Pessebre Virtual. Disseny i exportació

Memòria del projecte de final de carrera corresponent als estudis d'Enginyeria Superior en Informàtica presentat per Toni Bullon i dirigit per Enric Martí.

Bellaterra, Juliol de 2009

El firmant, Enric Martí Godia, professor del Departament de Ciències de la Computació de la Universitat Autònoma de Barcelona

CERTIFICA:

Que la present memòria ha estat realitzada sota la seva direcció per Toni Bullón Clusella

Bellaterra, Juliol de 2009

Firmat: Enric Martí Godia

Agraïments

Aquest projecte final de carrera ha estat possible realitzar-lo gràcies al suport del meu director de projecte Enric Martí, que sempre ha estat a la meva disposició quan l'he necessitat, ja fos en horari de feina com fora d'horari. Ha ajudat en tot el que ha estat a les seves mans, fins i tot proporcionant material que després ha resultat indispensable.

Li voldria agrair especialment el fet d'haver-me donat la oportunitat de realitzar un projecte tant relacionat amb el disseny gràfic i el món tridimensional, ja que no és molt habitual trobar projectes d'aquest estil per a que els puguem realitzar els alumnes com a Projecte Final de Carrera. També ha estat molt important per a mi tenir total llibertat a l'hora de crear el Diorama, demostrant una total confiança en la meva persona.

Cal destacar també la col·laboració en tot moment de l'Associació de Pessebristes de Sabadell per haver-me deixat accedir a tot allò que els hi he demanat. Tot i ser simplement els destinataris o clients d'aquest projecte s'han involucrat en la meva feina donant-me totes les comoditats possibles per dur a terme la visualització del vídeo final en l'exposició de pessebres de Sabadell.

Per últim m'agradaria donar les gràcies al Ferran Poveda per ensenyar-me i ajudar-me a fer l'escaneig de les figures en les seves hores de feina. Sense la seva ajuda i l'escàner de 3D el Pessebre no hauria estat el mateix, hauria requerit moltes més hores de temps de modelat, fins i tot setmanes o mesos.

Resum

Aquest projecte és una part d'un projecte més ampli consistent en estudiar un format gràfic que permeti exportar una escena modelada en Blender i importar aquesta mateixa escena en un entorn interactiu basat en Visual C++ amb OpenGL. D'aquesta forma, disposem de la capacitat de modelat de Blender i de la interacció i visualització de la llibreria OpenGL. Aquest format ha de representar geometria i textures imprescindiblement, i si és possible, d'altres factors importants com il·luminació, visualització i moviment.

La part del projecte explicada en aquesta memòria consisteix en estudiar el format gràfic més adient per representar els diferents factors de realisme de l'escena (geometria, textura, etc.) havent triat el format OBJ per la seva capacitat de representació i fàcil edició. Per a provar el format, s'ha dissenyat un diorama de pessebre utilitzant les capacitats de modelatge de Blender. Pel que respecta les figures, aspecte important per a considerar l'escena com a pessebre, s'ha utilitzat un escàner 3D que ha obtingut representacions de malla 3D, a partir de figures reals de pessebre, que posteriorment han estat texturades.

S'ha generat un vídeo del diorama de pessebre que permet veure'n tots els detalls navegant amb el punt de vista per l'escena. Aquest vídeo s'ha exposat en la mostra de pessebres de la Associació Pessebrista de Sabadell el Nadal del 2008.

Resumen

Este proyecto es una de las dos partes de un proyecto más amplio que consistente en un formato gráfico que permita exportar una escena modelada en Blender y importar esta misma escena a un entorno interactivo basado en Visual C++ con OpenGL. De este modo, disponemos de la capacidad del modelado de Blender y de la interactividad y visualización de la librería OpenGL. Este formato debe representar geometría y texturas imprescindiblemente, y si es posible, otros factores importantes como iluminación, visualización y movimiento.

La parte del proyecto explicada en esta memoria consiste en estudiar el formato gráfico más idóneo para representar los diferentes factores de realismo de la escena (geometría, textura, etc.) habiendo escogido el formato OBJ por su capacidad de representación y fácil edición. Para probar el formato se ha diseñado un diorama de belén usando las capacidades de modelado de Blender. Por lo que a las figuras se refiere, aspecto muy importante para conseguir la escena como un belén, se ha usado un escáner 3D que ha obtenido representaciones de malla 3D, a partir de figuras reales de belén, que posteriormente han sido texturizadas.

Se ha generado un vídeo del diorama del belén que permite ver todos los detalles navegando con el punto de vista por la escena. Este vídeo ha sido expuesto en la muestra de belenes de la Asociación Belenista de Sabadell la Navidad de 2008.

Abstract

This project is one of the two parts of a larger project that consists in a graphical format that allows to export a scene modeled in Blender and import this same scene in an interactive environment based on Visual C + + with OpenGL. Thus, we have the capability of modeling of Blender and the interactivity and visualization of the OpenGL library. This format should represent essential geometry and textures, and if possible, other important factors such as illumination, visualization and movement.

The part of the project described in this report is to study the most appropriate graphical format to represent the different factors of realism of the scene (geometry, texture, etc.). OBJ format have been chosen for his ability to represent and easy editing. To test the format has been designed diorama of a scene using the modeling capabilities of Blender. As far as figures are concerned, very important aspect to the scene as a crib, has used a 3D scanner that has received representations from 3D mesh, based on actual figures of Bethlehem, which have subsequently been texturized.

A video of the scene has been made where you can see all the details with the sailing point of view for the scene. This video has been exhibited in the sample of scenes from the Bethlehemist Association of Sabadell on Christmas 2008.

Índex

1 Introducció

Aquest projecte s'emmarca dins de les aplicacions gràfiques i més concretament en l'apartat virtual. Les representacions gràfiques han estat evolucionant molt ràpidament, estant cada dia més a l'abast de l'usuari tradicional, podent crear un món similar al que ens envolta. A aquest realisme visual, l'acompanya el factor d'interacció per part de l'usuari amb l'escenari, el qual avui dia, ja no es regeix únicament al àmbit dels videojocs, sinó que el podem trobar en moltes altres aplicacions gràfiques.

La simulació d'ambients 3D per ordinador amb els que l'usuari pot interaccionar en temps real i de forma autònoma és avui dia una possibilitat a l'abast de tots.

El realisme gràfic interactiu permet a l'usuari, com espectador, formar del escenari on es mou. Com editor, permet formar-se el seu propi entorn. Per una banda es busca la facilitat de creació de nous escenaris, cada vegada més similars als reals. I per altre banda, la interacció del usuari en ells.

El projecte final de carrera "Pessebre Virtual" consisteix en l'anàlisi i desenvolupament d'una aplicació que ens permeti d'una manera senzilla, poder realitzar l'edició i visualització d'un Diorama de Pessebre en tres dimensions, en el nostre cas, un pessebre on es pugui navegar mitjançant l'ús d'un ratolí i un teclat, tenint així la capacitat de moure'ns per l'escena i poder contemplar minuciosament, i des de l'angle desitjat, detalls que en un Diorama tradicional ens costaria contemplar o simplement no podríem. Ademés, ha de donar-nos la sensació de formar part del mateix pessebre.

Per una banda es vol fer servir el programari de codi obert i gratuït anomenat *Blender* per a realitzar el que seria tot el modelat i exportació del escenari. Amb aquesta eina es poden aconseguir grans resultats pel que fa al modelat i texturitzat 3D, però té baixa capacitat a l'hora d'aplicar interacció.

Per l'altre banda es treballarà amb l'entorn gràfic de Visual C++[1] amb OpenGL[2], utilitzat a l'assignatura de Gràfics per Computador II. En aquest cas hi ha una alta capacitat de crear interacció i una baixa capacitat per modelar una escena. Per tant la complementació d'aquests dos pot donar bons resultats i aconseguir la mateixa impressió visual en Blender que en l'entorn de Visual C++ amb OpenGL.

A partir d'aquí el projecte es divideix en dues parts o dos projectes complementaris: la part de disseny i exportació a un format gràfic, i la part de importació del format gràfic i interacció.

Aquesta memòria fa referència a la primera part, disseny i exportació del escenari. Com es pot apreciar a la Figura 1, hi ha una part estrictament relacionada entre els dos projectes, que és la del format de transport i les proves que comporta. Per aquest motiu, en alguns casos es parlarà de conceptes relacionats amb l'altre projecte, però sempre conceptes que afectin la part del disseny, modelat i l'exportació.

[1] *Visual C++ (també conegut com MSVC, Microsoft Visual C++) és un entorn de desenvolupament integrat (IDE) per llenguatges de programació C, C++ i C++/CLI. Està especialment dissenyada per el desenvolupament i depuració de codi escrit per les API's de Microsoft Windows, DirectX i la tecnologia Microsoft .NET Framework.*

[2] *OpenGL (Open Graphics Library) és una especificació estàndard que defineix una API multillenguatge i multiplataforma per escriure aplicacions que produeixin gràfics 2D y 3D.*

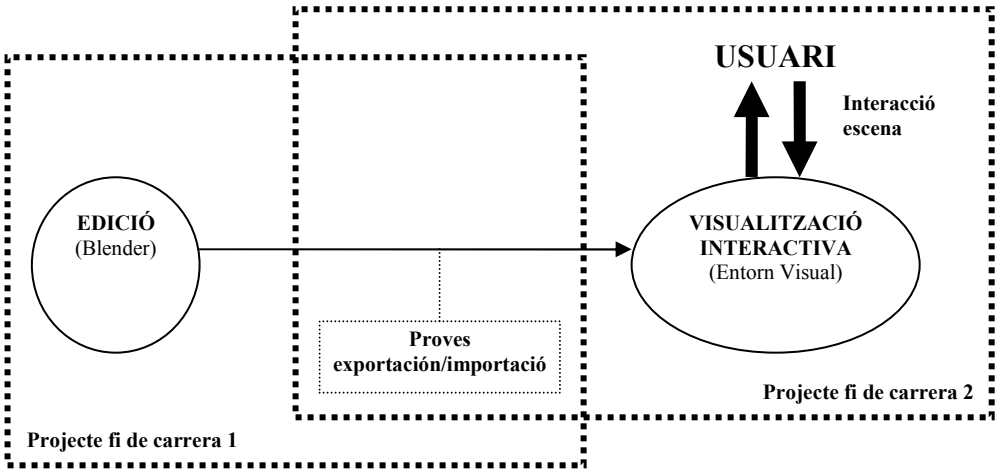


Figura 1. Esquema de la distribució dels projectes.

Des de l'associació de pessebristes de Sabadell va sorgir la idea d'utilitzar les noves tecnologies en l'art del pessebre, dins de l'exposició anual de pessebres de Sabadell, un diorama que no fos el típic pessebre construït a ma, sinó una representació virtual d'un pessebre. D'aquesta manera s'introduiria, en una tradició de casa nostra totalment artesana, el disseny 3D i les noves tecnologies, cosa que fins a dia d'avui, nosaltres no hi tenim constància.

1.1 Què és un Pessebre?

El pessebre és la representació del naixement del nen Jesús construïda amb molsa, troncs, suro, i figures de fang de personatges de la vida rural. El pessebre és un dels quadres costumistes més estesos a la Mediterrània cristiana. El seu origen podria estar en les antigues creences de tenir imatges dels déus a casa com a protecció, encara que ha sofert tantes evolucions que això no es pot assegurar. A la Figura 2 es mostra el que coneixem com a pessebre.

A la societat tradicional catalana el pessebre era representatiu i emblemàtic de les festes, molt abans de que l'arbre de Nadal fos introduït. Tot i ser una tradició que comparteixen molts països, és aquí on se li atorga més importància i s'hi dedica més temps a la seva elaboració.

(a)

(b)

Figura 2. Pessebre tradicional (a). Diorama tradicional (b).

La primera representació catalana apareix a començaments del segle IV a la catedral de Barcelona. També les representacions cristianes que apareixen en forma de relleus als sarcòfags romans del segle VI que són escenes del naixement, amb els Reis, alguns animals i pastors, esculpides a la pedra. A finals del segle XIII els personatges adquireixen autonomia pròpia i passen a formar quadres dins dels temples. Va ser l'any 1223 que Sant Francesc d'Assís va fer una representació del naixement al voltant d'una menjadora d'animals –un estable semblant a aquell en què segons el cristianisme va néixer el nen Jesús–, considerada per molts l'origen del pessebre.

Al segle XVI, les autoritats catòliques més tradicionals van voler fer front a la Reforma protestant amb el Concili de Trento impulsant cultes nous, com ara una rememoració del naixement en una part de la casa. El pessebre més antic del qual es té coneixement es va fer a Praga l'any 1562.

A Catalunya, les primeres figures no arriben fins al segle XVII, però al segle XVIII ja hi ha notables escultors com el murcià Salzillo, els catalans Vallmitjana i Amedeu o els valencians Josep Esteve Bonet i Josep Ginés els que crearien imatges que posteriorment servirien com a model per a altres representacions [Pes].

A dia d'avui, a part de moltes famílies que construeixen pessebres a casa seva, hi ha una gran quantitat de pessebristes professionals i no professionals que dediquen moltes hores a fer diorames i maquetes de pessebres a diferents escales per què puguin ser exposats i la gent tingui la oportunitat de veure'ls.

Són moltes les associacions arreu de Catalunya que cada any s'endinsen en aquest art per mantenir i gaudir d'una de les tradicions més importants que es duen a terme durant la fi i entrada d'any.

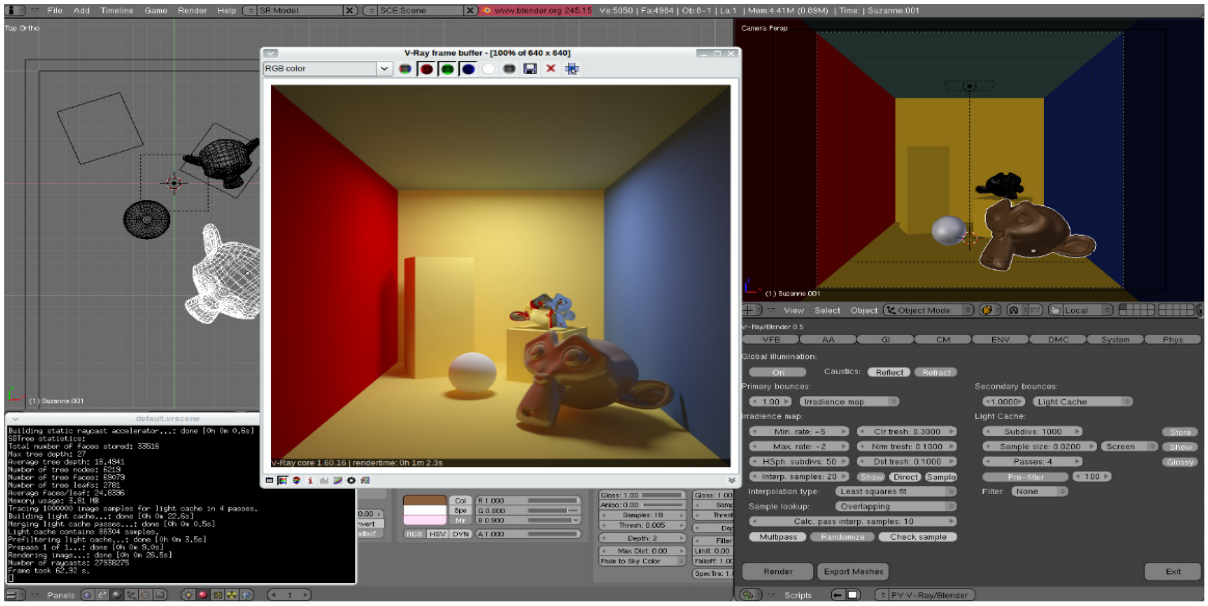
Tot i la importància del naixement de Crist i les diferents escenes bíbliques que durant tota la història s'han anat representant en els pessebres, és cada dia més notable. L'aparició d'escenes modernes o no ambientades en la religió, sent moltes vegades escenes rurals o fins hi tot actuals. Això sí, sempre amb algun rerefons o relació amb el que mana la tradició. A la Figura 3 podem veure un exemple del que seria un diorama de pessebre amb aires actuals.

Figura 3. Diorama de pessebre amb aspecte més modern del que marca la tradició.

1.2 Blender i Blender Foundation

Blender [Pbf] és un programa multiplataforma, dedicat especialment al modelat i creació de gràfics tridimensionals. El programa va ser inicialment distribuït de manera gratuïta, però sense el codi font, amb un manual disponible a la venda, encara que posteriorment va passar a ser software lliure. Actualment és compatible amb totes les versions de Windows, Mac OS X, Linux, Solaris, FreeBSD i IRIX.

Té una molt peculiar interfície gràfica d'usuari, que es critica com a poc intuïtiva, doncs no es basa en el sistema clàssic de finestres; però té, per altra banda, avantatges importants sobre aquestes, com la configuració personalitzada de la distribució dels menús i vistes de càmera. La Figura 4 mostra un parell d'imatges de la interfície de Blender.



(a)

(b)

Figura 4. Distrubució per defecte (a). Distribució per modelar (b).

En 1988, Ton Roosendaal (Figura 5) va co-fundar l'estudi d'animació holandès NeoGeo. Que ràpidament es va convertir en l'estudi més gran d'animació 3D a Holanda i en una de les més destacades empreses d'animació 3D a Europa. L'empresa va crear produccions per a grans clients corporatius tals com la companyia multinacional d'electrònica Philips, que van ser premiades (European Corporate Video Awards de 1993 i 1995). A NeoGeo, Ton Roosendaal fou el responsable tant de la direcció artística com del desenvolupament intern del software. Després d'una acurada deliberació, va decidir que l'eina actual 3D utilitzada en l'estudi era massa vella i voluminosa de mantenir i actualitzar i necessitava ser reescrita des del principi. En 1995, aquesta reescriptura va començar i estava destinat a convertir-se en el software de creació 3D que ara coneixem com Blender. Mentre NeoGeo continuava millorant Blender, Roosendaal se'n va adonar que Blender podria ser utilitzat com una eina per a altres artistes fora de l'estudi.

Figura 5. Ton Roosendaal. Propietari de Blender Foundation

En 1998, va decidir crear un nova companyia anomenada Not a Number (NaN) derivada de NeoGeo per desenvolupar Blender. En la base de NaN, estava el desig de crear i distribuir gratuïtament una suite de creació 3D compacta i multiplataforma. En aquell moment, això va ser un concepte revolucionari ja que la majoria dels programes comercials de modelat valien milers de dòlars. NaN esperava aconseguir una eina de modelat i animació d'un nivell professional per al públic en general. El model de negoci de NaN consistia en proporcionar productes comercials i serveis al voltant de Blender. En 1999, NaN va assistir a la seva primera conferència en el Siggraph amb un gran esforç per promocionar Blender. La primera convenció del Siggraph per Blender en 1999 va ser un autèntic èxit i va provocar un enorme interès tant de la premsa com dels assistents a la convenció. Blender va ser un gran èxit i es va confirmar el seu increïble potencial.

A lèxit aconseguit al Siggraph, a principis de l'any 2000, NaN va aconseguir un finançament de 4,5 milions d'euros procedents a uns inversors. Aquesta gran aportació de diners va permetre a NaN expandir ràpidament les seves operacions. Aviat NaN va arribar als més de 50 treballadors treballant al voltant del món intentant millorar i promocionar Blender. A l'estiu del 2000, Blender 2.0 va ser publicat. Aquesta versió integrava un motor

de jocs a la suite 3D. Al final del 2000, el número d'usuaris registrats al web de NaN va sobrepassar els 250.000.

Malauradament, les ambicions i oportunitats de NaN no van coincidir amb les capacitats de la companyia ni amb la realitat del mercat de l'època. Aquest sobredimensionament de l'empresa va conduir a una reestructuració creant una companyia (NaN) més petita i amb nous fons provinents dels inversors. Sis mesos més tard, el primer producte comercial de NaN, Blender Publisher va ser llençat. Aquest producte va ser dirigit cap a l'emergent mercat de medis interactius en 3D basats en entorns web. Degut al fracàs en les vendes i al continuat clima de dificultats econòmiques, els nous inversors van decidir donar fi a les activitats de NaN. Això també incloïa parar el desenvolupament de Blender. Si bé existien clarament defectes en la versió de Blender, amb una arquitectura interna del software complexa, característiques inacabades i una IGU no molt comuna, la magnífica ajuda de la comunitat i els clients que havien comprat Blender Publisher en el passat va provocar que Roosendaal no pogués permetre que Blender desaparegués. Com rellançar una nova companyia amb un equip suficientment gran de desenvolupadors no era factible, al març de 2002, Ton Roosendaal va fundar l'organització no lucrativa Blender Foundation (Fundació Blender).

El primer objectiu de la Fundació Blender va ser trobar una manera de continuar el desenvolupament i la promoció de Blender com un projecte de codi obert basat en la comunitat de usuaris. Al juliol de 2002, Ton Roosendaal va aconseguir obtenir dels inversors de NaN un "sí" per a que la Funcdació Blender portés a cap el seu pla de que Blender fos de codi obert. La campanya de "Alliberin a Blender" tenia que aconseguir 100.000 EUR per què la Fundació pogués comprar els drets del codi font i els de propietat intel·lectual de Blender als inversors de NaN i, posteriorment, alliberar Blender a la comunitat de codi obert. Amb un entusiasta grup de voluntaris, entre els que es trobaven varis ex treballadors de NaN, va ser llançada la companyia de "Alliberin a Blender". Per la sorpresa de tothom, la campanya va arribar al objectiu de 100.000 EUR en només 7 setmanes. El diumenge 13 d'octubre de 2002, Blender va alliberar al món sota els terminis de la Llicència Pública General de GNU (GLP). El desenvolupament de Blender continua fins els nostres dies conduit per un equip de voluntaris procedents de diverses parts del món i liderats pel creador de Blender, Ton Roosendaal.

1.3 Característiques de Blender

- Multiplataforma, lliure, gratuït amb una mida d'origen realment petita comparat amb altres paquets de 3D, depenent dels sistema operatiu del sistemea oparatiu en el que s'executa.
- Capacitat per una gran varietat de primitives geomètriques, incloent corbes, malles poligonals, buits, NURBS, metaballs.
- Juntament amb les eines d'animació s'inclouen cinemàtica inversa, deformacions per armadura o quadrícula, vèrtexs de càrrega i partícules estàtiques i dinàmiques.
- Edició d'àudio i sincronització de vídeo.
- Característiques interactives per jocs com detecció de col·lisions, recreacions dinàmiques i lògica.
- Possibilitats de renderitzat intern versàtil i integració externa amb potents traçats de raigs o "raytracer" lliures com kerkythea o YafRay.

- Llenguatge Python per automatitzar o controlar varies tasques.
- Blender accepta formats gràfics com TGA, JPG, Iris, SGI, o TIFF. També pot llegir fitxers Inventor.
- Motor de jocs 3D integrats, amb un sistema de caixes lògiques. Per a més control es fa servir programació en llenguatge Python.
- Simulacions dinàmiques per softbodies, partícules i fluids.
- Modificadors apilables, per la aplicació de transformació no destructiva sobre malles.
- Sistema de partícules estàtiques per simular cabells i pelatges, al que s'han afegit noves propietats entre les opcions de shaders per aconseguir textures realistes.

1.4 Objectius

El principal objectiu del projecte és trobar el format gràfic i la manera de treballar entre aquests dos softwares, de manera correcte, els quals es volen complementar per arribar a crear un entorn virtual interactiu.

Aquest format ha d'exportar primordialment la geometria i la textura dels objectes, i en la mesura del possible també altres termes com la il·luminació, càmeres i moviment.

Un cop funciona correctament la exportació i importació fent servir el format anterior, part indispensable del projecte global, es dissenya i modela un diorama de pessebre nadalenc (escenari i personatges) com un escenari en 3 dimensions, per exportar-ho seguidament al segon software, on es treballarà per aconseguir una interacció amb l'usuari en forma de videojoc.

El fet de fer un pessebre virtual ens suposa per tant, innovar en aquest camp tradicional del tema nadalenc. Afegint així una utilitat més al món dels dissenys de Diorames, i la seva expansió al mon de les noves tecnologies.

Aquesta memòria fa referència al projecte de la part de modelat i exportació. Això fa, que a banda dels objectius globals, apareguin d'altres més específics, com el d'aconseguir un modelat, texturitzat i detall, que permetin el màxim realisme possible sense sobrepassar la capacitat de funcionament del software o màquines que posteriorment rebran aquests objectes o escenari. Si el nombre de polígons d'un o molts objectes és massa gran, pot haver-hi una ineficiència a l'hora de treballar amb tot el conjunt d'objectes o a l'hora de realitzar el moviment a temps real. Una manera, per exemple, d'arribar a fer un objecte molt realista amb poca geometria podria ser el fet de fer servir una textura molt realista, com ara una fotografia.

1.5 Requisits

Seguidament explicarem els aspectes que ha de complir el format gràfic triat, i les necessitats que ens requerirà la creació del projecte.

1.5.1 Aspectes del format gràfic

Aspectes indispensables a suportar pel format gràfic:

- Geometria i topologia

- Textures o materials

Aspectes a estudiar molt interessants:

- Llums
- Moviment (rotació i translació)
- Posicionament de càmeres

1.5.2 Necessitats

El primer que es necessita és, evidentment, el programari de modelat Blender. Per a poder fer-lo servir, cal disposar d'un ordinador amb complements competents per la feina que volem realitzar. Tot i que la Fundació Blender va treien noves versions del software molt periòdicament i pel projecte se'n han utilitzat vàries, explicarem aproximadament els requisits de la màquina amb que es podria treballar.

- Requisits mínims per treballar amb Blender:
  - Processador: 300 MHz.
  - Memòria: 128 MB.
  - Vídeo: 16 MB.
  - Espai lliure en disc: 20 MB.
  - Resolució de pantalla: 1024 x 768.
  - glibc 2.3.6.
  - Python 2.5.
  - Tarja gràfica compatible amb OpenGL.

- Requisits recomanats per treballar amb Blender:
  - Processador: 2 GHz.
  - Memòria: 2 GB.
  - Vídeo: 256 MB.
  - Resolució de pantalla: 1920 x 1200

A continuació, el capítol 2 explica el desenvolupament del projecte en les diferents parts que aquest s'ha elaborat. Seguidament al capítol 3 es mostren els resultats finals obtinguts. Per últim el capítol 4, conclusions i millores, fa un balanç del que ha aportat la realització del projecte a nivell personal i a nivells més generals, i parla de les millores que es podrien dur a terme.

2 Desenvolupament

Dintre d'aquest capítol trobem l'explicació de les diferents parts de que consta el desenvolupament del escenari, incloent l'apartat de exportació dels models. Primerament parlarem del format gràfic i tot seguit de tot el referent a aspectes del disseny general del escenari, objectes i figures.

2.1 Anàlisi

Per a realitzar aquest projecte era imprescindible disposar d'un programari de modelat. Al ser Blender un software gratuït i accessible no va haver-hi cap problema pel que fa la seva disponibilitat.

Després d'estudiar els diferents formats d'exportació de Blender, es va trobar un format útil sense necessitat de crear-ne un de nou. Igualment s'ha examinat a fons l'estructura d'aquest, és a dir, com està organitzada la informació dins dels fitxers creats. Aquest estudi ha ajudat molt a l'hora d'interpretar errors inesperats i, evidentment, en la creació de la importador, part imprescindible del projecte complementari al nostre.

Pel que fa a l'ordinador utilitzat per modelar, era prou potent com per no tenir problemes de rendiment o de representació a mesura que l'escenari anava creixent. Processador, memòria i tarja gràfica han estat les parts més importants per no tenir cap problema. Blender et permet amagar objectes o mostrar-los amb diferents nivells de detall, però en alguns casos és necessari treballar amb la càrrega simultània dels objectes.

La recaptació de textures, principalment requeria accés a la xarxa de xarxes per trobar webs que permetessin descarregar textures gratuïtament amb la màxima qualitat i varietat possible. Com més endavant explicarem, en alguns casos s'han utilitzat fotografies entre d'altres imatges obtingudes amb diferents tècniques.

Les figures del pessebre s'haurien pogut modelar de la mateixa manera que les cases i el decorat, però finalment i contra tota expectativa varem poder fer servir un escàner 3D, la qual cosa va permetre agilitzar molt la feina, aconseguint impressionants resultats. El fet de disposar de figures reals de pessebre va fer que des del primer moment es pensés en aquest sistema.

Després d'estudiar la viabilitat del projecte, varem pensar que era perfectament viable, però com també era d'esperar, els problemes de importació i texturitzat han estat presents durant tot el projecte. Un altre factor que també era preocupant era la representació a temps real, però com ja s'ha modelat a consciència, finalment no ens hem trobat amb aquests tipus de problema a l'hora de representar l'escenari a l'entorn OpenGL.

El que varem tenir clar des del primer moment, tant el company del projecte complementari com jo, va ser que l'exportació i importació havia de funcionar abans de començar a endinsar-me en el disseny del diorama. Així ho varem fer, i va ser primordial.

2.2 Format d'exportació

Fins ara no es disposava d'un format adient per exportar els objectes modelats en Blender al software de programació, ja que les textures d'aquests objectes exportats perdien la relació amb la malla del model o s'espatllava. Llavors, la solució que es podia dur a terme per a solucionar-ho era la de realitzar un nou format gràfic, modificar-ne un

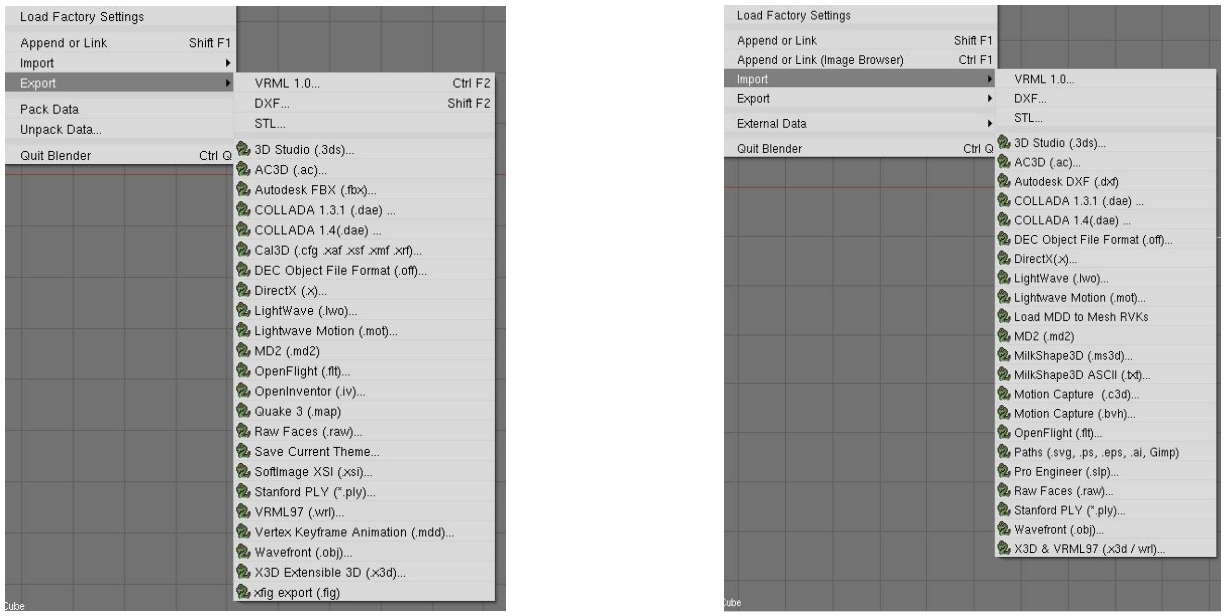
d'existent o trobar-ne algun que pogués treballar correctament dins del gran ventall d'exportadors de que disposa el Blender.

El primer que varem pensar va ser que el que més necessitàvem exportar era la geometria dels objectes amb les coordenades de les corresponents textures. Per aquest motiu, ens varem centrar en aquests dos conceptes del format, donant prioritat a proves i desenvolupament per què el format funcionés bé en l'exportació de geometria i textura. Una vegada això funcionés, entraríem en l'estudi de la viabilitat i proves d'altres conceptes interessants com podien ser la exportació de llums, moviment o posicionament de càmeres. Tot i ser funcionalitats que es poden realitzar des del mateix entorn de programació, podria donar molta agilitat i comoditat el fet de que el format permetés transportar també aquesta informació addicional.

Les proves inicials per trobar el format adequat s'han basat en la geometria i el texturitzat, per aquest motiu es va optar per anar mirant un per un com treballaven tots els formats d'exportació de Blender. Aquest estudi ha consistit en realitzar objectes senzills com ara un cub, assignar un material i una textura i anar fent proves amb la exportació i importació del propi Blender.

Aquestes proves es feien sense sortir del programa de modelat. Això ho varem fer per què després d'analitzar els problemes amb el format 3ds, varem trobar que qui no tractava bé la informació de les textures era el mateix Blender. Fins ara no es sabia ben bé d'on venien els problemes, si d'un software o del altre. És per això que el primer procés de proves es va centrar en exportar el cub amb un format concret i tornar a importar el resultat des del mateix programa. A partir d'aquí, un primer descart de formats va ser pels que no superaven aquest procés. Si apareixien problemes d'alguna classe es passava a un altre format. A la Figura 6.a es poden veure tots els formats d'exportació que Blender incorpora per defecte.

No tots els formats que Blender pot exportar, es poden també importar. Per tant les proves descrites només servien per uns quants dels formats. A la Figura 6.b veiem els formats que Blender pot importar.



(a) (b)

Figura 6. Formats d'exportació de Blender (a). Formats de importació de Blender (b).

Un factor molt important a tenir en compte és també que, l'estructura del fitxer o fitxers de text que s'acaben creant quant exportem objectes ha de ser el més senzill possible, ja que d'aquesta manera, a l'hora de realitzar la recaptació de la geometria i les textures es pogués dur a terme d'una manera entenedora i eficaç. Per tant, no hi havia prou amb trobar un format que faci bé l'exportació, sinó que ademés ha de crear uns arxius clars i entenedors.

Tot això feia pensar que no es trobaria cap format predefinit que tingués tot el que necessitava pel projecte, però després de moltes proves varem arribar al format Wavefront OBJ. Aquest format realitzava bé les exportacions i importacions des de Blender i els arxius generats eren molt senzills i clars.

El OBJ [FWO] és un format d'exportació/importació sense comprimir de plataforma UNIX que serveix, bàsicament, per intercanviar i emmagatzemar dades 3D. Aquest format és un estàndard per representar dades poligonals en format ASCII (tot i que també permet representar format binari).

Una característica molt important per aquest projecte, és que aquest format permet tenir múltiples imatges com a textura dins d'un mateix objecte. Això ens permet exportar cases senceres com un únic objecte o fins i tot el pessebre sencer.

L'únic inconvenient que hi podríem trobar és que la mida màxima de l'arxiu generat té un límit, tot i que en el nostre cas no ens afecta, i si així fos, simplement hauríem de dividir l'arxiu en més objectes i per tant més arxius més petits.

Tot i que aquests arxius poden ser ASCII (.Obj) o binari (.Mod), per aquest projecte només s'ha treballat amb ASCII. Per tant l'explicació que segueix es recolza d'exemples amb format ASCII. Cal a dir que per la versió utilitzada, el format OBJ és compatible tant amb els objectes poligonals com amb els objectes de forma lliure (corbes i cercles Bezier[3], corbes i cercles Nurbs[4], etc.). La geometria poligonal fa servir punts, línies i cares per definir els objectes, mentre que la geometria de forma lliure fa servir corbes i superfícies.

Per aquest projecte no s'ha modelat amb formes lliures, però ha estat simplement una qüestió personal, ja que no s'ha cregut convenient, ni ha estat en cap moment necessari o indispensable haver de modelar amb formes Bezier o formes Nurbs. Per altre banda, com ja comentarem més endavant, si que s'ha fet servir alguna d'aquestes tècniques per realitzar el recorregut de la càmera en la seva animació.

A la Figura 7.a es mostra un cub modelat amb Blender. Com es pot apreciar a la imatge, aquest cub té 8 vèrtexs, 6 cares i 12 arestes. A més cada cara té una normal que apunta en la direcció perpendicular de la cara i cada vèrtexs també té una normal que té un angle proporcional a les arestes que tenen com a punt inicial o final aquest vèrtex.

A la Figura 7.b es pot apreciar l'espai de coordenades 2D on es representen els vèrtexs, arestes i cares del objecte anterior després d'haver fet el desplegat UV per aconseguir el desplegat de les cares i poder aplicar les textures corresponents. En l'apartat de texturitzat s'aprofundeix en el funcionament d'aquesta tècnica. Ara per ara, només interessa posar aquest exemple per mostrar com el format OJB tracta tota aquesta informació.

---

[3] *Les formes Bézier són un sistema desenvolupat cap als anys 60. Es basen en el mètode de descripció matemàtica de les corbes ideat per Pierre Bézier .*

[4] *NURBS (acrònim anglès de l'expressió Non Uniform Rational B-Splines) és un model matemàtic molt utilitzat en la computació gràfica per generar i representar corbes i superfícies.*

(a) (b)

Figura 7. Cub com a objecte 3D (a). Espai 2D de coordenades de textura (b).

El format ha d'anar escrivint tota la informació necessària dins del fitxer de text que més tard és convertirà en l'arxiu amb extensió .obj. Aquesta informació, en aquest cas bàsica, són les coordenades de vèrtexs i cares en l'espai 3D per una banda, i les coordenades dels vèrtexs referents al desplegat de la malla en l'espai 2D.

Cal a dir que el format OBJ de Blender genera una finestra emergent abans d'exportar, perquè l'usuari pugui triar diferents opcions en l'exportació, com el fet d'exportar o no normals, o el de agrupar els objectes separats en un de sol. Més endavant parlarem d'aquestes opcions.

Imaginem que el nostre arxiu exportat es diu "cub.obj". A continuació és mostra com quedaria el contingut d'aquest arxiu si exportem el cub anterior.

```
# Blender3D v248 OBJ File:
# www.blender3d.org
mtllib cub.mtl
v 1.000000 -1.000000 -1.000000
v 1.000000 -1.000000 1.000000
v -1.000000 -1.000000 1.000000
v -1.000000 -1.000000 -1.000000
v 1.000000 1.000000 -1.000000
v 1.000000 1.000000 1.000001
v -1.000000 1.000000 1.000000
v -1.000000 1.000000 -1.000000
vt 0.923808 0.923808
vt 0.923808 0.076192
vt 0.076192 0.076192
vt 0.076192 0.923809
vt 0.923809 0.923808
vt 0.076192 0.923808
vt 0.076191 0.076192
```

```
vt 0.923808 0.076191
vt 0.076191 0.923808
vt 0.923809 0.076192
usemtl Material
s off
f 1/1 2/2 3/3 4/4
f 5/5 8/6 7/7 6/8
f 1/8 5/1 6/9 2/3
f 2/8 6/1 7/9 3/7
f 3/3 7/4 8/1 4/10
f 5/1 1/2 4/3 8/4
```

A partir d'aquí el primer que trobem és la referència a l'arxiu complementari al OBJ que té la informació sobre el material. La comanda *mtllib* crida el arxiu *cub.mtl*. Blender crea aquest fitxer si el objecte exportat té un material associat. Més endavant s'explicarem la informació que conté aquest arxiu amb més detall. De moment seguim amb la informació de la geometria.

A continuació apareixen les coordenades x, y i z dels vèrtexs que formen l'objecte. La lletra "v" indica vèrtex i els valors són els de la x, la y i la z respectivament. Si per exemple tinguéssim un vèrtex repetit, veuríem una línia repetida. Per tant, com hem dit que el cub té 8 vèrtexs, al arxiu apareixen 8 línies de la forma v x y z. A la Figura 8 veiem el cub amb les coordenades dels vèrtexs.

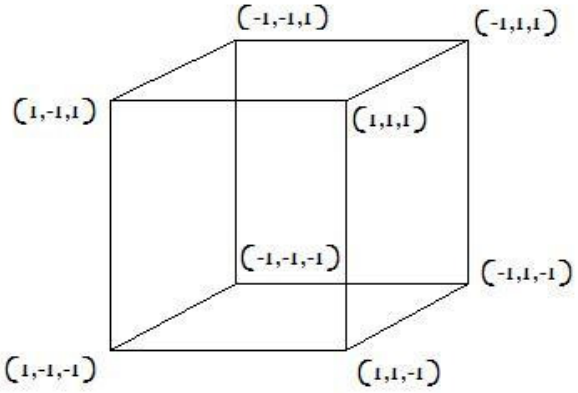


Figura 8. Coordenades de vèrtexs

Després de la informació dels vèrtexs seguim amb les coordenades de la textura en l'espai de dos dimensions del UV Maping. El format d'aquestes línies ve precedit per "vt" i seguit de les coordenades w i v de cada vèrtexs. Cadascuna d'aquestes línies ens diu on està un punt concret dins d'un espai que va de (0,0) fins (1,1). Exactament l'espai que engloba la textura. Si algun valor surt dels límits simplement es van repetin els mateixos límits de coordenades en l'espai. Per tant, a efectes òptics serà com repetir a imatge.

Varies línies juntes representaran una cara de la figura. El exportador ordena aquestes línies amb els seus propis índexs dins del fitxer de manera que després es puguin relacionar amb les línies que representen els vèrtexs del objecte, on de la mateixa manera, cadascuna tindrà un índex associat.

Si hi ha alguna informació o coordenada repetida, no s'escriurà dues vegades, només apareixerà una línia representant tots aquells vèrtexs o coordenades de textures iguals; com és el cas d'aquest exemple, on en teoria hi haurien d'aparèixer 24 línies, una per cada vèrtex projectat. I només n'apareixen 10. La xifra de 24 surt de que si cada cara té 4 vèrtexs i tenim 6 cares, al espai 2D realment hi ha 24 punts projectats, però com està passant aquí, les projeccions frontals donen lloc a molts punts amb les mateixes coordenades. A la Figura 9 veiem aquest, on es representa la textura en el interval que va de (0,0) a (1,1).

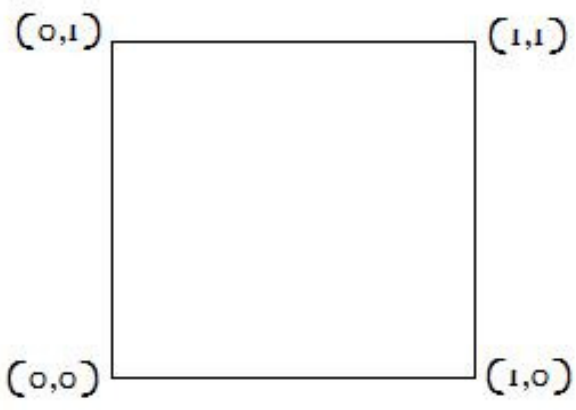


Figura 9. Coordenades de textures

Tot seguit la comanda *usemtl* ens indica el material que hem de fer servir, pel grup de vèrtexs anteriors. Com pot haver-hi varis materials o textures per un mateix objecte, llavors podem tenir diferents materials o textures per cada cara del objecte. Per tant necessitem saber quines imatges o propietats té cada part de la malla. En el cas que això passi, ens trobarem amb línies de coordenades de textures intercalades amb la línia de *usemtl*.

La lletra "s" ens identifica el grup de suavitzat. Per aquest projecte no s'ha fet servir el suavitzat de Blender, ja que l'entorn OpenGL ja té una tècnica de suavitat basada en la llum. Per aquest motiu, tal i com es veu a l'exemple, la lletra "s" ve seguida d'un "off" que indica la manca de grups de suavitzat.

Per últim ens trobem amb la informació referent a les cares del objecte. La lletra "f" ens indica el començament d'una línia de text que ens construirà una cara. La manera en que el format OBJ representa aquestes cares és relacionant cada vèrtex definit per les línies de la forma v x y z amb la coordenada de textura definida a les línies amb la forma vt x y. És a dir, si tenim f 1/1 2/2 7/8 4/9 significa que aquesta cara està formada pels vèrtexs 1, 2, 7 i 4 (numeradors) . Aquests valors indiquen la línia de definició de cada vèrtex, és a dir, en ordre, el primer vèrtex,el segon, el setè i el quart. Llavors el denominador ens indica amb quina coordenada de textura està relacionat cadascun d'aquests punts. En aquest exemple tindríem que el vèrtex 1 està associat a la primera coordenada de textura. Abans s'ha dit

que quan s'escrivien les línies referents a les coordenades de textures, els punts repetits només es posaven una vegada; és per això, que en l'associació de les cares es poden repetir valors en el denominador. El numerador repetirà també valors sempre que dues cares comparteixin el mateix punt. Evidentment, hi haurà tantes línies que comencen amb "f" com cares tingui el nostre model.

Aquests termes explicats són els bàsics que tindria un objecte senzill modelat amb Blender. Encara podria ser més senzill si l'objecte no tingués un material o una textura associada. Si només tinguéssim la geometria, llavors no tindríem ni el arxiu mtl ni la informació relacionada amb les textures.

L'estructura de l'exemple anterior seria la de exportar geometria i textura, que és el nostre objectiu principal. A partir d'aquí el format OBJ permet exportar altres propietats com ara les normals dels vèrtexs, efectes diversos del material, etc. Però com en el projecte no s'han utilitzat, no s'explicaran detalladament. En els annexes de la memòria apareixen les diferents opcions que permet el format OBJ i les seves possibilitats.

Pel que fa al arxiu del material (mtl), a continuació es mostra quin seria el seu contingut pel cas del cub anterior. El arxiu *mtl* és l'encarregat de descriure els materials definits al arxiu *obj*.

```
# Blender3D MTL File:
# Material Count: 1
newmtl Material
Ns 96.078431
Ka 0.000000 0.000000 0.000000
Kd 0.640000 0.640000 0.640000
Ks 0.500000 0.500000 0.500000
Ni 1.000000
d 1.000000
illum 2
```

El primer que trobem en aquesta estructura és l'exponent especular (shininess), que s'identifica amb la paraula *Ns* i indica que el següent "token" és una cadena que representa un valor numèric on el seu rang varia normalment entre 0 i 1000.

A continuació ens trobem la paraula *ka*, que ens dona el color de la llum ambiental (ambient). Indica que els següents tres "tokens" corresponen a tres valors de tipus flotant (entre 0 i 1) que defineixen els components RGB del color de la llum ambiental que serà reflectit en el material.

Kd ens determina el color difús del material (diffuse). De la mateixa manera que el cas anterior, tenim tres valors flotants, però ara aquests valors RGB marquen el difús.

La paraula *ni* i *d* ens indiquen l'índex de refracció i valor alfa respectivament. Per defecte són 1 tant un com l'altre. Un valor de 1.1 per la alfa ens indica que el material és completament opac, mentre que un valor de 0.0 ens donarà un objecte totalment transparent.

Per últim trobem la paraula *illum*, la qual ens habilitat o inhabilita els models d'il·luminació anteriors. Si té un valor de zero estem ignorant tota la il·luminació. Si tenim

un 1 simplement inhabilitem la il·luminació especular. Per defecte tindrem el valor 2, que deixa habilitades totes les il·luminacions: ambient, difusa i especular.

Totes aquestes dades són prou interessants, però de moment no s'han tingut en compte, ja que depenen bastant de la llum que tingui la escena final i d'alguns factors més relacionats amb l'entorn OpenGL. De moment, doncs, n'hi ha prou en saber que aquests valor hi són i que en qualsevol moment es poden provar i jugar amb ells.

2.2.1 Exportació en Blender amb Wavefront OBJ.

Quan volem exportar un model, després d'haver-li donat forma i haver-li assignat una o vàries textures, simplement hem de seleccionar aquell o aquells objectes que ens interessin i seleccionar l'opció *Export* del menú *File.* Dins del desplegable s'ha de seleccionar *Wavefront (Obj)*. Ens apareixerà un menú d'opcions després d'haver introduït la ruta i el nom de com volem guardar l'arxiu generat, triarem les opcions que ens interessin i exportarem. A la Figura 10 mostrem el menú emergent que s'obre per triar les opcions d'exportació del format OBJ.

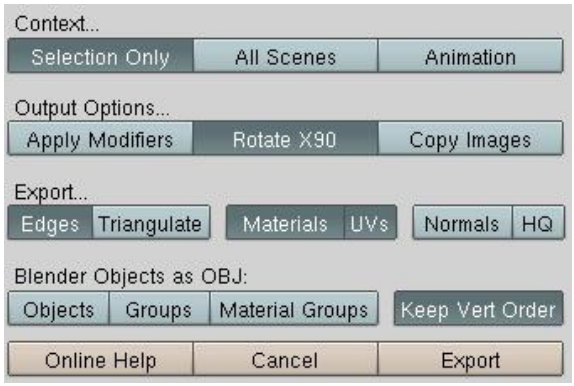


Figura 10. Menú d'opcions del format d'exportació OBJ.

El primer que ens permet triar és el que volem exportar de l'escena. Podem activar exportar els elements seleccionats, exportar tota l'escena o exportar l'animació. Per defecte, com es pot veure a la figura, tenim només activada la primera de les tres opcions, ja que és el cas més comú.

El segon grup de botons ens permet definir les opcions de sortida. Blender disposa d'un seguit d'algorismes que modifiquen de diferents maneres els models. Quan es realitzen aquestes modificacions (Modifiers) és molt comú deixar-los sense aplicar per si en algun moment no es volen fer servir. Bé, doncs la primera opció d'aquest grup és aplicar aquests "modifiers" i que els resultats siguin exportats. El botó "Rotate X90" és el que està activat per defecte i el que fa es canviant-se el eix de la Z pel eix de les Y. En altres paraules, ens rota el model 90 graus en l'eix de les X. Això és necessari perquè en moltes aplicacions, com en el nostre cas, l'entorn de programació, amb el l'eix de les Y com el eix vertical. Blender pel contrari treballa amb l'eix de les Z com l'eix vertical, i per això cal aplicar la rotació abans d'exportar. Finalment, dins de les opcions de sortida trobem l'opció de copiar imatges, si ho activem l'exportador farà una copia de les imatges que té l'objecte com a textures per enganxar-la al mateix lloc on es crea el arxiu OBJ. Això simplement ens

estalviarà el fet d'haver de copiar nosaltres mateixos les textures allà on es vulgui guardar el arxiu.

El tercer grup "Export" ens dona per defecte activades les opcions "Edges", "Materials" i "UVs". La primera ens exporta les arestes del objecte sempre i quan estigui activat. El segon i el tercer d'aquests tres, fan referència al material i les coordenades de textura UV Faces. Aquestes dues opcions són indispensables pel nostre projecte, ja que és exactament el que dona problemes en altres formats. Per tant deixarem aquests dos últims botons activats. A part d'aquests trobem unes altres opcions desactivades per defecte, són la de "Triangulate", que ens transformarà aquells polígons que no siguin de tres costats en triangles (hi ha moltes aplicacions que treballen només amb triangles), la de "Normals" que ens exportarà també els valors que defineixen les normals dels vèrtexs de l'objecte (ara per ara no ho necessitem) i per últim l'opció "HD" que si està activat permet l'exportació de normals d'alta qualitat per renderitzat (la deixem desactivada).

Finalment podem triar entre exportar objectes, grups i grups de materials o pel contrari, el que hi ha marcat per defecte: "keep vèrtex order" que ens mantindrà els vèrtexs tal i com nosaltres ho tenim al Blender.

Un cop estudiat el funcionament del exportador d'aquest format triat i realitzades les proves necessàries des de la part de disseny es va dur a terme la creació d'un importador per part del company del projecte complementari. L'últim pas per tant era provar que l'exportació/importació tingués èxit, ja que era la part indispensable per poder tirar el projecte endavant. Tenir èxit significa que el que es visualitzés a l'entorn OpenGL havia de ser igual que el modelat d'origen realitzat amb BLender, evidentment tant en la part de modelat com en la part de texturitzat. Podem veure diferències en la il·luminació, però això ara per ara és normal, ja que a l'entorn de Visual C++ encara s'hi treballa. Com veiem a les Figures 11.a i 11.b l'objecte s'exporta correctament pel que a malla i textura es refereix.

(a) (b)

Figura 11. Visualització a Blender (a). Visualització a l'entorn d'OpenGL (b).

2.3 OBJ extens

Als objectius del projecte s'ha comentat que les principals funcionalitats del format gràfic eren l'exportació de geometria amb textures, i que hi d'altres factors com il·luminació, càmeres i moviment que serien interessants d'estudiar. En aquest apartat explicarem alguna cosa més sobre aquestes funcionalitats.

2.3.1 Animacions

Pel que respecta al moviment d'objectes dins de l'escena, seria interessant que poguéssim exportar una animació concreta o un cicle d'animació des de Blender. Posem per exemple que volguéssim fer el moviment de les aspes d'un molí. El format OBJ ens permet exportar aquest moviment (botó *Animation* del menú emergent), però el que fa és crear un fitxer .obj i un .mtl per cada frame que avança l'objecte en qüestió. En la majoria dels casos, el fitxer .mtl serà igual per un mateix objecte, ja que normalment no canviarem les seves textures o característiques en funció del frame on estigui. Per tant, aquí hi podríem intervenir per no tenir informació repetida. Pel que fa al fitxer .obj ens mostrarà simplement la posició de cada vèrtex de la malla en aquell instant de temps. De manera que la única informació que varia d'un fitxer a l'altre és la posició de les coordenades que s'hagin mogut d'un frame a l'altre. A la Figura 12.a podem apreciar la posició dels vèrtexs d'un cub en un estat inicial (frame 1). La Figura 12.b ens mostra la part que varia del segon fitxer .obj després de desplaçar el cub -5 unitats en el eix de les Y's (frame 2). Com es pot apreciar en la figura, el increment es fa a la tercera coordenada (Z), això és degut a la rotació de 90º abans mencionada.

```
v  1.000000 -1.000000 -1.000000
v  1.000000 -1.000000  1.000000
v -1.000000 -1.000000  1.000000
v -1.000000 -1.000000 -1.000000
v  1.000000  1.000000 -1.000000
v  0.999999  1.000000  1.000001
v -1.000000  1.000000  1.000000
v -1.000000  1.000000 -1.000000
```

(a)

```
v  1.000000 -1.000000 -6.000000
v  1.000000 -1.000000 -4.000000
v -1.000000 -1.000000 -4.000000
v -1.000000 -1.000000 -6.000000
v  1.000000  1.000000 -6.000000
v  0.999999  1.000000 -4.000000
v -1.000000  1.000000 -4.000000
v -1.000000  1.000000 -6.000000
```

(b)

Figura 12. Fitxer .obj del primer frame (a). Fitxer .obj del segon frame (b).

Amb aquesta informació el programador de l'entorn podria reproduir el mateix moviment del objecte en l'escena importada des de Blender.

Els fitxers .obj són arxius de text que ocupen molt poc espai, per tant, en principi no ha de ser un problema el fet de tenir un arxiu per cada frame generat. Si això acabes essent perjudicial a l'hora de treballar sempre es podrien buscar altres solucions com transformar de forma externa aquests arxius en un de sol o de forma interna modificant el codi del exportador OBJ. Per afegir aquesta funcionalitat a aquest projecte, la opció de llegir un fitxer per cada frame creiem que pot ser la més senzilla i eficaç d'utilitzar, ja que les

animacions que hi podríem fer no serien gaire complicades. De fet, si ho fossin, valdria la pena mirar algun altre format que exporti animacions.

2.3.2 Il·luminació

Blender permet crear varis tipus de llums (lamp, Area, Sun, Spot, Hemi o Ambient oclusion) amb diferents propietats (distància, color, intensitat, etc.), però el format OBJ, ara per ara, no pot exportar aspectes de la il·luminació d'escena.

Per a millorar les prestacions del format sobre aquest aspecte, s'ha pensat una solució. Com s'ha dit a la part d'explicació del format, els comentaris dins del fitxer generat s'identifiquen per anar precedits del símbol #. Per tant, podríem generar codi extra amb informació dels llums, però que s'escrivís com a comentari dins del arxiu .obj. D'aquesta manera l'exportador original no es veuria alterat i seguiria treballant com abans, però el nostre programador del importador podria tenir una informació extra per si volgués fer-la servir.

Pel que respecta a les propietats dels llums, Blender en té unes i l'OpenGL en té unes altres. Caldria mirar si val la pena agafar propietats de Blender o posar nosaltres mateixos el codi que volguéssim per a definir-les.

Pel que respecte les posicions d'aquests llums, el que s'ha pensat és crear punts o vèrtexs a les posicions dels llums, de manera que podem aconseguir la posició d'aquell llum dins de l'escena, simplement agafant la coordenada d'aquest punt. Crear les llums amb Blender és molt més senzill que fer-ho amb OpenGL, i a part, té més sentit que sigui l'artista qui els defineixi. Per tant, el que es faria és crear un objecte amb el mateix nom que el llum, que només tingués un vèrtex, i a l'hora d'exportar, seleccionar els punts en comptes del propi llum. A la Figura 13 es veu un exemple de com es podria fer això. A la Figura 13 es pot apreciar el punt de color rosa que hi ha al mig de la llum. Aquest punt seria el que exportaria.

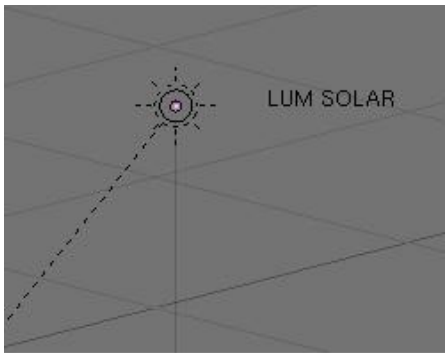


Figura 13. Punt al centre de la llum.

Cal a dir que OpenGL només permet 8 llums simultànies. Llavors, seria qüestió de carregar unes o altres en funció del que es volgués fer en cada moment.

A continuació es mostra el codi que podríem afegir al exportador de python per escriure aquesta informació. Si el nostre objecte (punt) li diem "ob", llavors aquestes dues línies escriurien el nom del llum seguit de la coordenada, sempre això sí, amb el símbol "#" davant per no modificar el que genera el format OBJ original.

```
file.write('#nom llum %s\n' % me.name)
file.write('#coordenada %.6f %.6f %.6f\n' % tuple(ob.verts[0].co))
```

Seguidament veiem com quedaria la informació dins del fitxer .obj que genera l'exportador. Allà on li haguessim dit que escriguis apareixerien aquestes dues línies referents al nom i la posició de la llum.

```
#nom llum LlumSolar
#coordenada 4.077725 5.903692 -1.002858
```

2.3.3 Càmeres

Blender també permet crear càmeres dintre del escenari, però de la mateixa manera que les llums, aquestes no tenen geometria, i el format OBJ tampoc les pot exportar. Posar càmeres des del mateix software de modelat pot ser molt útil pel programador, ja que la creació de càmeres des del mateix entorn d'OpenGL pot ser una mica pesat i necessitar moltes proves.

Per tant la metodologia ideal seria que també fos l'artista qui situés a l'escena les diferents vistes o càmeres amb les que després s'haurà de visualitzar l'escena.

Per fer-ho s'ha pensat una solució semblant al cas anterior. Un cop estiguin creades les càmeres allà on vulguem, crearem un pla, i el posicionarem al que seria la pantalla de la càmera amb les mateixes dimensions que el marc de visualització que marca la càmera. Per últim dividirem el pla en 4 triangles iguals, de manera que podem tenir un punt central, com es mostra a la Figura 14.

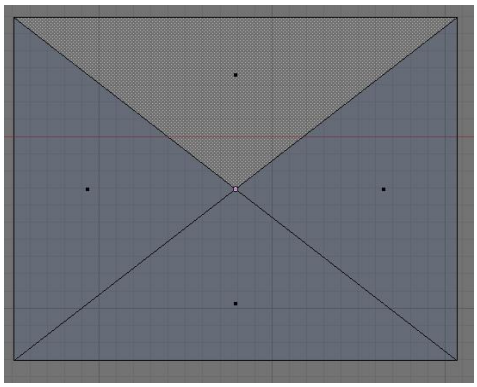
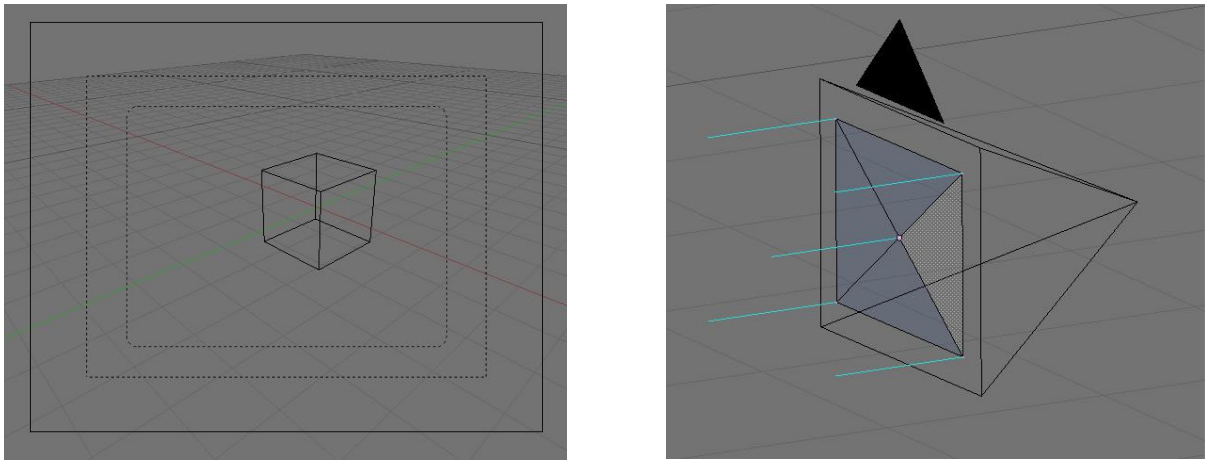


Figura 14. Pla subdividit en 4 triangles.

Llavors del que es tracta és d'exportar el pla en comptes de la càmera en sí. Escriurem al fitxer les coordenades dels 5 vèrtexs, i activarem la opció "Normals" del menú emergent del exportador OBJ. D'aquesta manera tindrem la posició de la càmera, els vèrtexs que delimiten els marcs de visualització i les normals dels vertexs, que ens indicaran la direcció on està enfocant la càmera.

Des de l'entorn aquestes dades es llegiran del fitxer gràfic i es faran servir per posar els atributs de la càmera en qüestió.

A la Figura 15.a podem veure el marc d'una càmera a Blender. La línia discontinua més exterior és el marc de visualització on hauríem d'encaixar el pla. A la Figura 15.b veiem com quedaria el pla encaixat a la càmera.



(a) (b)

Figura 15. Vista des de la càmera de Blender (a). Càmera amb pla encaixat i normals (b).

Si a l'hora d'escriure el codi python el nostre objecte li diem "ob", llavors les línies de codi que ens guardarien la informació de les coordenades i les normals podrien ser les següents.

```
for v in ob.verts:
      file.write('#coordenada %.6f %.6f %.6f\n' % tuple(v.co))
      file.write('#vn %.6f %.6f %.6f\n' % veckey3d(v.no))
```

Cal a dir que a l'hora de posar-les al codi, no estarien al mateix bucle i hi hauria codi abans i després que hi afectaria, però per aclarir el concepte es mostra tot junt en aquestes dues línies.

Llavors la informació dins del fitxer gràfic apareixeria tal i com es mostra a continuació. En realitat es llegeix una normal (direcció) per cada vèrtex, però com són iguals se'n escriu només una. A partir d'aquí el programador, al entorn, simplement hauria de calcular quin és el centre del pla, però amb una funció comparativa senzilla ho podria saber.

```
#coordenada 5.986991 4.191413 7.752461
#coordenada 5.987063 4.183339 6.741508
#coordenada 5.987063 4.940890 6.735457
#coordenada 5.986991 4.948964 7.746411
#coordenada 5.987027 4.566152 7.243959
#vn -1.000000 -0.000001 -0.000071
```

2.4 Disseny

A continuació en aquest capítol hi trobarem tota la part relaciona amb el modelat dels objectes del pessebre, l'aplicació de textures i tots els aspectes que s'han considerat importants a l'hora de dur a terme el disseny sencer del diorama.

2.4.1 Disseny general del escenari

El disseny del diorama el protagonitza un carrer principal d'un poble envoltat de cases ambientades en el que seria un poble rural de l'època.

Quan es va platejar el disseny es va pensar en un diorama que donés lloc a moviment, és a dir, que l'usuari s'hagués de moure per arribar a veure-ho tot bé. D'alguna manera s'havia de buscar un factor que justifiqués el fet de fer un diorama en 3D. En els pessebres tradicionals, l'espectador simplement es limita a mirar, i tot i el realisme i l'art amb que estan fets, en el fons sap que allò que veu no és real. El fet de crear una escena més amplia i interactiva fa que la persona, que fins ara, només podia quedar-se rere del vidre, pugui entrar dins de la maqueta i explorar com si es transportés a aquella època.

A la Figura 16 es pot veure un primer esbós que es va dibuixar per plasmar les idees que fins ara només s'havien pensat. A la sortida del poble hi ha un riu i passat el riu hi trobem l'establia amb el naixement.

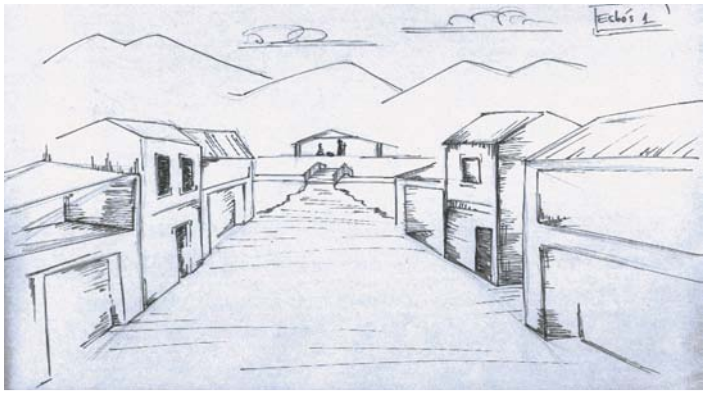


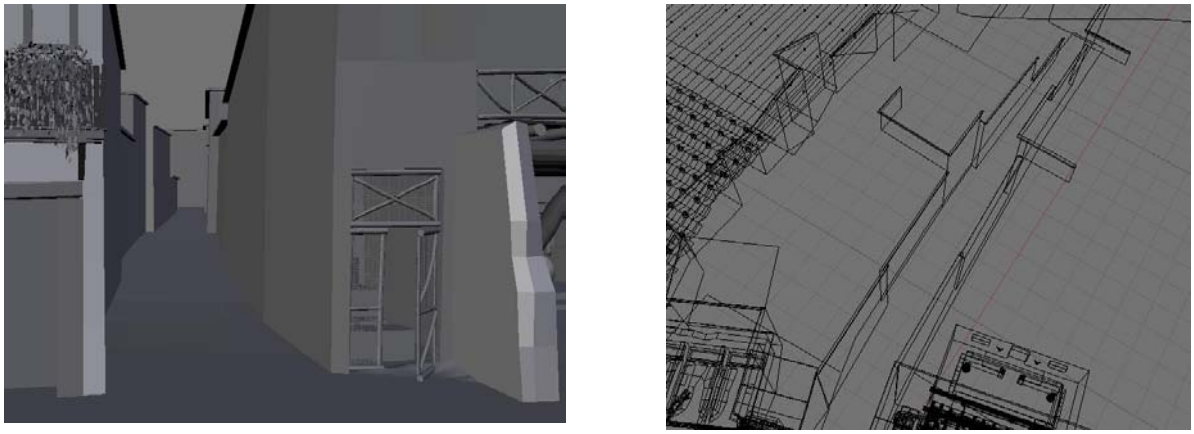
Figura 16. Esbós del diorama.

Com a paisatge s'ha dissenyat un espai tancat simulant una maqueta, posant un panell cilíndric amb cel i muntanya. Per a donar encara més la sensació de que ens trobem en un diorama, s'ha posat un marc de fusta "gegant" per simular el vidre frontal que tenen els diorames. La Figura 17 mostra la textura que s'ha fet servir com a paisatge. La seva mida és molt gran, però val la pena no escatimar en aquests casos, ja que com més qualitat més realisme.

Figura 17. Textura del paisatge (9000 x 1217).

L'objectiu de posar un fons cilíndric i tancat és per simular que l'espectador està dins d'una maqueta i pot navegar-hi com si fos un personatge del pessebre. En el fons, no s'ha buscat donar la sensació de que el paisatge no té fi.

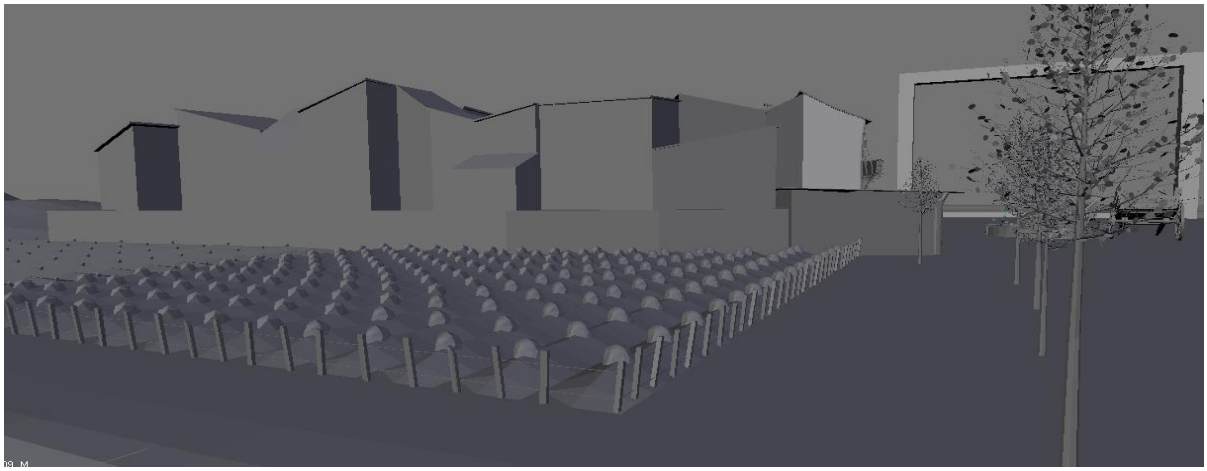
S'havia de donar la sensació de que les cases continuaven per darrera i no que només hi havia una primera fila d'edificis. Per fer-ho s'ha optat per fer un carrer que s'endinsa cap al poble. La Figura 18.a mostra aquest carrer, on només s'ha modelat la geometria visible. A la Figura 18.b podem apreciar, des de la vista aèria, la poca geometria que té aquest carreró.



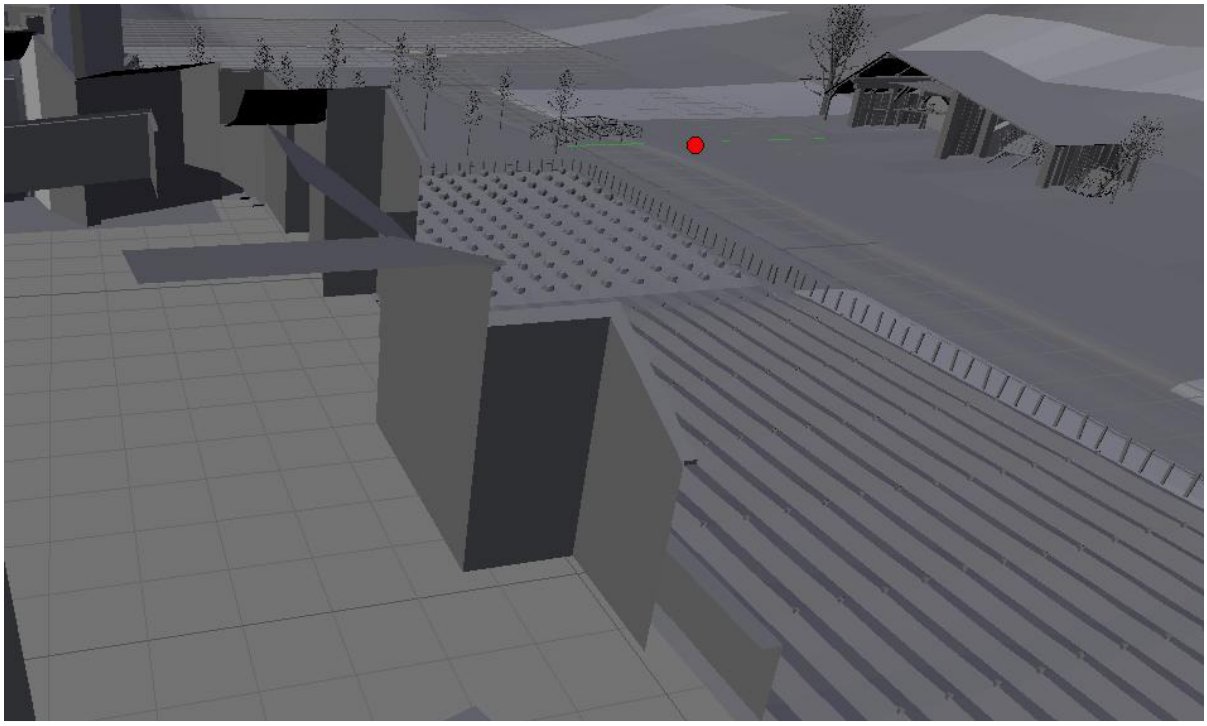
(a) (b)

Figura 18. Vista que veu l'espectador del carreró (a). Geometria del carreró des de la vista aèria (b).

Pel cas en que l'espectador està fora del poble també s'ha de donar aquest efecte. S'han modelat sostres i parets que donen volum a la escena. Són cases on només s'ha modelat allò que es pot veure des d'algun punt del pessebre. És la mateixa tècnica utilitzada en el carreró anterior, però amb la diferència que ara els possibles angles de visió del espectador poden ser més diversos. La Figura 19.a mostra una d'aquestes vistes des de fora del poblat. A la Figura 19.b, hi podem veure, novament, la "trampa" que ens permet bons resultats amb una quantitat de polígons molt reduïda. El punt vermell que apareix a la imatge seria la posició on estaria l'espectador per veure aproximadament la vista de la Figura 19.a.



(a)



(b)

Figura 19. Vista del l'espectador fora del poble (a). Detall de les cases "falses" (b).

Un factor molt important ha estat que la complexitat dels objectes i la mida de les textures fos l'adequada en relació a la proximitat en que veurem aquell objecte concret quan naveguem per l'escena. Si una casa només la veurem des de lluny, no cal que aquesta tingui molta poligonalitat ni una textura amb molta qualitat.

Per al disseny de les cases, s'ha agafat alguna referència de fotografies de pessebres, però la majoria de detalls dels dissenys ha anat sorgin a mesura que s'anava avançant en el modelat.

A l'hora de triar el tipus de cases a modelar, es va pensar en vivendes, però també en algun edifici que fos una espècie de taller de fusta, o un estable d'animals. S'havien de representar varis models prou diferents i en no molt espai. A la Figura 20 podem veure l'exemple del modelat de la fusteria del poble.

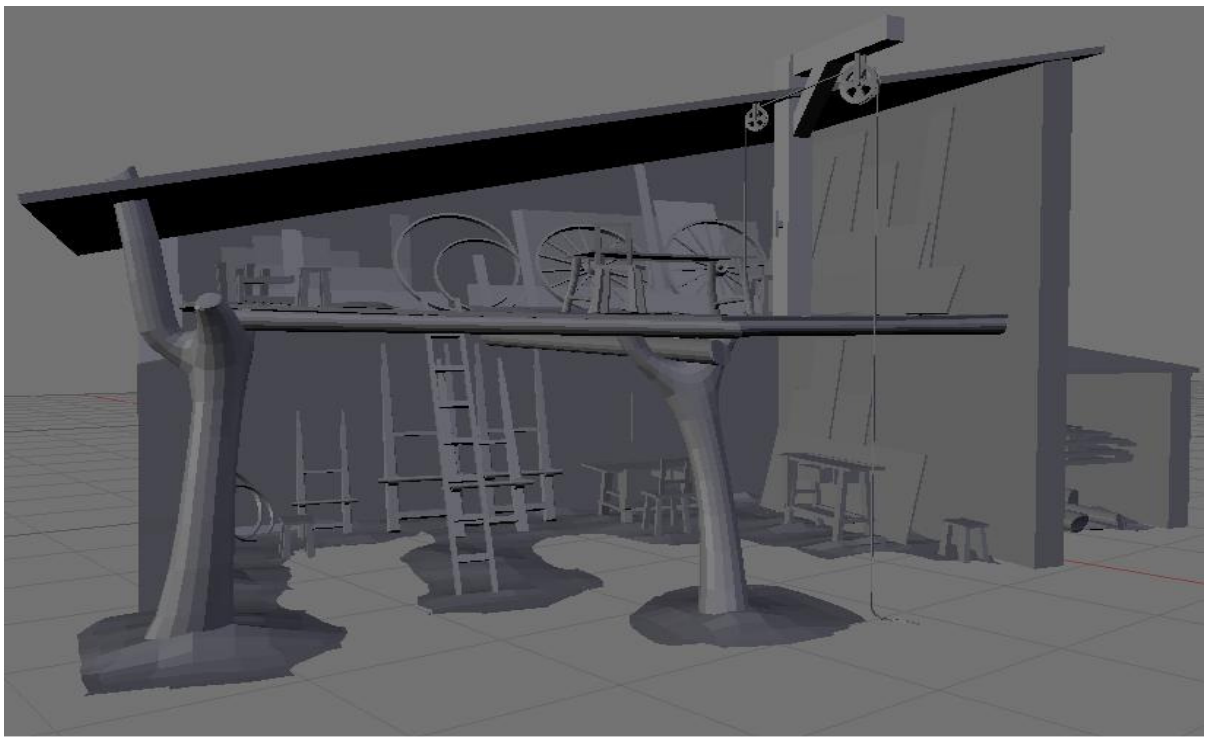


Figura 20. Modelat d'un edifici en forma de fusteria.

A part de tot el disseny d'edificis, la qual ha estat la que més temps i atenció ha requerit, hi havia altres detalls que finalment serien indispensables pel disseny final del diorama. Aquests serien arbres, l'hort, figures, i paisatge en general. Pel que fa als arbre i vegetació en general, s'ha fet servir un script de codi Python anomenat *Gen3. Blender tree generator*, que permet crear arbres a partir de la tria de diferents característiques com número de fulles, forma, número de branques, etc. [Sga]. A la Figuar 21.a i 21.b es poden apreciar els resultats de fer dos arbres totalment diferents. Cal a dir que l'arbre de la Figura 21.b ha requerit modificacions posteriors al arbre creat pel generador per aconseguir un disseny més espectacular.

(a)

(b)

Figura 21. Arbre senzill generat amb Gen3 (a). Arbre complex generat amb Gen3 (b).

Per alguns detalls de vegetació en general com plantes o semblants, també s'ha fet servir la mateix tècnica, però principalment per aconseguir la malla de les fulles i branques, ja que posteriorment s'ha hagut de retocar molt la geometria per aconseguir el resultat desitjat. La Figura 22 ens mostra un exemple del Gen3 aplicat a plantes.

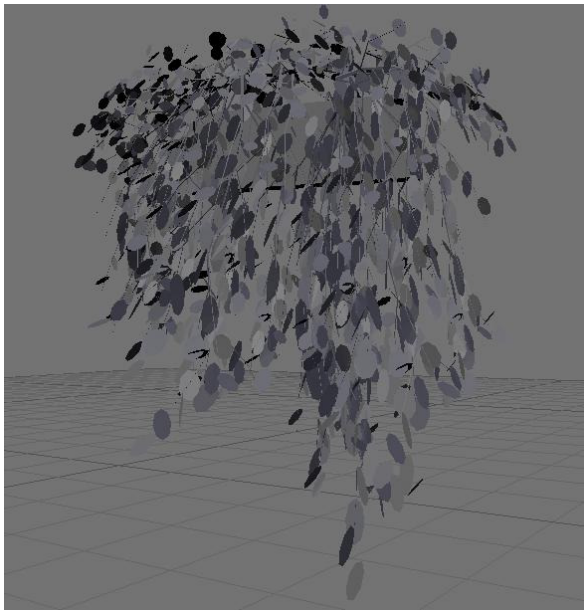
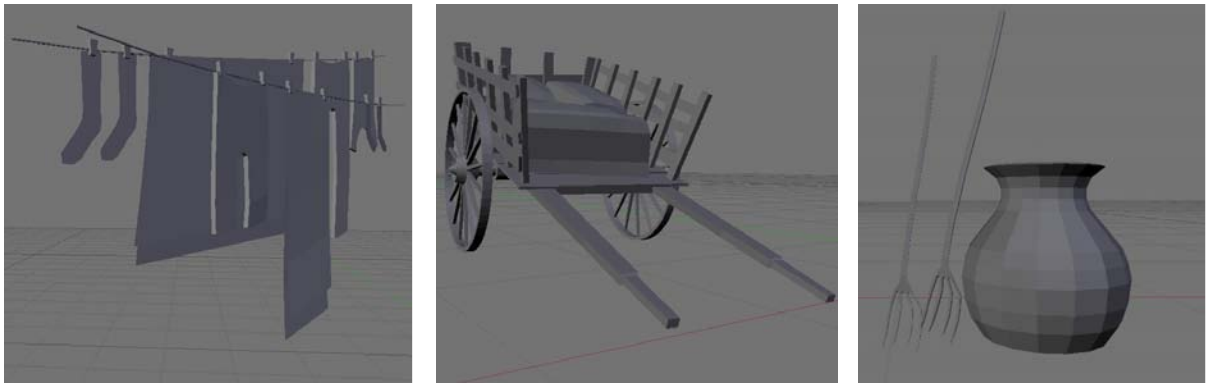


Figura 22. Modelat d'un test amb plantes utilitzant una base de fulles generada amb Gen3.

Una part molt important per donar realisme a l'escena ha estat la introducció de petits objectes com ara tot tipus d'estris, elements decoratius, carretons, roba estesa, etc. Tots aquests detalls s'han anat modelant en funció del lloc de l'escena on s'havien de situar. No s'ha pensat un disseny o distribució prèvia al modelat, sinó simplement han estat idees que han anat sorgint durant el procés de creació del pessebre. El detall i complicació del modelat d'aquestes peces o objectes ha variat depenent de la simplicitat del mateix objecte. Un petit exemple de tot això seria el que veiem a la Figura 23.



(a) (b) (c)

Figura 23. Roba estesa (a). Carretó amb palla (b). Jerro amb forques (c).

A partir d'aquí, passem a explicar com s'ha modelat amb Blender i com s'ha texturitzat. Pel que fa les figures del pessebre, cap el final del capítol s'explica com s'han creat, ja que la metodologia utilitzada requereix una explicació més detallada.

2.4.2 Trets importants

El procés de modelat dels objectes ha estat la part més costosa, ja que quan es modela s'han de tenir en compte molts factors que et limiten el disseny. Des del primer moment s'ha de saber quin serà l'ús final de les malles que es crearan, ja que no és el mateix fer objectes per crear un vídeo o una animació, que per crear un videojoc o una aplicació interactiva. En el segon cas és molt important treballar amb malles no gaire complexes o, dit d'una altre manera, amb baixa poligonalitat.

Aquest projecte forma part d'una aplicació interactiva, per tant s'havia de trobar una bona relació entre objectes poc pesats i un bon acabat realista. Per aconseguir aquest propòsit es va dedicar gran part del projecte a aconseguir bones textures que compensessin les limitacions de les malles.

Hi ha vàries maneres d'aconseguir bones textures. En el nostre cas varem fer-ho de tres maneres diferent:

1. Fer fotografies. Dissenyar un pessebre significa que es necessita recrear un escenari amb aspecte de poble rural o, si més no, antic. Hem anat a algun poble rural i hem enregistrat parets, porticons i finestres, entre d'altres. El principal inconvenient d'aquest sistema és el de la llum natural i les ombres, a part de l'òptica necessària per fotografiar una paret gran. També varem utilitzar la càmera per les textures de les figures. La Figura 24.a i 24.b mostren dues fotografies que s'han utilitzat com a textura. Per la figura de la vaca i per la porta d'una casa respectivament.

(a)

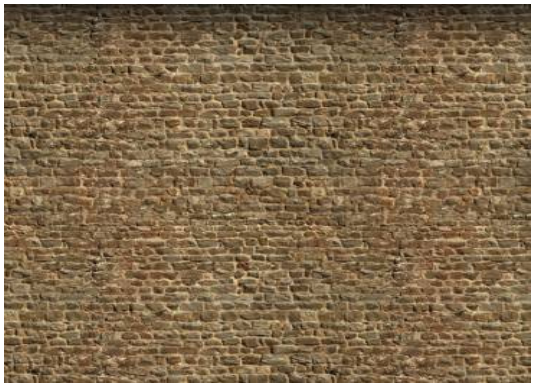
(b)

Figura 24. Fotografia com a textura de figura (a). Fotografia com a textura de porta (b).

2. Un altre sistema és el d'entrar a algun videojoc en mode espectador i mitjançant l'opció Imprimir Pantalla del teclat aconseguir textures. Concretament es va fer amb el videojoc *Counter Strike Source* [CSS], el qual permet volar per tota la pantalla i posicionar-se allà on es vulgui en mode espectador. Cal a dir que els nivells o pantalles d'aquest joc estan ambientats en gran part en pobles antics. Quan es fa un videojoc s'acostuma a crear textures grans amb la repetició de textures més petites intentant dissimular el defecte. Per tant, fent servir aquesta tàctica podia aconseguir grans textures ja treballades per fer-les

servir en un videojoc. Les Figures 25.a, 25.b i 25.c mostren un exemple de textures de videojoc utilitzada pel pessebre.

(a)



(b)

(c)

Figura 25. Textures extretes d'un videojoc (a), (b) i (c).

3. Per últim està Internet. Hi moltes pàgines web on s'hi poden trobar bones textures sense haver de pagar per elles [CGt]. Aquesta opció va ser la més utilitzada ja que els dos casos anteriors no donen molta diversitat, i a més requereixen molt més temps i manipulacions. A continuació apareixen algunes d'aquestes imatges. Les Figures 26.a, 26.b i 26.c són un exemple de textures extretes d'internet.

(a)

(b)

(c)

Figura 25. Textures extretes d'Internet (a), (b) i (c).

En més o menys proporció, pràcticament totes les textures es van haver de tractar. Aquest tractament, en gran part, es va fer per retocar imatges, retallar-les o en casos més complexes, per combinar diferents imatges o fer repeticions. Un exemple de combinació d'imatges és el que podem veure a la Figura 26.a 26.b i 26.c, on a partir d'una textura de sorra i una de gespa, s'aconsegueix una tercera d'un camí. Per a fer totes modificacions i tractaments es va fer servir el software Adobe Photoshop CS2 i CS4.

(a) (b) (c)

Figura 26. Textura pedra (a). Textura sorra (b). Textura combinada. Camí (c).

2.4.3 Modelat

Quan parlem de modelar, en el fons estem dibuixant en un espai de coordenades de 3 dimensions en comptes de dues. Podem entendre el dibuix 3D com la composició de varis dibuixos 2D.

Sempre és molt recomanable que quan es vol modelar un objecte en 3D es dibuixin prèviament uns esbossos, que permetin traslladar aquella idea que tens al cap al 2D des de diferents perspectives. Si hi ha dificultats, apareixeran en aquest moment i seran fàcilment

corregibles. El fet de plantejar clarament els projectes des del principi evita moltes frustracions i la pèrdua de moltes hores de feina.

Quan es vol realitzar un objecte concret no molt complex, pot ser de molta utilitat tenir dibuixos de les diferents vistes, ja que en Blender, com en la majoria de programaris d'edició 3D, es poden posar imatges de fons per dibuixar-hi a sobre (*background image*), tal i com veiem a la Figura 27. Llavors, si per exemple tenim planta, alçat i perfil d'un objecte a modelar, podem fer-ho resseguint les línies de la plantilla que ens hem creat.

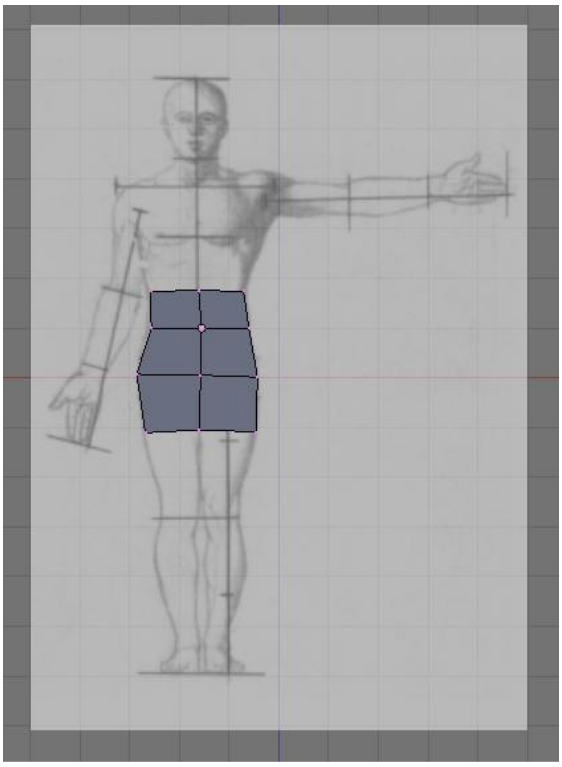
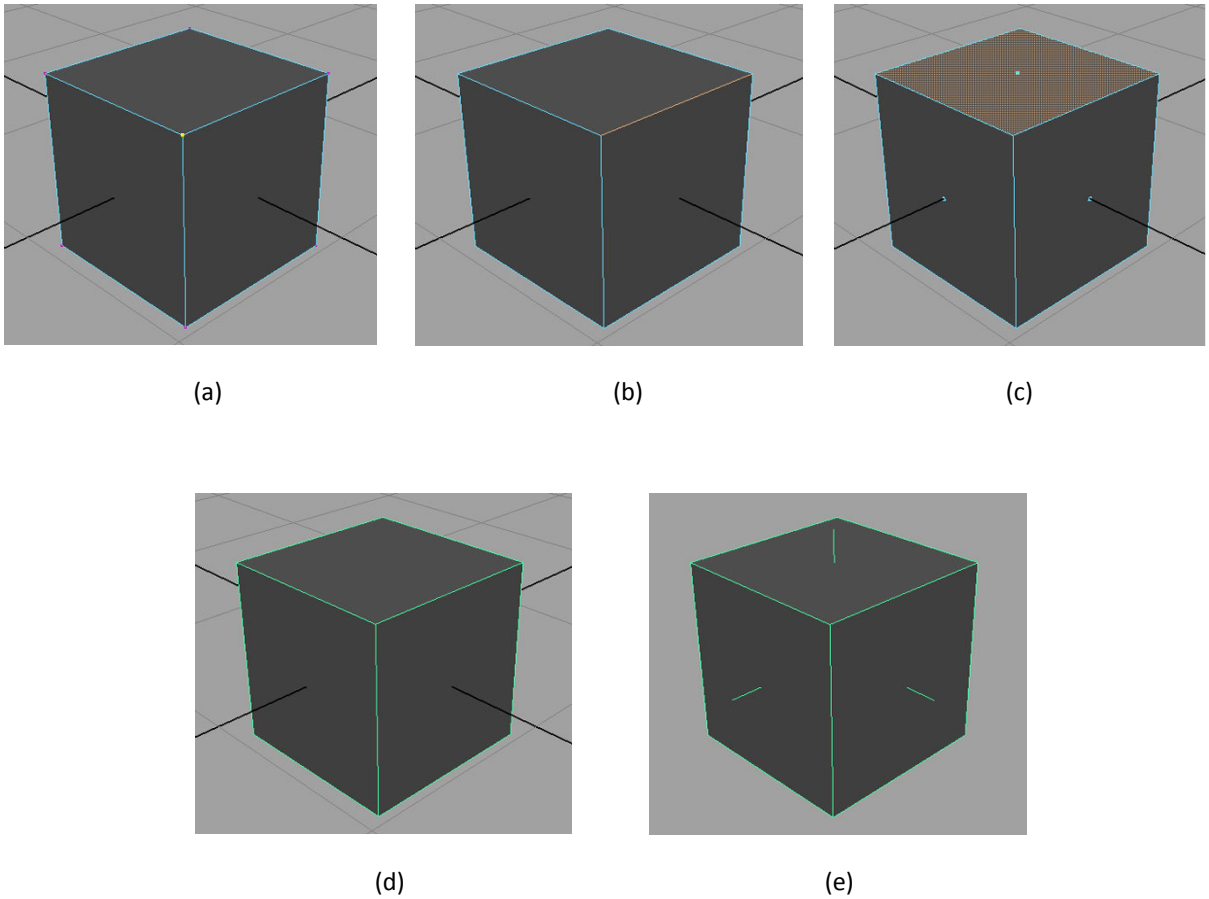


Figura 27. Tècnica de *Background* aplicada a un cos humà..

Per començar a treballar amb un software de disseny s'han de tenir en compte primer quins són els elements que formaran els nostres objectes i per tant la nostre escena. A continuació s'introdueixen els conceptes bàsics que cal saber sobre aquests elements en qualsevol software de 3D [Fmp]:

- Vèrtex: punt en l'espai 3D, que té com atribut una posició en X, una en Y, i una en Z. Els vèrtexs són punts que al connectar-los formen figures poligonals. Figura 27.a.
- Aresta: connecta dos vèrtexs. Les arestes tenen normals per controlar la suavitat o duresa que representen quan uneixen cares. Figura 27.b.
- Cares: Al connectar tres arestes o més s'obté una cara. Aquestes són les que descriuen la superfície del model poligonal. El mínim són tres arestes per formar un triangle, que és el mínim per a què es consideri un polígon. Figura 27.c.
- Malla: Al connectar varies cares s'obté una malla. Les malles són els objectes en sí, i a aquests se'ls hi apliquen les textures més endavant. Figura 27.d.

- Normals: Les normals són vectors que determinen la direcció en la que apunten les cares del modelat. En teoria les cares tenen un sol costat, és a dir, només són visibles quan la normal de la cara està apuntant a la càmera. Tot i això, la majoria de programaris mostren per defecte cares de doble costat. Figura 27.e.



(a) (b) (c)

(d) (e)

Figura 27. Vèrtexs (a). Arestes (b). Cares(c). Malla (d). Normals (e).

Pel que fa a les cares, s'ha dit que a partir de tres arestes ja podem tenir una cara. Rectangles i triangles són els únics polígons que suporta Blender.

En general, als programaris de modelat, es treballa per defecte amb polígons de quatre costats, que són més fàcils de mantenir que els triangles. Els quadrats en sí, estan compostos de dos triangles, que s'oculten a la vista per facilitar la feina. El problema dels quadrats és que quan no estan tots en un mateix pla espacial, el software es confon i no sap com representar la superfície aquesta es veu inestable. La decisió de fer servir triangles o quadrats depèn, en definitiva, de l'ús que es vagi a donar al model. Si el destí del model és convertir-se en superfície de subdivisió (un malla inicial passa a tenir més vèrtexs per aconseguir millors resultats de visualització), lo més recomanable és treballar amb quadrats, ja que l'algorisme de conversió dona millors resultats amb aquests, i amb la resolució del model la majoria de les cares quadrades no patiran problemes dels triangles interiors. Per una altre banda, si el model està destinat a un videojoc per exemple, o a algun altre sistema de visualització on estiguem limitats en la quantitat de cares a fer servir, és recomanable treballar amb triangles, ja que en aquests casos volem tenir el control total sobre la silueta de la figura [Fmp].

Per aquest projecte s'ha optat per fer servir quadrats sempre que sigui possible. Tot i ser una aplicació destinada a ser una espècie de videojoc, és més interessant obtenir millor resultats de visualització que no pas el control total sobre la silueta del objecte. No tindrem en compte l'animació dels objectes per què en realitat els objectes no seran pas animats, sinó una càmera que viatgi per l'escenari. Ademés, el modelat i texturitzat és molt més còmode si es treballa amb quadrats.

El modelat de polígons fa referència a modelar formes definides com col·leccions de vèrtexs connectats per arestes rectes que a l'hora formen cares poligonals. Aquestes formes s'anomenen malles. Editar una malla implica afegir, eliminar o moure vèrtexs, arestes i cares. La Figura 28 ens mostra el que seria una malla d'un objecte senzill, on es pot apreciar els vèrtexs, arestes i cares.

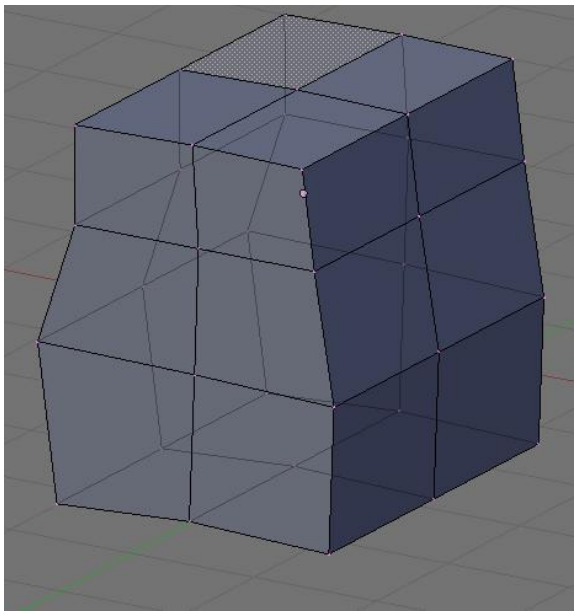


Figura 28. Malla d'una figura. Vèrtexs i arestes formant cares poligonals.

El temps que porta a un ordinador calcular informació 3D sobre una malla depèn principalment del número de vèrtexs de que es compongui. Per aquesta raó, és sempre millor fer servir els menys vèrtexs possibles per representar de manera precisa la forma desitjada de la malla. Existeix també una altre raó per fer això: les malles amb menys vèrtexs són més fàcils d'editar i texturitzar. Si tenen molts vèrtexs, es fa difícil mantenir les superfícies de la malla tan suavitzades e igualades com en malles amb menys vèrtexs.

Tot i existir un conjunt fixa de eines bàsiques utilitzades en el modelat de polígons, existeixen diferents punts de vista per modelar, agrupats en dues grans classes: modelat per extrusió (modelat "poligonal per polígons") i modelat de caixa (*box modeling*). En el modelat per extrusió, el modelador comença amb una petita porció d'un model, que podria ser un polígon, o una aresta, o inclús un punt, i treballa des d'aquest punt d'inici fent servir extrusió i d'altres eines. En el modelat de caixa, el modelador comença amb un senzill objecte 3D, normalment un cub, i fa servir subdivisions, retalls i desplaçaments per modelar la forma en el model desitjat.

Aquest dos punts de vista no són mútuament excloents. De fet, la construcció de inclús un model complexa normalment incorpora elements d'ambdós enfocaments. Tot el modelat de polígons fa ús d'extrusió i subdivisió, així que marcar una diferència clara entre

aquests dos enfocaments de modelat no té molt valor. No obstant, encara que és fundamentalment una qüestió de preferència personal, triar començar amb un cup per començar el model o començar amb un senzill polígon o vèrtex, determinarà els següents passos que necessita per crear el model.

En aquest projecte s'ha treballat tant amb una tècnica com l'altre, i en la majoria dels casos fent servir les dues simultàniament i ajuntant després les diferents malles fer formar-ne una de sola.

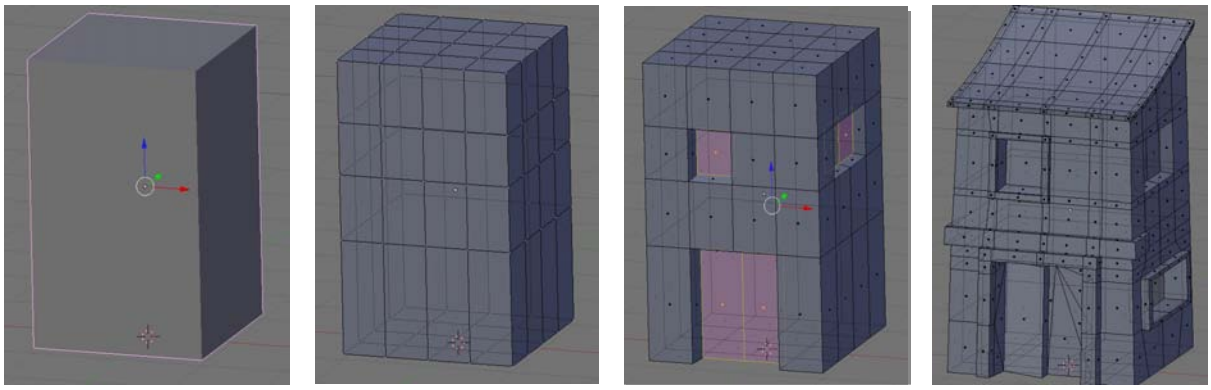
Un exemple ben clar d'un objecte fet amb modelat de caixa seria el de una casa o edifici qualsevol. Inicialment es comença amb un objecte cúbic, al qual se li van fent divisions i retalls, per exemple, per crear els forats de les finestres o les portes. A continuació les figures mostren quin seria el procés per crear l'estructura bàsica d'una casa com les del pessebre.

Comencem amb un rectangle amb les dimensions que vulguem que tingui el nostre objecte aproximadament. En aquest exemple es comença amb un cup, però Blender disposa d'altres primitives com el con, el cilindre, l'esfera, etc. que també poden ser un bona elecció depenent de l'objecte final desitjat.

Com mostra la Figura, el següent pas és fer les subdivisions necessàries a la malla per començar a donar forma. És molt important no fer més subdivisions de les necessàries, ja que cada veda que en fem una estem augmentant el número de vèrtexs que formen la malla. És recomanable no fer totes les subdivisions al inici, sinó anar-ne fent a mesura que les anem necessitant.

Un cop arribat a aquest punt s'ha de començar a jugar amb la tècnica d'extrusió per anar donant la profunditat adequada a les cares. Aquesta tècnica simplement duplica aquelles vèrtexs, cares i arestes que estiguin seleccionats creant nova geometria al voltant de les noves cares que es formen. Com es pot apreciar a la Figura, això és molt útil per la creació de finestres, portes i moltes altres formes. De la mateixa manera que en pas anterior, s'ha d'anar amb molt de compte de no crear nova geometria extra.

Per últim ja es pot començar a fer el modelat directe, que consisteix en anar creant les formes a la malla que volem que presenti l'objecte. Aquesta part és la més complexa i amb infinites possibilitats. Si s'ha de seguir amb les divisions i extrusions seguirem vigilant de no crear un objecte més pesat del necessari. La insistència en l'optimització de la geometria es comença a veure demostrada com més detall va tenint la malla. És evident que l'objecte comença a tenir moltes més arestes i punts dels que voldríem. Cal a dir que el sistema de modelat poligonal per polígons dona més precisió en aquest aspecte i no es creen divisions innecessàries en parts de la malla que no ho necessiten. Tot i així, en qualsevol moment es poden treure vèrtexs reeditant l'objecte i creant noves cares. Un factor que no s'ha comentat, però que en aquest projecte és molt útil, és el d'eliminar totes aquelles cares que després no seran visible, com podrien ser el sòl de la casa o qualsevol part de la geometria que quedi dins del edifici. A la Figura 29.a es veu la primitiva inicial, la qual modifiquem per aconseguir les dimensions i formes genèriques de l'objecte. A continuació, a la figura 29.b es mostra com la tècnica de divisió de malla ens ajuda crear més geometria a la figura per poder-la manipular millor. La següent tècnica utilitzada és la de extrusió, com es pot veure a la Figura 29.c, ens permet endinsar o treure parts de la geometria per crear forats o sortints. A la Figura 29.d s'observa el que es pot aconseguir mitjançant la manipulació de la malla fent servir només aquestes tècniques.



(a) (b) (c) (d)

Figura 29. Primitiva base (a). Subdivisió inicial (b). Tècnica d'extrusió (c). Detallat del objecte (d).

Per explicar aquest exemple s'ha dissenyat un casa com un mateix objecte, però a la pràctica, si per exemple volem que algunes parts siguin de fusta i d'altres de totxana, el més adient és crear la casa amb objectes separats. D'aquesta manera, el texturitzat de cada part, serà molt més senzill i precís. En l'exemple anterior, tant el sostre com les finestres, portes i vigues s'haurien modelat per separat, ja que al final del procés totes les parts es poden ajuntar, i si es desitja, en una de sola. A part, el fet de crear varies malles per cada peça de la casa redueix molt tota aquesta malla extra.

2.4.4 Texturitzat

La tècnica principal que es pot utilitzar per a texturitzar objectes és el mapejat UV. Consisteix en l'assignació d'una textura 2D a les cares d'un model 3D seguint les coordenades de la superfície (les coordenades UV). És un dels mètodes més populars i efectius per texturitzar personatges animats, però a la vegada és un dels més difícils i tediosos de fer servir quan es volen obtenir bons resultats. Blender suporta texturitzat UV, però no dona masses facilitats d'ús. Per això és comú que per realitzar el mapejat es recorri a software extern com LithUnwrap3D, UVMapper, etc.

El desplegat de cares consisteix en "desplegar" o "desenrotllar" el model en l'espai 2D (unwrap), de manera que podem assignar les coordenades de textura a cada cara de l'objecte. Aquest mètode requereix manipulació de la malla, i en alguns caos prou complicada, però permet obtenir molt bons resultats. A continuació explicarem l'aplicació de textures pas a pas.

El mapejat UV, consisteix en ajustar una imatge com a textura a l'objecte, de manera que si animem l'objecte, la textura vagi amb ell. És molt idoni pel motor de jocs de Blender, els recorreguts virtuals o walktrough o simplement per aconseguir una imatge fixa molt realista. En alguns casos pot reduir molt el temps de modelat, ja que proporciona algunes tècniques per mostrar formes sense la necessitat de modelar-les. Per la realització d'aquest projecte la raó més important per fer servir UV és la del realisme que proporciona, ja que l'animació es realitza en un software diferent.

La finestra de l'editor UV ens permet aplicar la textura a les cares per separat o a la malla completa. Les Coordenades UV són 2D i es diuen així per diferenciar-les de les Coordenades XYZ. A la Figura 30 podem veure els icones d'edició del editor per texturitzar.

Figura 30. Barra d'eines del editor UV Faces

Per a realitzar el mapejat UV, el més important és que la malla estigui ben modelada, és a dir, amb la posició de les normals correctes i sense que hi hagi vèrtexs dobles, doncs tot això podria crear formes entranyes a la superfície. En el cas del duplicat de vèrtexs, Blender disposa de la tècnica *Remove Doubles*, la qual ens permetrà solucionar aquets problema de manera senzilla i controlada.

Quan ja tenim un objecte modelat passem a la fase del desplegat. Hi ha varies maneres de fer-ho, però es poden dividir en dos sistemes: amb costures i sense. La costura consisteix bàsicament en marcar un seguit de vèrtexs o arestes del objecte de manera que es pugui retallar la malla a l'hora de fer el desplegat. L'altre opció és més ràpida però és el programa el qui tria com es fa els desplegat, dins, això sí, d'un seguit de mètodes o algorismes entre els que l'usuari pot triar mitjançant un menú. L'opció amb costura també passa per la tria d'aquests algorismes de bolcat després d'haver marcat la malla. La Figura 31 mostra els diferents tipus de desplegat que podem triar.

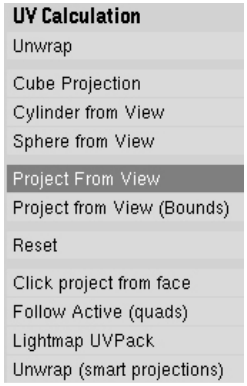


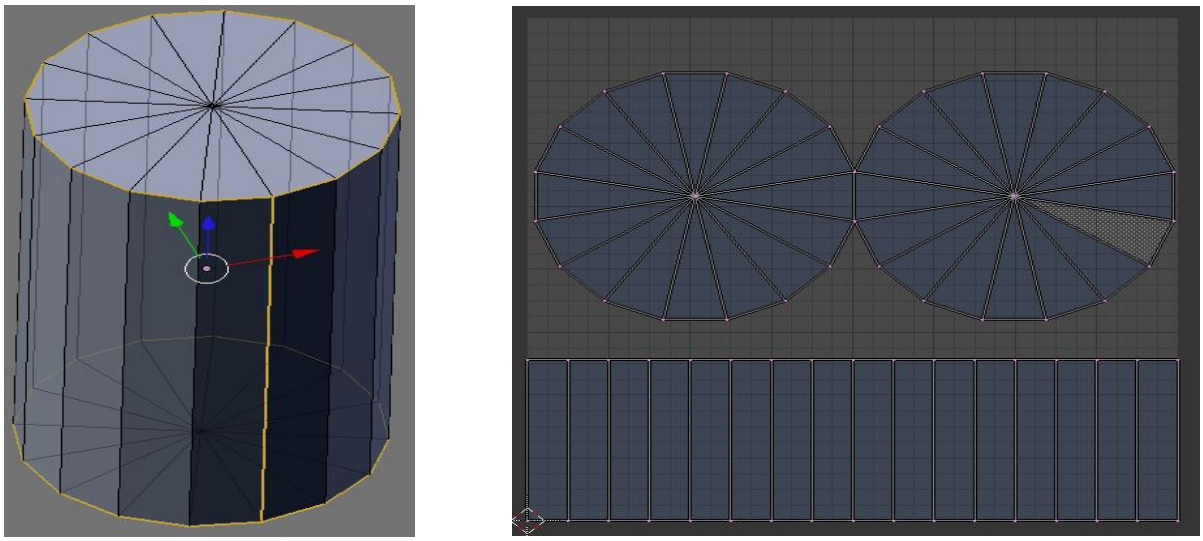
Figura 30. Llistat d'algorismes de desplagat de malla.

El desplegat amb costura és ideal per objectes cilíndrics. En aquest projecte s'ha fet servir bàsicament pels troncs dels arbres. Tot seguit explicarem els dos tipus de desplegat que s'han dut a terme en el projecte.

1. Desplegant un cilindre. Aquest sistema de retall s'assembla molt als retallables en paper de figures geomètriques que tots hem fet quan érem petits i després els muntàvem en 3 dimensions. Tot i explicar la tècnica per a cilindres, es evident que també és aplicable a qualsevol objecte, però la feina extra que comporta no surt a compte en la resta de formes, a no ser que l'objecte tingui parts corbes i complexes.

Per fer-ho es selecciona un eix vertical sobre el qual es desplega el cilindre, i els dos cercles que composen les tapes de d'alt i de a baix, si n'hi ha. A la Figura 31.a veiem el perfil ressaltat per una línia taronja.

Les cares del objecte queden totalment desplegades sobre l'editor 2D de UV, on novament, si cal, es poden modificar els vèrtexs de la malla i per últim assignar la/les imatge/es desitjades. La Figura 31.b ens ho mostra.



(a) (b)

Figura 31. Costura d'un cilindre (a). Desplegat amb costura d'un cilindre (b).

Un cop arribats a aquest punt es pot triar entre posar una imatge directament i quadrar-la amb els vèrtexs de la malla o optar per exportar l'objecte desplegat en format targa (.tga), d'aquesta manera es pot obrir la imatge des de qualsevol editor d'imatges i aplicar la pintura o fotografies amb la comoditat de que en tot moment es sap quina zona estem pintant. De la mateixa manera es torna a guardar la nova imatge sense les línies de la malla per aconseguir una textura que encaixa perfectament amb les cares del nostre objecte. La Figura 32 mostra el tronc d'un arbre després de passar per tots aquests passos.

Figura 32. Tronc texturitzat amb desplegat UV utilitzant costures de malla.

2. Desplegat de la resta d'objectes. Pels objectes no cilíndrics o més planars n'hi a prou amb anar seleccionant les cares que ens interessi texturitzar i desplegar amb l'algorisme *Project From View* ó *Project From View (Bounds).* Aquests projecten els vèrtexs amb la mateixa vista amb que l'usuari està veient la figura en aquell moment.

Els objectes quadrats, parcialment quadrats o plans són, al cap i a la fi, la majoria dels objectes de que es compon l'escenari, ja que aquest està format per cases, i edificis en

general, els quals estan dissenyats majoritàriament amb geometria plana. No ens oblidem dels personatges, el seu texturitzat és un cas a part, i s'explica en un altre apartat.

Anem a veure un exemple del desplegat per cares o sense costura. Ho farem amb l'exemple d'un dels edificis modelats pel pessebre. Es tracta d'anar canviant de posició al voltant de la figura per poder anar projectant les diferents vistes d'aquesta. És un procés que en alguns casos pot ser una mica costos, ja que podem arribar a tenir moltes cares. El divers ventall d'eines que ens ofereix Blender és indispensable en aquests casos. Eines com ocultació de cares, selecció inversa, selecció múltiple, etc. poden fer-nos la vida més fàcil a l'hora de marcar la part de malla que ens interessa. A demés, hi ha una connexió entre la finestra 3D de visualització i la finestra 2D de UV. Per exemple, es pot triar l'opció "on" al marcar els vèrtexs de la finestra de UV, i també queden marcats a la de 3D. Això pot ser de molta utilitat per comprovar que no s'ha oblidat de desplegar alguna cara.

Tal i com es mostra a la Figura 33, partim d'un objecte pràcticament cúbic. A continuació, del que es tracta és de posicionar la càmera (en projecció ortogonal) perpendicular a una de les parets de la casa.

En aquesta posició de càmera, es seleccionen totes les cares que quedin perpendiculars a la normal de la càmera i es tria l'opció de desplegat *Project From View* per copiar els vèrtexs seleccionats al editor 2D del UV maping. Pot haver-hi cares que formin petites corbes o formes no planes. Simplement s'ha de tenir més cura a l'hora de seleccionar-les, triant sempre la vista on més tros es vegi pla.

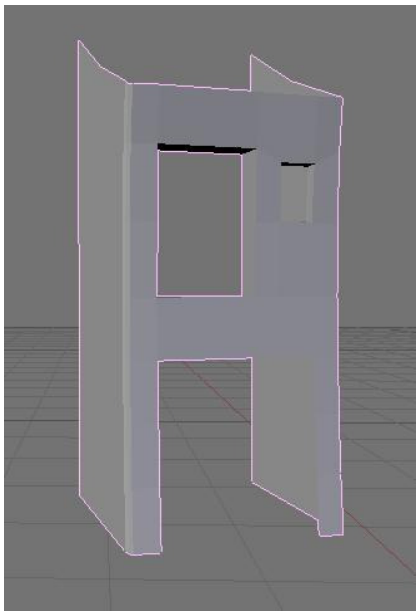


Figura 33. Objecte a texturaritzar.

Després de repetir aquests passos per a totes les vistes que interessi s'obté una projecció completa de totes les cares del objecte. Normalment quan parlem de parets amb totxanes o amb algun dibuix concret s'ha de tenir molt en compte que quan es miri una cantonada de la paret ha de donar l'efecte de que la totxana o dibuix continua d'una paret a un altre, tal i com s'apreciaria al món real. Per aconseguir aquest efecte farem sevir el propi editor 2D. Manipulant els vèrtexs podem aconseguir per exemple que el dibuix a les cantonades tingui sentit. Com es pot veure a la Figura 34, el resultat serà com si s'hagués xafat la casa en un únic pla.

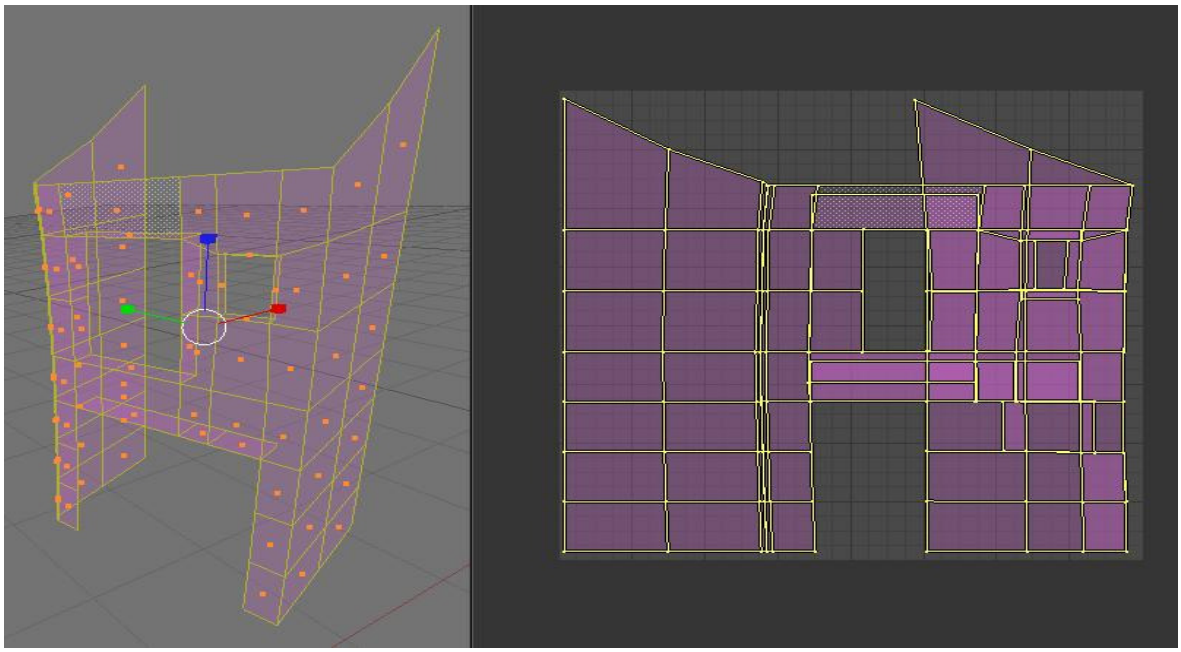


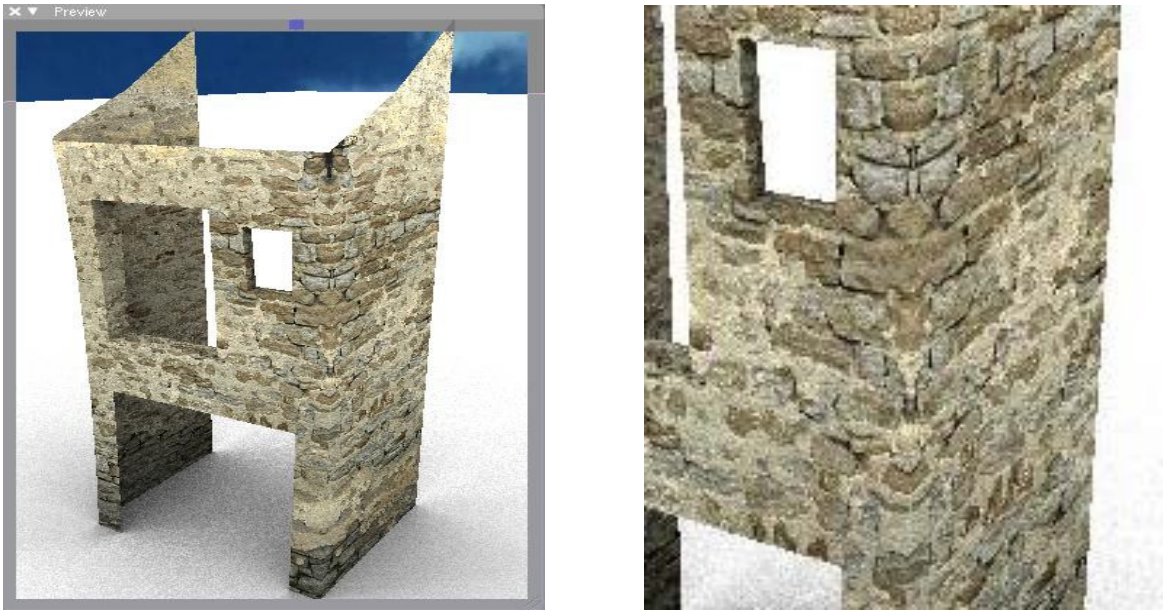
Figura 34. Desplegat.

Un cop arribats a aquest punt, només falta aplicar la textura. Podem triar entre exportar una imatge .tga amb el dibuix de la malla o posar directament la imatge sobre aquesta finestra. No serà necessari exportar, en aquest cas, ja que només es vol aplicar una única textura a tot l'objecte. És molt important que la imatge triada sigui de qualitat i a poder ser amb una mida raonablement gran. Això també donarà molts millors resultats. L'editor ens permet visualitzar la malla desplegada sobre la textura triada, de manera que serà més fàcil ajustar els vèrtexs allà on vulguem. La Figura 35 mostra el que l'usuari veu després de seguir els passos anteriors.

Figura 35. Desplegat amb textura.

La tècnica d'exportar el desplegat pot arribar a ser de molta utilitat en els casos on es necessiten imatges molt grans, ja que amb un editor de imatges podem arribar a simular una sola textura amb moltes de més petites, sense que es noti el canvi d'una a l'altre.

Per últim a la Figura 36.a es pot veure el resultat final de fer tot el procediment anterior. Aquí és pot apreciar el realisme que dona el fet de fer servir fotografies i de treballar bé la posició dels vèrtexs de la malla. A la Figura 36.b es pot apreciar perfectament com a les cantonades de la casa la totxana continua d'una paret a una altre. És molt probable que aquest mètode repeteixi el dibuix d'una de les parets com el invertit de l'altre. Per evitar-ho s'hauria de treballar amb textures molt homogènies on no es poguessin donar aquests casos, o tenir una textura prou gran com per desplegar-hi tota la malla sense tenir que repetir el dibuix, però no sempre és pràctic o possible fer-ho d'aquesta manera. Tot i així els resultats són prou bons i molt millors que si s'apreciés un tall del dibuix cada vegada que comencés una paret nova .



(a) (b)

Figura 36. Resultat final del model (a). Detall de la cantonada (b).

2.4.5 Escanejat de les figures

L'escena modelada podria correspondre a qualsevol poblet, sigui català, europeu, o palestí. El que confereix l'escena com a pessebre és, a la fi, la inclusió de figures nadalenques, en aquest cas les figures de pessebre, com els reis macs, el caganer, etc. i la inclusió d'escenes tradicionals com podria ser l'anunciament de l'àngel o el mateix naixement.

A part de les figures típiques del pessebre l'escena havia de tenir figures ambientades en la època i el decorat, per tant, a demés s'hi volien posar altres personatges o animals per donar vida al que s'estava modelant.

A partir d'aquí, modelar personatges pot ser molt complicat i esdevenir un procés molt llarg. A Internet moltes vegades és poden trobar models 3D gratuït, però el que nosaltres necessitàvem no és podia trobar fet i ho havíem de crear nosaltres.

Finalment, l'opció més viable va sorgir de la possibilitat de poder fer servir un escàner en 3 dimensions, el qual ens permetria obtenir figures 3D a partir de simples figures reals de pessebre.

Els sistemes comercials que poden digitalitzar objectes 3D són relativament cars i per tant poc assequibles per la majoria de possibles usuaris. L'any 2007 apareix el *David Scanner*, un nou software capaç de fer-ho notablement, sense la necessitat de gastar-se tants diners.

El científic Dr. Simon Winkelbach i Sven Molkenstruck, científics investigadors del Institut de Robòtica i Informàtica de la Universitat Tècnica Brunswick, juntament amb el Director del Institut Prof. M. Frederick Wahl van fer possible aquesta eina tan simple i eficaç. El 14 de Setembre de 2006 a Berlín, van rebre el prestigiós Gran Premi de la Deutsche Arbeitsgemeinschaft für Mustererkennung [Dscn]. La Figura 37 mostra la caixa del producte.

Figura 37. Programari David Scanner.

Els requisits més importants d'aquest software és que ha d'anar acompanyat d'una càmera (per exemple càmera web) i un làser. Cap d'aquests objectes fan que el cost total s'elevi gaire, ja que també els podem aconseguir per preus raonables.

El procés de funcionament és prou delicat, però no pas complicat. Si volem escanejar un objecte hem de tenir una cantonada formada per dos plans que a l'hora formin un angle recte, on hi posarem una plantilla de punts. Aquesta plantilla es fa servir per calibrar la càmera, la qual ha d'enfocar el centre del plans. A continuació, la Figura 38 mostra la interfície del David Scanner per fer el calibrat de la càmera.

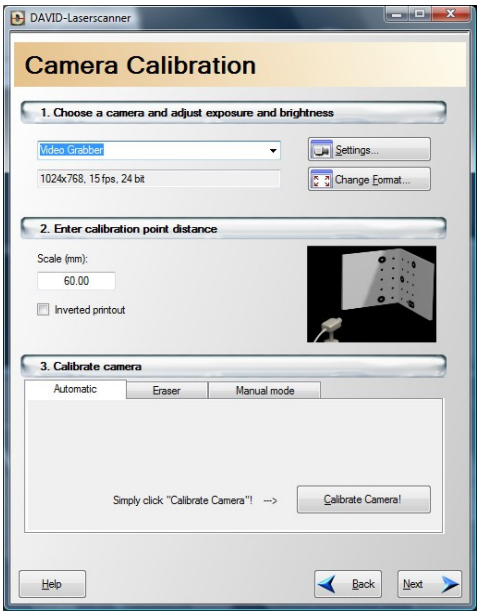
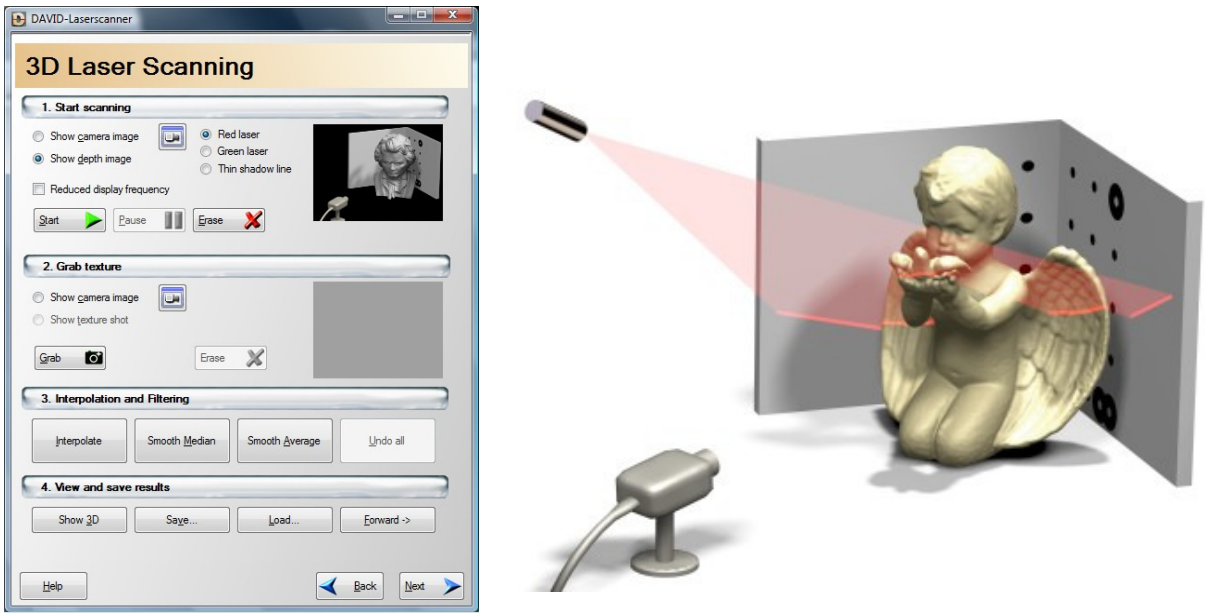


Figura 38. Menú del calibrat de la càmera.

L'objecte que vulguem escanejar s'ha de posat entre els dos plans tal i com es mostra a la Figura 39.b, i a poder ser sobre alguna espècie de base giratòria, que ens permeti mantindré l'angle de rotació cada vegada que fem una part del objecte. La Figura 39.a mostra la interfície que ens guiarà en l'escaneig amb el làser.



(a) (b)

Figura 39. Menú del escaneig del làser (a). Representació del escaneig (b).

El principal avantatge és que no es necessita una estructura complexa d'exploració mecànica. L'inconvenient és que el làser a la mà pot moure's.

David Scanner genera en temps real les dades en 3D i ho mostra en la pantalla del ordinador. Anirem passant la línia del làser per tot l'objecte fins que estiguem satisfets amb

el resultat. Repetirem aquest procés per diferents angles per poder construir l'objecte 3D. Cada vegada, el programa crearà un fitxer .obj amb la malla escanejada en cada angle. A la Figura 40 podem veure el resultat que se'ns mostra a la interfície del escàner al passar el làser per la figura.

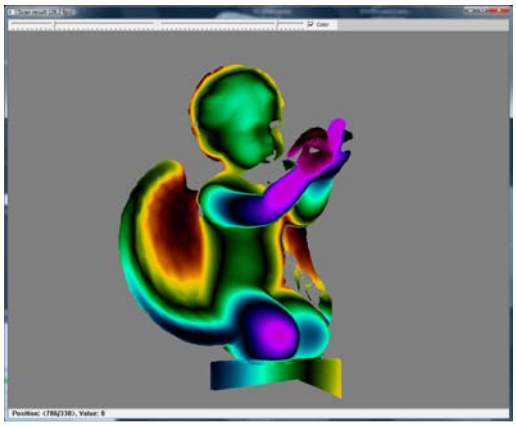


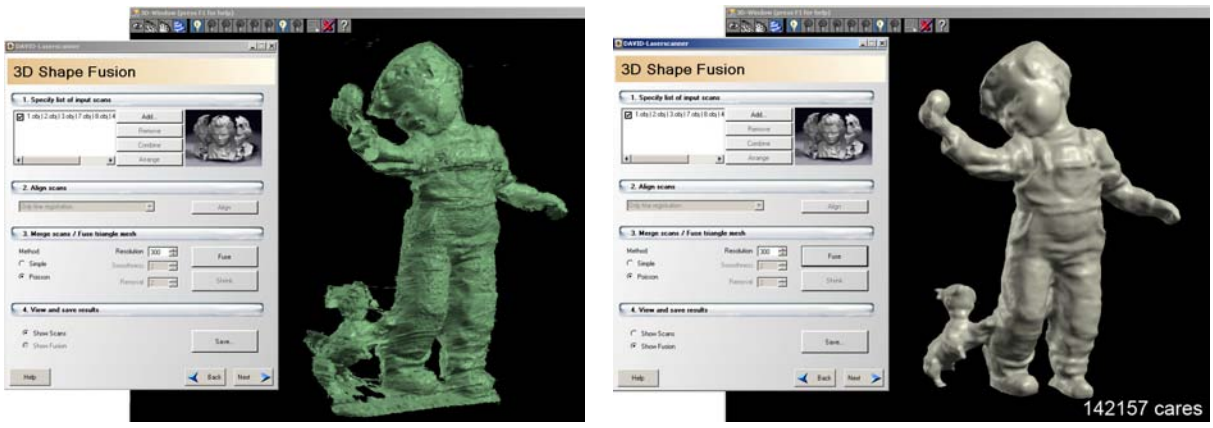
Figura 40. Resultat d'aplicar el làser.

A continuació el propi software ens permet construir l'objecte 3D final a partir de la fusió dels diferents .obj creats anteriorment. A la Figura 41 es poden apreciar tots els escanejos fets en cadascuna de les rotacions de la peça.

Figura 41. Parts de la figura escanejada abans d'ajuntar-les en una de sola.

Un cop es fusionen les parts, només falta suavitzar la malla resultant amb el nivell de detall que creguem convenient i tornar a exportar l'objecte final en format .obj. Podem veure el resultat de la fusió a la Figura 42.a abans de ser suavitzada, i el resultat després de suavitzar a la Figura 42.b.

El David scanner també té la propietat de que pot posar textura al voltant de la malla simplement prenent una imatge amb la mateixa càmera per cadascun dels angles. No varem poder aprofitar aquesta útil funció per què la càmera era en blan i negre; a demés, el detall de la malla que genera el scanner és massa complexa i al reduir-la dins del Blender es perd la relació de la textura amb la malla, ja que l'estem modificant.



(a) (b)

Figura 42. Fusionat dels objectes (a). Fusionat dels objectes amb suavitzat (b).

Per tant el texturitzat de les figures es va haver de fer com els demés objectes modelats pel pessebre; desplegant la malla del objecte i texturitzant les cares amb fotografies en color de les mateixes figures reals. La Figura 43 mostra la fotografia de la peça real de pessebre, que a demés ha esdevingut la pròpia textura del objecte en 3D.

Figura 43. Fotografia de la figura de pessebre original.

A la Figura 44 podem veure el resultat final d'una figura de pessebre després d'haver estat escanejada, exportada a Blender com a .obj i texturitzada. Com podem veure el resultat és prou bó.

Figura 44. Renderitzat final d'una figura del pessebre.

2.5 Realització del Vídeo final.

Com s'ha comentat al inici d'aquest escrit, el producte final d'aquest projecte està destinat a l'Associació Pessebrista de Sabadell, ja que el diorama virtual es vol exposar conjuntament amb la resta de diorames tradicionals, a la exposició de pessebres que té lloc cada any durant els dies de Nadal.

Per aquest primer any s'ha realitzat un recorregut per dins de la escena, no interactiu, des del mateix Blender i conseqüentment se n'ha tret una seqüència de vídeo final. Tot i que l'any vinent es pretén exposar tot el projecte interactiu (conjunt de projecte de disseny i projecte de programació), ara per ara s'ha dut a terme aquest vídeo per poder exposar una primera versió del Pessebre Virtual veure la reacció del espectador de l'exposició de Sabadell al trobar-se un diorama d'aquestes característiques.

El primer que s'ha fet és donar a l'escena una il·luminació adequada per obtenir uns bons resultats. Blender disposa de varis tipus d'il·luminació i llums en general. Per exemple, per aclarir les zones on hi alguna torxa, s'ha utilitzat el tipus de llum "àrea". Com es pot veure a la Figura 45, aquest tipus de llum marca un requadre per la zona que es vol il·luminar.

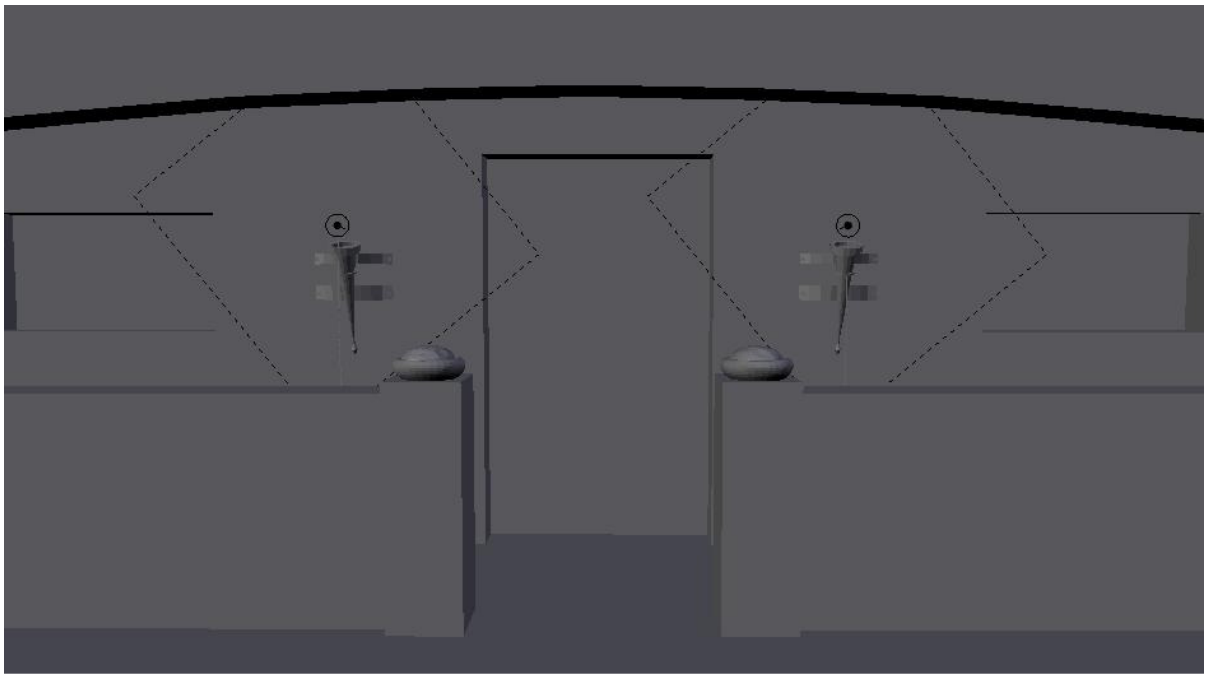
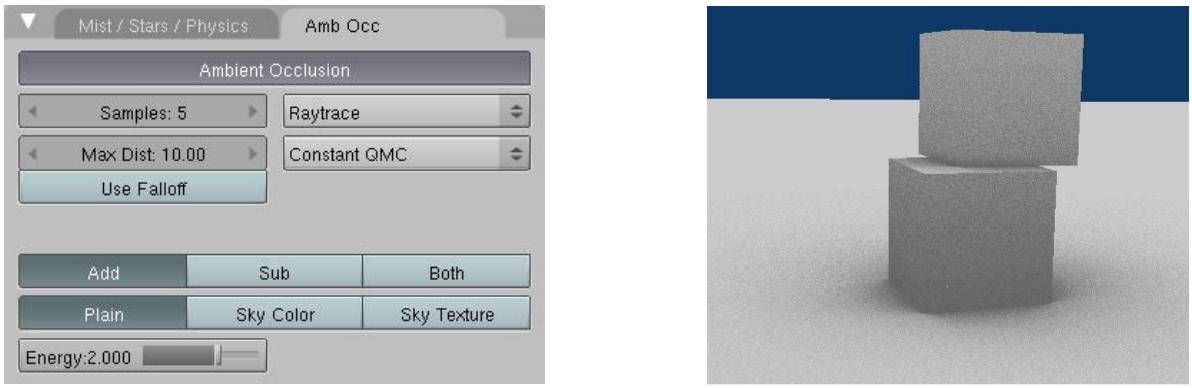


Figura 45. LLums tipus “àrea” per les torxes.

Llavors, per a la il·luminació global de l’escena s’ha triat l’efecte “ambient ocusion”, que calcula la distància que hi ha des dels objectes fins al terra en funció d’una intensitat de llum i uns paràmetres donats. La Figura 46.a mostra les opcions pel “ambient oclusion”, mentre que la Figura 46.b ens mostra el resultat d’aplicar aquesta il·luminació a un parell de cubs contra un pla que fa de terra.



(a) (b)

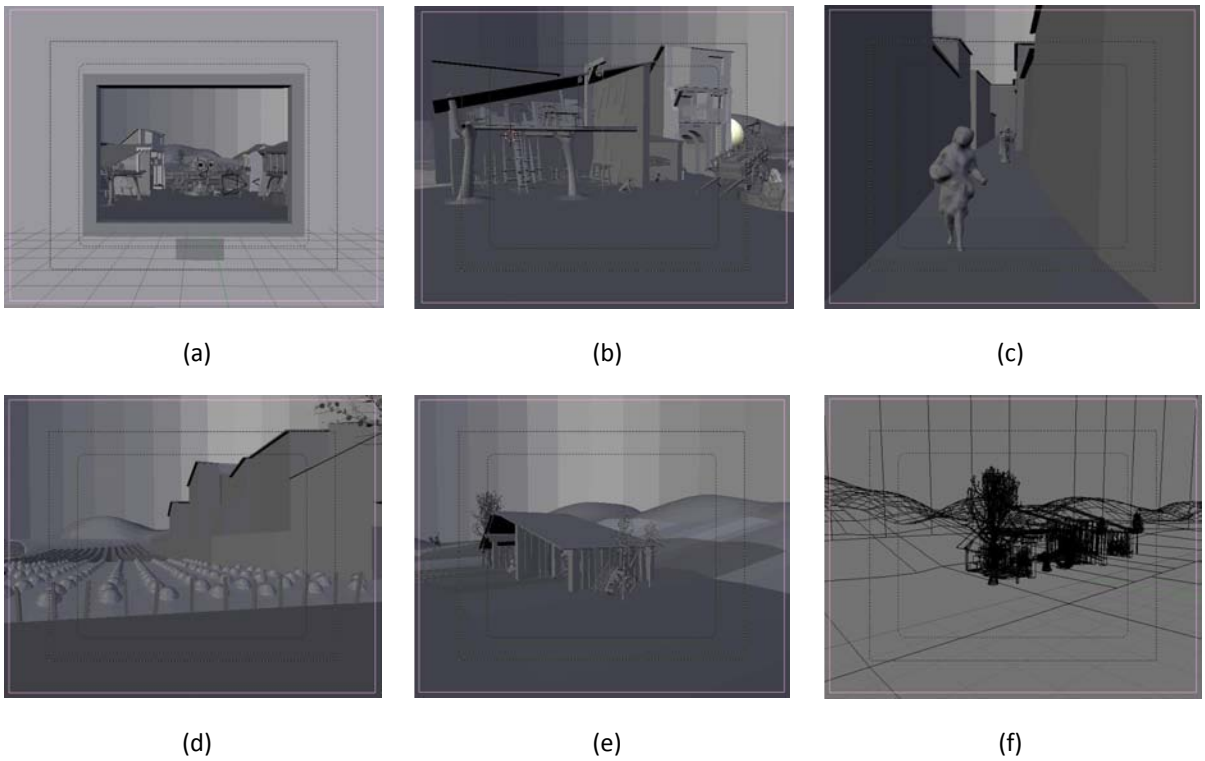
Figura 46. Menú *ambient oclusion* (a). Efecte *ambient oclusion* (b).

Normalment, a les exposicions on s’han de veure moltes obres, i els espais són reduïts, l’espectador no pot romandre molta estona contemplant una sola obra, ja que col·lapsaria el flux de les demés persones que han de veure l’obra en qüestió i les següents a aquesta. Per aquest motiu el vídeo final del Pessebre no podia durar més d’uns 2 minuts aprox.

Per una altre banda, la velocitat de la càmera no pot ser molt elevada, ja que l’espectador ha de poder contemplar tots els detalls de l’escena i no s’ha d’estressar.

Per aquests dos motius es va decidir muntar un vídeo a partir de varis vídeos més petits realitzats amb diferents càmeres. D'aquesta manera es podia arribar a tots els racons de l'escenari sense necessitat de allargar el vídeo més del compte.

La Figura 47 ens mostra el inici del recorregut de les 6 càmeres que s'han creat, de les quals n'han sortit 6 escenes de vídeo diferents que després s'han ajuntat.



(a) (b) (c)

(d) (e) (f)

Figura 47. Càmera 1 (a). Càmera 2 (b). Càmera 3 (c). Càmera 4 (d). Càmera 5 (e). Càmera 6 (f).

Per fer els moviments de les càmeres s'han utilitzat les corbes IPO, les quals permeten moure qualsevol objecte de Blender només dient la posició del objecte en un frame determinat. Cada vegada que gravem una posició, es calcula el recorregut que hi fa l'objecte des del frame anterior guardat fins al actual. L'avantatge de treballar amb aquestes corbes és que mitjançant el propi editor de corbes IPO de Blender, es pot modificar el recorregut movent els nodes que es creen al guardar una posició en un instant de temps. Les corbes IPO permeten no només modificar la posició d'una objecte, sinó també la rotació i l'escalat [BCI]. En el nostre cas, els objectes són les pròpies càmeres i a la Figura 48 podem veure-hi les corbes IPO de la càmera 1. Com es pot veure a la imatge, nosaltres només hem gravat dades relacionades amb la translació (Loc) i la rotació (Rot).

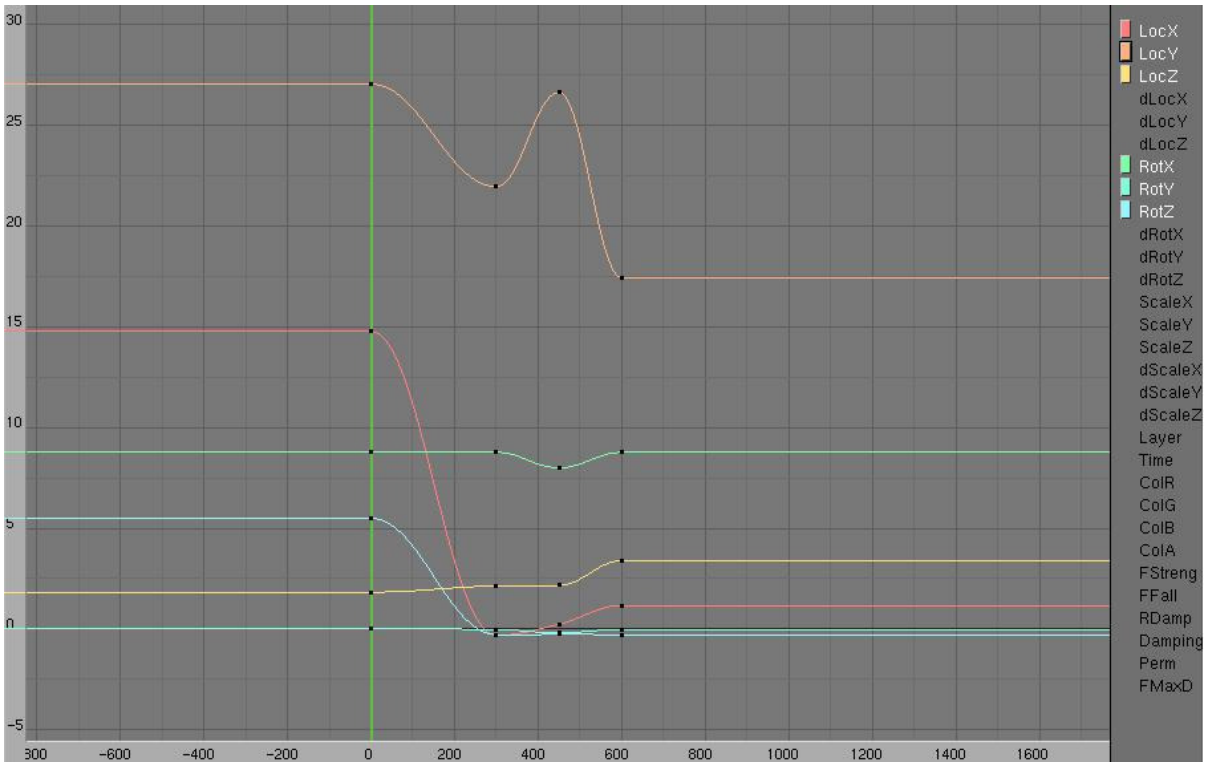


Figura 48. LLums tipus "àrea" per les torxes.

Al finalitzar el recorregut de l'ultima càmera, just rere l'establia on s'hi representa el naixement, varem pensar de mostrar els "crèdits" del projecte a partir de l'animació de text 3D utilitzant la mateixa tècnica que pel moviment de les càmeres. A la Figura 49 es pot apreciar un dels últims fotogrames del recorregut de l'ultima càmera, on pareixen els "crèdits" esmentats.

Per ajuntar els 6 vídeos en un de sol, s'ha utilitzat el software *Virtualdub* que permet capturar i processar vídeo [Vdb]. És un programa gratuït per plataformes de Windows de 32 bits i ens ha estat molt útil, ja que d'una forma molt senzilla, hem pogut ajuntar tots els fotogrames de totes les escenes.

El resultat final és un clip de vídeo en format avi comprimit amb DivX. La seva mida és de 47,7 Mb i la resolució és de 1280 x 1024 píxels.

Figura 49. Vista de l'animació dels crèdits des de la càmera 6.

3 Resultats

Els resultats del projecte han estat prou bons. La sensació de realisme és probablement millor a la esperada, ja que al iniciar el projecte no estàvem segurs de poder trobar textures tan bones de forma gratuïta.

Pel que fa al moviment de les càmeres, els resultats també han estat molt acceptables, ja que els canvis d'escena i moviments han quedat molt suaus i han complert amb el seu objectiu de mostrar tot l'escenari en menys temps i sense deixar-ne cap detall. Cal a dir que quan una càmera fa una rotació sobre l'eix vertical, la imatge perd nitidesa, tot i no haver fet moviments massa bruscos.

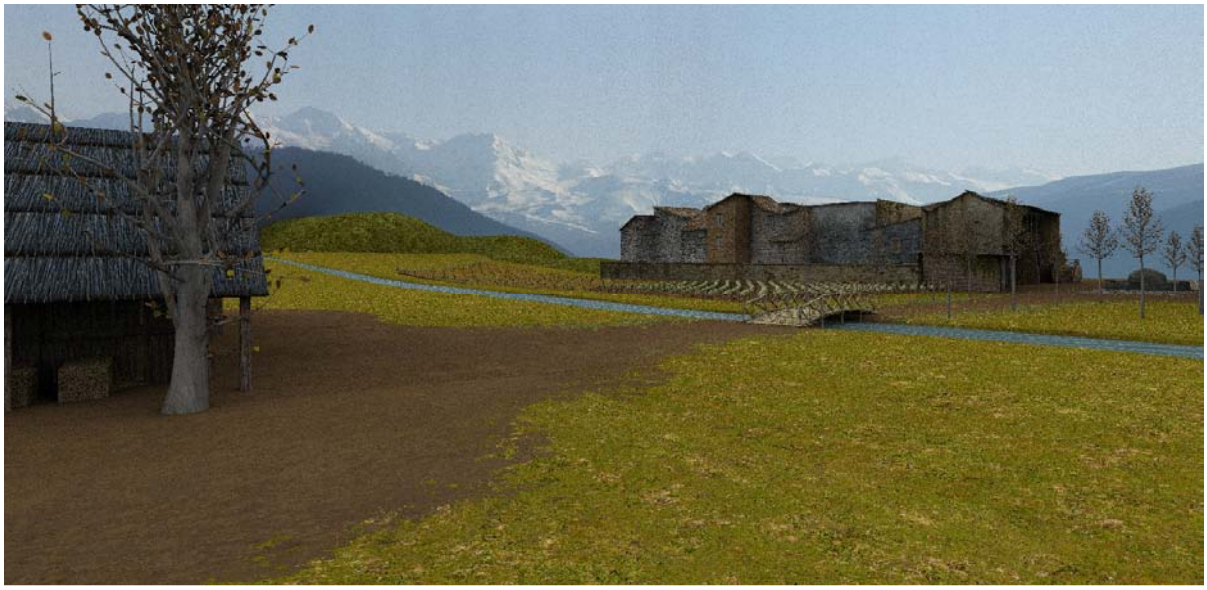
Comencem doncs amb els resultats finals de la seqüència de vídeo, i tot seguit mostrarem resultats d'exportar el pessebre des de Blender fins l'entorn de Visual C++. A continuació, a la Figura 50 es poden veure algunes imatges del diorama.

(a)

(b)

(c)

(d)



(e)

Figura 50. Vista inicial fora del diorama (a). Carreró (b). Centre del poble (c). Establia (d). Vista des de fora del poblat (e).

Les vistes des d'una posició llunyana poden semblar resultants, però si observem alguns detalls, veiem com realment s'ha aconseguit grans resultats. La Figura 51 mostra alguns dels resultats en objectes més concrets.

(a)

(b)

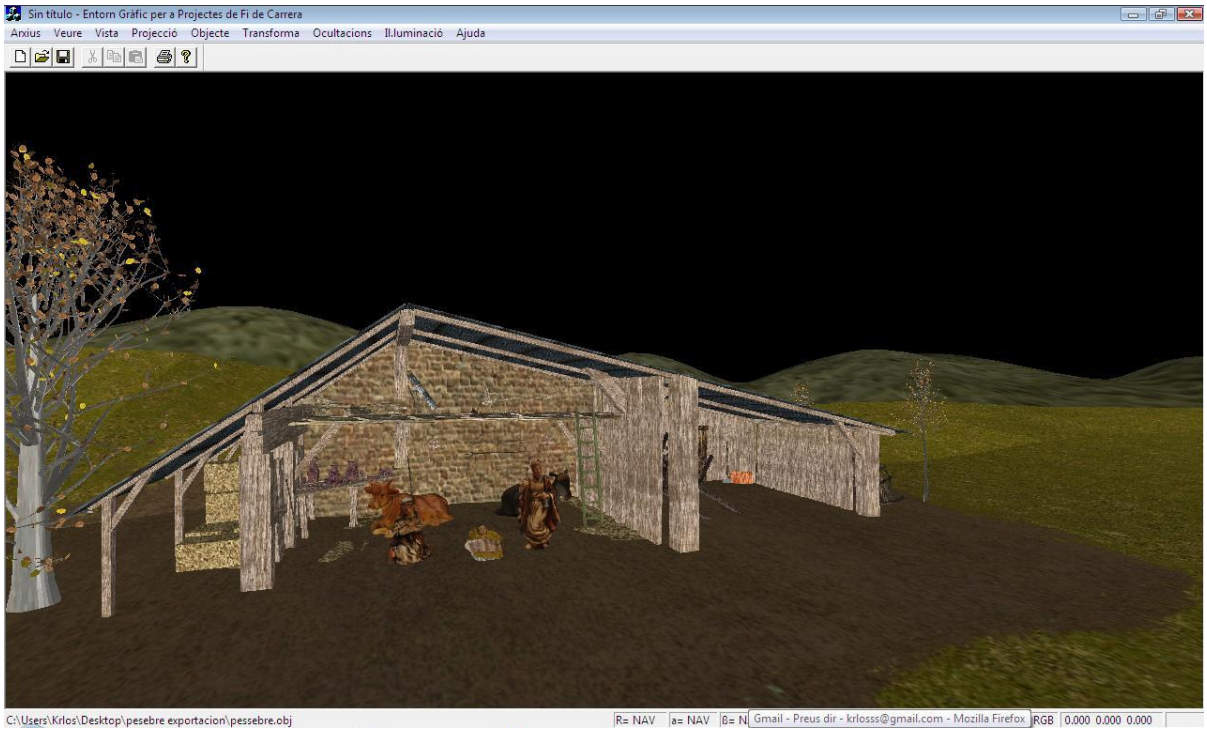
(c)

(d)

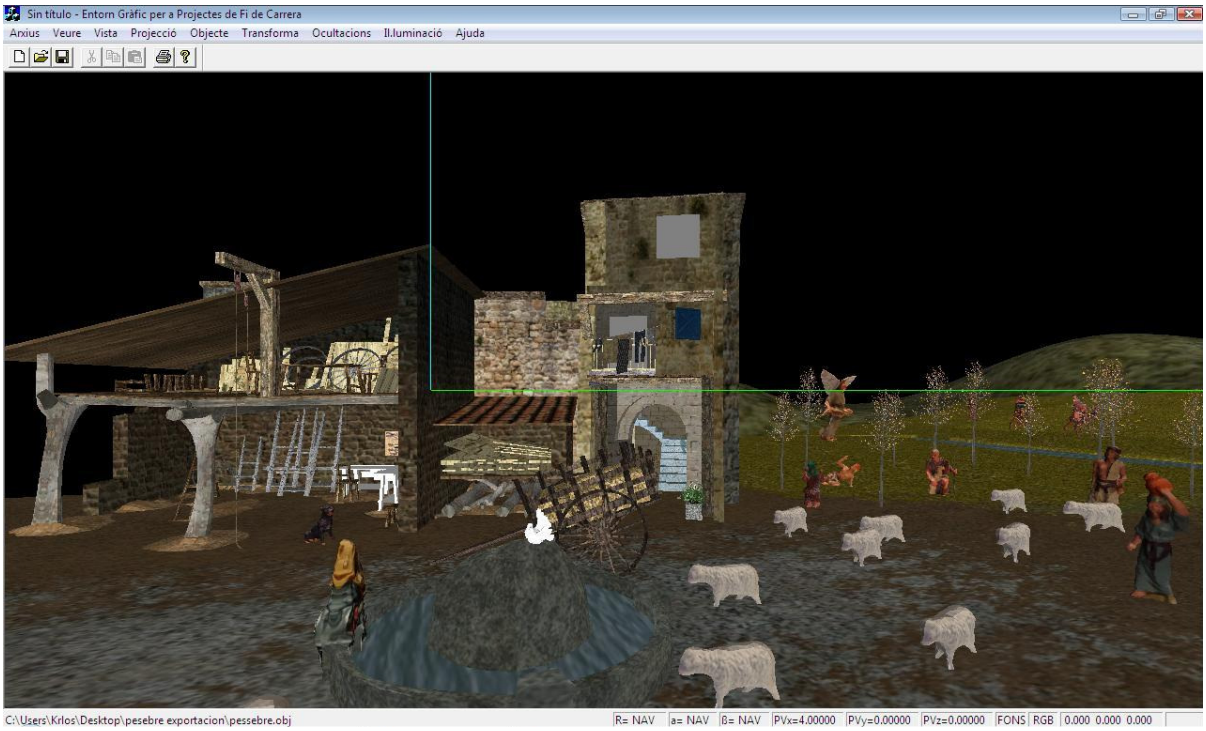
(e)

Figura 51. Test amb flors (a). Casa des de terra (b). Casa amb lianes (c). Figura Verge Maria (d). Figura Josep (e).

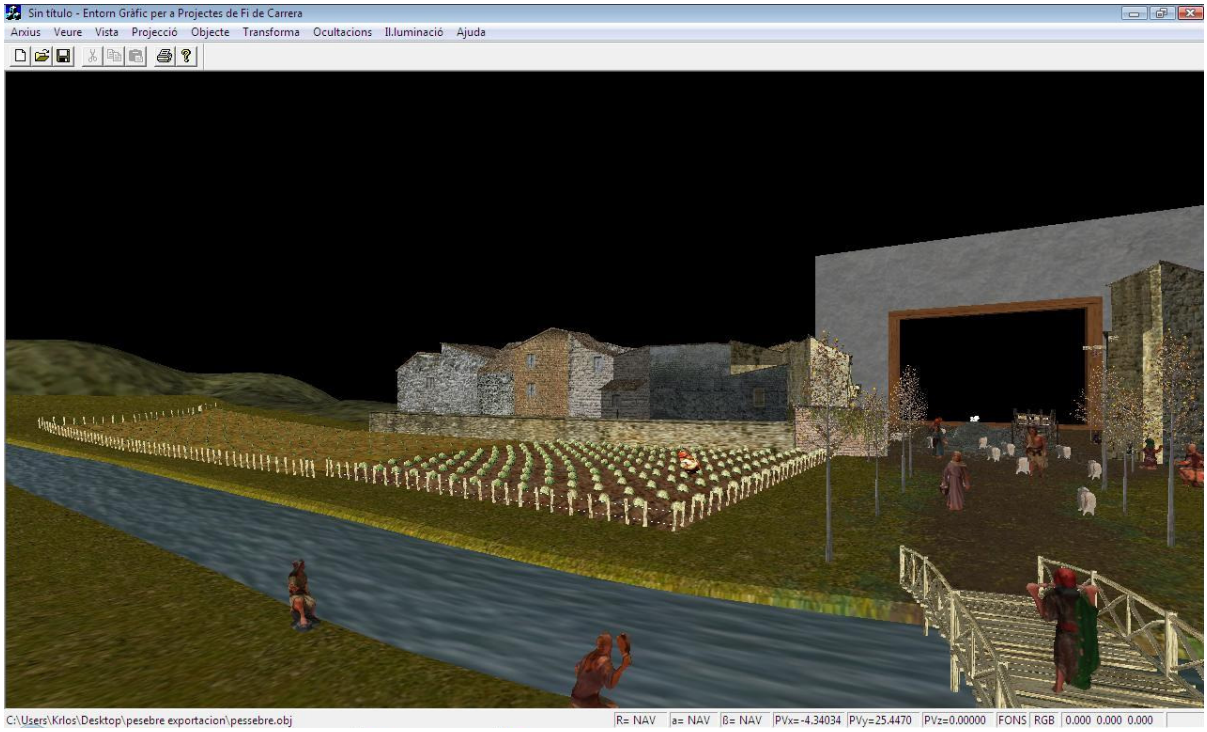
Pel que fa a resultats al exportar la informació cap a l'entorn de programació, també han estat bons, ja que les textures apareixen correctament i els objectes surten en les posicions correctes. La Figura 52 ens mostra l'exportació de tot el pesssebre.



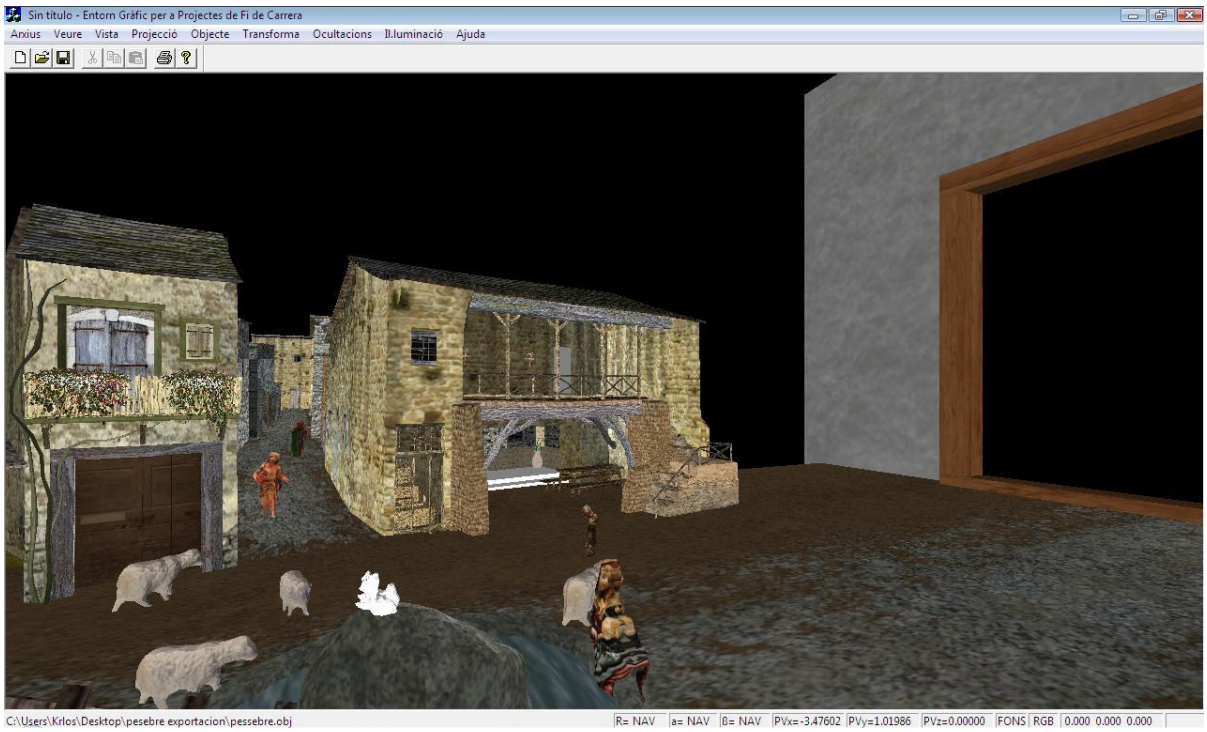
(a)


Sin título - Entorn Gràfic per a Projectes de Fi de Carrera
Arxius Veure Vista Projecció Objecte Transforma Ocultacions Il.luminació Ajuda
C:\Users\Krlos\Desktop\pesebre exportacion\pessebre.obj
R= NAV a= NAV ß= NAV PVx=4.00000 PVy=0.00000 PVz=0.00000 FONS RGB 0.000 0.000 0.000

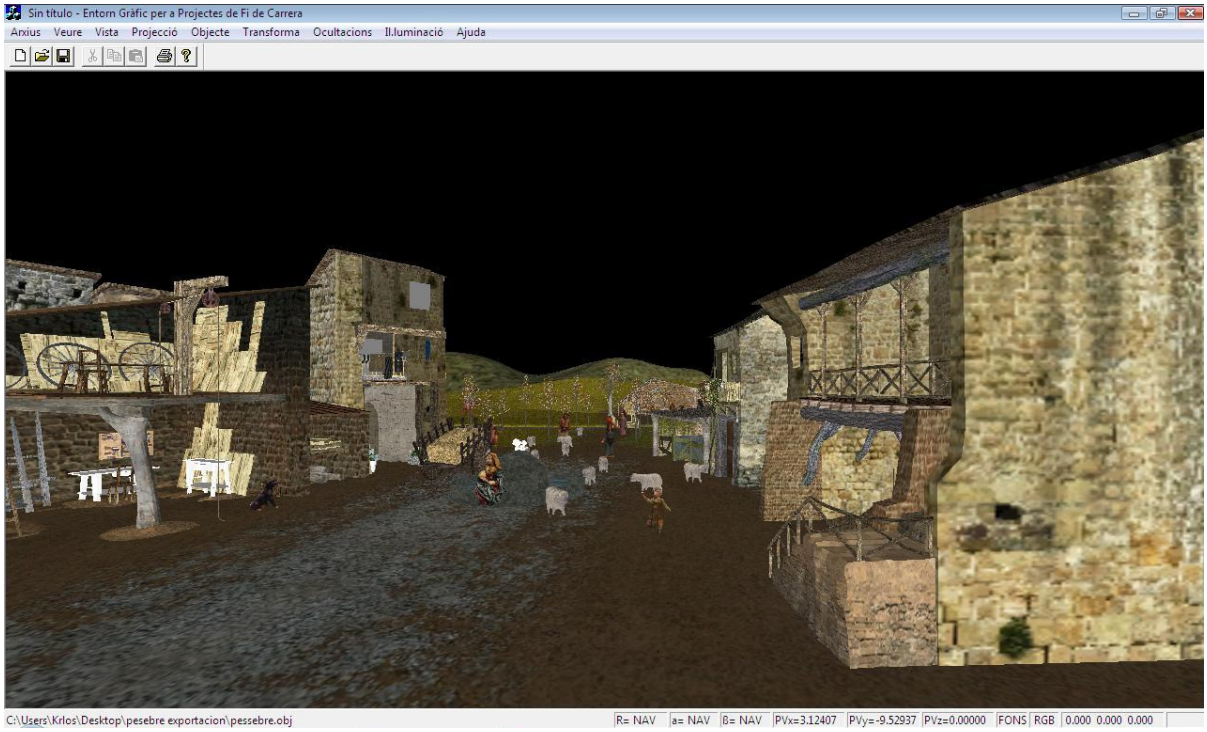
(b)


Sin título - Entorn Gràfic per a Projectes de Fi de Carrera
Arxius Veure Vista Projecció Objecte Transforma Ocultacions Il.luminació Ajuda
C:\Users\Krlos\Desktop\pesebre exportacion\pessebre.obj
R= NAV a= NAV ß= NAV PVx=-4.34034 PVy=25.4470 PVz=0.00000 FONS RGB 0.000 0.000 0.000

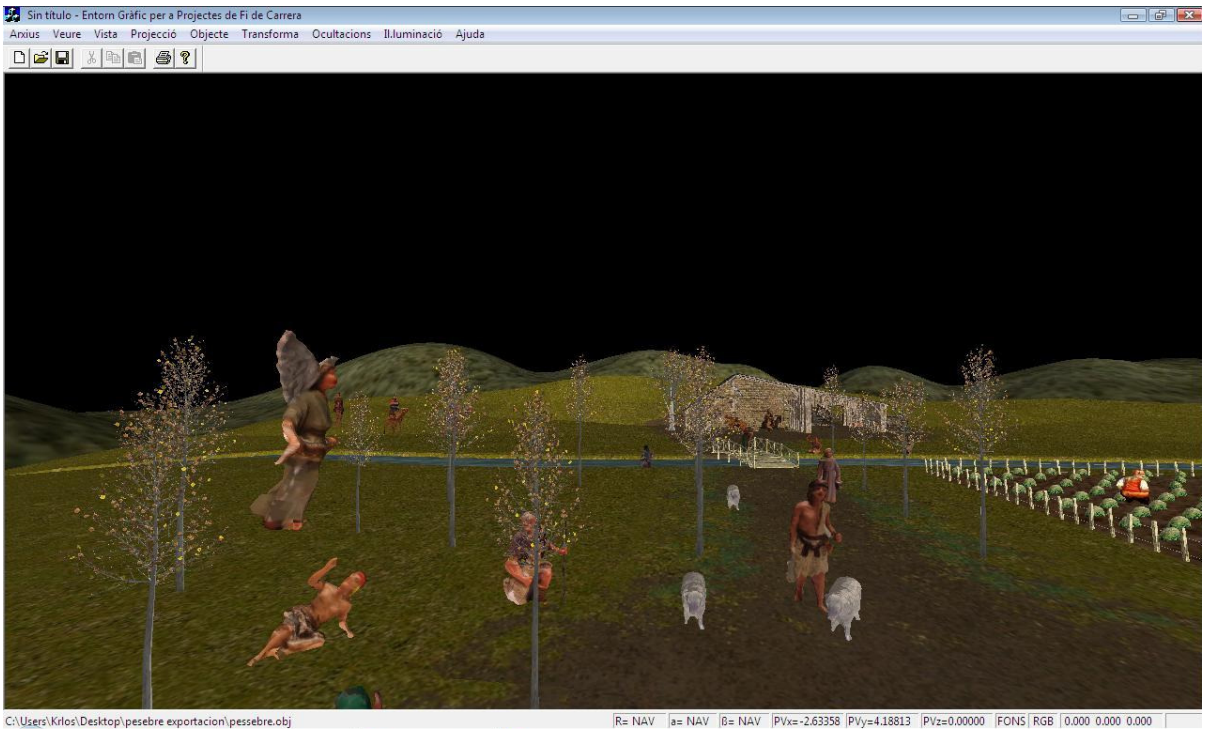
(c)


Sin título - Entorn Gràfic per a Projectes de Fi de Carrera
Arxius Veure Vista Projecció Objecte Transforma Ocultacions Il.luminació Ajuda
C:\Users\Krlos\Desktop\pesebre exportacion\pessebre.obj
R= NAV a= NAV ß= NAV PVx=-3.47602 PVy=1.01986 PVz=0.00000 FONS RGB 0.000 0.000 0.000

(d)


Sin título - Entorn Gràfic per a Projectes de Fi de Carrera
Arxius Veure Vista Projecció Objecte Transforma Ocultacions Il.luminació Ajuda
C:\Users\Krlos\Desktop\pesebre exportacion\pessebre.obj
R= NAV a= NAV ß= NAV PVx=3.12407 PVy=-9.52937 PVz=0.00000 FONS RGB 0.000 0.000 0.000

(e)



(f)

Figura 52. Resultat de visualitzarel pessebre sencer després de fer la importació.

4 Conclusions i millores

- S'ha estudiat un mecanisme d'exportació que permeti representar els objectes en 3 dimensions, dissenyats amb Blender, a un entorn de Visual C++ i OpenGL interactiu.

- S'ha dissenyat un diorama de pessebre virtual força realista, del qual s'ha generat un vídeo que es va mostrar en la XXXX Mostra pessebrista del Nadal 2008 de l'Associació de Pessebres de Sabadell.

- En el cas del format gràfic s'ha aconseguit exportar la geometria i les textures sense problemes, i s'han pensat solucions pel que seria l'exportació de la il·luminació, càmeres i moviments. La millora aquí consistiria en desenvolupar aquestes idees de manera que per tal d'ampliar les funcionalitats del format OBJ bàsic.

- Personalment, aquest projecte m'ha aportat moltes coses, des de coneixements nous, fins a la millora de les metodologies emprades en projectes d'aquesta mena.

- Durant la realització del diorama han sorgit molts problemes per la falta de coneixements del programari emprat i per falta d'experiència, però quan es treballa amb programes com el Blender, la massiva quantitat de funcionalitats i la no sempre intuïtiva interfície fan que aquestes coses passin.

- Pel que fa el format gràfic, no hi ha hagut gaires problemes, ja que una vegada realitzades les proves pel funcionament, només calia entendre la forma en que es tractava la informació. Al inici del projecte teníem bastants dubtes sobre si es podria aconseguir tot el proposat, però l'avantatge era que podíem provar a petita escala, tot allò que volíem fer més tard. Per exemple, per a saber si el format funcionava correctament no calia haver dissenyat una casa del diorama, simplement modelant u cub o qualsevol altre objecte senzill i després aplicant-hi una textura ja podíem saber si funcionaria o no.

- Com a conclusions generals, podem dir que hem trobat molt interessant poder realitzar un projecte d'aquesta mena, i estem molt agraïts, ja que un projecte final de carrera com aquest no tothom té la oportunitat de fer-lo. El resultat final val, de lluny, totes les hores dedicades i les preocupacions passades.

- Esperem que aquest projecte no acabí aquí i que altres alumnes puguin seguir millorant-lo, ja que en són moltes les coses que encara s'hi poden arribar a fer. De moment estem orgullosos d'haver introduït algunes de les noves tecnologies al món dels pessebres, i poder formar part d'una tradició tant important com és aquesta.

Referent al disseny, estem molt orgullosos dels resultats, però com a tot treball artístic, s'hi poden aplicar infinites millores, i infinites modificacions, de les que destacaríem:

- Millora 1: Dissenyar un paisatge més detallat amb menys espais amb muntanyes i camp en general. Modelar més cases o construccions en general i repartir-les pels voltants del poble.

- Millora 2: Posar més figures, vehicles i animals, que donessin encara més vida al poblat. Aquí també podríem incloure la inserció de més vegetació i arbres en general.

- Millora 3: Mirar la viabilitat de que l'escena pugui representar un paisatge nevat.

5 Bibliografia

[BCI] M. Galán. Blender. Curso de Iniciación. Infobook's, S.L. 2007

[CGt] CG textures. http://www.cgtextures.com/

[CSS] Counter Strike Source. http://store.steampowered.com/app/240/

[Dscn] David Scanner. http://www.david-laserscanner.com/

[Fmp] Luis Santander. Fundamentos del modelado poligonal. http://www.yetzero.cl/fundamentos-del-modelado-poligonal

[FWO] Format Wavefront OBJ. http://computacion-grafica-avanzada.blogspot.com/2009/02/especificaciones-del-formato-wavefront.html

[PBF] Programari Blender Foundation. http://es.wikipedia.org/wiki/Blender

[Pes] Wikipedia. http://ca.wikipedia.org/wiki/Pessebre

[Sga] Script generador d'arbre (gen3-0.5). http://www.geocities.com/bgen3/

[Vdb] Virtualdub. http://www.virtualdub.org/

Anex. Format gràfic OBJ

Estructura dels arxius

Els següents tipus de dades poden ser inclosos en un fitxer .obj. Els parèntesis indiquen el identificador de cada tipus de dada dins del fitxer.

Vèrtexs de dades:

- Vèrtexs geomètrics (v).
- Vèrtexs de textura (vt).
- Normals dels vèrtexs (vn).
- Paràmetre d'espai entre vèrtexs (vp). Atributs de les formes lliures.
- Formes racionals o no racionals dels tipus corbes o superfícies.
- Grau (deg).
- Matriu base (bmat).
- Mida de pas (step).

Elements:

- Punt (p).
- Línia (l).
- Cara (f).
- Corba (curv).
- Corba 2D (curv2).
- Superfície (surf).

Formes lliures corbo/superfície. Declaracions del cos:

- Valors dels paràmetres (parm).
- Retallada del bucle exterior (trim).
- Retallada del bucle interior (hole).
- Corba especial (scrv).
- Punt especial (sp).

- Declaració final (end).

Connectivitat entre formes lliures de superfícies:

- Connexió (con).

Agrupació:

- Nom de grup (g).
- Grup de suavitzat (s).
- Grups de fusió (mg).
- Nom de l'objecte (o).

Activar/mostrar atributs:

- Interpolació de bisellat (bevel).
- Interpolació de color (c_interp).
- Interpolació de dissolució (d_interp).
- Nivell de detall (lod).
- Nom del material (usemtl).
- Llibreria de material (mtllib).
- Ombra fosa (shadow_obj).
- Traçat de raigs (trace_obj).
- Tècnica d'aproximació de corba (ctech).
- Tècnica d'aproximació de superfície (stech).

<u>Vèrtexs de dades:</u>

Els vèrtexs de dades proporcionen coordenades per:

- Vèrtexs geomètrics.

- Vèrtexs de textura.
- Vèrtexs de normals.

Per objectes de forma lliure, els vèrtexs de dades també ofereixen:

- Espai de paràmetres entre vèrtexs.

El vèrtex de dades està representat per quatre llistes de vèrtexs; una per cada tipus de vèrtex de coordenades. Per altre banda el sistema de coordenades es fa servir per especificar les ubicacions de les coordenades.

El següent exemple és una part d'un fitxer .obj que conté els quatre tipus d'informació de vèrtexs.

```
v    -5.000000    5.000000    0.000000
v    -5.000000   -5.000000    0.000000
v     5.000000   -5.000000    0.000000
v     5.000000    5.000000    0.000000

vt   -5.000000    5.000000    0.000000
vt   -5.000000   -5.000000    0.000000
vt    5.000000   -5.000000    0.000000
vt    5.000000    5.000000    0.000000

vn    0.000000    0.000000    1.000000
vn    0.000000    0.000000    1.000000
vn    0.000000    0.000000    1.000000
vn    0.000000    0.000000    1.000000

vp    0.210000    3.590000
vp    0.000000    0.000000
vp    1.000000    0.000000
vp    0.500000    0.500000
```

Quan els vèrtexs són localitzats al Visualitzador Avançat, són enumerats de forma seqüencial, començant des de 1. Aquesta referència numèrica es fa server en la declaració d'elements.

Sintaxi:

Les següents declaracions de sintaxi estan llistats en ordre de complexitat.

- v x y z w :

  Declaracions de geometria poligonal i geometria de formes lliures. Especifica un vèrtex geomètric i les seves coordenades x y z. Les corbes racionals i les superfícies requereixen una quarta coordenada homogènia, anomenada pes.

  La x, y i z són les coordenades x, y i z del vèrtex. Aquests són números en coma flotant que defineixen la posició del vèrtexs en l'espai de tres dimensions.

  La w és el pes requerit per les corbes i superfícies racionals. No és requerit per les corbes i superfícies no racionals. Si no s'especifica cap valor per aquest atribut, per defecte serà de 1.0. Normalment es recomana que el valor d'aquest pes no sigui mai negatiu o zero, ja que pot portar a un punt no definit dins d'una corba o una superfície.

- vp u v w:

  Declaració de geometria de formes lliures. Especifica un punt en l'espai de paràmetres d'una corba o superfície.

  L'ús determina quantes coordenades són requerides. Els punts especials per corbes requereixen un punt de control d'una dimensió (u únicament) en l'espai de paràmetres de la corba. Els punts especial per les superfícies requereixen un punt de dues dimensions (u i v) en l'espai de paràmetres de la superfície. Els punts de control pels retalls de corbes no racionals requereixen les coordenades u i v. Els punts de control pels retalls de cornes racionals requereixen les coordenades u, v i w (pes).

  u és el punt en l'espai de paràmetres de la corba o la primera coordenada en l'espai de paràmetres de la superfície.

  v és la segona coordenada en l'espai de paràmetres de la superfície.

  w és el pes requerit pel retall de corbes racionals. Si s'especifica un valor, llavors per defecte serà 1.0.

- vn i j k:

  Declaració de geometria poligonal i geometria de forma lliure. Especifica el vector normal amb les components i, j i k.

Els vèrtexs de normals afecten la suavitat del ombrejat i el rederitzat de la geometria. Pels polígons, els vèrtexs de la normal es fan servir en lloc de la actual faceta de normals. Per superfícies, els vèrtexs normals són interpolats per tota la superfície i reemplacen la actual superfície real analítica. Quan els vèrtexs normals són presents substitueixen els grups de suavitzat.

i, j i k són les coordenades i, j i k pel vèrtex normal, que són nombres de punt flotant.

- vt u v w:

Declaració de vèrtex tant per geometria de polígons com per formes lliurees. Especifica un vèrtex de textura i les seves coordenades. Un textura de una dimensió requereix únicament la coordenada de textura u, una textura de dues dimensions requereix ambdues, u i v. Una textura de tres dimensions requereix les tres coordenades.

u és el valor per la direcció horitzontal de la textura.

v és un valor opcional. És el valor per la direcció vertical de la textura. Per defecte és 0.

w també és opcional. És el valor de la profunditat de la textura. Per defecte també és 0.

<u>Especificació de formes lliures corbes/superfícies:</u>

L'especificació de les formes lliure com corbes i superfícies dona lloc a tres passos:

- Especificació del tipus de corna o superfície (basis matrix, Bezier, B-spline, Cardinal o Taylor) fent servir atributs de les formes lliures corbes/superfície.
- Descripció de la corba o superfície amb declaració d'elements.
- Subministrament d'informació addicional, fent servir cos de declaracions de formes lliures corbes/superfície.

Resum

Aquest projecte és una part d'un projecte més ampli consistent en estudiar un format gràfic que permeti exportar una escena modelada en Blender i importar aquesta mateixa escena en un entorn interactiu basat en Visual C++ amb OpenGL. D'aquesta forma, disposem de la capacitat de modelat de Blender i de la interacció i visualització de la llibreria OpenGL. Aquest format ha de representar geometria i textures imprescindiblement, i si és possible, d'altres factors importants com il·luminació, visualització i moviment.

La part del projecte explicada en aquesta memòria consisteix en estudiar el format gràfic més adient per representar els diferents factors de realisme de l'escena (geometria, textura, etc.) havent triat el format OBJ per la seva capacitat de representació i fàcil edició. Per a provar el format, s'ha dissenyat un diorama de pessebre utilitzant les capacitats de modelatge de Blender. Pel que respecta les figures, aspecte important per a considerar l'escena com a pessebre, s'ha utilitzat un escàner 3D que ha obtingut representacions de malla 3D, a partir de figures reals de pessebre, que posteriorment han estat texturades.

S'ha generat un vídeo del diorama de pessebre que permet veure'n tots els detalls navegant amb el punt de vista per l'escena. Aquest vídeo s'ha exposat en la mostra de pessebres de la Associació Pessebrista de Sabadell el Nadal del 2008.