Universitat Politècnica de València

Escola Tècnica Superior d'Enginyeria del Disseny

Grau en Enginyeria en Electrònica Industrial i Automàtica

Treball Final de Grau

ÚS DE FERRAMENTES DE CODI OBERT PER LA REPRESENTACIÓ TRIDIMENSIONAL DEL MEDI AQUÀTIC

Documents:

1 Memòria

2 Pressupost

Autor:

Xavier Esteve Melià

Tutor:

Àngel F. Perles Ivars

Juliol 2014

Juliol 2014

Agraïments

Primerament volia agrair el suport del meu tutor Àngel. Sense ell, aquest projecte no s'haguera fet realitat. També agrair tot el personal de la universitat que m'ha ajudat al llarg de cinc anys d'esforç, treball i sacrifici que hui culminen. Per últim, agrair l'amor de la meua família. Endavant per guiar-me, darrere per alçar-me i al costat per recolzar-me. Gràcies.

Juliol 2014

Juliol 2014

Resum

Aquesta és l'era de la informació. Multituds de dades de diversos tipus apareixen a tots els àmbits de la vida de les persones. La sensació de connectivitat és més real que mai. La necessitat de transmetre aquesta informació de manera fiable, neta i precisa és evident. Així sorgeix aquest projecte que pretén, dins d'un entorn com el aquàtic, utilitzar ferramentes de software open source per representar en tres dimensions escenaris per al seua observació i estudi. Un xicotet pas d'iniciació al món de la simulació virtual.

Paraules clau: informació, connectivitat, software, open source, simulació virtual.

Juliol 2014

Juliol 2014

Resumen

Esta es la era de la información. Multitud de datos de diverso tipo aparecen en todos los ámbitos de la vida de las personas. La sensación de conectividad es más real que nunca. La necesidad de transmitir esta información de manera fiable, limpia y precisa es evidente. Así surge este proyecto que pretende, dentro de un entorno acuático, utilizar herramientas de software open source para representar en tres dimensiones escenarios para su observación y estudio. Un pequeño paso de iniciación en el mundo de la simulación virtual.

Palabras clave: información, conectividad, software, open source, simulación virtual.

Juliol 2014

Juliol 2014

Abstract

This is the information age. Crowd of data of various types appear in all areas of people's life. The feeling of connectivity is more real than ever. The need to transmit this information reliably, cleanly and precisely is evident. So this project aims, in an aquatic environment, using open source software tools to represent three-dimensional scenarios for their observation and stud. One small step into the world of virtual simulation.

Keywords: information, connectivity, software, open source, virtual simulation.

Juliol 2014

Juliol 2014

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Memòria del TFG

Ús de ferramentes de codi obert per la representació tridimensional del medi aquàtic

Autor: Xavier Esteve Melià

Tutor: Àngel F. Perles Ivars

Juliol 2014

Tabla de contingut

1 Objecte del projecte 2
2 Justificació del projecte 2
2.1 Antecedents 2
3 Alternatives y solució adoptada 3
3.1 Solucions al motor gràfic 3
3.2 Solucions a la modelització 4
4 Descripció detallada i justificació de la solució adoptada 5
4.1 Motor gràfic 5
4.2 Modelitzador 3D 7
4.3 Objectes a modelitzar 8
4.3.1 Glider 8
4.3.2 Cistella de piscifactoria 9
5 Execució 10
5.1 Sistema Operatiu (SO) 10
5.2 Instal·lació i muntatge del motor gràfic 10
5.3 Instal·lació d'un model aquàtic per OSG 15
5.4 Modelització tridimensional 20
5.4.1 Glider 20
5.4.2 Cistella de piscifactoria 38
5.5 osgOcean funcionant 51
6 Conclusions 55
7 Bibliografia 57

1 Objecte del projecte

La finalitat d'aquest projecte es realitzar una recreació tridimensional d'un determinat ambient, en aquest cas el medi aquàtic, mitjançant tècniques informàtiques de codi obert, també conegudes com *open source*, amb l'objectiu de presentar informació útil d'una forma visual, intuïtiva i senzilla.

2 Justificació del projecte

2.1 Antecedents

Avui, no existeix cap dubte, és l'era de la informació. Les hores dedicades a Internet augmenten sense fre, la connectivitat constant s'ha fet indispensable per una gran part de la població i la dependència de la mateixa és cada volta més acusada. Xarxes socials, *smartphones*, el flux de dades entre milions de persones a lapsus de temps cada volta més reduïts és innegable. Però no a soles el gran públic ha vist com la seua accessibilitat a la informació s'ha incrementat: com enginyers en electrònica i informàtica, les corresponents àrees de negoci es caracteritzen pel seu dinamisme.

Així doncs, tan important és el volum de la informació a estudiar com la seua representació. Les representacions gràfiques han sigut sempre una ferramenta indispensable per ajudar a comprendre de manera ràpida dades de tot tipus. Encara i així, l'avanç del temps ha fet aquestes formes de comunicació a voltes massa complexes. És així com poden sorgir noves tècniques de representació: motors gràfics, programes de modelatge tridimensionals, entorns de desenvolupament... Totes elles s'empren per facilitar la comprensió, impacte o repercussió d'una informació que d'altra manera podria passar desapercebuda.

De l'actual importància d'aquestes disciplines naix el següent projecte, que pretén una xicoteta aproximació a un món immens, com és el de la representació tridimensional, de tal forma que resulte una escenificació que aglutine introduccions als motors gràfics, al modelatge en tres dimensions i a la possible interacció programable que puga haver-hi per una relació més propera i interactuable.

3 Alternatives y solució adoptada

3.1 Solucions al motor gràfic

El motor gràfic es el conjunt de ferramentes de programació, es a dir, software, encarregat de recrear i donar dinamisme a escenes visuals. Si hi existeix una indústria lligada plenament en el desenvolupament de motors gràfics es la dels videojocs, encara que tenen uns altres camps d'actuació, com l'ensenyança i projectes científics com el que ací es tracta.

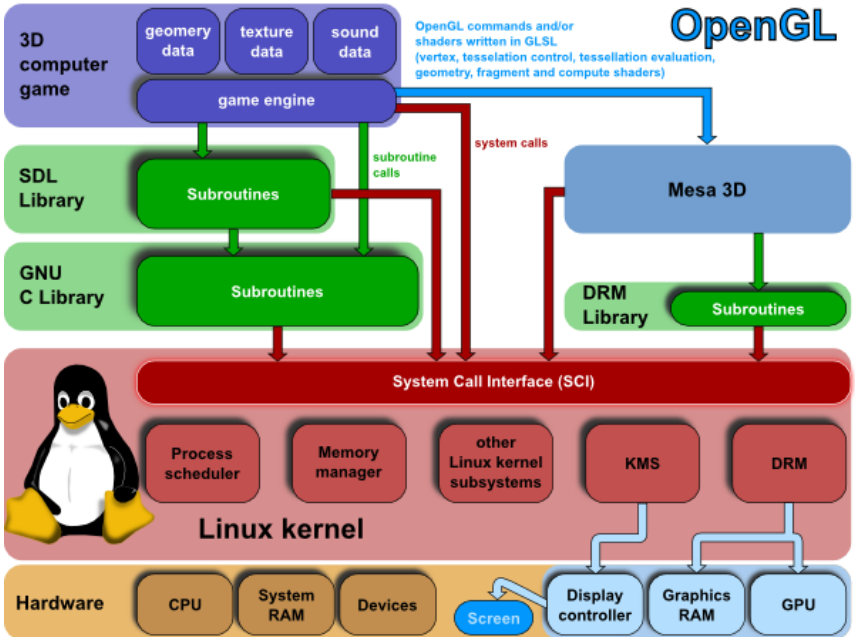

Fig. 3.1.1

Tipus de components d'un motor de videojocs qualsevol a Linux (Wikipèdia)

Es poden trobar múltiples motors gràfics en l'actualitat, i de molts diversos tipus. Per exemple, s'estima que un grup de treball moderadament gran orientat al disseny de videojocs per plataformes de nova generació, com la PlayStation 4 o Xbox ONE, pot crear un nou motor exclusiu en dos mesos. També hi han, però, motors reputats i de gran pes, com Unreal Engine o Havok, que altres estudis poden emprar i modificar per adaptar-los a les seues necessitats.

A aquest món, com qualsevol altre relacionat amb el software, les llicències tenen una gran importància. Per utilitzar el Unreal Engine anteriorment nomenat s'haurà de pagar, però a la versió 4 de pròxima aparició a més a més els consumidors tindran accés complet al codi font del motor. No obstant això hi ha alternatives lliures, les quals han format part del procés de selecció d'aquest projecte. Un d'ells és l'id Tech 4, dissenyat per John Carmack i popular per fer realitat jocs com Doom 3, Quake 4 o Wolfenstein. OGRE 3D [2] és un motor de renderitzat lliure en una gran comunitat donant-li suport, molt emprat també per fer videojocs menys comercials. Altres, com OpenSceneGraph (OSG) [3] no han tingut gran repercussió en aquesta indústria, però és molt utilitzat en altres àmbits, que compta en solucions singulars com la del esquema oferit (fig. 4.1.2).

3.2 Solucions a la modelització

Una volta tractats els motors gràfics, cal discernir el programa de modelització tridimensional que s'emprarà. Si el motor gràfic s'encarrega de afegir moviment i dinamisme a una determinada escena, es obvi que ha d'existir un escenari. Aquest conformarà una sèrie d'elements tridimensionals, més o menys complexos, els quals han d'haver sigut dissenyats anticipadament. Així doncs, els programes de modelització son les ferramentes que s'utilitzen per realitzar aquests dissenys.

Com succeeix amb els motors gràfics, existeix gran varietat de programes de modelització disponibles per als usuaris. De nou, una gran divisió que es pot fer d'aquests és distingir entre software lliure i propietari, a més del gratuït. Tots ells són molt emprats al cada volta més rellevant cinema d'animació, a més dels videojocs, i per descomptat diverses aplicacions científiques. Els camps d'actuació poden ser virtualment il·limitats: el disseny d'un nou producte, la recreació virtual d'un cos humà, modelitzacions per i simular esdeveniments físics, etc. Al món de l'enginyeria és freqüent l'aparició de problemes de dificultosa resolució teòrica, i la solució més factible sol ser la simulació mitjançant software específic. La modelització en 3D pot ser no tan sols útil, sinó essencial a l'hora de realitzar simulacions cada volta més precises i properes al comportament real que es pretén endevinar i planificar.

Fig. 3.2.1

Exemple de modelització amb Blender (blender.org)

Al grup de software de modelització tridimensional comercial es troben , per exemple, AutoDesk 3DS MAX [4], emprat per la saga Halo, AutoDesk Maya, ZBrush, i molts més. Després hi son els gratuïts, aquests que, encara que no s'haja de pagar llicència per al seu ús (normalment restringit a l'àmbit personal) no obrin el seu codi a la comunitat, és a dir, son freeware però no *open source*. Un exemple és el conegut SketchUp [5], adquirit per la companyia Google en 2006 i després tornat a vendre. Des del 2013 hi ha dos versions: el SketcUp Make, gratuït, i el Pro, de pagament. Dins dels de tipus *open source* son, per exemple, Blender, K-3D i Art of Illusion. Tenint en compte la motivació d'aquest treball, es restringeix la recerca als programes de codi lliure.

4 Descripció detallada i justificació de la solució adoptada

4.1 Motor gràfic

El motor gràfic escollit és OpenSceneGraph (OSG). La característica principal per la que ha sigut escollit per a la realització d'aquest projecte és que es tracta d'un software de codi obert[1]. El codi ha sigut escrit en llenguatge C++ i amb l'ús de OpenGL.

Fig. 4.1.1
Logo oficial OpenSceneGraph

OpenGL és una API[2] que descriu la realització mitjançant funcions contingudes al seu codi de figures geomètriques bàsiques. A partir d'ella, altres aplicacions són capaços d'usar-la per crear representacions bidimensionals i tridimensionals més avançades. Va nàixer l'any 1992 de la mà de Silicon Graphics Inc. El seu competidor a plataformes Windows és Direct 3D, part de DirectX. OpenGL és compatible en plataformes com Windows, Mac OS, GNU/Linux i més.

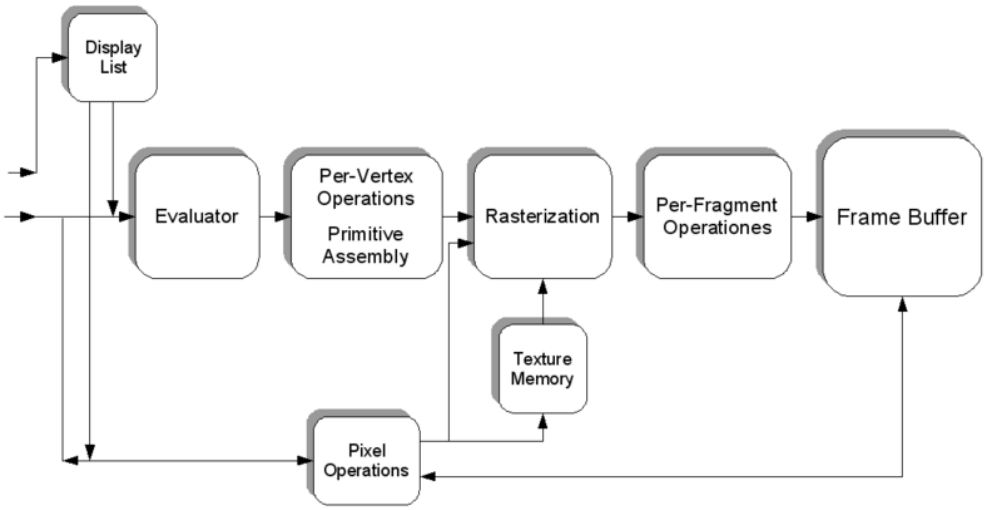

Fig. 4.1.2
Passos efectuats per representacions gràfiques

[1] ***codi obert (open source):*** *Tot aquell software que no sols és gratuït, sinó que a més fica a la disposició de l'usuari el seu codi Font perquè estudiat, manipulat, etc.*

[2] ***API (Application Programming Interface):*** *Conjunt de procediments que són utilitzats per la comunicació del software i aprofitar funcionalitats de cadascú d'ells per realizar determinades tasques.*

El començament d'OSG va ser l'any 1998 gràcies a Don Burns. Posteriorment, l'any 1999 es va unir a l'equip el que és considerat també màxim desenvolupador del projecte, Robert Osfield. Al llarg dels anys la seua rellevància i suport dins de la comunitat de programador ha anat augmentant.

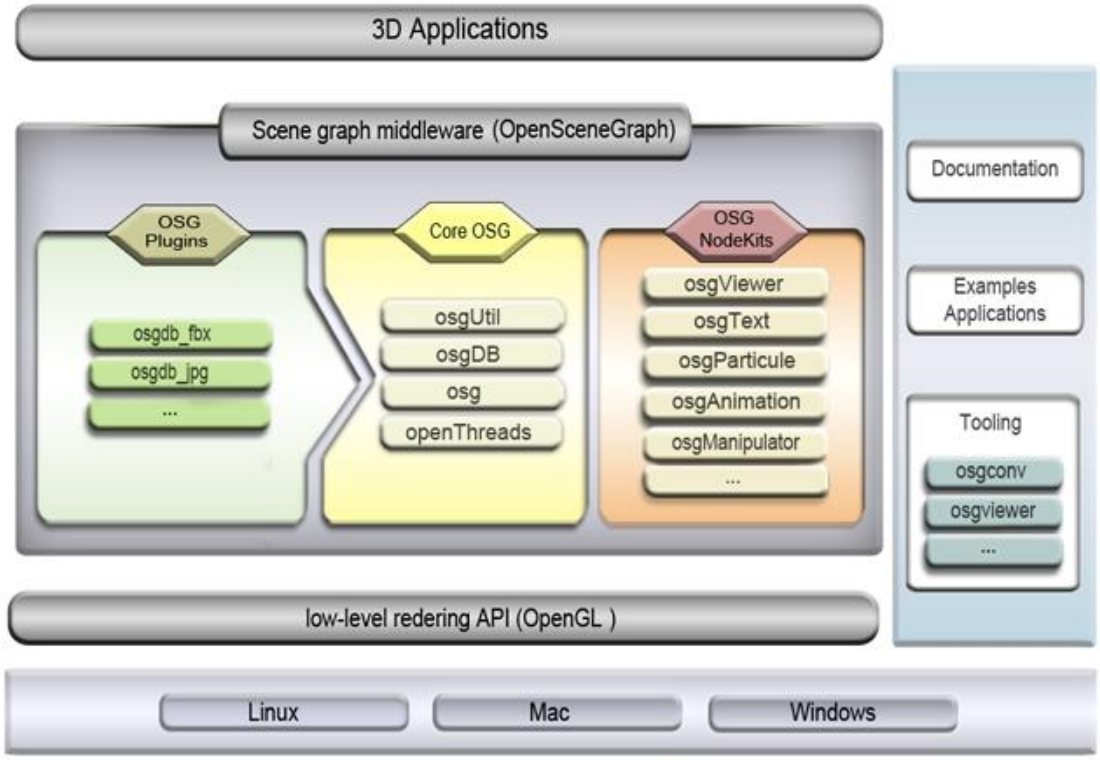

Fig. 4.1.3

Resum gràfic de l'arquitectura d'OSG (Wikipèdia)

La primera versió estable llançada ha sigut la 1.0 ha 2005. La última, la 3.2.1, al 4 de juliol de 2014. Ha sigut a partir de la 3.0.0 quan s'ha donat suport a plataformes mòbils (iOS i Android). OSG ha sigut utilitzat per videojocs, realitat virtual, softwares de simulació i aplicacions científiques que utilitzen recursos visuals i de modelització tridimensional. Un esquema simple de la seua arquitectura es troba dalt d'aquestes línies (fig. 4.1.3).

Serà OSG el motor gràfic elegit per la seua versatilitat, integració contrastada en diverses aplicacions i entorns de desenvolupament, i per comptar en una sòlida base d'usuaris que ajuden a un constant desenvolupament.

4.2 Modelitzador 3D

La ferramenta de modelització escollida es Blender. Encara que a aquest treball s'emprarà Blender com modelitzador 3D, també pot ser usat per moltes altres tasques, com ara il·luminació, edició de vídeos, animació, i fins i tot desenvolupament de videojocs. És un software multiplataforma, disponible a Windows, Mac OS X, GNU/Linux, Solaris, FreeBSD i IRIX.

Blender té els seus orígens a la companyia d'animació holandesa NeoGeo, en 1988, co-fundada per Ton Roosendaal. Una dècada més tard, al 1998, Ton va crear una companyia lligada a NeoGeo de nom Not a Number. Pretenia desenvolupar un software de modelatje i animació 3D a l'abast de tots els usuaris. A l'any 2000 es llança la versió de Blender 2.0, amb importants novetats, però dificultats econòmiques van obligar a parar el desenvolupament de Blender. Encara i així, al 2002, Ton Roosendaal, amb l'ajuda de la comunitat va fundar Blender Foundation.

Aquesta organització va alliberar el codi de Blender, incloent-lo a la categoria de software *open source*. El suport continu de la comunitat, encapçalats per Ton Roosendaal, fa possible que Blender siga actualitzat i millorat de continu. La última versió disponible és la 2.71 el 26 de juny de 2014.

Fig. 4.2.1
Portada de l'última versió de Blender (2.71)

Finalment s'opta per Blender, un software de versatilitat contrastada i defensat i millorat de forma contínua per una fidel i enorme comunitat. No obstant això, Blender és també conegut per una interfície[3] d'usuari confusa i difícil d'assimilar pels nous usuaris.

[3] ***interfície:*** *Mètode de representació gràfica que el software crea a la pantalla perquè l'usuari puga treballar amb ell de manera còmoda i eficaç.*

4.3 Objectes a modelitzar

4.3.1 Glider

Com qualsevol escena gràfica animada, com un projecte de simulació en aquest cas, a més del motor gràfic que faça evolucionar tot el que continga la pantalla, també hi existeix un determinat objecte gràfic tridimensional que pot ser renderitzat prèviament. Ja siga un cotxe en un simulador automobilístic o un avió en un simulador de vol, sol haver-hi una entitat dins l'escenari perquè siga controlada per l'usuari o que simplement aparega com referència dins del projecte.

Al present treball, ja que es planteja la recreació d'un escenari aquàtic, és obvia la presència d'un determinat aparell que tinga a veure en aquest medi. Degut a que l'objecte del projecte és informàtic i merament informatiu i visual no hi han cap raons perquè el aparell a modelitzar siga l'un o l'altre. Es per açò que ha sigut escollit per la seua modelització un glider i una cistella de piscifactoria

Fig. 4.3.1.1
Fotografia d'un glider submarí (Wikipèdia)

Un glider (fig. 4.3.1.1) és un dispositiu que pertany a la categoria dels AUV's[4] (Autonomous Underwater Vehicles) [7]. Són emprats per recollir dades d'interès en entorns aquàtics. La seua peculiaritat es que, en un mecanisme mecànic particular, són capaços de transformar moviments verticals en horitzontals mitjançant unes ales instal·lades al seu cos. Es caracteritzen per no necessitar hèlices ni motors addicionals, pel qual la seua autonomia és molt superior a la mitja, encara que també fa que siguen més lents. A més, son una de les solucions més assequibles que es poden trobar per aquestes faenes.

[4] ***AUV's (Autonomous Underwater Vehicles) [8]:*** *Aquells dispositius que són capaços de realitzar determinades faenes baix de l'aigua sense supervisió de cap operari*

4.3.2 Cistella de piscifactoria

Altre objecte que es pot modelitzar per ser inclòs a una escena aquàtica es una piscifactoria. En aquest cas s'ha optat per una cistella de piscifactoria comú (fig. 4.3.2.1).

Fig. 4.3.2.1
Piscifactoria (Wordpress)

Les piscifactories [9] son recintes que permeten la cria de peixos que després seran consumits. Al tractar-se d'un espai tancat, quasi totes les condicions ambientals dels peixos són supervisades, com el menjar i la qualitat de l'aigua. Presenta avantatges com el control exhaustiu al que es sotmet la cria, així que arriben al mercat en unes característiques prèviament estudiades, a més de que suposa una pèrdua d'espècimens menor.

Com que es tracta d'un instrument prou emprat en l'actualitat i comprén àrees d'estudi de moltes disciplines científiques (com enginyeria, biologia, etc.), a més d'estar sotmeses a vigilància i recerca de dades constant, és lògic pensar que un model seu puga resultar útil per proveir simulacions que aprofundisquen en ella.

5 Execució

5.1 Sistema Operatiu (SO)

Seguint la filosofia d'emprar software open source, el treball s'ha desenvolupat en un ordinador amb un SO en base GNU/Linux, més concretament la distribució de KDE KUbuntu 12.04 (fig. 5.1.1).

Fig. 5.1.1
Logo distribució GNU/Linux KUbuntu

Encara que hauria de ser possible realitzar el projecte en una plataforma Microsoft Windows, ja que software com Blender és compatible en les dos, la naturalesa oberta dels motors gràfics que es pretenen muntar van a fer que treballar en Linux siga més senzill. Es per açò perquè s'ha escollit aquest SO.

5.2 Instal·lació i muntatge del motor gràfic

Una volta escollit el motor gràfic i justificat la seua elecció, es presenten a continuació les instruccions per un correcte muntatge global de tot el software necessari. Cal recordar que treballem en un SO basat en Linux, i que la guia de més endavant canviarà en funció del SO de l'elecció de l'usuari.

L'objectiu final es aconseguir fer funcionar el motor *osgOcean*. Abans s'hauran de realitzar uns passos previs. El primer es instal·lar *OpenSceneGraph* (OSG) per sí mateix (versió 3.0.1). A Linux el procediment hauria de ser tan simple com obrir el repositori[5] (p. ex. *Synaptics*), buscar *OpenSceneGraph* i instal·lar. Però, encara que es pugen veure resultats factibles, aquesta versió és insuficient. Es necessita, doncs, la versió per desenvolupadors, diferenciada comunament per una terminació *–dev* a l'arxiu.

[5] ***Repositori [10]:*** *Software amb accés a una llista de paquets que permet a l'usuari gestionar-los lliurement, encarregant-se ell mateix de les possibles dependències que pogués haver-hi.*

Primerament es descarrega la versió 3.0.1 d'OSG [13]. S'obri el terminal, que a aquesta distribució rep el nom de *Konsole* (SHC[6] ctrl+alt+T). L'ús de combinacions de tecles pot arribar a ser important per tal d'estalviar temps i tindre una major soltesa a l'hora de treballar. Si aquest SHC no estiguera configurat es fa clic dret al botó KDE (llançador d'aplicacions) i clic en Editar Aplicacions. Es busca l'aplicació *Konsole* en System, clic en Advanced i definir el *shortcut* que més plaga a l'usuari (fig. 5.2.1).

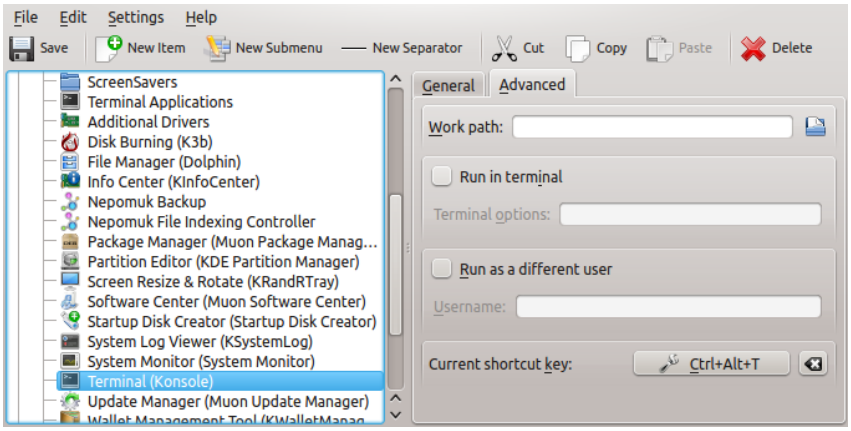

Fig. 5.2.1

Definir shortcuts en escritoris KDE

Així doncs, s'obri el terminal *Konsole* i s'escriu:

```
sudo apt-get build-dep openscenegraph
```

D'aquesta manera es crearà l'arbre de dependències de l'OSG i s'instal·laran els paquets corresponents. Si no fora possible completar la instrucció s'escriu abans:

```
sudo apt-get update
```

A l'hora de compilar paquets mitjanament complexos és una pràctica habitual crear un directori superior al que s'acaba de descarregar (normalment nomenat *build*) en que es produiran les execucions auxiliars necessàries.

```
cd <directori anterior a OpenSceneGraph-3.0.1>
```

```
mkdir build
```

[6] ***SHC:*** *Abreviatura de shortcut. Literalment drecera (atajo). Fa referència al conjunt de tecles que cal pressionar, segons programa o SO emprat, per donar una instrucció ràpidament.*

Seguidament s'empren les instruccions que es troben al *README.txt* de l'OSG, que simplificades son:

```
cd build
```

```
cmake ../OpenSceneGraph-3.0.1 –DCMAKE_BUILD_TYPE=Debug
```

Si *cmake* no estigués instal·lat, fer:

```
sudo apt-get install cmake
```

Així es crearan a la carpeta *build* anteriorment buida els arxius de CMake necessaris per fer funcionar aquesta versió d'OSG. Una volta finalitzat el procés, es procedeix en la instrucció:

```
make
```

Aquesta fase pot ser prou llarga, depenent en gran manera del hardware en que treballem. Ací es troba una de les raons perquè no és recomanable l'ús d'un virtualitzador; aprofitar tota la potència que puga oferir el processador és vital perquè els temps d'espera siguen minimitzats.

Pot aparèixer un error relacionat amb FFmpegDecoder. Per resoldre-ho cal obrir el fitxer localitzat al directori *OpenSceneGraph-3.0.1/src/osgPlugins/ffmpeg/FFmpegDecoder.cpp* i afegir la següent capçalera a l'inici:

```
#include <libavutil/mathematics.h>
```

Per comprovar que OSG funciona correctament es poden activar els exemples disponibles. Des del terminal s'apunta al directori arrel de OSG i amb la instrucció *ls* apareixen tots els arxius que inclou. Es localitza un anomenat *runexamples.bat*[7]. El codi que conté només farà executar de seguit tots els exemples que són inclosos a la versió d'OSG que s'haja descarregat. Cal garantir que aquest arxiu siga executable. Hi son dos opcions.

[7] ***.bat:*** *Extensió dels arxius batch. De tipus MS-DOS, contenen codi seqüencial que permet executar instruccions en ordre*

Des de un gestor de fitxers qualsevol (p. ex. *Dolphin*), es fa clic dret a l'arxiu en concret (en aquest cas *runexamples.bat*) i, a la pestanya *Permisos*, marcar la casella *És un executable* (fig. 5.2.2).

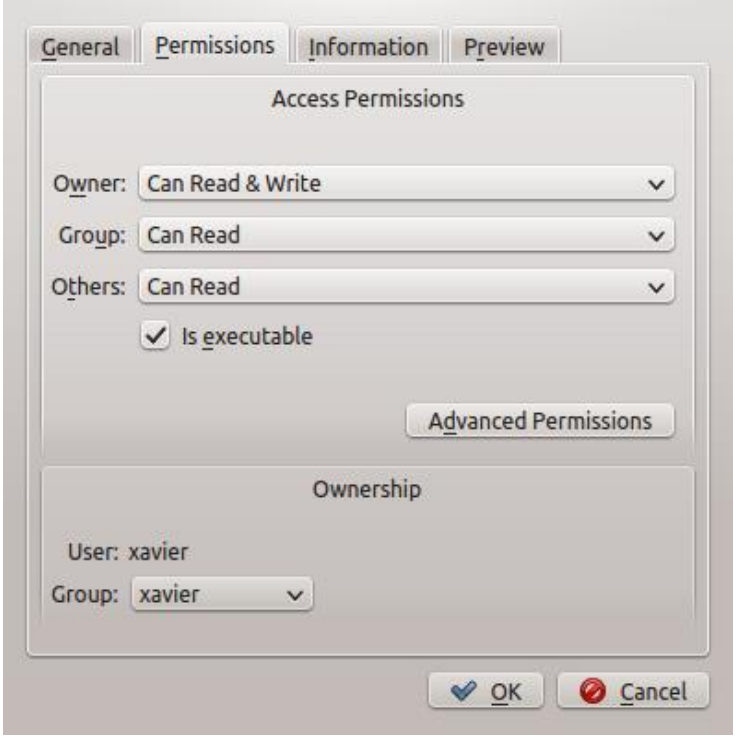

Fig. 5.2.2

Fer executable un arxiu

L'altra manera es des del terminal:

```
cd <directori on es troba el fitxer a fer executable>

chmod +x <nom de l'arxiu que rebrà permisos d'execució>
```

Una volta configurat l'arxiu *runexamples.bat* es fa:

```
cd <directori on es troba runexamples.bat>

sh ./runexamples.bat
```

Començaran a aparèixer pantalles amb el exemples. Fer clic en ESC per passar a una altra (fig. 5.2.3). En fer clic esquerre e la pantalla i desplaçar el ratolí es canvia la perspectiva de la càmera i en la roda es pot fer zoom a les zones d'interès.

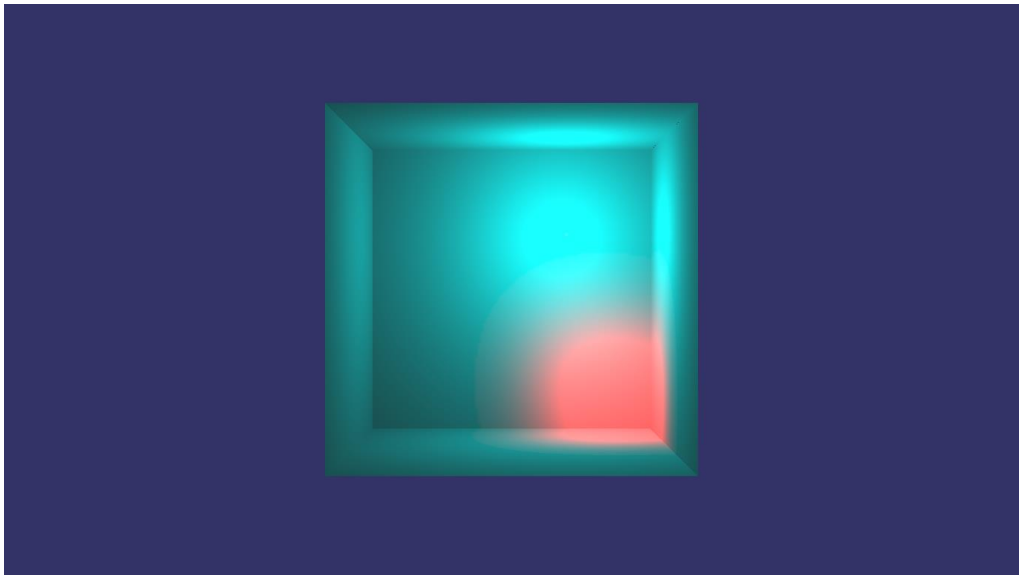

Fig. 5.2.3

Exemple animat de OpenSceneGraph

En alguns casos, pressionant *S* apareix informació addicional, com el framerate[8], ús de memòria de l'ordinador, etc. (fig. 5.2.4).

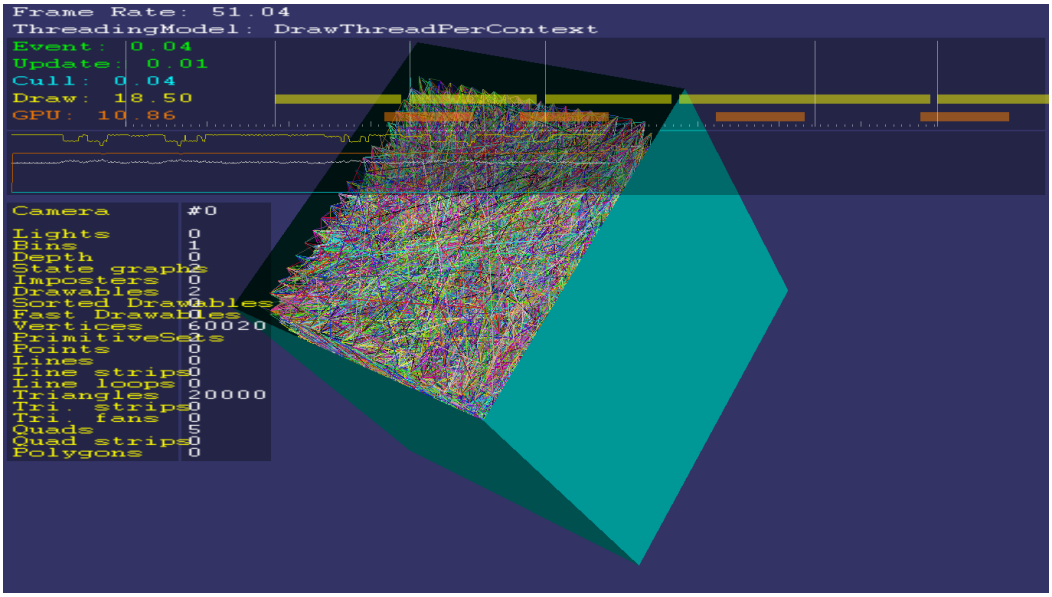

Fig 5.2.4

Exemple d'OSG amb informació gràfica

[8] ***framerate:*** *Freqüència en que apareixen imatges per segon en una animació. A major quantitat d'informació (píxels) menor serà el framerate.*

5.3 Instal·lació d'un model aquàtic per OSG

Una volta finalitzat el procés es passa a la instal·lació de *osgOcean*. Es descarreguen els arxius comprimits (*.rar*) de la pàgina corresponent [14] i es copien les carpetes *boat*, *island* i textures descomprimides de l'arxiu *osgOcean-Resources-1.01.rar* a la carpeta *resources* dins de *osgOcean*. Es recomana seguir el mateix procediment d'abans, crear una carpeta *osgOcean* global i dins una altra que contindrà tots els directoris que s'hagen descomprimit de l'arxiu abans esmentat.

Abans de res, cal analitzar les dependències de *osgOcean*. Per descomptat, la de *OpenSceneGraph* ja està completada. També hi necessitarà la vinculació amb una llibreria de transformades de Fourier. Es descarrega l'arxiu comprimit [15] i es descomprimeix. Posteriorment es segueixen les instruccions de l'arxiu de text INSTALL resumides en les següents línies:

```
cd <directori principal on es troba la llibreria fftss>
```

```
./configure --with-recommended
```

(Afegir el *flag* –with-recommended per garantir la correcta compilació de la llibreria).

```
make
```

```
sudo make install
```

Una volta instal·lada la llibreria, al terminal s'escriu:

```
cd <directori anterior a osgOcean>
```

```
mkdir
```

```
cd build
```

I a continuació:

```
cmake ..
```

Es generaran els arxius i enllaços necessaris perquè funcione *osgOcean*, però hi hauran alguns que requeriran ser vinculats pel mateix usuari. Així doncs, es treballarà en la ferramenta ccmake[9]. Es tecleja:

```
ccmake ..
```

És possible que s'haja d'instal·lar *ccmake*. Es tecleja:

```
sudo apt-get install cmake-curses-gui
```

[9] ***ccmake:*** *Software visual bàsic emprat per configurar-hi manualment els aspectes relacionats en el cmake i necessaris per la correcta compilació de determinats programes.*

S'obrirà una finestra (fig. 5.3.1) en la que apareixeran, en la columna de l'esquerra les llibreries a les que necessita vincular-se *osgOcean*, i a la de la dreta, els directoris on són localitzades a l'HDD (Hard Disk Drive). Potser prou de les llibreries no siguen trobades pel propi *cmake*, així que, com resa en les instruccions de baix, s'utilitza el cursor per localitzar-lo en l'opció a canviar i es pressiona [enter].

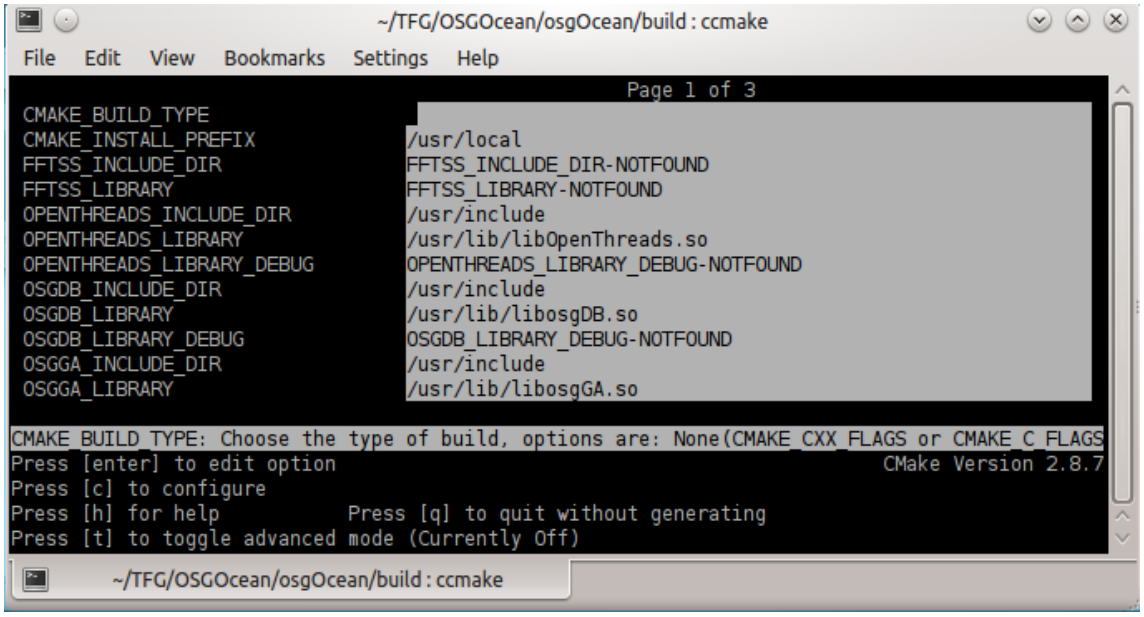

Fig. 5.3.1

UI del CCMake generat des del terminal

Aquest es només un exemple de les errades que poden sortir a l'hora de compilar *osgOcean*. El següent pas serà introduir manualment els directoris que *cmake* no haja trobat per sí mateix. Una opció que potser estalvie temps és buscar, mitjançant un gestor de paquets[10], els paquets que coincideixen en els que es necessiten per compilar el software, i consultar els arxius que ha instal·lat i la seua ubicació exacta. A la figura 5.3.2 es pot veure com localitzar el paquet de *FFTW*[11] per tal de afegir el directori correcte.

Es recomana continuar en aquest mètode fins que siga possible per omplir correctament les dependències de l'*osgOcean*.

[10] ***gestor de paquets:*** *Software emprat en SO's de base Linux que tracta de forma visual els repositoris (veure nota 2). Permet, entre altres coses, veure els paquets disponibles i instal·lats. Alguns exemples són els de Synaptic, Muon, etc.*

[11] ***FFTW (Fast Fourier Transform in the West):*** *Paquet emprat per calcular transformades de Fourier preinstal·lat en distribucions Linux.*

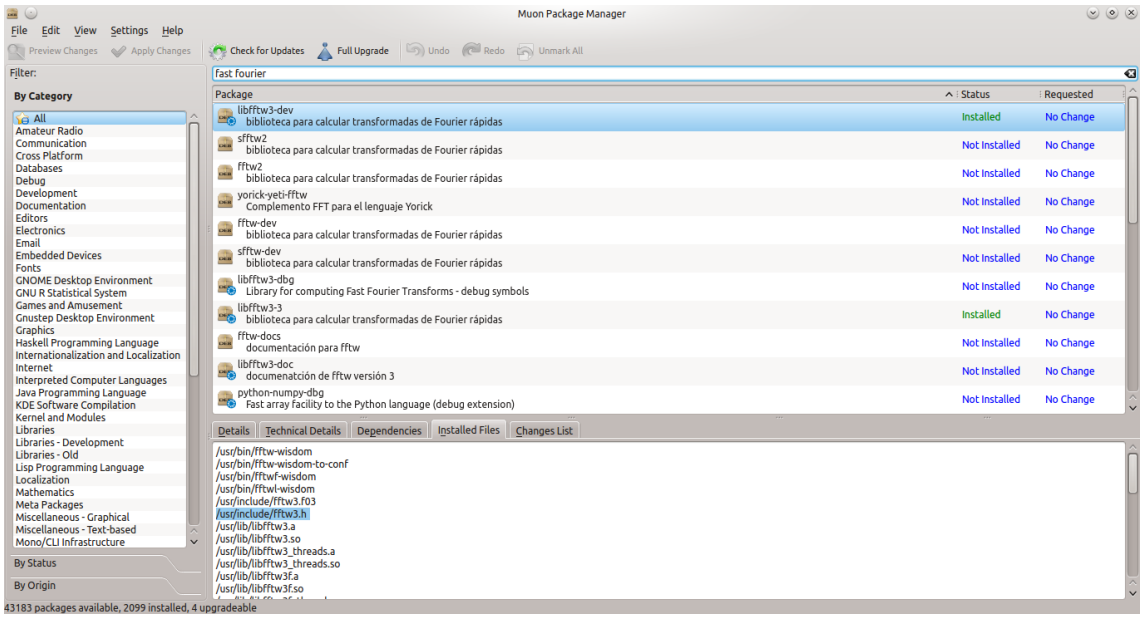

Fig. 5.3.2

Visualització de arxius instal·lats pel paquet libfftw3-dev al gestor de paquets Muon

Encara que es puga vincular el projecte en qualsevol llibreria de transformades de Fourier, *osgOcean* agafa per defecte la llibreria FFTSS (de llicència LGPL), pel qual es recomana emprar aquesta. Alguns dels problemes que es poden trobar a l'hora de fer-ho amb libfftw3 (la que es troba per defecte a *KUbuntu*) es que no es troben fitxer com fftw3.h. Açò es deu a que la FFTSS, en la que ha sigut comprovada la seua compilació, substitueix aquest *include*[12]per *fftw3compat.h*. Es per això perquè es recomana muntar la FFTSS.

[12] ***include:*** *Arxius que solen portar la extensió .h (p. ex. en llenguatge C) que contenen els noms del arxius en el codi necessari perquè els cossos dels arxius al que fan referència puguen funcionar apropiadament.*

Es torna a executar el *ccmake* al directori *build* de l'*osgOcean* i s'introdueixen als camps requerits les direccions de les llibreries i *includes* que siguen demanats. Aquestes dependran de on s'hagen instal·lat les mateixes al HDD de l'ordinador. Una possible solució és la que ací es presenta (fig. 5.3.3).

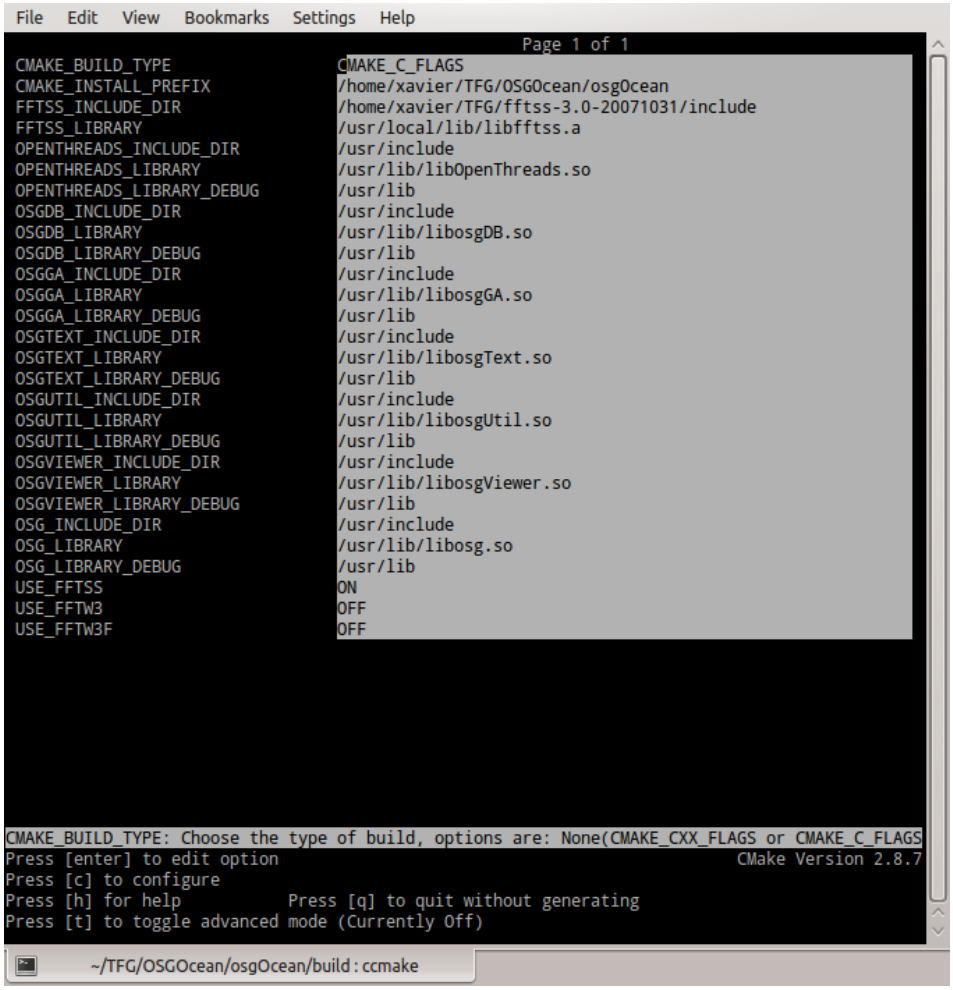

Fig. 5.3.3

Exemple de CCMake funcional

Així doncs, dins d'aquesta pantalla, es pressiona C per configurar i G per generar. Serà llavors quan apareguen possibles errors que caldrà corregir. Una volta corregits i ja en el terminal, es tecleja *make*. Aquest procés pot ser una mica llarg. En aquesta instrucció es compila tot el projecte utilitzant les direccions que han sigut introduïdes anteriorment amb ajuda del *CCMake*. Una volta finalitzat el temps d'espera, es fa make install perquè es facen còpies de la compilació als directoris *osgOcean/bin* i *osgOcean/lib*.

Per últim es crearà una nova carpeta al directori arrel d'*osgOcean*, anomenada per exemple *build-osgocean*, i es seguiran els següents passos:

```
cd <directori arrel osgOcean>

mkdir build-osgocean

cd build-osgocean

cdmake ../osgOcean

sudo make install
```

Posteriorment, aquesta cacrpeta serà emprade per tornar a muntar l'exemple funcional oceanExample per fer sortir modificacions que es veuran més endavant.

5.4 Modelització tridimensional

5.4.1 Glider

En aquest pas es detalla com modelitzar amb la ferramenta lliure Blender 2.71 [16] l'objecte submarí escollit: el glider.

Per modelitzacions de cossos i objectes simples, una de les tècniques més assolides és crear una de les formes primitives de les que disposa Blender i, a partir d'ella, anar modificant-la amb les ferramentes proporcionades fins arribar a obtindre la forma final desitjada. Així doncs, es comença per un cilindre simple. El primer que es nota al iniciar Blender es que ja és creat un escenari que no està buit. Conté un cub, una càmera i un focus de il·luminació. A continuació una breu explicació del UI de Blender (fig. 5.4.1.1).

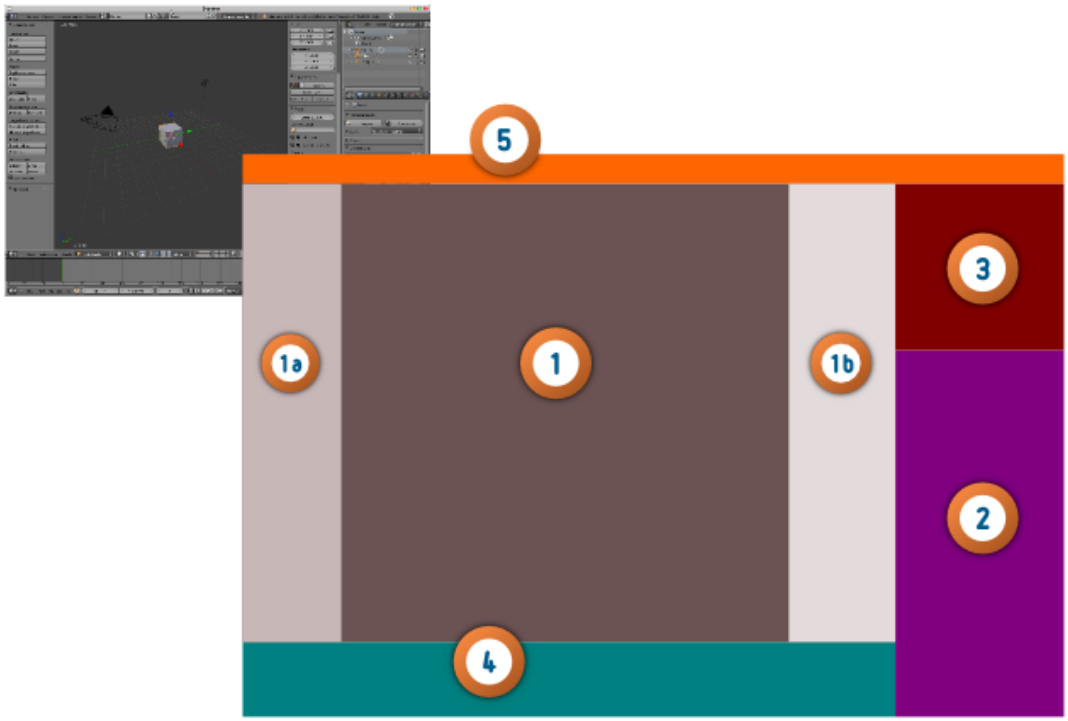

Fig. 5.4.1.1

Descomposició per àrees del UI de Blender (intef)

Àrea 1: On es treballa el projecte 3D
Àrea 1a: Quadre Ferramentes (SHC "T")
Àrea 1b: Quadre Propietats (SHC "N")
Àrea 2: Panels
Àrea 3: Llistat
Àrea 4: Línia de temps
Àrea 5: Editor

Per una guia detallada del funcionament de Blender es recomana seguir la guia creada per l'intef (Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado) [6].

Es suprimeix el cub de l'escenari i s'introdueix el cilindre des de *Create -> Cilinder* (fig. 5.4.1.2).

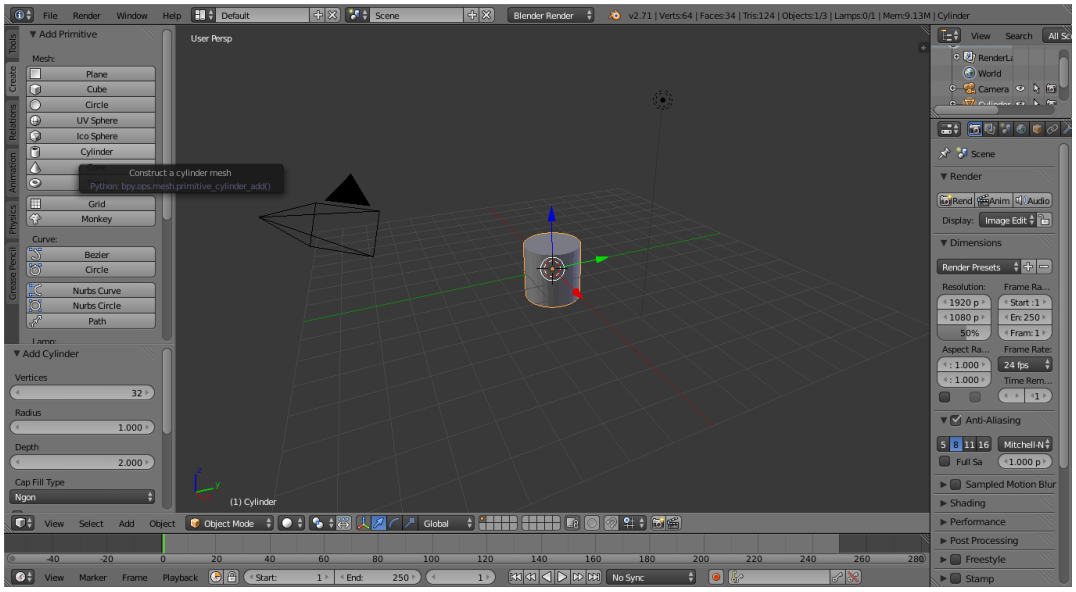

Fig. 5.4.1.2

Introducció de formes geomètriques senzilles

És necessari a partir d'ací (i constantment) fer ús dels manipuladors. Els manipuladors són les ferramentes que proporciona Blender per fer transformacions bàsiques com translacions (*G*), rotacions (*R*), escales (*S*), etc. Es poden fer de manera més gràfica amb el ratolí mitjançant les opcions activades en groc (fig. 5.4.1.3) en *Object Mode*, encara que tant en la pràctica real com a la present memòria es farà referència a elles pels seus shortcuts (SHC).

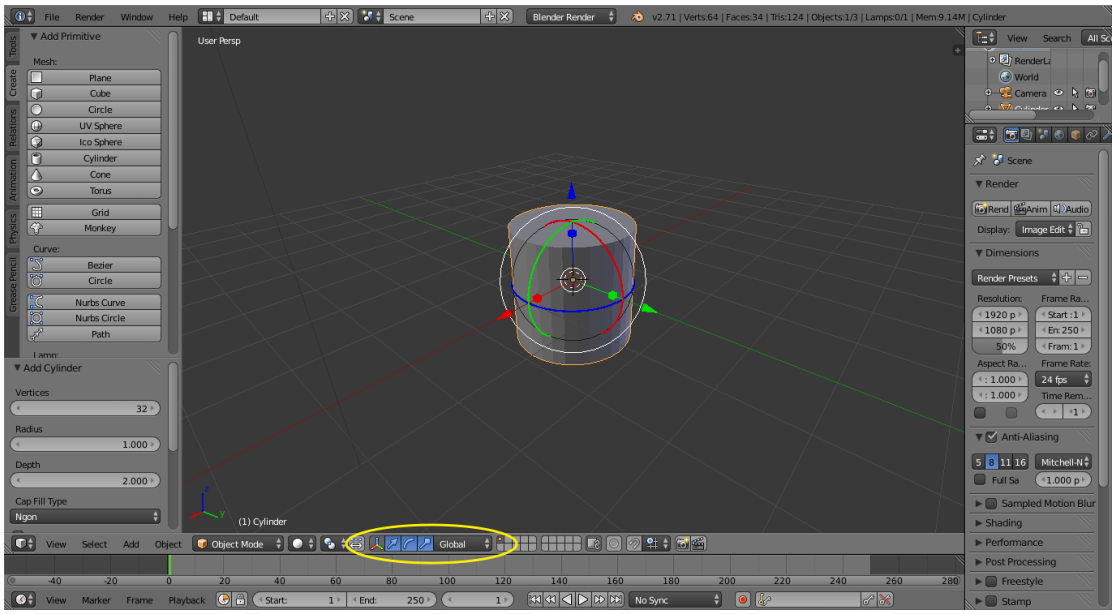

Fig. 5.4.1.3

Manipuladors de transformacions

Cal tindre en compte que els paràmetres bàsics de les figures geomètriques introduïdes han de ser configurades abans de fer cap modificació a l'objecte. Després no hi haurà oportunitat de fer-ho. Així doncs es col·locarà un cilindre de radi 0,5 [13]en horitzontal amb *R, X, 90, Intro*. Amb *R* es treballa en rotació, en *Y* es bloqueja l'eix Y per treballar només en ell i en 90 es diu a Blender que es vol girar l'objecte seleccionat 90º. Després d'açò es pretén allargar el cos del glider per utilitzar-ho com a base de la modelització. Açò es fa en *S*, *X*, i es desplaça el ratolí fins arribar a una longitud adequada. Es fa clic en 1 al teclat numèric[14] per tindre una vista de perfil i 5 per fer-la ortogonal en lloc de en perspectiva. Hauria de quedar quelcom així (fig. 5.4.1.4).

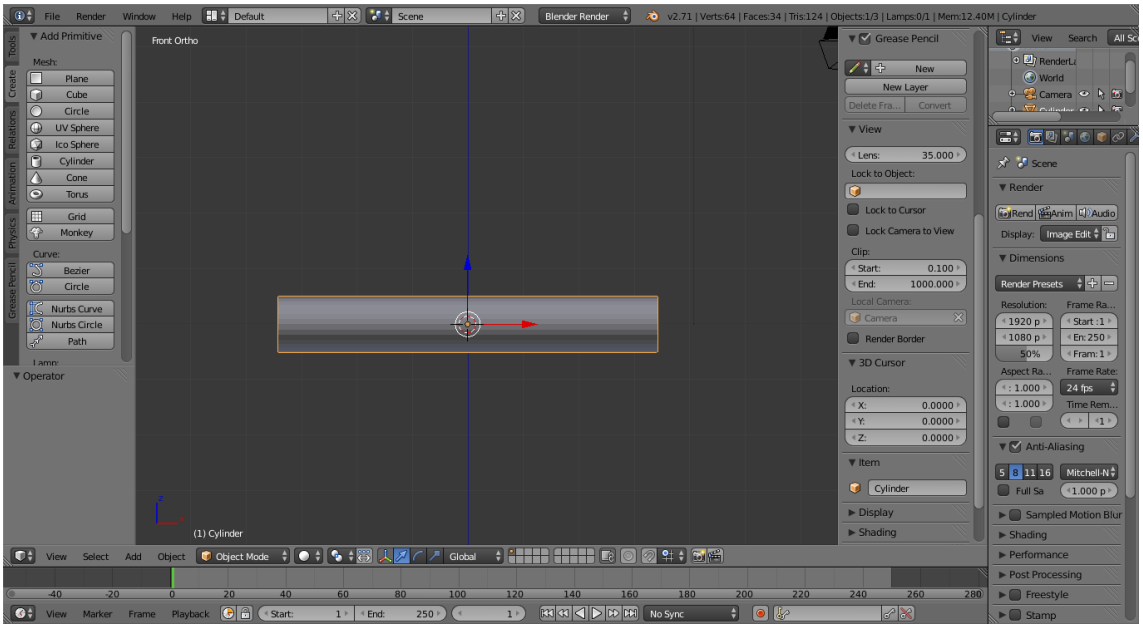

Fig. 5.4.1.4
Cos del glider

En les simples instruccions anteriors es pot abordar la creació del cap del glider. Es crea un cilindre de 0,6 de radi d'anell, un altre de 0,55 més llarg i, per últim, una esfera que farà de terminació de l'aparell. S'adjunta imatge (fig. 5.4.1.5) per fer-se una idea de com hauria de ser la representació a aquest pas.

[13] *Pot sorprendre referir-se a magnituds com la longitud sense unitats, però cal tindre present que Blender no és un programa CAD (computer-aided design), així que en compte de buscar la precisió de la magnitud es busca una representació més atractiva que funcional.*

[14] *L'aprofitament de tecles a Blender es tal que els nombres del teclat numèric tenen funcions distintes, en aquest cas de vista sobre l'escena. Es recomana utilitzar un equip que compte en teclat numèric per guanyar soltesa i agilitat.*

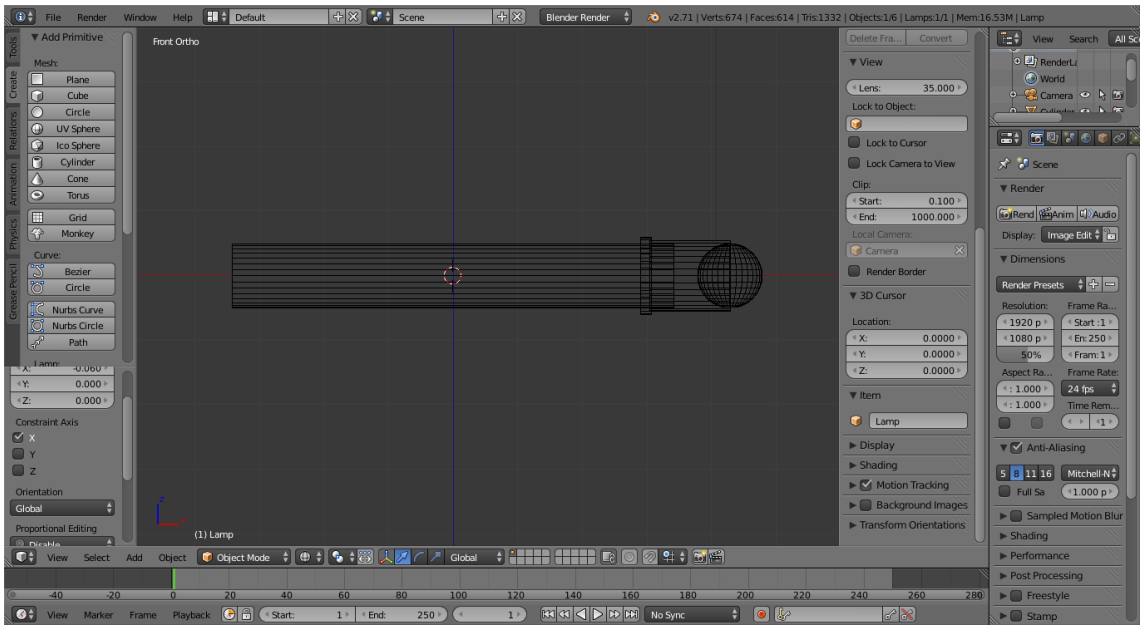

Fig. 5.4.1.5

Cos del glider i cap

Perquè es veja millor s'ha passat al mode wireframe, disponible a la barra inferior al *Object Mode* (fig. 5.4.1.6). També es pot, simplement, fer clic al seu shortcut: Z.

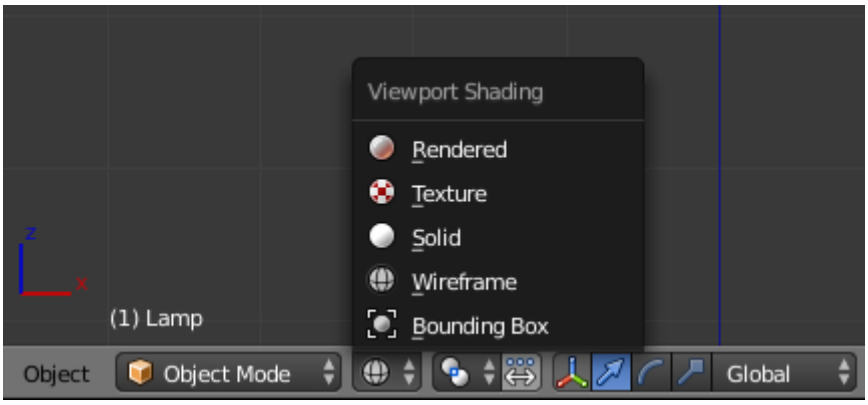

Fig. 5.4.1.6

Canvis de visualització a Object Mode

Per la part de darrere es pot recórrer a una ferramenta molt emprada: *copy paste*. A Blender es fa en Shift+D. D'aquesta manera es copiarà l'objecte abans seleccionat i es procedirà a la manipulació del nou sense que s'afecte a l'original. Per la cua es comença amb un con de radi , i profunditat 2,0.Es manipula fins que quede així (fig. 5.4.1.7).

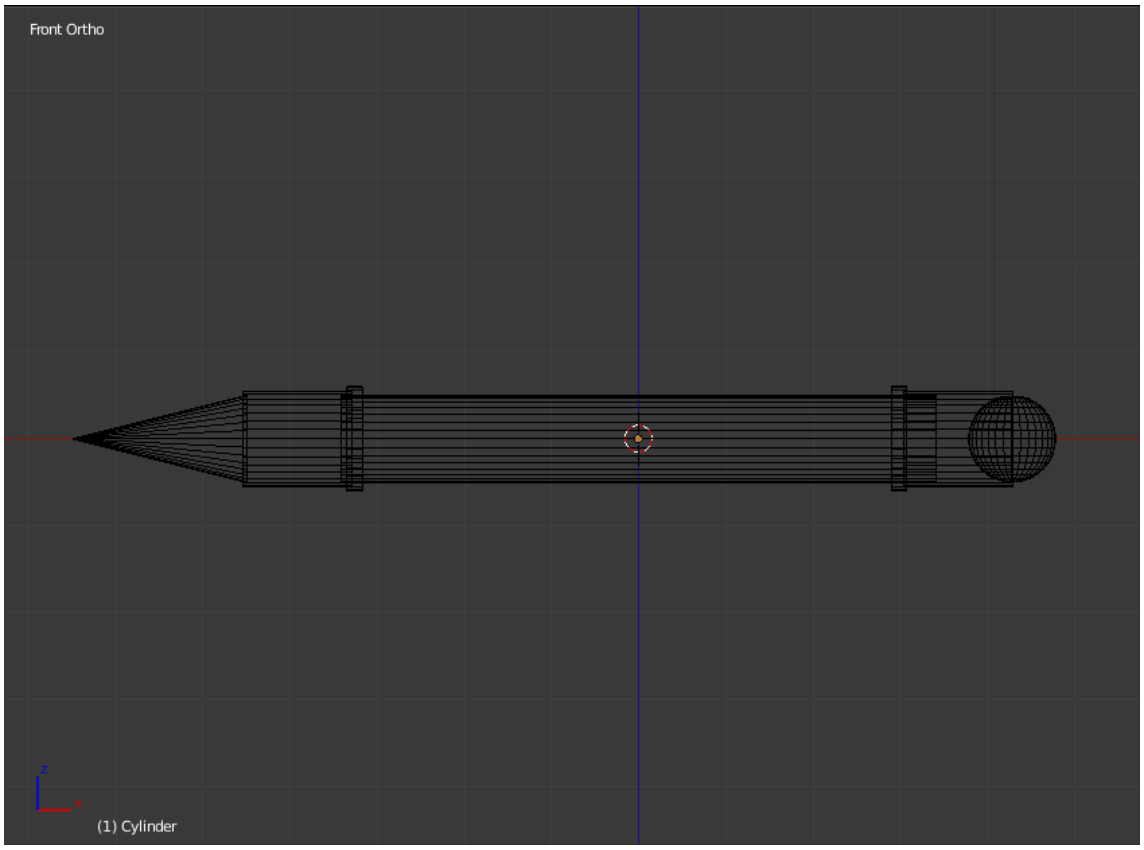

Fig. 5.4.1.7

Cos glider avançat

Per la cua es fa un llarg cilindre de radi 0,05. A la part final un cilindre de 0,1 i un altre superposat de 0,09 (fig. 5.4.1.8)

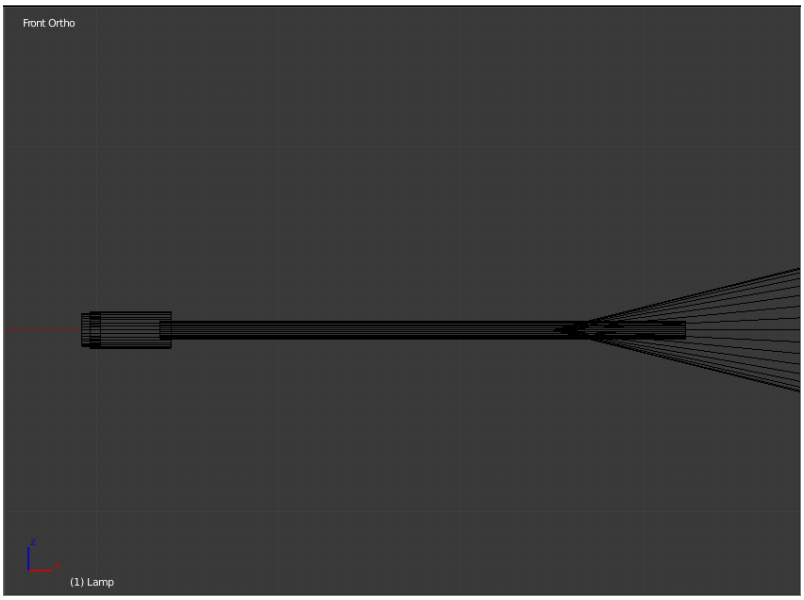

Fig. 5.4.1.8

Cua del glider

A continuació es procedirà a la creació de les aletes de la cua. Per començar es crearà un cub que es manipularà com hem vist abans fins que tinga unes dimensions adequades, sense importar encara la forma (fig. 5.4.1.9).

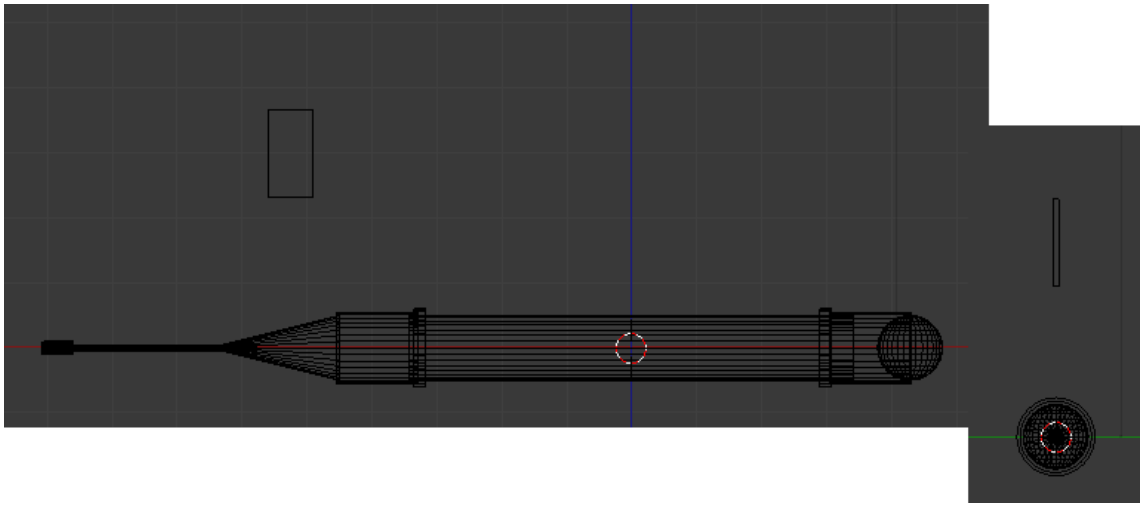

Fig. 5.4.1.9
Forma primitiva de les aletes

Una volta inclosa la forma primitiva per treballar-hi en ella, cal manipular-la perquè parega realment una aleta. Es passa, llavors, al *Edit Mode*. En aquest mode, la gran característica és que no es poden seleccionar els objectes per sí mateixos, sinó que s'han de seleccionar vèrtexs, eixos i cares. Es seleccionen la parella de vèrtexs del cantó superior esquerra i es mouen amb el shortcut *G* cap a la dreta (*G*, *X* i menejar el ratolí). Com ja s'haurà experimentat, és convenient tindre el ratolí prop de la zona d'operació perquè es puga estendre la manipulació fins on es vullga (fig. 5.4.1.10).

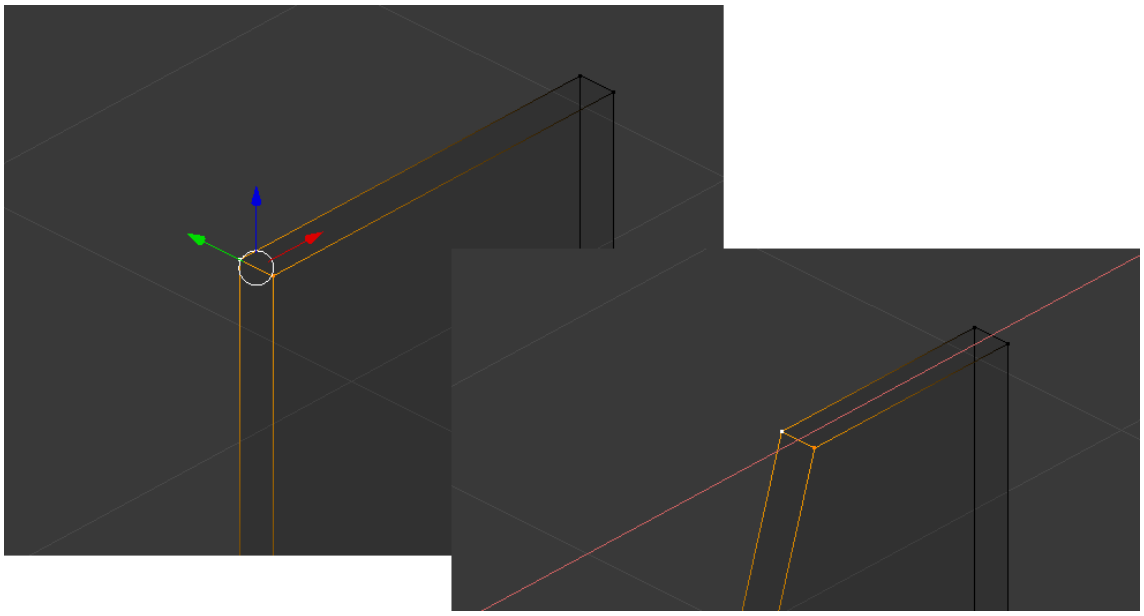

Fig. 5.4.1.10
Creació d'una aleta trasera

Després es farà el compartiment posterior. Fer l'estructura principal es senzill, però s'aborda un nou problema: el cantó redó del mateix. Per fer-lo. Es crea una xicoteta estructura de la mateixa amplària i altura que el compartiment, ja que va a ser part d'ell, però tan curta com gran es pretén que siga el cantó. Una volta creat aquest objecte, es passa a *Edit Mode* i es selecciona l'eix que es pretén bisellar. Es fa *Ctrl+B* i es desplaça el ratolí fins que el bisellat ocupe tota la llargària que es pretenga. Una volta fet, sense fer clic per confirmar, es meneja la roda del ratolí perquè el bisellat siga més o menys pronunciat (fig. 5.4.1.11).

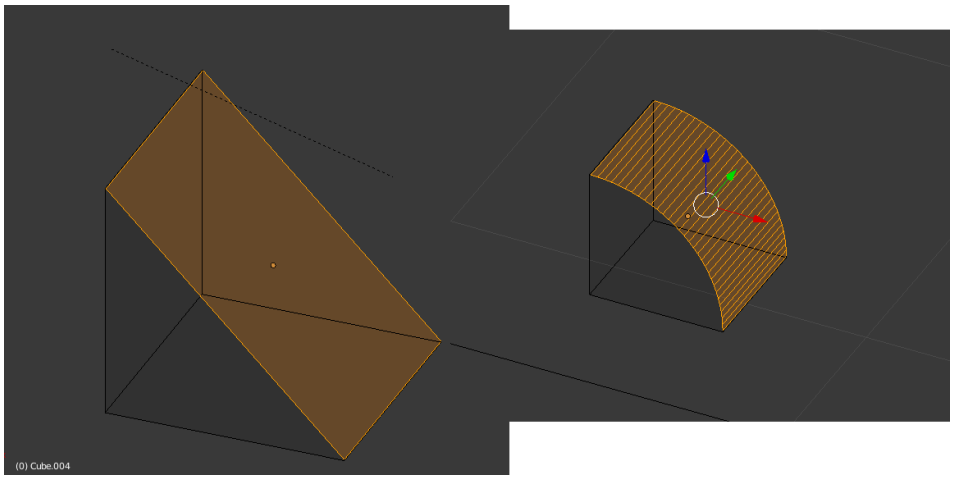

Fig. 5.4.1.11
Bisellat inicial i final

Per finalitzar la part de la cua s'ajunten les dos parts del compartiment, primer individualment fins que encaixen al gust del dissenyador. Una volta es té l'objecte final, és preferible que es tracte com un únic objecte en compte dels dos que son. Es per això que, ja físicament units, es seleccionen els dos objectes i es pressiona *Ctrl+J* per unir-los. A partir d'ara es comportaran com si d'un únic objecte es tractara. S'ajunta en la part posterior i queda una cosa tal que així (fig. 5.4.1.12).

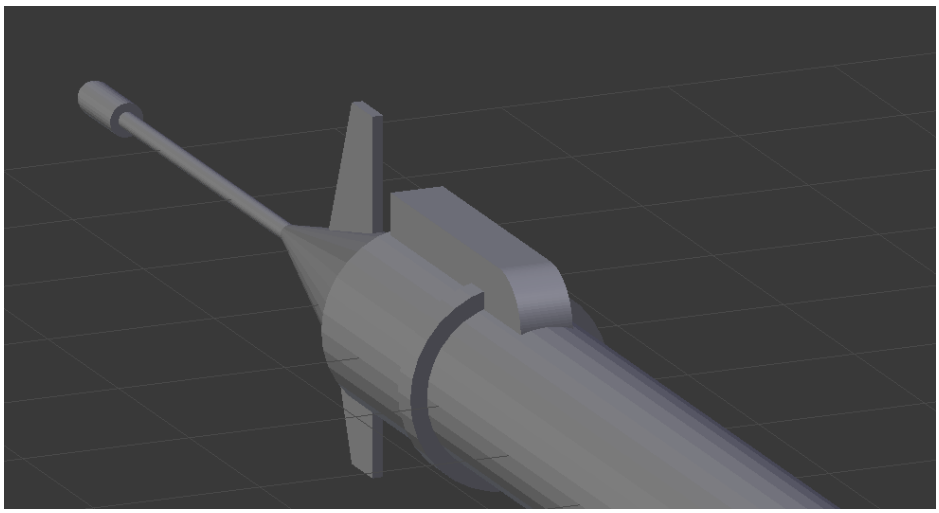

Fig. 5.4.1.12
Aspecte final de la cua del glider

Seguidament es passa a la modelització de les ales principals. Primerament es crea un anell a la part central del glider (semblant al dos anteriors) i es modelitzen les ales. Per tal fi, es crea un cub que es redimensiona fins que puga servir de base. La part posterior acaba en punta, així que ja en el *Edit Mode* es selecciona la cara posterior i es redueix la mateixa amb *S*, *Z* i en menejar el ratolí (fig. 5.4.1.13).

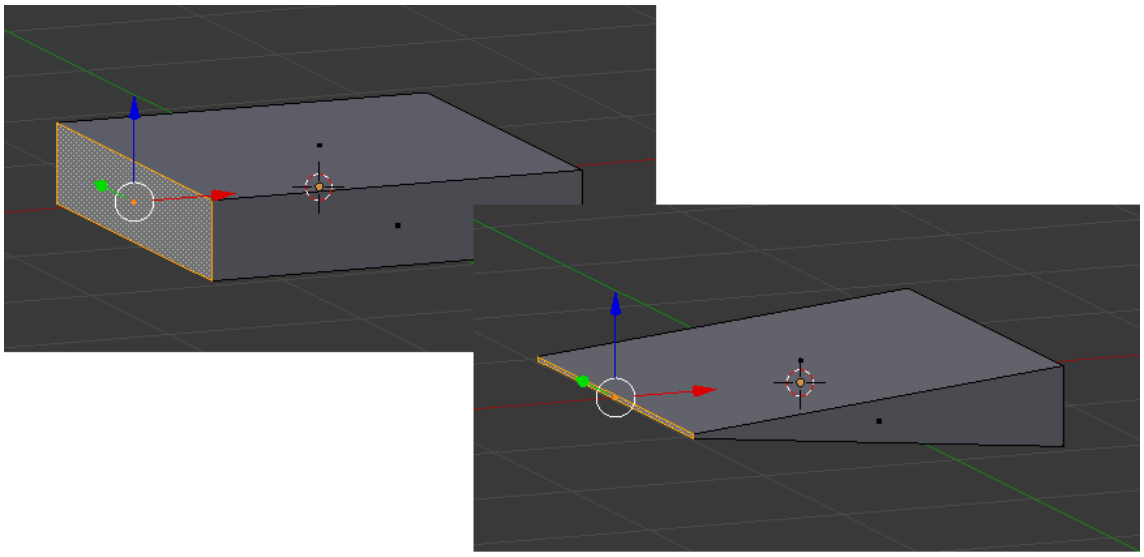

Fig. 5.4.1.13
Part posterior de l'ala central

Ara es repeteix la tècnica del bisellat de la pàgina anterior en les dues arestes de la part frontal i es seleccionen les divisions pertanyents per tindre el resultat desitjat (fig. 5.4.1.14).

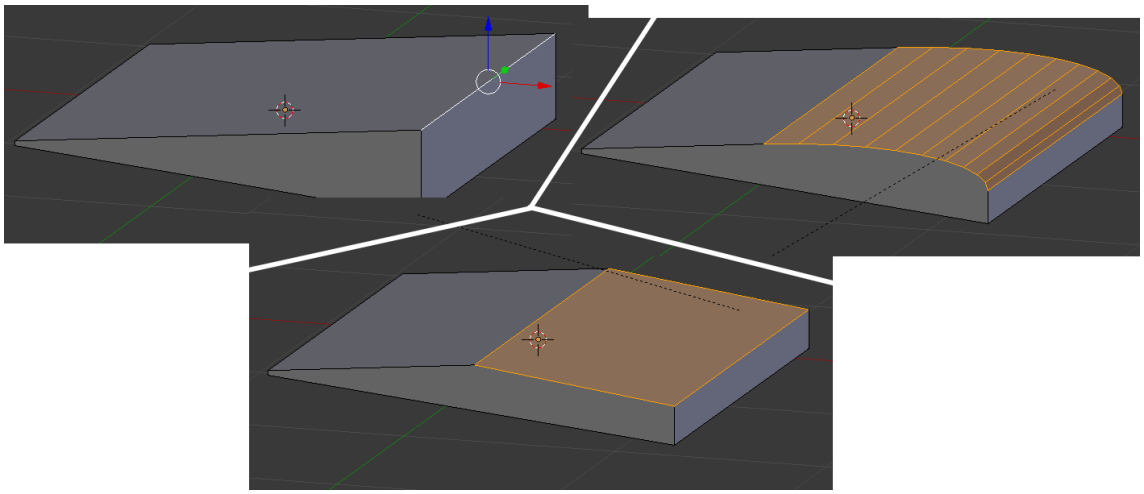

Fig. 5.4.1.14
Part frontal de l'ala central

Finalment, s'inclinen les ales 5º respecte a la posició horitzontal i la modelització en sí mateixa queda finalitzada. A continuació algunes imatges del resultat.

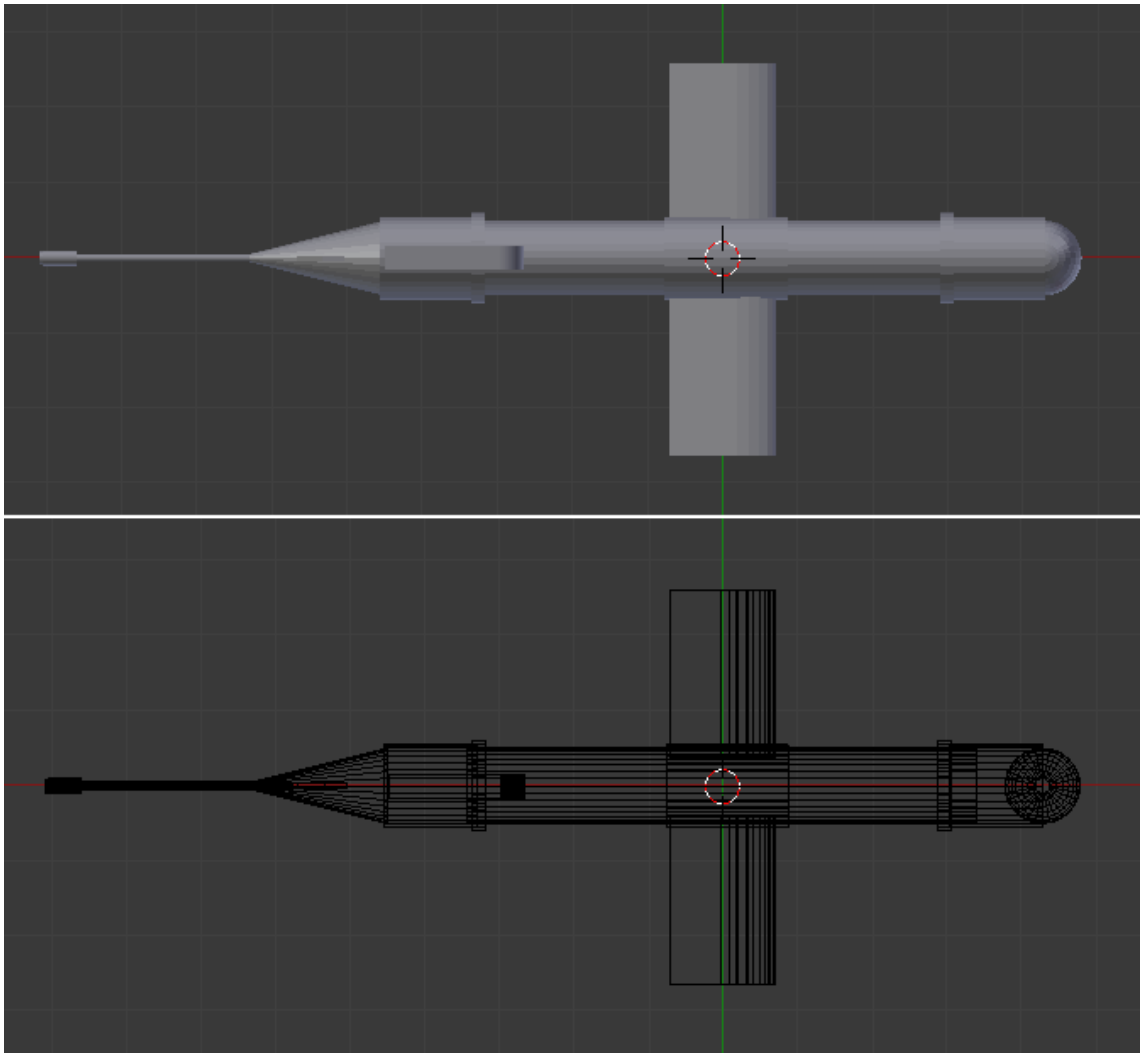

Fig. 5.4.1.15
Vista zenital del glider

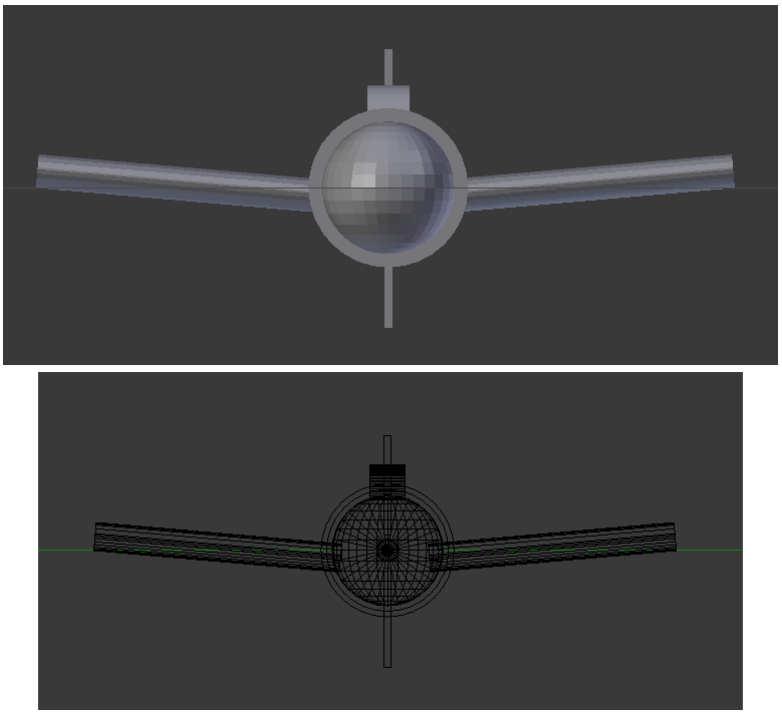

Fig. 5.4.1.16
Vista frontal del glider

Als següents passos es tractaran aspectes que també afecten al procés de modelització però que poden resultar secundaris segons l'aplicació. En qualsevol cas s'expliquen a continuació el passos a seguir. Primerament es seleccionen tots els objectes que formen part del glider i se'ls aplica un ombrejat suau: *Shading -> Smooth* (fig. 5.4.1.17).

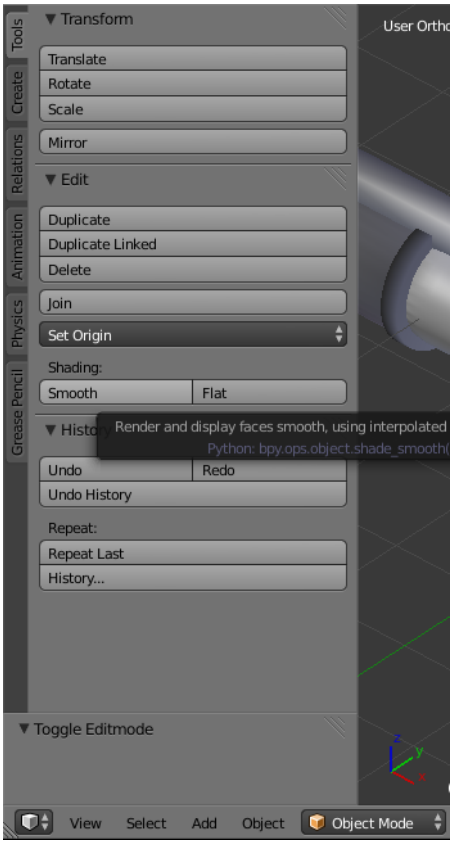

Fig. 5.4.1.17
Activació ombrejat suau a Blender

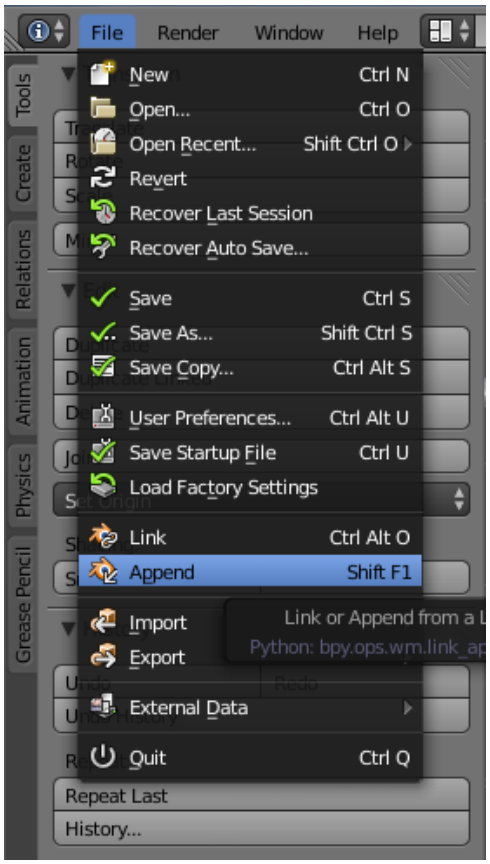

Fig. 5.4.1.18

Vincular materials a Blender

Un dels primers serà escollir els material que conformaran la totalitat del glider. Per descomptat es poden fer a mà, en el propi editor de Blender, ajustant multitud de paràmetres fins donar en la combinació correcta. No obstant això, també es pot optar per altra solució: descarregar-los d'internet. Existeixen múltiples repositoris de materials per Blender a internet, com el *Open Material Repository*, de *Blender Materials [11],* en el qual tots els materials que es troben són lliures per ser inclosos en tot tipus de aplicacions (comercials i no comercials). Una volta descarregats els materials d'interès es recomana guardar-los al directori arrel del projecte (p. ex. a la carpeta Materials). Ja dins de Blender es fa clic en *Files -> Append* (fig. 5.4.1.18). Cal anar al directori on s'han guardat els fitxers .blend, seleccionar la carpeta *Materials*, els materials corresponents i clic en *Link/Append from Library*.

Una volta fet açò, es selecciona un objecte i des de Panels, a l'opció Materials, es crea un nou i es selecciona un dels que s'hagen vinculat (fig. 5.4.1.19).

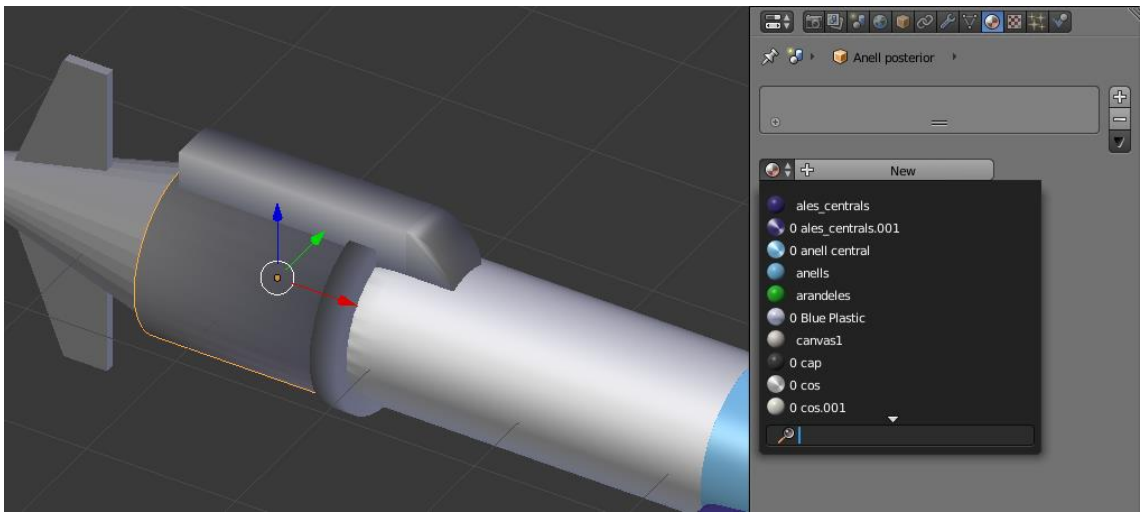

Fig. 5.4.1.19

Vincular un material a un objecte

Els materials emprats han sigut trets del repositori del enllaç anterior. A continuació es presenta una imatge amb el colors en hexadecimal utilitzats per la modelització (fig. 5.4.1.20).

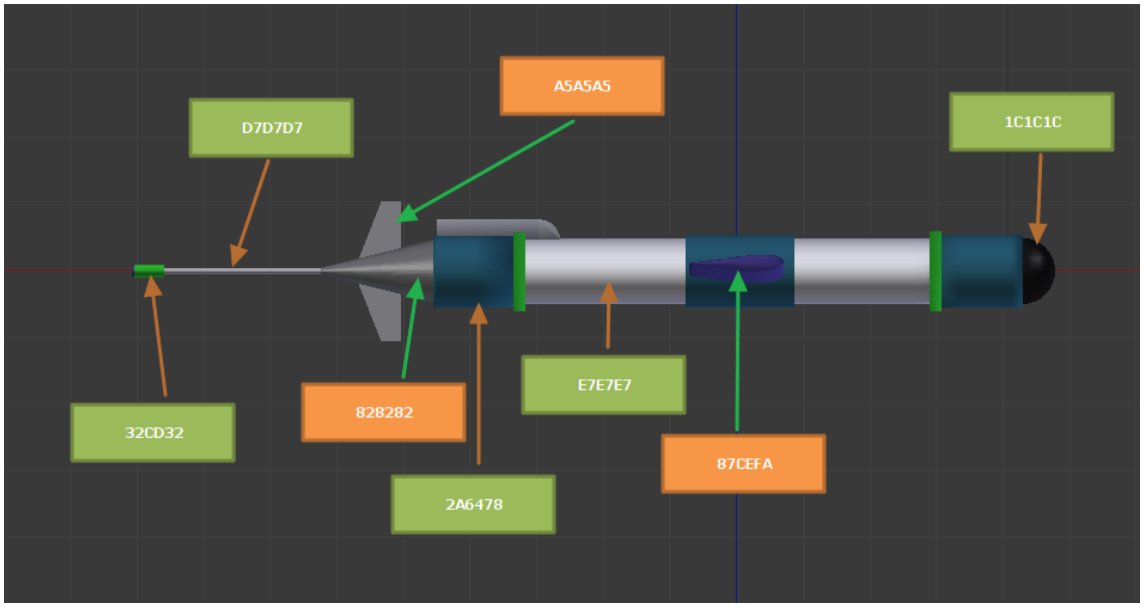

Fig. 5.4.1.20

Colors en hexadecimal del glider

A tots els objectes se'ls aplicarà el modificador *Subdivision Surface*. A Blender, els modificadors son múltiples, i pretenen donar als objectes qualitats físiques que tinguen un comportament semblant a la realitat per fer dels projectes dels usuaris una experiència més fidel. Un dels més emprats es el que ha sigut anteriorment esmentat, *Subdivision Surface*, que suavitza les arestes del cossos per donar-los una aparença més vistosa. S'aplica des de *Panels*, *Modifiers -> Add New Modifier -> Subdivision Surface* (fig. 5.4.1.21).

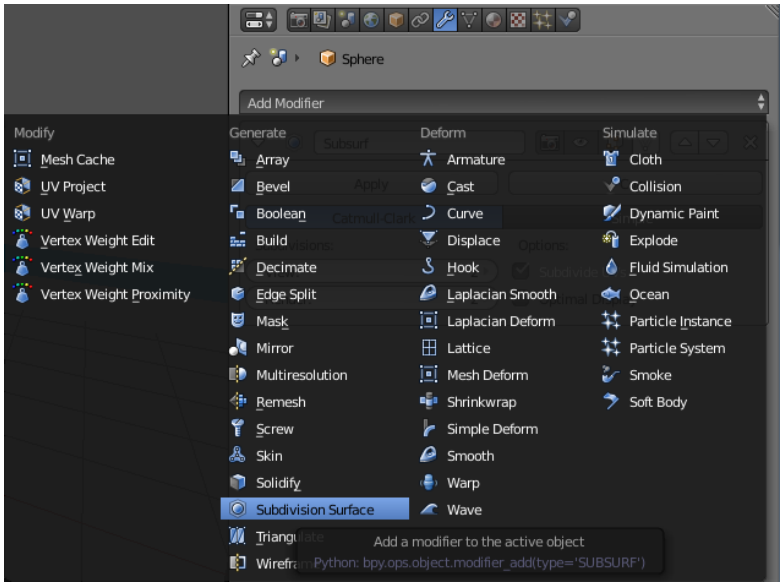

Fig. 5.4.1.21

Afegir modificadors a Blender

Pot observar-se que el resultat no siga l'esperat, sobretot als cantons dels objectes on s'ha aplicat. Açò es dona perquè els eixos, vèrtexs i cares que configuren els cossos inicialment no són suficients . Per a remeiar-ho, es selecciona l'objecte que estiga donant problemes, es passa a *Edit Mode* i es fa *Ctrl+R* (fig. 5.4.1.22). Així apareixerà, segons la posició del ratolí, un nou anell d'eixos que donarà major resolució al cos. Amb la roda del ratolí es diu quants eixos es pretén que apareguen, i desplaçant-lo es confirma la seua posició (com ja s'ha dit, normalment, als extrems).

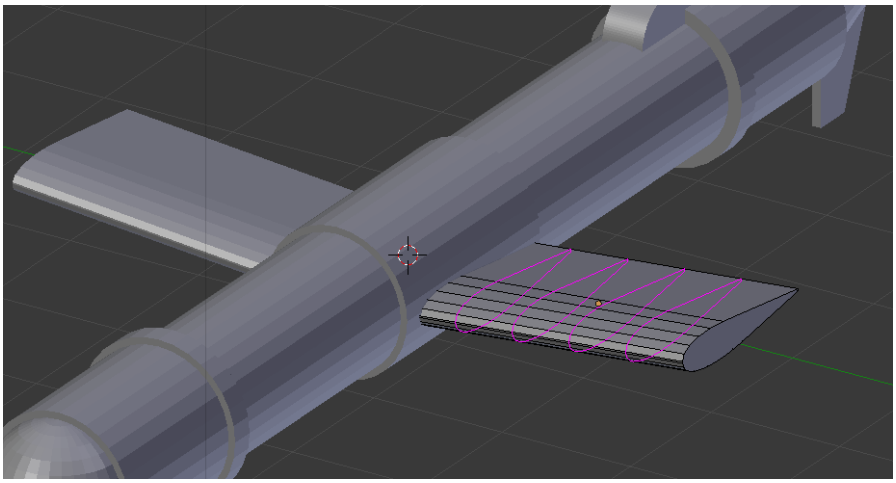

Fig. 5.4.1.22
Creació de bucles a Edit Mode

Una volta fetes totes aquestes modificacions el glider pot quedar així (fig. 5.4.1.23).

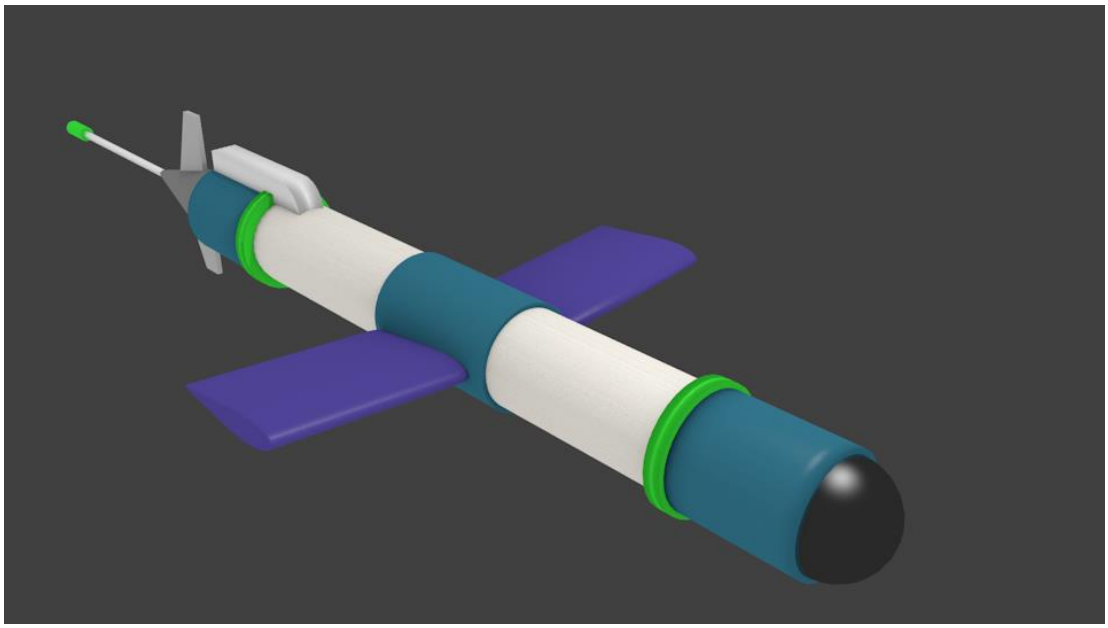

Fig. 5.4.1.23
Render[15] del glider

[15] ***Render:*** *Imatge presa des de la cambra de Blender amb tots el modificadors aplicats*

Ara es pretén afegir un xicotet detall: el logo de la UPV (Universitat Politècnica de València) al glider. Açò requerirà un ús específic d'altres modes de Blender. El primer pas serà descarregar el logo de la UPV de la pàgina web oficial. Després, a Blender, es clica a la selecció de mode i es dona a *UV Editing*. La pantalla presentarà un aspecte com el de la figura (fig. 5.4.1.24).

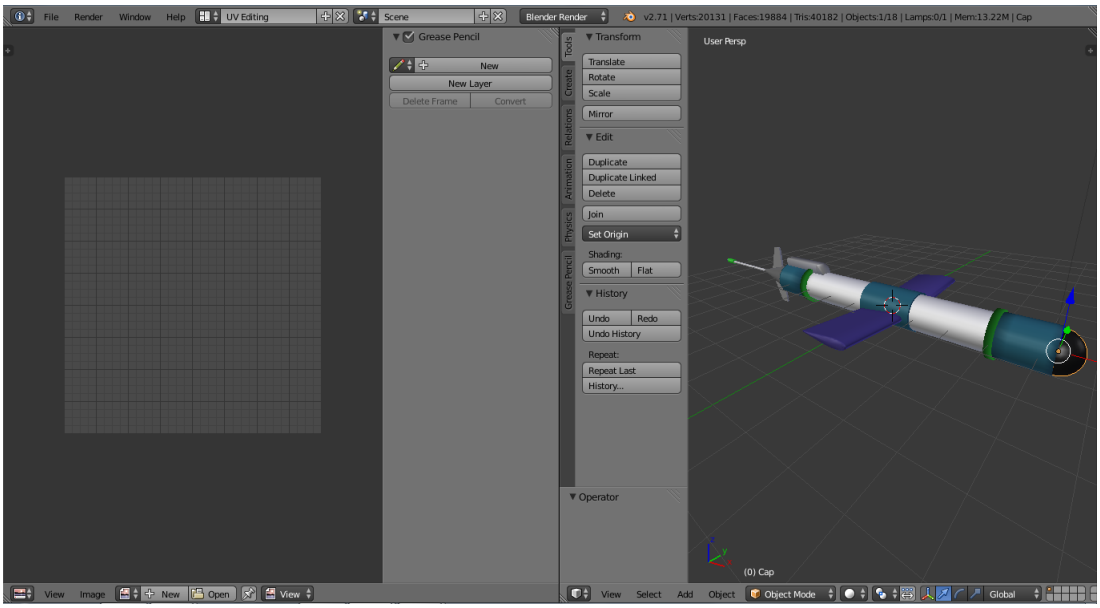

Fig. 5.4.1.24

Pantalla amb el UV Editing

Es fa clic en Open i es selecciona el logo de la UPV anteriorment descarregat. (fig. 5.4.1.25).

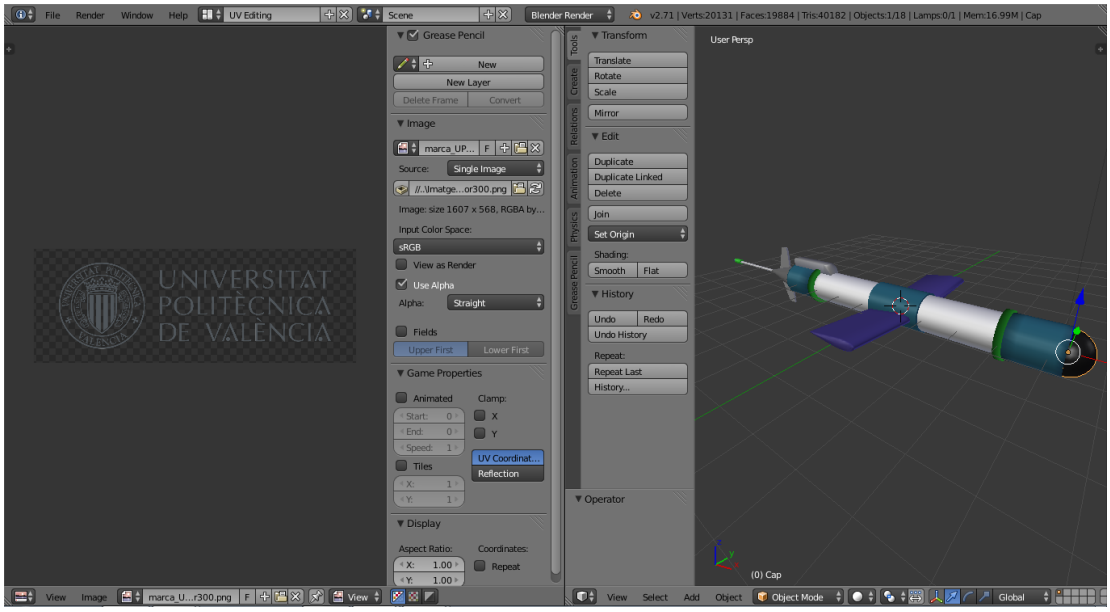

Fig. 5.4.1.25

UV Editing amb el logo de la UPV carregat

A la finestra *Scene* es selecciona l'objecte al que es ficarà el logo (el cos del glider) i en *Edit Mode* es seleccionaran les cares a les quals es ficarà tot el logo (fig. 5.4.1.26).

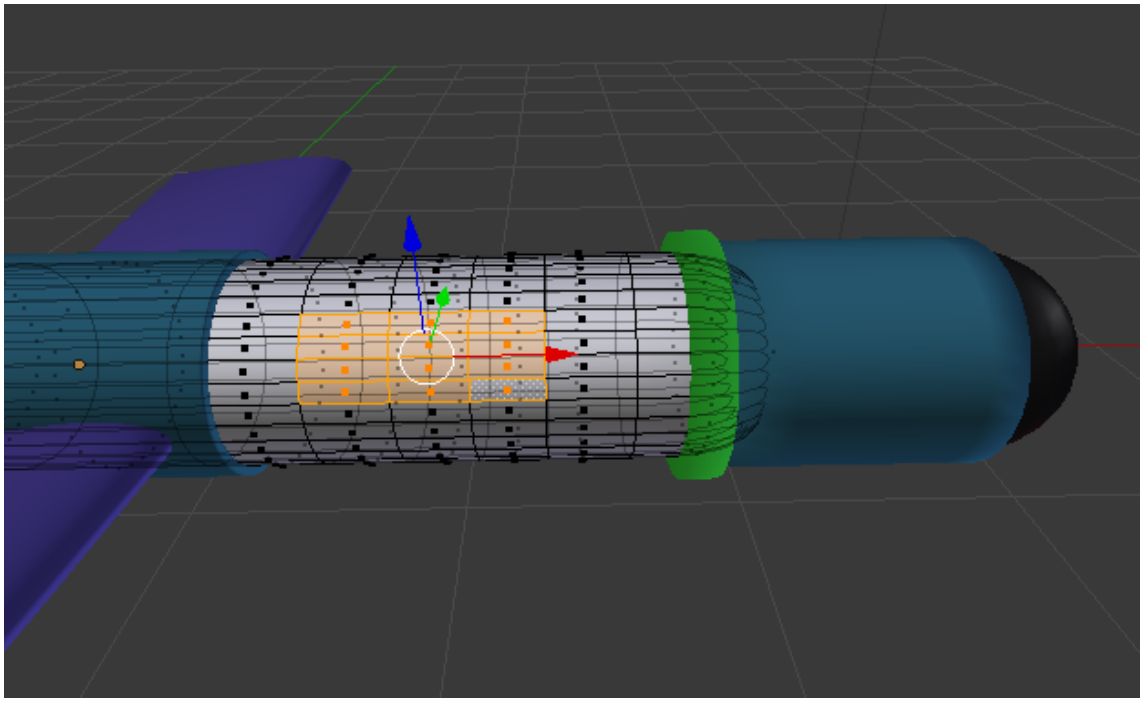

Fig. 5.4.1.26

Selecció cares per desimboltura

Una volta seleccionades es fa clic en *Mesh -> UV Unwrap -> Unwrap* (fig. 5.4.1.27).

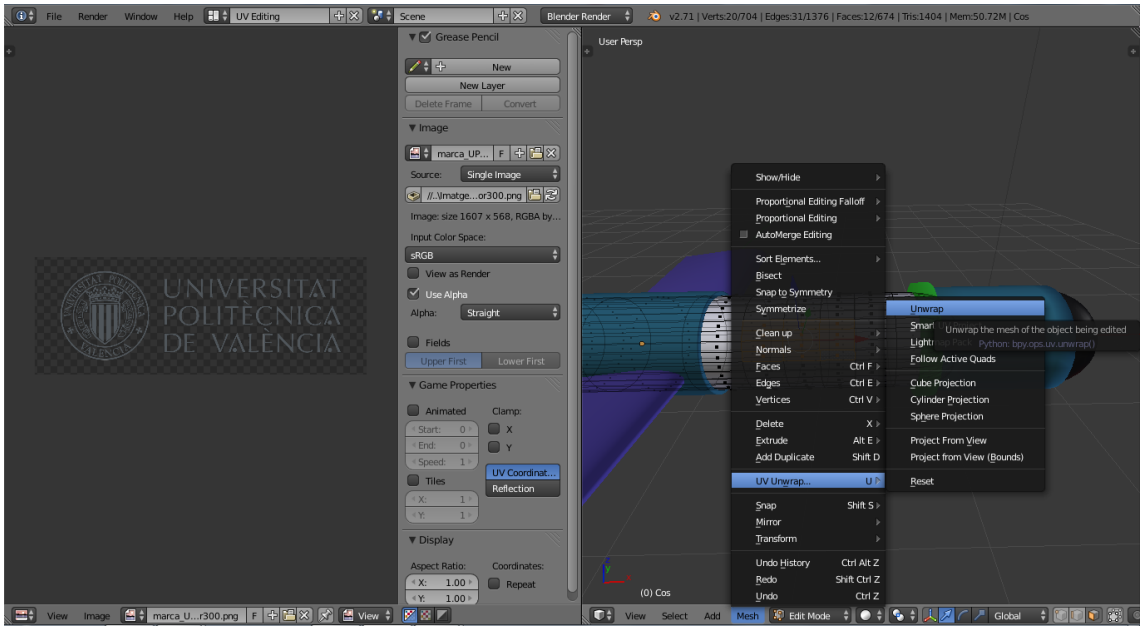

Fig. 5.4.1.27

Desembolicar cares per al logo

Açò transformarà la superfície abans seleccionada en l'*Edit Mode* en una malla que es superposarà al logo del *UV Editing* (fig. 5.4.1.28). La malla pot variar segons el nombre de cares que s'hagen seleccionat

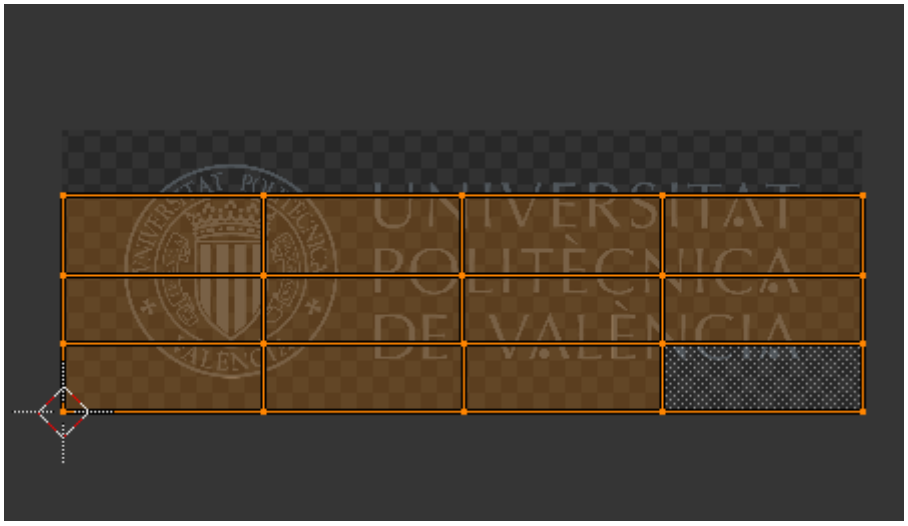

Fig. 5.4.1.28
Malla desembolicada al logo

Una volta redimensionada la malla, es torna al mode *Default*. Si es volen apreciar els canvis en temps real s'ha de clicar al mode d'observació *Texture* (fig. 5.4.1.29).

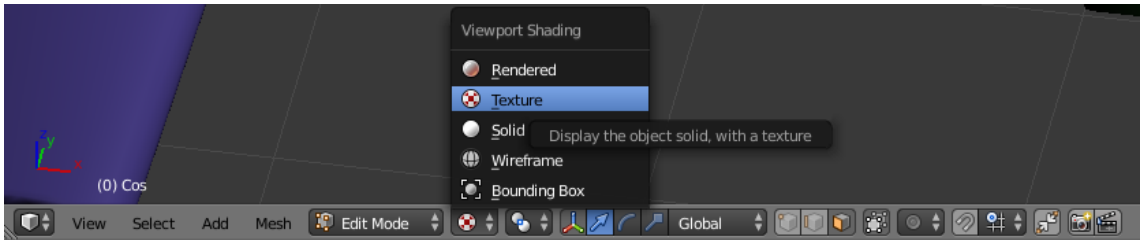

Fig. 5.4.1.29
Canviar a mode Texture

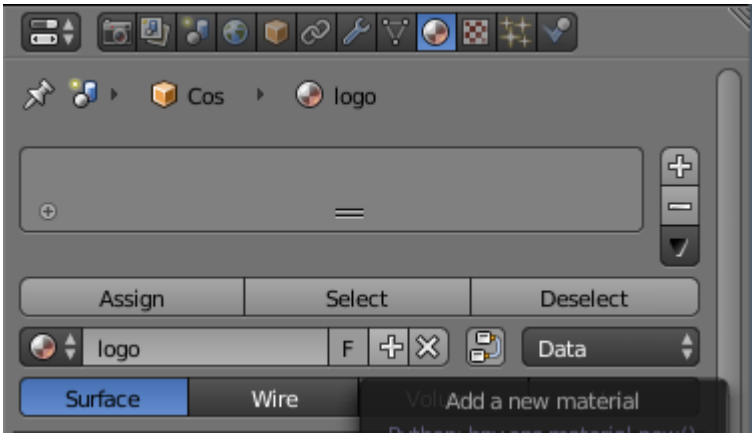

Fig. 5.4.1.30
Assignació d'un nou material

A continuació s'ha de crear un nou material per aquesta regió. Com que es pretén que el fons de la imatge siga el mateix objecte, es dona clic al signe + del costat del material que ja té assignat la malla, es torna a seleccionar el mateix material i es reanomena d'una manera diferent per diferenciar-los (fig. 5.4.1.30).

Per últim cal anar a *Panels*, en *Textures*. Ací s'ha de seleccionar el tipus *Image or Movie*. Més baix, a l'apartat Image en canvia l'*Alpha* per *Premultiplied*. A *Image Mapping* es selecciona *Extension -> Clip* i a *Mapping*, *Coordinates -> UV* (fig. 5.4.1.31).

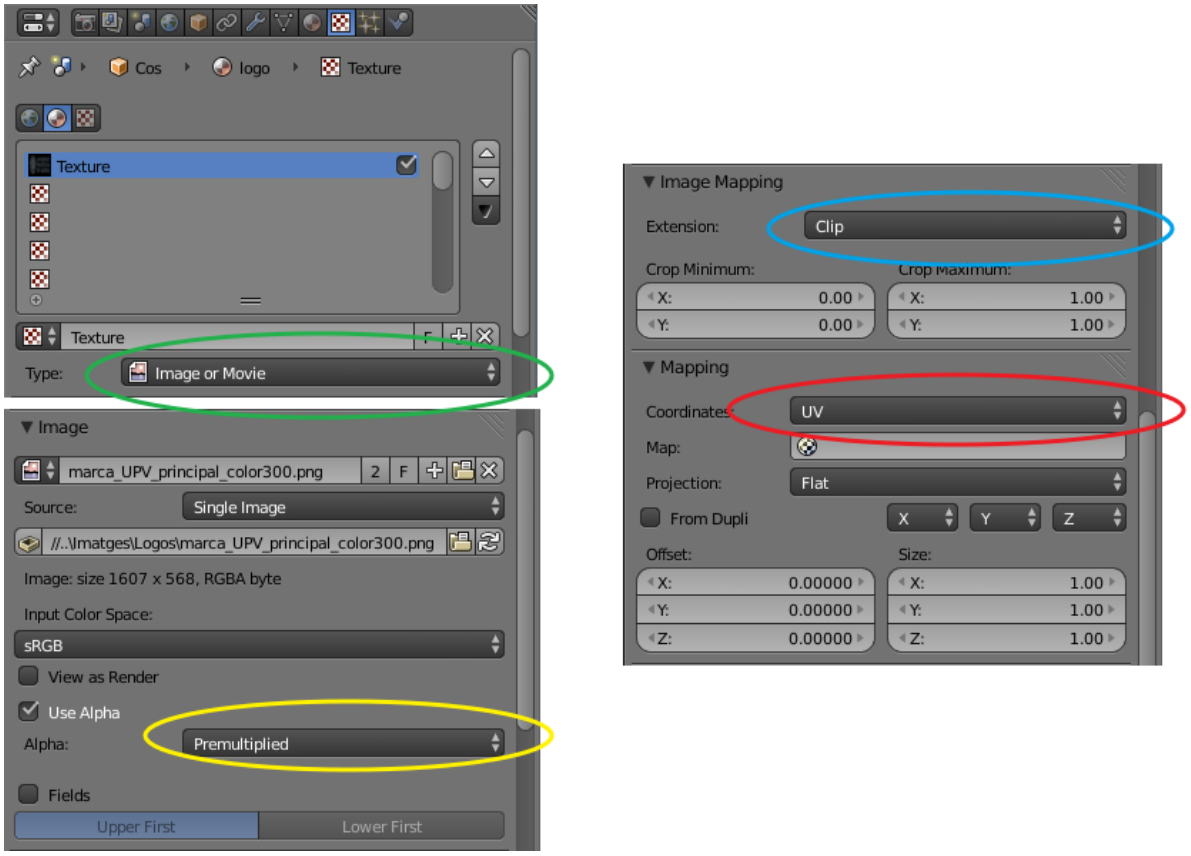

Fig. 4.1.31

Configuracions necessàries per ficar un logo

Un possible resultat pot ser el següent (fig. 4.1.32).

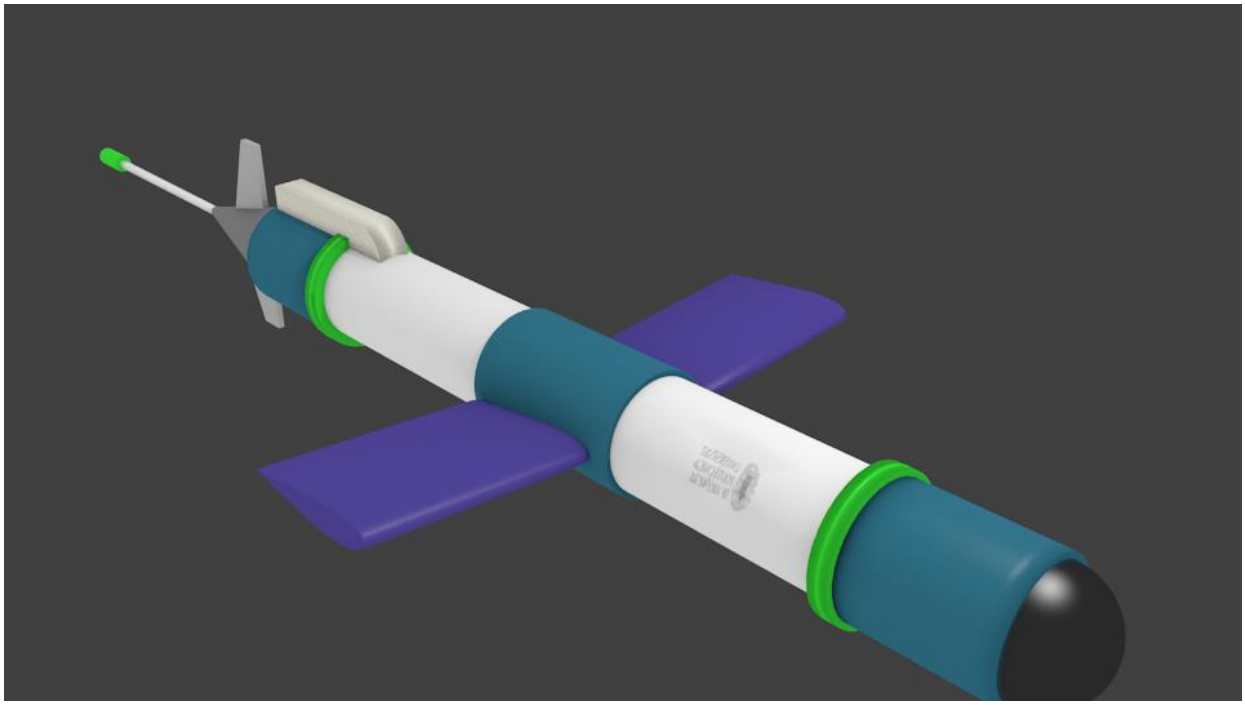

Fig. 4.1.32
Imatge final del glider amb logo

Nota: La il·luminació emprada, coneguda com il·luminació bàsica, s'aconsegueix activant de de *Panels -> World -> Ambient Oclussion* a *Samples*: 10 i amb un llum puntual amb *Ray Shadow*, *Samples*: 6 i *Soft Size*: 1.0

5.4.2 Cistella de piscifactoria

Continuant en la ferramenta Blender, es pretén modelitzar una cistella de piscifactoria tipus. S'obri una nova escena de Blender i s'elimina el cub inicial. Es crea una esfera i es col·loca el punt de vista com *Front Ortho* (1 i 5 del teclat numèric. Es fa clic en Z dins de la escena per passar a un mode de visualització millor per les operacions que s'han de realitzar. (fig. 5.4.2.1).

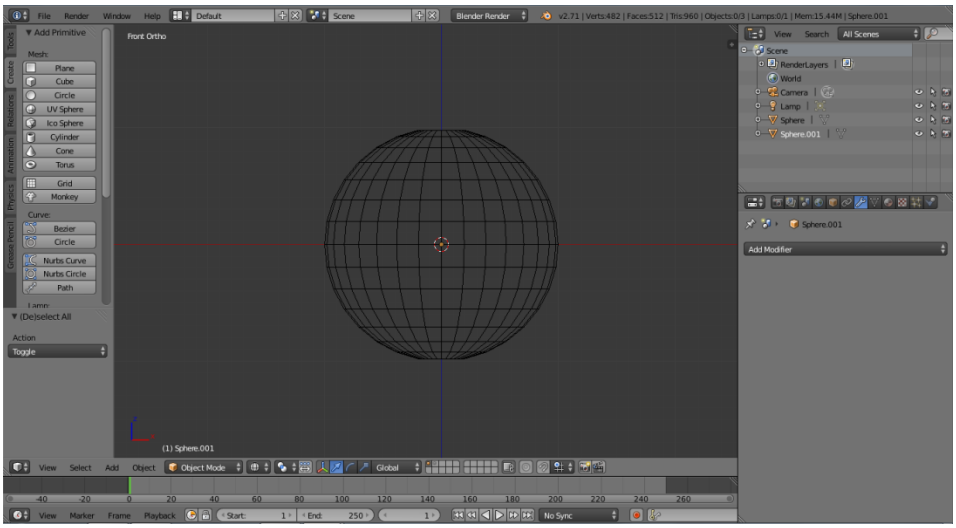

Fig. 5.4.2.1

Esfera inicial

Es passa a *Edit Mode* (TAB), tenint l'esfera seleccionada en *Object Mode*. Tots els vèrtexs i eixos apareixeran marcats. Fer clic en A per seleccionar/deseleccionar tots els vèrtexs d'un objecte. Per seleccionar en un àrea rectangular introduïda per ratolí es fa clic en B i s'utilitza clic dret per ajustar-la. Es suprimeixen els següents vèrtexs (fig. 5.4.2.2).

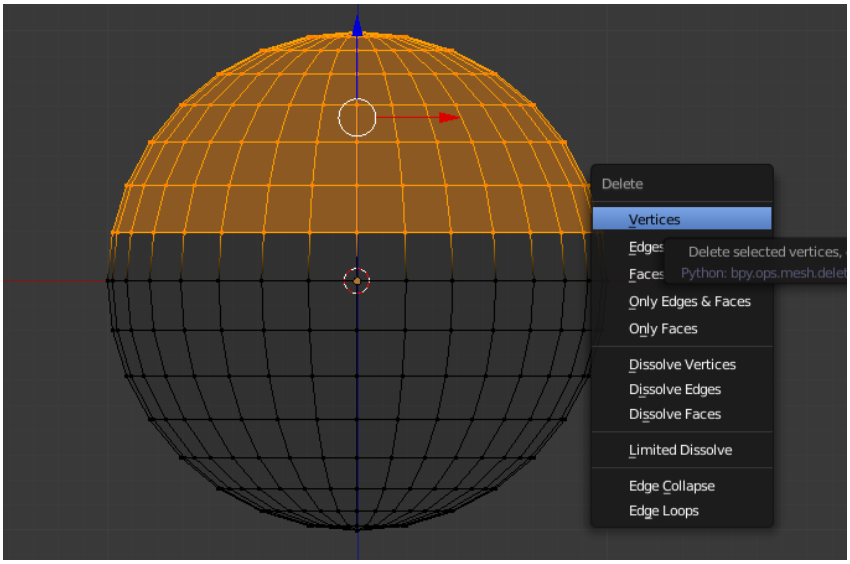

Fig. 5.4.2.2

Vèrtexs a eliminar

Es seleccionen els de baix i s'escalen per Z igual a cero (*S*, *Z*, 0) per tindre una part de baix plana i correctament mallada (fig. 5.4.2.3).

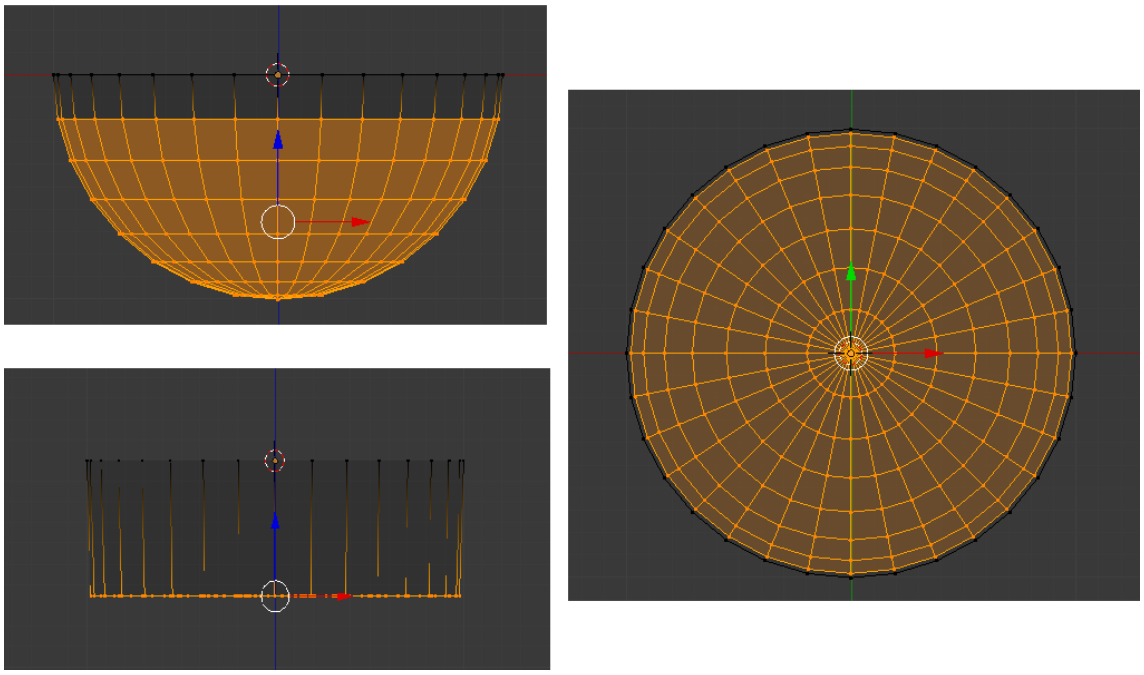

Fig. 5.4.2.3

Esfera després de transformació

Es selecciona l'anell d'eixos superior amb *Alt+Clic dret* i es meneja cap amunt (*G*, *Z*, ratolí) per der la cistella més alta. Una volta fet açò, es rota en l'eix Z 45º (*R*, *Z*, *45*). S'aconsegueix una estructura així (fig. 5.4.2.4).

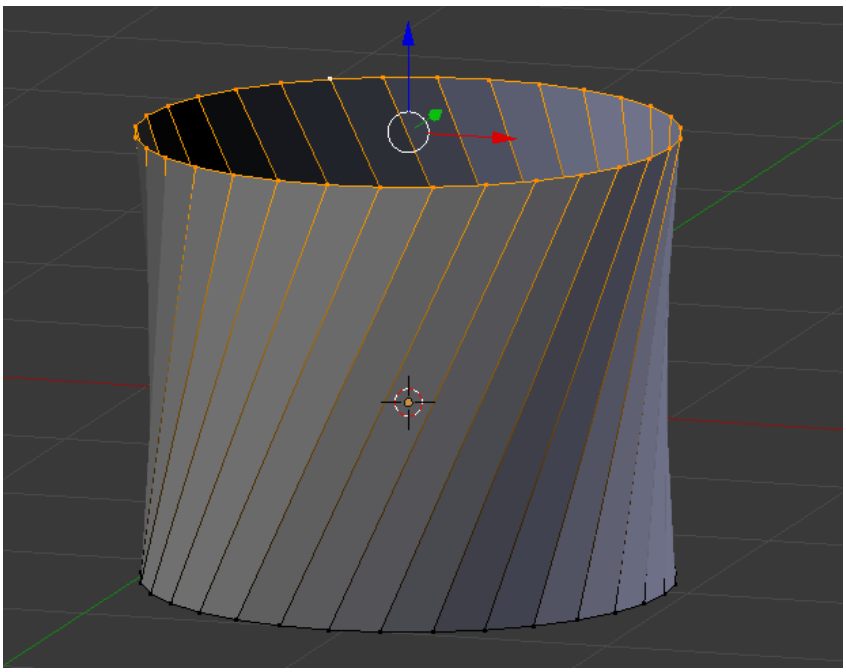

Fig. 5.4.2.4

Esfera transformada amb gir

A continuació, en *Object Mode* (TAB), es duplica l'objecte amb *Shift+D* i, de volta a *Edit Mode*, es gira l'anell superior en Z -90º (*R*, *Z*, -90) (fig. 5.4.2.5).

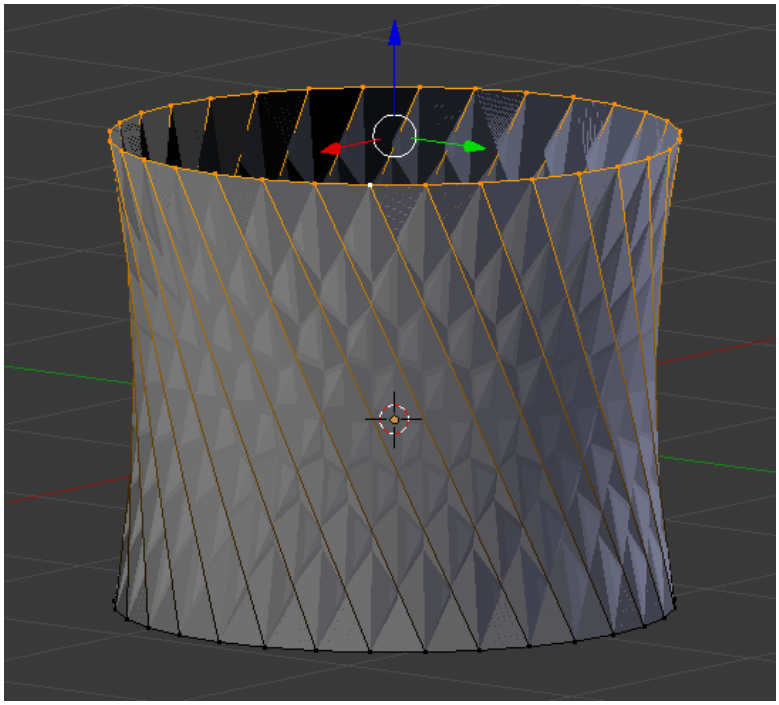

Fig. 5.4.2.5

Dos esferes girades superposades

Ara cal ficar la vista per baix, amb 7 i repetides voltes a 8 ó 2, i seleccionar els següents eixos de la cara de baix amb *Shift+Alt+Clic dret* (fig. 5.4.2.6).

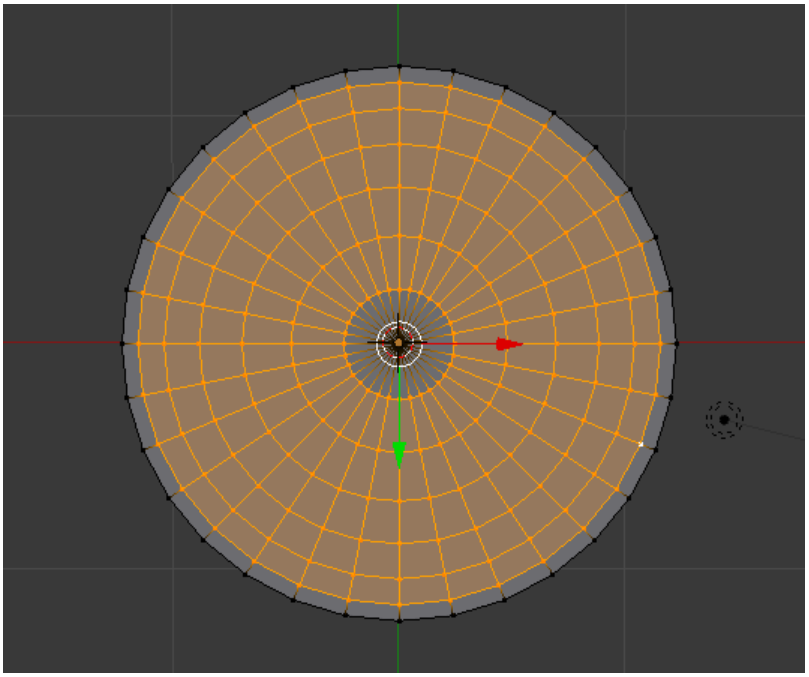

Fig. 5.4.2.6

Eixos que s'han de marcar

Després es fa una rotació en Z al gust (*R*, *Z*, ratolí) fins que quede així (fig. 5.4.2.7).

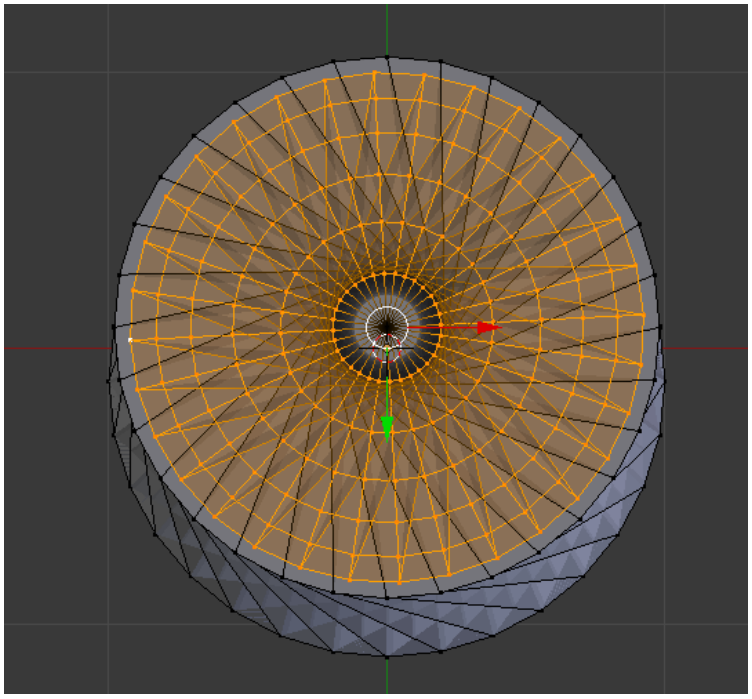

Fig. 5.4.2.7
Eixos rotats en Z

Es torna a l'*Object Mode* (TAB) i es selecciona l'objecte. De nou a l'*Edit Mode* es da clic en A per seleccionar tots els eixos del cos. Es fa clic en SPACE per obrir el buscador de Blender, es tecleja *wireframe* i es dona a ENTER (fig. 5.4.2.8).

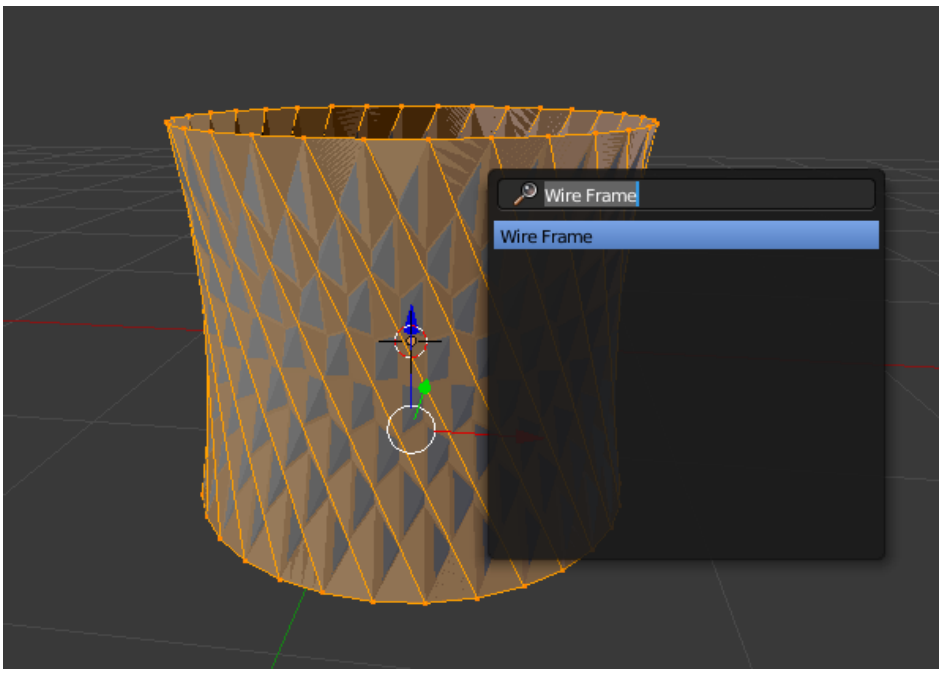

Fig. 5.4.2.8
Fer Wire Frame a l'Edit Mode

Aquesta funcionalitat recentment afegida és de molta utilitat per casos com aquest. S'observa que l'objecte s'ha convertit en un conjunt de fils que es corresponen en els eixos que eren visibles a l'*Edit Mode*. Es torna a fer el mateix procés en l'altre objecte (fig. 5.4.2.9).

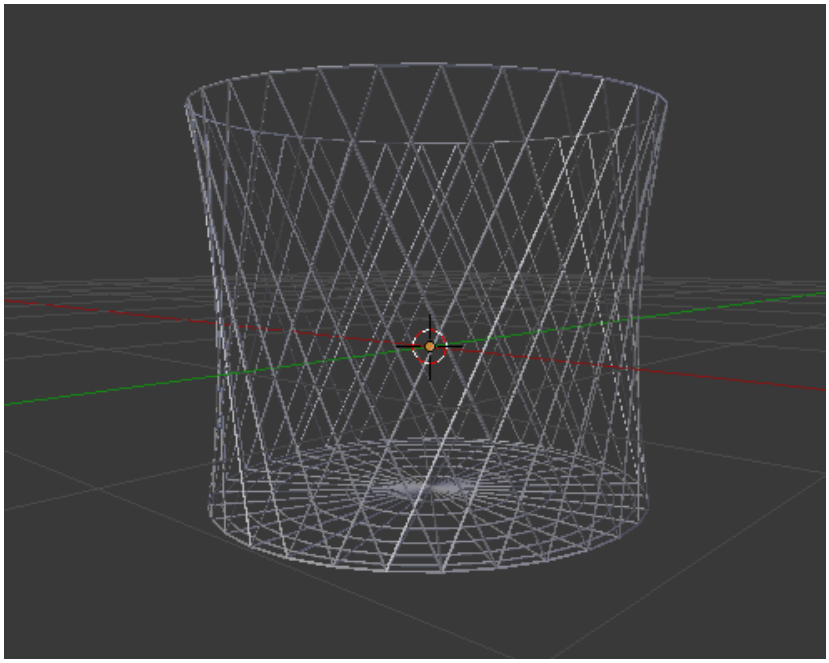

Fig. 5.4.2.9

Cos de la cistella

Posteriorment es col·loca un toroide que farà de flotador a la part superior. Tindrà un radi màxim de 0,95 i un radi mínim de 0,04. Es fica dalt amb *G*, *Z* i moviment de ratolí. Des de *Transform –> Scale* es pot variar la seua grandària (fig. 5.4.2.10).

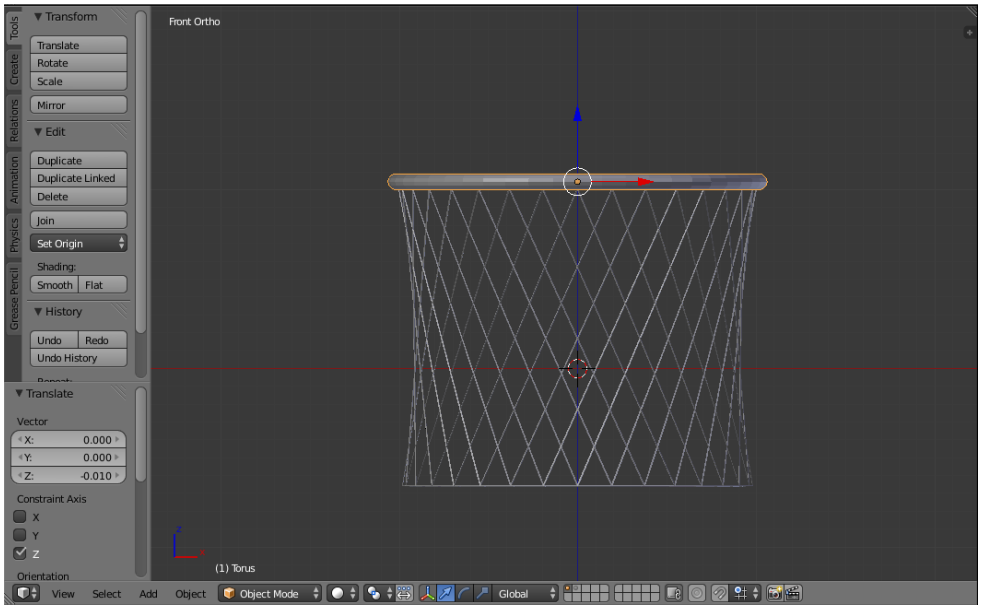

Fig. 5.4.2.10

Toroide superior introduït

Després es farà un procediment similar al de les esferes. S'introduirà un cilindre de 50 vèrtexs i 0,3 de profunditat sobre el toroide per fer de reixa superior. Es col·loca i en Edit Mode es gira l'anell d'eixos superior 10º en Z (R, Z, 10) (fig. 5.4.2.11).

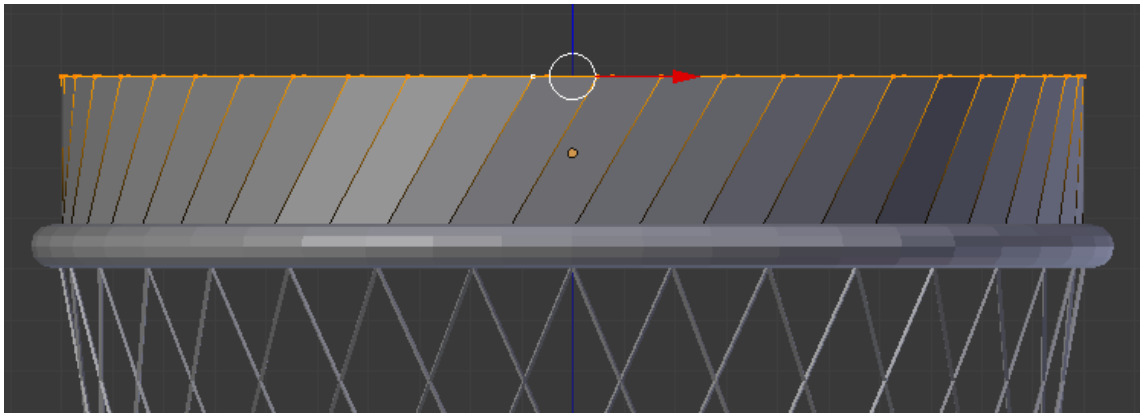

Fig. 5.4.2.11
Cilindre col·locat i girat

Es copia i es gira l'anell superior -20º en Z (*R*, *Z*, -20) i es torna a aplicar el *wireframe* (fig. 5.4.2.12).

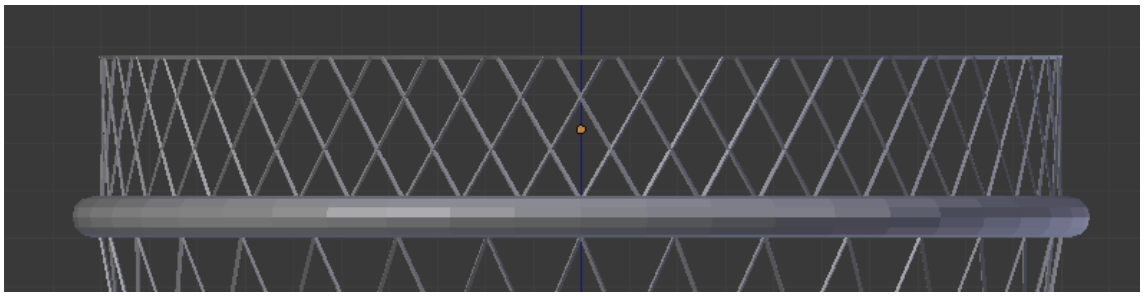

Fig. 5.4.2.12
Segona reixa completada

Per al segon flotador es requeriran columnes de suport. Es bloqueja la vista a Front Ortho (1 i 5 si cal) i es crea un cilindre de radi 0.03 i profunditat 0.3. Es desplaça el cilindre fins que quede així (fig. 5.4.2.13).

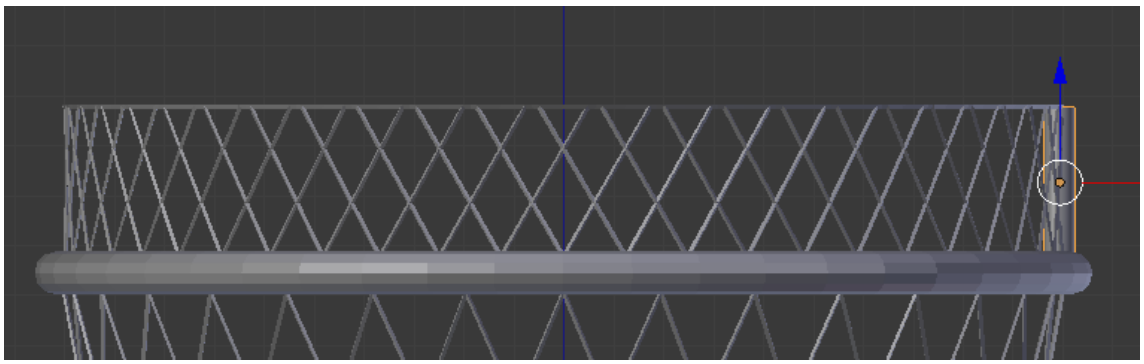

Fig. 5.4.2.13
Primer cilindre de suport

Ara es pretén repetir aquest cilindre 16 voltes al llarg del toroide. Es fa de la següent forma. Per major comoditat es fica el punt de vista *Top Ortho* (7 i 5 si cal). A més, es pot portar el cilindre a una capa buida (seleccionant-lo en *Object Mode* i fent clic en M) perquè la resta dels objectes no siga molèstia. Ara es canvia el *Pivot Point* a *3D Cursor* en lloc del *Median Point* per defecte (fig. 5.4.2.14). Açò farà que les transformacions posteriors tinguen influència segons el cursor 3D. El cursor 3D es manipula fent clic esquerre dins de l'escena. Si s'han seguit els passos segons la guia, el cursor 3D hauria de estar al centre de l'escenari. Si no fora així, amb *Shift+C* torna a la seua posició inicial i el punt de vista canvia per oferir un pla amb tots els objectes de l'escena.

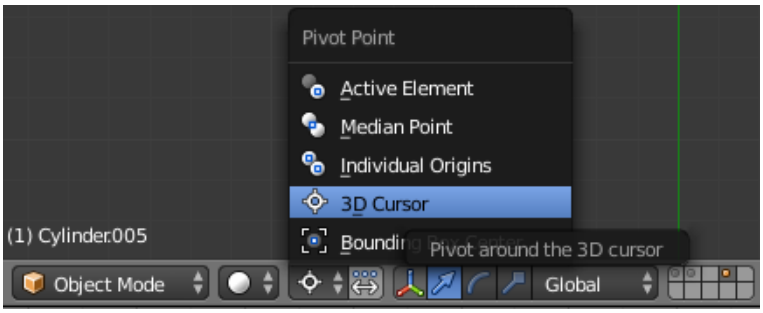

Fig. 5.4.2.14

Canvi de Pivot Point

Una volta clares aquestes consideracions prèvies, es procedeix amb les instruccions. Es selecciona el cilindre i es fa *Shift+D* i a continuació *R*, *Z*, 22.5 (resultat de dividir 360º del toroide entre les 16 divisions que es pretenen). Es fan repeticions fins completar el cercle (fig. 5.4.2.15).

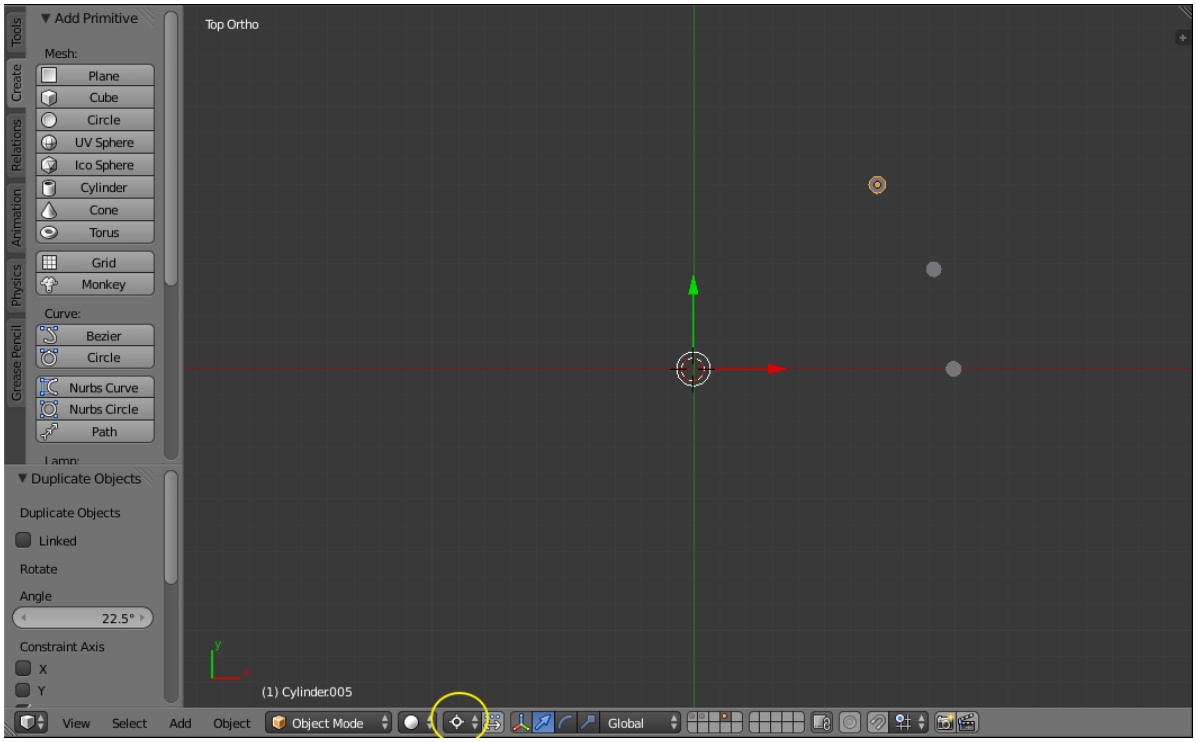

Fig. 5.4.2.15

Creació de cilindres equidistants canviat el Pivot Point

Es fica un segon toroide dalt dels cilindres de suport de iguals dimensions que el primer toroide. En *Object Mode* es selecciona el toroide, *Shift+D*, i es desplaça en Z 0,3, (*G*, *Z*, 0.3) (fig. 5.4.2.16).

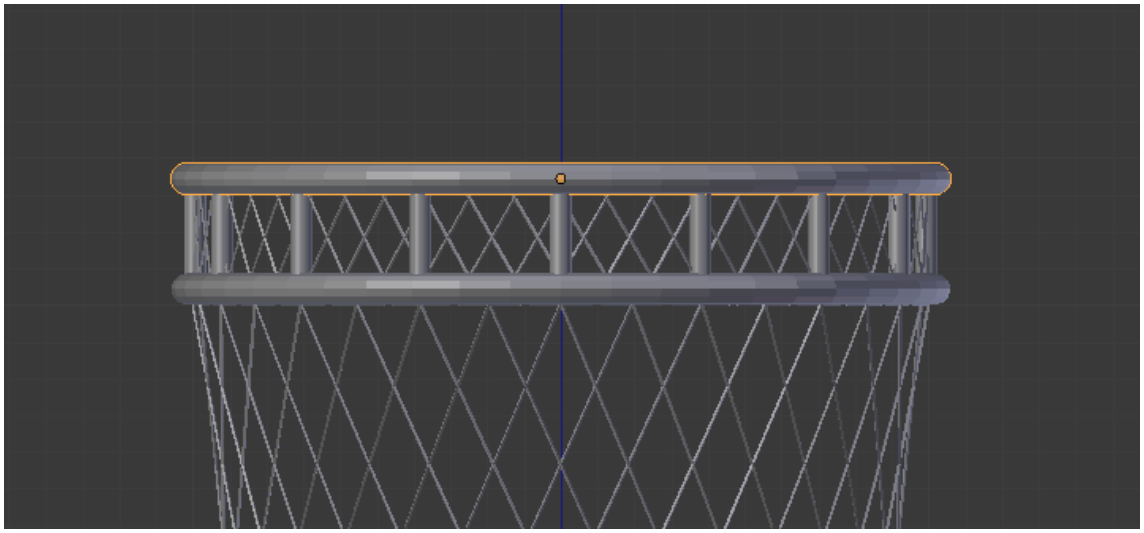

Fig. 5.4.2.16
Creació del segon toroide

Imatge en perspectiva de la cistella fins a aquest punt (fig. 5.4.2.17).

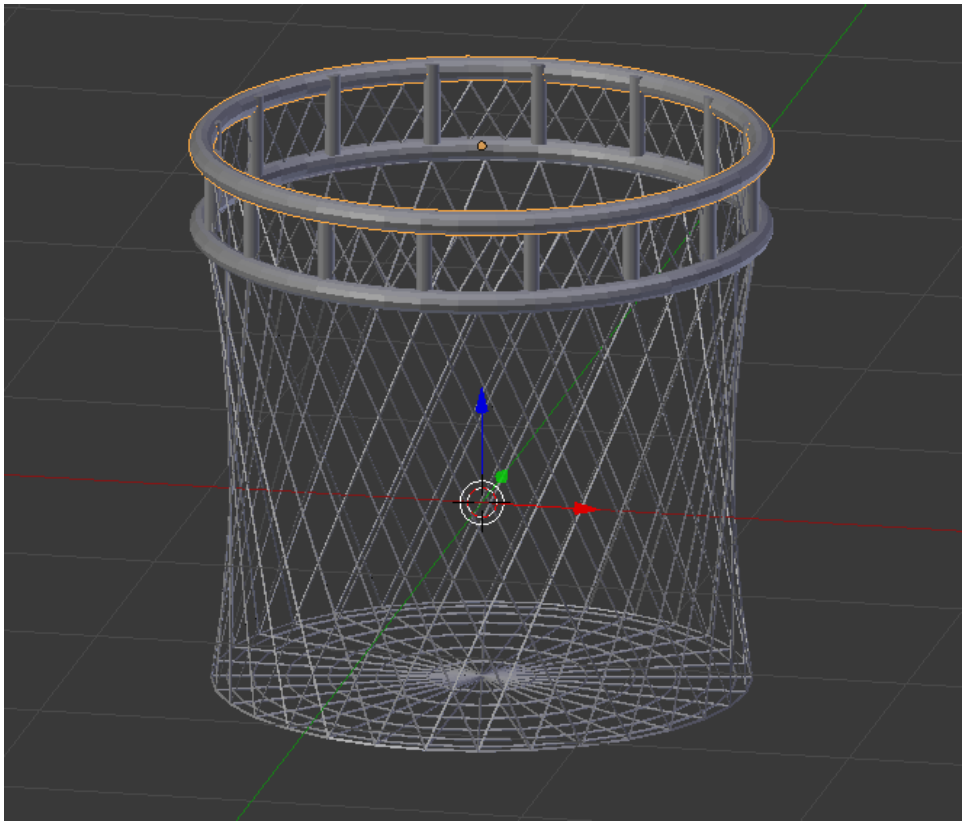

Fig. 5.4.2.17
Cistella en procés de modelització

Per fer la plataforma, es crea un plànol i es desplaça cap amunt amb G,Z fins que arribe al primer toroide (fig. 5.4.2.18).

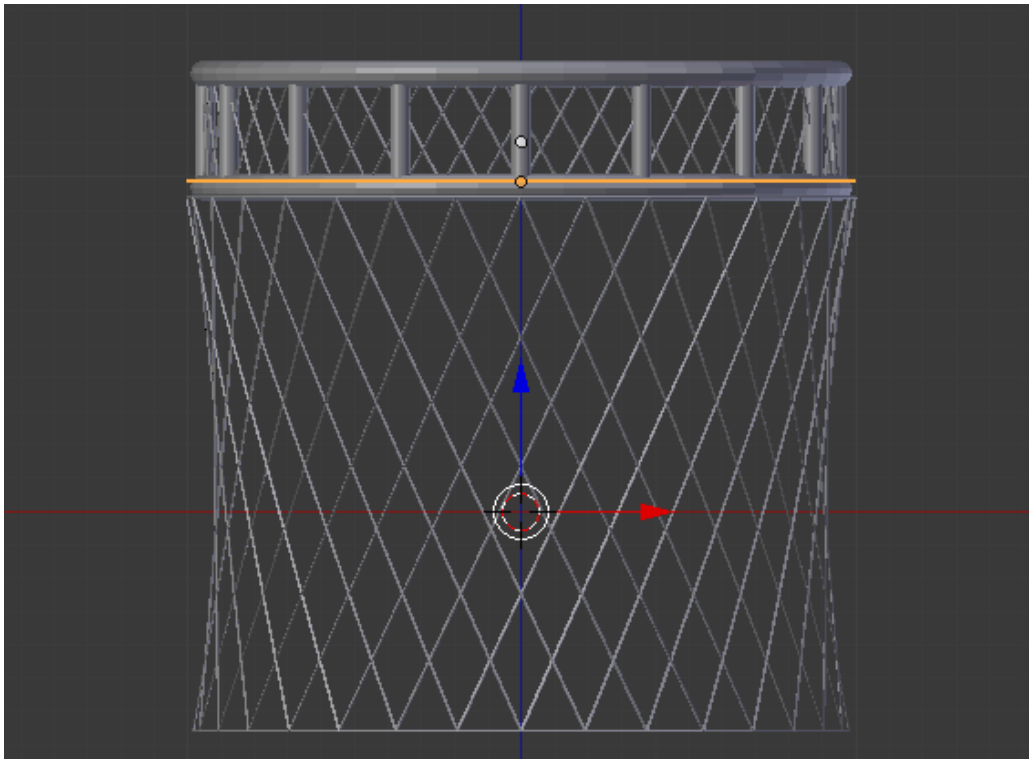

Fig. 5.4.2.18

Posicionament plànol

Es canvia el punt de vista a *Top Ortho* (7) i s'escala fins tindre una grandària suficient (fig. 5.4.2.19). A més es duplica el toroide de baix i s'escala (fig. 5.4.2.19).

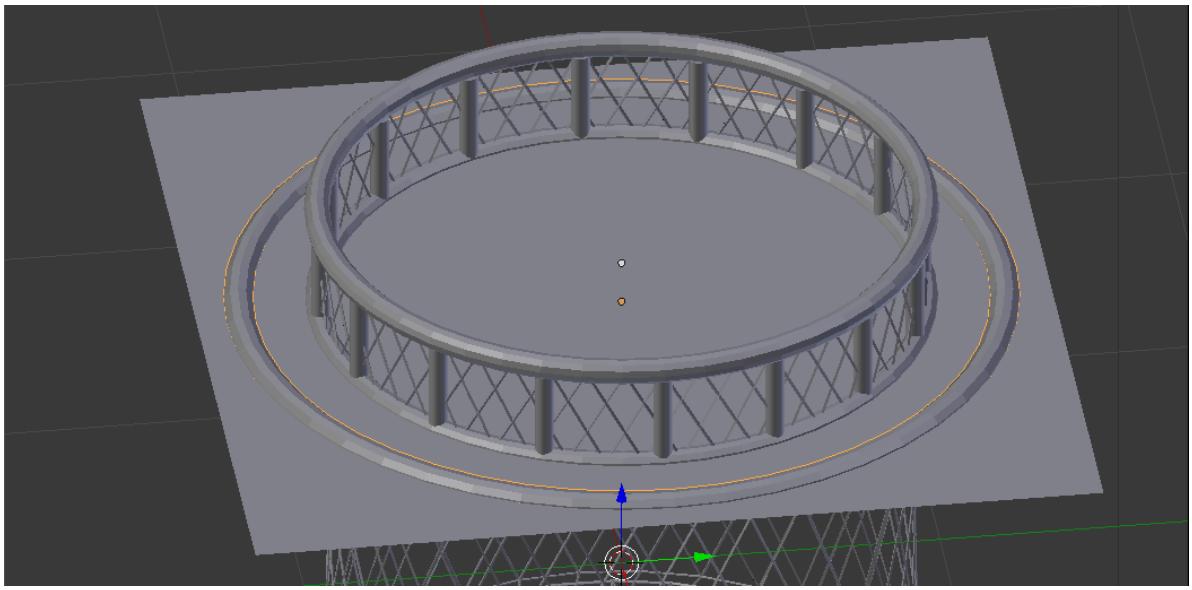

Fig. 5.4.2.19

Segon toroide i plànol escalats

A continuació es restringeix el plànol perquè quede com un anell limitat pels toroides. Es selecciona el plànol i es fa clic a *Modifiers -> Boolean* (fig. 5.4.2.20).

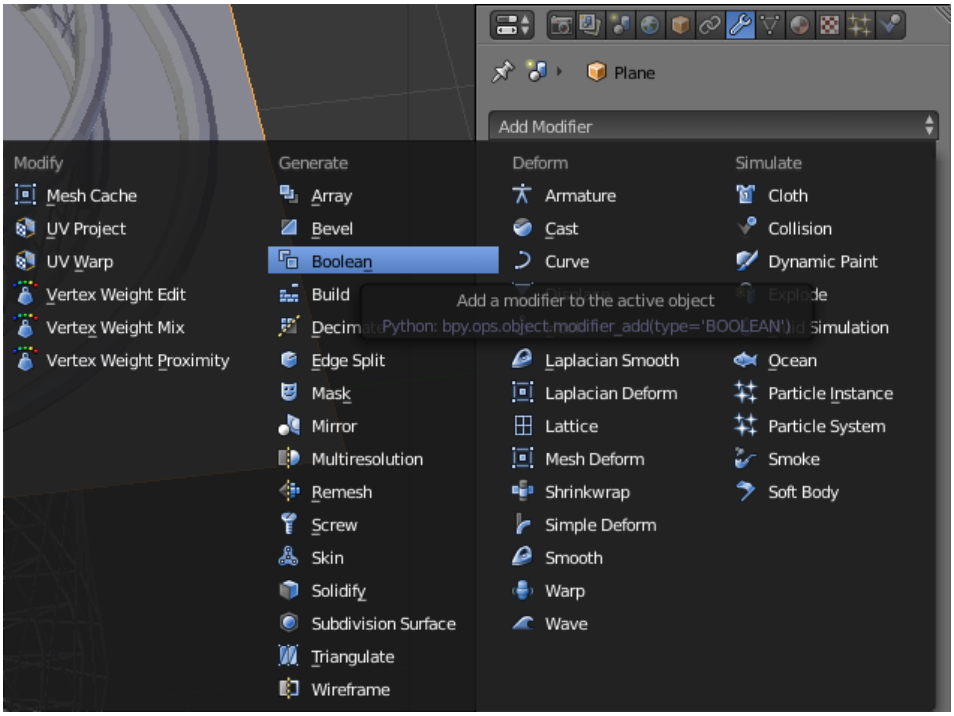

Fig. 5.4.2.20

Modificador booleà

Es selecciona *Operation -> Difference* i el objecte que actuarà com "ganivet" (fig. 5.4.2.21).

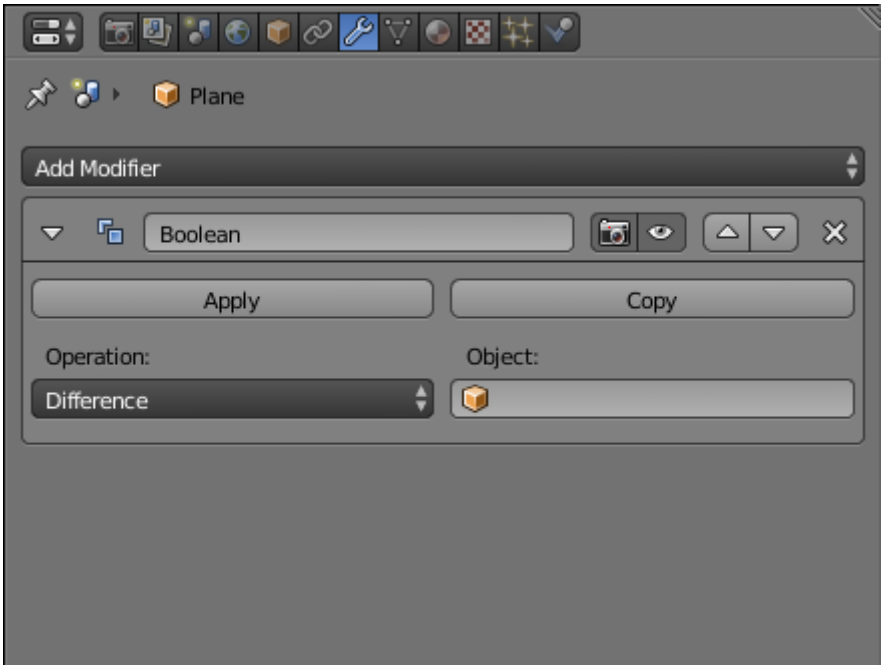

Fig. 5.4.2.21

Operador diferència del modificador booleà

Repetint la operació la plataforma quedarà així (fig. 5.4.2.22).

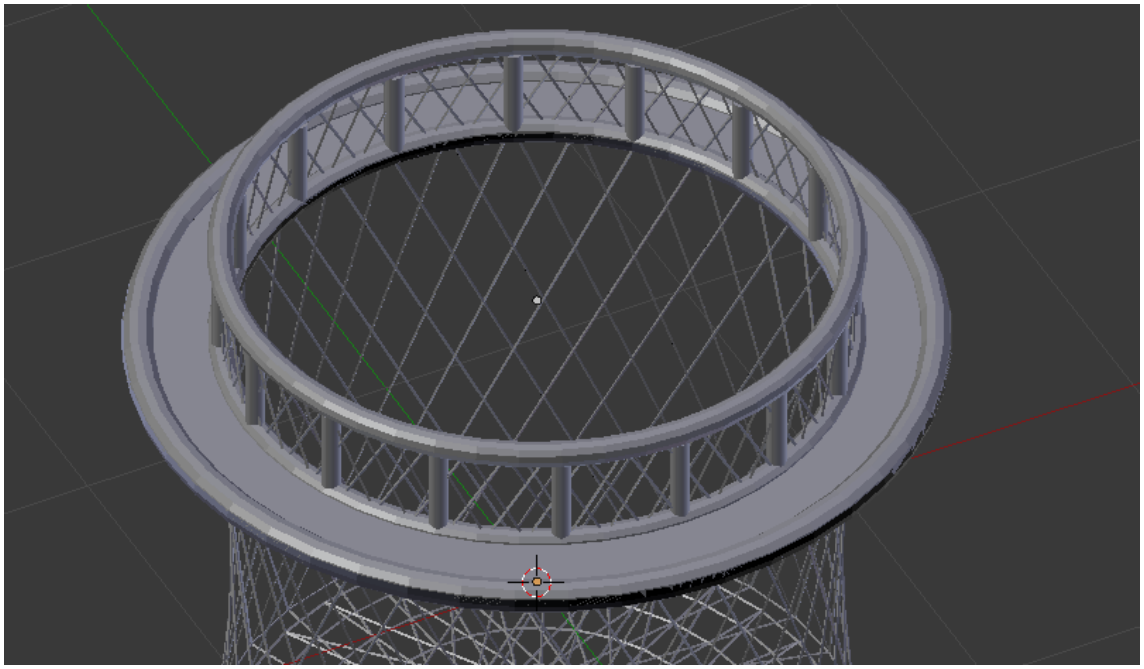

Fig. 5.4.2.22
Plataforma acabada

El que sembla estar bé potser no ho estiga. El pla retallat ha passat de ser un pla a una forma geomètrica no desitjada que pot distorsionar el resultat a l'hora de aplicar modificadors, textures, etc. Així doncs, es selecciona el plànol i es mou amb *M* a una altra capa buida (fig. 5.4.2.23).

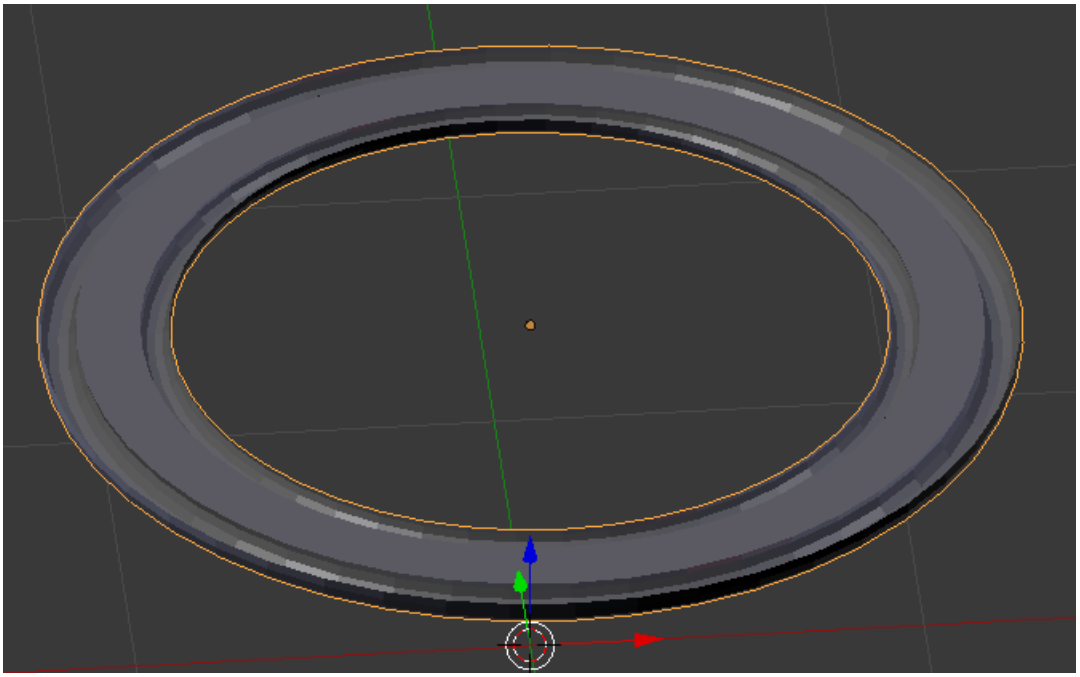

Fig. 5.4.2.23
"Plànol" a canviar

Es passa e Edit Mode i s'observen distintes cares, en aquest cas les de la figura (fig. 5.4.2.24).

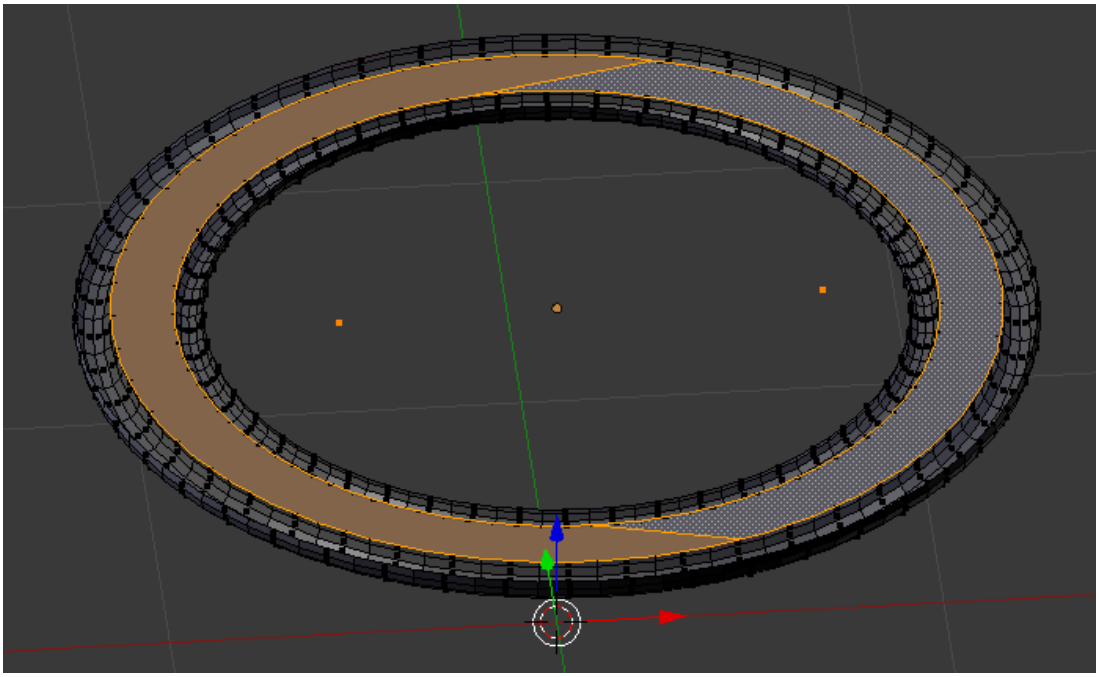

Fig. 5.4.2.24

"Plànol" en Edit Mode

Es seleccionen les dos cares d'interès, es fa clic en *P -> Selection* per desvincular-les. Ara, en *Object Mode*, ja es poden seleccionar els dos objectes per separat (fig. 5.4.2.25).

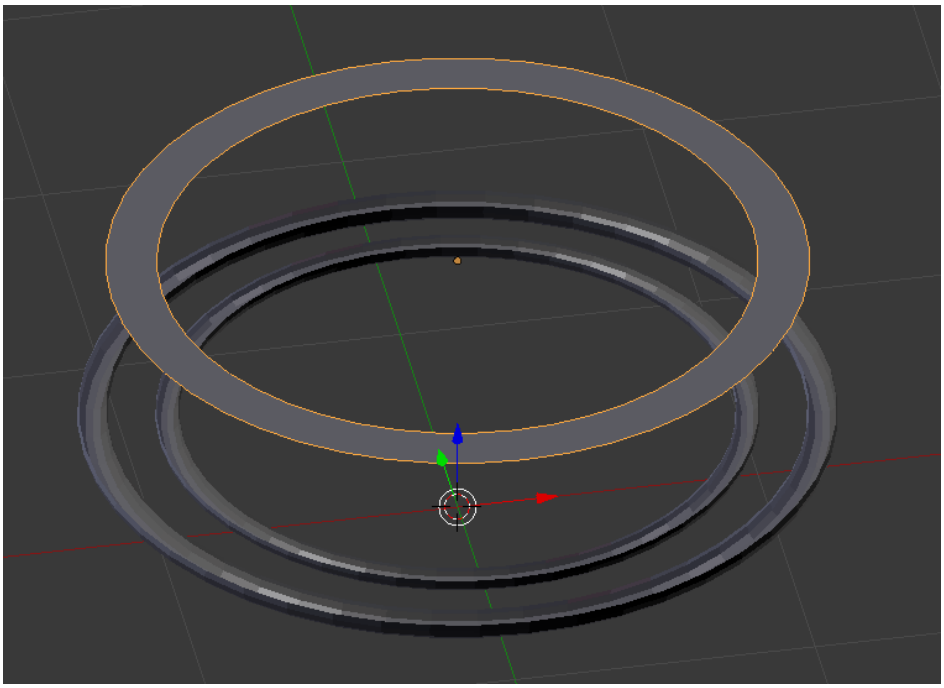

Fig. 5.4.2.25

Plànol de interès separat de la resta de l'objecte inicial

Una volta completada la modelització es passa a la fase dels acabats. Seguint els passos descrits a l'apartat 5.3 a partir de la pàgina 30, es descarreguen materials del repositori lliure de Blender (enllaç 5.4.2.26) i es segueix la guia (fig. 5.4.2.26).

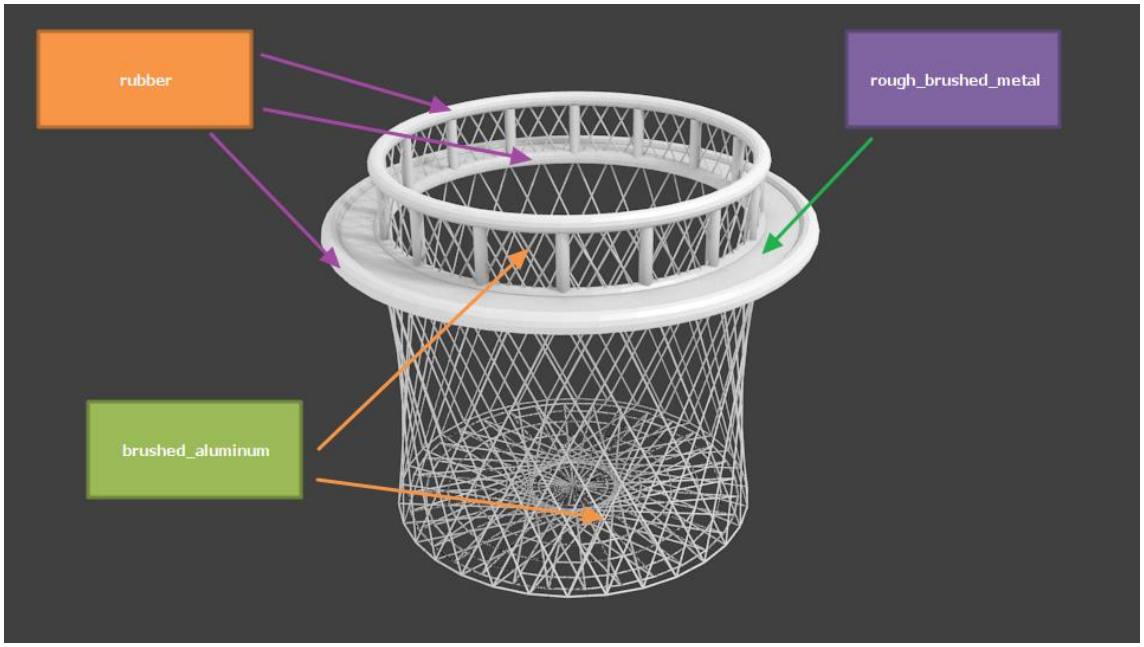

Fig. 5.4.2.26
Render amb anotacions dels materials emprats

Per últim, jugant amb els modificadors de subdivisió i suavitzant ombres es pot aconseguir un resultat final com aquest (fig. 5.4.2.27).

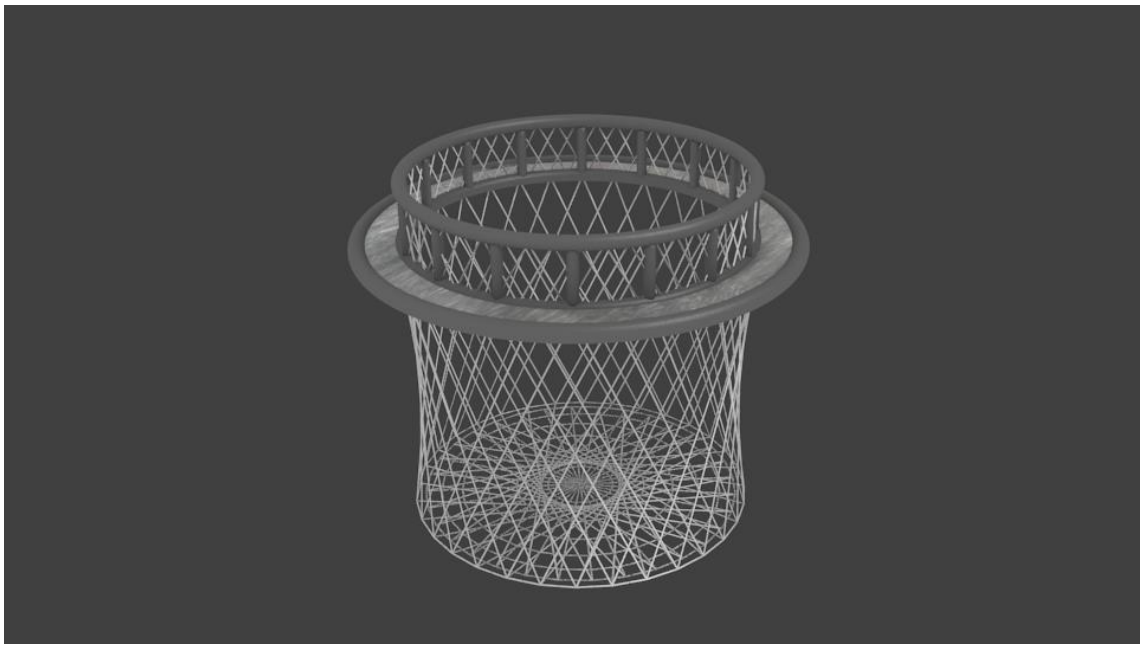

Fig. 5.4.2.27
Cistella de piscifactoria

5.5 osgOcean funcionant

Una volta completats tots el passos anteriors es hora de fer funcionar el motors gràfics. A més, s'inclouran els models tridimensionals que ací s'han especificat.

Primerament, des del terminal, cal apuntar al directori arrel d'*osgOcean*, en el que ja s'havia creat la carpeta *build-osgocean*. Es crearà una nova, anomenada per exemple *build-osgocean*, i es seguiran els següents passos:

```
cd <directori arrel osgOcean>
```

```
cd build-osgocean
```

En aquest directori es fa:

```
cd bin
```

```
./oceanExample
```

A pantalla s'hauria d'obrir una nova finestra amb *osgOcean* funcionant (fig. 5.5.1).

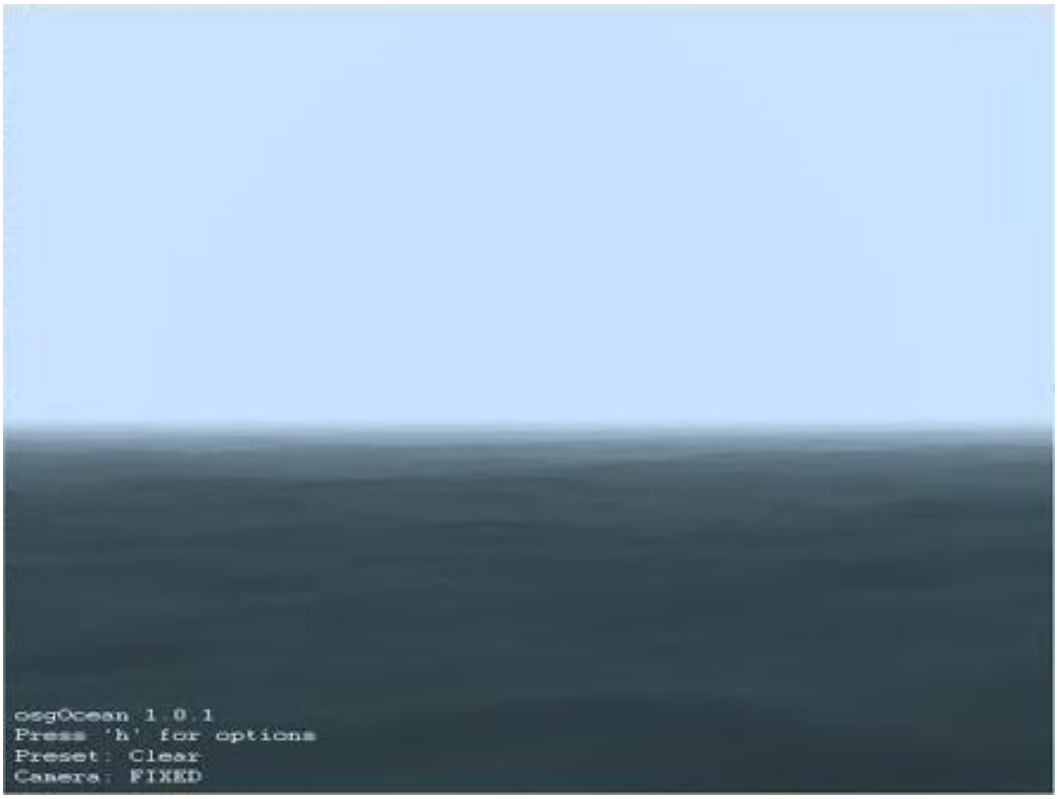

Fig. 5.5.1
osgOcean funcionant

Amb el flag --help s'obrin al terminal les instruccions d'execució que es poden donar a *osgOcean* per obtindre resultats diferents. És possible que hi haja errors o que el motor no es veja apropiadament. Per açò es recomana executar *oceanExample* així:

```
./oceanExample --disableShaders --testCollision
```

Hauria d'aparèixer per defecte un vaixell a oceanExample. La manera de fer-lo aparèixer es anar al directori .../osgOcean/src/oceanExample i obrir el fitxer application.cpp. Apareixerà, a la línia 819, un codi tal que així:

```
if (testCollision)
{
        osgDB::Registry::instance()->getDataFilePathList().push_back("resources/boat");
        const std::string filename = "boat.3ds";
        osg::ref_ptr<osg::Node> boat = osgDB::readNodeFile(filename);
        .
        .
        .
```

Cal fer les següents modificacions perquè quede així:

```
if (testCollision)
{
        osgDB::Registry::instance()->getDataFilePathList().push_back("resources/boat");
        //const std::string filename = "boat.3ds";
        const std::string filename = "/usr/local/bin/resources/boat/boat.3ds";
        osg::ref_ptr<osg::Node> boat = osgDB::readNodeFile(filename);
        .
        .
        .
```

Ja que *oceanExample* no trobava l'arxiu boat.3ds cal buscar-ho (si no ha passat res estrany hauria de ser al directori inclòs en el codi) i introduir-ho, sense oblidar comentar la línia anterior. Es guarda l'arxiu i es torna, des del terminal, a fer:

```
cd <directori arrel osgOcean>
cd build-osgocean
make
```

Aquest procés no hauria de tardar massa. Es fan un parell de instruccions, entre les quals està la de la línia que s'acaba de modificar. Així doncs, es fa:

```
cd bin

./oceanExample --disableShaders –testCollision
```

Si tot ha anat bé, hauria de poder veure's el vaixell a l'escena (fig. 5.5.2).

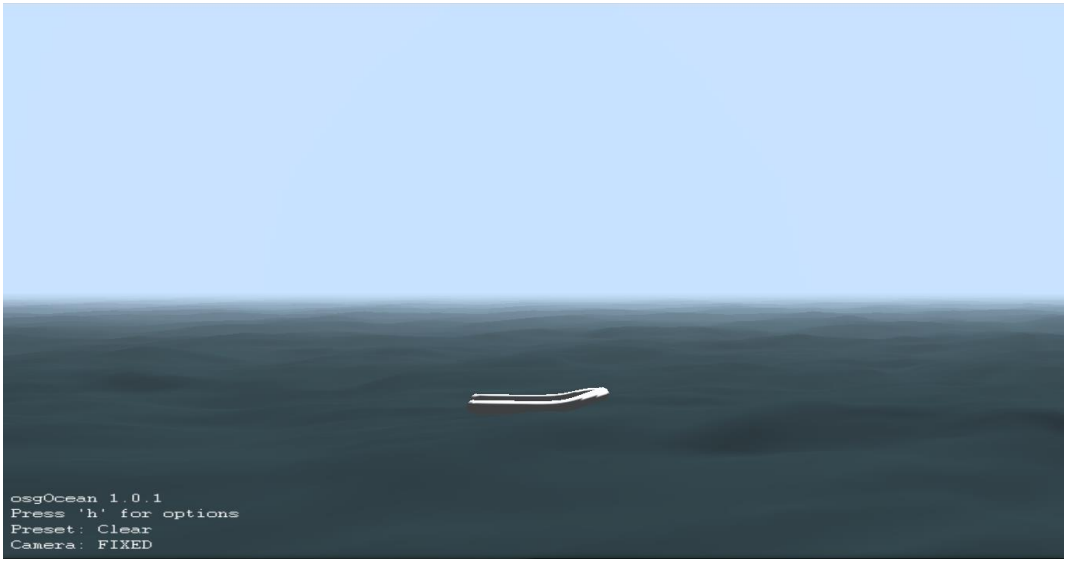

Fig. 5.5.2

oceanExample amb vaixell

Coneixent què modificar, no hi ha cap impediment en carregar a l'escena els models que s'han creat anteriorment. S'obri el model en Blender i es fa clic en *File -> Export -> 3D Studio (.3ds)* (fig. 5.5.3).

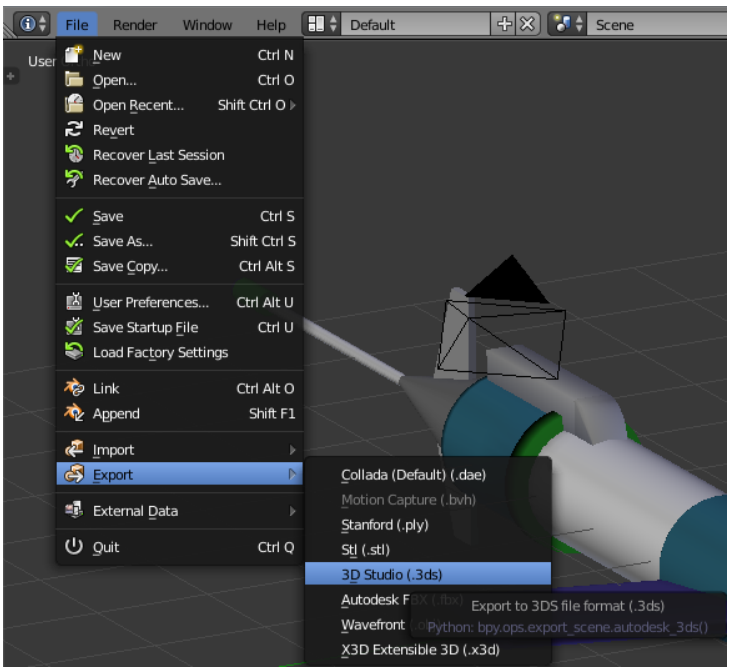

Fig. 5.5.3

Convertir arxiu .blend a .3ds

Una volta guardat el fitxer, es torna a obrir l'arxiu application.cpp i es canvia la línia abans introduïda per:

```
if (testCollision)

{

        osgDB::Registry::instance()->getDataFilePathList().push_back("resources/boat");

        //const std::string filename = "boat.3ds";

        //const std::string filename = "/usr/local/bin/resources/boat/boat.3ds";

        const std::string filename = "direcció al glider.3ds";

        osg::ref_ptr<osg::Node> boat = osgDB::readNodeFile(filename);

        .

        .

        .
```

Es tornen a fer els passos anteriors i, al iniciar *oceanExample*, hauria d'aparèixer el model carregat, en aquest cas el glider (fig. 5.5.4).

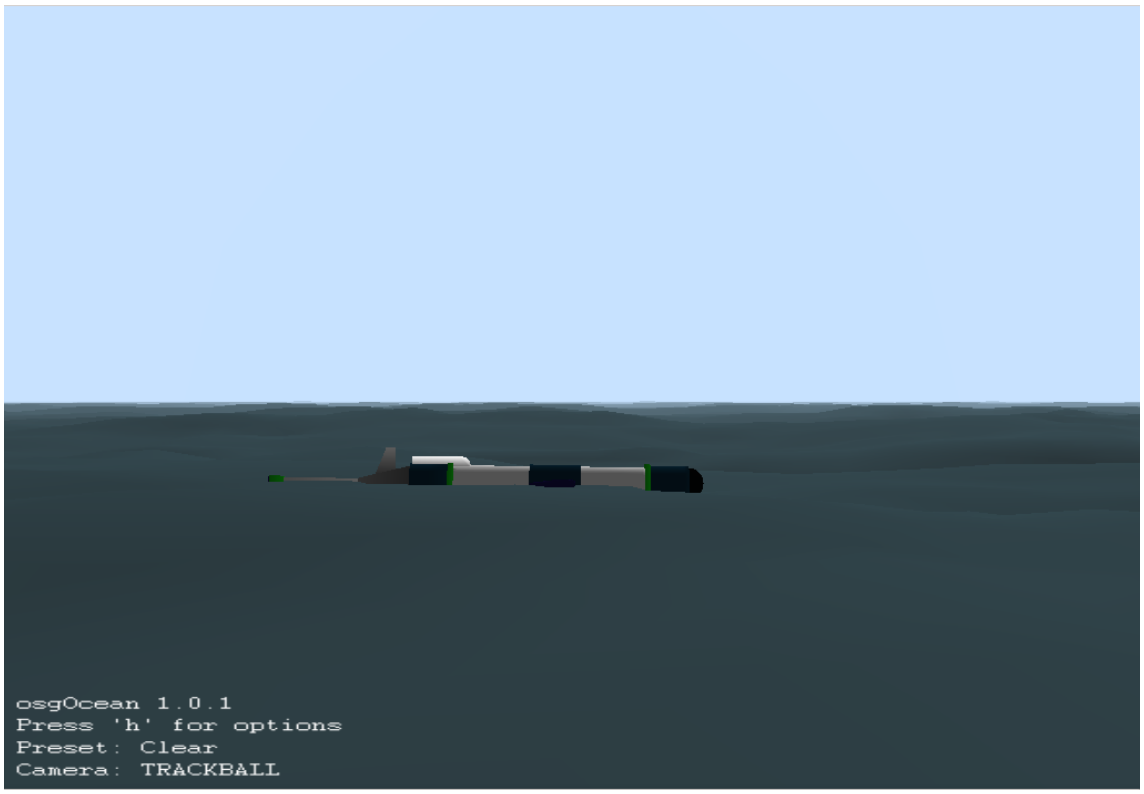

Fig. 5.5.4

oceanExample amb el glider

6 Conclusions

El present projecte va nàixer al tindre consciència de la relació humana amb la informació. A mesura que avança el temps aquesta exposició és major, la connectivitat és fàcil i, a voltes, es fa quasi obligatòria.

A les aplicacions d'enginyeria relacionades amb els camps de la electrònica, la automàtica i la informàtica sempre es pot reduir un únic concepte. Hi ha una informació que es pretén rebre (sensors), tractar (sistemes informàtics), i en alguns casos actuar en conseqüència (actuadors). Tenint aquesta filosofia clara, és evident que més abans que tard hi haurà de aparèixer una comunicació realitat-humà, la informació captada pels dispositius dissenyats ha de ser transmesa a l'individu d'alguna forma. Es ací quan apareixen les interfícies, les cares que el software fica a disposició dels usuaris per recopilar informació, observar cóm es modifica o, fins i tot, fer manipulacions en ella.

Aquest apartat pot arribar a ser de vital importància. Hi han multitud de processos, com el pilotatge d'avions, trens, controls de sistemes delicats, El maneig dels quals ha de ser intuïtiu, fàcil i ràpid. En aquest treball s'ha elegit un medi en particular, l'aquàtic, i s'ha fet ús de múltiples ferramentes per tractar de crear una realitat simulada que tinga a veure en la realitat "real". Un exercici de connexió entre les xifres i la informació pura donada segons distintes interfícies i un entorn gràfic tridimensional, net i comprensible per qualsevol persona.

Per últim, recordar la importància del software lliure al desenvolupament actual. Que aquest treball siga un exemple dels magnífics resultats que es poden arribar a aconseguir mitjançant l'esforç continuat i constant de milers de persones anònimes a les que s'anomena comunitat. Gràcies a ella, els límits de l'aprenentatge semblen no tindre fi.

7 Bibliografia

[1] Wikipedia, List of 3D modeling software

Enllaç: http://en.wikipedia.org/wiki/List_of_3D_modeling_software

[2] OGRE 3D

Enllaç: http://www.ogre3d.org/

[3] OpenSceneGraph (OSG)

Enllaç: http://www.openscenegraph.org/

[4] AutoDesk 3DS MAX

Enllaç: http://www.autodesk.es/products/3ds-max/overview

[5] SketchUp

Enllaç: http://www.sketchup.com/es

[6] Intituto Nacional de Tecnologías Educativas y de Formación del Profesorado, Blender: 3D en la educación

Enllaç: http://www.ite.educacion.es/formacion/materiales/181/cd/indice.htm

[7] Wikipedia, Glider Submarino

Enllaç: http://es.wikipedia.org/wiki/Glider_Submarino#cite_note-2

[8] Wikipedia, Autonomous Underwater Vehicle

Enllaç: http://en.wikipedia.org/wiki/Autonomous_underwater_vehicle

[9] Wikipedia, Piscicultura

Enllaç: http://es.wikipedia.org/wiki/Piscicultura

[10] Community Help Wiki (Ubuntu), What are Repositories?

Enllaç: https://help.ubuntu.com/community/Repositories/Ubuntu

[11] Blender Materials, The Open Material Repository

Enllaç: http://matrep.parastudios.de/index.php

[12] KUbuntu 12.04, Descàrrega de software

Enllaç: http://cdimage.ubuntu.com/kubuntu/releases/12.04/release/

[13] OpenSceneGraph 3.0.1, Descàrrega de software

Enllaç: http://trac.openscenegraph.org/projects/osg//wiki/Downloads

[14] osgOcean, Descàrrega de software

Enllaç: https://code.google.com/p/osgocean/downloads/list

[15] FFTSS, Descàrrega de software

Enllaç: http://www.ssisc.org/fftss/

[16] Blender 2.71, Descàrrega de software

Enllaç: http://www.blender.org/download/

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Pressupost del TFG

Ús de ferramentes de codi obert per la representació tridimensional del medi aquàtic

Alumne: Xavier Esteve Melià

Tutor: Àngel F. Perles Ivars

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

A causa de la pròpia filosofia d'aquest projecte, els gastos del mateix són mínims, i es basen en un hardware suficient com per a fer front a les ferramentes de simulació emprades, connexió a internet i els costos de personal.

Concepte	Cost per hores $\left(\frac{€}{h}\right)$	Hores (h)	Cost total (€)
Hardware	-	-	500
Personal	30	300	9000
		Total	9500

Les hores dedicades poden desglossar-se de la següent manera:

Activitat	Cost per hores $\left(\frac{€}{h}\right)$	Hores (h)	Cost total (€)
Recerca d'informació	30	50	1500
Adequació i muntatge del software	30	100	3000
Modelització 3D	30	100	3000
Redacció	30	50	1500
		Total	9000