Drexel University's Gaming I/II

Presents:

"Your brain is your worst enemy and your only hope!"

Written by Hardik Bhatt, Arjun Gupta, Jeff Lorenz, Phillip Lyon,

Craig Stevenson and Peter Stratton

Version 4.0

Friday, March 20, 2009

INTRODUCTION

The purpose of this document is to give clear and complete explanation of what *Maxwell's Demon* is and what it hopes to accomplish. By reading this document in its entirety you should get a clear picture of what the game world looks like, what game play will be like, and how the player navigates the game world.

Additionally, you will have a better understanding of what data we hope to collect from the users as they play. Brain controlled devices in games are on the cutting edge in the game industry and little is known about how the brain works with these devices. Our hope is that by making a brain-controlled device an integral part of a fun and engaging environment, we will gain new information about how the brain processes different scenarios while in a gaming environment.

Finally, we hope that our work can be used as a foundation for future BCI game development. The meticulous construction of this document is to aid another development team in continuing where our work finished.

Design History

The progress that has been made on the overall design of this game is the result of a two-term project for the gaming group. Below is the current version history, summarizing the changes we have made throughout the life of this document. We are currently in v3.0 of this document.

Version 1.0

The initial game design document was created to layout principles and story elements that the group had come up with. Our initial focus was to design a game that was fun to play yet incorporated the use of a novel control setup. While we did incorporate the use of a brain device, our initial prototype build did not use the device as an integral component to game play and neglected the potential for data gathering and innovative research. For that reason, we have decided to rework our initial ideas and change our focus to incorporate meaningful research results.

Version 2.0

In an effort to incorporate a platform for data reception, the overall game had to be redesigned. Our new goal is to construct an environment where the player is presented with various tasks that can be slightly altered to pinpoint what changes affect the user while using a brain to computer interface. With that said, the need for the game to be engaging is still a top priority. The group feels that our new game idea incorporates the ability to test the user and provide them an entertaining experience that differs from current on the shelf video games.

Version 3.0

Our new plan is to have a game that has a more specific use for the BCI device, and to have an environment where we can test the effects of different distractions while the user is playing the game. Thus, we have decided to work on four main rooms with different themes and distractions, and refine the three hypotheses that our project will be attempting to test.

FUNCTIONAL SPECIFICATION

GAME OVERVIEW

Philosophy

The creation of this game exists to serve two purposes. First, to experiment with Brain Computer Interfaces and determine what qualities of game design can accommodate this relatively new technology. It is the goal of this group to determine what works and what doesn't work when the player is using a head mounted control device similar to the Fnir or NeuroSky.

Figure 1. Drexel University's FNir Device.

Figure 2. NeuroSky's EEG device.

The second goal this group would like to accomplish is to design an engaging platform for data collection from the player. While this first and foremost is our attempt at a fun and exciting storyline, we want to be able to present the world to the player and be learning a lot about the player and how their brain responds to our stimuli behind the scenes.

Maxwell's Demon has been developed using the Blade3D game engine by Digini, Inc. We chose this game engine mainly because a previous project, *Lazy Brains*, was used to create a BCI prototype using the Fnir, one of the primary types of controllers available to us. Additionally, Blade3D has all the characteristics of a completed game engine, now that the public release has gone live. While some of the Beta versions had serious limitations, we feel that the full release will deliver on its promises. For that reason, we have decided to continue using Blade3D to develop *Maxwell's Demon*.

GAME MECHANICS

Core Game Play

- The game is a first-person perspective puzzle game that allows the user to manipulate the environment to clear various hazardous rooms.

- Harnessing the power of brain-controlled devices the user will be able to position floor tiles to form his path, using levels of concentration or attention and relaxation or meditation.

- The main goal of the player is to use floor tiles that have various behaviors (see Section X, *Game Play Elements*) to get from the room's entrance to the exit without being harmed by the environment.

- Various environmental obstacles exist to prevent the player from reaching their goal.

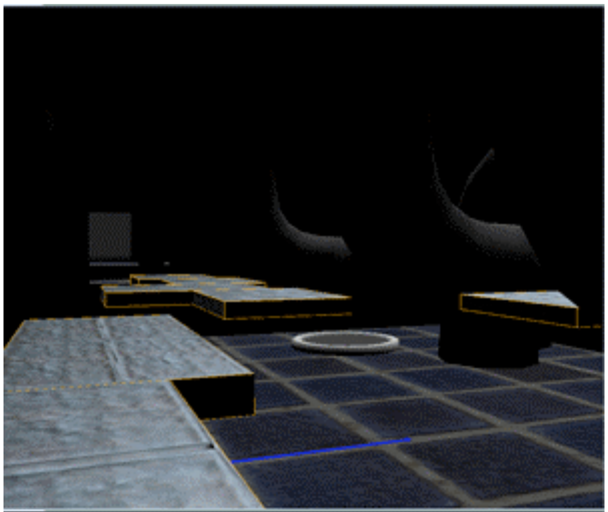
- In-game auditory and visual distractions present secondary challenges that attempt to interrupt BCI communication between the player and the game.

Game Flow

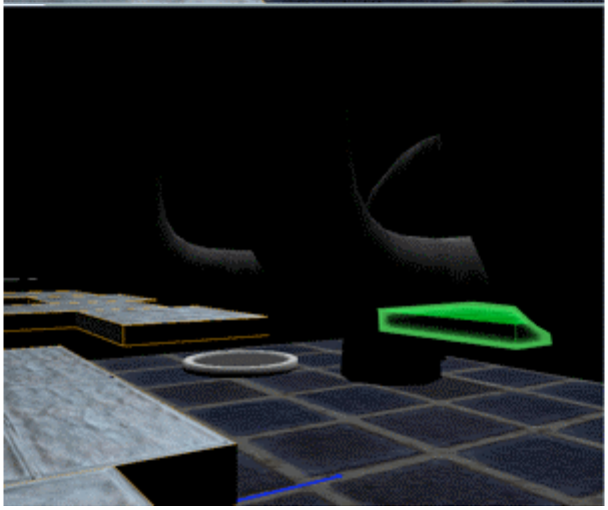
SUMMARY

This section describes how a typical level would play out from the gamer's perspective.

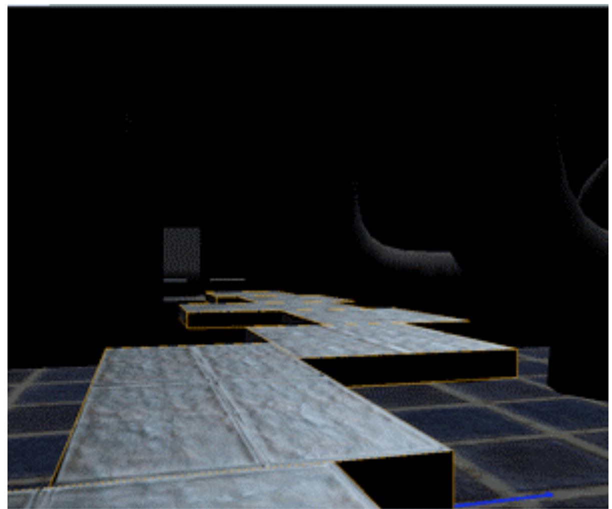
1. After the player has been fitted with the BCI device and started *Maxwell's Demon* they will encounter the first level. The player finds him or herself in a room filled with pipes and steam vents. In their direct view is an exit door which is currently unreachable, as a platform is not properly placed for a clear path.



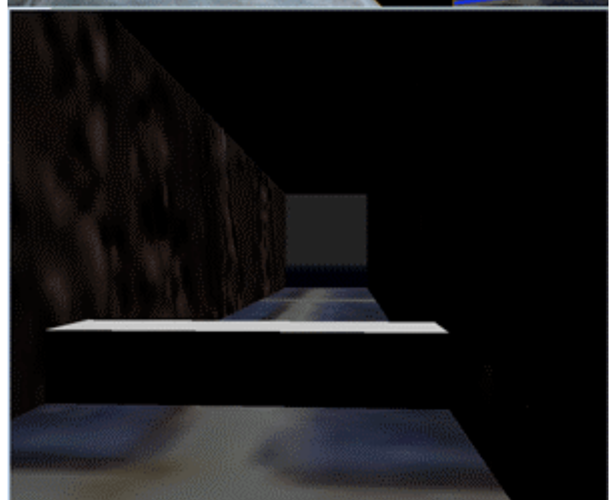
2. In order to navigate to the exit, the player must find and move the missing platform. The first step in moving a platform is selection. Once a platform (that is movable) has been selected it will begin to glow green.



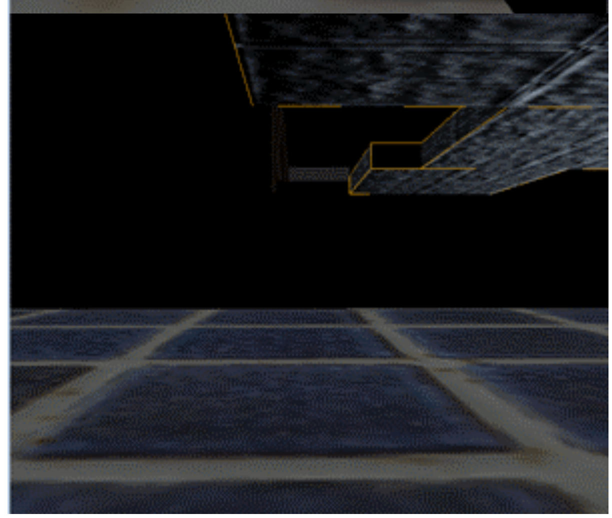
3. By focusing the player's concentration, the displaced (missing) platform will slowly move to the desired position, making a clear path to the exit.



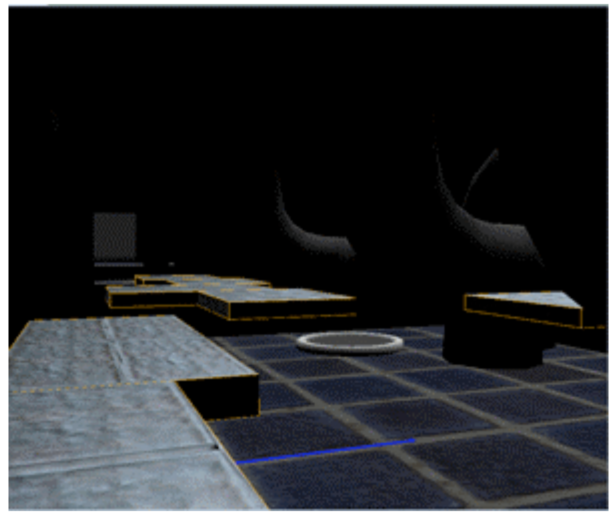
4. Now that there is a clear path to the exit, the player can walk over the path and proceed to the next challenge, in the following room.



5. However, if a player falls off of the path or attempts to jump a distance that is too great, the player will fall to their death.



6. The consequence for death is restarting the level from the beginning with all platforms placed in their initial locations. Each level the player enters will be increasingly more difficult caused by more complex platform movements, auditory or visual distractions that prevent communication with the BCI device (ie. Lower concentration levels make platform movement more difficult).



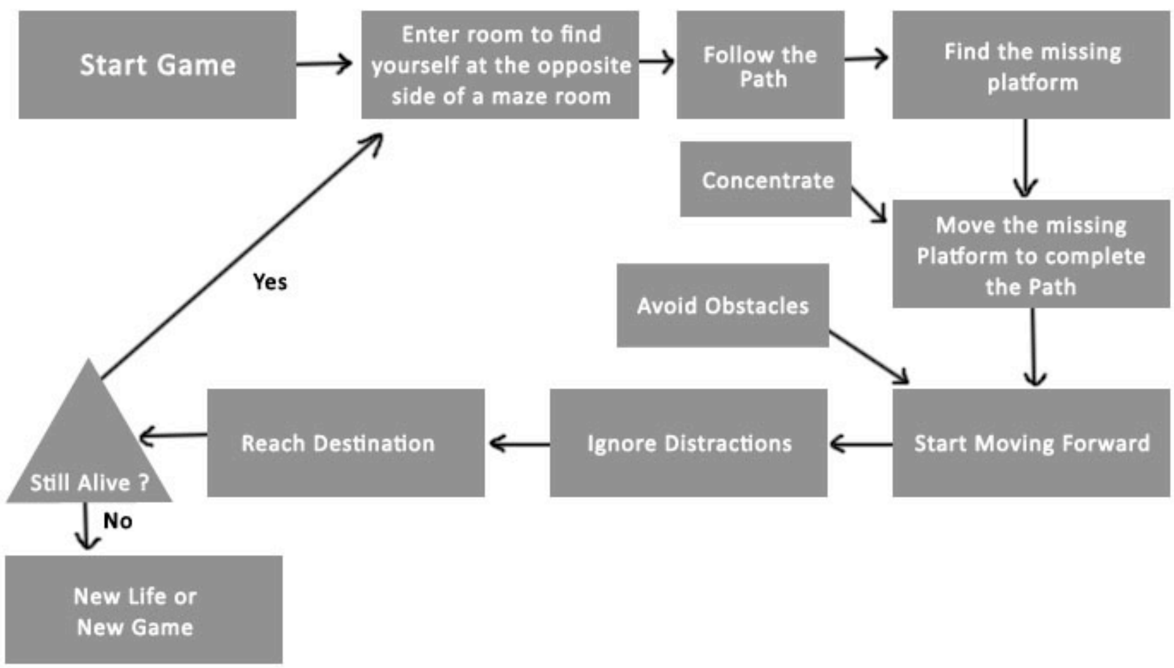


Figure 3. Game Flow Diagram

Characters

Player Character

As a function of a first person perspective game, the player character in the game is never seen. The player is simply viewing the world of *Maxwell's Demon* from the eyes of the person trapped inside it.

Platforms

Platforms are the player's lifeline with the game world. They will need to be manipulated in order for the player to reach the exit. While some platforms have no initial motion and can be moved in the game world, others have constant motion but cannot be translated in three dimensional space. For this reason, platforms also exist as an obstacle the player must overcome. (For a thorough explanation of platform behaviors, see *Game Play Elements*)

Environmental Hazards

Players must avoid environmental hazards at all costs. Hazards will cause the player immediate death and will force the player to lose any progress in the current room by resetting all platforms to their initial position and transporting the player back to the start position.

Game Play Elements

SUMMARY

The player can control their character using the gamepad and the BCI device.
While stationary platforms need to be manipulated to clear paths to the exit, motion platforms cannot be moved by the player and have their own behaviors.
Obstacles and Hazards block or kill the player while trying to escape the room.
Distractions attempt to interrupt communication between the player and the BCI device.

Player Character

The player has the following abilities throughout the game play experience:

- Four-directional movement via the Game Pad analog stick.
- Camera control movement via right analog stick combined with movement.
- Platform selection via looking at the platform and centering it on the screen.
- Platform movement via the brain computer interface device (NeuroSky, Fnir, etc.)

Platforms

- Platforms can have motion or be motionless.
    - **Motionless Platforms**
        - Only selected motionless platforms can be manipulated by the player.
    - **Motion Platforms**
        - Motion platforms fall into one of the following categories:
            - **Translate Block –** Translates periodically on either X, Y, or Z axis.
            - **Wobble Block –** Attempts to tip the player off balance.

Obstacles/Hazards

Obstacles are objects which cannot be manipulated by the player. Obstacles are generally hindrances to the player that should be avoided.

- Pits
    - Pits are found in every level of *Maxwell's Demon* and line the floor of each room. If a player falls from a platform into the pit, all platforms will be reset to their initial positions and the player will be transported back to the starting position. Pits should be avoided at all costs.
- Gears
    - Gears are the most commonly found obstacle in the rooms of *Maxwell's Demon*. Gears are generally environment decorations that block the path of the player.

- Trip Hammers
    - Trip hammers are sometimes found swinging from the ceiling of more advanced rooms. If a pendulum is obstructing the player's path, they should either find a safer path or time their movements precisely to avoid being knocked into a pit or other hazard.

Distractions

Distractions differ from obstacles because they do not directly injure or penalize the player in any way. Distractions exist only to prevent communication between the player and the BCI connection that controls platform movement. Players do not have control over distractions. Distractions can be auditory, visual or temporal in nature.

- Auditory Distractions
    - Sirens
        - Constant Fire alarm.
        - Sudden Fog horn.
    - Hissing Steam
        - Constant leaking gas.
        - Boiling tea-pot.
    - Ticking Sounds
        - Clock-like ticking
    - Heart Beat
- Visual Distractions
    - Steam clouds
        - Block the player's visibility of gaps in path.
        - Hide the player's view of movable platforms.
    - Blinking lights
    - Lens Effects
- Physical Distractions
    - Wobbling Platforms

Game Physics and Statistics

The physics of the game are designed to replicate the experience of a normal human walking around on platforms. The acceleration and speed of the character is designed to be similar to those of an average human. The BCI activity elements will create forces and effects beyond the scope of human potential. The motion of platforms being influenced by BCI will respond as if being subjected to a magnetic or gravitational force, simulating the supernatural power of the mind.

The platforms in the game exhibit unnatural physical properties. The platforms all ignore the effects of gravity, and many of them move on their own, without any visible influence.
This suggests the mystical nature of the platforms. The platforms motion is not influenced by the character's interaction with the platforms. For example, a translating platform retains its velocity even when the player jumps on the platform.

Additionally, when a character gets on a platform, the character should not slide around like a real object might, on a real platform.

USER INTERFACE

SUMMARY

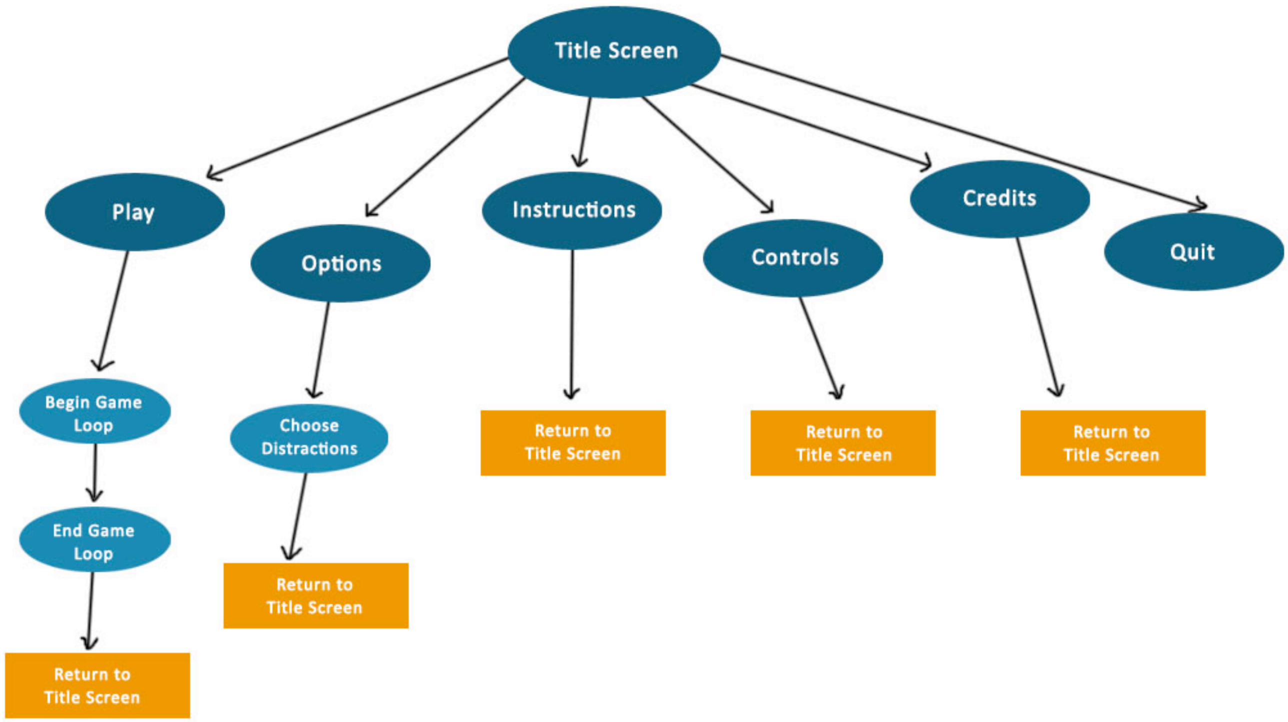
HUD elements have been kept to minimal to allow a greater view of the scene. A bar that represents BCI feedback is displayed for the player. As the attention level of the user increases, the red portion of the bar rises, and drops again as the attention level decreases.

Maxwell's Demon features a minimalistic HUD that will inform the player of the attention level being registered from the BCI, and displays in the form of a vertical bar. The Game Pad's right analog stick controls the players view. To target, the movable platform is required to be centered in the players view.

The BCI controls the pushing and pulling strength of the force that moves the tiles around the board. A threshold level of attention will be required to engage the 'movement mode' of an object. Once the movement mode is engaged, it will remain engaged until the attention level dips back below the threshold level. When the movement mode is engaged, increasing the attention level will draw the object closer, and the object will move closer in the direction of the player's center of view.

Ideally, the game is set up such that certain objects in the environment are perceived, as reacting to the players will. The player will focus on a moveable object, concentrate (raising attention level), and in response the object will move.

Flowchart


Title Screen
Play
Options
Instructions
Controls
Credits
Quit
Begin Game Loop
End Game Loop
Return to Title Screen
Choose Distractions
Return to Title Screen
Return to Title Screen
Return to Title Screen
Return to Title Screen

Menu Workflow

Play – Start the game

Options – This menu allows choosing of the various distractions that are desired to be enabled in a particular level. This will allow individual study of the effects of different distractions on the user. Every level offers a different set of distractions that can be enabled or disabled.

Instructions – Describes the workng of the game along with the usage of the BCI device.

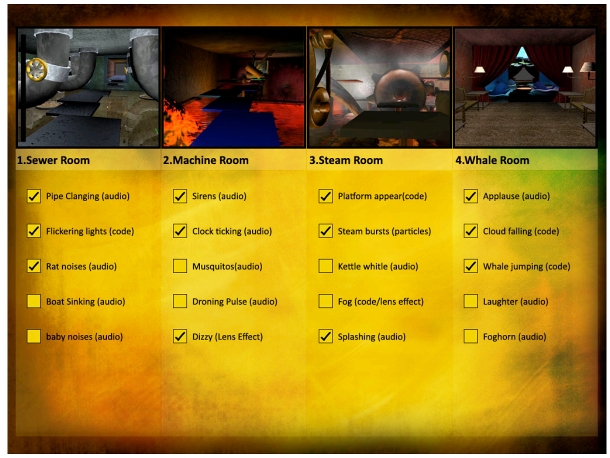
Controls – This menu displays the controls of the game with the game pad and the BCI device.

Credits – Displays the developers and their information.

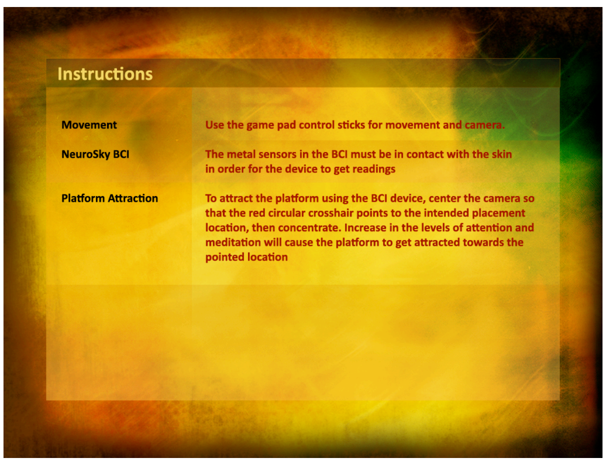
Quit – Quits the game.

GUI Objects

Main Menu

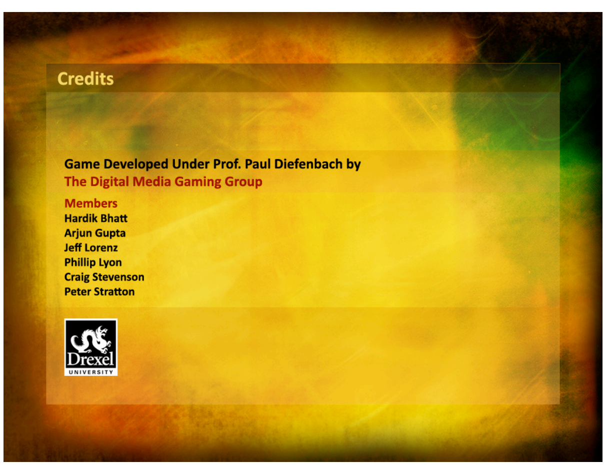


Options



Instructions

Controls



Credits

ART AND VIDEO

Overall Goals

SUMMARY

The world consists of large moving mechanical parts. Art should reflect old, rusty machinery.
The color palette is generally earth-tone browns, gold, and yellow. However, individual rooms may vary based on their function, ie. "Whale Room".

The world of *Maxwell's Demon* takes place inside an enormous perpetual motion machine with gears and pistons. The machine itself is powered by steam and unknowingly to the player, their actions. Therefore, the general aesthetic of the game consists of metal, rusting gears. The color palette consists of earth-tone browns, gold, yellow, and gray. However, certain rooms take on an individual life of their own (a result of the player character's imagination) and may break free from the standard mechanized aesthetic. One such example is the Whale Room, which is comprised of varying shades of blue and gray. Connecting to the story element of *Maxwell's* Demon existing within the player's imagination allowed the development team freedom to create challenging puzzle rooms with variety. With that said, all artists should stay consistent to old machinery and steam punk visual styling. Concept imagery for level design was drawn from images of Victorian machines and the fantasy world of steam punk (machinery powered by steam).

Concept Imagery

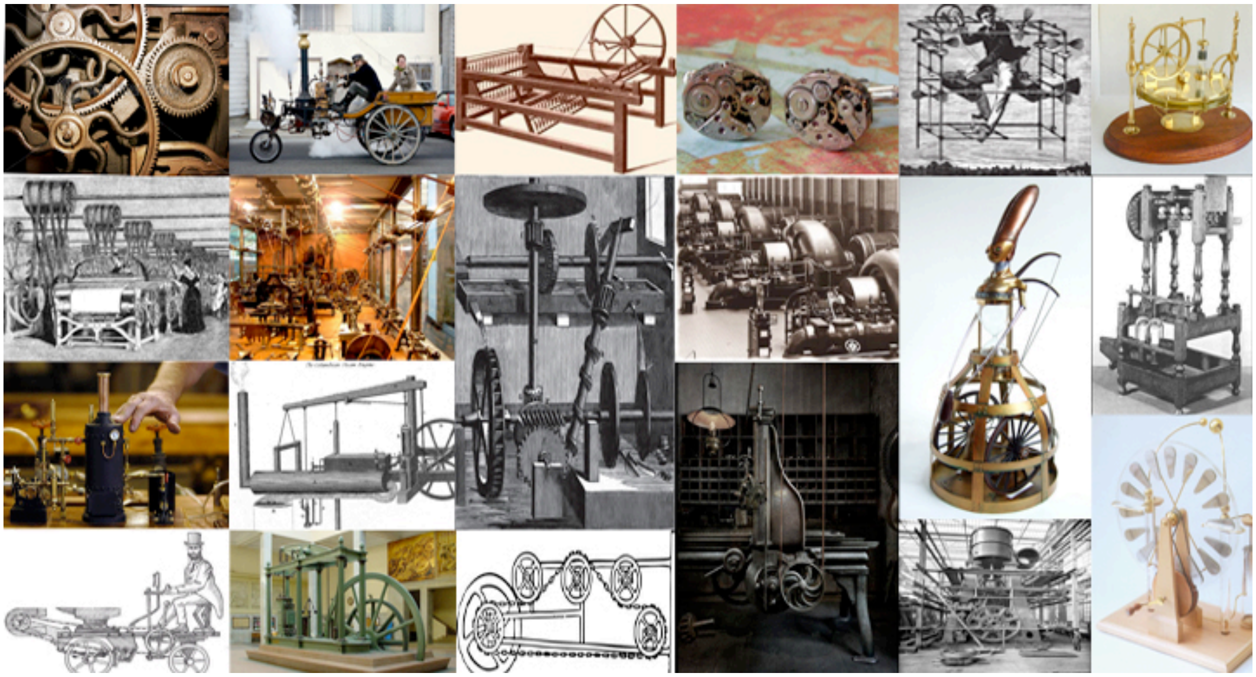
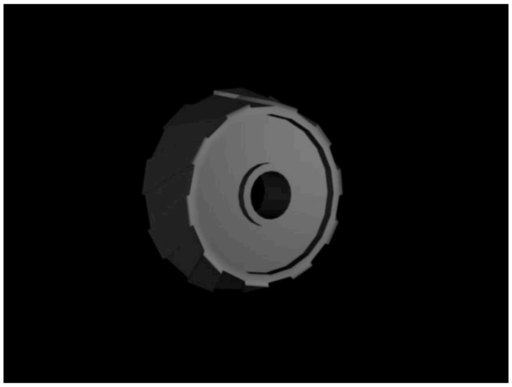


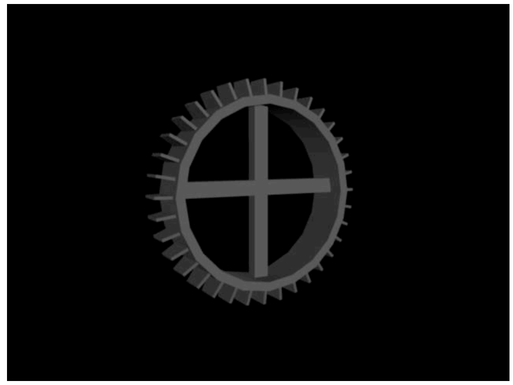
Figure 4. Collection of Victorian machines.

The environment the player is traveling through consists largely of moving machinery and various gears, pulleys, and belt systems all powered by steam. In order to get a good frame of reference, the development team looked at Victorian age machines that exist both in reality and the fantasy world of the Steam Punk aesthetic. Above are various images that we collected in order to put our level designers in the right frame of mind. All future level designs should stay consistent with this world we have begun to create.

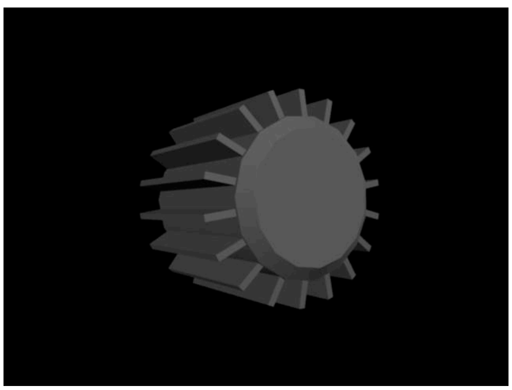
3D Model Inventory



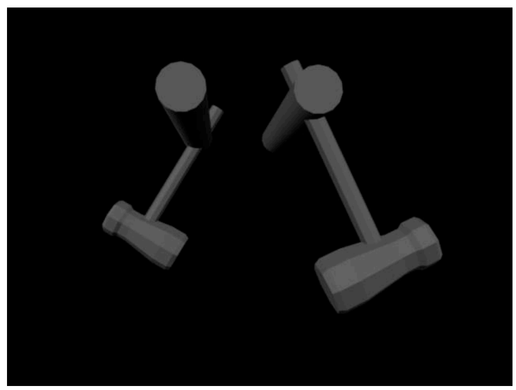
Gear 1



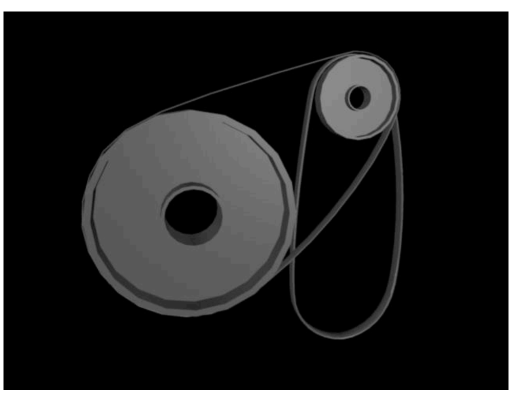
Gear 2



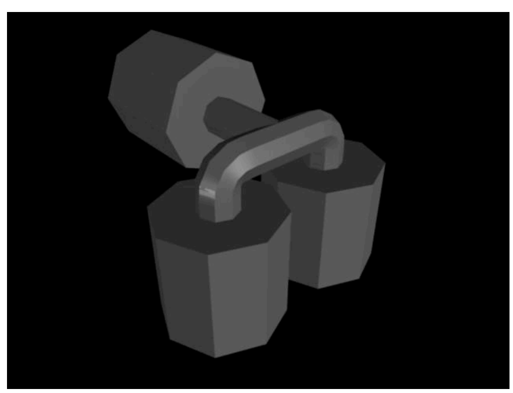
Gear 3



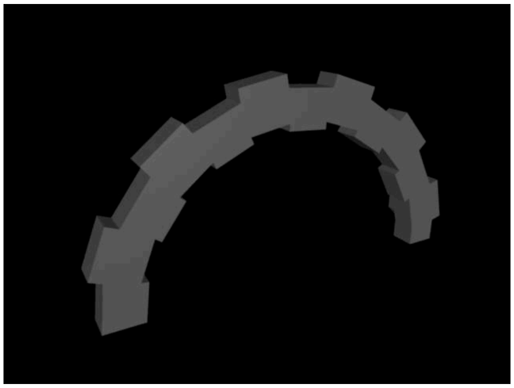
Trip Hammers



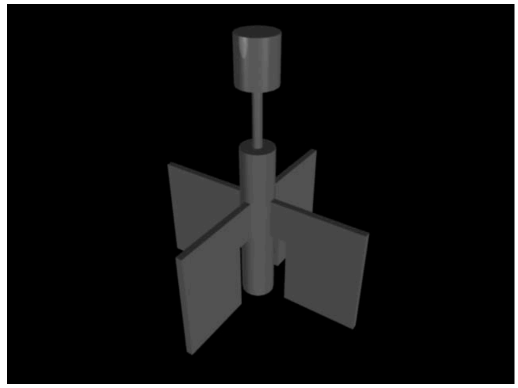
Rollers and belts



Boiler

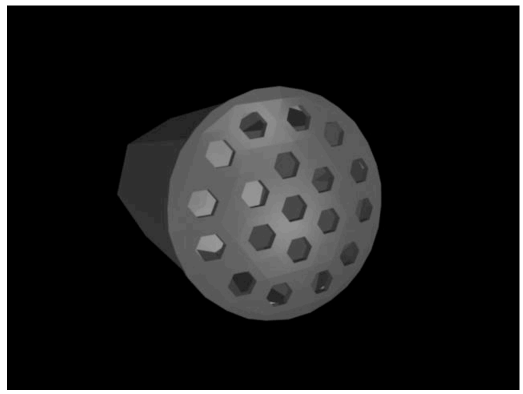


Frame



Turbine

Fan



Steam Vent

SOUND AND MUSIC

Overall Goals

Create realistic yet fantastical ambient soundscapes that honor the steam punk aesthetic and technology for the various rooms and their environments. In addition to the ambient sound, wav and mp3 audio files that interact with the player's own actions and manipulation of objects as well as mechanical objects etc in the room will be added using the positional sound design capabilities of Blade3D. In theory, the creation of a 3D sound environment should be possible.

Sound FX

Sound effects will be generated from both field recordings and Foley sounds recorded specifically for *Maxwell's Demon*. The audio will be edited and mixed using Ableton Live and Logic Pro and then inserted positionally into Blade3D. Certain ambient sound files may be mixed with binaural signal processing, a psychoacoustic processor, which emulates the way the ears hear. It does so by extending the perceived stereo spectrum and simulating sound source positions, creating an environment closer to that experienced with surround sound, but used primarily for headphones. Sound can be perceived not only back to front, left to right, but above and below as well.
For use in *Maxwell's Demon*, Binaural processing will only be used for ambient sound, i.e. room tones or sounds coming from above or below the player, leaving the positional sound to work within a more dynamic background.

Music

A repetitive minimalist score (reminiscent of works by Steve Reich) composed to loop and change intensity and tonal quality throughout the various rooms, will create momentum and a driving sense of urgency. The score will reflect changes based on the danger the player encounters using swells of sound and instrumentation that fits the tempo of the looping music. The music loops will be constructed at times over 5 minutes and layered on top of each other to create the sense of an organic score.

STORY

Story Concerns

The main focus of the development of this game is that of research into Brain Computer Interfaces and their role in video gaming technology. Therefore, our main concern is validating our three proposed hypotheses. However, providing an entertaining experience to the player is also a very important factor in game design, and a failing to provide an adequate story could cause extraneous results from our research. For example, if the player becomes bored, could the BCI readings be affected? This is a concern that the development team would like to avoid in the interest of proper research and good fun gaming.

Overview

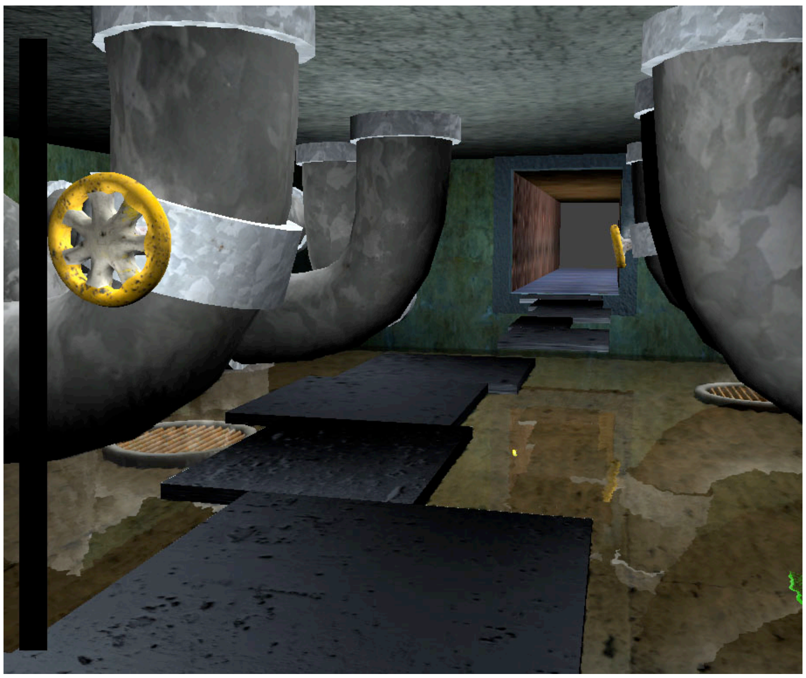
The story takes place in both the game's "reality" and within the mind of the player character simultaneously. All game play events exist within the mind of the player character. By creating a story that allows the player's imagination to be the setting, it allowed certain creative freedom for our team to develop rooms with variety and also allows future development groups the ability to create rooms to their own design, but staying consistent within the defined specifications.

After suffering injuries from a serious car accident, the player has been hospitalized and placed in a drug induced coma while surgeons attempt to revive the player. Unknowingly to the player, they enter a dream state, conjuring up the world of *Maxwell's Demon*. The player wakes up inside a fantastical machine powered by steam and the motion of its own actions. Trapped inside menacing rooms, it becomes clear that the player cannot stay in one spot and must find exit or surely perish. Each room the player enters becomes more difficult and frustrating then last, mimicking the struggle in the real world as his/her life hangs in the balance. Suddenly, the player discovers that they have control of this environment. The ability to move platforms with their thoughts begins to shed new light on this strange new world. Finally it is clear, that the player must summon all of their brain's power in order to escape the contraption known as *Maxwell's Demon*.

LEVEL REQUIREMENTS

An initial goal for this project was create a platform where new rooms could be added which allow the testing of new hypothesis for the brain controlled interface. Our project consists of four puzzle rooms that increasingly become more difficult with the introduction of more obstacles, hazards, and platform manipulation.

Baseline Room



Game Play Purpose

The purpose of the baseline room is to introduce the player to the control scheme and interactions present in the *Maxwell's Demon*.

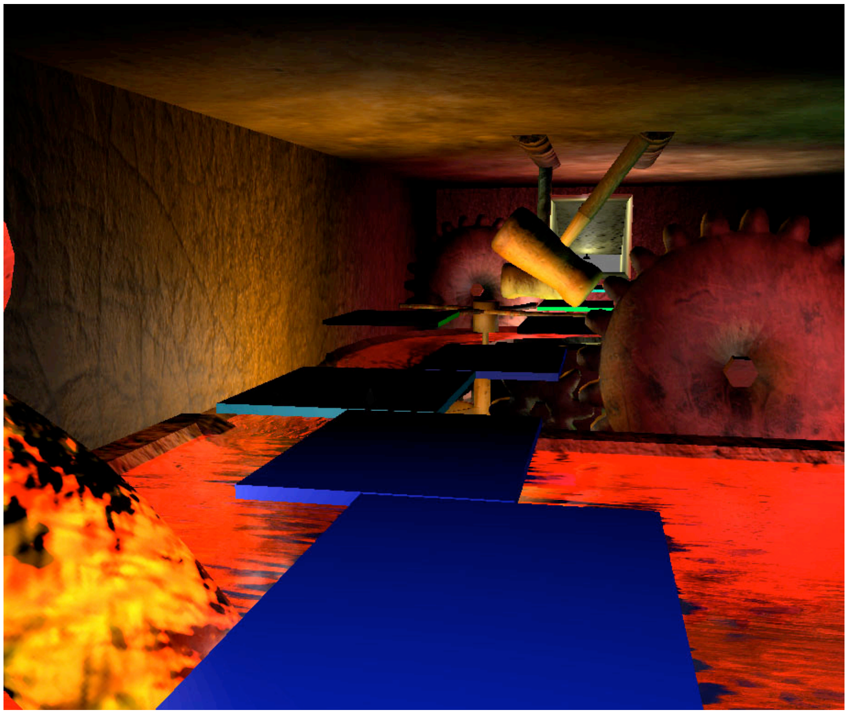
Research Purpose

The baseline room gives the development team a reading from the BCI device that we will be able to measure and compare during future game play. The level design is purposefully simple in order to focus on the player's ability to communicate with the BCI device. No auditory or visual distractions are present in this initial room.

Required Skills

The player is faced with the simple task of selecting the hidden platform on the right and concentrating in order to enable its movement to the rest of the blocks. This will give the player a clear path to the exit and allow them to enter the first room with a series of challenges.

Machine Room



Game Play Purpose

The purpose of *Machine Room* is to introduce the player to a more challenging puzzle. This room has multiple hazards and obstacles that must be overcome for the player to succeed in reaching the exit.

Research Purpose

Machine Room is the first puzzle room that we will be testing one of our hypotheses. In this case, we are trying to determine how well a BCI device works as an active controller (for more information, see Appendix A).

Required Skills

In *Machine Room* the player has two gaps in the path to the exit. By concentrating, the player can manipulate the two platforms, shown above in green, and make their way to the exit. However, the player needs to dodge disappearing platforms, shown in red, and the pendulums that are swinging from the ceiling.

The Steam Room

Game Play Purpose

The *Steam Room* takes the difficulty level and increases it a few notches. The player is faced with auditory and visual distractions, and multiple motion platforms. Distractions include a intermittent fog horn blast sound effect, steam particle effects, steam sound effects, moving gears from every view, and haze that disrupts the viewport. Constructed to mimic a iron smelting factory, the floor is lined with hot molten metal which will ensure certain death to the player.

Research Purpose

The *Steam Room* is an attempt to throw multiple distractions at the player and determine if audio and visual elements interrupt communication of the BCI device, an additional hypothesis.

Skills Required

The *Steam Room* requires the player to reuse movable platforms in different locations throughout the map in order to make it to the unseen exit. The player must combat various forms of distractions as well as proper timing order to board the motion platforms at the correct time.

The Whale Room

Game Play Purpose

The purpose of the *Whale Room* is to present the player with a different type of challenge. In previous rooms, player's needed to concentrate in order to move platforms. In this room, the control scheme changes. Frustration levels move this platform. While the room seems like a very easy puzzle to overcome, we believe it will prove to be more difficult.

Research Purpose

In the *Whale Room* the development team is attempting to test frustration levels through a BCI device. Since this area has not previously been explored, the team will be looking for variations in reading from previous puzzles in an attempt to record a pattern among multiple players. Because of the exploratory nature of this room, the team decided to make the *Whale Room* the final test. If no player is able to finish this puzzle, other tests would be unreachable.

Skills Required

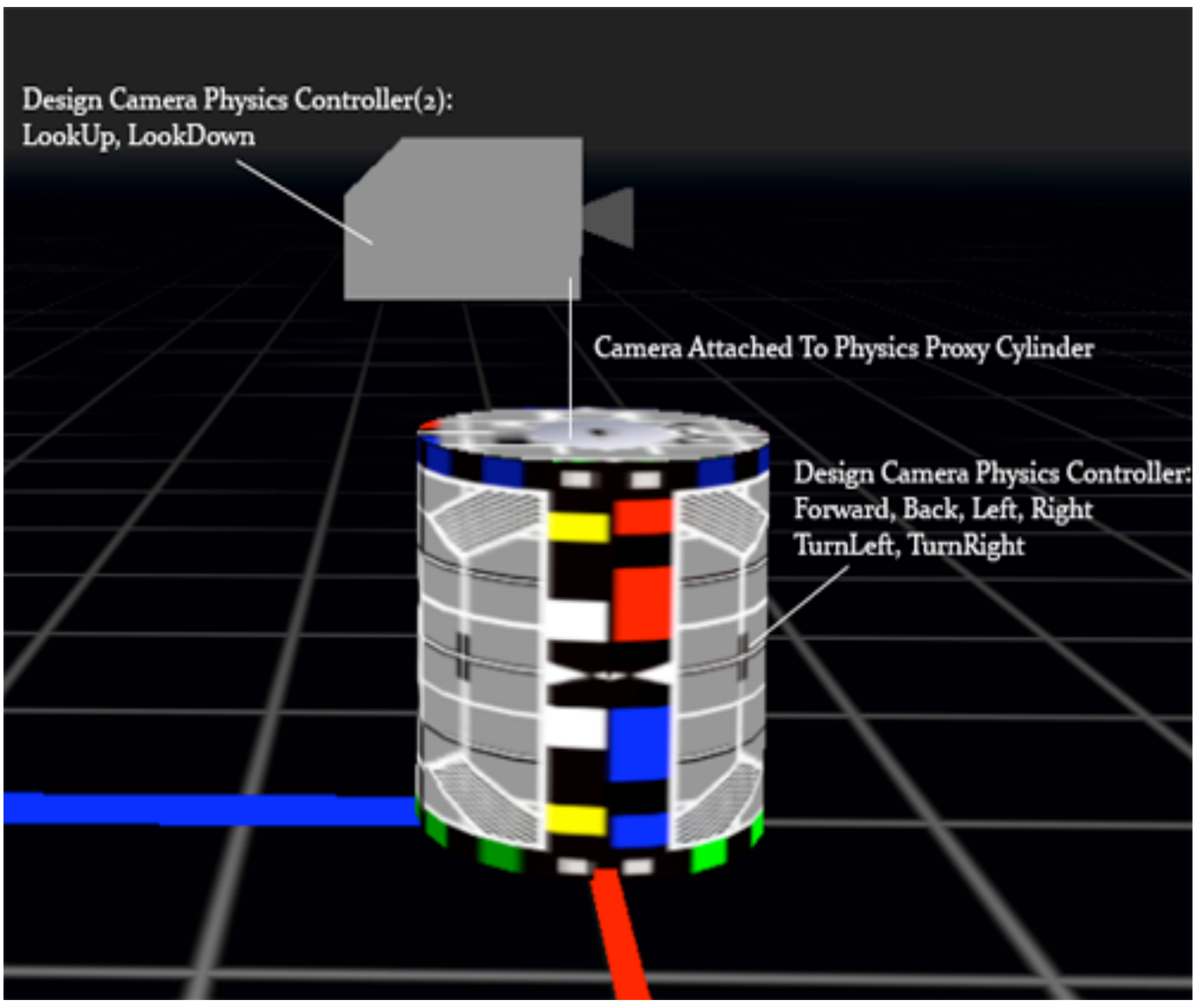
While this room seems very simple, the development team believes it will be one of the more difficult rooms. The player must move the single platform onto the path by raising their frustration level. In an attempt to combat this, the room consists of soft blue colors and whale song auditory distractions that are attempting to calm the player.

TECHNICAL SPECIFICATION

Game Physics and Statistics

Character

Character control interactions are handled through a combination of Blade3D's built in physics controller types with custom control code. The character control proxy consists of an invisible cylinder assigned with a design camera control component. All undesirable control options are disabled, leaving only the ability to move forward, back, left, and right and the ability to turn left and turn right. Through a series of control graphs, the first person camera is attached above the cylinder, and has additional freedom of looking upward or downward, though this has no effect on the actual movement of the proxy cylinder. The camera springs back to the horizontal whenever the player is not currently looking upward or downward. Due to the design camera physics controller ignoring the effects of global gravity, a downward force is applied to the proxy cylinder via a graph to correct.



Platforms:

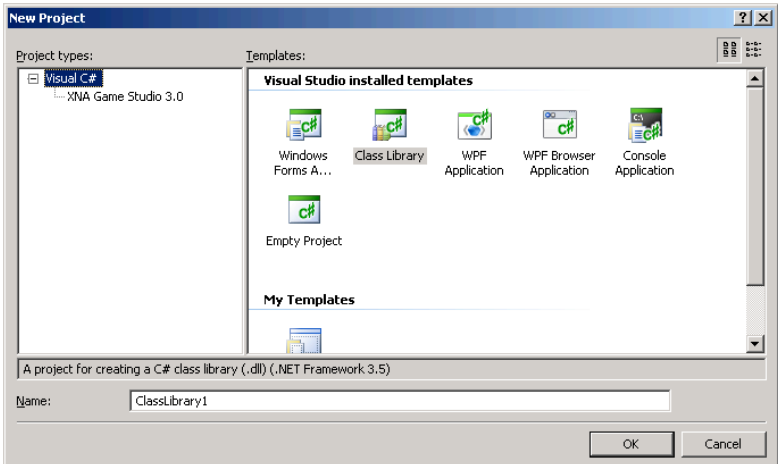
- Immobile
    - Standard platforms are designed to be stationary. These platforms are cube primitives with rigid bodies and cube collision. Additionally, these platforms are not subject to gravity, do not react to collisions, and are immobilized.
- Active
    - There are several cases of platforms that move automatically. Specifically, the translation platform, the rotation platform, and the disappearing platform. The translation and rotation platforms are similar, in that they utilize physics attach to components and volume trigger components.   Basically, the platforms oscillate between two or more nulls, in terms of either rotation or translation. These platforms also ignore gravity and do not react to collisions.
- Actionable
    - These are the most important platforms in terms of game play. These platforms react to input from the BCI device.   The primary actionable platform is moved in space in response to user input. When activated, these objects gravitate towards specific waypoints, based upon the user's center view point.

Code Objects

1.) Guide to Writing Components

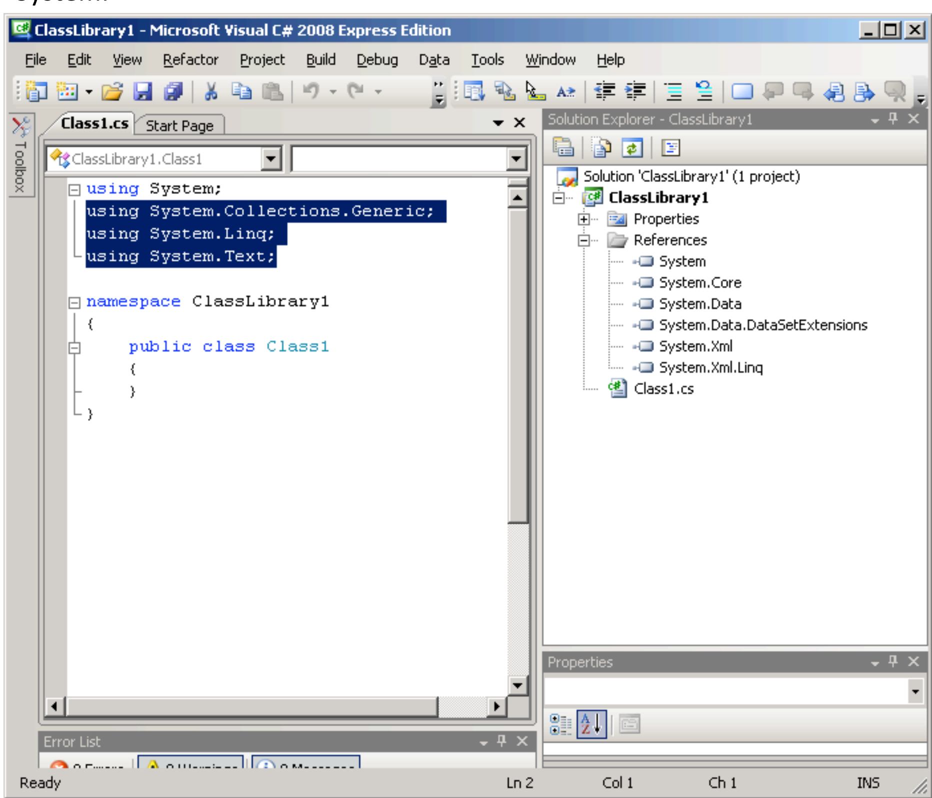
Creating a Visual Studio Project

a. Create a new C# Class Library Project, naming it whatever you want.



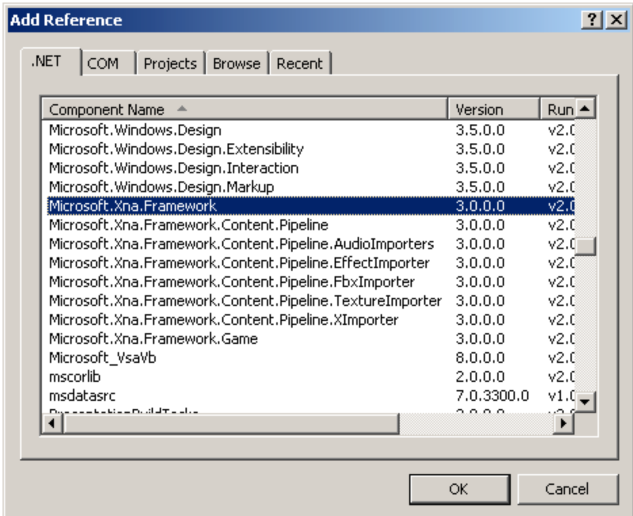
b. Delete everything in the "using" section except for "using System;".

c. Open up the "References" section in the Solution Explorer and delete everything except "System."

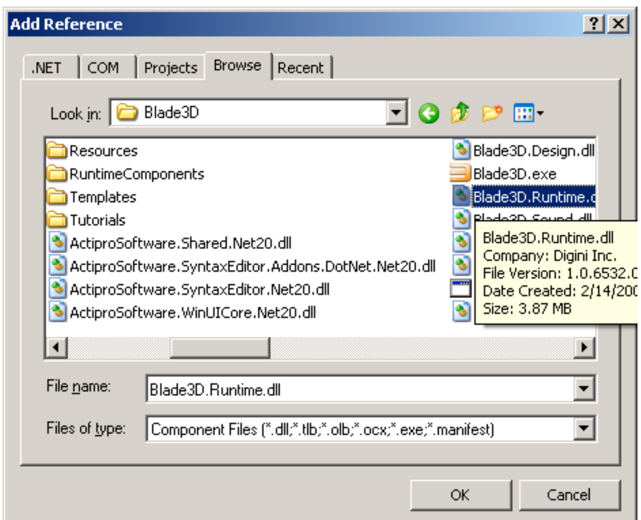


d. Now right-click on the "References" and choose add a reference.

e. Add Microsoft.Xna.Framework from the list of .NET references.



f. Now go to "Browse." Go to C:/Program Files/Digini Inc/Blade3D and choose Blade3D.Runtime.dll.



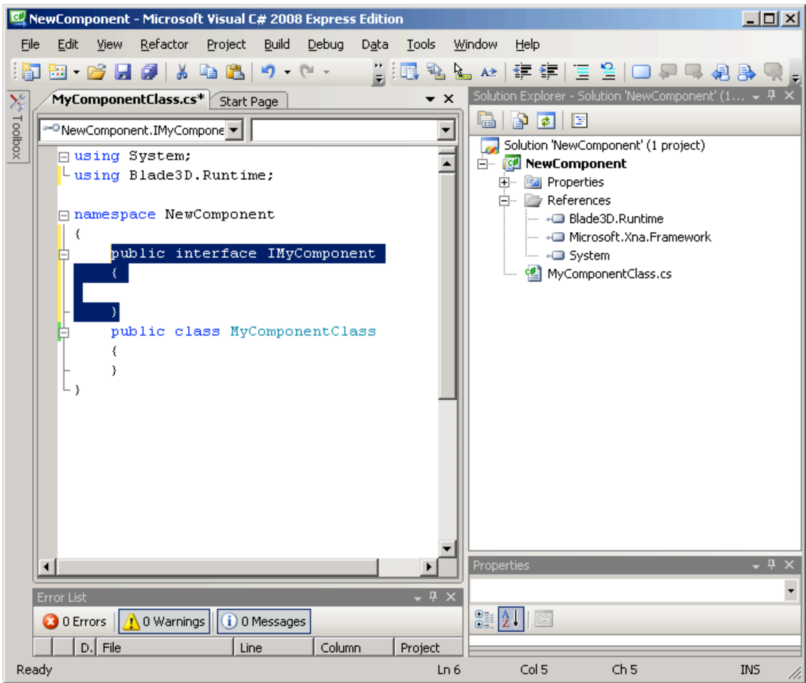
g. Back in the code window, below "using System;" add "using Blade3D.Runtime;".

h. A quick way to save this whole project this to go to file and then click on “Close Solution” and save your whole project somewhere you can find it.

2.) Coding Guidelines:

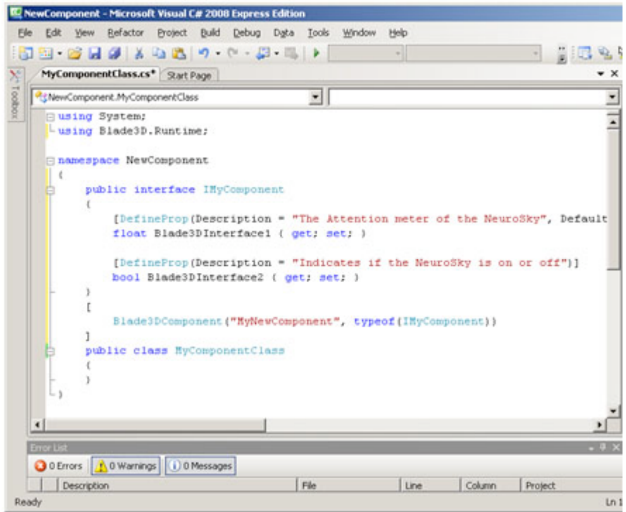
a. In order to see the properties window inside Blade3D, you must define an interface for your custom class.

b. Between your namespace and class function simply add a public interface. Name it something meaningful so that you can refer back to in it your code.



c. Now add some Blade 3D Logic [DefineProp(*custom properties*)]
This will let Blade3D know that it should display this in the engine as a property of the component. There are various properties you can assign inside the parenthesis, but the most common ones are Description and DisplayName (the text that is displayed when you hover the mouse over it, and the name that the property will display). The others will pop up from Intelligence and are usually self-explanatory.

d. Next, you should write the class which will use this interface, the following should go above your class:



   i. *category name* refers to which submenu the component will appear in inside the context menu when you right-click and object and choose "Add Component."
   ii. *type* is simply which type to use, and it will most likely always be the interface we wrote for our purposes, so use "typeof(*interface name*)."

e. The class must inherit from the interface, as well as "BaseComponent." (i.e. className : ComponentBase, interfaceName)
f. Inside the class, we have to define the property objects for our interface. This is the way to do it: public Property< *type* > variableName;
   i. *type* is the type that you chose when you wrote the variable in the interface (string, int, float, etc). Inside the constructor, you should *new* each Property.
   ii. variableName = new Property< *type* >(this);
g. Next, you need to handle the interface variables:
   To make it easy to detect put them in between regions
   #region Interface Methods
   type interfaceName.variableName
   {
     get { return this.*variableName*.Value; }
     set { this.*variableName*.Value = value; }
   }
   #endregion

3.) Writing Functions

a. The following are some miscellaneous functions that you can override inside your class (note that these are just a few examples, and are also subject to change or deletion).
    i. void Attach() – This function is called when the component is attached to an object in the Blade3D Scene.
    ii. void Detach() – This function gets called when the component is removed from an object in the Blade3D scene (called automatically when Blade3D closes).
    iii. void PostCreate() – This function is called immediately after Attach() is completed in Blade 3D
    iv. void OnCommand() – This function is called whenever the object receives a command (note: commands are now called "Events"). There are other functions such as OnKeyDown, OnMouseDown, etc that handle more specific commands, but every command goes into OnCommand() first before being sent to the more specific handler.

b. Both the Attach() and Detach() function has a base.Attach(); and base.Detach() and call in it which attached and detach itself to an object in Blade 3D. However before that it needs information on what to put on that component but calling the:
```
this.SceneItem.PreUpdate += MyObjectCode;
```
c. The MyObjectCode is simply another function within you class that is being called by the Attach() function every Blade 3D frame.

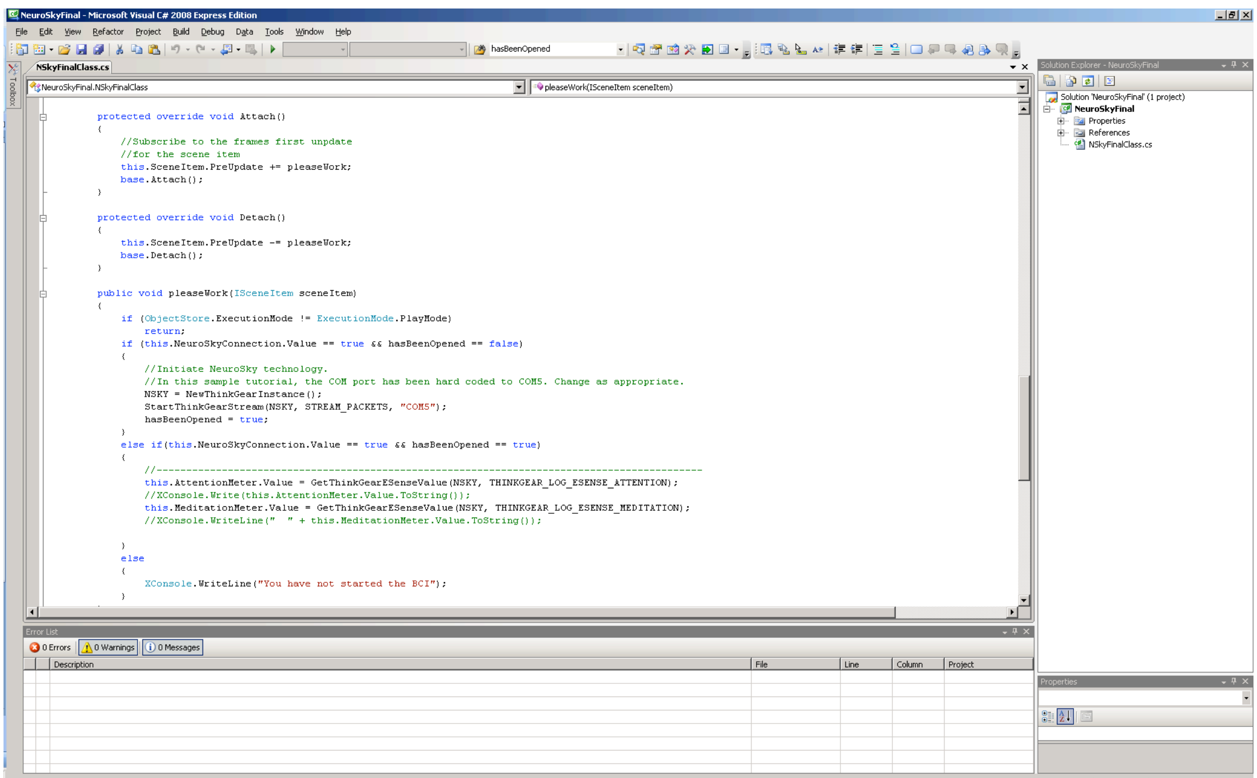
d. The myObjectCode required a ISceneItem function in it to run properly:
```
public void myObjectCode(ISceneItem sceneItem):
```
Everything else is simply basic coding.

4.) Building you DLL

a. You do not compile class libraries, you only build them (F6 hotkey).
b. Blade3D must be closed because if it is open it will be using the DLL that you want to write over, and you will get an access denied error.
c. Once built, navigate to the \bin\Release folder to get your completed .dll file.
d. Your .dll must be in the Blade 3D file in C:/Program Files/Digini Inc/Blade3D in order for you to get the component in the game engine.

NeuroSky Component

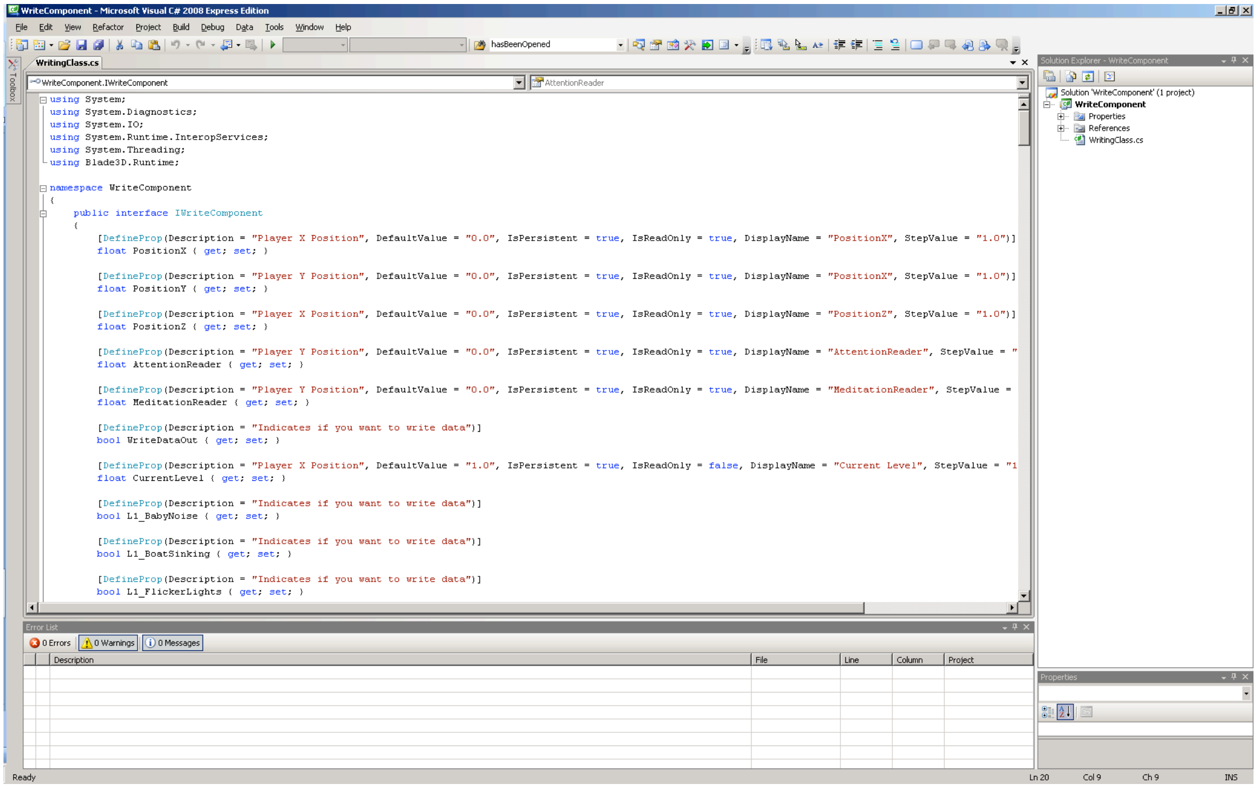
This component takes advantage of the thinkgear.dll which directly communicated with the NeuroSky Hardware device. While the device this both wireless (connected by Bluetooth) and wired (by USB connection). The component mostly utilizes the Wired connection by reading in the specific port that the NeuroSky is connected to and reading in the Brain data through that connection. The component would first communicate with the tinkgear.dll library to get a unique ID which I would use to connect to the hardware device and read in Attention and Meditation data. This data is updated every second but is read in by the component every frame rate with no significant lag.



The resulting output would be a Blade 3D NeuroSky Component called NeuroSkyFinal.dll which would communicate with the game engine and provide it with BCI information.

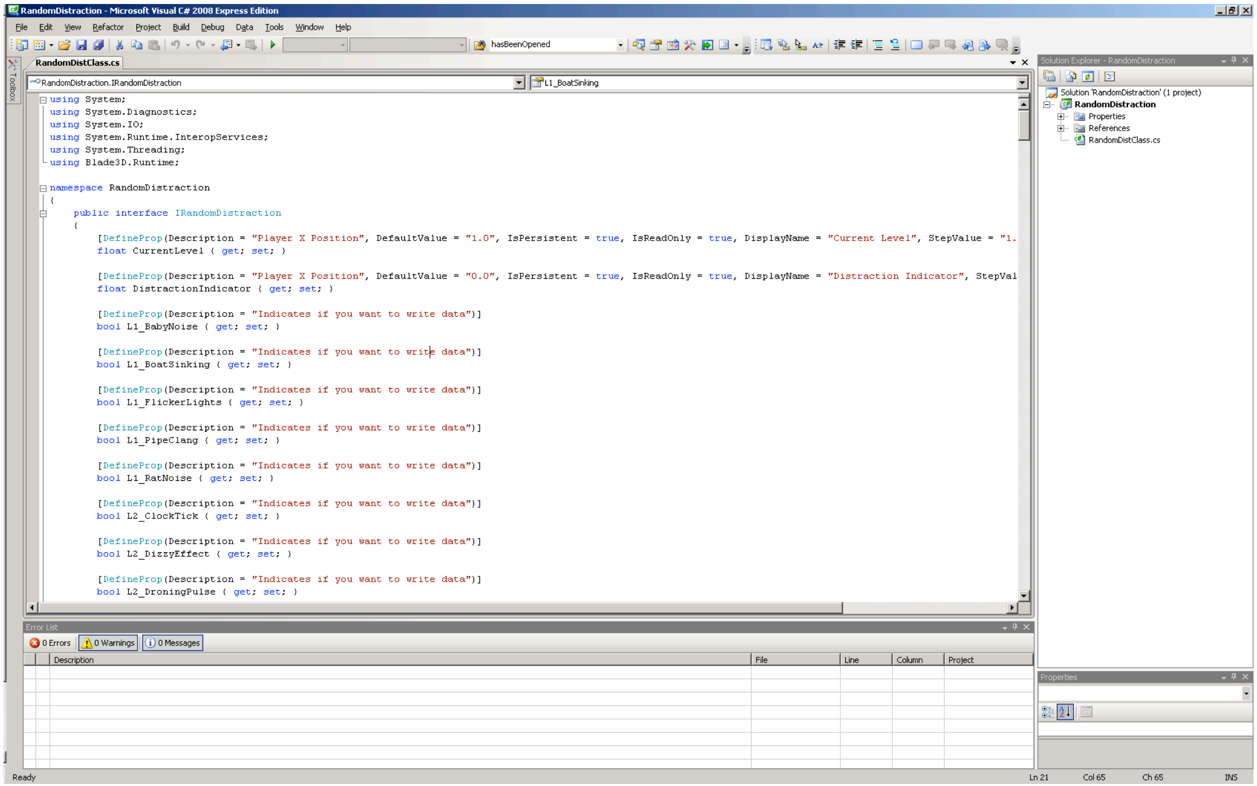
XML Writing Component

This component essentially created an XML file with all the data gathering information needed for research. Like Position of the player, timestamp, the players attention level, the players meditation level, and which distraction is currently occurring. This data is later parsed in a flash program which will outline a graph of your game play.



Random Distraction

The Ransom Distraction component basically selects a random component to use in a given level so that an action of that distraction can be triggered within the game engine during game play.


RandomDistraction - Microsoft Visual C# 2008 Express Edition
File Edit View Refactor Project Build Debug Data Tools Window Help
hasBeenOpened
RandomDistClass.cs
RandomDistraction.IRandomDistraction
L1_BoatSinking
using System;
using System.Diagnostics;
using System.IO;
using System.Runtime.InteropServices;
using System.Threading;
using Blade3D.Runtime;
namespace RandomDistraction
{
public interface IRandomDistraction
{
[DefineProp(Description = "Player X Position", DefaultValue = "1.0", IsPersistent = true, IsReadOnly = true, DisplayName = "Current Level", StepValue = "1.
float CurrentLevel { get; set; }
[DefineProp(Description = "Player X Position", DefaultValue = "0.0", IsPersistent = true, IsReadOnly = true, DisplayName = "Distraction Indicator", StepVal
float DistractionIndicator { get; set; }
[DefineProp(Description = "Indicates if you want to write data")]
bool L1_BabyNoise { get; set; }
[DefineProp(Description = "Indicates if you want to write data")]
bool L1_BoatSinking { get; set; }
[DefineProp(Description = "Indicates if you want to write data")]
bool L1_FlickerLights { get; set; }
[DefineProp(Description = "Indicates if you want to write data")]
bool L1_PipeClang { get; set; }
[DefineProp(Description = "Indicates if you want to write data")]
bool L1_RatNoise { get; set; }
[DefineProp(Description = "Indicates if you want to write data")]
bool L2_ClockTick { get; set; }
[DefineProp(Description = "Indicates if you want to write data")]
bool L2_DizzyEffect { get; set; }
[DefineProp(Description = "Indicates if you want to write data")]
bool L2_DroningPulse { get; set; }
Solution Explorer - RandomDistraction
Solution 'RandomDistraction' (1 project)
RandomDistraction
Properties
References
RandomDistClass.cs
Error List
0 Errors
0 Warnings
0 Messages
Description
File
Line
Column
Project
Properties
Ready
Ln 21
Col 65
Ch 65
INS

Art Instructions

Modeling Basics:

-We use Autodesk Maya 2008 for this demonstration

-Standard Low-Poly modeling rules apply

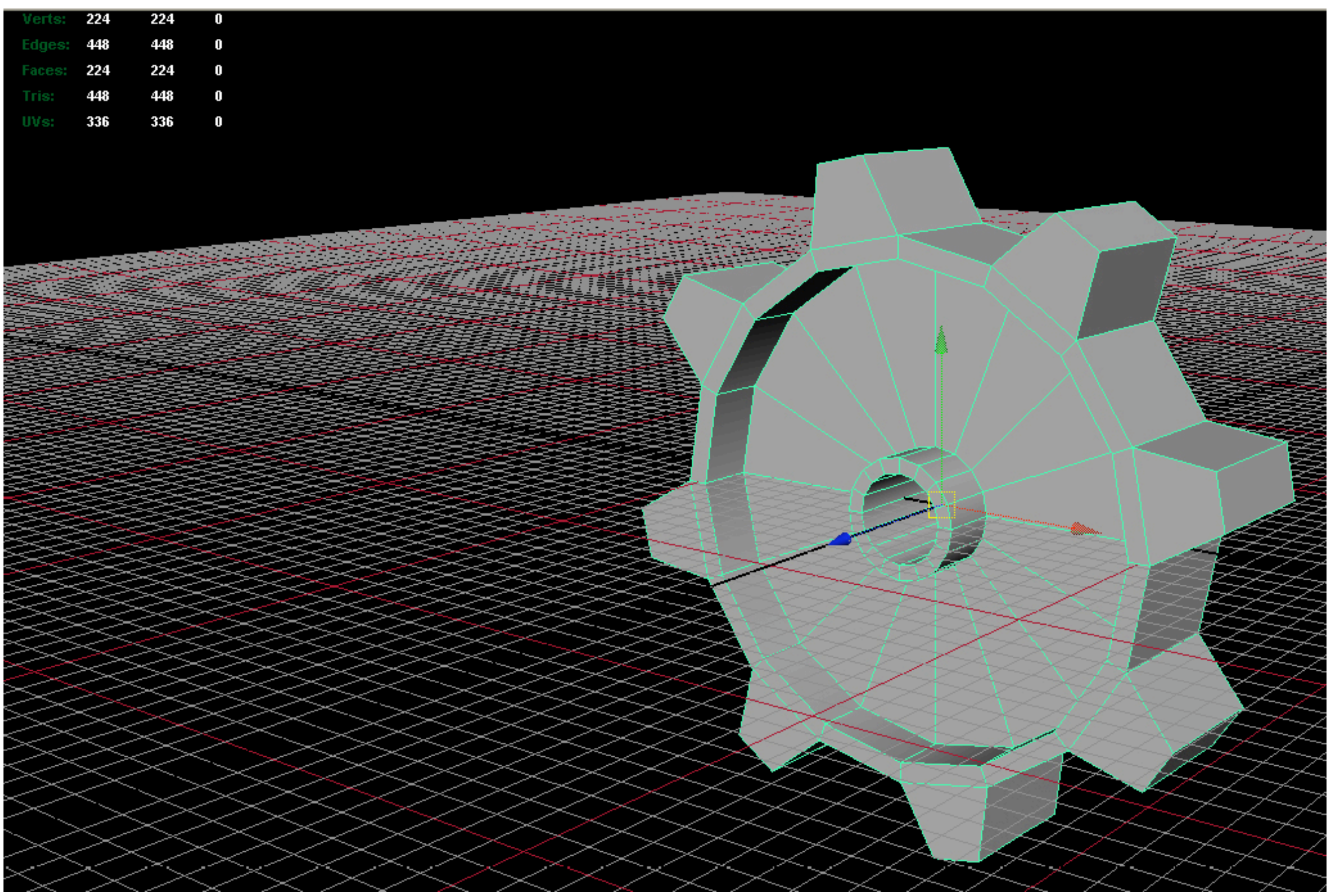
-Faces must have no more than 4 sides and no less than 3

-Objects should have their pivot point at the world center

-Delete all history and freeze the model's transformations before export

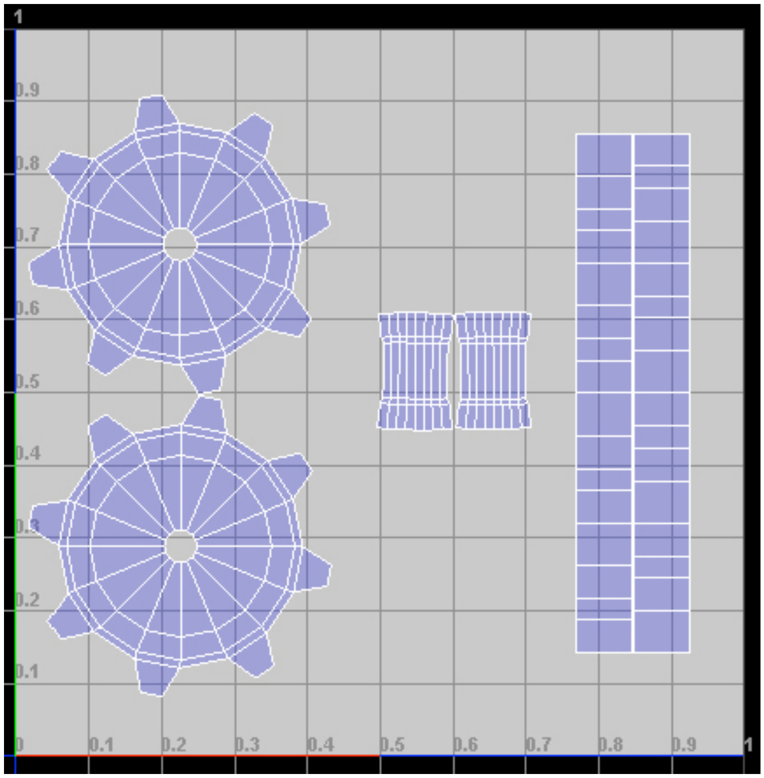
-A standard Blinn shader should be applied

-Test your scene scale to the Blade3D scale before you begin

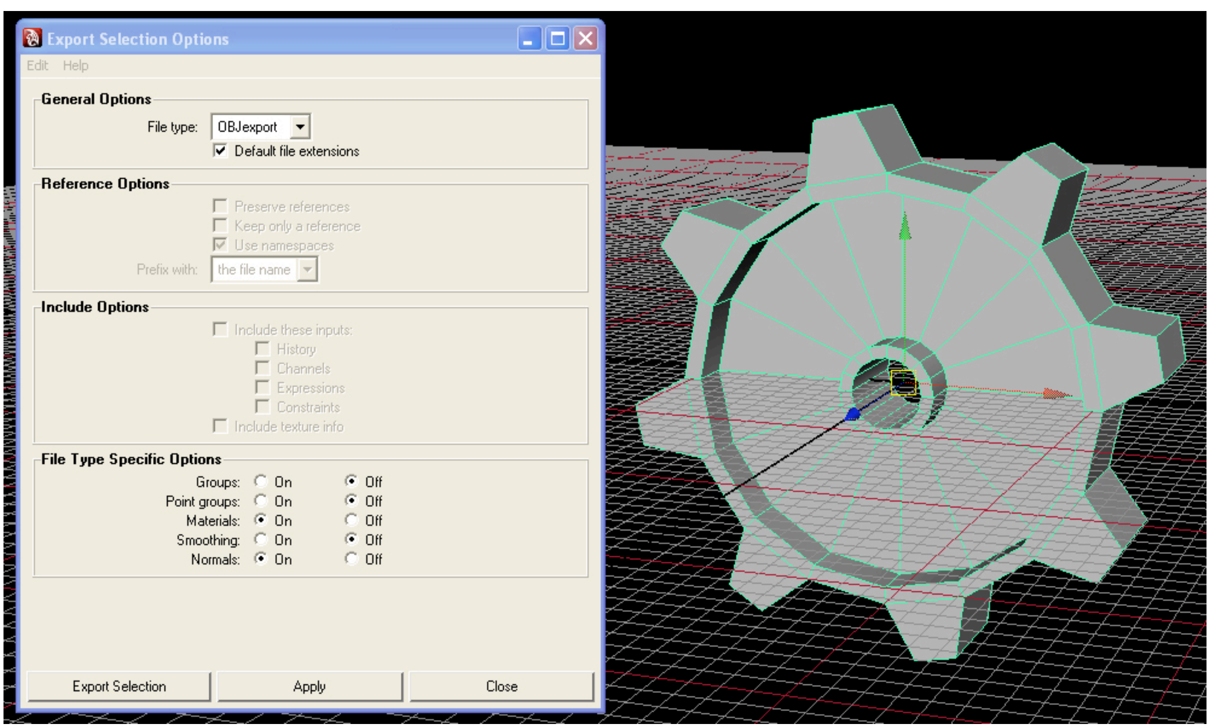


-Object should have it's UV's mapped out correctly

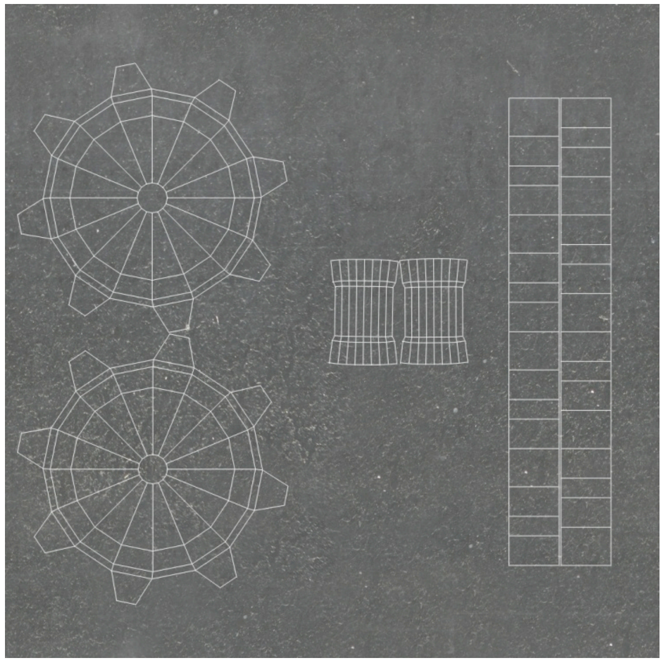
-Normalize your UV's before exporting a Uvmap



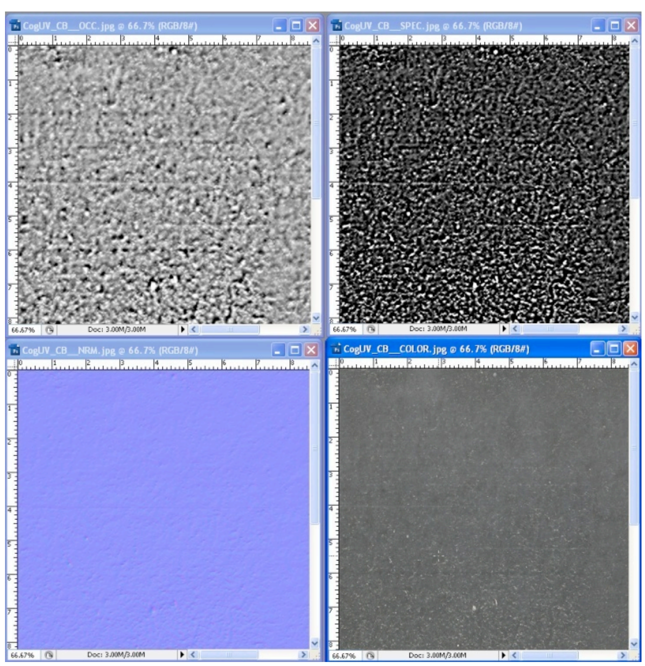
-Export file in OBJ format



-Color Texture done in Photoshop



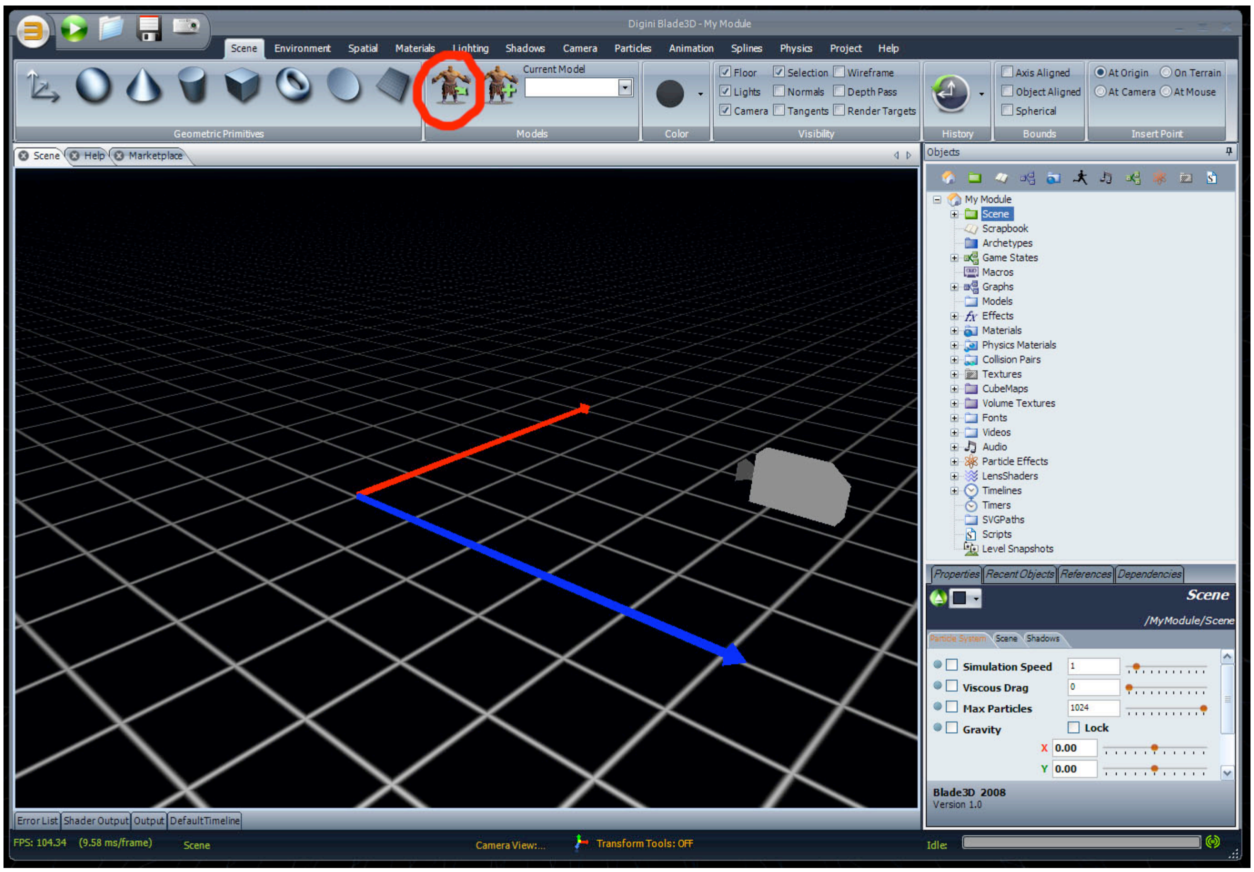
-Normal, Ambient Occlusion, and Specular maps done in CrazyBump



Procedure for importing 3d models (.obj) into blade 3d

We have used Maya to create all our 3d models, and have made use of .obj format to export them and bring into Blade 3d. All the models are to be exported without the textures as they create errors in Blade 3d. The correct way to apply textures to the 3d models is to apply them within Blade 3d.

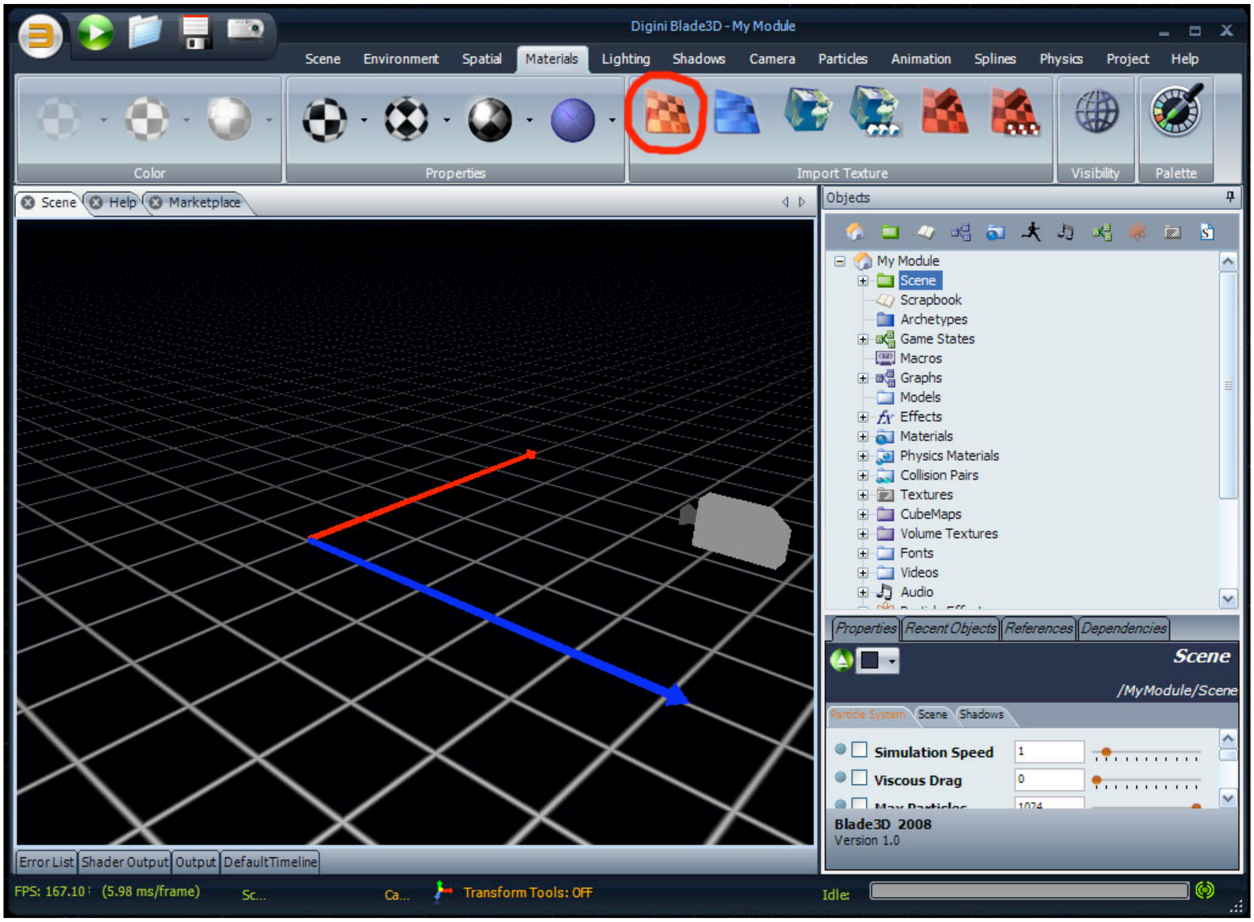
To import an .obj model click on the "Import Model" icon in the "Scene tab" as shown below and browse the path of the object to bring into the stage



The position of the object in the stage will depend on its position saved from Maya on the world axis. To move or re-position the object go to "Spatial tab" and click the "Translate" icon .

Procedure for importing textures into blade 3d

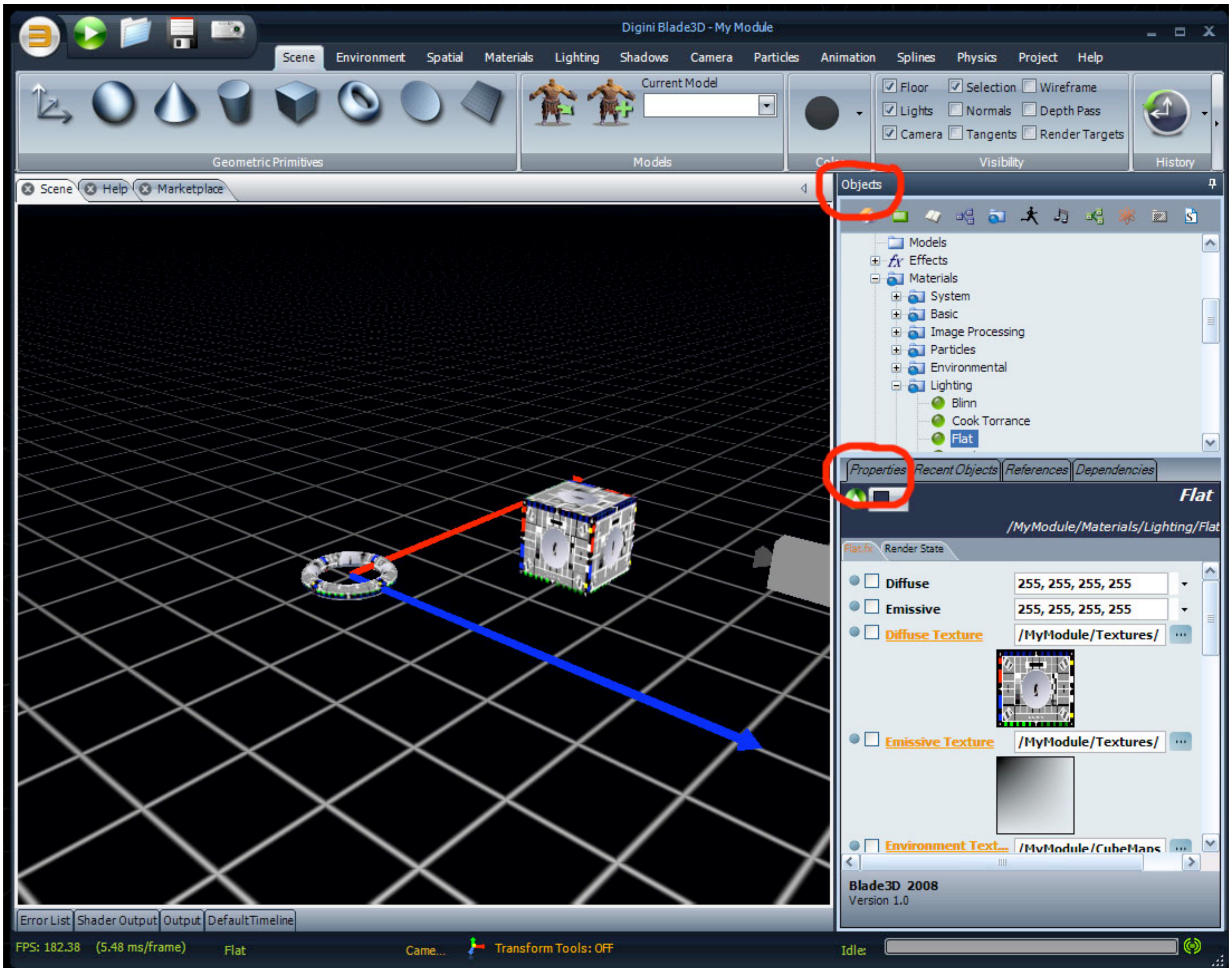
To import textures, go to the "Materials Tab" and click "Import Plain Texture" Icon



Browse all the textures including bump map, specular map, occlusion etc. and click import. Once the textures are imported, they are listed in the object library.

Applying textures to 3d objects

To apply textures to an object, select the object and right click on it. From the option box select "Locate Material". The material information will appear in the properties window.



In the "Properties" window browse for Diffuse, Normal and Specular Textures which are to be applied on the object. The texture can also be scaled using the "UV Scale" option in the properties window.

APPENDIX A: RESEARCH QUESTIONS

Hypothesis 1:
Can a BCI Device be effectively implemented as an active input device in gaming?

The development group would like to determine if BCI devices are the next step in video gaming controllers. While hardware manufacturers have developed small demos to showcase the devices capabilities, few actual games have been constructed. Through *Maxwell's Demon* our group would like to find out BCI devices are comfortable and powerful enough to be used as a main control scheme in a video game.

In *Maxwell's Demon*, the first hypothesis is repeatedly tested via platform manipulation. The player must use the device to change the position of the movable platform. We hope to discover that the length of play has a medical effect on player.

Hypothesis 2:
Do auditory and visual distractions effect BCI communication?

In our game, we present the player with the specific task of exiting a room. In more advanced levels, we present auditory and visual distractions as an attempt to interrupt communication between the player's brain and our software.

Additionally, this information can be useful to determine if the player's real world conditions will affect BCI response.

Hypothesis 3:
Can a BCI device detect frustrations levels?

In the *Whale Room*, we attempt to detect a player's frustration level through the BCI device. This has previously not been tested and is purely exploratory.

APPENDIX B: RESEARCH DOCUMENT

List of Terms

VEP - The visual evoked potential (VEP) is the electrical response of the brain's primary visual cortex to a visual stimulus.

SCP - The slow cortical potential (SCP) is a potential shift in the bioelectrical brain signal. Negative SCPs are typically associated with movement and other functions involving cortical activation, while positive SCPs are usually associated with reduced cortical activation. Negative shifts are usually the electrical response of the brain's primary visual cortex to a visual stimulus.

P300 - The P300 is a positive evoked potential that appears 300 ms after presentation of an attended stimulus embedded in a sequence of irrelevant stimuli. Its spatial signature is a spherically symmetric distribution with the vertex (electrode position Cz) as its center. A typical P300 waveform resembles the shape of a Gaussian distribution with a half-width of 150 ms, and has an amplitude of a few up to the order of 100 microvolts. Typically, a large number of trials needs to be averaged in order to obtain a clear P300 waveform. This fact is usually attributed to a low signal-to-noise ratio of the signal. However, there is no evidence to rule out the alternative view that the P300 originates from a stimulus-related transient phase synchronization of constant-amplitude activity rather than a consistent single-trial amplitude waveform.

SSEP - SSEPs (steady state evoked potentials) occur when sensory stimuli are repetitively delivered at high enough rates so that the relevant neuronal structures are prevented to return to their resting states. Ideally, the discrete frequency components remain constant in amplitude and phase within an infinitely long time period. In practice, SSEPs never completely fulfil this definition of an ideal SSEP. Depending on the modality used one discriminates between SSVEP (steady state visually evoked potentials) and SSSEP (steady state somatosensory evoked potentials).

Event related (de)sychronization - ERD/ERS is the task-related or event-related change in the amplitude of the oscillatory behavior of specific cortical areas within various frequency bands. An amplitude (or power) increase is defined as event-related synchronization while an amplitude (or power) decrease is defined as event-related desynchronization. Like event related potentials, ERD/ERS patterns are associated with sensory processing and motor behavior.

Review of Relevant Research in Brain Computer Interfaces

Defining BCI

Brain Computer Interfaces or BCI can be defined as a new means of communication from the user's brain directly to a receiving device, most commonly a computer application or a robotic prosthetic (Wolpaw et al). While most research in BCI has been in the medical field, the potential for its application in video games as a new form of controller has been gaining popularity.

Types of BCI Devices

There are various ways in which a computer device can read information from the brain. Therefore, different types of BCI devices have been constructed to accomplish this task. Each of these devices has their own advantages and disadvantages. The most common form of recording of brain information is electrical activity. However, other technologies such as MEG (magnetoencephalography), NIRS (near-infrared spectroscopy) [7], and fMRI (functional magnetic resonance imaging) [8] are used as well. MEG and fMRI equipment is large and expensive, which limits the practical application. NIRS is small and affordable, but like fMRI, NIRS is based on hemodynamic responses, and thus the time constants are relatively long. Bioelectrical brain signals recorded by EEG (electroencephalography), ECoG (electrocorticogram) [4, 9, 10], and intracortical electrodes [3, 5, 11-14] are the most common methods of BCI research. Since the EEG is noninvasive and thus readily available it is by far the most often used recording technique [2, 15-21] (Wolpaw et al).

Another accessible and cost effective BCI device is the fNIR or Functional Near InfraRed technology, seen below in Figure 1. Bunce explains its importance in detail,

> "The functional states of tissue, specifically brain tissue, can influence its optical properties through environmental stimuli, blood oxygenation levels, and electrochemical activity.

Functional optical imaging uses these tissue properties to by using light in the range of 700-900 nm to measure physiological changes in the brain. Neural activity uses glucose metabolism as a fuel source, which increases glucose and oxygen consumption among the capillary bed. This reduction of glucose and oxygen increases cerebral blood flow (CBF) and cerebral blood volume (CBV) in the brain known as neurovascular coupling. The CBF carries glucose and oxygen to the reduced area, which is brought later by oxygenated hemoglobin. The oxygenation transported usually exceeds the needed amount, leading to large amounts of cerebral blood oxygenation in active areas of the brain. The oxygenated and deoxygenated hemoglobin have optical properties that are in the visible and near infrared light range. Because of this, when there is a change in brain concentration (e.g. a change in hemoglobin oxygenation level) these molecules can be measured by optical methods that the fNIR employs.

The fNIR device uses a continuous wave system (CW) model, where light is either applied as a continuous or a slow pulse of light on the tissue of the individual's forehead. These systems use less sophisticated detectors and have time issue delays, but can be created to be more compact and less expensive. LEDs are used in the CW model to increase safety. This allows it to be used in any working or educational condition, as well as within the field."

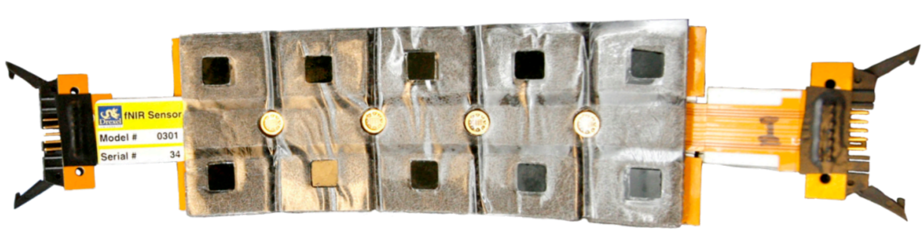


Figure 5. Drexel University's fNIR device.

Data Processing

After information from the brain is gathered it must first be processed before it is usable for its desired application. Signal processing examines patterns in the acquired, raw data. Regardless of the technology used to acquire signals from the brain, specific features must be extracted from these raw signals in order to produce a working BCI. The kinds of patterns that have been employed for this purpose include VEP (visually evoked potentials) [22], SCP (slow cortical potentials) [23], P300 (potential 300 msec) [24], SSVEP [25], dynamics of brain osciallations (Event related synchronization/desynchronization, ERD/ERS) [26, 27], and multiunit spiking patterns [3, 13]. All of these methods have been shown to be useful for implementing practical working BCI systems (Wolpaw et al). Figure 2, below explains the communication flow cycle of a typical brain computer interface loop.

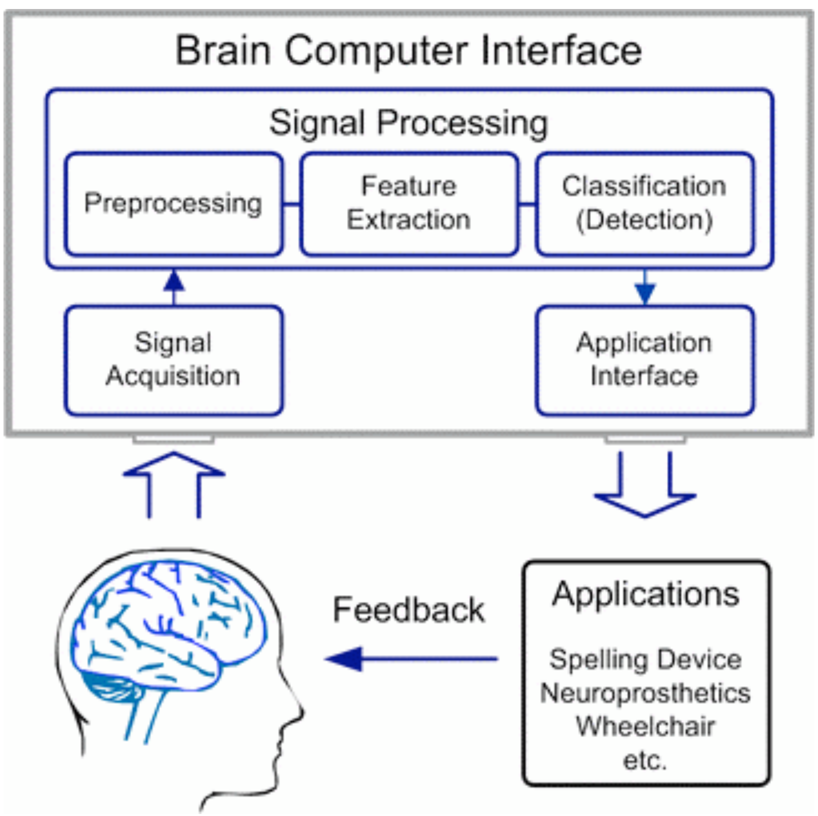


Figure 6. Scheme of a Brain Computer Interface

BCI systems typically operate in one of two modes, synchronous (cue-based or system driven) or asynchronous (self-based or user driven). In a synchronous mode, the system tells the user when to perform a particular mental task that conveys their intent to the system. In an asynchronous mode, the user conveys such intents at will whenever they wish to do so. Clearly this results in an interface, which is far more natural to use, and is more useful, however it is much more technically challenging to develop this type of BCI as opposed to a synchronous BCI. Ultimately, asynchronous BCIs will yield more usable applications, however in the meantime research remains active in both areas since the development of synchronous BCIs continues to develop our understanding of what methods and features are useful for implementing BCIs (Wolpaw et al).

Strategies of Experimentation

In order to get validated results from a BCI system, the user must be able to reproduce stable patterns of activity. Researchers have used a number of strategies to achieve this goal.

Operant conditioning is a strategy in which specific characteristics of the brain signal are monitored and provided to the subject in some form of feedback. Based on this feedback, the subject attempts to control that particular characteristic to produce the desired feedback. For example, the band power in a particular frequency band might be monitored and presented to the subject as a vertical bar indicating the current power. The subject may then attempt to make the bar grow taller or shorter by learning to modulate the particular rhythm being monitored.

Motor imagery is another strategy in which the subject imagines the execution of some muscle movement such as, for example, left and right hand movements. The system then attempts to recognize brain signal components, which differentiate one imagined motion from the other. Since the patterns elicited by motor imagery are associated with cortical areas most directly connected to the brain's normal motor output channels, this strategy seems to be a very suitable experimental strategy for BCIs (Wolpaw et al).

Focused visual attention to a certain item or directing the gaze to a flickering light is another experimental strategy used in BCI research.

BCI and Video Games

For years BCI devices have been used for physiological imaging and monitoring. While research is ongoing in the medical field to make BCI systems faster, more accurate, and less evasive, the video game industry has looked to BCI as a possible controller device. Other than EEG and fNIR, the other brain monitoring technologies are not plausible for use in a consumer product due to their size and expensive nature (Bunce, 2006). However through lower cost, portability, and availability EEG and fNIR-based devices have become a realistic possibility for BCI controllers (Oum 2008).

Two such controllers now on the consumer market are the Emotiv Epoch headset and the Neurosky headsets, seen below in Figure 3 and 4 respectively.

Figure 7. Emotiv Epoch Headset

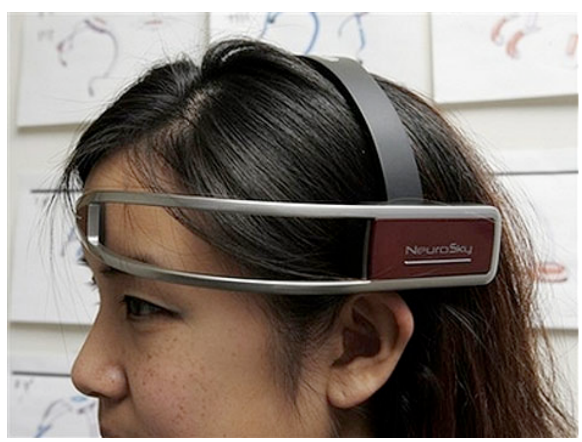


Figure 8. NeuroSky Headset

The Emotiv device boasts the ability to read and interpret 30 separate mental states by reading EEG levels and facial muscle recognition. Priced at only $299, this product has been developed for video game developers looking to create the next generation of game controls at an affordable price. The device also has medical applications, where the headset has been interfaced to a computer controlled wheelchair. The user has the ability to control direction with eye blinks and speed with a smile (Emotiv 2008).

The NeuroSky headset also uses EEG to detect changing brain wave patterns and uses gyroscopic sensors to assist in latency readings of the EEG. NeuroSky is currently marketing its device to both technology developers and consumers. Along with the creation of the NeuroSky headset, they have created two products, which are expected to arrive at store shelves in the upcoming months. The *Force Trainer* is a toy licensed by the Star Wars franchise, that allows the user to “train” their minds by adjusting the speed of a small fan motor using concentration levels and levitating a small ball. Similarly, NeuroSky is releasing *MindFlex*, licensed by Mattel, where the user controls fan speed to move a ball through a physical obstacle course. These toys are set to price around $100 dollars. While their applications may not due much for BCI research, they do represent the first Brain-Computer-Interface device to be presented to the mass market. NeuroSky’s signal processing software reads the user’s “attention” and “meditation” levels which can then be applied to various applications.

While no official games have been released using a BCI device, Drexel University has created a game prototype called *Lazy Brains*. Drexel’s Bio-Medical engineering department developed their own Fnir device, which was then used to interface to the Digital Media department’s prototype. The game was developed using Digini’s Blade3D engine and used biofeedback to have the user raise their levels of concentration in order to levitate objects in the game, see figure 5 below.

Figure 9. Lazy Brains Sewer Task. (Oum 2008).

While the technical stepping-stone was a major accomplishment for both departments, the game itself repeatedly uses the same mechanic. Ken Oum's, a Drexel Digital Media Masters student, thesis project is currently in the works to develop a game that uses BCI as an active controller of game play.

References

Birbaumer, N. "Brain–Computer-Interface Research: Coming of Age." International Federation of Clinical Neurophysiology 117 (2005): 479-483. Elsevier.

Bunce S, Izzetoglu M, Izzetoglu K, Onaral B, Pourrezaei K, (2006). "Functional Near Infrared Spectroscopy: An Emerging Neuroimaging Modality". *IEEE Engineering in Medicine and Biology Magazine, Special issue on Clinical Neuroengineering,* 25(4):54 – 62

Emotiv. Emotiv Home Page, http://www.emotiv.com, site accessed September 2, 2008

Izzetoglu M, Bunce S, Izzetoglu K, Onaral B, Pourrezaei K, (2007). "Functional Brain Imaging Using Near Infrared Technology for Cognitive Activity Assessment". *IEEE Engineering in Medicine and Biology Magazine, Special issue on on the Role of Optical Imaging in Augmented Cognition*, 26(4):38 – 46

Neurosky. Neurosky Home Page. http://www.neurosky.com, site accessed September 2, 2008.

J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain-computer interfaces for communication and control," Clin Neurophysiol, vol. 113, pp. 767-91, 2002.