes placed in the fMRI scanner and utilized for interacting with the scene were registered, with their respective commands. These are shown in Figure 72.

So, the following entries were added in the “DefaultInput.ini” configuration file, which is responsible of defining the key bindings for the *Input Manager* to handle:

```
; Right Button Box (the buttons respond like pressing numbers 1,2,3,4 in the keyboard)
.Bindings= (Name="one", Command="GBA_JTurnRight")
.Bindings= (Name="two", Command="GBA_JLookUp")
.Bindings= (Name="three", Command="GBA_JLookDown")
.Bindings= (Name="four", Command="GBA_JFire")
; Left Button Box (the buttons respond like pressing numbers 6,7,8,9 in the keyboard)
.Bindings= (Name="six", Command="GBA_JTurnLeft")
.Bindings= (Name="seven", Command="GBA_JLookUp")
.Bindings= (Name="eight", Command="GBA_JLookDown")
.Bindings= (Name="nine", Command="GBA_JMoveForward")
```

Then, in the same configuration file, the commands bound to the buttons were further assigned to a specific method that would handle them in the *VRExperimentPlayerController* class:

```
.Bindings=(Name="GBA_JTurnLeft",Command="JTurnLeft_P | OnRelease JTurnLeft");
.Bindings=(Name="GBA_JTurnRight",Command="JTurnRight_P | OnRelease JTurnRight");
.Bindings=(Name="GBA_JLookUp",Command="JLookUp_P | OnRelease JLookUp");
.Bindings=(Name="GBA_JLookDown",Command="JLookDown_P | OnRelease JLookDown");
.Bindings=(Name="GBA_JMove", Command="JMove_P | OnRelease JMove");
.Bindings=(Name="GBA_JFire",Command="JProcessItem_P | OnRelease JProcessItem");
```

Each one of the methods *JTurnLeft*, *JTurnRight*, *JLookUp*, *JLookDown*, *JMove* and *JProcessItem* assigned to each button resides in the *VRExperimentPlayerController* class and handles the specific action that it is assigned to.

One major issue arising from the button boxes hardware was the fact that they did not report how long each button was pressed, but only that it was pressed, no matter if it was being held pressed for a specific amount of time. This was due to the fact that the hardware drivers were reporting the “button released” message immediately after the “button pressed” message to the operating system.

The impact of this issue for the application was that the user could not keep a button pressed to, for instance, navigate the scene, e.g. keep the “Move” button pressed to keep moving forward. In order to address this issue, only the “button released” messages were being monitored. The user was instructed to press and release a button to start an action and then press it and release it again to stop it. So, for example, if the user wanted to move forward in the virtual scene, they were supposed to press the “Move” button and the Pawn inside the scene started to move forward until the user repressed the “Move” button.

The navigation in the virtual scenes was not analog to user’s input, but rather it was simulated between the “start” and “end” instructions of the user. For this implementation to go through, there were 5 boolean variables declared in *VRExperimentPlayerController*:

```
//variables used to handle participant's navigation
var bool turningLeft;
var bool turningRight;
var bool lookingUp;
var bool lookingDown;
var bool movingForward;
```

Each method registered in the input configuration file that handled a type of movement was designed to reverse the value of its corresponding boolean variable, while disabling the other possible moves. So, the respective methods corresponding to participant’s button presses, as registered in the configuration file, were the following:

```
exec function JTurnLeft() {
    if(self.bCinematicMode)
        return;

    turningLeft = !turningLeft;
    turningRight = false;
    lookingUp = false;
    lookingDown = false;
```

```
        movingForward = false;
    }

    exec function JTurnRight() {
        if(self.bCinematicMode)
            return;

        turningRight = !turningRight;
        turningLeft = false;
        lookingUp = false;
        lookingDown = false;
        movingForward = false;
    }

    exec function JLookUp() {
        if(self.bCinematicMode)
            return;

        lookingUp = !lookingUp;
        lookingDown = false;
        turningRight = false;
        turningLeft = false;
        movingForward = false;
    }

    exec function JLookDown() {
        if(self.bCinematicMode)
            return;

        lookingDown = !lookingDown;
        lookingUp = false;
        turningRight = false;
        turningLeft = false;
        movingForward = false;
    }

    exec function JMoveForward() {
        if(self.bCinematicMode)
            return;

        movingForward = !movingForward;
        lookingUp = false;
        lookingDown = false;
        turningRight = false;
        turningLeft = false;
    }
}
```

Each one of these methods is responsible to handle a specific type of navigation, either looking around towards a specific direction, or moving the participant's Pawn forwards. In order to do this, each method reverses the boolean value of its respective variable, while setting all other variables to false. The reversal is due to the fact that the same button is used to either start or stop the respective movement.

Every method initially checks whether the controller class is set to be in cinematic mode, as defined in the *self.bCinematicMode* variable and if this is the case, it does not respond to the button press. This prevented the participant from being able to navigate in the 3D scene, when this was not intended, as was the case when the participant was required to answer the questionnaires.

The main method that handled the Pawn's movement inside the synthetic scene was *ProcessMove*, which was called periodically and automatically by UDK. Inside this method, the boolean variables were checked and if they were true, the Pawn would perform the respective movement with its predefined speed. For example, in order to check if the Pawn should move forwards, the *movingForward* variable was checked and then the Pawn's *Acceleration* was adjusted according to its rotation inside the scene and the predefined speed in the Pawn's class, in its *default properties* block. The *ProcessMove* method was as follows:

```
function ProcessMove (float DeltaTime, Vector newAccel, EDoubleClickDir DoubleClickMove, Rotator DeltaRot)
{
    local Vector loc;
    local Rotator rot;
    local Rotator NewRotation;
    local Vector eloc, norm, end;
    local TraceHitInfo hitInfo;
    local Actor traceHit;
    local vector X,Y,Z;
    if( Pawn == None ) // this means that there weren't any pawns handled by this controller?!
    {
        logWrite(ParticipantNumber, CurrStage, "Fatal Error", "Fatal error occured. No Pawn Found.");
        return;
    }

    if(movingForward) //if the movingForward variable is true, the pawn should move forwards
    {
        GetAxes(Pawn.Rotation,X,Y,Z);
        newAccel = MovingSpeed*Normal(X);
        //newAccel.X += 100;
    }
    Pawn.Acceleration = newAccel; //this sets the pawn's acceleration to the new one - 0 if
    movingForward is false, MovingSpeed otherwise

    NewRotation = Rotation;

    if(lookingUp) //if the looking up variable is true, the pawn should look upwards
        NewRotation.Pitch += LookingSpeed;
    else if(lookingDown) //otherwise, if the looking down variable is true, the pawn should look down
        NewRotation.Pitch -= LookingSpeed;

    if(turningLeft) //if turning left is true, the pawn should turn towards its left
        NewRotation.Yaw -= LookingSpeed;
    else if(turningRight) //else if the turning right variable is true, the pawn should turn towards its right
        NewRotation.Yaw += LookingSpeed;
```

```
Self.SetRotation(NewRotation); // the final rotation is applied to the pawn, changed according to the
movement variables
}
```

4.5 Handling the Stages of the Experiment Using States

The experiment was set to be performed in three stages, with different type of available interactions and methodology during each stage. It would not be efficient to create separate copies of the same synthetic scene in order to only change the events that the 3D scene would respond to, so there had to be a way for the controller to be aware of the specific stage of the experiment for each loaded synthetic scene and generate the respective events that were required for each specific experimental stage.

The Controller's experiment flow control was designed to be managed through different states, each one attached to a specific stage of the experiment. The following variables were declared to be saved to the Controller's configuration file and loaded each time the Controller was instantiated:

```
var config array<int> Scenes; // the scenes that will be displayed, in the sequential order they will be
displayed
var config int CurrScene; // the scene that is currently being displayed
var config int CurrStage; // the current stage of the experiment (between 1, 2, 3);
var config array<int> Images; // the emotional images that will be displayed, in the order they will be
displayed.
var config int CurrImage; // the image that is currently being displayed
var config int ParticipantNumber; //the auto-incremental participant number is used to differentiate log
file names.
```

When a new experiment started, the *StartExperiment* method in the Controller class was executed. This method initialized all the experiment variables and then loaded the first 3D scene, as can be seen below.

```
exec function StartExperiment() {
    initializeExperiment();
    CurrScene = 0; //the first map to be loaded will be on index 0 of array Scenes
    CurrStage = STAGE_1; // this indicates that the experiment will start from the first stage.
    CurrImage = 0; //sets the next image to be displayed as the first in the images array
    ParticipantNumber++; //I increase the participant number, so that new logs will be created for this
experiment.
    SaveConfig(); //saves the current configuration in the INI file for next levels to be ready and able to find
it
    ConsoleCommand("open "$levelNames[Scenes[CurrScene]]"); //the open console command finds and
loads the scene given as an argument. The levelNames array contains the string representation name of
each virtual scene. So, this command in fact loads the first virtual scene of the experiment.
}
```

The *initializeExperiment* method initialized the *Scenes* array, which contained the synthetic scenes that would be loaded for the experiment in the order that they needed to be loaded. Also, it initialized the *Images* array, which contained the emotional images that would be displayed, in the sequential order that they would be displayed, during the second stage of the experiment.

During the first stage, the method chose a random virtual scene as the one that the experiment would start with. Then it chose randomly the virtual scenes in the same time of day and then moved on the sequentially next time of day, in the following sequence:

Morning → Midday → Afternoon → Evening → Morning → ...

This sequence of virtual scenes was repeated for the second stage of the experiment, with the exception that no wireframe scenes were displayed during that stage. For the 3rd stage, only two scenes were required to be displayed, the Evening Indoors scene and the respective wireframe scene, so the method simply chose randomly which one would be displayed first.

The second stage of the experiment dictated that the participant should watch nine predefined emotional images in each synthetic scene, three from each category of pleasant, neutral and unpleasant respectively. Although the same emotional images were displayed in each virtual scene across participants, the order that they were displayed was randomized and the resulting sequence of emotional images was saved in the *Images* array.

Whenever a new 3D scene was loaded, a new Controller was instantiated and loaded its variables from the configuration file. Then, it entered the *FindCurrentExperimentStage* state, which checked the variables signifying the experiment's stage and the 3D scene that was currently rendered and set its state accordingly. The following code is the implementation of this state:

```
state FindCurrentState
{
  ignores ShowMenu, ProcessMove;
  event BeginState(Name PreviousStateName)
  {
    if(CurrStage == STAGE_START)
      GotoState('ShowingMenu');//this means that the experiment has not started yet. The participant is watching the start screen and we are waiting to press the button (or training perhaps?).
    else if(CurrScene<14) //we are in the first stage of the experiment, because the first stage of the experiment consists of 14 separate virtual scenes
      GotoState('Stage1');
    else if(CurrScene == 14 && CurrStage == STAGE_1) //The first stage is finished, but the second has not started yet, so it is time for the first break in the experiment
      GotoState('ShowingMenu');
    else if(CurrScene < 21) //we are in the second stage of the experiment, because the 2nd stage of the experiment consists of 7 scenes.
      GotoState('Stage2');
    else if(CurrScene == 21 && CurrStage == STAGE_2)
      GotoState('ShowingMenu');
    else if(CurrScene >= 21) // we are in the third stage of the experiment
      GotoState('Stage3');
  }
}
```

From there on, the Controller could initialize the timers and the events needed for the specific stage of the experiment and plan the flow of events for that stage accordingly. So, each virtual scene could listen to all types of events, referring to all stages of the experiment and react to them, however the Controller would only generate events for the current stage of the experiment.

4.6 Logging of Events and Participants' Actions Mechanism

The application was recording every event that was happening, as well as every action of the participant in a separate log file for each stage, in order to be able to understand and analyze the data gathered from each experiment. Each generated event that changed the state of the experiment, such as the loading of a new virtual scene, or moving on from free navigation in a virtual scene to answering the questionnaires was recorded in the log file. Each log entry reported the specific event that occurred, along with the exact time it happened. The time was measured in milliseconds after the start of the current stage of the experiment, which was assumed to be time point 0.

In order to implement the log file operations, a .dll file was bound to the controller class, providing the necessary methods to record each log entry. This was done because UDK's support for I/O operations is limited, avoiding extreme overhead for the application, since UnrealScript is very slow and inefficient for such operations. So, whenever a new log entry needed to be recorded in the log file, the controller could simply call the C/C++ function residing inside the .dll file and let it perform the operation.

In order to make a function residing in the .dll visible in an UnrealScript class, it had to be declared in that class with a *dllimport* and a *final* modifier. The functions that were in the .dll file and were included in the controller class were the following:

```
dllimport final function initStartTime();
dllimport final function loadStartTime();
dllimport final function logWrite(int pn, int stage, string title, string msg);
dllimport final function logRating(int pn, int stage, string title, string msg, float r1);
dllimport final function logRatings2(int pn, int stage, string title, string msg, float r1, float r2);
dllimport final function logRatings4(int pn, int stage, string title, string msg, float r1, float r2, float r3, float r4);
dllimport final function logSyncPulse(int pn, int stage, string reason, string title, string msg);
dllimport final function logSyncPulse2(int pn, int stage, string reason, string title, string msg, float r1);
dllimport final function logSyncPulse4(int pn, int stage, string reason, string title, string msg, float r1, float r2, float r3, float r4);
```

When a new stage of the experiment started and the controller entered the respective state referring to that stage, it would check whether the scene that was currently being rendered was the first in the current stage of the experiment. If that was the case, it would ask for the execution of the *initStartTime* function, residing in its bound .dll file, otherwise it would ask for the execution of the *loadStartTime* function, in the same .dll file.

The .dll file had a global variable named *startTime*, which was set to be of *struct timeb* type, located in <sys/timeb.h>. So, the functions were designed to be the following:

```
__declspec(dllexport) void initStartTime() //this function initiates the start time and saves it in a file
{
    ftime(&startTime);
    FILE* fp = 0;

    fopen_s(&fp, "startTime.txt", "w");

    fprintf(fp, "%ld\n", startTime.time);
```

```

    fprintf(fp, "%d", startTime.millitm);

    fclose(fp);
}

__declspec(dllexport) void loadStartTime() //this function initializes the startTime from a file.
{
    FILE* fp = 0;
    fopen_s(&fp, "startTime.txt", "a+");

    fscanf_s(fp, "%ld", &startTime.time);
    fscanf_s(fp, "%d", &startTime.millitm);
    fclose(fp);
}

```

Then, whenever an action had to be recorded, the function responsible to record the event could check the current time and subtract the start time, thus finding the time passed since the start of the current stage of the experiment, in milliseconds.

Every action that the participant performed in a virtual scene was also recorded in the log file, as well as the participants' responses in the questionnaires. More specifically, the viewpoint of the participant was recorded, by inserting a log entry every time the object in the center of the screen changed. Also, the virtual scene was divided in nine different equal-sized zones and the participant's movement from one zone to another was recorded.

The tracking of the participant's movement inside a 3D scene could be performed in the *ProcessMove* function mentioned above, a log entry would be created whenever the *movingForward* variable was true and the Pawn should move forwards, polling the new location of the pawn in the 3D scene. However, this proved to be extremely inefficient since the *ProcessMove* function is executed several times per second. This approach would lead to a vast amount of log entries referring to participant movement inside the 3D scene, even for a very small and limited move. Also, it would be difficult to translate later the exact location of the Pawn used from the participant from the recorded XYZ coordinates.

This issue was addressed by dividing the room and the yard into nine equally-sized numbered square zones and assigning a *Trigger* actor in the center of each zone, as can be seen in Figure 57.



Figure 57: The zones into which the yard was divided. The triggers in the center of each zone are highlighted.

Each one of these triggers was invisible to the participant navigating the virtual scene, but they automatically generated a *Touch* event whenever the participant's Pawn entered their respective proximity zone. The *Touch* events were captured in Kismet and an action to record the event in the log file was performed. For example, Figure 58 displays the *Touch* event node of such a trigger and its connection to the *TrackMovement* action.

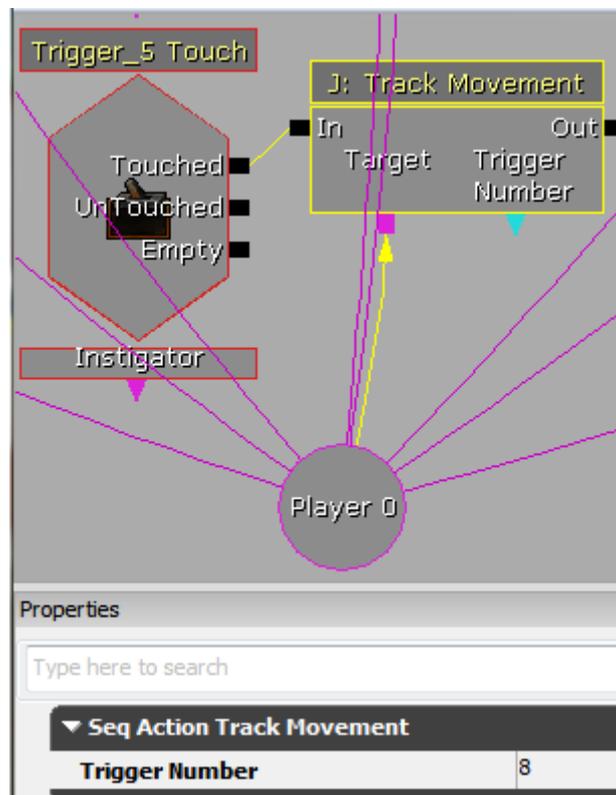


Figure 58: The trigger's "Touch" generated event evokes the *TrackMovement* method in the controller class.

The Controller class's *TrackMovement* method was evoked from the generated event and it requested that the event was recorded in the log file, by calling the *logWrite* function in the .dll file, as can be seen below:

```
exec function TrackMovement(SeqAction_TrackMovement tm)
{
    logWrite(ParticipantNumber, CurrStage, "Participant Moved", "The participant moved to area: " $
tm.TriggerNumber $.");
}
```

The log file keeps records about where the participant was looking at. As with the movement tracking of the participant, this could be tracked in the *ProcessMove* method of the Controller class. However, the same issue still existed, e.g. the fact that for a very simple move or looking around, there would be too many recorded events. Therefore, there was ambiguity in relation to determining where the participant was looking at. Instead, it was preferred to record the object in the center of the participant's viewpoint and add a new log entry each time it changed.

In the *ProcessMove* method, which was discussed above, the following code was inserted pertinent to the tracking of the participant's change of viewpoint:

```
// the following are used to track the object in the center of the screen
GetActorEyesViewPoint(loc, rot);
if(newAccel.X == 0 && newAccel.Y == 0 && DeltaRot.Pitch == 0 && DeltaRot.Yaw == 0)
    return;

if(DeltaRot.Pitch != 0 || DeltaRot.Yaw != 0)
{
    end = loc + normal(vector(rot))*32768; // trace to "infinity"
    traceHit = trace(eloc, norm, end, loc, true, hitInfo);
    if(traceHit != none && traceHit.Tag != viewpointCenterTarget.Tag && traceHit.Tag != 'Trigger')
    {
        viewpointCenterTarget = traceHit;
        logWrite(ParticipantNumber, CurrStage, "Participant Changed Viewpoint", "The participant
changed viewpoint. The new viewpoint is <X:"$rot.Pitch$" | Y:"$rot.Yaw$" > The new target in the
center is: <"$traceHit.Tag$" >");
    }
}
```

In this block of code, we check if the actor in the center of the screen is different from the recorded previous actor, in which case a new log entry is requested.

There were several other events that occurred in the 3D scenes, including the loading of a new 3D scene, the display of questionnaires and the answer to the questionnaires. For each one of these, the Controller class used the imported from the .dll file functions to request the recording of that specific event. Then, the .dll would append to the log file the new entry, after finding out the time passed since the start of the current stage of the experiment. An example function residing in the .dll that was used to record a new log entry is the following:

```
__declspec(dllexport) void logWrite(int pn, int stage, wchar_t* title, wchar_t* msg) {
```

```

FILE* fp = 0;
char filename[100] = "";
sprintf_s(filename, "%s_%d_%d%s", "log", pn, stage, ".csv");
fopen_s(&fp, filename, "a");
if(fp == NULL)
    return;
struct timeb now;
ftime(&now);
double n = now.time;
double s = startTime.time;
double diff = (n - s) * 1000;
diff += now.millitm - startTime.millitm;
fprintf(fp, "%.0lf,", diff);
size_t origsize = wcslen(title) + 1;
const size_t newsize = 500;
size_t convertedChars = 0;
char t[newsize];
wcstombs_s(&convertedChars, t, origsize, title, _TRUNCATE);
fprintf(fp, "%s,", t);
origsize = wcslen(msg) + 1;
convertedChars = 0;
char message[newsize];
wcstombs_s(&convertedChars, message, origsize, msg, _TRUNCATE);
fprintf(fp, "%s\n", message);
fclose(fp);
}

```

4.7 Time Limits Control

The experimental stages were controlled by previously specified time limits and due to the fact that it was conducted inside the fMRI scanner, the application needed to be timed perfectly in order to be synchronized with the brain images acquired by the scanner. The experiment consisted of three different stages, each one requiring specified timings and this had to be reflected in the application. Such time limits were the time available to the participant to navigate a scene before they had to be presented with a questionnaire, or the time the participant had available in order to answer a questionnaire before the next scene had to be loaded.

The Controller class was developed in order to control the time limits and react so as not to exceed them. As already mentioned above, two of its properties that were declared to be saved and loaded from the configuration file were an index to the current stage of the experiment and an index to the scene currently loaded and rendered. Upon initialization, when a new scene was loaded and visible, the controller checked these properties to find out which was the current stage of the experiment and adjust its timers accordingly.

The controller class used *Timers* to control the time limits. It created a new timer to count up to the amount of time before an event had to occur. When a timer expired, it called the specific method that simulated the event required to happen. For example, in order to restrict the participant to navigate a 3D scene for a limited amount of time before they were shown a questionnaire, the Controller class created a timer as shown:

```
SetTimer (FIRST_STAGE_TIME_SECONDS, false, 'showQuestionnaires');
```

The timer in the example was set to execute the method *showQuestionnaires* residing in the Controller class, after a specific amount of time, indicated by the *FIRST_STAGE_TIME_SECONDS* constant contained in the *Controller*. This method was then responsible for activating the event that it was connected to, e.g. the event to display the questionnaires in this example, as shown below. The *checkActivate* method of the *showQEvent* is responsible to activate the indices of the event that are signified from the indices array passed as an argument. Every slot of the indices array contains an index that should be activated in the event node in Kismet.

```
function showQuestionnaires() {
    local array<int> indices;
    indices[0] = 1;
    showQEvent.CheckActivate(Self, Self, false, indices, false);
}
```

According to this example, Figure 59 shows the event that would get activated in Kismet, after this method call. The activated event slot would then activate the action that is connected to it, the *Open Gfx Movie* action in this example. This way, the required event is successfully generated and handled and the action responding to it is performed, after the predefined time has passed.

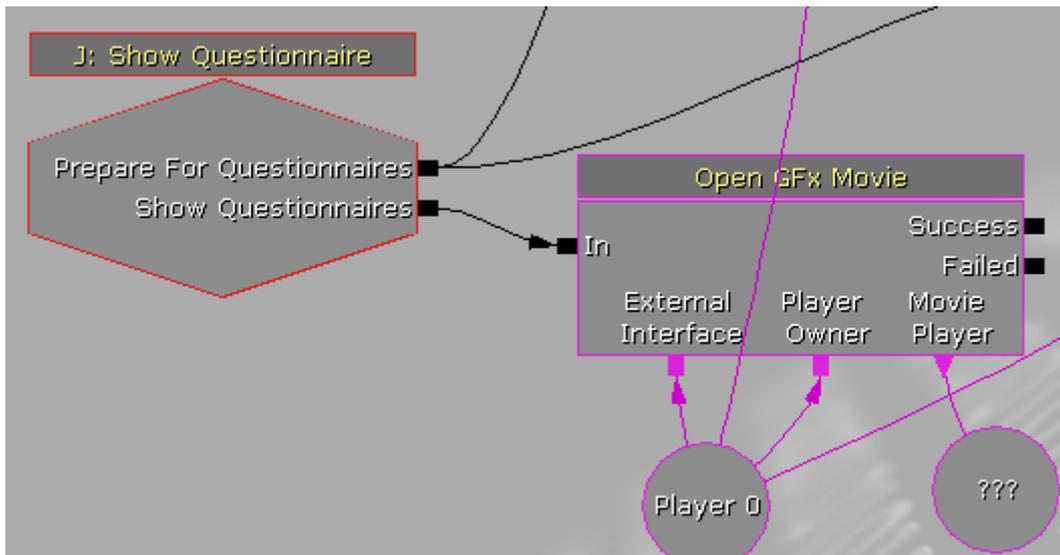


Figure 59: When the index 1 of *ShowQuestionnaires* event is activated, its *Show Questionnaires* slot (the second) would be activated, activating the *Open Gfx Movie* action in turn.

4.8 Emotional images slideshow

The second stage of the experiment was designed to include a slideshow of previously selected emotional images on a TV screen located inside the virtual scenes. These images were imported in the Unreal Editor’s library and a specific material was created for each emotional image, as well as the TV’s default “turned-off” material.

As already explained, the controller class was responsible to find out which was the current stage of the experiment and handle the time limits control. During the second stage,

after a specific amount of time that the participant had available to navigate in the virtual scene, an event was fired that signaled the start of the emotional image slideshow and the specific image that was destined to be projected next. Each 3D scene included in the second stage of the experiment captured these events and responded to them by instructing the change of the material on the TV surface.

When the projection of the emotional images started, a timer was set to start and call the *startTVProjection* method when it finished. This method generated an event that the projection was about to start, activating the index of the first image to be displayed.

```
function startTVProjection() {
    local array<int> indices;
    logWrite(ParticipantNumber, CurrStage, "Emotional Image Projection Start", "The projection of
    emotional images on the television begins now.");
    indices[0] = Images[CurrImage];
    indices[1] = 64; // indicates that an image is going to be projected
    CurrImage++;
    imagesShownInLevel++;
    tvChanger.CheckActivate(WorldInfo, self, false, indices, false);
}
```

Each index of the “Change TV Screen” event was connected to an action that changed the screen of the TV to display the respective emotional image.

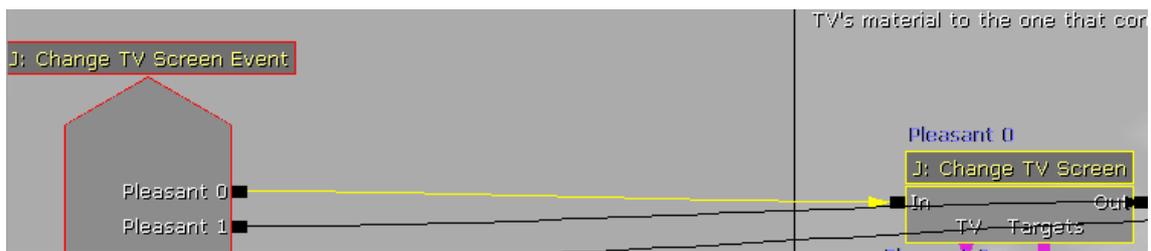


Figure 60: The Pleasant 0 index of the Change TV Screen event was connected to an action that changed the TV screen to the respective emotional image.

After a two second delay, a Flash User Interface was displayed on top of the screen, inquiring the user’s impression of the displayed emotional image. After 10 additional seconds, the Flash interface informed the controller class that it had to be closed and a new event for the next emotional image was fired. A sample part of the kismet sequence is shown in Figure 61.

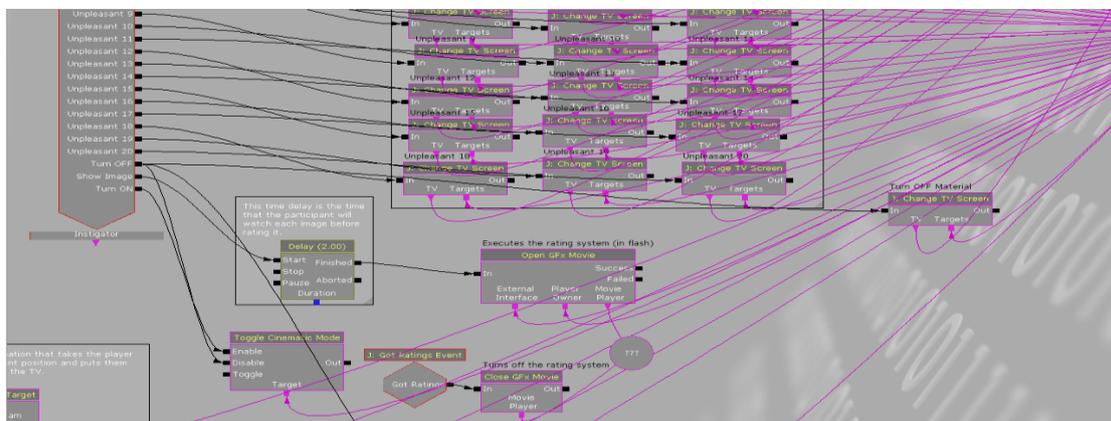


Figure 61: Sample part of the Unreal Kismet sequence for the emotional image slideshow in the 2nd stage of the experiment.

Warm Fluorescent. The *Warm Fluorescent* event is activated, triggering in turn the Matinee responsible of changing the light colour to simulate the *Warm Fluorescent* light type.

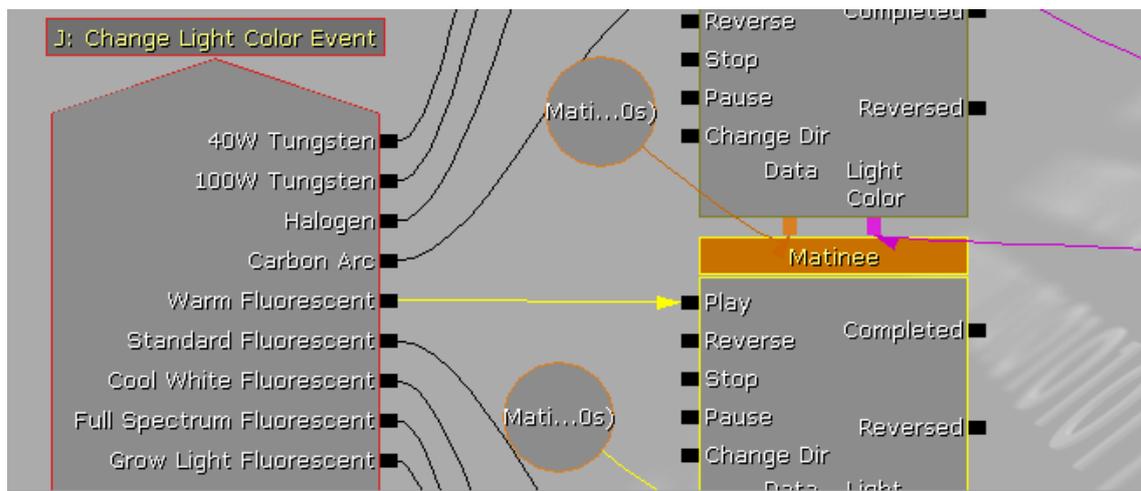


Figure 63: The Kismet sequence responsible of changing the artificial indoors light type, during the third experimental stage. When the "Warm Fluorescent" index of the "Change Light Color" event was activated, the Matinee that was responsible to change the artificial indoors light color to "Warm Fluorescent" was activated in turn.

During the third experimental stage, the participant was requested to alter the displayed light's brightness to the most comfortable value before being presented with the questionnaire inquiring the impressions of lighting. The Kismet sequence that handled the change of the displayed light's brightness in real-time, according to user's input, is shown in Figure 64. The *Lever Moved* event was activated when the participant moved the cursor displayed on the screen to signify that a new brightness value should be applied. This event triggered a sound to be played, to notify the participant that the new brightness would shortly be applied. At the same time, the *Change Brightness* action was also activated, with the new *brightness* value passed as an argument; it is the blue circle depicted on Figure 64, connected to both the event and the action. This action was responsible to change the artificial light's brightness to the new value.

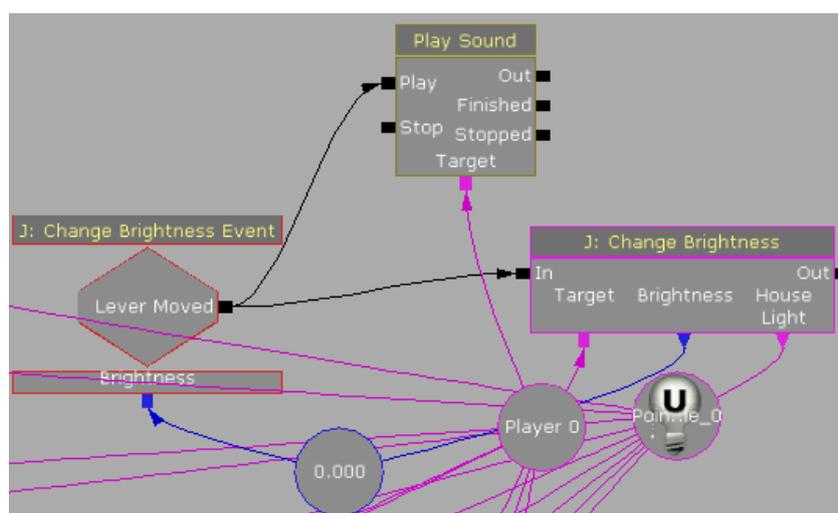


Figure 64: The Kismet sequence responsible to handle participant's input to change the displayed light's brightness. When the participant moved the cursor to alter the brightness, the *Lever Moved* event was activated, triggering the *Change Brightness* action, with the new brightness value passed as a parameter (the blue circle connected to both the event and the action).

4.10 Synchronization with the fMRI Scanner

The experiments were conducted inside an fMRI scanner and the application was required to be perfectly synchronized with the scanner, in order to be able at a later stage to find the exact state of the application associated with each brain image acquired by the scanner. The approach that was used to address this issue was to instruct the application to send a specific sound sync pulse to be recorded from the PC that was dedicated to recording the spike signals sent from the synchronization box of the scanner, as well as the heart pulses and the heart pulse oximetry of the participant. The sound sync pulses were directed through the left audio channel of the application, leaving only the right audio channel to be heard by the participant.

The required synchronization between the fMRI scanner and the application was achieved by sending such sound sync pulses to be recorded as an analog spike signal, whenever an important event occurred in the application while at the same time recording that event and the exact time it happened in the log file. A sample screenshot of the spike application recording the signals can be seen in Figure 65.

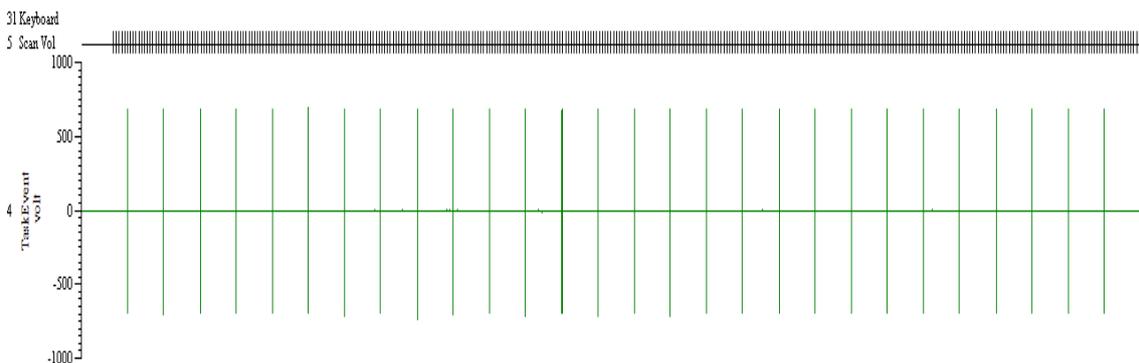


Figure 65: A sample screenshot of the spike recording application. In the first line, each black bar signifies a new brain image acquired from the scanner. In the second line, each green bar is a sound sync pulse sent from the application. The heart pulse and the pulse oximetry signals were omitted.

In the image shown above, the spikes sent by the fMRI scanner's synchronization box can be seen in the first line, with each black bar being a new brain image gathered by the scanner. The second line displays the recording of the sound sync pulses, which were sent each time a new event occurred in the application. The log file could describe the state of the application for a specific brain image since the only information needed was the sound sync pulse that was sent last before the acquirement of that image.

An example of the synchronization mechanism through sound sync pulses was when a new 3D scene was loaded and rendered. The *Controller* class, after entering the state that reflected the experiment's current stage, would send a sync pulse through the left audio channel in order to be recorded by the spike application and would then record the sync pulse in the log file, as described below:

```
ClientPlaySound(SyncPulse); //plays the specified sound
logSyncPulse(ParticipantNumber, CurrStage, "Sync Pulse", "Map Start", "The current map
|$levelNames[Scenes[CurrScene]]$" | has now started."); //records a new log entry about the last sync
pulse, registering that it was played because a new virtual scene started.
```

5 Chapter 5 – UI Implementation

Although UDK includes preliminary support for User Interfaces (UI), the ability to embed Flash User Interfaces was appropriate in relation to the requirements of the experiments, since the UIs were required to be interactive and to be displayed transparently on top of the rendered scene. These requirements could only be fulfilled with the use of embedded Flash UI applications. The UIs required besides navigation of the 3D scenes included initial menus displaying instructions before the start of a new stage of the experiments as well as embedded questionnaires inquiring the participants' impressions of the virtual scene, the emotional images, as well as the user set lighting condition.

Each Flash UI was designed so as to be displayed on top of the currently rendered virtual scene. The participant's ability to navigate and look around the virtual scene was disabled every time a Flash UI was displayed, so that the input could be captured by the Flash UI itself. Figure 66 depicts a Flash questionnaire being displayed on top of the virtual scene.

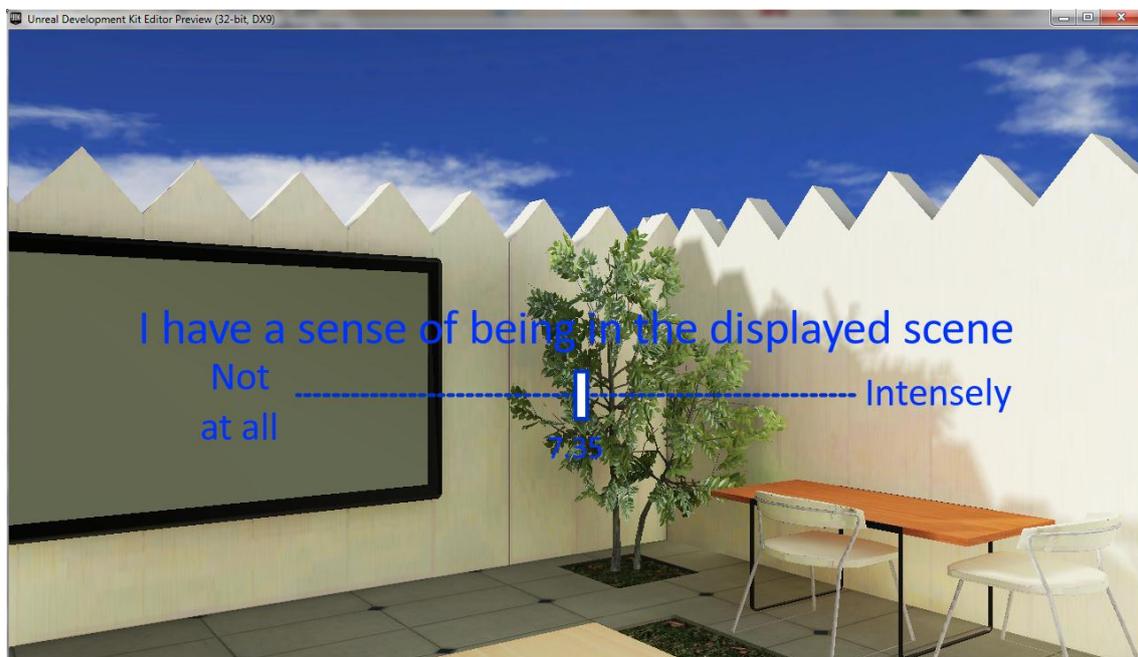


Figure 66: A Flash UI questionnaire being displayed on top of the currently rendered virtual scene.

Further below in this chapter, the Flash UIs that were used in the application will be described. They are divided in two categories, the Flash menus and the Flash questionnaires and they will be presented in their respective subsections.

5.1 Application menus as Flash UIs

The application used in the experiments included several menus, such as the start and end menus of the application, as well as the menus shown during the breaks of the experimental process, between the different stages. Each menu displayed useful information concerning the next part of the experiment, reminding the participant what he was requested to do next. Also, the menus contained graphical representations of the button boxes,

explaining the function of each button and allowing the participant to test it and see it activated after each press. The menu screen that appeared when the experiment started is depicted on Figure 67. On the lower left and on the lower right of the menu, as seen in Figure 67, is a rough graphical approximation of the button boxes that the participants used as a response pad (Figure 72). The color of the buttons in the graphical representation is the same as in the original button boxes. Next to each of the buttons of the button boxes is a short explanation of its function in the next experimental stage. The buttons on the right button box were assigned to turn right, look up, look down and start an interaction with a 3D object, while the buttons on the left button box were assigned to turn left, look up, look down and start / stop moving forward.

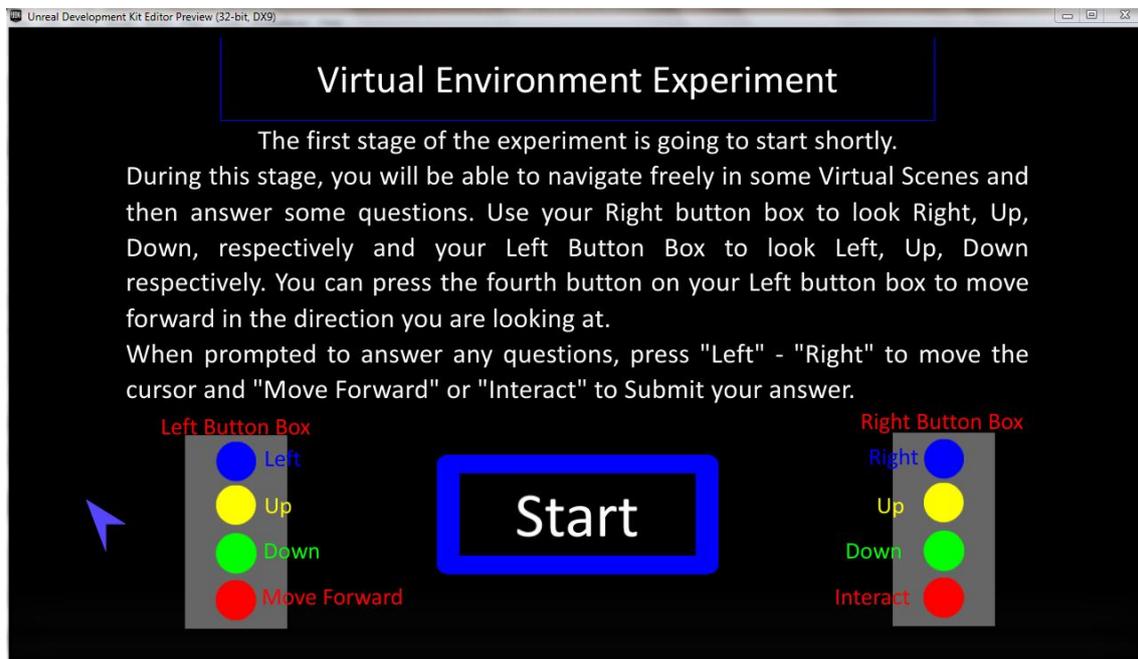


Figure 67: The start Flash menu that was displayed when the experiment application was started.

In order for UDK to embed a Flash application as a UI and display it, it is required to import the .SWF file of the Flash UI. Then, we can load it and display it in an empty scene by connecting the *Open Gfx Movie* action to the *Level Loaded and Visible* event, which is automatically generated and activated by UDK, when the virtual scene becomes visible. The *Open Gfx Movie* action is provided by UDK and it accepts an imported Flash UI as an argument, which it loads and displays on the screen or on the specified surface, if one has been specified. For the specific needs of the menu screens in the experiments, the Flash menus should be displayed at the center of the screen.

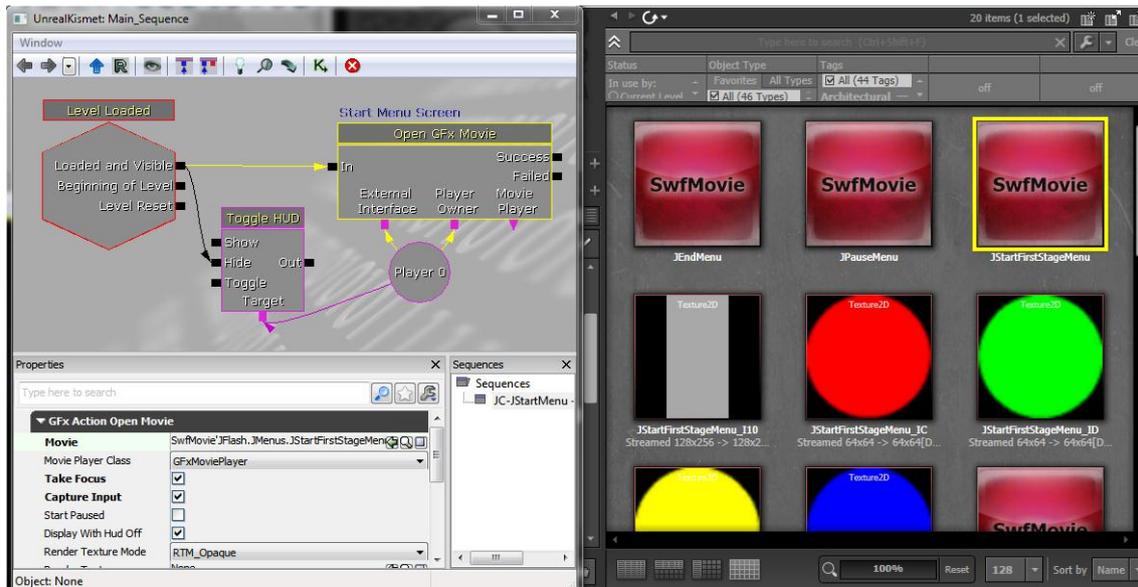


Figure 68: The start menu Flash UI is loaded and displayed, immediately after the virtual scene becomes visible.

The participant’s ability to navigate is disabled by selecting the “Take Focus” and “Capture Input” options in the Open Gfx Movie action. Also, it can be seen that the Loaded and Visible event is also connected to the Toggle Hud action, which hides the HUD.

When the experiment started, or restarted, as was the case for the menus displayed during the breaks between different experimental stages, the *Start* button was pressed from the researchers – not the participants. The press of the *Start* button in the Flash UI resulted in the execution of the call to the responsible method in the Controller class to start the next stage of the experiment. In the previous example, the press of the *Start* button resulted in the following code to be executed:

```
ExternalInterface.call("StartStage1");
```

UDK’s Flash player would then search for the *StartStage1* method in the Controller class and execute it, which would lead to the start of the first stage of the experiment. Then, the controller would call the *StartExperiment* method, which would initialize and initiate the experiment starting from the first stage.

The reason that each new stage had to be initiated by the researchers was due to the fact that the experiments were conducted inside an fMRI scanner. Before each subsequent stage of the experiment could be started, the scanner should have already started acquiring brain images. So, there was need to wait for the fMRI scanner to begin and then the researchers pressed the *Start* button in the Flash UI menu.

5.2 Questionnaires designed as Flash UIs

Two questionnaires were administered to the participants at specific instants during the experiments. The questionnaires were displayed on the screen, allowing the participant to still see the virtual scene while answering the questionnaires (Figure 62). The participant was not allowed to navigate the scene while answering the questionnaires, but rather stay still at a predefined spot. The answers each participant gave were recorded in the log files.

The first of the two questionnaires was displayed during the first and the third experimental stages and it involved four questions, which were:

1. *“Do the things around you in the displayed environment feel unreal?”*
The participant was required to indicate his response by moving a cursor on a scale from 0 to 100, where 0 was marked as “Not at all”, while 100 was marked as “Intensely unreal”.
2. *“Do you yourself feel unreal?”*
The participant was required to indicate his response by moving a cursor on a scale from 0 to 100, where 0 was marked as “Not at all”, while 100 was marked as “Intensely unreal”.
3. *“The displayed environment feels comfortable.”*
The participant was required to indicate his response by moving a cursor on a scale from 0 to 100, where 0 was marked as “Not at all”, while 100 was marked as “Intensely”.
4. *“I have a sense of being in the displayed scene.”*
The participant was required to indicate his response by moving a cursor on a scale from 0 to 100, where 0 was marked as “Not at all”, while 100 was marked as “Intensely”.

The second questionnaire was displayed during the second experimental stage and it contained a single question, in order for the participant to rate each displayed emotional image. The question that was displayed was:

- *“How pleasant / unpleasant was the image?”*
The participant was required to indicate his response by moving a cursor on a scale from 0 to 100, where 0 was marked as “Intensely Unpleasant”, while 100 was marked as “Intensely Pleasant”.

When answering the above questions, the participant moved a cursor on a scale in order to indicate their response to the respective question. The Flash UIs allowed the participant to move a cursor upon a scale. When the participant pressed the button to submit the response, the Flash UI reported the response to the Controller class for it to be recorded in the log file. An example of a Flash UI questionnaire can be seen in Figure 69.

Questions 1 and 2 related to the depersonalization syndrome. Question 3 related to ratings of comfort according to the specific lighting visualized. Question 4 related to assessing the ‘sense of presence’ while being exposed to a photorealistic or wireframe scene of the space.

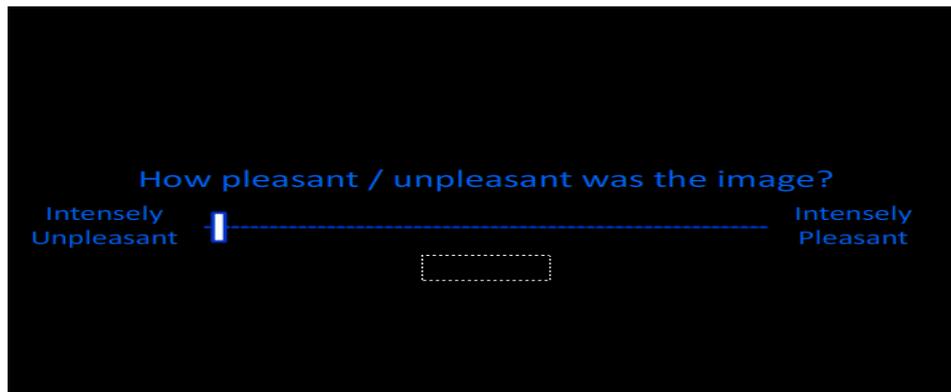


Figure 69: The Flash UI questionnaire that was displayed during the second experimental stage.

When the participant pressed the button to submit an answer, the Flash UI questionnaire reported the answer to the Controller class by calling the respective method of the Controller class. Following the example shown in Figure 69, the Flash UI would execute the following code to call the method of the controller class responsible of recording the rating of the emotional image by the participant, as well as playing a sound sync pulse:

```
ExternalInterface.call("RatingButtonPressed", Scale.value);
```

As explained previously, this would result to the execution of the “RatingButtonPressed” in the controller class and a sound sync pulse would be played and recorded, along with the reported answer to the questionnaires.

Fair comparison of replies across participants required that the experimental process was timed accurately. This was essential because the analysis of the brain image data across participants would be extremely difficult if each brain image of each participant did not correspond to the same brain image for the other participants. The participants should always be in the same state of the experiment after a specific amount of time has elapsed since the start of the current stage of the experiment. For example, 10” after the start of the current stage of the experiment, every participant should be still navigating the first virtual scene.

The fact that each participant answers each question at a different pace had to be addressed. If the Flash application allowed each participant to move on to the next question when the answer was submitted, it would soon lead to spiraling out of control in terms of time synchronization. The approach used was that the participants had a predefined time period available to respond to each question. If they submitted their answer before the time available expired, they would see a “Please wait” message, until it did expire. If the available time passed without the participant pressing the submit button, then the participant was assumed to have failed to submit an answer to that question. The Flash UI would still report the latest value that the cursor was placed upon and then moving on to the next question. However, it would give the reported answer a negative sign to denote that the participant did not press the submit button.

When the questionnaire Flash UI was fully answered and after it had reported the participant’s answers to the *Controller* class, the controller could stop the Flash UI from being displayed by activating an event connected to the *Close Gfx Movie* action, as described in

Figure 70. More specifically, as shown in Figure 72, the *Got Ratings* event is activated by the Controller class as soon as the responses to the questionnaire are received. This event is captured in the Kismet sequence and it triggers the *Close Gfx Movie* action, which is responsible to stop the Flash player associated with the Flash UI questionnaire.

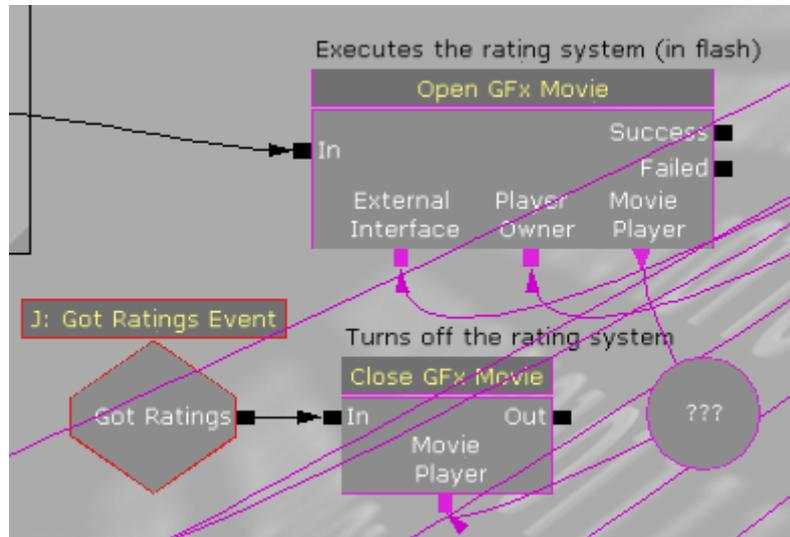


Figure 70: When the “Got Ratings” event is activated, it activates the Close Gfx Movie action, which stops UDK’s Flash player from displaying the Flash UI.

5.3 Light Control Panel as Flash UI

During the third stage of the experiments, the participants were allowed to manipulate the indoors artificial light, in terms of its brightness. This was supposed to become possible by displaying a scale on top of the screen with a range between 0.1 and 0.9, where 0 was marked as “Low Brightness” and 0.9 was marked as “High Brightness”. An example image can be seen in Figure 71.



Figure 71: The light control panel Flash UI

Whenever a participant tried to adjust the light brightness by moving the cursor on the scale, the Flash UI reported the change to the controller class, which in turn generated an event that the brightness changed. This event was captured in Unreal Kismet and an action was connected to it, which handled the change of the light’s brightness.

6 Chapter 6 – Experiments

6.1 Materials

This experiment was designed to explore the effect of modification of light and emotional stimuli on the feeling of ‘reality’ in relation to the depersonalization syndrome, the feeling of ‘presence’ in relation to being exposed to a simulation as well as on the well-being of the normal and ultimately patient populations when being exposed to varied quality conditions of the Virtual Environment (VE), ranging from photorealistic and wireframe. We aim to explore the biological correlates of variations in presence and subjective feelings of depersonalization and comfort by creating a VE which can be projected into an fMRI scanner, enabling the monitoring of neural activation patterns in response to a range of manipulations of the VE. The brain activation data will be correlated with subjective reports of presence as well as subjective impressions of lighting and self-realization and also with other physiological variables reflecting autonomic activity, e.g. heart rate variability, which is measured during scanning by the use of MR-compatible pulse oximetry. This will enable us to record a physiological ‘signature’ of different states of presence and perceived reality experienced by healthy volunteers in the scanning environment. Information on the neurobiological correlates of variations will then inform future work on analogous disturbances of experience seen in psychiatric patient groups. From a computer graphics point of view, such neurocorrelates will serve as automatic, non-conscious, self-reported metrics of fidelity of a simulation.

6.1.1 Participants

12 male participants from the University of Sussex postgraduate and other associated research staff populations were recruited. The participants were naive as to the purpose of the experiment. All participants had normal or corrected to normal vision and no reported neuromotor impairment.

6.1.2 Apparatus

The VEs were presented at VGA resolution on the screen of an fMRI, with a Field-of-View comprising 50 degrees diagonal. A Current Designs HHSC-2x4-C response pad was utilized for rotation, forward movement and interaction with the displayed environment (ideally based on hardware), such as manipulating lighting. The viewpoint was set in the middle of the synthetic scene. Rotation was restricted to 180 degrees vertically (pitch). Navigation around the scene was restricted so that participants would not intersect with the virtual objects. The application ran on a standard PC connected to the screen of the fMRI as Clone-Mode. The experiment was conducted inside an fMRI, which recorded the brain images data. Another standard PC was used to record the spikes sent from the fMRI’s synchronization box, as well as pulse oximetry and sound sync pulses from the application.



Figure 72: Photo of the Current Designs HHSC-2x4-C response pad used in the experiments.

6.1.3 Visual Content

The synthetic scenes presented to participants were of either high-quality or including wireframe objects. The high-quality synthetic scenes were rendered with full textures and materials over the 3D objects. Wireframe can be defined as a display type that shows the geometric object made up of its edges and drawn as lines resembling a model made of wire, with the average colour of the texture in the high-quality scene.

The high-quality synthetic scene viewed consisted of a five by five meters room connected through a door to a yard of equal size. The room contained a sofa, a coffee table, a dinner table, four chairs, a plant, a bookcase, a TV and an entry phone. The yard contained matching furniture as indoors and a fence allowing a direct view to the sky and the sun (Figure 42 and Figure 43).

The low-quality synthetic scene was the wireframe version of the high-quality one. Every 3D object in the scene was displayed as wireframe. The walls, ceiling, floor, fence and sky were kept as surfaces, in order for the lighting effects to still be visible in the wireframe version. The wireframe mesh of each object was coloured in utilizing the average colour of its texture.

The scenes were lit by a dominant directional light representing the sun. The sunlight varied according to the time of day in terms of brightness and colour. The evening indoors scene was lit by a single ceiling mounted artificial light which remained constant representing a standard fluorescent light. During = the third stage of the experiment, = the light configuration changed and the participants manipulated the lighting brightness themselves.

In Sections 6.2.4, 6.2.5 and 6.2.6, we will offer more details in relation to the visual content utilized during each of the three experimental stages.

6.2 Methods

6.2.1 Experimental Procedures

Before entering the fMRI scanner, a preliminary training phase dedicated to each participant took place. When the initial training was completed, the actual experiment was

conducted inside the fMRI scanner. The experiment was set to be completed in three stages by each participant. During the first stage, participants were instructed to freely navigate and look around each synthetic scene presented to them, either indoors or outdoors according to the visual conditions listed in Section 6.2.4. During the second stage, participants interacted with the television placed in the synthetic scene and rated the displayed emotional images in relation to a pleasantness rating. Lastly, during the third stage, participants manipulated the indoor lighting brightness while the artificial lighting configuration changed. There was a brief break after each stage of the experiment. The experimental stages are described in detail below. The fMRI scanner is shown in Figure 74. The participants were placed lying inside the fMRI scanner, with their heads placed in the coil. Through a mirror on top of the coil, the participants were able to view the VEs displayed behind them on the projector screen.

6.2.2 Experimental Setup

The experiment was conducted inside the fMRI scanner (Figure 74). In order to synchronize the interactive 3D application with the fMRI scanner, several hardware parts had to be set up including the fMRI scanner, the synchronization box, the analog signal box, the button boxes, the sound mixer, the visual stimuli PC, on which the application was executed and the spike PC, which recorded the spike signals sent from the synchronization box. Figure 73, which describes the experimental setup, shows that the *Visual Stimuli PC*, which executed the interactive application, was connected to the fMRI projector and displayed the VEs on the projector screen. The button boxes were connected to that PC. The PC's sound card was connected to a sound mixer, which separated the two audio channels and directed the left towards the analog signal box and the right towards the participants' headphones. The analog signal box also received signals from the synchronization box which was connected to the fMRI scanner, as well as from the heart rate recorder attached to the participant's toe. The spike PC was dedicated in recording the inputs received from the analog signal box.

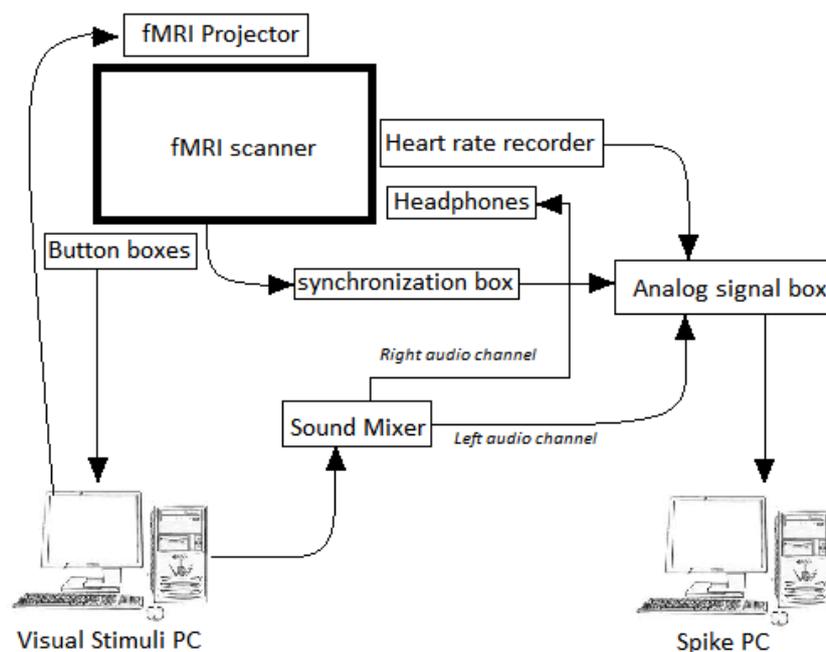


Figure 73: The experimental setup.



Figure 74: The fMRI scanner inside which the experiments were conducted. The participants' heads were placed inside the coil that can be seen in the fMRI scanner. Through a mirror on top of the coil, the participants could see the VEs displayed on the projector screen behind them.

6.2.3 Training

Before the actual experiment inside the fMRI scanner, participants were trained on a standard PC in order to learn how to move and interact with the VE. For the purposes of the training, the participants were asked to use the same button boxes (Figure 72) as in the actual

experiment. The participants viewed a virtual room comprising of white walls, floor without a ceiling so the sky was visible. Primitive objects were placed in the room, both regular 3D objects and wireframe. Training also took place in relation to the second stage of the experiment. Participants viewed a TV mounted on a wall, which was used to display sample emotional images different from the ones actually displayed during the experiment. Participants were sequentially trained for all three stages of the experiment, learning how to move, interact with the VE and how to respond to the questions asked in each stage. The synthetic scenes used for the training in relation to the three experimental stages are shown in Figure 75, Figure 76 and Figure 77 for each stage respectively.

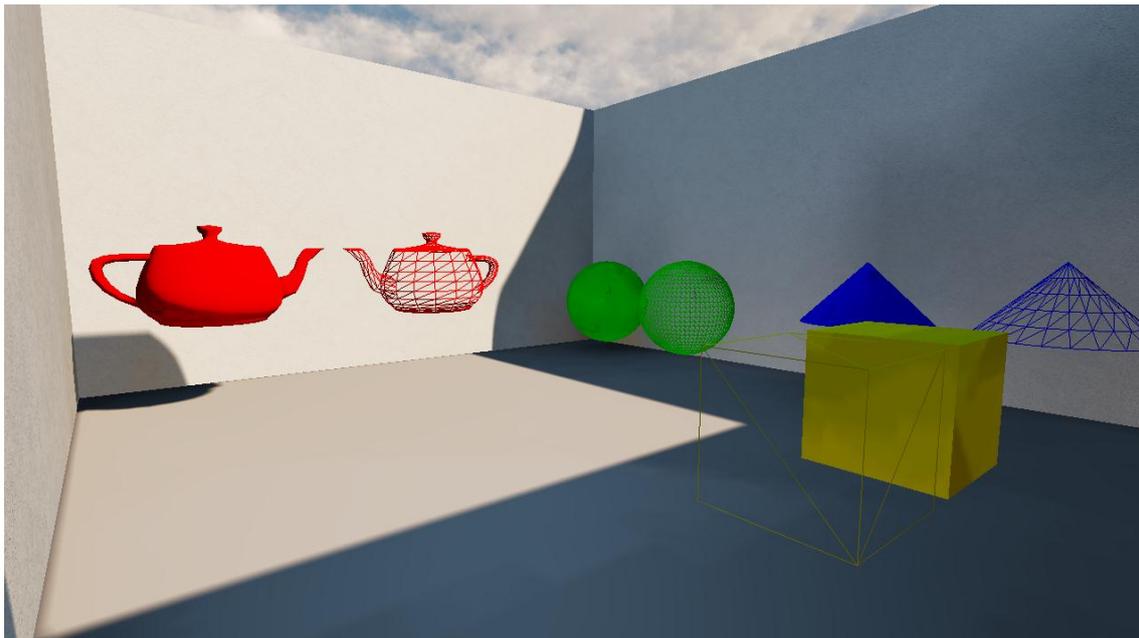


Figure 75: Screenshot of the training synthetic scene in relation to the first stage of the experiment.

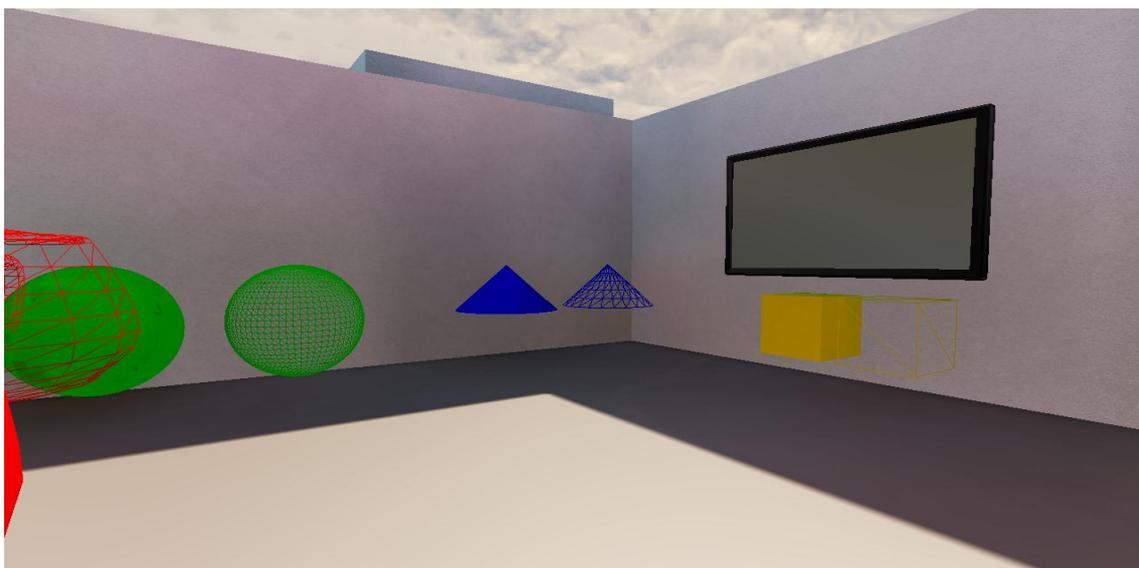


Figure 76: Screenshot of the training synthetic scene in relation to the second stage of the experiment.

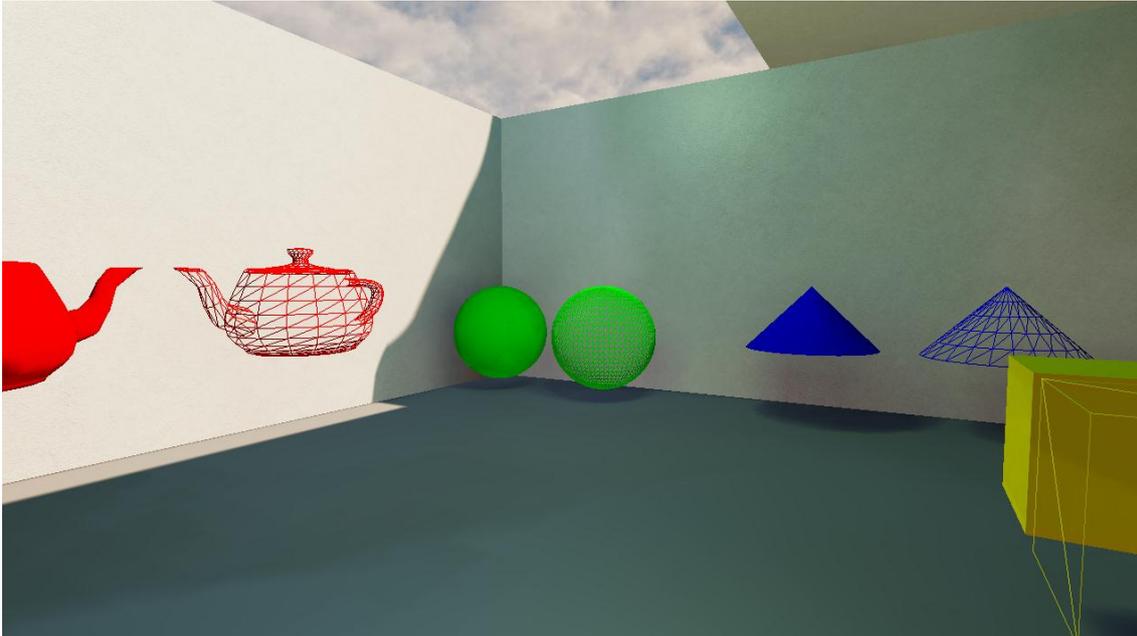


Figure 77: Screenshot of the training synthetic scene in relation to the third stage of the experiment.

6.2.4 First stage

During the first stage of the experiment, participants were exposed to an interactive simulation of the synthetic scene in one of the conditions below. Each scene varied considerably with regard to the indoor and outdoor lighting based on time of day and artificial lighting configurations. Participants could navigate the space inside the room or the yard, however, in each case, they could not move from indoors to outdoors and vice-versa. When in both locations, the participant could view the adjacent space through the glass. Each participant viewed both the high-quality and the low-quality version of each scene.

- Morning indoors: The dominant directional light is set to simulate the morning sun and the participant navigates the space indoors.
- Morning outdoors: The dominant directional light is set to simulate the morning sun and the participant can navigate the yard.
- Midday indoors: The dominant directional light is set to simulate the mid-day sun and participant navigates the space indoors
- Midday outdoors: The dominant directional light is set to simulate the mid-day sun and the participant can navigate the yard.
- Afternoon indoors: The dominant directional light is set to simulate the afternoon sun and the participant navigates the space indoors
- Afternoon outdoors: The dominant directional light is set to simulate the afternoon sun and the participant can navigate the yard.
- Evening indoors: The room is artificially lit (evening) and participant navigates the space indoors.

Lightmass was employed in order to produce a high quality VE across conditions, taking into account the position and the properties of the directional light simulating sunlight.

During the first stage of the experiment, participants were required to sequentially navigate each virtual scene for 42". The participants began the experiment by navigating around the scene as rendered at a random time of day, either morning, midday, afternoon or evening, at a randomly selected location either indoors or outdoors and at a random version of each scene (wireframe or high quality). Participants moved on randomly to the next unvisited version of the scenes of the same time of day, in a different location and/or different quality. Then they moved on to the next scene viewed following a time-relevant sequence, e.g. to the following time of the day in the sequence morning → midday → afternoon → evening → morning.

After the time available to freely navigate each scene elapsed, the participants were automatically moved to a predefined location, in order to be placed to a viewpoint that allowed the participant to have a full view of each virtual scene. At the same time, a short sound was initiated in order to remind the participants that the questionnaire is going to show up. Then, they were presented with a questionnaire, comprising of four questions, as detailed in section 5.2, allowing for ten seconds to answer each question. If the participants did not manage to answer the question, the application automatically moved on to the next question and reported the last value the participant had selected as a possible missed rating. After the last question was either answered or skipped, the next virtual scene was loaded and rendered.

6.2.5 Second stage

When each participant had navigated the 3D scenes and completed answering the questions administered to them after each scene was viewed, there was a short break of about thirty seconds. Subsequently, instructions related to the second stage of the experiment were displayed on the screen. During the break, participants were asked whether they were comfortable and wanted to move on to the next stage of the experiment. Moreover, a short reminder of the requirements of the second stage was administered.

Moving on to the next stage of the experiment, participants were exposed to the scenes in the order they were viewed during the first stage of the experiment. However, now they only navigated the high-quality synthetic scenes and they were instructed to interact with the television placed in the room. Figure 78 and Figure 79 show a sample screenshot of the emotional image slideshow, in the midday indoors synthetic scene, with a pleasant and an unpleasant emotional image displayed on the television respectively.



Figure 78: Sample screenshot of the Midday Indoors scene with a pleasant emotional image displayed.



Figure 79: Sample screenshot of the Midday Indoors scene with an unpleasant emotional image displayed.

Each participant was allowed ten seconds to initiate this interaction in each scene, otherwise it was performed automatically. This interaction initiated the projection of a slideshow on the TV screen. The slideshow contained a range of emotional images each one displayed for two seconds. There were three randomly displayed blocks of images which were previously ranked as unpleasant, neutral or pleasant respectively. Subsequently, the participant rated each image viewed on the TV by adjusting a bar representing an 'intensely unpleasant to intensely pleasant' scale. The participant was allowed ten seconds to rate each image, during which time the image was still projected on the TV. The rating system closed if the participant didn't rate each image before the end of time and the last selected rating was kept as a possible missed rating. The images displayed in each virtual scene were the same for all participants, with the exception that they were presented in a randomized sequence of blocks of three images, one block belonging to each category of pleasant, neutral or unpleasant.

The emotional images used in the experiment were selected from the International Affective Picture System (IAPS) (Lang, Bradley & Cuthbert 1995). The IAPS is being developed to provide a set of normative emotional stimuli for experimental investigations of emotion and attention. The goal is to develop a large set of standardized, emotionally-evocative, internationally-accessible, color photographs that includes contents across a wide range of semantic categories. The IAPS (pronounced eye-aps) is being developed and distributed by the NIMH Center for Emotion and Attention (CSEA) at the University of Florida.

A set of these images are employed in the experimental study presented in order to assess the effect of lighting variations (day light vs artificial light) on the perceived emotional intensity of such images lit differently because of the time of day or the configuration of the artificial lighting utilized. There were 63 pictures selected, 21 representing each category of pleasant, neutral and unpleasant scores respectively.

6.2.6 Third stage

When the second stage was over, the second break in the experiment occurred and there were instructions for the third stage displayed, as previously. After a short chat with the participant, the next stage of the experiment began.

During the third and final stage of the experiment, the participant was exposed randomly to either the wireframe, or the high-quality version of the evening indoors scene. The lighting configuration switched randomly to one of the available light types, i.e. 40 watt tungsten, 100 watt tungsten, halogen, carbon arc, standard fluorescent and cool white fluorescent light types. While an artificial lighting configuration was presented, the participant was asked to change the brightness to their most comfortable value by adjusting a scale from low to high brightness, or vice versa.

During exposure to each lighting configuration, a pre-rendered walk of the room was displayed lasting for fifteen seconds. Subsequently, each participant was asked to answer a questionnaire including the same four questions posed during the first stage of the experiments as detailed in section 5.2 allowing for ten seconds for each to be answered. If a question was not answered, the application automatically moved on to the next question and reported the last value the participant had selected as a possible missed rating. Then, the next lighting configuration was displayed. The order of presentation of either the high quality or the low quality scenes was counter balanced.

6.3 Statistical Analysis

In this section, the basic statistical principles used to analyze the data acquired through the participants' responses to the questions during each stage of the experiment will be presented.

6.3.1 Repeated-Measures Generalized Linear Model (GLM)

The Generalized Linear Model (GLM) is a flexible generalization of ordinary linear regression. The GLM generalizes linear regression by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its predicted value. The tests performed were based on the repeated-measures GLM, for the responses to each question.

Each GLM test is based on two factors, in order to create the axes, on which the data will be plotted. These factors were different for each stage of the experiment. For instance, relevant to the first stage of the experiment, there were two factors, the *reality* involving two levels, either wireframe (WF) or photorealistic and the *day-time-place* involving seven levels, i.e. the morning, midday, afternoon and evening and indoors and outdoors locations. The GLM test relevant to the second stage of the experiment involved two factors, *day time* with four levels, i.e. morning, midday, afternoon and evening and *valence*, involving three levels, pleasant, neutral or unpleasant. For the third stage of the experiments, the two factors of the GLM test were *reality*, which included two levels, WF or photorealistic and *luminescence*, involving six levels, one for each of the available light types.

The purpose of the GLM tests was simply to obtain the plots of the mean values across groups for each question it was performed on and visualize the data. The statistics produced from the GLM were otherwise not reliable due to the violation of various requirements for GLM, e.g. distribution of data, sphericity, etc.

6.3.2 Wilcoxon signed-rank test

The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test used when comparing two related samples, or repeated measurements on a single sample to assess whether their population means differ, i.e. it's a paired difference test. It can be used as an alternative to the paired Student's t-test when the population cannot be assumed to be normally distributed, or the data is on the ordinal scale (Lowry 2011). The test is named for Frank Wilcoxon (1892–1965) who, in a single paper, proposed both it and the rank-sum test for two independent samples (Wilcoxon, 1945).

Due to the small sample size and implicitly abnormal distribution of the behavioural data, we used non-parametric tests for statistical analysis. Also, given that we acquired repeated measures from the same subjects, we used a non-parametric alternative of the paired t-test – specifically, the Wilcoxon's matched pairs test.

6.3.3 Friedman test

The Friedman test is a non-parametric statistical test. Similar to the parametric repeated measures ANOVA, it is used to detect differences in treatments across multiple test attempts. The procedure involves ranking each row (or block) together, then considering the values of ranks by columns. Applicable to complete block designs, it is thus a special case of the Durbin test. Classic examples of use are:

- N wine judges each rate K different wines. Are any wines ranked consistently higher or lower than the others?
- N wines are each rated by K different judges. Are the judges' ratings consistent with each other?
- N welders each use K welding torches and the ensuing welds were rated on quality. Do any of the torches produce consistently better or worse welds?

The Friedman test is used for one-way repeated measures analysis of variance by ranks. In its use of ranks it is similar to the Kruskal-Wallis one-way analysis of variance by ranks. Friedman test is widely supported by many statistical software packages.

6.3.4 Mann-Whitney Test

The Mann-Whitney test is a non-parametric statistical hypothesis test for assessing whether one of two samples of independent observations tends to have larger values than the other. It is one of the most well-known non-parametric significance tests.

The test involves the calculation of a statistic, usually called U, whose distribution under the null hypothesis is known. In the case of small samples, the distribution is tabulated, but for sample sizes above ~20 there is a good approximation using the normal distribution. Some books tabulate statistics equivalent to U, such as the sum of ranks in one of the samples, rather than U itself. The U test is included in most modern statistical packages. It is also easily calculated by hand, especially for small samples.

6.4 Results

In this section, the results occurring from the analysis of the data derived from each of the three stages of the experiment will be presented. The analysis is going to be based on the

participants’ responses to the questions presented to them after their exposure to the 3D scenes involved in each stage of the experiment. The statistical data analysis was conducted using an industry-standard application (IBM SPSS).

6.4.1 First Stage Results

The first GLM test was conducted according to participants’ responses to the first question, which is shown in Figure 80. The GLM was based on two factors: “reality”- involving two levels, either wireframe (WF) or realistic and “day time- place”- involving seven levels, e.g. the morning, midday, afternoon and evening and indoors and outdoors locations.

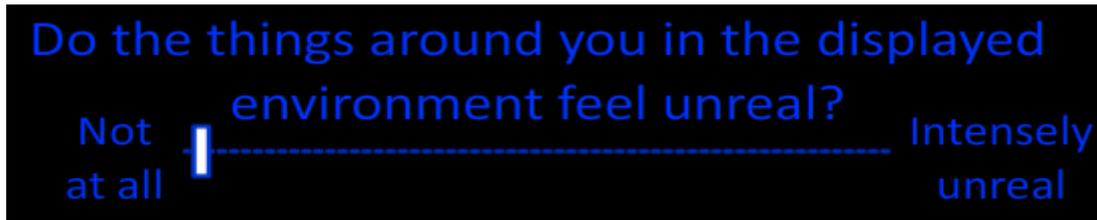


Figure 80: The first question of the questionnaire presented to participants after their free navigation in each virtual scene of the first stage of the experiment.

Table 3 displays the within-subjects variables used for the GLM test associated to the first question of the experiment.



Table 3: Configuration of the repeated-measures GLM test for the first question presented during the first stage of the experiment

Stage 1: Free navigation; question 1.

Model: within-subjects factors- realistic/ WF and place_daytime

The descriptive statistics for this test can be seen in Table 4 in relation to data acquired from twelve participants. Descriptive statistics is a branch of statistics dealing with

summarization and description of collections of data—data sets, including the concepts of arithmetic mean. Descriptive statistics inform about the mean value and the standard deviation of the participants’ responses to the first question, during the first stage of the experiment, after being exposed to each synthetic scene.

Descriptive Statistics

	Mean	Std. Deviation	N
Morning_Indoors	30.0000	18.09068	12
Midday_Indoors	34.2500	17.48831	12
Afternoon_Indoors	48.3333	20.70719	12
Evening_Indoors	42.9167	19.47706	12
Morning_Outdoors	39.1667	23.43592	12
Midday_Outdoors	37.5000	23.97916	12
Afternoon_Outdoors	46.2500	21.11925	12
Morning_Indoors_WF	76.2500	18.72347	12
Midday_Indoors_WF	74.5833	20.83030	12
Afternoon_Indoors_WF	73.3333	14.19560	12
Evening_Indoors_WF	77.9167	18.02250	12
Morning_Outdoors_WF	76.2500	18.35570	12
Midday_Outdoors_WF	64.5833	31.51178	12
Afternoon_Outdoors_WF	80.4167	13.72760	12

Table 4: Descriptive Statistics of Question 1 presented during the first stage of the experiments.

Figure 81 displays the Estimated Marginal Means for participants’ responses in Question 1, during the first stage of the experiments.

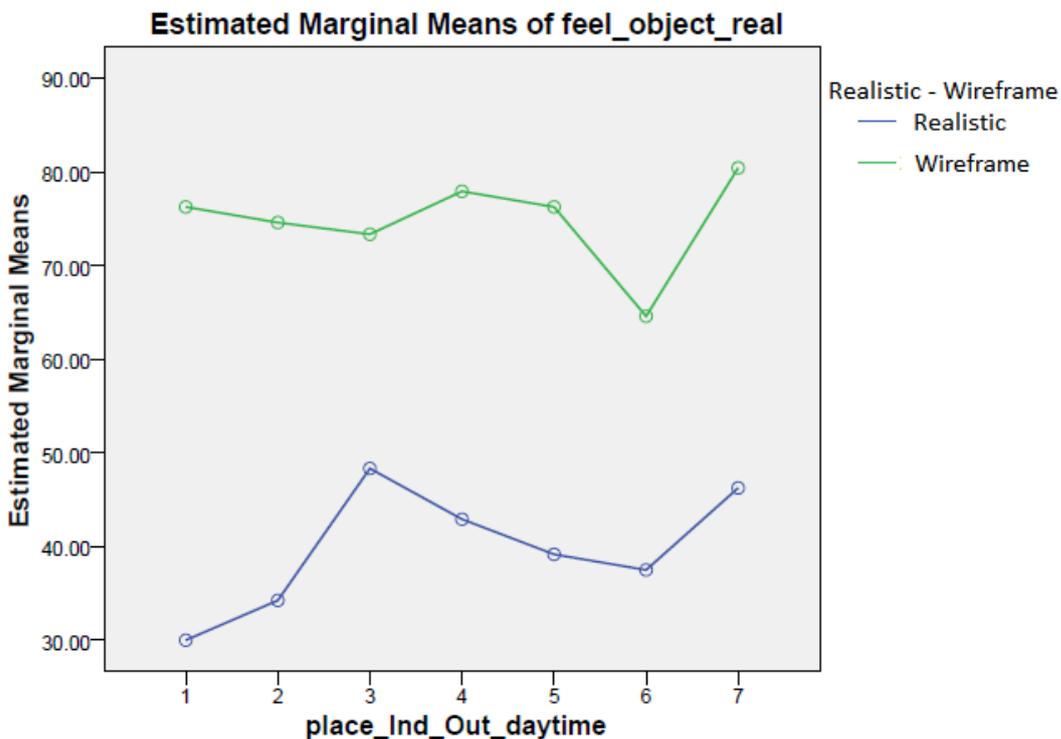


Figure 81: Estimated Marginal Means for Question 1 presented during the first stage of the experiments. X axis: 1. Morning indoor; 2. Midday indoor; 3. Afternoon indoor; 4. Evening indoor; 5. Morning outdoor; 6. Midday outdoor; 7. Afternoon outdoor.

A Wilcoxon test was conducted in relation to data derived from answering the four questions during the first stage of the experiments. The tests were conducted separately for each of the two possible locations, i.e. indoors, in the room, or outdoors, in the yard.

The configuration of the Wilcoxon test for the four questions in the indoors (room) location is displayed in Figure 82.

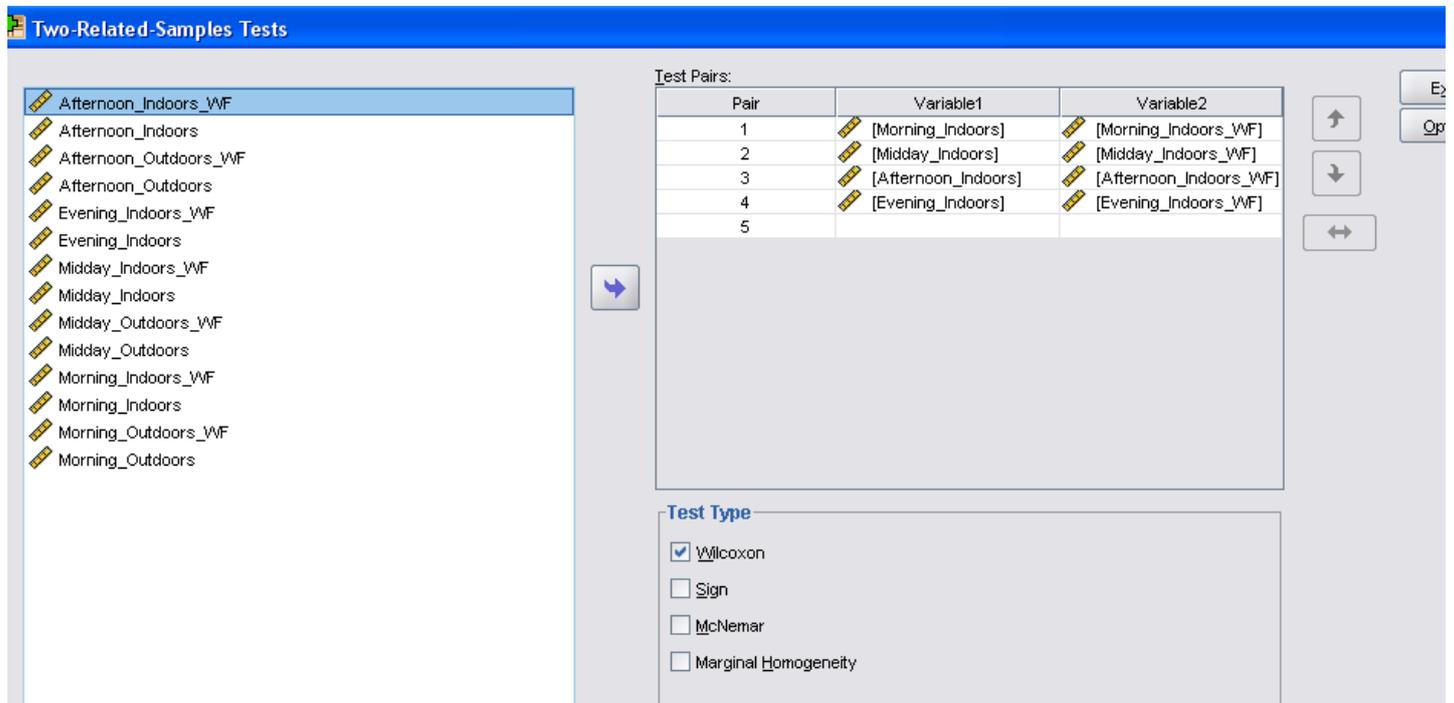


Figure 82: Configuration for the Wilcoxon test for the Indoors Locations.

The descriptive statistics of the mean responses to the first question during the first stage (Figure 80) when navigating the indoors 3D scenes only can be viewed in Table 5, as well as the standard deviation and the maximum and minimum response values.

Descriptive Statistics					
	N	Mean	Std. Deviation	Minimum	Maximum
Morning_Indoors	12	30.0000	18.09068	5.00	75.00
Midday_Indoors	12	34.2500	17.48831	10.00	75.00
Afternoon_Indoors	12	48.3333	20.70719	15.00	75.00
Evening_Indoors	12	42.9167	19.47706	20.00	80.00
Morning_Indoors_WF	12	76.2500	18.72347	40.00	100.00
Midday_Indoors_WF	12	74.5833	20.83030	25.00	100.00
Afternoon_Indoors_WF	12	73.3333	14.19560	60.00	100.00
Evening_Indoors_WF	12	77.9167	18.02250	45.00	100.00

Table 5: Descriptive Statistics for Question 1 presented during the indoors virtual scenes. WF stands for Wireframe.

Test Statistics^b

	Morning Indoors_WF - Morning Indoors	Midday Indoors_WF - Midday Indoors	Afternoon Indoors_WF - Afternoon Indoors	Evening Indoors_WF - Evening Indoors
Z	-3.062 ^a	-3.065 ^a	-2.955 ^a	-2.937 ^a
Asymp. Sig. (2-tailed)	.002	.002	.003	.003

a. Based on negative ranks.

b. Wilcoxon Signed Ranks Test

Table 6: Wilcoxon test statistics for Question 1 presented during the indoors virtual scenes.

A quick glance of the means shows a large variance between the means associated with the photorealistic scenes as opposed to the means associated with the wireframe scenes. As shown in Table 6, the statistics produced from the Wilcoxon test in relation to the first question comparing responses to pairwise photorealistic and wireframe scenes of the same time of day in the indoors only location suggest that responses after being exposed to the wireframe (WF) conditions are significantly different when compared to responses after being exposed to the photorealistic scenes indoors. More specifically, given the descriptive statistics in Table 5, we can assume that the participants felt that the objects in all the wireframe 3D scenes were perceived significantly more unreal than those in the realistic scenes.

The Wilcoxon test was conducted to analyze the data derived from participants' responses to the questions after exposure to the outdoors virtual scenes. Figure 83 shows the configuration of the Wilcoxon test.

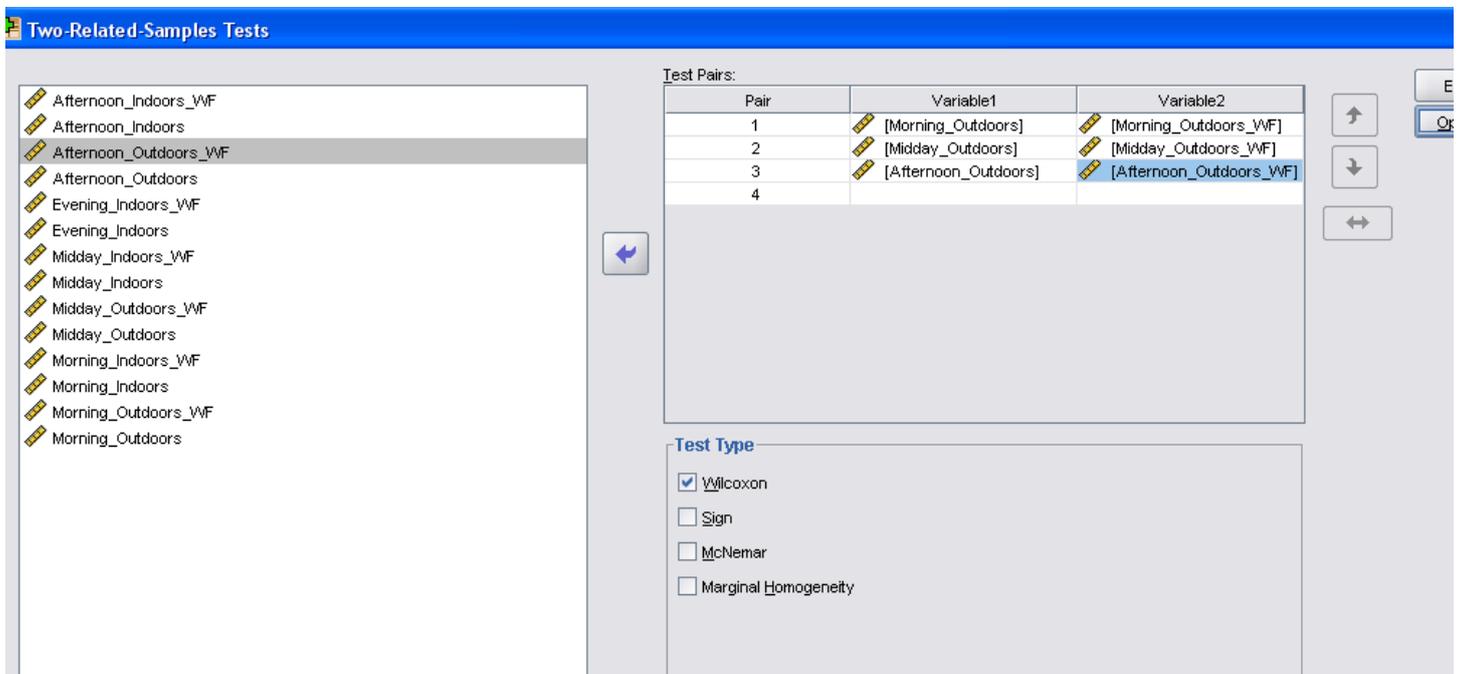


Figure 83: The Wilcoxon test configuration for the participants' responses in the questions, after navigating the outdoor synthetic scenes.

The descriptive statistics of the participants’ responses to the first question (Figure 80) after exposure to the outdoors 3D scenes are shown in Table 7. The corresponding test statistics are shown in Table 8. The statistical analysis shows that responses to the first question after being exposed to the photorealistic scenes were significantly different than those after being exposed to the wireframe scenes. The descriptive statistics can help in further interpreting that participants felt that the objects are significantly more unreal after exposure to the wireframe outdoors 3D scenes, rather than after exposure to the respective photorealistically-rendered ones.

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Morning_Outdoors	12	39.1667	23.43592	15.00	85.00
Midday_Outdoors	12	37.5000	23.97916	10.00	90.00
Afternoon_Outdoors	12	46.2500	21.11925	25.00	80.00
Morning_Outdoors_WF	12	76.2500	18.35570	35.00	100.00
Midday_Outdoors_WF	12	64.5833	31.51178	.00	100.00
Afternoon_Outdoors_WF	12	80.4167	13.72760	65.00	100.00

Table 7: Descriptive Statistics for Question 1 presented during the outdoors virtual scenes.

Test Statistics^b

	Morning Outdoors_WF - Morning_ Outdoors	Midday Outdoors_WF - Midday_ Outdoors	Afternoon Outdoors_WF - Afternoon_ Outdoors
Z	-3.061 ^a	-2.041 ^a	-2.940 ^a
Asymp. Sig. (2-tailed)	.002	.041	.003

a. Based on negative ranks.

b. Wilcoxon Signed Ranks Test

Table 8: Test Statistics for Question 1 presented during the outdoors virtual scenes.

The second question of the experiment can be seen in Figure 84. The GLM test conducted for the first question of this experiment was applied to this question as well, as can be seen in Table 9. Table 10 and Figure 85 contain the descriptive statistics and the estimated marginal means respectively, calculated according to participants’ responses in that question.



Figure 84: The second question of the questionnaire presented to participants after their free navigation in each synthetic scene during the first stage of the experiment.

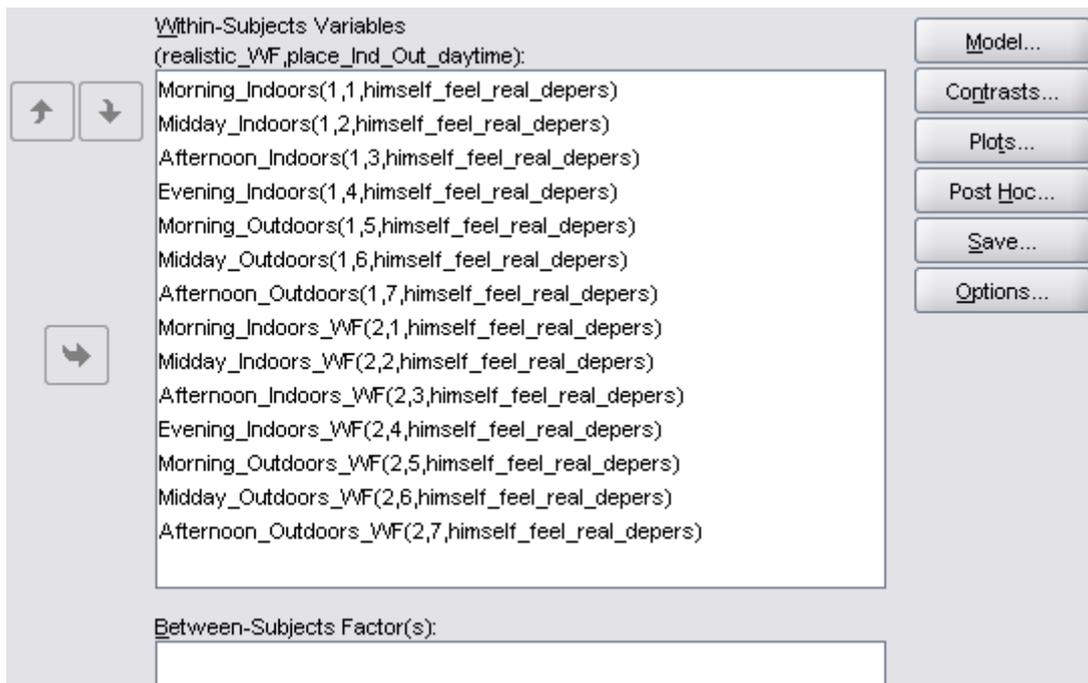


Table 9: Configuration of the repeated-measures GLM test for the second question presented during the first stage of the experiment

[Stage 1: Free navigation; question 2](#)

[Model: within-subjects factors- realistic/ WF and place_daytime](#)

Descriptive Statistics

	Mean	Std. Deviation	N
Morning_Indoors	29.1667	24.38641	12
Midday_Indoors	30.4167	22.50842	12
Afternoon_Indoors	30.4167	25.26751	12
Evening_Indoors	30.0000	25.04541	12
Morning_Outdoors	27.5000	25.18116	12
Midday_Outdoors	29.5833	24.99621	12
Afternoon_Outdoors	32.5000	26.24188	12
Morning_Indoors_WF	38.3333	31.43054	12
Midday_Indoors_WF	39.5833	31.87036	12
Afternoon_Indoors_WF	42.9167	29.42312	12
Evening_Indoors_WF	46.2500	35.74690	12
Morning_Outdoors_WF	35.0000	29.46493	12
Midday_Outdoors_WF	39.1667	30.95696	12
Afternoon_Outdoors_WF	42.5000	33.94514	12

Table 10: Descriptive Statistics for Question 2 presented during the first stage of the experiments.

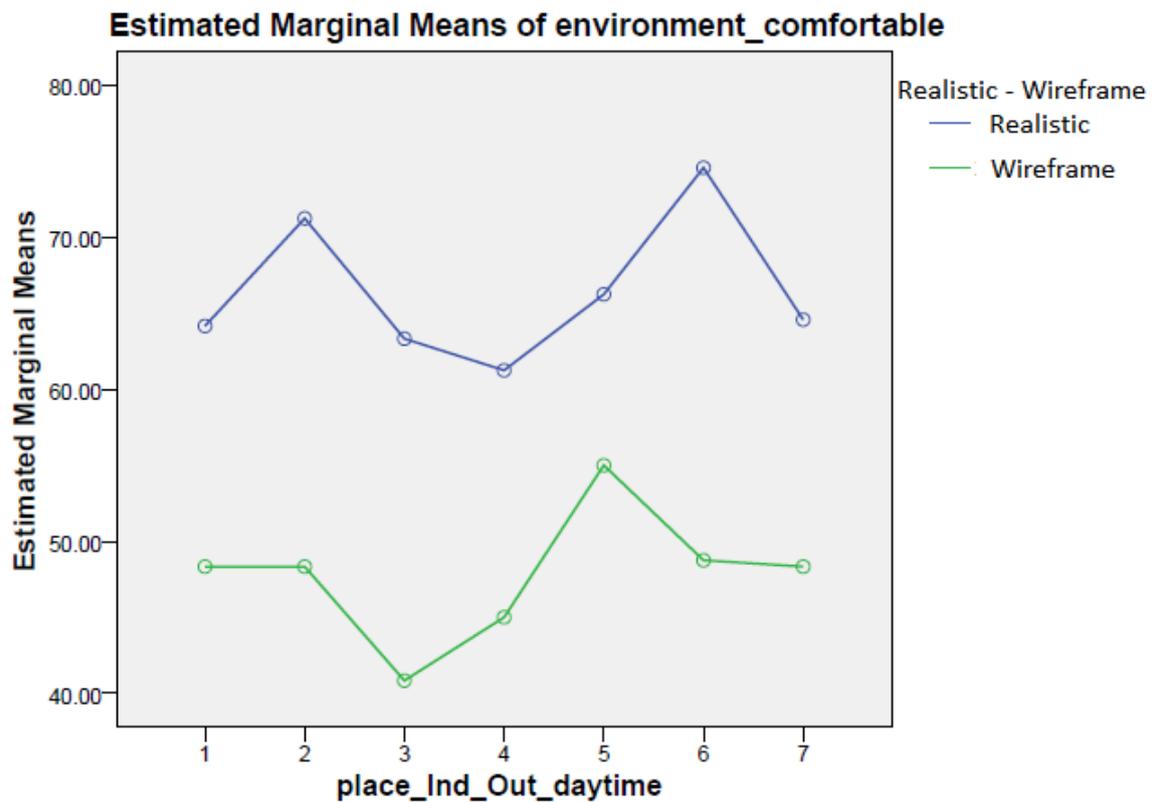


Figure 85: Estimated Marginal Means for Question 2 presented during the first stage of the experiments. X axis: 1. Morning indoor; 2. Midday indoor; 3. Afternoon indoor; 4. Evening indoor; 5. Morning outdoor; 6. Midday outdoor; 7. Afternoon outdoor.

The results of the Wilcoxon test for the second question (Figure 84) after exposure to the indoors synthetic scenes are shown in Table 11 which depicts the descriptive statistics of responses to the second question of the first stage of the experiment, including the mean values, the standard deviation and the minimum and maximum values of responses after exposure to each 3D scene. Table 12 contains the statistics for the Wilcoxon test in relation to responses to the second question. The test suggests that responses to the second question, which probes for the feeling of depersonalization of the participants after exposure to each scene, were significantly different only after exposure to the afternoon and evening conditions, resulting to people experiencing a higher degree of depersonalization after exposure to the wireframe (WF) indoor scenes. Comparing the test statistics with the respective descriptive statistics, it is shown that participants feel significantly more depersonalized while navigating the evening and afternoon indoors wireframe scenes, than the respective realistic ones. It has to be noted that responses after exposure to the afternoon indoors scene have led to near significant results ($p=0.058$).

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Morning_Indoors	12	29.1667	24.38641	.00	75.00
Midday_Indoors	12	30.4167	22.50842	.00	70.00
Afternoon_Indoors	12	30.4167	25.26751	.00	80.00
Evening_Indoors	12	30.0000	25.04541	.00	70.00
Morning_Indoors_WF	12	38.3333	31.43054	.00	90.00
Midday_Indoors_WF	12	39.5833	31.87036	.00	90.00
Afternoon_Indoors_WF	12	42.9167	29.42312	.00	80.00
Evening_Indoors_WF	12	46.2500	35.74690	.00	95.00

Table 11: Descriptive Statistics for Question 2 presented during the indoors synthetic scenes.

Test Statistics^b

	Morning Indoors WF - Morning Indoors	Midday Indoors WF - Midday Indoors	Afternoon Indoors WF - Afternoon Indoors	Evening Indoors WF - Evening Indoors
Z	-1.689 ^a	-1.193 ^a	-1.899 ^a	-2.677 ^a
Asymp. Sig. (2-tailed)	.091	.233	.058	.007

a. Based on negative ranks.

b. Wilcoxon Signed Ranks Test

Table 12: Test Statistics for Question 2 presented during the indoors synthetic scenes.

The data derived from participants’ responses to the second question (Figure 84) after exposure to the outdoors synthetic scenes resulted to the descriptive statistics and the Wilcoxon test statistics that are shown in Table 13 and Table 14 respectively. The second question investigated the level of participants’ perceived feeling of depersonalization. Participants felt significantly more depersonalized after exposure to the morning outdoors wireframe scene, than after exposure to the photorealistic one ($p=0.048$). There is also a significant trend for participants to feel more depersonalized after exposure to the afternoon outdoors wireframe scene than after exposure to the respective photorealistically-rendered scene ($p=0.57$).

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Morning_Outdoors	12	27.5000	25.18116	.00	75.00
Midday_Outdoors	12	29.5833	24.99621	.00	70.00
Afternoon_Outdoors	12	32.5000	26.24188	.00	80.00
Morning_Outdoors_WF	12	35.0000	29.46493	.00	85.00
Midday_Outdoors_WF	12	39.1667	30.95696	.00	85.00
Afternoon_Outdoors_WF	12	42.5000	33.94514	.00	95.00

Table 13: Descriptive Statistics for Question 2 presented during the outdoors synthetic scenes.

Test Statistics^b

	Morning Outdoors_WF - Morning Outdoors	Midday Outdoors_WF - Midday Outdoors	Afternoon Outdoors_WF - Afternoon Outdoors
Z	-1.976 ^a	-1.703 ^a	-1.904 ^a
Asymp. Sig. (2-tailed)	.048	.089	.057

a. Based on negative ranks.
b. Wilcoxon Signed Ranks Test

Table 14: Test Statistics for Question 2 presented during the outdoors synthetic scenes.

The third question (Figure 86) the participants were required to answer, after navigating each of the 3D scenes involved during the first stage of the experiment resulted to the means shown in Figure 86. The third question investigated the level of comfort in the synthetic scene. The variables created for the GLM test can be seen in Table 15. The descriptive statistics and the estimated marginal means computed from the test are shown in Table 16 and Figure 87 respectively.

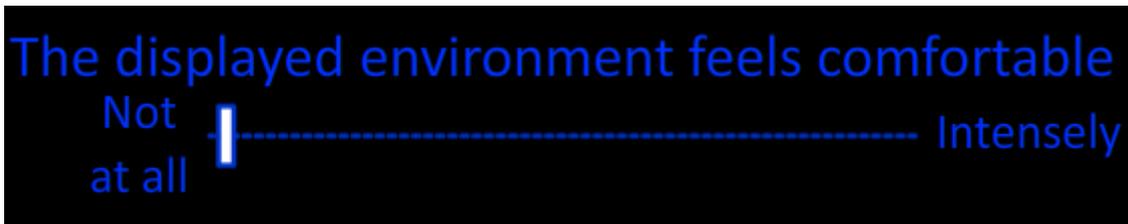


Figure 86: The third question of the questionnaire presented to participants after their free navigation in each synthetic scene during the first stage of the experiment.

Within-Subjects Variables (realistic_WF_place_Ind_Out_daytime):
Morning_Indoors(1,1,environment_comfortable)
Midday_Indoors(1,2,environment_comfortable)
Afternoon_Indoors(1,3,environment_comfortable)
Evening_Indoors(1,4,environment_comfortable)
Morning_Outdoors(1,5,environment_comfortable)
Midday_Outdoors(1,6,environment_comfortable)
Afternoon_Outdoors(1,7,environment_comfortable)
Morning_Indoors_WVF(2,1,environment_comfortable)
Midday_Indoors_WVF(2,2,environment_comfortable)
Afternoon_Indoors_WVF(2,3,environment_comfortable)
Evening_Indoors_WVF(2,4,environment_comfortable)
Morning_Outdoors_WVF(2,5,environment_comfortable)
Midday_Outdoors_WVF(2,6,environment_comfortable)
Afternoon_Outdoors_WVF(2,7,environment_comfortable)
Between-Subjects Factor(s):

Table 15: Configuration of the repeated-measures GLM test for the third question presented during the first stage of the experiment

Stage 1: Free navigation; question 3

Model: within-subjects factors- realistic/ WF and place_daytime

Descriptive Statistics

	Mean	Std. Deviation	N
Morning_Indoors	64.1667	21.72486	12
Midday_Indoors	71.2500	11.50593	12
Afternoon_Indoors	63.3333	16.14330	12
Evening_Indoors	61.2500	21.01136	12
Morning_Outdoors	66.2500	24.32031	12
Midday_Outdoors	74.5833	12.51514	12
Afternoon_Outdoors	64.5833	8.64931	12
Morning_Indoors_WF	48.3333	31.93269	12
Midday_Indoors_WF	48.3333	21.46173	12
Afternoon_Indoors_WF	40.8333	20.76200	12
Evening_Indoors_WF	45.0000	24.12091	12
Morning_Outdoors_WF	55.0000	29.46493	12
Midday_Outdoors_WF	48.7500	21.96330	12
Afternoon_Outdoors_WF	48.3333	23.77292	12

Table 16: Descriptive Statistics of Question 3 presented during the first stage of the experiments.

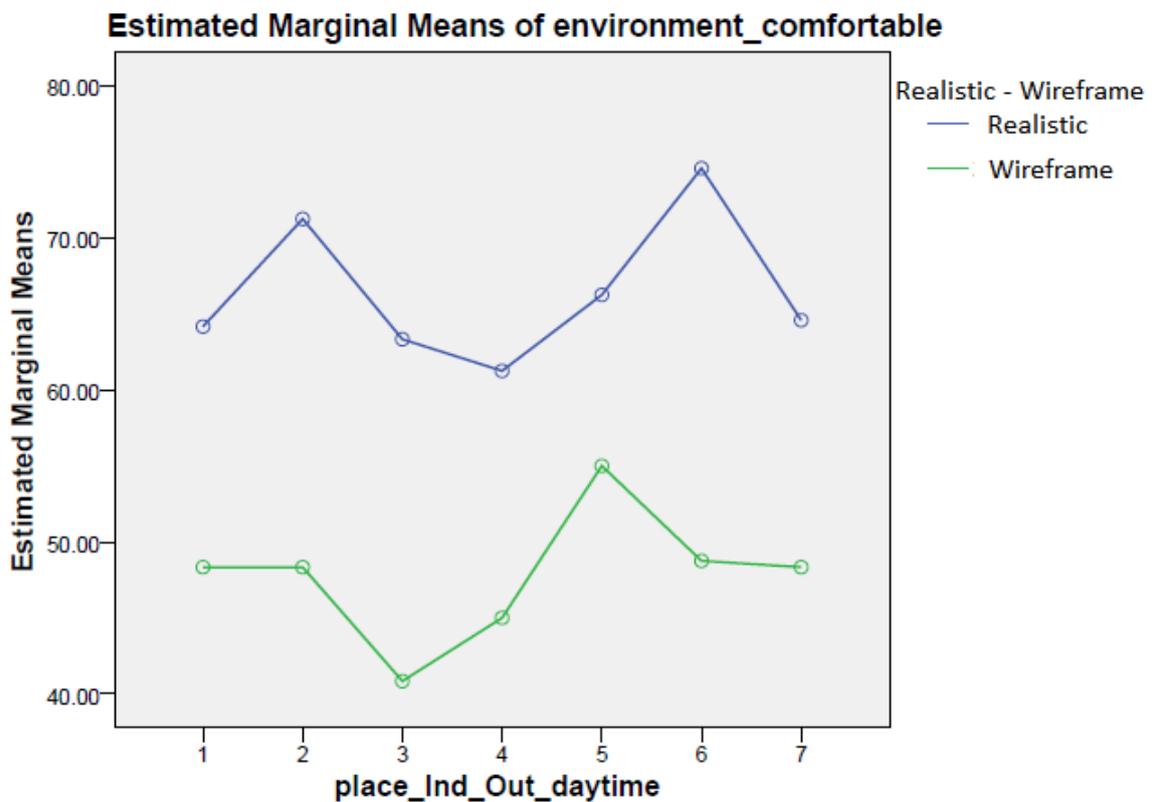


Figure 87: Estimated Marginal Means for Question 3 presented during the first stage of the experiments. X axis: 1. Morning indoor; 2. Midday indoor; 3. Afternoon indoor; 4. Evening indoor; 5. Morning outdoor; 6. Midday outdoor; 7. Afternoon outdoor

The results of the Wilcoxon test for the responses to the third question after exposure to the indoors synthetic scenes are shown in Figure 86. Table 17 shows the descriptive statistics in relation to the responses to the third question, while Table 18 provides the test statistics results. The third question explores how comfortable participants feel after exposure to each 3D scene. The results indicate that participants felt the environment to be more comfortable after exposure to photorealistically-rendered scenes, especially in relation during midday, afternoon and evening. Interestingly, there were no statistically significant differences in relation to perceived comfort after exposure to the morning scene between photorealistically-rendered and wireframe scene. These results, compared with the respective descriptive statistics, show that participants feel significantly more comfortable during midday, afternoon and evening after exposure to the photorealistically-rendered synthetic scenes, than in the respective wireframe ones.

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Morning_Indoors	12	64.1667	21.72486	.00	80.00
Midday_Indoors	12	71.2500	11.50593	50.00	90.00
Afternoon_Indoors	12	63.3333	16.14330	35.00	90.00
Evening_Indoors	12	61.2500	21.01136	.00	80.00
Morning_Indoors_WF	12	48.3333	31.93269	.00	90.00
Midday_Indoors_WF	12	48.3333	21.46173	.00	70.00
Afternoon_Indoors_WF	12	40.8333	20.76200	.00	70.00
Evening_Indoors_WF	12	45.0000	24.12091	.00	90.00

Table 17: Descriptive Statistics for Question 3 presented during the indoors virtual scenes.

Test Statistics^b

	Morning Indoors_WF - Morning Indoors	Midday Indoors_WF - Midday Indoors	Afternoon Indoors_WF - Afternoon Indoors	Evening Indoors_WF - Evening Indoors
Z	-1.738 ^a	-2.654 ^a	-2.913 ^a	-2.460 ^a
Asymp. Sig. (2-tailed)	.082	.008	.004	.014

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 18: Test Statistics for Question 3 presented during the indoors virtual scenes.

Table 19 contains the descriptive statistics for the responses to the third question (Figure 86) during the first stage of the experiment after exposure to the outdoors scenes. The results, as shown in Table 20, suggest that participants feel after exposure to the photorealistically-rendered outdoors 3D scenes significantly more comfortable than after exposure to the respective wireframe ones. This applies to all times of day, although, it has to be noted that responses after exposure to the afternoon outdoors scenes approach statistical significance, $p = 0.053$.

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Morning_Outdoors	12	66.2500	24.32031	.00	90.00
Midday_Outdoors	12	74.5833	12.51514	50.00	100.00
Afternoon_Outdoors	12	64.5833	8.64931	50.00	75.00
Morning_Outdoors_WF	12	55.0000	29.46493	.00	90.00
Midday_Outdoors_WF	12	48.7500	21.96330	.00	75.00
Afternoon_Outdoors_WF	12	48.3333	23.77292	.00	90.00

Table 19: Descriptive Statistics for Question 3 presented during the outdoors virtual scenes.

Test Statistics^b

	Morning Outdoors_WF - Morning_ Outdoors	Midday Outdoors_WF - Midday_ Outdoors	Afternoon Outdoors_WF - Afternoon_ Outdoors
Z	-2.094 ^a	-2.938 ^a	-1.933 ^a
Asymp. Sig. (2-tailed)	.036	.003	.053

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 20: Test Statistics for Question 3 presented during the outdoors virtual scenes.

The fourth question (Figure 88) the participants were required to answer, after navigating each of the 3D scenes involved during the first stage of the experiment resulted to the means shown in Figure 88. The fourth question investigated participants’ perceived feeling of presence during exposure to the synthetic scenes. The variables created for the GLM test can be seen in Table 21. The descriptive statistics and the estimated marginal means computed from the test are shown in Table 22 and Figure 89 respectively.

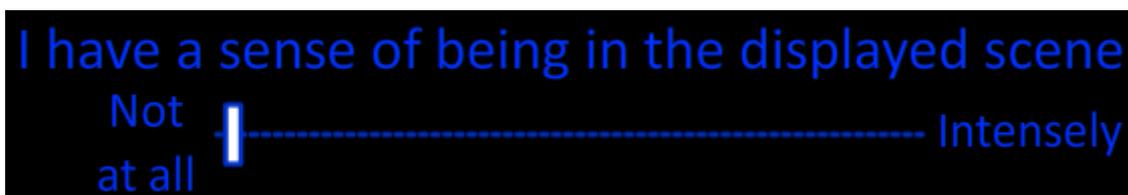


Figure 88: The fourth question of the questionnaire presented to participants after their free navigation in each synthetic scene during the first stage of the experiment.

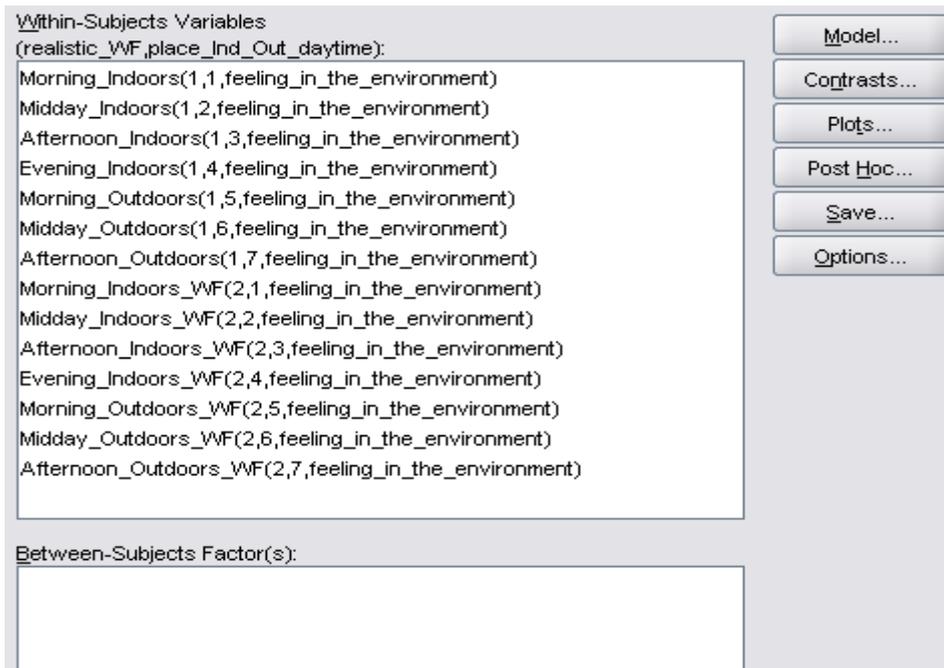


Table 21: Configuration of the repeated-measures GLM test for the fourth question presented during the first stage of the experiment

[Stage 1: Free navigation; question 4](#)

[Model: within-subjects factors- realistic/ WF and place_daytime](#)

Descriptive Statistics

	Mean	Std. Deviation	N
Morning_Indoors	47.5000	24.72577	12
Midday_Indoors	54.1667	26.78478	12
Afternoon_Indoors	52.0833	23.49645	12
Evening_Indoors	54.1667	27.45520	12
Morning_Outdoors	55.4167	26.49686	12
Midday_Outdoors	58.7500	28.53427	12
Afternoon_Outdoors	54.5833	25.80154	12
Morning_Indoors_WVF	47.5000	29.50347	12
Midday_Indoors_WVF	44.1667	26.44319	12
Afternoon_Indoors_WVF	39.5833	26.92231	12
Evening_Indoors_WVF	50.4167	26.58249	12
Morning_Outdoors_WVF	49.1667	28.35115	12
Midday_Outdoors_WVF	45.0000	26.71397	12
Afternoon_Outdoors_WVF	44.1667	26.27073	12

Table 22: Descriptive Statistics of Question 4 presented during the first stage of the experiments.

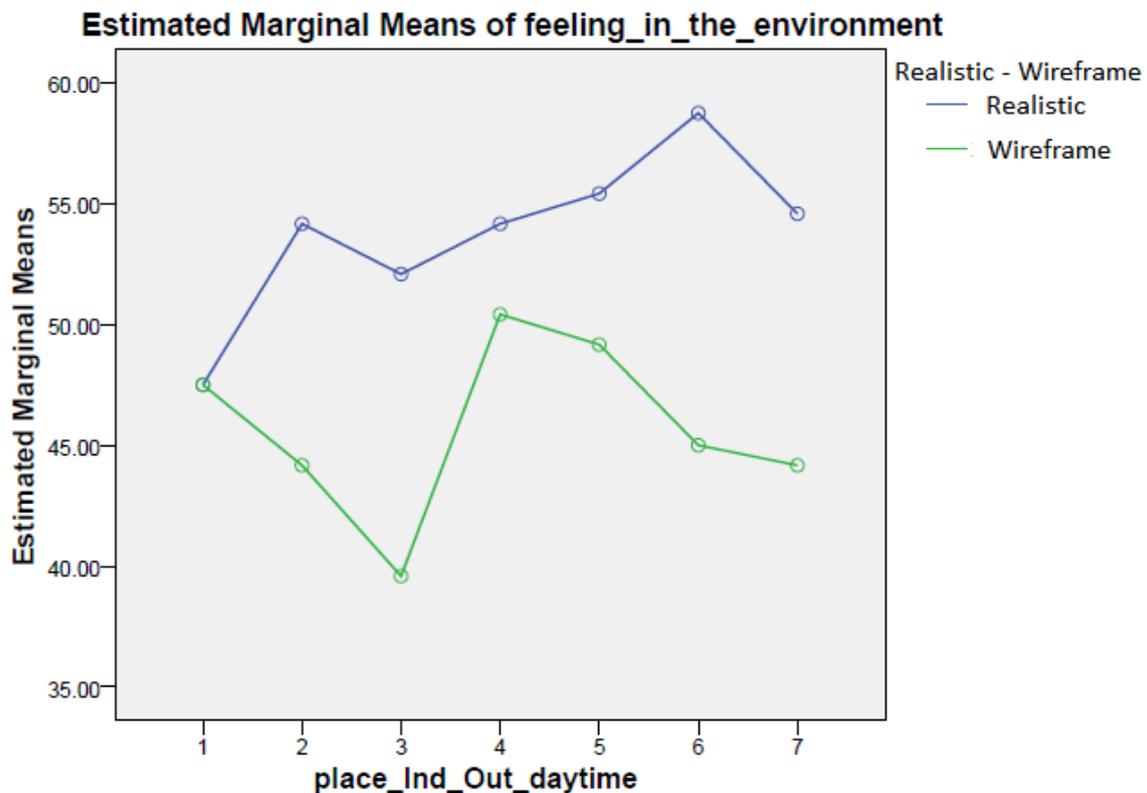


Figure 89: Estimated Marginal Means for Question 4 presented during the first stage of the experiments. X axis: 1. Morning indoor; 2. Midday indoor; 3. Afternoon indoor; 4. Evening indoor; 5. Morning outdoor; 6. Midday outdoor; 7. Afternoon outdoor.

A Wilcoxon test was conducted in relation to the indoors location for participants' responses to the fourth question during the first stage of the experiment (Figure 88). Table 23 shows the descriptive statistics of responses to the fourth question. Question 4 explored participants' feeling of presence after exposure to each 3D scene. The test statistics of the Wilcoxon test, as shown in Table 24, indicate that participants have a significantly different feeling of presence after exposure to the photorealistically-rendered scenes, especially during midday ($p=0.065$) and afternoon ($p<0.05$).

It is, therefore, revealed that participants have a significantly higher feeling of presence after exposure to the photorealistically-rendered scenes, rather than after exposure to the wireframe scenes, particularly during midday and afternoon. More specifically, participants feel significantly more present when navigating the photorealistically-rendered afternoon scene than the wireframe afternoon scene. Moreover, there is a trend of significance for participants to have a higher feeling of presence after navigating the realistic midday scene, than the respective wireframe.

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Morning_Indoors	12	47.5000	24.72577	.00	70.00
Midday_Indoors	12	54.1667	26.78478	.00	75.00
Afternoon_Indoors	12	52.0833	23.49645	.00	80.00
Evening_Indoors	12	54.1667	27.45520	.00	80.00
Morning_Indoors_WF	12	47.5000	29.50347	.00	90.00
Midday_Indoors_WF	12	44.1667	26.44319	.00	80.00
Afternoon_Indoors_WF	12	39.5833	26.92231	.00	70.00
Evening_Indoors_WF	12	50.4167	26.58249	.00	90.00

Table 23: Descriptive Statistics for Question 4 presented during the indoors virtual scenes.

Test Statistics^b

	Morning Indoors WF - Morning Indoors	Midday Indoors WF - Midday Indoors	Afternoon Indoors WF - Afternoon Indoors	Evening Indoors WF - Evening Indoors
Z	-.237 ^a	-1.844 ^a	-2.106 ^a	-1.074 ^a
Asymp. Sig. (2-tailed)	.812	.065	.035	.283

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 24: Test Statistics for Question 4 presented during the indoors virtual scenes.

A Wilcoxon test was conducted in relation to the responses to the fourth question (Figure 88) during the first stage of the experiment after exposure to the outdoors scenes. The produced descriptive statistics are shown in Table 25 and the test statistics in Table 26. Notably, the test statistics indicate that there is no significant difference of the participants' responses to the fourth question after exposure to the photorealistically-rendered outdoors 3D scenes and their respective wireframe version.

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Morning_Outdoors	12	55.4167	26.49686	.00	85.00
Midday_Outdoors	12	58.7500	28.53427	.00	90.00
Afternoon_Outdoors	12	54.5833	25.80154	.00	85.00
Morning_Outdoors_WF	12	49.1667	28.35115	.00	90.00
Midday_Outdoors_WF	12	45.0000	26.71397	.00	80.00
Afternoon_Outdoors_WF	12	44.1667	26.27073	.00	85.00

Table 25: Descriptive Statistics for Question 4 presented during the outdoors virtual scenes.

	Morning Outdoors_WF - Morning_ Outdoors	Midday Outdoors_WF - Midday_ Outdoors	Afternoon Outdoors_WF - Afternoon_ Outdoors
Z	-1.606 ^a	-1.585 ^a	-1.687 ^a
Asymp. Sig. (2-tailed)	.108	.113	.092

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 26: Test Statistics for Question 4 presented during the outdoors virtual scenes.

6.4.2 First Stage Brain Imaging Results

The analysis of the brain image data acquired through the fMRI scanner, while participants were allowed to freely navigate in the synthetic scenes involved in the first stage of the experiment, led to the results presented in Figure 90 and Figure 91. The figures are presented as preliminary results only since they were created for a small statistical threshold ($p = 0.05$) and have reduced statistical significance. The main preliminary result is that brain activation was influenced by the fidelity-quality of the displayed environment between the photorealistically-rendered and the wireframe synthetic scenes, only after exposure to the indoors synthetic scenes.

More specifically, there is a greater activation in regions previously implicated in “spatial navigation”, including navigation in VEs, after exposure to the indoors photorealistically-rendered scenes compared to the respective wireframe scenes. Examples of these regions, which are related to spatial navigation, are the superior frontal gyrus, the posterior cingulate and the cerebellum. This would suggest that in the realistic indoors virtual scenes, participants were practically “navigating” and exploring the environment. The same brain activity is not evoked by the wireframe synthetic scenes. Also, it is important to note that for the outdoors synthetic scenes this is not true, since there is no difference in brain activation between the photorealistically-rendered and the wireframe outdoors synthetic scenes.

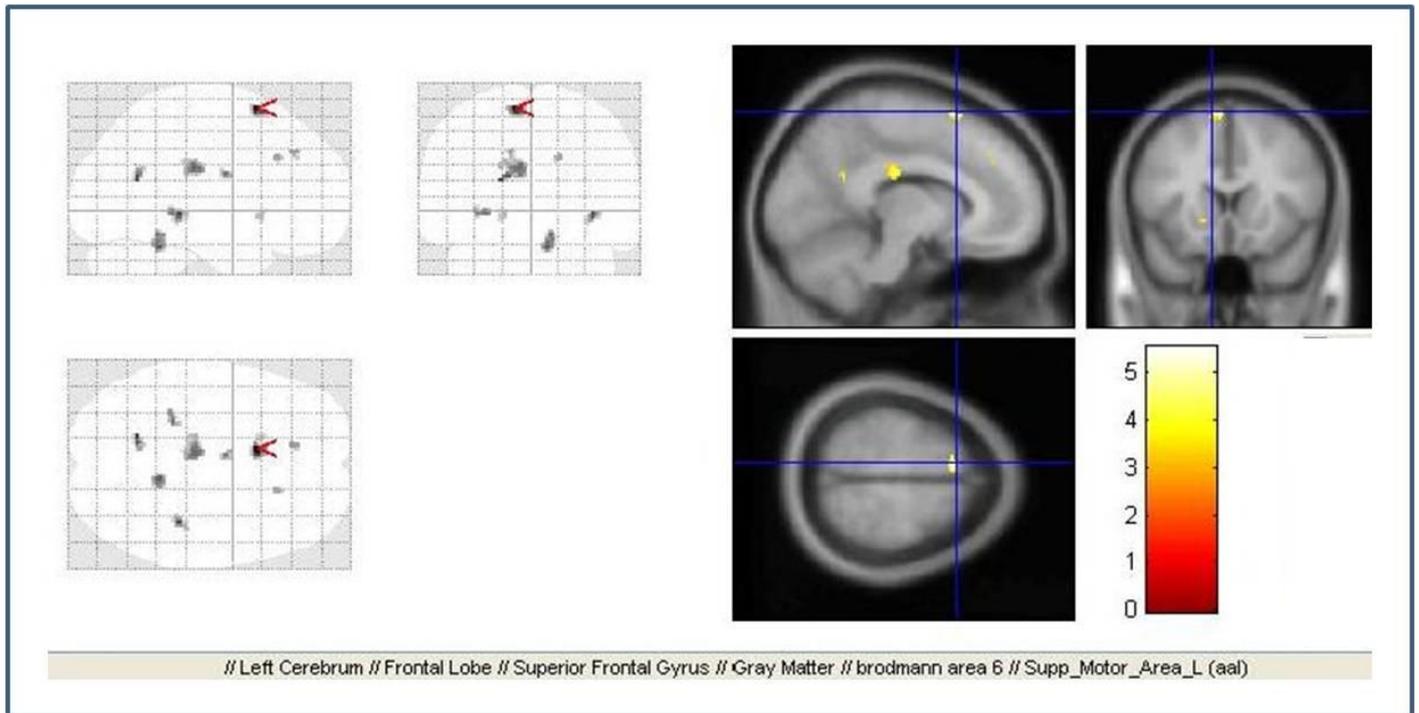


Figure 90: Brain images acquired from the fMRI scanner during the first stage of the experiment. This image shows midline regions like superior frontal gyrus and posterior cingulate.

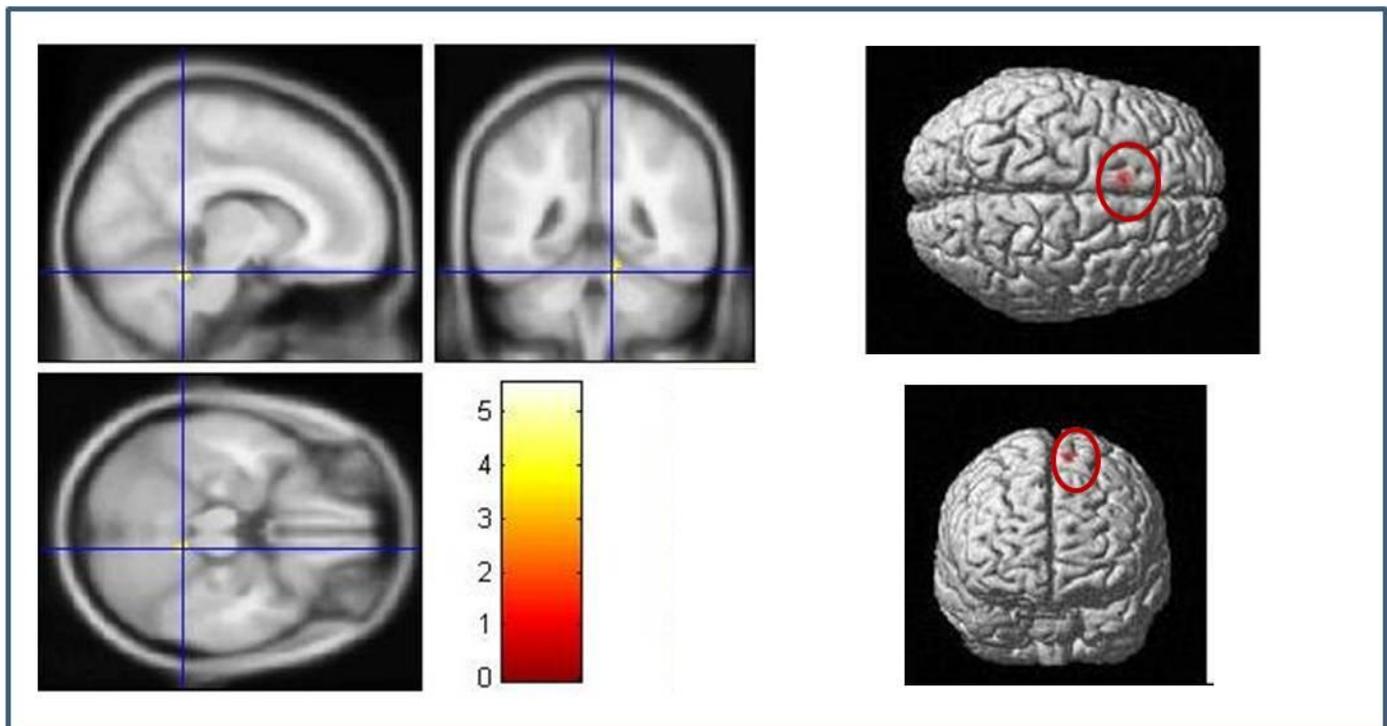


Figure 91: Brain images acquired from the fMRI scanner during the first stage of the experiment. This image shows the cerebellum and the superior frontal gyrus cluster.

6.4.3 Second Stage Results

In order to study and plot the data acquired from participants' responses during the second stage of the experiments when being exposed to the photorealistic 3D scenes, a repeated measures GLM test was conducted of two factors, i.e. the time of day, either morning, midday, afternoon or evening and valence, either pleasant, neutral or unpleasant. It is important to note that the means represent scores of a specific valence during a certain day time and in a certain location, e.g. indoors or outdoors. For example:

$$\text{Morning Indoors Pleasant} = (\text{picture 18} + \text{picture 3} + \text{picture 6})/3.$$

Figure 92 shows the question that was presented to participants after each displayed emotional image, in order to acquire the participant's rating of the image. The possible ratings ranged from 0 to 100, with 0 meaning *Intensely Unpleasant* and 100 meaning *Intensely Pleasant*. If the cursor was in the middle of the scale, with value of 50, the emotional image was assumed to be rated as neutral.

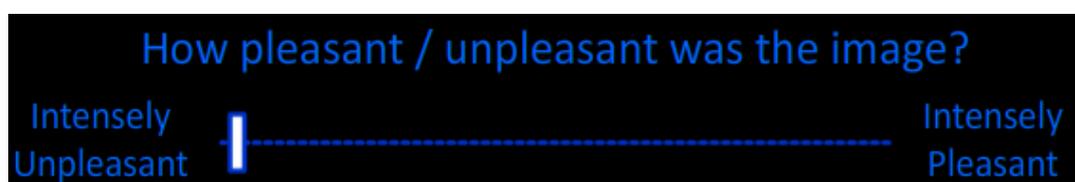


Figure 92: The question shown after each emotional image was displayed, probing for the participant's rating of the image.

The descriptive statistics of the responses acquired after exposure to the indoors 3D scenes are shown in Table 27. The plot of the estimated marginal means is shown in Figure 93. From the estimated marginal means, it is indicated that *pleasant* pictures are seen as less pleasant in the morning, while *unpleasant* pictures are seen as most unpleasant in the evening when viewing the indoors synthetic scenes.

Descriptive Statistics

	Mean	Std. Deviation	N
Morn_Ind_Pleasant	61.5152	9.64522	11
Morn_Ind_Neu	55.1515	10.09550	11
Morn_Ind_Unpl	25.4545	13.12527	11
Midd_Ind_Pleasant	69.8485	11.04170	11
Midd_Ind_Neu	50.6061	3.82047	11
Midd_Ind_Unpl	22.8788	12.86880	11
Aft_Ind_Pleasant	67.7273	9.69744	11
Aft_Ind_Neu	52.8788	7.67720	11
Aft_Ind_Unpl	24.0909	11.57976	11
Even_Ind_Pleasant	67.8788	8.36962	11
Even_Ind_Neu	55.7576	6.02604	11
Even_Ind_Unpl	17.7273	13.04421	11

Table 27: Descriptive statistics of the emotional image ratings in the indoors virtual scenes.

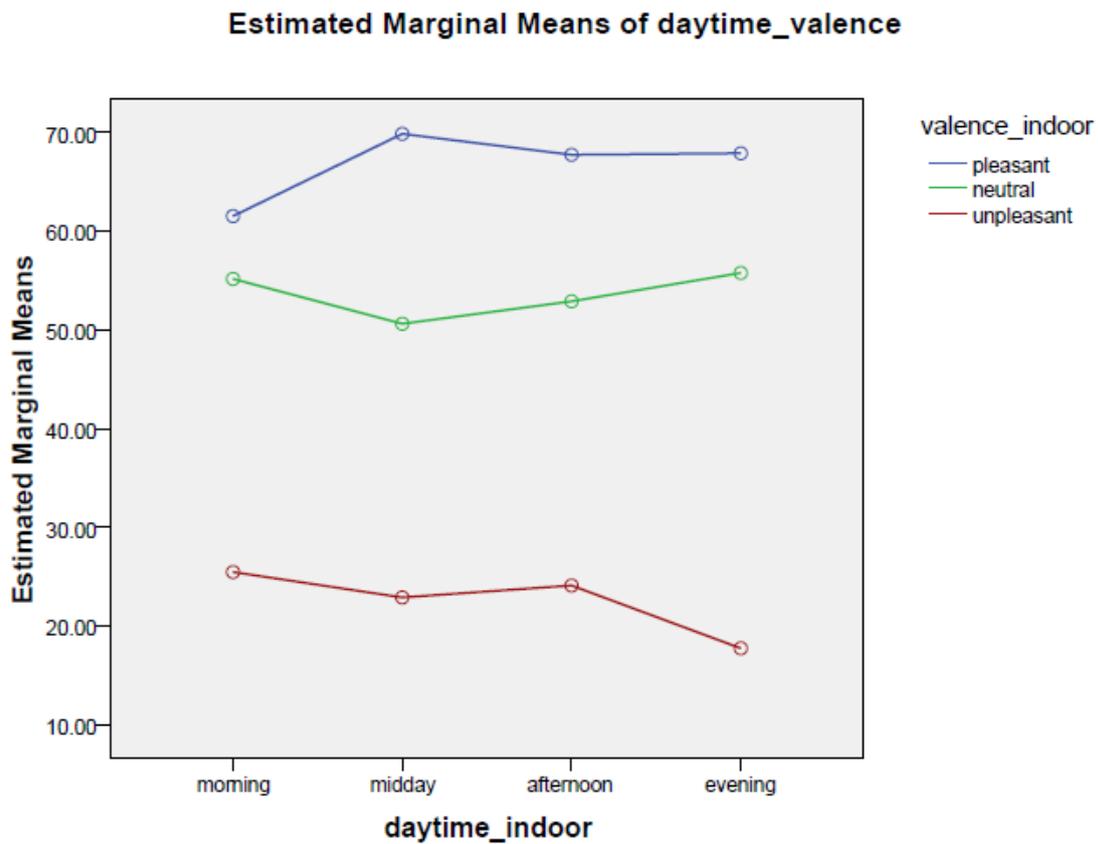


Figure 93: Estimated Marginal Means of the emotional image ratings in the indoors virtual scenes.

A repeated measures GLM test was applied to the data acquired from participants' responses to the question (Figure 92) displayed after each emotional image was displayed when being exposed to the outdoors 3D scenes during the second stage of the experiment. Table 28 shows the mean emotional image ratings after being exposed to the outdoors virtual scenes, i.e. the mean values, the standard deviation and the sample size. The respective estimated marginal means, as seen in Figure 94, show that "pleasant" pictures are seen as more pleasant in the afternoon, while "unpleasant" pictures are seen as less unpleasant in the afternoon, in the outdoors scenes.

Descriptive Statistics

	Mean	Std. Deviation	N
Morn_Out_Pleasant	66.8182	8.98933	11
Morn_Out_Neu	52.8788	4.08866	11
Morn_Out_Unpl	26.9697	12.35624	11
Midd_Out_Pleasant	67.8788	12.02061	11
Midd_Out_Neu	50.7576	3.36350	11
Midd_Out_Unpl	25.3030	12.66707	11
Aft_Out_Pleasant	72.2727	12.67703	11
Aft_Out_Neu	50.9091	4.90825	11
Aft_Out_Unpl	31.2121	11.69261	11

Table 28: Descriptive Statistics of the emotional image ratings after viewing the outdoors virtual scenes.

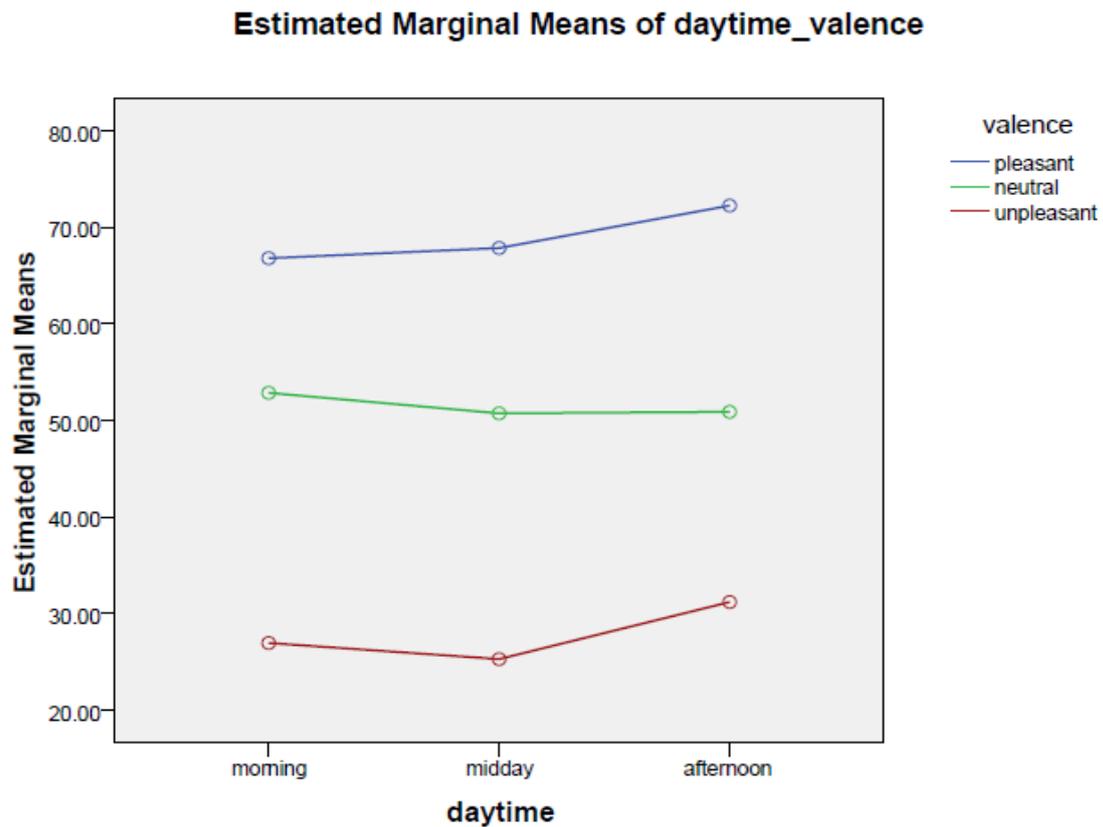


Figure 94: Estimated Marginal Means of the emotional image ratings during exposure to the outdoors synthetic scenes.

The statistical analysis test conducted on the data was the Wilcoxon signed-rank test. The test was based on the GLM plots, which suggested potential significance between midday pleasant and morning pleasant, morning neutral and midday neutral, evening neutral and midday neutral and morning unpleasant and evening unpleasant.

The descriptive statistics with the mean values, the standard deviation and the minimum and maximum values for the ratings of the emotional images displayed during each synthetic scene are shown in Table 29. The corresponding test statistics are shown in Table 30. The results suggest that after exposure to the indoors scenes, previously rated pictures as pleasant were significantly rated as more pleasant during midday compared to morning. Neutral pictures were perceived as more “pleasant” during the evening compared to midday. Unpleasant pictures were perceived as less unpleasant during the morning compared to the evening.

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Midd_Ind_Pleasant	11	69.8485	11.04170	55.00	91.67
Morn_Ind_Neu	11	55.1515	10.09550	43.33	80.00
Even_Ind_Neu	11	55.7576	6.02604	50.00	68.33
Morn_Ind_Unpl	11	25.4545	13.12527	.00	43.33
Morn_Ind_Pleasant	11	61.5152	9.64522	46.67	76.67
Midd_Ind_Neu	11	50.6061	3.82047	45.00	55.00
Even_Ind_Unpl	11	17.7273	13.04421	.00	40.00

Table 29: Descriptive Statistics for the Wilcoxon test on the emotional image ratings during exposure to the specified indoors synthetic scenes.

Test Statistics^b

	Morn_Ind_Pleasant - Midd_Ind_Pleasant	Midd_Ind_Neu - Morn_Ind_Neu	Midd_Ind_Neu - Even_Ind_Neu	Even_Ind_Unpl - Morn_Ind_Unpl
Z	-2.199 ^a	-1.735 ^a	-2.077 ^a	-2.250 ^a
Asymp. Sig. (2-tailed)	.028	.083	.038	.024

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 30: Wilcoxon test statistics on the emotional image ratings during exposure to the specified indoors synthetic scenes.

A Wilcoxon signed-rank test was conducted based on the GLM plots of the participants’ responses after exposure to the outdoors 3D scenes during the second stage of the experiment. This test suggested no overall significance.

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Aft_Out_Pleasant	11	72.2727	12.67703	55.00	88.33
Aft_Out_Unpl	11	31.2121	11.69261	13.33	48.33
Morn_Out_Pleasant	11	66.8182	8.98933	56.67	88.33
Morn_Out_Unpl	11	26.9697	12.35624	1.67	45.00

Table 31: Descriptive statistics of Wilcoxon test on the emotional image ratings during exposure to the specified outdoors scenes.

Test Statistics^b

	Morn_Out_Pleasant - Aft_Out_Pleasant	Morn_Out_Unpl - Aft_Out_Unpl
Z	-1.123 ^a	-1.009 ^a
Asymp. Sig. (2-tailed)	.262	.313

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 32: Wilcoxon test statistics on the emotional image ratings during exposure to the specified outdoors synthetic scenes.

A Wilcoxon test was carried out for the participants’ responses during the second stage of the experiment after exposure to the morning, midday and afternoon 3D scenes, including only participant ratings when viewing the unpleasant images. The test resulted to the descriptive statistics shown in Table 33 and significance statistics shown in Table 34. These suggest that there is a significant difference of participant ratings when viewing unpleasant images when exposed to the midday outdoors scene, compared to the afternoon one: the unpleasant pictures were seen as significantly less “unpleasant” in the afternoon.

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Aft_Out_Unpl	11	31.2121	11.69261	13.33	48.33
Morn_Out_Unpl	11	26.9697	12.35624	1.67	45.00
Midd_Out_Unpl	11	25.3030	12.66707	6.67	45.00

Table 33: Descriptive statistics of the Wilcoxon test on the unpleasant emotional image ratings during exposure to the morning, midday and afternoon outdoors synthetic scenes.

Test Statistics^b

	Morn_Out_Unpl - Aft_Out_Unpl	Midd_Out_Unpl - Aft_Out_Unpl
Z	-1.009 ^a	-2.567 ^a
Asymp. Sig. (2-tailed)	.313	.010

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 34: Wilcoxon test statistics on the unpleasant emotional images during exposure to the morning, midday and afternoon outdoors synthetic scenes.

6.4.4 Third Stage Results

During the third stage of the experiments, the participants were exposed to the evening indoors 3D scenes, viewed both when wireframe and photorealistic, under different lighting conditions varied in terms of the type of the indoors artificial light type. Under each light type, the four questions presented to participants during the first stage were displayed and the participants were required to respond. The data analysis involved the two tests conducted for the data acquired during the previous stages of the experiment, a repeated-measures GLM test and a non-parametric Wilcoxon test, for the participants' responses to the questions acquired during the third stage of the experiment.

It is important to note that the second question (Figure 84) was dropped, due to the high variability across subjects. While two participants gave zero answers under all the luminescence conditions communicating that they "felt themselves not at all unreal", certain participants answered even with a score of 85. One participant was excluded from the results, since his responses were *zero* under all luminescence conditions when viewing both the realistic and the wireframe synthetic scenes.

The repeated-measures GLM test was conducted including two factors, i.e. the "reality" factor of two levels being the realistic and the wireframe version of the evening indoors 3D scene and the "luminescence" factor of six levels representing the light types employed, e.g. the 40 Watt Incandescent Tungsten, the 100 Watt Incandescent Tungsten, the Halogen, the Carbon Arc, the Standard Fluorescent and the Cool White Fluorescent. The test was conducted simply to obtain the plots of the means across groups for the four questions and visualize the data, since the stats were not reliable due to the violation of various requirements of GLM analysis, e.g. distribution, sphericity, etc.

For the responses to the first question (Figure 80) during the third stage of the experiment, the produced descriptive statistics of the GLM test are shown in Table 35 and the respective estimated marginal means are presented in Figure 95.

Descriptive Statistics

	Mean	Std. Deviation	N
Tungsten_40W	44.58	19.360	12
Tungsten_40W_WF	73.33	18.257	12
Tungsten_100W	45.00	15.226	12
Tungsten_100W_WF	70.83	18.688	12
Halogen	43.33	15.126	12
Halogen_WF	67.92	21.262	12
CarbonArc	42.50	17.645	12
CarbonArc_WF	75.00	17.838	12
Standard_Fluorescent	45.42	22.101	12
Standard_Fluorescent_WF	74.17	13.286	12
Cool_White_Fluorescent	47.92	19.477	12
Cool_White_Fluorescent_WF	72.92	15.733	12

Table 35: Descriptive statistics for the GLM test for the first question during the third stage of the experiment.

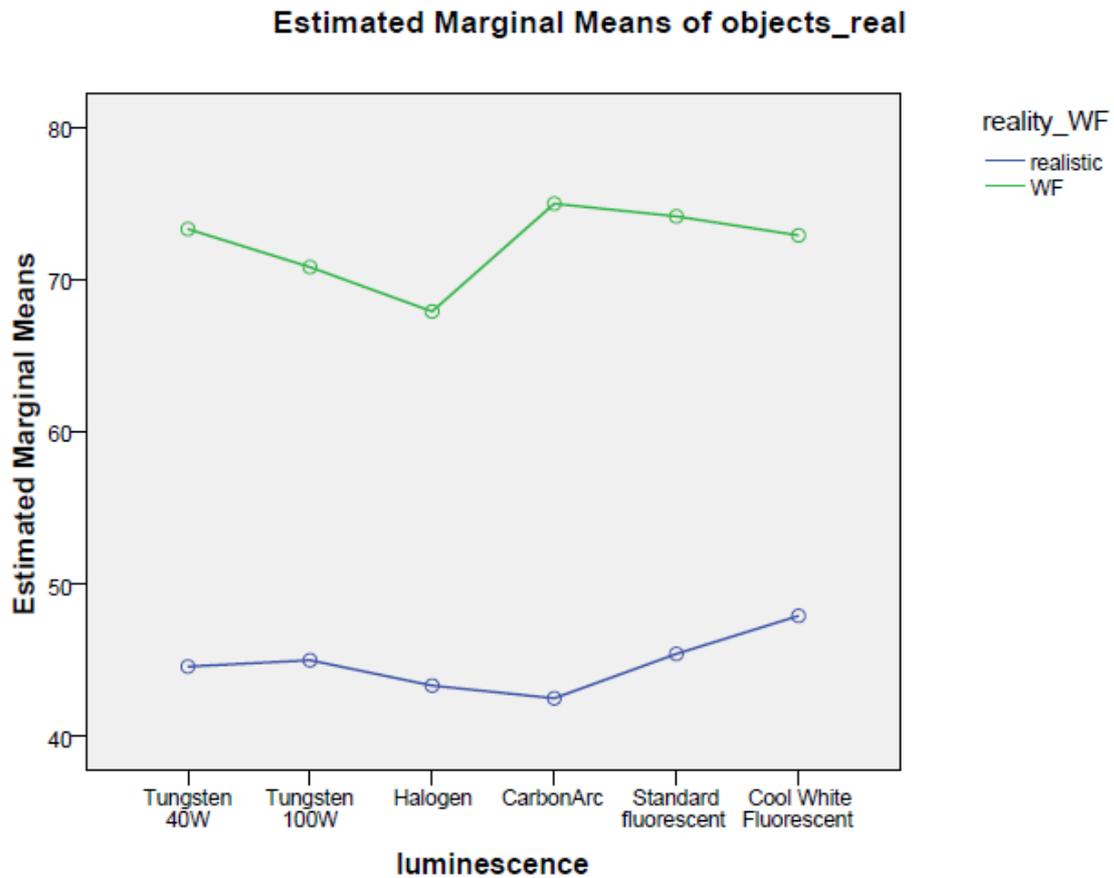


Figure 95: Estimated Marginal Means for the first question during the third stage of the experiment.

The Wilcoxon test conducted of the responses to the first question during the third stage of the experiment produced the statistics shown in Table 36. The results show that the participants felt that after exposure to the wireframe evening indoors 3D scene the objects were significantly more unreal than when viewing the photorealistically-rendered scene under all the different lighting conditions applied. This result was similar during the first stage of the experiments (Table 6).

Test Statistics^b

	Tungsten_40W_WF - Tungsten_40W	Tungsten_100W_WF - Tungsten_100W	Halogen_WF - Halogen	CarbonArc_WF - CarbonArc
Z	-2.943 ^a	-2.458 ^a	-2.703 ^a	-2.949 ^a
Asymp. Sig. (2-tailed)	.003	.014	.007	.003

a. Based on negative ranks.

b. Wilcoxon Signed Ranks Test

Test Statistics^b

	Standard_Fluorescent_WF - Standard_Fluorescent	Cool_White_Fluorescent_WF - Cool_White_Fluorescent
Z	-2.716 ^a	-2.536 ^a
Asymp. Sig. (2-tailed)	.007	.011

a. Based on negative ranks.

b. Wilcoxon Signed Ranks Test

Table 36: Wilcoxon test statistics for the first question during the third stage of the experiment.

In order to determine whether there was a significant difference of participants' responses to the first question (Figure 80), when viewing different artificial light types, a non-parametric Friedman test was performed on the data, as an alternative to repeated measures GLM. The test was conducted separately for responses after exposure to the photorealistically-rendered 3D scene and the wireframe one. The descriptive statistics of the test, including the mean, minimum and maximum values and the standard deviation can be seen in Table 37, as well as the Friedman test results. As the results show, the luminescence effect did not influence the evaluation of objects as more or less real when viewing the photorealistically-rendered 3D scenes.

NPar Tests K Related Samples; alternative to repeated measures GLM

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Tungsten_40W	12	44.58	19.360	20	85
Tungsten_100W	12	45.00	15.226	15	65
Halogen	12	43.33	15.126	25	75
CarbonArc	12	42.50	17.645	15	75
Standard_Fluorescent	12	45.42	22.101	15	85
Cool_White_Fluorescent	12	47.92	19.477	20	80

Friedman Test

Test Statistics^a

N	12
Chi-Square	4.828
df	5
Asymp. Sig.	.437

a. Friedman Test

Table 37: Descriptive statistics and Friedman test results for the first question after exposure to the photorealistic synthetic scene during the third stage of the experiment.

The Friedman test was conducted of the participants' responses to the first question after exposure to the wireframe 3D scene during the third stage of the experiment when viewed under each light type applied. The corresponding descriptive statistics, with mean, minimum and maximum values, as well as the standard deviation are shown in Table 38, along with the Friedman test results. Similarly to the results for participants' responses after exposure to the photorealistically-rendered scene, participants' responses after exposure to the wireframe scene do not reveal any luminescence effect influencing the evaluation of objects as more or less real.

NPar Tests; K related samples

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Tungsten_40W_WF	12	73.33	18.257	35	100
Tungsten_100W_WF	12	70.83	18.688	30	100
Halogen_WF	12	67.92	21.262	30	100
CarbonArc_WF	12	75.00	17.838	50	100
Standard_Fluorescent_WF	12	74.17	13.286	60	100
Cool_White_Fluorescent_WF	12	72.92	15.733	40	100

Friedman Test

Test Statistics^a

N	12
Chi-Square	2.149
df	5
Asymp. Sig.	.828

a. Friedman Test

Table 38: Descriptive statistics and Friedman test results for the first question after exposure to the wireframe synthetic scene during the third stage of the experiment.

The repeated-measures GLM test conducted of the participant’s responses to the third question during the third stage of the experiment involved two factors; the “reality” factor, of two levels, i.e. realistic and wireframe and the “luminescence” factor of six levels, i.e. the 40 Watt Incandescent Tungsten, the 100 Watt Incandescent the Tungsten, the Halogen, the Carbon Arc, the Standard Fluorescent and the Cool White Fluorescent light types respectively. The descriptive statistics are shown in Table 39.

Descriptive Statistics

	Mean	Std. Deviation	N
Tungsten_40W	62.27	14.725	11
Tungsten_40W_WF	50.45	15.565	11
Tungsten_100W	62.27	14.206	11
Tungsten_100W_WF	45.00	13.601	11
Halogen	62.73	13.297	11
Halogen_WF	43.18	15.696	11
CarbonArc	64.55	10.357	11
CarbonArc_WF	39.55	16.040	11
Standard_Fluorescent	55.45	16.801	11
Standard_Fluorescent_WF	38.64	17.043	11
Cool_White_Fluorescent	56.36	18.040	11
Cool_White_Fluorescent_WF	46.36	14.678	11

Table 39: Descriptive statistics for the GLM test for the third question during the third stage of the experiment.

The estimated marginal means that were plotted from the GLM test are shown in Figure 96.

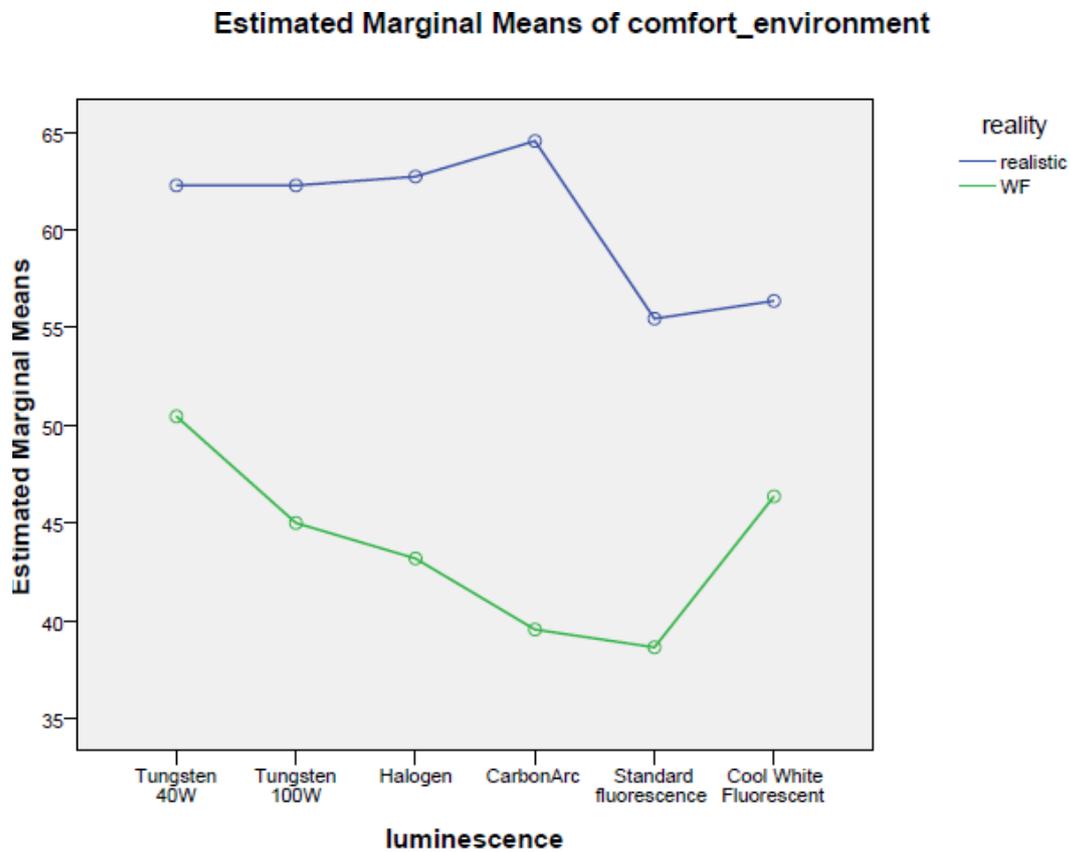


Figure 96: Estimated Marginal Means for the third question during the third stage of the experiment.

The Wilcoxon test performed on the data corresponding to participants' responses to the third question during the third experimental stage led to the results described in Table 40. The results suggest that participants felt the displayed environment significantly more comfortable after exposure to the photorealistically-rendered scenes, rather than after exposure to the wireframe scenes, under all the lighting conditions applied, except from the cool white fluorescent light type. Interestingly, when viewing the scene under the cool white fluorescent light type, there is no difference of participants' responses after exposure to the photorealistically-rendered and the wireframe scene when rating the environment comfort. In other words, after exposure to the photorealistically-rendered scene lit by the cool white fluorescent light type applied, the participants felt the environment to be less comfortable.

NPar Tests; 2 related samples

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Tungsten_40W	11	62.27	14.725	35	80
Tungsten_100W	11	62.27	14.206	35	80
Halogen	11	62.73	13.297	35	80
CarbonArc	11	64.55	10.357	50	85
Standard_Fluorescent	11	55.45	16.801	25	80
Cool_White_Fluorescent	11	56.36	18.040	20	85
Tungsten_40W_WF	11	50.45	15.565	25	70
Tungsten_100W_WF	11	45.00	13.601	25	65
Halogen_WF	11	43.18	15.696	20	65
CarbonArc_WF	11	39.55	16.040	15	60
Standard_Fluorescent_WF	11	38.64	17.043	15	65
Cool_White_Fluorescent_WF	11	46.36	14.678	25	70

Test Statistics^b

	Tungsten_40W_WF - Tungsten_40W	Tungsten_100W_WF - Tungsten_100W	Halogen_WF - Halogen	CarbonArc_WF - CarbonArc
Z	-2.512 ^a	-2.354 ^a	-2.375 ^a	-2.941 ^a
Asymp. Sig. (2-tailed)	.012	.019	.018	.003

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Test Statistics^b

	Standard_Fluorescent_WF - Standard_Fluorescent	Cool_White_Fluorescent_WF - Cool_White_Fluorescent
Z	-2.229 ^a	-1.613 ^a
Asymp. Sig. (2-tailed)	.026	.107

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 40: Descriptive statistics and Wilcoxon test results for the third question during the third stage of the experiment.

In order to determine whether there was any difference of participants' responses to the third question during the third stage of the experiment between light types applied to the photorealistically-rendered scenes, a non-parametric Friedman test was conducted. The results of this test are shown in Table 41. This test suggested that after exposure to the photorealistically-rendered scenes, the luminescence effect did not influence the participants' evaluation of the displayed environment's comfort. Descriptive statistics show that exposure to the photorealistically-rendered scenes lit by the standard fluorescent and the cool white fluorescent light types is rated as the least comfortable.

NPar Tests; K related samples

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Tungsten_40W	11	62.27	14.725	35	80
Tungsten_100W	11	62.27	14.206	35	80
Halogen	11	62.73	13.297	35	80
CarbonArc	11	64.55	10.357	50	85
Standard_Fluorescent	11	55.45	16.801	25	80
Cool_White_Fluorescent	11	56.36	18.040	20	85

Friedman Test

Test Statistics^a

N	11
Chi-Square	6.685
df	5
Asymp. Sig.	.245

a. Friedman Test

Table 41: Descriptive statistics and Friedman test results for the third question after exposure to the photorealistic synthetic scene during the third stage of the experiment.

A Friedman test was conducted of participants' responses during the third stage of the experiment corresponding to the third question after exposure to the wireframe scenes. The descriptive statistics and the test results are shown in Table 42. This test suggested that after exposure to the wireframe scene, the luminescence effect did influence the participants' evaluation of the displayed environment's comfort. Descriptive statistics show that when lighting wireframe scene with the standard fluorescent and the cool white fluorescent light types applied, the environment was rated as the least comfortable. Such mean variations should be explored more employing a higher number of experimental participants. Since there is not statistical significance, such effects are just to be observed but no clear conclusions could be drawn at this point.

NPar Tests; K related samples WF

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Tungsten_40W_WF	11	50.45	15.565	25	70
Tungsten_100W_WF	11	45.00	13.601	25	65
Halogen_WF	11	43.18	15.696	20	65
CarbonArc_WF	11	39.55	16.040	15	60
Standard_Fluorescent_WF	11	38.64	17.043	15	65
Cool_White_Fluorescent_WF	11	46.36	14.678	25	70

Friedman Test

Test Statistics^a

N	11
Chi-Square	15.924
df	5
Asymp. Sig.	.007

a. Friedman Test

Table 42: Descriptive statistics and Friedman test results for the third question after exposure to the wireframe synthetic scene during the third stage of the experiment.

The repeated-measures GLM test performed of the participants’ responses to the fourth question during the third stage of the experiment, after exposure to the photorealistically-rendered scene and the wireframe scene with each light type applied provided the descriptive statistics and the estimated marginal means shown in Table 43 and Figure 97 respectively. It is important to note that the mean differences between lighting conditions are not large.

Descriptive Statistics

	Mean	Std. Deviation	N
Tungsten_40W	55.00	21.213	11
Tungsten_40W_WF	48.64	24.504	11
Tungsten_100W	52.73	21.721	11
Tungsten_100W_WF	49.09	25.867	11
Halogen	54.55	20.549	11
Halogen_WF	50.00	26.173	11
CarbonArc	52.73	21.836	11
CarbonArc_WF	50.45	29.108	11
Standard_Fluorescent	53.18	22.054	11
Standard_Fluorescent_WF	52.27	22.953	11
Cool_White_Fluorescent	45.45	17.386	11
Cool_White_Fluorescent_WF	49.55	26.024	11

Table 43: Descriptive statistics for the fourth question during the third stage of the experiment.

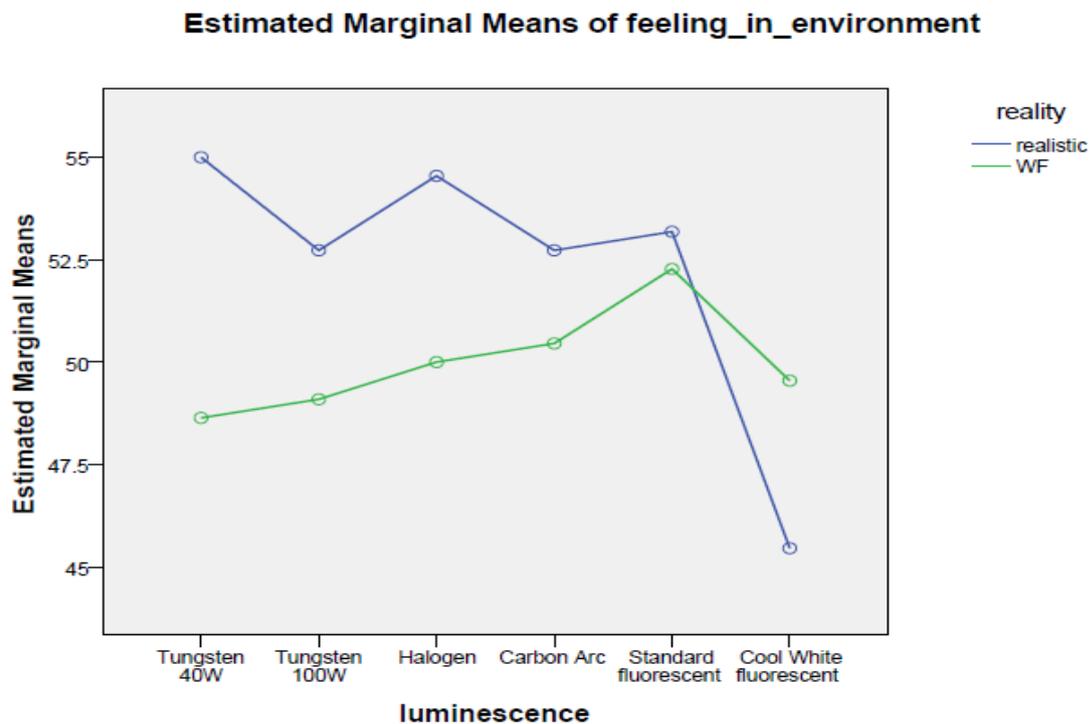


Figure 97: Estimated Marginal Means for the fourth question during the third stage of the experiment.

Results derived when applying the non-parametric tests did not reveal a statistical difference of participants’ responses after exposure to the photorealistically-rendered scene and the wireframe 3D scene both lit by different light types applied.

NPar Tests; 2 related samples

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Tungsten_40W	11	55.00	21.213	5	80
Tungsten_100W	11	52.73	21.721	0	75
Halogen	11	54.55	20.549	10	80
CarbonArc	11	52.73	21.836	0	75
Standard_Fluorescent	11	53.18	22.054	10	85
Cool_White_Fluorescent	11	45.45	17.386	5	65
Tungsten_40W_WF	11	48.64	24.504	10	80
Tungsten_100W_WF	11	49.09	25.867	15	85
Halogen_WF	11	50.00	26.173	5	85
CarbonArc_WF	11	50.45	29.108	0	100
Standard_Fluorescent_WF	11	52.27	22.953	15	90
Cool_White_Fluorescent_WF	11	49.55	26.024	10	80

Wilcoxon Signed Ranks Test

Test Statistics^c

	Tungsten_40W_WF - Tungsten_40W	Tungsten_100W_WF - Tungsten_100W	Halogen_WF - Halogen	CarbonArc_WF - CarbonArc
Z	-1.434 ^a	-.632 ^a	-.758 ^a	-.845 ^a
Asymp. Sig. (2-tailed)	.152	.527	.448	.398

a. Based on positive ranks.

c. Wilcoxon Signed Ranks Test

Test Statistics^c

	Standard_Fluorescent_WF - Standard_Fluorescent	Cool_White_Fluorescent_WF - Cool_White_Fluorescent
Z	-.051 ^a	-.358 ^b
Asymp. Sig. (2-tailed)	.959	.721

a. Based on positive ranks.

b. Based on negative ranks.

c. Wilcoxon Signed Ranks Test

Table 44: Descriptive statistics and Wilcoxon test results for the fourth question during the third stage of the experiment.

The Friedman non-parametric test was used to check for statistical differences in participants’ responses to the fourth question after exposure to the photorealistically-rendered scene when applying different light types.. The respective descriptive statistics and test results are shown in Table 45. This test suggested that after exposure to the

photorealistically-rendered scene, the luminescence effect did not influence participants’ feeling of presence in the displayed environment. Descriptive statistics show that the exposure to the cool white fluorescent light type resulted in the lower degree of presence in the VE.

NPar Tests; K related samples realistic

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Tungsten_40W	11	55.00	21.213	5	80
Tungsten_100W	11	52.73	21.721	0	75
Halogen	11	54.55	20.549	10	80
CarbonArc	11	52.73	21.836	0	75
Standard_Fluorescent	11	53.18	22.054	10	85
Cool_White_Fluorescent	11	45.45	17.386	5	65

Friedman Test

Test Statistics^a

N	11
Chi-Square	6.532
df	5
Asymp. Sig.	.258

a. Friedman Test

Table 45: Descriptive statistics and Friedman test results for the fourth question after exposure to the photorealistic synthetic scene during the third stage of the experiment.

The Friedman test applied to participants’ responses to the fourth question during the third stage of the experiment revealed similar results, as shown in Table 46. After exposure to the wireframe 3D scene, no luminescence effect influenced participants’ feeling of presence in the displayed environment.

NPar Tests; K related samples WF

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Tungsten_40W_WF	11	48.64	24.504	10	80
Tungsten_100W_WF	11	49.09	25.867	15	85
Halogen_WF	11	50.00	26.173	5	85
CarbonArc_WF	11	50.45	29.108	0	100
Standard_Fluorescent_WF	11	52.27	22.953	15	90
Cool_White_Fluorescent_WF	11	49.55	26.024	10	80

Friedman Test

Test Statistics^a

N	11
Chi-Square	.586
df	5
Asymp. Sig.	.989

a. Friedman Test

Table 46: Descriptive statistics and Friedman test results for the fourth question during the third stage of the experiment.

During the third stage of the experiment, while an artificial lighting configuration was presented, the participant was asked to change the brightness to their most comfortable value by adjusting a scale from low to high brightness, or vice versa (Figure 71). There were five available brightness values for the participants to choose from, i.e. 0.1, 0.3, 0.5, 0.7 and 0.9. Values 0.0 and 1.0 were not available, because it would be either completely dark in the synthetic scene or extremely bright respectively.

Table 47 shows participant’s responses for the most comfortable brightness value of the displayed artificial light type during the third stage of the experiment in the wireframe and the photorealistically-rendered synthetic scenes.

PN	40 Watt	40 Watt WF	100 Watt	100 Watt WF	Halogen	Halogen WF	Carbon Arc	Carbon Arc WF	Std Fluor	Std Fluor WF	Cool White Fluor	Cool White Fluor WF
1	0.900	0.700	0.500	0.500	0.900	0.500	0.900	0.500	0.100	0.300	0.900	0.700
2	0.900	0.900	0.700	0.500	0.400	0.500	0.100	0.500	0.300	0.300	0.300	0.300
3	0.100	0.500	0.300	0.500	0.100	0.500	0.100	0.100	0.100	0.300	0.100	0.300
4	0.300	0.700	0.700	0.700	0.500	0.409	0.700	0.300	0.300	0.300	0.300	0.500
5	0.533	0.500	0.300	0.500	0.300	0.300	0.100	0.100	0.500	0.463	0.300	0.460
6	0.900	0.300	0.300	0.900	0.100	0.500	0.500	0.300	0.100	0.700	0.500	0.900
7	0.300	0.100	0.100	0.100	0.100	0.100	0.100	0.100	0.100	0.100	0.100	0.100
8	0.500	0.500	0.300	0.300	0.300	0.300	0.300	0.300	0.300	0.300	0.300	0.300
9	0.500	0.700	0.700	0.900	0.700	0.900	0.700	0.700	0.700	0.700	0.500	0.500
10	0.500	0.300	0.300	0.300	0.300	0.300	0.100	0.300	0.900	0.463	0.300	0.460
11	0.900	0.517	0.700	0.500	0.500	0.500	0.500	0.367	0.500	0.700	0.500	0.700
12	0.100	0.100	0.500	0.300	0.500	0.100	0.100	0.500	0.300	0.700	0.100	0.300
13	0.500	0.900	0.100	0.500	0.500	0.409	0.350	0.700	0.350	0.700	0.350	0.460
Means	0.533	0.517	0.423	0.500	0.400	0.409	0.350	0.367	0.350	0.463	0.350	0.460

Table 47: Participants' responses for their most comfortable brightness value for each of the displayed artificial light types during the third stage of the experiment. WF stands for wireframe.

According to the responses, it can be argued that participants felt more comfortable with higher brightness values for the *Standard Fluorescent* and the *Cool White Fluorescent* light types during exposure to the wireframe scene, compared to the photorealistic scene.

A hierarchical cluster analysis based on the four questions of the questionnaire presented to the participants during the third experimental stage revealed 2 clusters (groups) of subjects, mainly separated by the ratings of questions two (“Do you yourself feel unreal?” - Figure 84) and one (“Do the things around you in the displayed environment feel unreal?” -- Figure 80)

These two clusters can be characterized as follows:

- **Cluster 1:** People who have lower scores for question 1: i.e. they see the environment as more “real” and higher scores for question 2: i.e. they see themselves as more “unreal”.
- **Cluster 2:** People who have higher scores for question 1: i.e. they see the environment as more “unreal” and lower scores for question 2: i.e. they see themselves as more “real”.

The existence of the two clusters which signifies specific patterns of responses shows that the participants’ answers to these questions were a product of thought and not of luck.

Table 48 displays the results of the statistical analyses conducted on the data acquired from participants’ responses during the first and the third experimental stages. Statistical analyses involved two types of non-parametric techniques, a Wilcoxon 2 related samples test and a Mann-Whitney test. The Wilcoxon test for 2 related samples’, which was an equivalent of a repeated measures GLM test was used on Z scores of participants’ responses to each question for each light type and each type of scene they were exposed to, e.g. photorealistically-rendered or wireframe, as dependent measures. Z scores were derived from the raw data normally distributed to satisfy the requirements of the statistical analysis. So, there were twelve measures, i.e. Z score Tungsten 100 Watt photorealistic, Z score Tungsten 100 Watt wireframe, Z score Tungsten 40 Watt photorealistic, Z score Tungsten 40 Watt wireframe, Z score Carbon Arc photorealistic, Z score Carbon Arc wireframe, Z score Cool White Fluorescent photorealistic, Z score Cool White Fluorescent wireframe, Z score Halogen photorealistic, Z score Halogen wireframe, Z score Standard Fluorescent photorealistic, Z score Standard Fluorescent wireframe.

The Mann-Whitney test for 2-independent groups was used as a comparison between the two clusters / groups of participants, with the composite scores of participants’ responses to the fourth question (Figure 88) during exposure to the synthetic scenes as dependent variables. The fourth question investigated participants’ perceived feeling of presence. The composite scores were the sum of the Z scores of participants’ responses to the fourth question for both the photorealistically-rendered and the wireframe synthetic scenes for each light type they were exposed to, e.g. the composite score for the Halogen light type would be: ‘sum of responses to the fourth question when exposed to the photorealistic Halogen light + equivalent when exposed to the wireframe Halogen’. So, there were six parameters, one for the composite score of each light type, i.e. Tungsten 100 Watt, Tungsten 40 Watt, Carbon Arc, Cool White Fluorescent, Halogen, and Standard Fluorescent. The results of the Mann-Whitney test are displayed in the last row of Table 48.

Results suggest that participants seem to have a higher feeling of presence in the VE during exposure to the photorealistically-rendered scenes lit by the 40 Watt Tungsten artificial light type compared to perceived presence when exposed to the wireframe scene lit by the 40 Watt Tungsten artificial light type. It is important to note that the two clusters of participants significantly differ in respect to their feeling of presence for the 100 Watt Tungsten artificial light type. During exposure to the synthetic scenes lit by the 100 Watt Tungsten light type independent of rendering fidelity condition, participants in cluster 1 have a higher perceived feeling of presence in the displayed environment compared to participants in cluster 2. Overall

from all results combined, the 40 Watt Tungsten and the 100 Watt Tungsten light types seem to reveal a difference in the feeling of presence, both in brain imaging analysis and in statistical analysis.

Behavioural results for free navigation (session 1) and adjusting the artificial light (session 3):

Session and contrasts	Unreality of environment	Unreality of self	"Comfort"	"Presence"
Session 1: Free navigation				
Realistic vs. WF indoor	*p(.001)/p(.002)/p(.008)/p(.001)	*NS/NS/p(.02)/p(.004)	*p(.002)/p(.003)/p(.001)/p(.001)	*p(.031)/p(.061)/p(.009)/p(.053)
Realistic vs. WF outdoor	**p(.001)/p(.02)/p(.002)	**NS/p(.005)/NS	**p(.001)/p(.001)/p(.003)	**p(.011)/p(.031)/p(.012)
Cluster 1 vs. Cluster 2 indoor	*NS/NS/NS/p(.045)	*p(.005)/p(.001)/p(.001)/p(.009)	*NS/NS/NS/NS	*NS/NS/NS/NS
Cluster 1 vs. Cluster 2 outdoor	**NS/NS/NS	**p(.009)/p(.001)/p(.001)	**NS/NS/NS	**NS/NS/NS
Session 3: Adjusting artificial light				
Realistic vs. WF	^NS/p(.002)/p(.004)/NS/p(.015)/p(.027)	^p(.053)/p(.015)/p(.003)/p(.047)/p(.011)/NS	^p(.036)/p(.001)/p(.001)/p(.015)/p(.005)/p(.004)	^NS/p(.004)/NS/NS/NS/NS
Cluster 1 vs. Cluster 2	^NS/p(.008)/p(.008)/p(.002)/p(.05)/NS	^p(.001)/p(.001)/p(.001)/p(.002)/p(.001)/p(.001)	^NS/NS/NS/NS/NS/NS	^p(.036)/NS/p(.066)/NS/NS/NS
Legend for Session 1: Free navigation (First experimental stage) *=morning / midday / afternoon / evening **= morning / midday / afternoon Legend for Session 3: Adjusting artificial light (Third experimental stage) Tungsten 100W/ Tungsten 40W/ Carbon-White/ Cool White Fluorescent/ Halogen/ Standard Fluorescent p=p value for asymptotic significance 2-tailed NS=non-significant				

Table 48: Behavioural results for the first and the third experimental stages.

6.4.5 Third Stage Brain Imaging Results

In relation to the brain imaging analysis, the statistical model that was used was a GLM (Generalized Linear Model). The factors that were used were the type of the displayed artificial light, i.e. Tungsten 100 Watt, Tungsten 40 Watt, Carbon Arc, Cool White Fluorescent, Halogen, and Standard Fluorescent), the “reality” level, i.e. exposure to either the photorealistically-rendered synthetic scene or the wireframe scene and “cluster membership”, either group 1 or group 2, between the groups that were formed through cluster analysis.

The matrix created for the GLM test is shown in Figure 101 and it includes six light types (40 Watt Tungsten, 100 Watt Tungsten, Halogen, Carbon Arc, Standard Fluorescent, Cool White Fluorescent) x 2 reality levels (photorealistic vs. wireframe) x 2 clusters (cluster 1 vs. cluster 2). A brain imaging analysis, which differentiates between groups and “reality” conditions, was based on this GLM test, creating a contrast between the 100 Watt Tungsten light type and the rest of the light types the participants were exposed to. This means that brain imaging results derived from the 100 Watt Tungsten lighting type were significantly different compared to brain imaging derived from the other light types in relation to brain areas activated during exposure. Figure 98 shows the following:

- GLM= full factorial; the 6 x 2 x 2 matrix; it is basically an ANOVA model.
- On the diagonal there are the cells of the matrix: 1-> 24.
- On the right columns are named like: con_0014.img, etc. These are processed images based on data collection. Basically con_0014.img is the image when the subjects were rating the questions under Tungsten 100 realistic. Con_0008.img when they were rating Tungsten 100 WF, etc. These images are the variables of interest which express brain activation under these conditions, i.e. Tungsten 100 realistic, Tungsten 100 WF, Tungsten 40 realistic, Tungsten 40 WF, etc. Also, cell 1 is for cluster 1 subjects, cell 2 for cluster 2 subjects, and so on.

A 6 x 2 x 2 design in a numerical form is described by 1 1 1 (Tungsten 100 realistic cluster 1), 1 1 2 (Tungsten 100 realistic cluster 2), 1 2 1 (Tungsten WF 100 realistic cluster 1), 1 2 2 (Tungsten WF 100 realistic cluster 2) etc and so on similarly for the six light types until we get twenty-four cells.

GLM in Imaging is conducted by Using a matrix notation we can write any models like

$$Y = X\beta + e$$

Y a vector for each data point, X the design matrix where each column is a vector representing groups/conditions/continuous predictor, β a vector (length = nb of column of X) of the coefficients to apply on X in order to minimize e the error (what is not modeled/explained). Matrix algebra offers a unique solution for all models:

$$\beta = (X^T X)^{-1} X^T Y$$

→ using pseudoinverse in matlab: $\text{betas} = \text{pinv}(X^T X) * X^T * Y$.

For each brain image acquired after exposure to the 100 Watt Tungsten, the goal is to define the appropriate β (weights) which would signify specific differences between the brain images

acquired after exposure to the 100 Watt Tungsten light type when compared to the equivalent weights for all other types. Therefore, to see the activations we need to put “weights” on betas to select specific comparisons in brain activity. For our first model, the matrix with 24 cells are shown in Figure 99.

The SPM 8.0 software is used to select the contrasts of interest (Figure 99). The contrast for the Figure 102 and Figure 103 is signified by the weights -5 5 -5 5 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1 1 -1. The weights numbered by -5 or 5 represent the weights for the Tungsten 100 lighting type in photorealistic or wireframe mode respectively. The 1s represent all other types in both viewing conditions. These weights are shown in Figure 100. The activated regions will be determined by this contrast. These would be the regions to make a difference under this contrast. We see this as a sort of difference. If you “contrast” the brain activity between cluster 1 and cluster 2, for realistic vs WF conditions under Tungsten 100, with all the other lights seen as cluster 1 – cluster 2, realistic – WF you will get the regions shown in Figure 102 and Figure 103. But it would take effort to detail this contrast however, the main outcome out of this analysis is that brain activity when people are rating realistic vs. WF under Tungsten 100 is different than brain activity for realistic vs. WF for all the other lights and the regions who make the difference are those in Figure 102 and Figure 103.

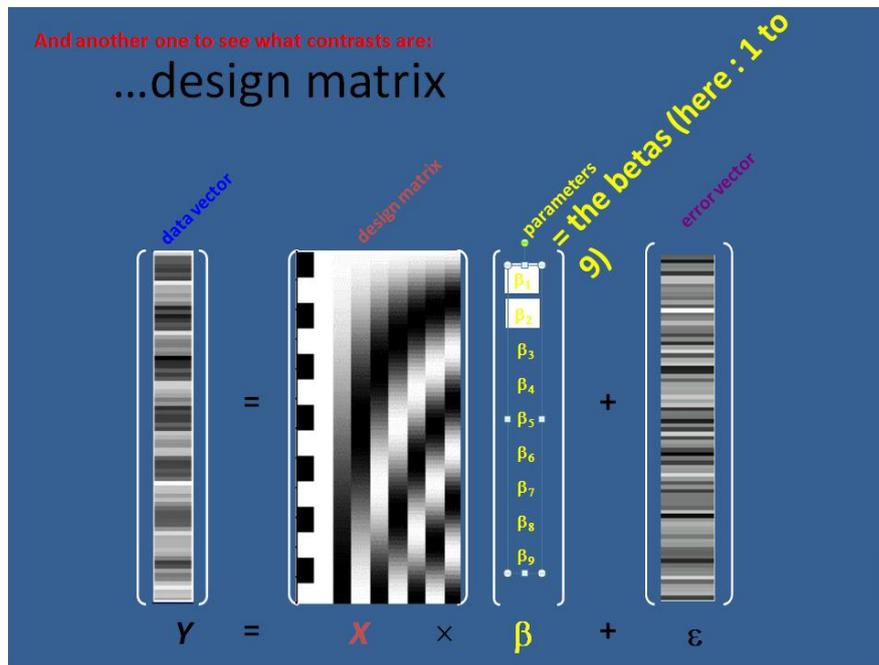


Figure 98: Design Matrix for Brain Imaging.

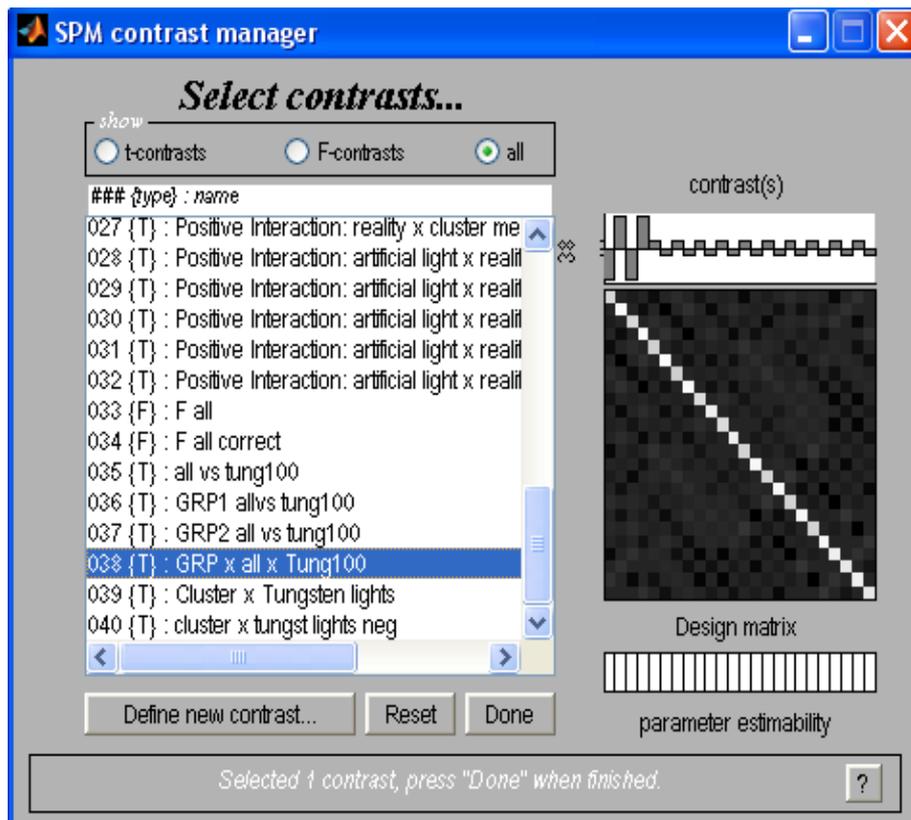


Figure 99: SPM software, selection of contrast.

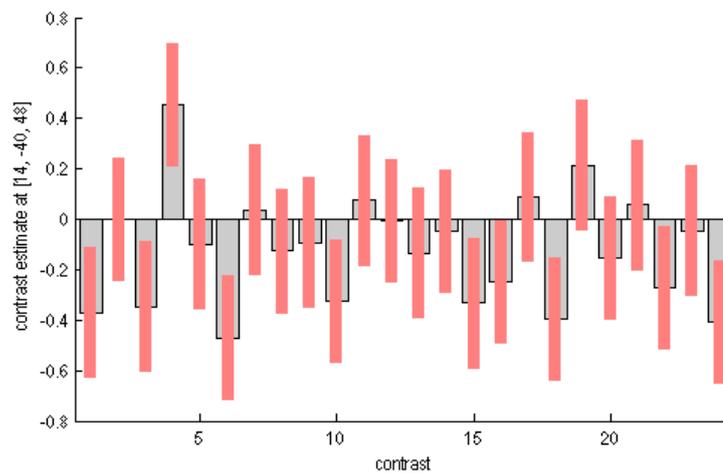
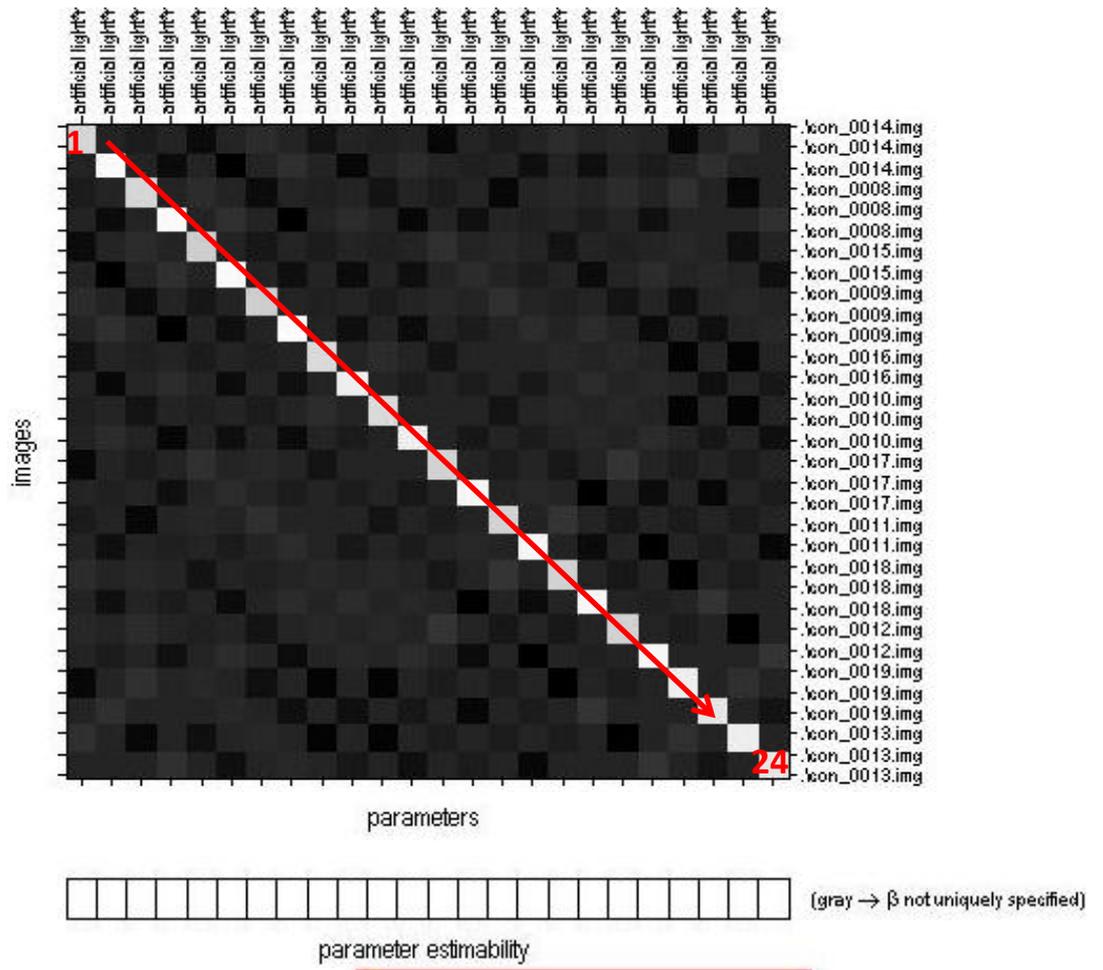


Figure 100: Contrast weights.



Design description...

- Design** : Full factorial
- Global calculation** : omit
- Grand mean scaling** : <no grand Mean scaling>
- Global normalisation** : <no global normalisation>
- Parameters** : 24 condition, +0 covariate, +0 block, +0 nuisance
24 total, having 24 degrees of freedom
leaving 132 degrees of freedom from 156 images

Figure 101: The 6x2x2 matrix used in the GLM test.

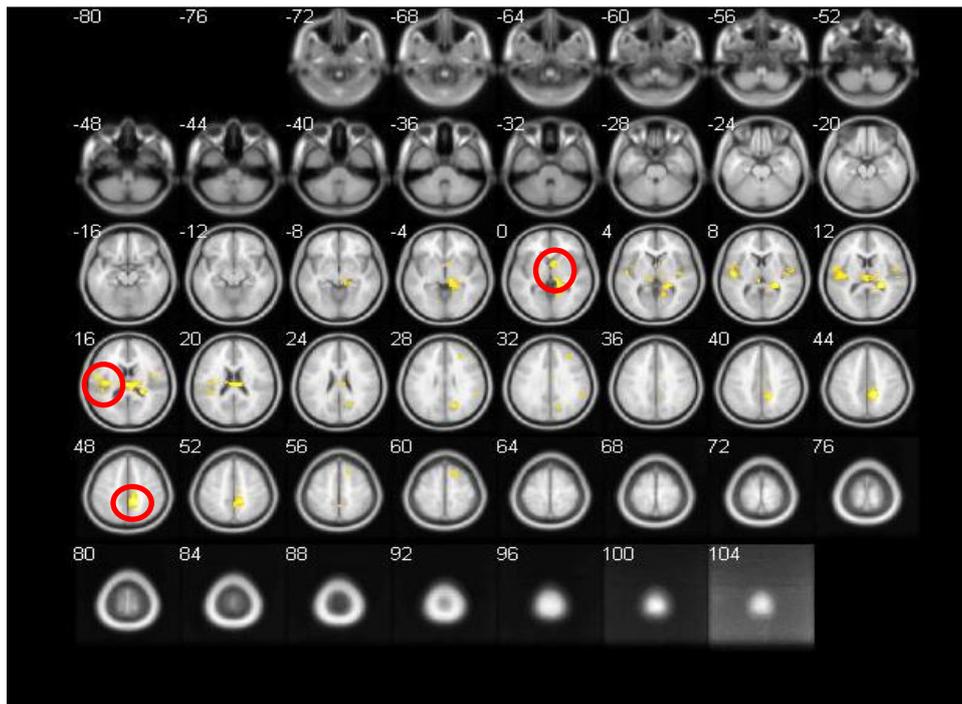


Figure 102: Brain Imaging Analysis results for the third experimental stage. This is an overview of the contrast.

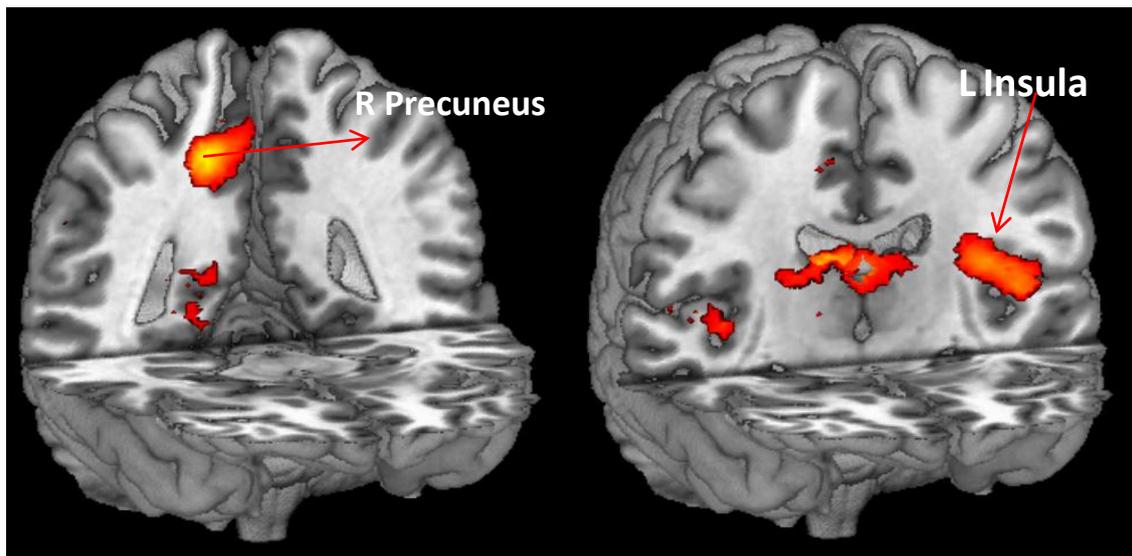


Figure 103: Brain Imaging Analysis results for the third experimental stage. The brain regions are highlighted.

The results under this contrast are significant in the regions of *R Precuneus*, *L Insula*, and *R Caudate*, which were previously implicated in tasks of investigating the sense of self. The results are displayed in Figure 102 and Figure 103.

A second GLM test was planned, with two factors, the “reality” factor, which included two levels, i.e. exposure to the photorealistically-rendered or the wireframe synthetic scene and the “light” factor signifying the artificial light types that lit the scene, e.g. 40 Watt Tungsten, 100 Watt Tungsten, Halogen, Carbon Arc, Standard Fluorescent and Cool White Fluorescent. The Z scores of participants’ responses to the fourth question (Figure 88) presented to them during the third experimental stage which investigated participants’ perceived feeling of presence, were introduced as covariates and they interacted with the artificial light type that lit the scene.

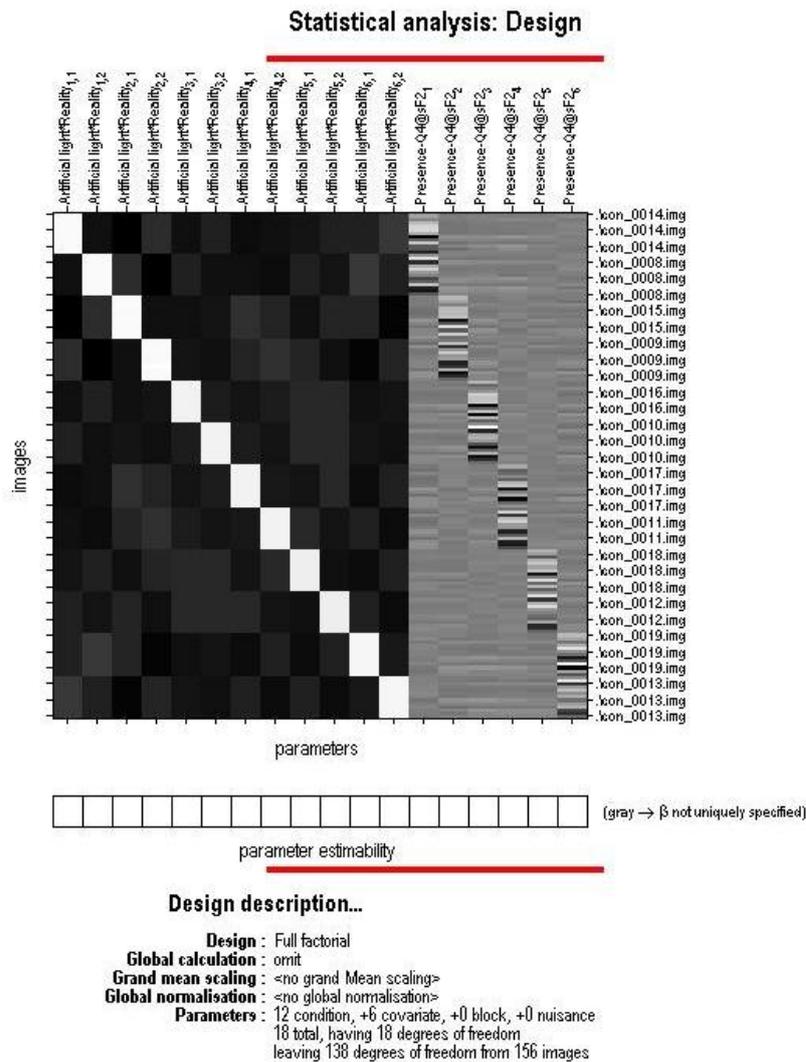


Figure 104: The 6x2 matrix used in the GLM test. The two factors are the "reality" (photorealistic vs. wireframe) and the available light types (6 light types). The Z scores of participants’ responses to the presence question were used as a covariate and they are shown in the last six columns (in lighter gray).

Figure 104 shows the 6 (artificial light types) x 2 reality levels (photorealistic vs. wireframe) matrix used. It is like an ANCOVA model, testing for differences in slopes of correlations across the subjects when they are exposed to different lights. This is almost the same with the first GLM, but we do not have cluster differences, therefore it is a 6 x 2 matrix, of 12 cells and we have a covariate, the last six columns in lighter gray (Figure 104). The covariate is the Z score from question 4- presence, and it interacts with the type of light. It is

like an ANCOVA model, for which we test for differences in slopes of correlations across the participants when they are exposed to different lights. In other words, what are the regions with strongest correlations with the self-reported score of presence, when the subjects are exposed to Tungsten 100 realistic, Tungsten 100 WF etc? This test was designed in order to find the regions with strongest correlations with the Z scores of participants' responses to the fourth question – investigating perceived feeling of presence – when the subjects are exposed to a specific quality version of the scene lit by a specific artificial light. Therefore, the interesting issue is to identify whether different responses to the question signified different brain activation patterns.

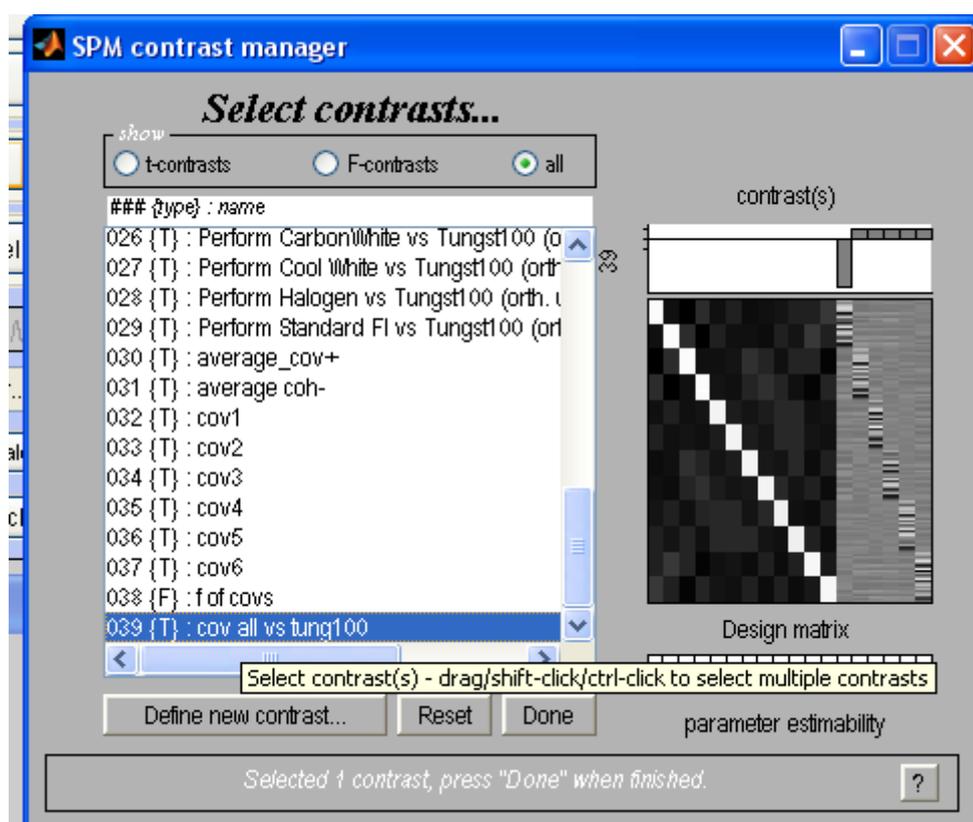


Figure 105: Contrast based on GLM.

A contrast was based on this GLM test, contrasting the brain images acquired during exposure to the 100 Watt Tungsten light type vs. the brain images acquired after exposure to the rest of the artificial light types. Now here is the contrast for this model:

0 0 0 0 ... 0 (up to 12) -5 1 1 1 1 1;

Why zeros? We are only interested in the interaction of covariate with the type of light and the effect of this interaction on brain activity. When we contrast the slope of correlation between brain activity and presence composite score for both photorealistic and wireframe under Tungsten 100 signified with -5, in both realistic and WF conditions, vs. all the other lights also realistic and WF, we get the regions in Figure 106 . Therefore, the interesting issue here is to

explore whether the Tungsten 100 light type, independent of viewing condition affects the rating of presence both in imaging and in self-report.

The results are displayed in Figure 106. The results show that similar brain regions to the ones appearing in the previous contrast which were previously implicated in tasks of investigating the sense of self (Figure 102 and Figure 103) are correlated with the Z scores of participants' responses to the fourth question, which probed for feelings of presence.

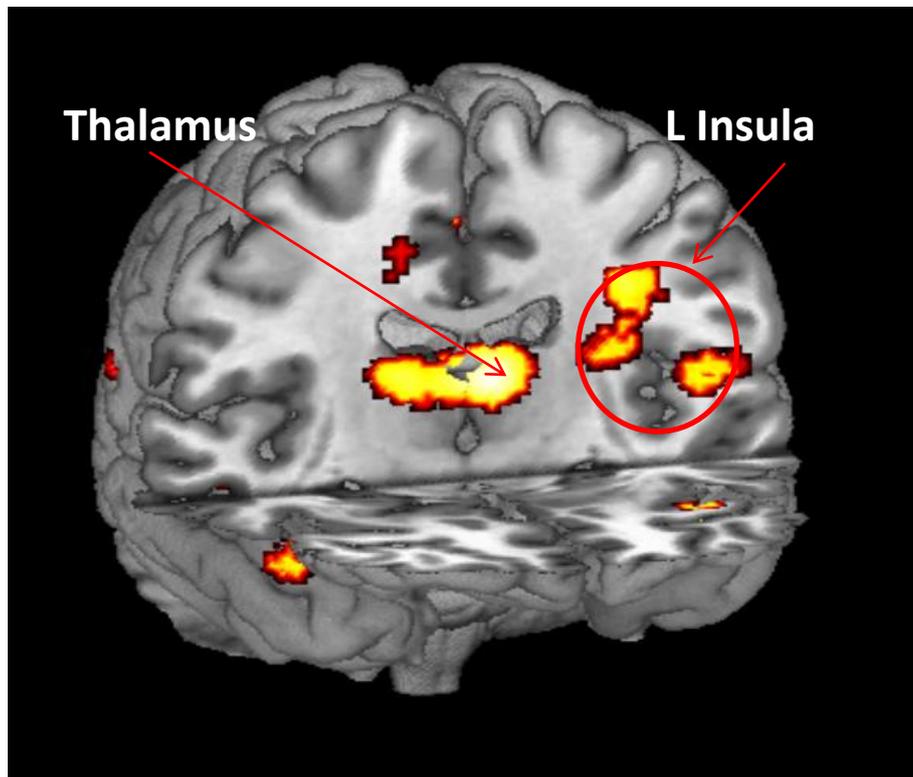


Figure 106: Brain Imaging Analysis Results. For this analysis, the scores of participants' responses to the fourth question presented to them during the third experimental stage, which investigated participants' perceived feeling of presence, were introduced as a covariate interacting with the light type applied in the synthetic scene.

6.5 Comments by Experiment Participants

When the experiment was completed and the participants were assisted out of the fMRI scanner, a short session took place inquiring the participants' general comments and advice in relation to the experimental protocol they followed. This session was aimed at getting a better insight of the rationale behind the participants' responses to the questions presented to them during the experiment. In this section, the most interesting participants' comments will be presented.

The majority of the participants complained about the *very loud noise* produced by the fMRI scanner, while it was acquiring the brain images. Although the participants were equipped with fMRI compliant headphones which were designed to reduce the noise the noise was still noticeable. This noise, according to the participants, was very distracting and disrupted their feeling of presence in the VEs. One participant especially reported that due to

this noise he couldn't have any feeling of presence in any of the VEs at all, no matter if they were photorealistically-rendered or wireframe.

The fMRI's surface, on which the participants were lying during the experiments, was very narrow and uncomfortable. In addition, the coil put on the participants' heads, in order to prevent possible head movement that would destroy the brain images acquired by the fMRI scanner, made them feel uncomfortable. The experiment lasted for approximately one hour and some participants reported that, due to these factors it was strange to be required to answer about how comfortable a VE feels.

Due to the technical difficulties encountered in relation the button boxes, as described in detail in section 4.4, many participants felt that the navigation in the synthetic scenes was unnatural and awkward and there were instances that they couldn't navigate or look at a specific place. Some participants concluded that a joystick would have been a better response pad to use.

Some participants reported that the second question (Figure 84) presented to them during the first and the third experimental stage was quite dubious and unclear. This was expected since this specific question was exploring the derealization of participants suffering from DPD (Depersonalization Disorder) and it was included in order to be a part of a future work with patients-participants and not healthy population.

It is important to note that some participants mentioned that the animated tree placed in the yard added an extra level of realism in the scene. Its animation made them feel more present in the VEs.

6.6 Discussion

It was shown that the participants felt that the objects populating all the wireframe scenes were perceived significantly more unreal than those in the photorealistically-rendered scenes.. These results were expected since it was natural for participants to feel the wireframe objects more unreal than their photorealistically-rendered counterparts.

Participants felt significantly more depersonalized while navigating the evening and afternoon indoors wireframe scenes, than the respective photorealistically-rendered ones, as well as after exposure to the morning outdoors wireframe scene, than after exposure to the equivalent photorealistic one. There was also a significant trend for participants to feel more depersonalized after exposure to the afternoon outdoors wireframe scene than after exposure to the respective photorealistically-rendered scene.

The results indicated that participants felt the environment to be more comfortable after exposure to photorealistically-rendered scenes, especially during midday, afternoon and evening. Interestingly, there were no statistically significant differences in relation to perceived comfort after exposure to the morning photorealistically-rendered and wireframe indoors scenes. Participants' responses after exposure to the outdoors scenes suggested that participants felt the photorealistically-rendered scenes to be more comfortable than the wireframe scenes, including the morning scenes. It could be argued that either the small sample size could be the reason for this difference, or that the lighting effects from the

simulated sun light in the morning indoors wireframe scene could trigger the overall feeling of comfort, despite the environment being unrealistic.

It was revealed that participants had a significantly higher feeling of presence after exposure to the photorealistically-rendered scenes, rather than after exposure to the wireframe scenes, particularly during midday and afternoon. However, this difference did not occur in the participants' responses to the fourth question (Figure 88), which was related to assessing the 'sense of presence', after exposure to the photorealistically-rendered outdoors 3D scenes and their respective wireframe version. These results were further reinforced by the brain images data analysis since the main preliminary result is that brain activation was influenced by the fidelity-quality of the displayed environment, between the photorealistically-rendered and the wireframe scenes, only after exposure to the indoors virtual scenes.

It is still an open discussion explaining the fact that the photorealistically-rendered outdoors scenes did not generate a greater feeling of presence in relation to the wireframe outdoors scenes, unlike the photorealistically-rendered indoors scenes in comparison to the wireframe indoors scenes. More participants could help strengthen these results statistically potentially revealing the reasons that led to these results. However, it could be argued that the simulated sun's lighting effects applied to the wireframe outdoors scenes were offering enough cues to participants in order to perceive the same level of feeling of presence as when exposed to the photorealistically-rendered scenes.

In relation to the results derived from the participants' ratings of emotional images during the second experimental stage, the results suggested that after exposure to the indoors scenes, pleasant pictures were significantly rated as more pleasant during midday compared with morning, neutral pictures were seen as more "pleasant" during exposure to the evening indoors scene compared to midday indoors scene and unpleasant pictures were seen as less unpleasant during morning compared to evening. Moreover, it was shown that the unpleasant pictures were seen as significantly less "unpleasant" during the exposure to the afternoon outdoors scene, rather than during the exposure to the midday outdoors scene. These results were most likely produced by the lighting effects that were visible during the exposure to the photorealistically-rendered scenes, according to the time of day it was simulated.

The results derived from participants' responses during third stage of the experiment showed that the participants felt that after exposure to the wireframe evening indoors scene the objects were significantly more unreal than in the photorealistically-rendered scene, for all the different lighting conditions applied. Further examination showed that the luminescence effect did not influence the evaluation of objects as more or less real in either the photorealistically-rendered scenes or the wireframe scenes. It could be argued that these results derived from the difference in fidelity-quality, rather than due to the various artificial lighting effects. An important note was that after exposure to the wireframe scene with the Halogen light type applied the participants reported that the objects were perceived as the least "unreal".

The displayed environment was perceived to be significantly more comfortable after exposure to the photorealistically-rendered scenes, rather than after exposure to the wireframe scenes, for all the lighting conditions applied, except from the cool white

fluorescent light type. Interestingly, for the cool white fluorescent light type, there was no difference in participants' responses after exposure to the photorealistically-rendered and the wireframe scene in rating the environment comfort.

In other words, after exposure to the photorealistically-rendered scene, when the cool white fluorescent light type was applied, the participants felt the environment to be less comfortable. It could be assumed that the lighting effects produced from the cool white fluorescent artificial light type led the participants to feel uncomfortable in both the photorealistically-rendered and the wireframe scenes, exceeding the uncomfortable feeling produced from the low fidelity quality wireframe scenes.

Furthermore, it was shown that during exposure to the photorealistically-rendered scenes, the luminescence effect did not influence the participants' evaluation of the displayed environment's comfort. Although, the exposure to the photorealistically-rendered scenes when lit by the standard fluorescent and the cool white fluorescent light types applied the environment was rated as the least comfortable.

The results derived from participants' responses during exposure to the wireframe scene during the third stage of the experiment showed that the luminescence effect did influence the participants' evaluation of the displayed environment's comfort. The effect seemed to be triggered after exposure to the wireframe scene when lit by the standard fluorescent and the cool white fluorescent light types applied resulting in the environment was rated as the least comfortable.

The participants' feeling of presence was shown to not be influenced by the lighting effects produced from the various artificial light types during exposure to both the photorealistically-rendered and the wireframe scene. However, one fact that could be noted is that after exposure to the cool white fluorescent light type, the subjects had on average the lower rating of perceived presence in the VE.

7 Chapter 7 – Conclusions and Future Work

This thesis put forward the development of an interactive 3D application system to be projected inside an fMRI scanner. The application was developed in order to be used for fidelity experiments inside the fMRI scanner, exploring the effect of modification of light and emotional stimuli on the feeling of presence and well-being of the normal and ultimately patient populations when exposed to photorealistically-rendered or wireframe synthetic scenes. The experiments were aimed to explore the biological correlates of variations in presence and subjective feelings of “reality” sensing by creating photorealistically-rendered and wireframe VEs which could be projected into the fMRI scanner, enabling the monitoring of neural activation patterns in response to a range of manipulations of the VEs.

It was a challenge to develop an interactive lighting system to be displayed in fMRI displays due to the infrastructural and technical demands. fMRI experiments usually employ simple display material, for example using photographs, video clips or simple computerized stimuli. Using VEs in fMRI has the advantage that it is possible to involve participants in interactive animated environments which more realistically reflect social and emotional situations. VE technology has already successfully been applied to fMRI research, but only by a very small number of research groups.

The 3D lighting framework was developed in UDK, which enabled the development of interactively manipulated photorealistically-rendered synthetic scenes, as well as providing the necessary tools to overcome the technical difficulties of using the system in an fMRI display. The implementation of the 3D lighting framework was adjusted to address the challenges arising from the strict experimental protocol, including the creation of the synthetic scenes and the lighting configuration of the light types that were used in the experiments, e.g. the simulation of a sun at different times, as well as the simulation of artificial light types. In order to make the VEs interactive, ranging from navigation to interaction with the questionnaires, the 3D lighting framework responded to the fMRI-compliant button boxes, surpassing the challenge of handling user input.

The strict experimental protocol imposed time limits that the framework met, by specifying timers in order to generate events and corresponding actions, meanwhile logging every action taking place in the VEs in a log file, in order to be able to understand and analyze the data gathered from each experiment. During the third experimental stage, the 3D lighting framework provided the participants with the ability to manipulate the artificial light’s brightness in real-time, by altering the pre-computed shadow maps with the new brightness values. The UIs were implemented through the use of Flash applications embedded in the framework and displayed on top of the displayed synthetic scene.

7.1 Main contributions

The effectiveness of VEs has often been linked to the sense of presence reported by users of those VEs. *Presence* is defined as the subjective experience of being in one place or environment, even when one is physically situated in another. It is argued that VEs that generate a higher feeling of presence would result in transfer equivalent to real-world

situations, which would be extremely useful for VEs such as training simulators. However, there is always a trade-off between visual/interaction fidelity and computational complexity. The ultimate goal would be to find the perfect balance in generating a higher feeling of presence while keeping the VEs computationally cheap.

This thesis aimed on developing an interactive 3D lighting system for fidelity experiments inside an fMRI display. This innovative application allowed the researchers to explore the effects of visual fidelity on feelings of presence, depersonalization and impressions of lighting based on comfort. Feelings of presence were explored by responses to specific questions during the exposure to the synthetic scenes, as well as examining the brain activity and the heart rate during the navigation in the VEs. This study allowed the researchers to investigate the participants' neurocorrelates of fidelity at the same time as being immersed in the synthetic scenes, instead of self-report of fidelity or task performance, after the task has occurred.

The results derived from the participants' responses during the experiment indicated that the exposure to the photorealistically-rendered indoors scenes generated a significantly higher feeling of presence in the VEs, rather than the exposure to the wireframe indoors scenes. At the same time, the exposure to the photorealistically-rendered outdoors scenes did not induce significant differences to the participants' feeling of presence, compared to the exposure to the wireframe scenes, although the participants felt that the wireframe 3D objects placed in the wireframe scenes were significantly more unreal than their counterparts placed in the photorealistically-rendered scenes. This difference was arguably caused by the lighting effects that were visible in the wireframe scenes, generating a higher feeling of presence, despite the significantly more unreal 3D objects placed inside.

Participants felt significantly more depersonalized while navigating the evening and afternoon indoors wireframe scenes, than the respective photorealistically-rendered ones, as well as after exposure to the morning outdoors wireframe scene, than after exposure to the equivalent photorealistic one. There was also a significant trend for participants to feel more depersonalized after exposure to the afternoon outdoors wireframe scene than after exposure to the respective photorealistically-rendered scene.

The results indicated that participants felt the environment to be more comfortable after exposure to photorealistically-rendered scenes, especially during midday, afternoon and evening. Interestingly, there were no statistically significant differences in relation to perceived comfort after exposure to the morning photorealistically-rendered and wireframe indoors scenes. Participants' responses after exposure to the outdoors scenes suggested that participants felt the photorealistically-rendered scenes to be more comfortable than the wireframe scenes, including the morning scenes. It could be argued that either the small sample size could be the reason for this difference, or that the lighting effects from the simulated sun light in the morning indoors wireframe scene could trigger the overall feeling of comfort, despite the environment being unrealistic.

Preliminary results from the analysis on brain image data acquired during the first stage of the experiments show that brain activation was influenced by the fidelity-quality of the displayed environment, between the photorealistically-rendered and the wireframe scenes,

only after exposure to the indoors virtual scenes. There was a greater activation in regions previously implicated in “spatial navigation”, including navigation in VEs, after exposure to the indoors photorealistically-rendered scenes compared to the respective wireframe scenes. Examples of these regions, which are related to spatial navigation, are the superior frontal gyrus, the posterior cingulate and the cerebellum. This would suggest that in the realistic indoors virtual scenes, participants were practically “navigating” and exploring the environment. The same brain activity is not evoked by the wireframe synthetic scenes.

7.2 Implications for Future Work

The experiments described in detail in Chapter 6 were formally designed. However, certain improvements could be accomplished by the following actions:

- The experiments were designed to ultimately be performed on patient population, suffering from the Depersonalization Disorder (DPD). It would be extremely interesting to explore whether the VEs can increase or decrease the patients’ feelings of depersonalization or derealization. We expect that the system is now to be used on patient populations adjusted to satisfy specific ethical concerns imposed when running fMRI experiments involving patients
- It would be innovative and useful to test the feelings of presence and impressions of lighting in even more extreme or more subtle fidelity variations. This would provide the ability to compare the results and reinforce the conclusions.
- The loud noise produced by the fMRI scanner that according to participants could disrupt the feelings of presence should be handled with greater care. An efficient way would be to include more sounds emanating from the VEs that would distract participants’ attention of the noise.

8 References – Bibliography

- Ashdown 2004 Ian Ashdown, *Radiosity Bibliography* [www], <http://tralvex.com/pub/rover/abs-ian0.htm> (Accessed March 2004)
- Bailey and Witmer 1994 Bailey, J.H., Witmer, B.G. 1994. Learning and Transfer of Spatial Knowledge in a Virtual Environment. Proc. of the Human Factors & Ergonomics Society 38th Annual Meeting, 1158-1162, Santa Monica, CA: Human Factors & Ergonomics Society.
- Billinghamurst et al. 2002 Billinghamurst, M., Ellis, S.R., Mania, K., Steed, A. 2002. In Mania, K. (ed.) Usability Evaluation Techniques For Virtual Reality Technologies: Human Factors Issues, course notes, IEEE Virtual Reality 2002.
- Biocca et al. 2002 Biocca, F., Brooks, F.P. Jr., Ellis, S. R., Mania, K., Slater, M., Steed, A., Whitton, M. (2002). Understanding Virtual Environments: Immersion, Presence, and Performance. ACM Siggraph 2002 course notes, San Antonio, USA, full-day course
- Birn 2000 Birn, J. 2000. *[digital] Lighting and Rendering*. New Riders.
- Botella et al. 2007 Botella, C. García-Palacios, A., Villa, H., Baños, R. M., Quero, S., Alcañiz, M. and Riva, G. 2007. Virtual Reality Exposure in the Treatment of Panic Disorder and Agoraphobia: A Controlled Study. *Clinical Psychology and Psychotherapy Clin. Psychol. Psychother.* 14, 164–175 (2007). Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpp.524
- Bouknight 1970 Bouknight J. 1970. "A procedure for generation of three dimensional half-toned computer graphics presentations." *Communications of the ACM*, Vol. 13 (9) 527–536
- Boyce and Cuttle 1990 Boyce, P.R., Cuttle, C. 1990 "Effect of correlated colour temperature on the perception of interiors and colour discrimination performance", *Lighting Research and Technology*; 22; 19
- Brooks 1999 Brooks, F.P. Jr. 1999. What's Real About Virtual Reality. *IEEE Computer Graphics and Applications*, Nov.-Dec. 1999, 16-27.
- Carlin et al. 1997 Carlin, A.S., Hoffman, H.G., Weghorst, S. 1997. Virtual reality and tactile augmentation in the treatment of spider phobia: a case report, *Behaviour Research and Therapy*, Volume 35, Issue 2, February 1997, Pages 153-158, ISSN 0005-7967, 10.1016/S0005-7967(96)00085-X
- Chalmers and Cater 2002 Chalmers A.G, and Cater K. 2002 "Realistic Rendering in Real-Time." In proceedings of the 8th International Euro- Par Conference on Parallel Processing, Springer-Verlag 21-2
- Cohen and Greenberg 1985 Cohen M.F., and Greenberg D.P. 1985. "The Hemi-Cube: A radiosity solution for complex environments." In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19, pages 31-40.
- Cohen et al. 1988 Cohen M.F., Chen S.E., Wallace J.R., and Greenberg D.P. 1988. "A progressive refinement approach to fast radiosity image generation." In John Dill, editor, *Computer Graphics (SIGGRAPH 1988 Proceedings)*, volume 22, pages 75–84.
- Emmelkamp et al. 2002 P.M.G Emmelkamp, M Krijn, A.M Hulsbosch, S de Vries, M.J Schuemie, C.A.P.G van der Mast. 2002. Virtual reality treatment versus exposure in vivo: a comparative evaluation in acrophobia, *Behaviour Research and Therapy*, Volume 40, Issue 5, May 2002, Pages 509-516, ISSN 0005-7967, 10.1016/S0005-7967(01)00023-7.
- Ferwerda 2001 Ferwerda, J. 2001 Hi-fi rendering, *Campfire in Perceptually Adaptive*

References – Bibliography

- Graphics, 26 – 29 May 2001, <http://isg.cs.tcd.ie/campfire/jimferwerda2.html>
- Ferwerda 2003 Ferwerda, J.A. 2003. "Three varieties of realism in computer graphics." In IS&T/SPIE Conference on Human Vision and Electronic Imaging VIII, SPIE Proceedings Vol. 5007, 290-297
- Flynn et al. 1973 Flynn, J. E., Spencer, T. J., Martyniuk, O., Hendrick, C. 1973. "Interim study of procedures for investigating the effect of light on impression and behavior", *Journal of the IES, Oct. 1973*.
- Flynn 1977 Flynn, J. E. 1977. "A study of subjective responses to low energy and non-uniform lighting systems". *Lighting Design and Application, Feb. 1977*
- Flynn et al. 1979 Flynn, J. E., Hendrick, C., Spencer, T. J., Martyniuk, O. 1979. "A guide to methodology procedures for measuring subjective impressions in lighting". *Journal of the IES, Jan. 1979*
- Gibson and Hubbold 1997. Gibson, S., and Hubbold R.J. 1997. "Perceptually Driven Radiosity." *Computer Graphics Forum*, 16 (2): 129-140.
- Glassner 1984 Glassner, A.S. 1984. "Space Subdivision for Fast Ray Tracing", *IEEE Computer Graphics & Applications*, Vol.4, No. 10, pp 15-22.
- Glassner 2000 Glassner, A.S. 2000. *An Introduction to Ray tracing.*_Morgan Kaufmann.
- Goral et al. 1984 Goral C.M., Torrance K.E., Greenberg D.P., and Battaile B. 1984. "Modeling the interaction of light between diffuse surfaces." In proceedings of the *11th Annual Conference on Computer Graphics and Interactive Techniques*. Vol. 18 (3) 212-222
- Gouraud 1971 Gouraud H., "Continuous shading of curved surfaces", *IEEE Transactions on Computers*, 20(6): 623-628.
- Hall and Greenberg 1983 Hall R.A., and Greenberg D.P. 1983. "A Testbed for Realistic Image Synthesis." *IEEE Computer Graphics and Applications*, Vol. 3, No. 8. pp. 10-19.
- Hastings 2011 James Hastings-Trew. 2011. *Reproducing Real World Light* [www], <http://planetpixelemporium.com/tutorialpages/light.html> (Accessed February 2011)
- Hoffman et al. 2008 Hoffman, H.G., Patterson, D.R., Seibel, E., Soltani, M., Jewett-Leahy, L., Sharar, S.R. 2008. Virtual Reality Pain Control During Burn Wound Debridement in the Hydrotank. *Clinical Journal of Pain: May 2008 - Volume 24 - Issue 4 - pp 299-304 doi: 10.1097/AJP.0b013e318164d2cc*
- Houser et al. 2002 Houser, K.W., Tiller, D.K., Bernecker, C.A., Mistrick, R.G. 2002. "The subjective response to linear fluorescent direct/indirect lighting systems". *Lighting Research and Technology*; 34; 243
- Izso et al. 2009 Izso, L., Lang, E., Laufer, L., Suplicz, S., Horvath, Á. 2009. "Psychophysiological performance and subjective correlates of different lighting conditions". *Lighting Research and Technology* 2009; 41; 349
- Kajiya 1986 Kajiya, J.T. 1986. "The Rendering Equation." *ACM SIGGRAPH 1986 Conference Proceedings*, volume 20,143-150.
- Kajiya 1990 Kajiya, J.T. (1990). Radiometry and Photometry for Computer Graphics. *Proc. of ACM SIGGRAPH 1990 Advanced Topics in Ray Tracing course notes*.
- Kuller et al. 2006 Kuller, R., Ballal, S., Laike, T., Mikellides, B., Tonello, G. 2006. "The impact of light and colour on psychological mood: a cross-cultural study of indoor work environments". *Ergonomics*; 49: 14, 1496 – 1507
- Lang, Bradley & Cuthbert 2008 Lang, P.J., Bradley, M.M., & Cuthbert, B.N. (2008). International affective picture system (IAPS): Affective ratings of pictures and

References – Bibliography

- instruction manual. Technical Report A-8. University of Florida, Gainesville, FL.
- Languénou et al. 1992 Languénou E., Bouatouch K., and Tellier P. 1992. "An adaptive Discretization Method for Radiosity." *Computer Graphics Forum* 11(3): 205-216.
- Lathrop and Kaiser 2002 Lathrop, W.B. Kaiser, M.K. 2002. Perceived Orientation in Physical and Virtual Environments: Changes in Perceived Orientation as a Function of Idiothetic Information Available. Presence: Teleoperators and Virtual Environments,11(1), 19-32. MIT Press.
- Lischinski 2004 Lischinski, D., Radiosity, (Lecture notes) [www] <http://www.cs.huji.ac.il/~danix/advanced/notes3.pdf>
- Liu et al. 2005 Liu, G. Austen, E.L. Booth, K.S. Fisher, B.D. Rempel, M.I. Enns, J.T. 2005 Multiple object tracking is based on scene, not retinal, coordinates, *Journal of Experimental Psychology: Human Perception and Performance*, 31, 2, Apr. 2005, 235-247
- Lombard and Ditton 1997 Lombard, M., Ditton, T. 1997. At the Heart of it All: The concept of Presence. *JCMC* 3(2)
- Loomis 1992 Loomis, J.M. 1992. Presence and Distal Attribution: Phenomenology, Determinants and Assessment. SPIE Vol. 1666: Human Vision, Visual Processing and Digital Display III, 590-595.
- Lowry 2011 Lowry, Richard. *Concepts & Applications of Inferential Statistics* [www], <http://faculty.vassar.edu/lowry/ch12a.html> (Accessed March 2011).
- Mahnke 1996 Mahnke, F.H. 1996. "Color, Environment, & Human Response". USA: John Willey and Sons, Inc. 1996. ISBN: 0471-28667-2
- Maida et al. 1997 Maida, J, Aldridge, A and Novak, J. 1997. "Effects of Lighting on Human Performance in Training". *Design of Computing Systems: Social and Ergonomic Considerations*. Vol. 21B, pp. 877-880.
- Mania and Chalmers 2001 Mania, K. & Chalmers, A. 2001. The Effects of Levels of Immersion on Presence and Memory in Virtual Environments: A Reality Centred Approach. *Cyberpsychology & Behavior*, 4(2), 247-264
- Mania et al. 2003 Mania, K., Troscianko, T., Hawkes, R., Chalmers, A. 2003. Fidelity Metrics for Virtual Environment Simulations based on Human Judgments of Spatial Memory Awareness States. Presence, Teleoperators and Virtual Environments, 12(3), MIT Press, 296-310
- Mania and Robinson 2006 Mania, K., Robinson, A. 2006. "An experimental exploration of the relationship between subjective impressions of illumination and physical fidelity". *Computers & Graphics*. February, Vol. 29, 1.
- McCloughan et al. 1999 McCloughan, C.L.B., Aspinall, P.A., Webb, R.S. 1999. "The impact of lighting on mood". *Lighting Research and Technology*; 31; 81
- McNamara et al. 2000 McNamara, A., Chalmers, A., Troscianko, T., Gilchrist, I. (2000). Comparing Real and Synthetic Scenes using Human Judgements of Lightness. Proc. of EGWR 2000, 207-219.
- ME 2004 Molecular Expressions, *Physics of light And colour* [www], <http://micro.magnet.fsu.edu/primer/java/reflection/specular/>
- Medford et al. 2005 Medford N, Sierra M, Baker D, David AS. 2005. Understanding and treating depersonalisation disorder. *Advances in Psychiatric Treatment* 11, 92-100.
- Meehan 2001 Meehan, M. 2001. An Objective Surrogate for Presence: Physiological Response. 3rd International Conference on Presence, ISBN 9-386-1571-X.
- Meehan et al. 2002 Meehan, M., Insko, B., Whitton, M., Brooks, F.P.Jr. 2002. "Physiological measures of presence in stressful virtual environments". ACM

References – Bibliography

- Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2002
Volume 21 Issue 3, July 2002
- Meyer et al. 1986 Meyer, G.W., Rushmeier, H. E., Cohen, M.F., Greenberg, D.P. (1986). An Experimental Evaluation of Computer Graphics Imagery. *ACM Transactions on Graphics*, 5(1), pages 30-35.
- Mourkoussis et al. 2010 Mourkoussis, N., Rivera, F., Troscianko, T., Hawkes, R., Watten, P., Mania, K. 2010. Quantifying Fidelity for Virtual Environment Simulations Employing Memory Schema Assumptions. *ACM Transactions on Applied Perception*, ACM Press.
- NightLase 2004 NightLase, *Light and optics theories and principles* [www], <http://www.nightlase.com.au/education/optics/diffraction.htm>
- Nusselt 1928 Nusselt, W. 1928. "Grapische Bestimmung des Winkelverhältnisses bei der Warmestrahlung," *Zeitschrift des Vereines Deutscher Ingenieure* 72(20):673.
- Palmer 1999 Palmer, S.E. 1999. *Vision Science – Photons to Phenomenology*. Massachusetts Institute of Technology Press
- Phong 1975 Phong, B.T. 1975. "Illumination for Computer-Generated Images." *Communications of the ACM*, Vol 18 (6) 449— 455.
- Rothbaum et al. 1996 Barbara Olasov Rothbaum, Larry Hodges, Benjamin A. Watson, G.Drew Kessler, Dan Opdyke. 2006. Virtual reality exposure therapy in the treatment of fear of flying: a case report, *Behaviour Research and Therapy*, Volume 34, Issues 5-6, May-June 1996, Pages 477-481, ISSN 0005-7967, 10.1016/0005-7967(96)00007-1.
- Siegel and Howell 1992 Siegel R. and Howell J.R. 1992. *Thermal Radiation Heat Transfer*, 3rd Edition. Hemisphere Publishing Corporation, New York, NY.
- Sierra 2009 Sierra M. 2009. *Depersonalization: a new look at a neglected syndrome*. Cambridge University Press.
- Shloerb 1995 Shloerb, D.W. 1995. A Quantitative Measure of Telepresence. *Presence, Teleoperators and Virtual Environments*, 4(1), 64-80. MIT Press.
- Slater and Steed 1998 Slater, M., Steed, A., McCarthy, J., Maringelli, F. 1998. The Influence of Body Movement on Subjective Presence in Virtual Environments. *Human Factors*, 40(3), 469-477.
- Slater 2004 Slater, M. 2004. "How Colorful Was Your Day? Why Questionnaires Cannot Assess Presence in Virtual Environments". *Presence: Teleoperators and Virtual Environments*, August 2004, Vol. 13, No. 4 , Pages 484-493. (doi: 10.1162/1054746041944849)
- Stahre 2006 Stahre, B. 2006. *How to Convert Reality into Virtual Reality: Exploring Colour Appearance in Digital Models*". Goteborg: Chalmers University of Technology. ISBN/ISSN 1650-6340, 2006:03
- Stanney et al. 1998 Stanney, K.M., Salvendy, G., Deisigner, J., DiZio, P., Ellis, S., Ellison, E., Fogleman, G., Gallimore, J., Hettinger, L., Kennedy, R., Lackner, J., Lawson, B., Maida, J., Mead A., Mon-Williams, M., Newman, D., Piantanida, T., Reeves, L., Riedel, O., Singer, M., Stoffregen, T., Wann, J., Welch, R., Wilson, J., Witmer, B. 1998. Aftereffects and Sense of Presence in Virtual Environments: Formulation of a research and development agenda. Report sponsored by the Life Sciences Division at NASA Headquarters. *International Journal of Human-Computer Interaction*, 10(2), 135-187.
- Tarr et al. 1998 Tarr, M. J., Kersten, D., Bulthoff, H. 1998. "Why the visual recognition system might encode the effects of illumination". *Vision Research*. August, Vol. 38, 15-16, pp. 2259-2275
- Veitch et al. 2008 Veitch, J.A., Newsham, G.R., Boyce, P.R., Jones, C.C. 2008. *Lighting*

References – Bibliography

- appraisal, well-being and performance in open-plan offices: A linked mechanisms approach. *Lighting Research and Technology*; 40; 133
- Wagner 1987 Wagner, L. 1987 The effects of shadow quality on the perception of spatial relationships in computer generated imagery, in Proc. of Symposium on Interactive 3D Graphics, 1987, 39-42.
- Waller et al. 1998 Waller, D., Hunt, E., Knapp, D. 1998. The Transfer of Spatial Knowledge in Virtual Environment Training. Presence: Teleoperators and Virtual Environments, 7(2), MIT Press.
- Ward Larson and Shakespeare 1998 Ward Larson, G and Shakespeare, R. 1998. "Rendering with RADIANCE: The art and science of lighting simulation", San Francisco: Morgan Kauffman.
- Wilcoxon 1945 Wilcoxon, Frank. 1945. *Individual comparisons by ranking methods*. Biometrics Bulletin **1** (6): 80–83.
- Zimmons 2004 Zimmons, P.M. 2004. The Influence of Lighting Quality on Presence and Task Performance in Virtual Environments. Carolina: The University of North Carolina at Chapel Hill. UMI Order Number: AAI3140421
- 3D Model Repositories <http://archive3d.net/>
<http://www.3dmodelfree.com/>
<http://artist-3D.com/>
- Epic Games <http://www.epicgames.com/>
Model Textures <http://cgtextures.com/>
<http://archivetextures.net/>
- Ogre 3D <http://www.ogre3d.org/>
Unity 3D <http://unity3d.com/>
Unreal Development Kit <http://udk.com>