pd.tvz.hr
POLYTECHNIC & DESIGN Vol. 6, No. 4, 2018.
DOI: 10.19279/TVZ.PD.2018-6-4-15

UNREAL ENGINE 4 ENVIRONMENT WORKFLOW

TIJEK IZRADE OKRUŽENJA U UNREAL ENGINE 4

Damir Mačković[1], Andrija Bernik[2]

[1]Freelance 3D artist
[2]University North – University center Varazdin

Abstract

In order to understand the creation of 3D environments in Unreal Engine 4 it is necessary to study conventional games and their essential elements. It is also necessary to understand both game and level designer's roles in the computer game industry. We have covered the importance of their roles along with games and computer game industry in general which are described in the theoretical part of our paper. During the development of our project the S.S. Infinity, a complete documented workflow had been created and later described in our paper. The documented workflow includes descriptions of creating 3D models, textures, texture packages, lighting, post process effects, animations and creating 3D environment. Our result of the project is a fly-through animation which shows an early stage of game development and it is shown at the end of our conclusion.

Keywords: *3D environment, 3D modeling, games, level design, texturing, Unreal Engine 4*

Sažetak

Kako bi bilo moguće izraditi 3D okruženje za računalnu igru pokretanu Unreal Engineom 4 potrebno je prvo proučiti klasične igre te njihove elemente. Također potrebno je razumjeti industriju računalnih igara te uloge dizajnera igara i level dizajnera. Važnost dizajnera igara i level dizajnera zajedno sa igrama, industrijom računalnih igara opisani su u teorijskom dijelu rada. Za vrijeme rada na projektu S.S. Infinity osmišljen je tijek rada koji je dokumentiran u ovom radu. Dokumentirani tijek rada uključuje opise izrade 3D modela, teksturiranja, pakiranja tekstura za Unreal Engine 4, izrade 3D okruženja u Unreal Engine 4, postavljanje osvjetljenja, post proces efekata i izrade animacije. Izrađeno 3D okruženje je u početku izrađeno za računalnu igru, ali je zbog nedostatka vremena iskorišteno za izradu videoanimacije. Videoanimacija pokazuje rani stadij razvoja računalne igre S.S. Infinity.

Keywords: *3D okruženje, 3D modeliranje, igre, level dizajn, teksturiranje, Unreal Engine 4.*

1. Introduction

1. *Uvod*

Computer games are a medium which integrates many other mediums, and as a result they give the players feeling of full immersion. In order to understand computer games, we should first be familiar with the basic elements of the conventional games. It is essential to understand that the computer games are not that different from conventional games. As opposed to movies or music, their content is interactive. The players perceive the game's world as a new reality. Current 3D computer games are relying on the visual appearance and effects to create immersive 3D environments [1], which are essential in creating a new reality for the players. In game development, level designers are responsible for visual appearance, 3D environments main functionalities and gameplay. They work closely with the producer, art director, and the game designer to implement the game content in a 3D environment.[2] Game development workflow depends on the decisions of the four key people in a game studio; the producer, art director, game designer and the level designer.

The main purpose of our paper is to explain the key elements of games and both level and game designer's roles in creating 3D environments.

We will focus mainly on creating the 3D game environment using the Unreal Engine 4. The same job could bed one with Autodesk Maya as well and we have published those methods in our previous papers and in University textbooks. [3,4] The workflow for creating a 3D environment for a computer game may vary from a person or company's level. Authors have created the workflow during the development of a 3D environment for a PC game S.S. Infinity. The workflow is created to optimize time and simplify the development of the 3D environment. For this purpose have used the normal and bump maps, which allowed us to use high-poly details on low-poly models [5]. For 3D environment we have used Substance Designer and Painter workflow which is also recognized as a industry standard. [6]. Our work seen in this paper can be beneficial to someone who is just starting to learn level design, but it is not a guide on how to create a good 3D environment.

2. Games
2. *Igre*

In this paper, we are showing the method of creation full 3D virtual environment, but before we can discuss the level design and the development of the 3D environments, we should define games and how they work. It is easy to assume that everyone knows what a game is, but there are so many different types of games today that it is best not to make assumptions from personal experiences. To understand games, we have to identify their key elements and then define the games based on those elements. Only when we understand them we are able to understand computer games and their advantages over the conventional games.

Figure 1 *The S.S. Infinity logo*
Slika 1 *S.S. Infinity logo*

3. Unreal engine environment workflow
3. *Unreal engine 4 radno okruženje*

Workflows are an organized approach to developing content. Major development studios use their own well-tried workflows to use time and resources efficiently. During the development of the S.S. Infinity project, as well as its logo which is shown in figure 1., a new workflow had been developed which includes 3D modeling, texturing, the level design in Unreal Engine 4, lighting, post process effects, and animation.

3.1. The S.S. Infinity project
3.1. *Projekt S.S. Infinity*

The S.S. Infinity project was ment to be a first-person 3D adventure, in which the player discoverd the main elements of the cargo spaceship named S.S. Infinity. The S.S. Infinity is an autonomous cargo spaceship which is being used to transport dangerous cargo. The main goal of the game would be to find a way to the control room in order to gain access to ship's helm controls to correct its course. The spaceship has lost its original course during the ion storm, and it is supposed to be headed to a nearby star. The game would be set in four main rooms which form a uniform 3D environment, in which the drones occasionally carry cargo from one room to another. The player would have to gain access to the drones mechanics and figure out his way to the control room using logic. The 3D environment is run by Unreal Engine 4. Even though it was developed for a game, it was only used for a flyover 3D animation assignment at 3D animation course at University North.

3.2. The Software
3.2. *Programski paketi*

Several software packages have been used to create the S.S. Infinity 3D environment. To develop it, it was necessary to become proficient with Autodesk Maya 2016 for creating 3D models, Allegorithmic Substance Painter and Designer for texturing and Unreal Engine 4 to create the final 3D environment and flyover animation.

3.3. Project data structure
3.3. *Struktura projektnih podataka*

The development of a 3D environment is a very demanding task. Good organization of data is very important. The scope of a project can grow very quickly and organized data, as well as well defined naming convention, saves time. Workflow begins with using organizing tools within the used software. Maya 2016 has a great system for defining the project structure which can be modified depending on the requirements of the project. Predefined folder structure was used while using naming convention.

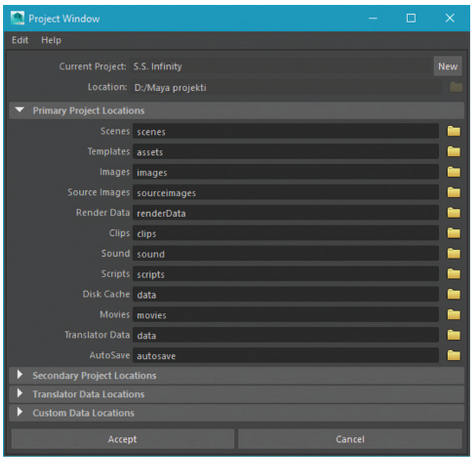


Figure 2 *The Maya 2016 Project window*

Slika 2 *Project window u Maya 2016*

3.4. Making the 3D environment
3.4. *Izrada 3D okruženja*

Various skills were needed in order to create the S.S. Infinity's 3D environment and a flyover animation. Skills like - 3D modeling, texturing, basics of level design, lighting, camera placement and video editing. Proper lighting and the 3D models with proper textures applied to them have a big role in the overall visual appearance of a 3D environment. They are used to establish the 3D environment's atmosphere. Before making the 3D models, it was necessary to create a rough sketch to prototype it. We have chosen to create the prototype using pen and paper because it asures more creative way to do it. The result of our work is shown on figure 3 in 2 dimensions, and in figure 4 we are showing the 3D models of interior design.

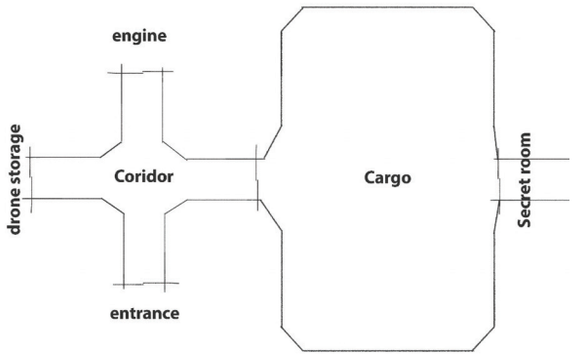


Figure 3 *The Recreated blueprint of the 3D environment*

Slika 3 *Rekreirani prostorni plan 3D okruženja*

3.4.1. Making the 3D models
3.4.1. *Izrada 3D modela*

The 3D models used to create the S.S. Infinity 3D environment were made, as it was previously published using Autodesk Maya 2016. After a prototype sketch had been made, It has been decided to use the modular approach common for the modular level design. It includes using many separate 3D objects that perfectly connect to each other, which together make the whole 3D environment. Just like objects in the real world do. The main approach to modular level design is to create the corner objects first, and then the walls, floors, etc. When all the 3D models are made and have textures applied to them, they can be imported into game engine in order to assemble the final 3D environment. After it is made according to the prototype, next it is necessary to create the props which will make the environment more interesting.

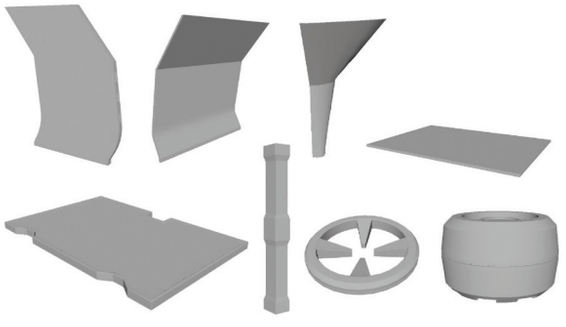


Figure 4 *Several 3D models of the S.S. Infinity 3D environment*

Slika 4 *Pojedini 3D modeli 3D okruženja S.S. Infinity.*

3.4.2. Texturing

3.4.2. *Izrada tekstura*

Since Unreal Engine 4 uses Metallic/roughness PBR shader, Substance Painter and Designer were used for PBR texturing. PBR stands for Physically based rendering. Creating PBR textures is possible using the Substance tools. Although using these tools makes texturing quite faster than before, it still takes a lot of time to texture one 3D model. To reduce the time needed for texturing and implementing 3D models in Unreal Engine, A modified Substance Designer and Painter workflow recommended by Allegorithmic were uesd.

Substance Designer was primarily used to create additional textures for texturing in Substance Painter. In order to create PBR textures in Substance Painter, it is necessary to bake a normal map from a high-poly model. The generated normal map will then be used for generating textures like ambient occlusion, curvature, and others. The game engine doesn't need those textures. They are only used for generating various weathering effects and surface details in Substance Painter.

The first thing that needs to be done when texturing with Substance tools is to bake the normal and curvature map. The process of generating those textures is described using the column 3D model. To import a 3D model in Substance Designer one should right click on the package folder icon and access the link 3D mesh option. After importing the high-poly and low-poly models one should generate the normal map. The normal map is a 2D bitmap image containing the data that tells the engine in which direction should the light bounce of the object. The usual workflow for creating normal map textures is to generate it from the high-poly model and then apply it to the low-poly one. That technique enables rendering fine detail information on a low-poly 3D models.

Generating the normal map from a high-poly mesh is often called baking. Baking is possible both within the Substance Painter and Designer texturing tools. For the S.S. Infinity project Substance Designer was primarily used for texture baking.

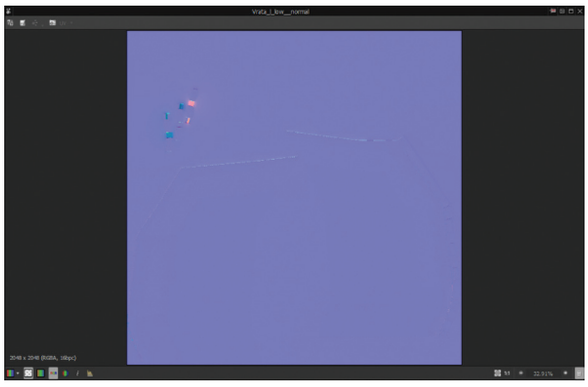


Figure 5 *Finished normal map texture in Substance Designer 5*

Slika 5 *Izrađena normal map tekstura u Substance Designer 5*

When the normal map is baked, the curvature map can be derived from it. To create a curvature map from a normal map in the Substance designer, the 2D texture containing it should be connected to a curvature node. Then, the only thing left is to adjust its intensity.

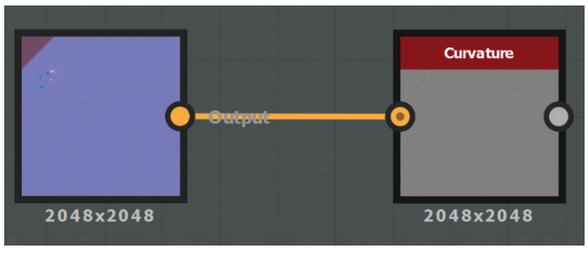


Figure 6 *Creating the curvature texture in Substance Designer 5*

Slika 6 *Izrada curvature teksture u Substance Designer 5*

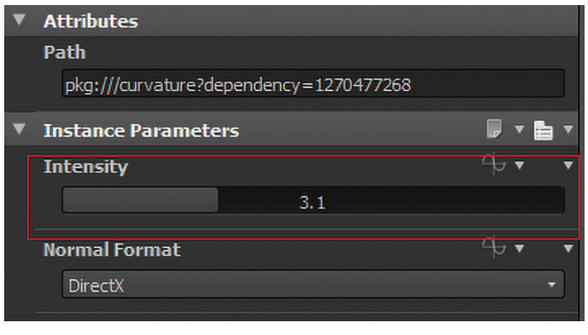


Figure 7 *Setting the curvature intensity*

Slika 7 *Podešavanje intenziteta curvature teksture*

Both normal and curvature map are exported from Substance Designer and imported in Substance Painter while opening a new project. After those were imported into Painter, the other maps were baked using bake texture window.

When all the necessary textures are created it is possible to create textures that will be used by Unreal Engine's PBR shader.

The texturing process was divided into several stages. Those include: creating height details, creating basic materials and creating emissive materials. Texturing in those three stages brought a unique visual style and reduced the time needed to finish the texturing. In figures 8-11. we are showing the results of using high quality texture maps (eg diffuse, normal and bump) to get the high resolution look on low poly 3D model. The method is broadly use din game industry.

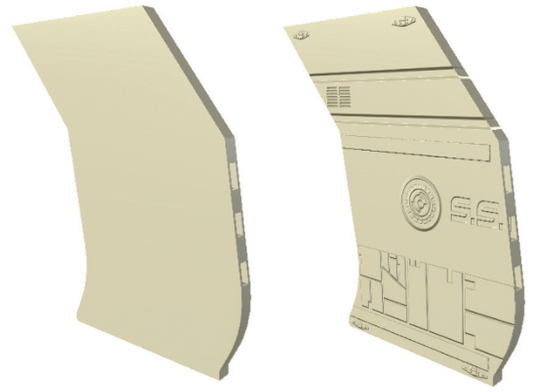


Figure 8 *The left door 3D model without height details (left picture)*

Slika 8 *3D model lijevih vrata bez height detalja (slika lijevo)*

Figure 9 *The left door 3D model with height details (right picture)*

Slika 9 *3D model lijevih vrata sa height detaljima (slika desno)*

Figure 10 *The left door 3D model with all textures applied*

Slika 10 *3D model lijevih vrata sa završenim teksturama*

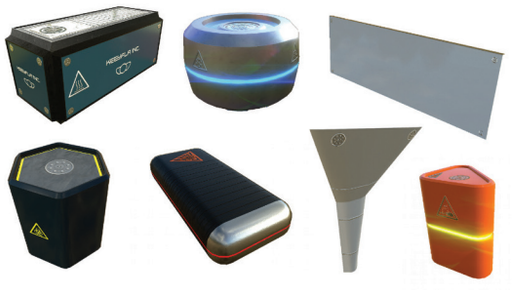


Figure 11 *Several 3D models with finished textures*

Slika 11 *Neki od 3D modela s izrađenim teksturama*

3.4.3. Export and packaging textures for Unreal engine 4

3.4.3. *Izvoz i pakiranje tekstura za Unreal Engine 4*

Unreal Engine supports the Substance plugin which enables managing texture packs created with Substance Designer. Unreal Engine 4 implements similar PBR material graph editor like the one in Substance Designer. That means that it is compatible with the texture outputs from Substance Designer. The Substance plugin for Unreal Engine 4 automatically imports all the required PBR textures from the package and creates the material and its instance. Using this plugin reduces the time needed to import and adjust the textures in Unreal Engine 4. The material in the Unreal Engine 4 is a visual script code that defines how the 3D models look in the 3D environment.

For the substance plugin to work, the user has to import the finished textures made with Substance Painter into Designer and connect their nodes to the appropriate texture outputs. For the texture scaling to work in Unreal Engine material instance, all the outputs should have resolution scaling relations set to *relative to parent*. After doing that, the textures are ready for packaging. Packing of the textures is done by right clicking on the main project folder and clicking on the *export .sbsr file.* Substance Designer will then create the texture package and Unreal Engine material and its instance. The texture scaling would be enabled in the material instance. Since all the textures are procedural, it is easy to dial different texture resolutions if needed and conserve computer resources.

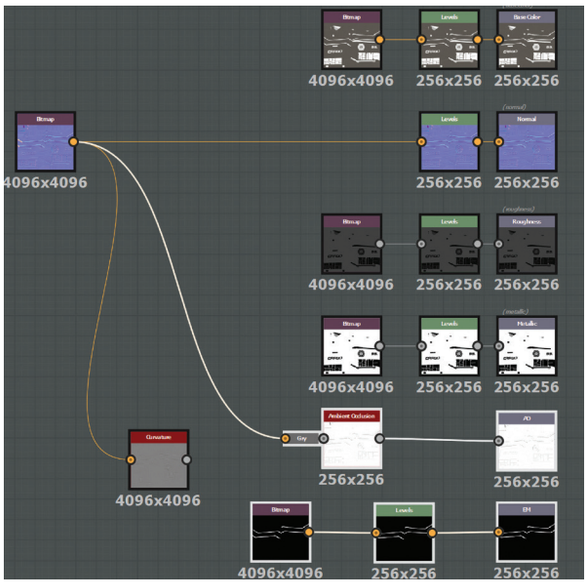


Figure 12 *The node grid in Substance Designer 5*

Slika 12 *Mreža čvorišta u Substance Designer 5*

Creating the textures for the S.S. Infinity project was a big task for us and it had an important role in the workflow. Creating and using a well-organized texture workflow reduced the overall time needed for texturing.

3.4.4. Importing 3D models and textures in Unreal Engine 4

3.4.4. *Uvoz 3D modela i pripadajućih tekstura u Unreal Engine 4*

Unreal Engine 4 has its own content browser which handles the project data organization. It handles all the necessary files for creating 3D environments, like special effects, character animation, 3D models, textures, materials and many more. The content browser automatically recognizes the file type on import and displays appropriate dialogs. Once the files are imported the user is able to search, reload or move the files with ease.

Importing the 3D models and textures had been done in several steps. 3D models were imported into the folder named meshes. Importing the fbx 3D models can be done by clicking the import button in the content browser, and then locating them on the hard drive. The import dialog will pop up and after clicking on the import all button, all the 3D models with their basic material slots will be imported.

The same steps were done while importing the texture packs. After all the 3D models and their textures were imported, it was necessary to check if there were any additional textures needed to be enabled in the material instances.

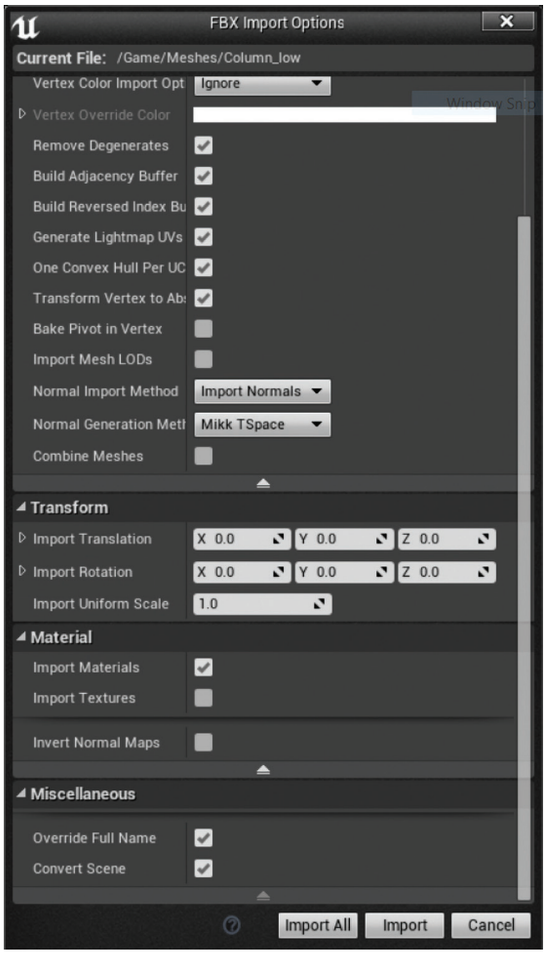


Figure 13 *Importing the 3D models in content browser*

Slika 13 *Uvoz 3D modela u content browseru*

Figure 14 *The 3D model with all textures*

Slika 14 *3D model sa svim teksturama*

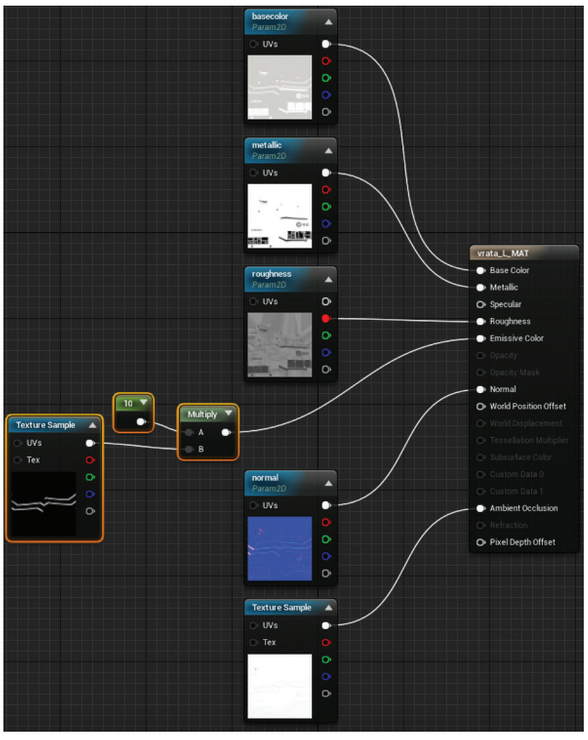


Figure 15 *The Finished material in material editor*

Slika 15 *Izrađen materijal u material editoru*

3.4.5. Making the corridor and the cargo hold in Unreal Engine 4

3.4.5. *Izrada koridora i skladišta u Unreal Engine 4*

Once all the assets were imported and organized, the creation of the 3D environment, the corridor, and the cargo hold could start. The corridor was created using the appropriate 3D models and move and rotation tools inside Unreal Engine 4. The door 3D models were placed at the beginning of the corridor, while the corner pieces 3D models were placed at the end.

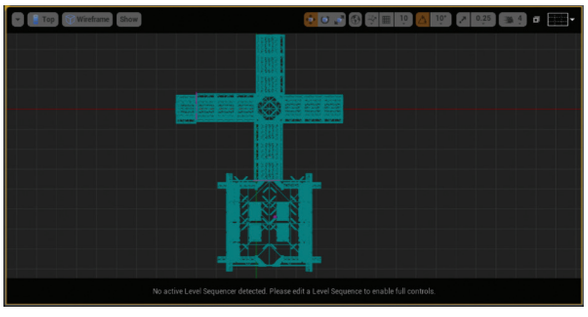


Figure 16 *The top view Wireframe of the S.S. Infinity 3D environment*

Slika 16 *Žičani prikaz gornjeg prikaza S.S. Inifinity 3D okruženja*

The doors would later be animated with the Unreal Engines sequencer. Synchronizing the grid and the units in Maya to the ones in the Unreal engine 4 came handy when snapping the 3D models to each other in order to prevent light leaks. The cargo hold was created with the similar approach.

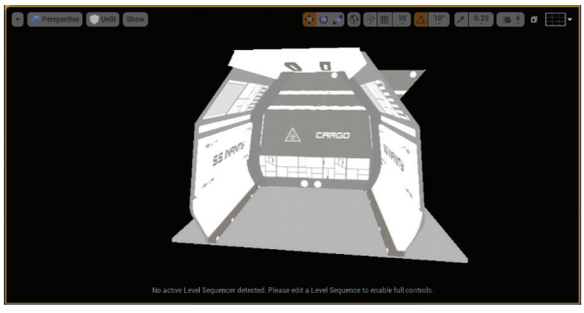


Figure 17 *The unlit corridor module*

Slika 17 *Neosvjetljen modul koridora*

3.4.6. Lighting

3.4.6. *Osvjetljenje*

Lighting in Unreal Engine 4 is made to conserve the computer resources as much as possible, since it is the most demanding to compute in real time. Most games use lightmaps that are applied to the 3D models at runtime. Lightmaps contain precomputed light and shadow information that would otherwise have to be computed on every frame. There are still situations that require the use of dynamic lights, and Unreal Engine enables the use of it.

Since at the time the 3D environment was going to be used only for a flyover animation, Dynamic lighting was used. To light the 3D environment the Nvidia VXGI indirect dynamic lighting that is not regularly shipped with Unreal Engine was used. Using the VXGI made lighting easier and less time-consuming. VXGI has a big role in the overall visual look of the 3D environment. It simulates the real world indirect lighting using voxels. Voxel is basically a 3D pixel. Indirect lighting is a light bounced off the objects in the environment. It is very computer resources expensive to calculate it in real time, so it is not normally implemented in the real-time engines. Indirect illumination can also be achieved using lightmaps.

VXGI simulates the indirect illumination using cone samples placed throughout the 3D environment and placing the voxels where the indirect illumination should occur. Using the indirect illumination is adding to the realism since it shows all the correct light information, even the one bounced off the reflective surfaces.

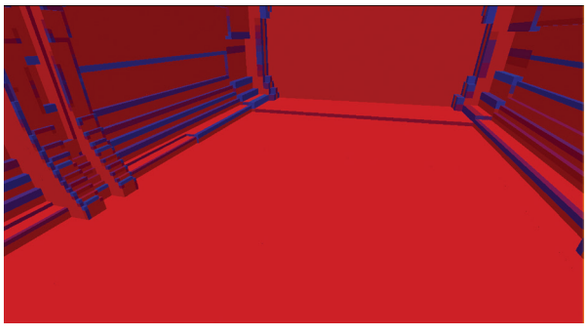


Figure 18 *The Voxelised 3D environment*
Slika 18 *Voxelizirano 3D okruženje*

3.4.6.1. Reflection and post proces effects
3.4.6.1. *Refleksije i post proces efekti*

To be able to use the post process effects, there should be a post process volume placed in the 3D environment. It is located in the volumes tab and can be dragged into the scene. Reflections come second after the lighting when it comes to consuming computer resources at runtime. They have to be contained inside reflection probes. This can be done by placing the "reflection captures" which record the scene in six directions and display it at reflective surfaces. There are spherical and cubic reflection captures inside Unreal Engine. To create a reflection environment for the S.S. Infinity both sphere and cubic reflection captures were used. The cube reflection capture was placed to encompass the whole 3D environment. The sphere one could be used for the same purpose but it wouldn't make a big difference since off using the VXGI.
Color grading was enabled in the post process volumes settings. Color grading can be done using the *LUT table files* made in Adobe Photoshop. LUT was made from the processed screenshots of the 3D environment and imported back to the LUT slot on the post process volume settings. Post process volume will then alter every frame according the LUT file. The goal of using the color grading was to achieve the filmic look.

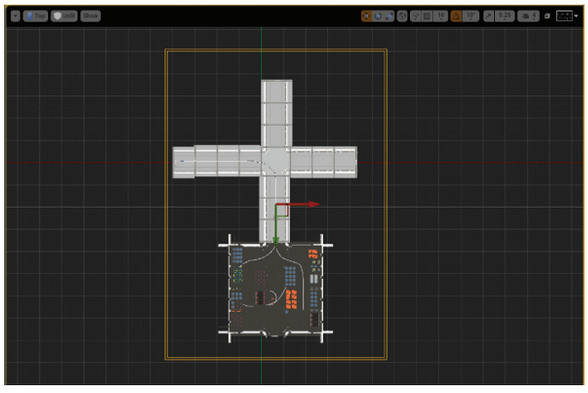


Figure 19 *The Polygonal reflection capture*
Slika 19 *Poligonalna zona hvatanja refleksija*

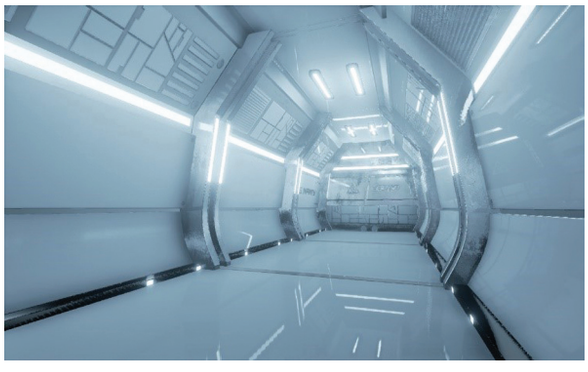


Figure 20 *The 3D environment without color grading*
Slika 20 *Prikaz 3D okruženja bez gradacije boja*

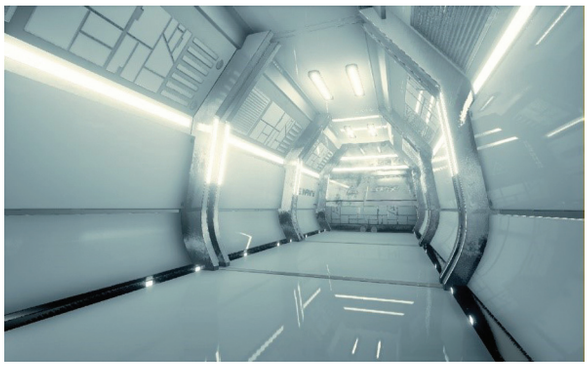


Figure 21 *The 3D environment with color grading*
Slika 21 *Prikaz 3D okruženja s gradacijom boja*

Lighting the 3D environment and adding the post-process effects marked the end of creating the S.S. Infinity 3D environment. The 3D environment was ready for further development of the S.S. Infinity PC game. To create a playable version of this 3D environment a lot of work was needed so at this stage it was only used to crate the flyover animation.

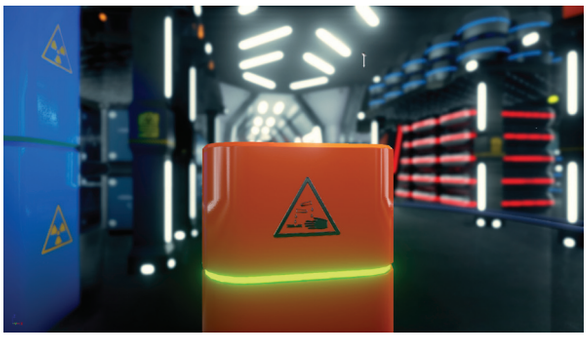


Figure 22 *The S.S. Infinity 3D environment screen capture*

Slika 22 *Snimak zaslona skladišta u S.S. Infinity 3D okruženju*

4. Creating the flyover animation

4. *Izrada animacije*

Animating objects can be done in Unreal Engines sequencer. It is a powerful tool which can be used to animate objects in almost the same way as with the dedicated software for animation like Maya. It can record animation and export the frames for editing with a dedicated video editing software. The Adobe Premiere was used to edit the flyover video animation.

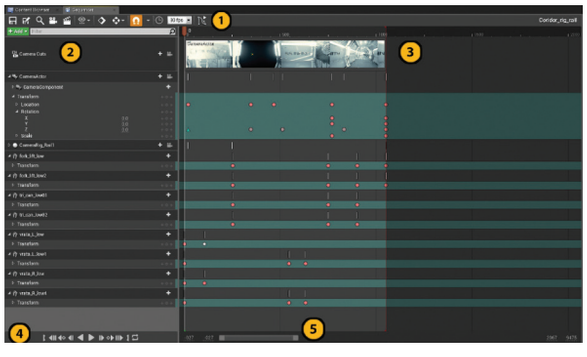


Figure 23 *The Unreal Engine 4 Sequencer*

Slika 23 *Unreal Engine 4 Sequencer*

4.1. Camera and 3D model animation

4.1. *Animacija kamere i 3D modela*

Camera animation in Unreal Engine 4 is done by using the camera rig rail object. It acts as the real camera rails. It is accessible in modes section of the Unreal Engine 4 user interface by cinematics tab. Camera rig rail has to have a camera attached which is also accessible on the cinematics tab. Attaching the camera to the rig rails can be done in more than one way.

The one that was used was by right clicking on the camera object in the scene and clicking on the attach > CameraRig_Rail. After attaching the camera to the rig rail it was necessary to draw its path in the 3D environment. The Rig rail object is a vector curve that the camera follows.

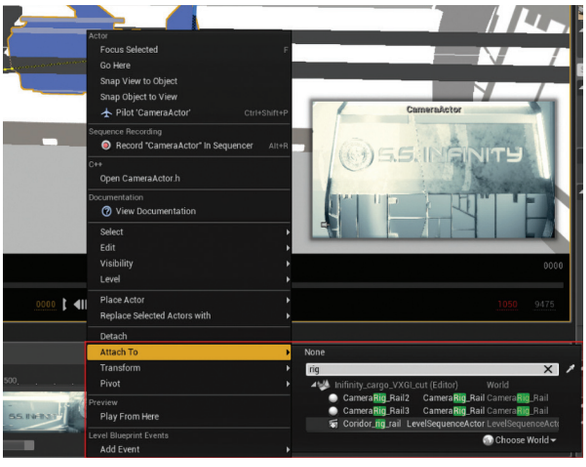


Figure 24 *Attaching the camera to the rig rail*

Slika 24 *Kamere s rig rail objektom*

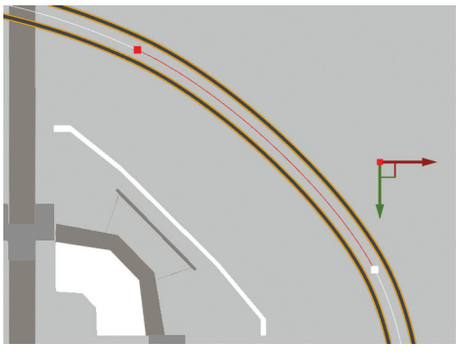


Figure 25 *The Rig rail object*

Slika 25 *Rig rail objekt*

The keyframing technique was used to animate the 3D models like the doors or the drones. As an example, the doors were animated by placing a keyframe at frame 0 at the initial location. A few frames after the doors were moved to an open position the keyframe was added. All the completed shots were exported as png files per frame at 1920x1080px.

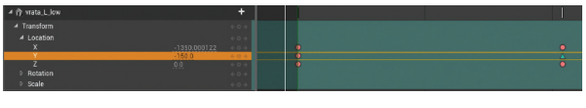


Figure 26 *Keyframing left door 3D models*

Slika 26 *Postavljanje keyframeova za 3D modele lijevih vrata*

4.2. Editing the video file
4.2. *Editiranje video datoteke*

The Adobe Premiere was used to edit the flyover video animation. To edit the video in Adobe Premiere, the user has to create a new project and a sequence and import the content. All the exported frames from Unreal Engine 4 were imported into Premieres sequence and edited by the shots in order. Music and the beginning and end titles were added at the end. The video animation was exported using MPEG-4 container with h.264 codec which enables the maximum image quality with minimum file size. Export dialog is accessible in by the *File > export media tab.*

Figure 27 *Finished video in Adobe Premiere CC*

Slika 27 *Izrađena montaža videa u Adobe Premiere CC*

5. Conclusion
5. *Zaključak*

We have found out that the development of the 3D environment for the computer game is a big task. In order to successfully finish it on time, a new workflow had to created. The biggest part of our workflow was the texturing of the 3D models. Substance tools were extremely useful for that task. The biggest advantage of using those tools is their procedural workflows which enable the creation of the unique PBR textures. We would argue that a great imagination is needed when texturing the 3D models, and with the proficiency of using the Substance tools that task is a joy. Unreal Engine 4 is the best free game engine on the market which can run the most realistic 3D environments. Its advantage over the other game engines is in enabling the users to create complex shader networks with zero conventional coding.

The end result of the S.S. Infinity project was the flyover animation that shows the early stage of the S.S. Infinity game development. Documented workflow can be useful to someone who is trying to learn about the level design, but it is not a guide on how to create a good 3D environment for a computer game.

The original flyover animation of our project is accessible via the link: vimeo.com/197947676

A few months after publishing the original undergraduate thesis, the development of the S.S. Infinity was continued.

This video is showing the current stage of its development and it is accessible via the link: vimeo.com/232176258 .

6. REFERENCE
6. *REFERENCES*

[1.] Bernik, A., & Sabati, Z., Generating Terrain And Hi Details Using Texture Maps, Tiskarstvo & Dizajn 2013, 75-81, 2013., ISBN: 978-953-7064-20-4

[2.] Adams, Ernest. Fundamentals of Game Design, 3nd Edition. Berkley, SAD: New Riders, 2014., ISBN: 978-0-321-92967-9

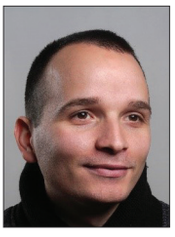
[3.] Bernik, A., Sabati, Z., & Raljević, L., Autodesk Maya – Modeliranje korištenjem poligona. Polytechnic and design, 4(1), 15-29, 2016., DOI: 10.19279/TVZ.PD.2016-4-1-03

[4.] Vusić, D., Sabati, Z., & Bernik, A. (2015). 3D modeliranje u primjerima 1. Sveučilište sjever, Varaždin, ISBN: 978-953-7809-26-3

[5.] Bernik, A., Sabati, Z., & Naranđa, N. (2015). Metode i tehnike naprednog teksturiranja. Polytechnic and design, 3(3), 316-325., DOI: 10.19279/TVZ.PD.2015-3-3-09

[6.] Wes McDermott: The PBR Guide: A Handbook for Physically Based Rendering, 3nd Edition, Allegorithmic, 2018., ISBN: 978-2490071005

AUTORI · *AUTHORS*



Damir Mačković
He completed his professional bachelor's (baccalaureus/ baccalaurea) degree in Multimedia and Graphic Technlogies at University North in Varaždin 2017. During his studies he intensively studied the 3D content creation. He worked his internship as a 3D environment artist at Lion Game Lion AAA studio. After graduating he continues working on 3D content creation and works as a freelance 3D artist.

Andrija Bernik - The unchanged biography is in the journal Polytechnic & Design
Vol. 5, No. 2, 2017.

Correspondence
abernik@unin.hr