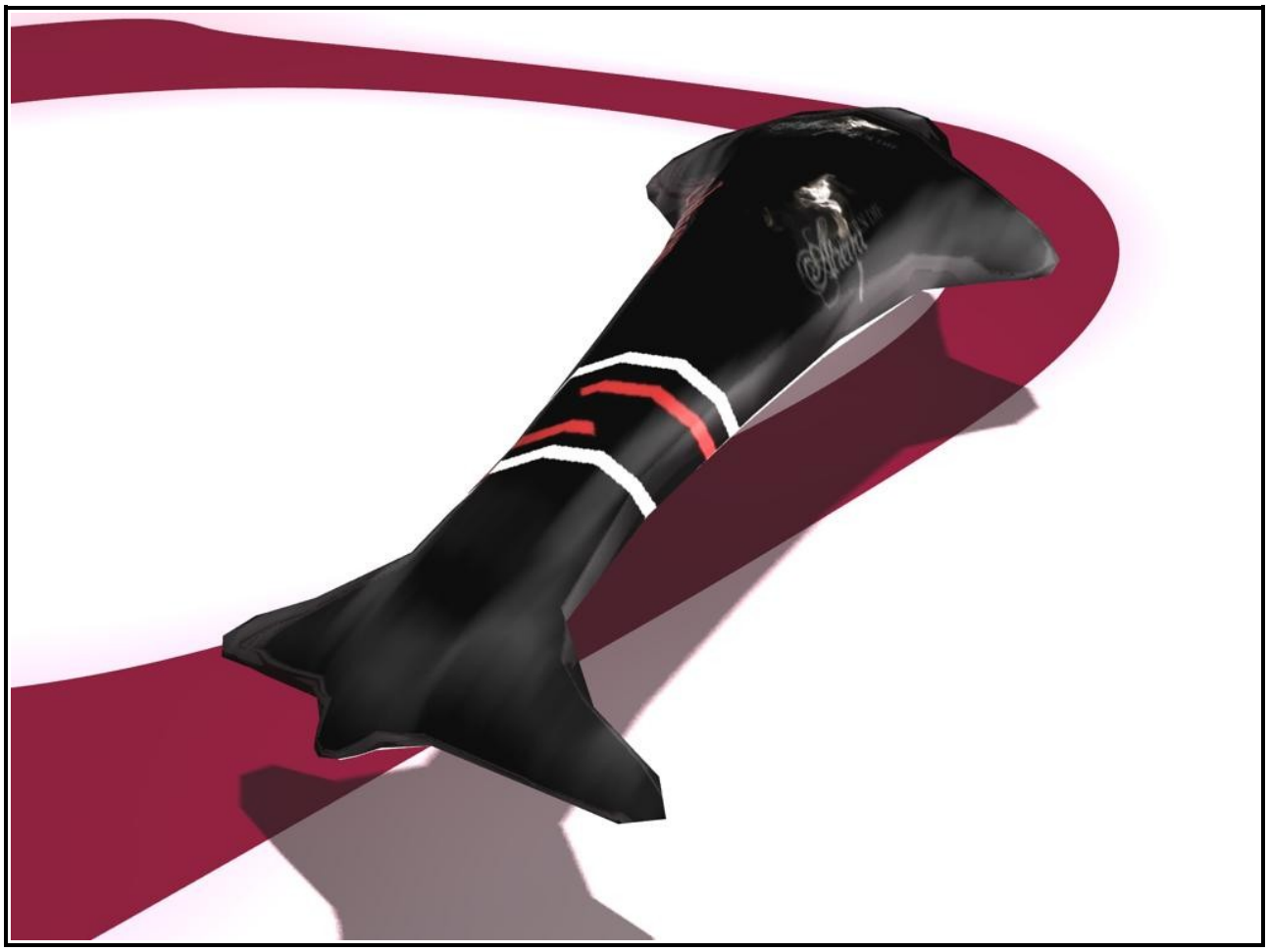
Torque Tutorial #2

Modifying the existing 'Starter Racing Kit' to 'Hovercraft Racing Game'
Tools: Softimage|XSI, Blender3D, GIMP, Terragen



By: Iwan Kartiko, 29/03/06

Table of Contents

Section 1, 2, 3, 4, 5 and 6 go together.
Section 7, 8, 9 and 10 are different topics and they are more like short-individual guides in creating more checkpoints, importing a heightmap, texturing terrain and setting up a skybox.

Introduction

Welcome to another Hovercraft Racing Tutorial. This second tutorial is quite similar to the first tutorial. In this second tutorial we will be using Softimage and Blender3D to create model for Torque Game Engine. This tutorial will look into modeling a simple hovercraft, adding checkpoints and creating a game world by utilising Terragen and Torque's built-in editor.

This tutorial was written based on Torque Game Engine v1.4, most of the steps explained here can be done also for v1.3.

I won't discuss Softimage|XSI in depth here. I'm assuming that you're already familiar with XSI user interface, Selection tools (Spacebar, T, Y, U, I, and M key) and Transformation tools (X, C and V key).

For this tutorial we'll be employing these following tools;

- Torque3D Game Engine
- Softimage|XSI
- Blender + Torque Exporter plug-ins
- jEdit (programmer text editor)
- Gimp (graphics manipulation program)
- Terragen (free photorealistic scenery generator)

Torque3D Game Engine
www.garagegames.com
The cheapest commercial game engine out there with TONS!!! of features (please check the web, it's too too much to list here) and an amazingly active community. There are 2 different type of licenses; Indie and Commercial. Indie cost only USD$100; targeted for hobbyist, for learning purpose and for commercial too if you're earning less than USD$250,000 annually. Check the web site for more information on Torque Game Engine. When you purchase TGE you will be given full C++ source code as well.

Softimage|XSI
www.softimage.com
Great digital animation software and the user interface has apparent workflow and key settings that increase productivity. Softimage comes in 3 different edition; Foundation, Essential and Advanced. Foundation is the least expensive with limited yet sufficient features to get work done. Advanced is the most expensive of all and it has everything Softimage|XSI has to offer; sysflex cloth, advanced character rigging, hair/fur, behavior, advanced polygon tools, simulations, etc.
Please refer to the website for more information about Softimage products.

Blender + Torque Exporter plug-ins
www.blender.org
Blender is an open source software for 3D modeling, animation, rendering, post-production, interactive creation and playback. Available for all major operating systems under the GNU General Public License.
You may create your model in Blender too if you're familiar with it. I've been using the program for quite sometime now and still I find the interface is annoying and confusing. You can work really fast and effectively with blender if you could memorise all the hotkeys. I will provide a brief

introduction to Blender in this tutorial. We'll be using blender to prepare a model for Torque Game engine as there is currently no exporter for XSI.

jEdit

www.jedit.org

This program is free as well. In order to run jEdit we must make sure that Java Runtime is installed in the computer. You may use your own editor of your choice.

Gimp

www.gimp.org

This is another free program to manipulate pictures, similar to Adobe Photoshop with less filters. We'll use this program to create texture for 3D model. You may use Adobe Photoshop for this tutorial too if you are familiar with it.

Terragen

http://www.planetside.co.uk/terragen/

A photorealistic scene rendering engine. It allows you to create a breathtaking natural scenery with lot of options to alter the atmosphere, fog, clouds, terrain surface, water, etc. In this tutorial, we'll be using this program to create a skybox for our game.

The free edition allows you to create terrain up to 513 x 513 pixels and maximum render resolution of 1280 x 960 pixels, non commercial use. If you wish to to get an output > 2000 x 2000 pixels you need to register and pay for the commercial license.

If you have a copy of Bryce, Vue or MojoWorld, you may use them also to create a skybox.

At the end of this tutorial I also have provided a troubleshooting section, which covers common problems that might be encountered during this tutorial. So, again, hold on to your hat, and enjoy the ride.

Section 1
- Setting up your Torque3D Game Engine

You got torque now? Great! What we are doing here is to set-up Torque3D Game Engine in your computer then copy the files that we required for this tutorial somewhere-else. The reason is when you modified the original file you might want to -at certain point- restore the original file for comparing purpose or script testing purpose. Re-installing TGE many times can be troublesome. So, leave the original files alone, copy all the required files somewhere-else and start working from the copied files.

Installing Torque3D Game Engine:

1. Install TGE to your computer.

2. Get into the folder which Torque was installed, you will notice there is a 'example' folder (*illustration 1*). Copy the content of 'example' folder to your own working directory.

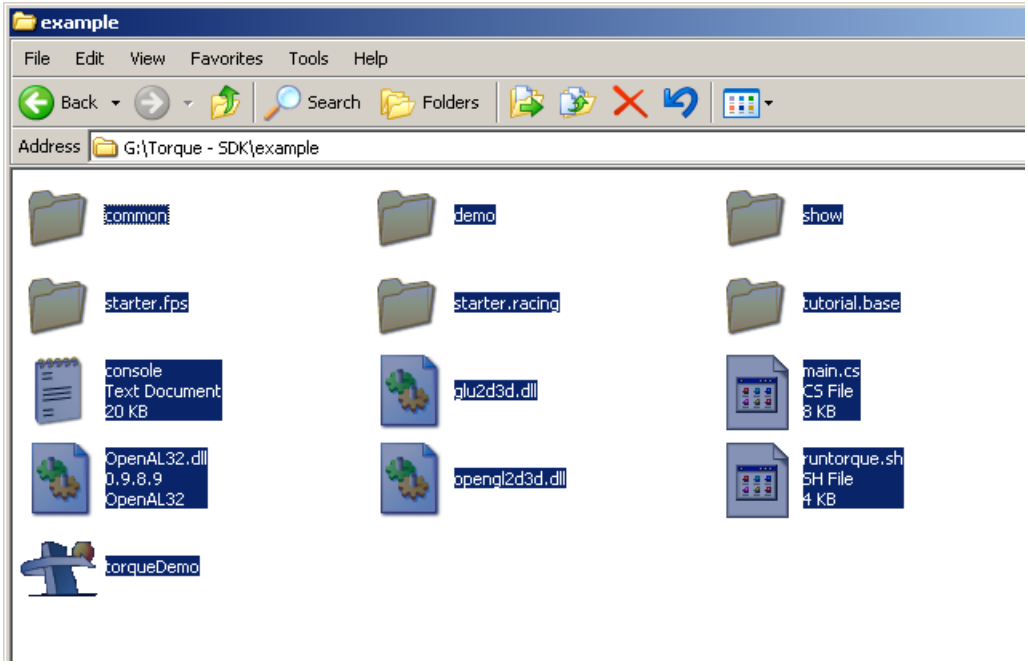


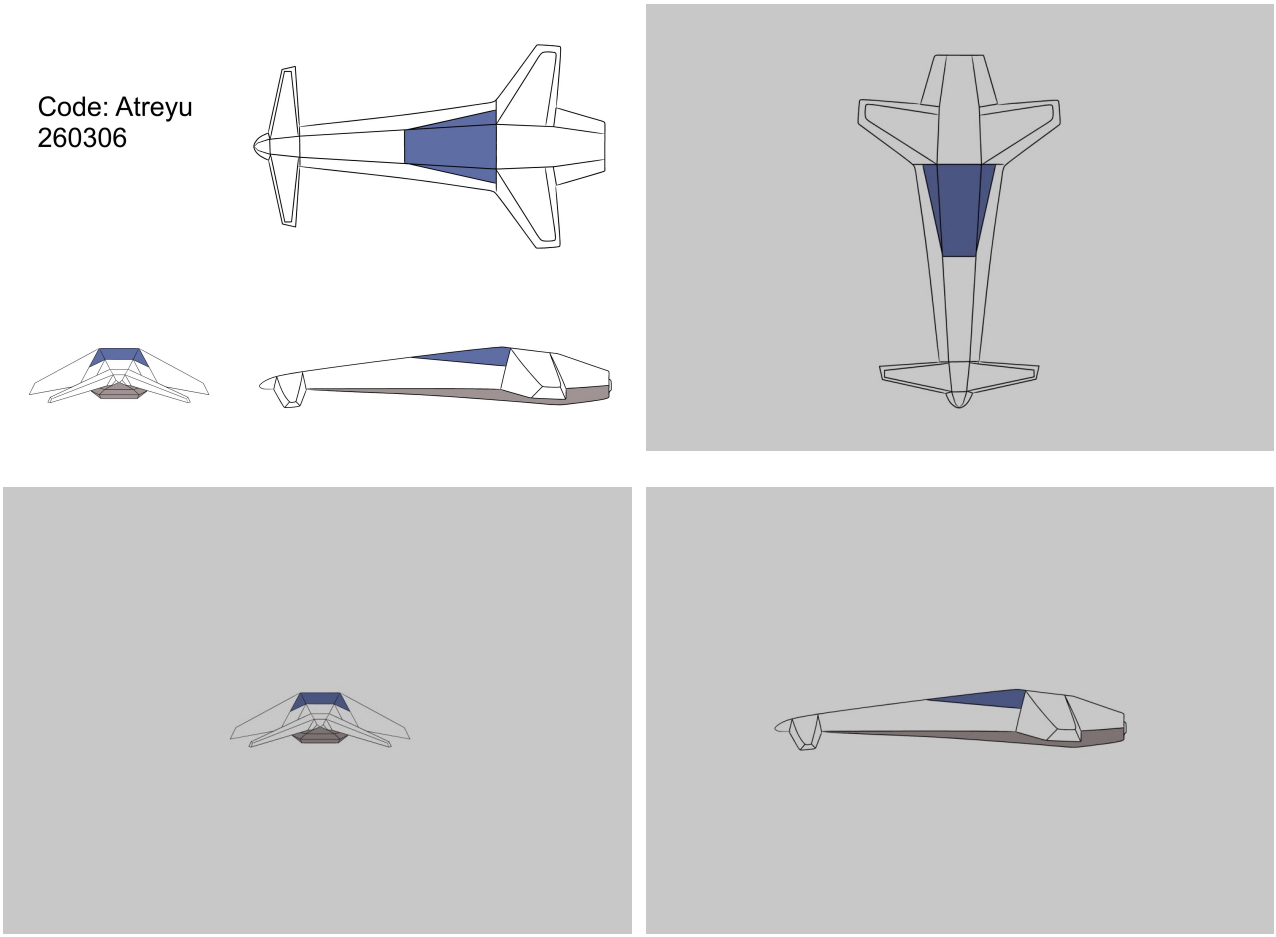
Illustration 1Contents of 'example' folder

3. Now we need to modify *main.cs* file so that torqueDemo loads *starter.racing* kit when we double click it. By default, torqueDemo would launch *Tutorial Base* (v1.4) or *starter.fps* (v1.3).

   Open up *main.cs* with jEdit or with your preferred editor, change the default game entry at the beginning of the file with *$defaultGame = "starter.racing";*

That's it, basically you have your own working folder with an executable that will automatically load Torque's Starter Racing Kit. If we ever need to get the original files, we can always get from the original installation folder without having to reinstall Torque every time. Let's move on...

Section 2
- Preparing 2D sketches



I mentioned already mentioned this the last tutorial and I'm going to mention this again. Before plunging yourself into 3D modeling world, lets do some sketches to guide your 3D modeling. When you sketch, please make sure that the model is vertically and horizontally in proportion.

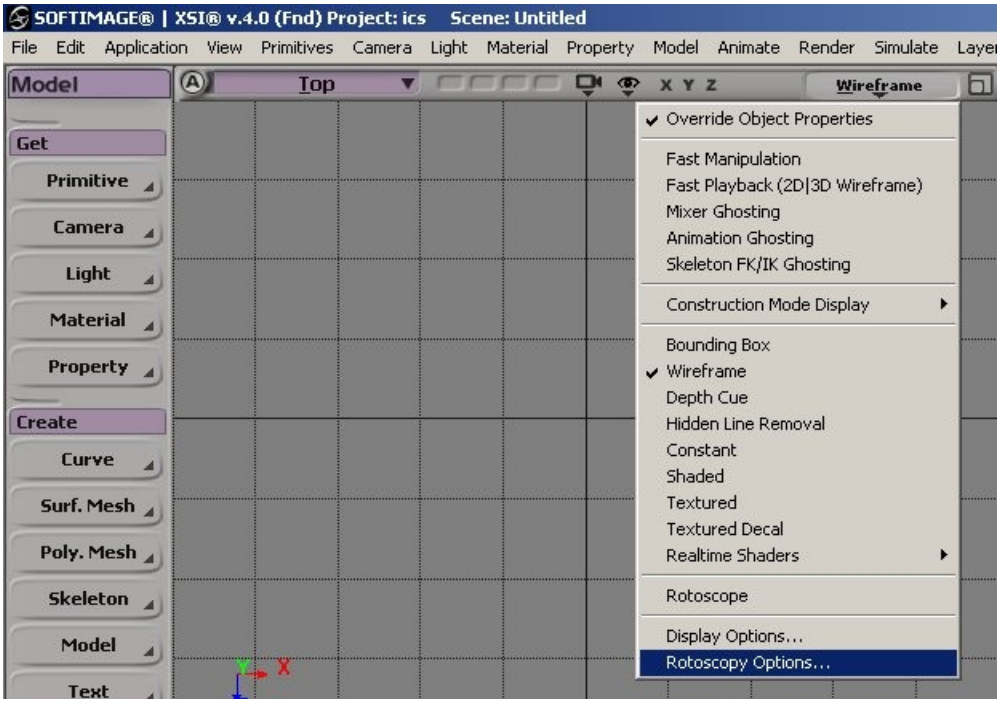
The sketches for top, side and front view were darkened because we'll use these sketches within Softimage. In Softimage, when we select a line or a polygon it will be highlighted white by default, seeing white on white is really troublesome, by having darkened sketches we can see the polygons sit on the sketch more clearly.

You may use your own hovercraft design for this tutorial. You can grab the techniques in this tutorial and apply them.

Your top view sketch might not be perfectly symmetrical, and that's all right. The sketches would still serve it's purpose to guide us in making 3d model.

Section 3
- 3D Modeling with Softimage

Rotoscope
Let's start with top viewport. On top viewport's click on **Wireframe -> Rotoscopy** Options...

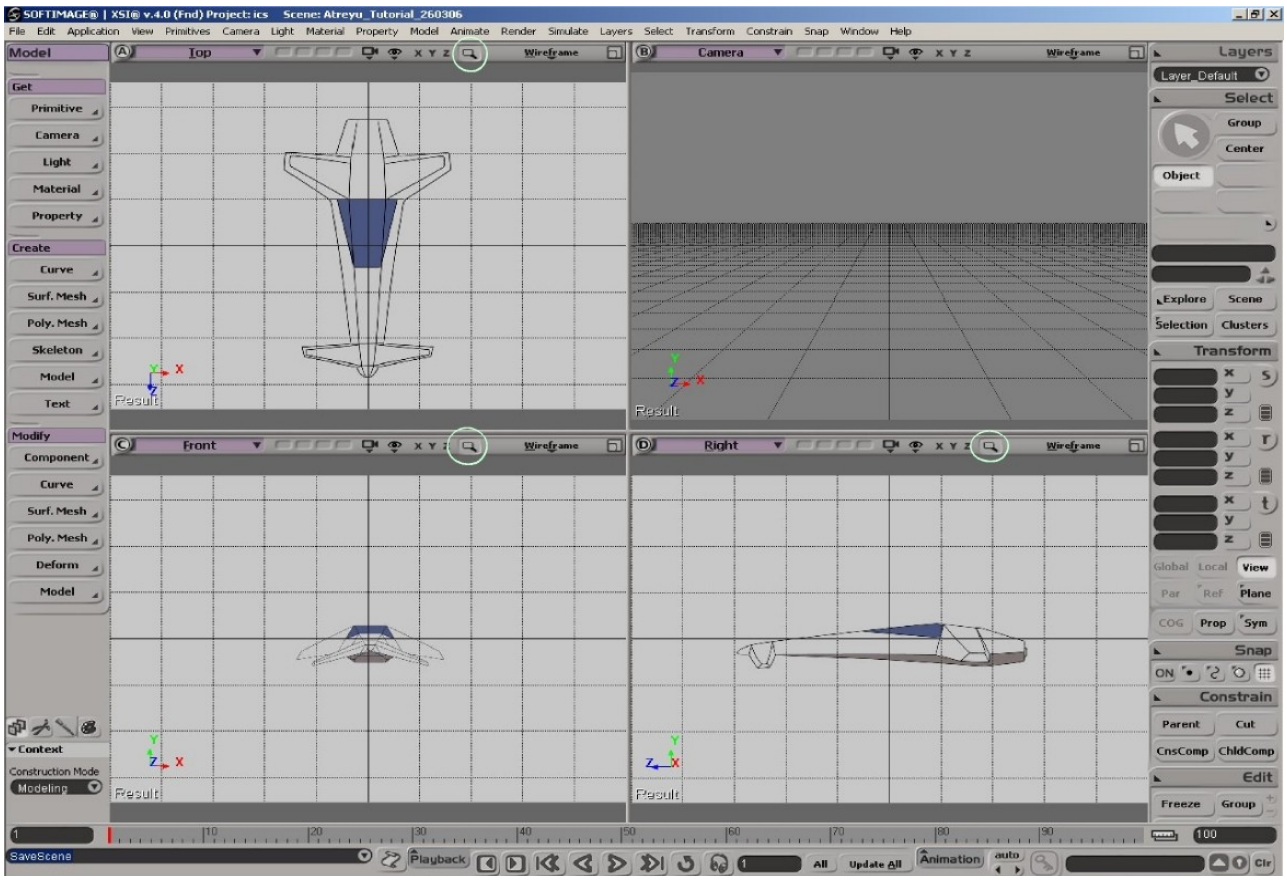


A PPG will appear, click on **New -> New from file**... highlight *atreyu_top.jpg,* LMB double-click to select file and close PPG. The rotoscoping is not enabled yet, to enable the rotoscoping click on **Wireframe -> Rotoscope**.

Do the same for side view and front view. Use atreyu_side.jpg and atreyu_front.jpg.

When you zoom in or track with rotoscope view enabled you will notice that the image does not change with size or placement with the grid and every objects in the scene. The rotoscope image is fixed in the center of the viewport. We don't want this, this would make the modeling process difficult as you'll need to pan and zoom into the viewport.

Once you loaded all the rotoscopy images, now lets track the camera movement with the rotoscopy images by clicking on the magnifier-looking icon on viewport top-bar. Do this for top, side and front viewport.

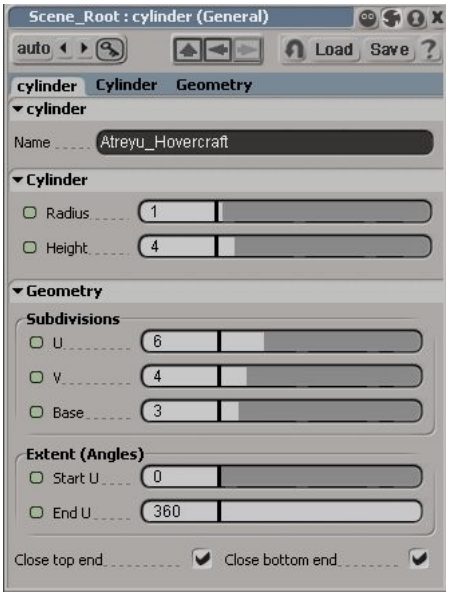


Save your scene. We've put up the images in the viewport and we're ready for the modeling part.

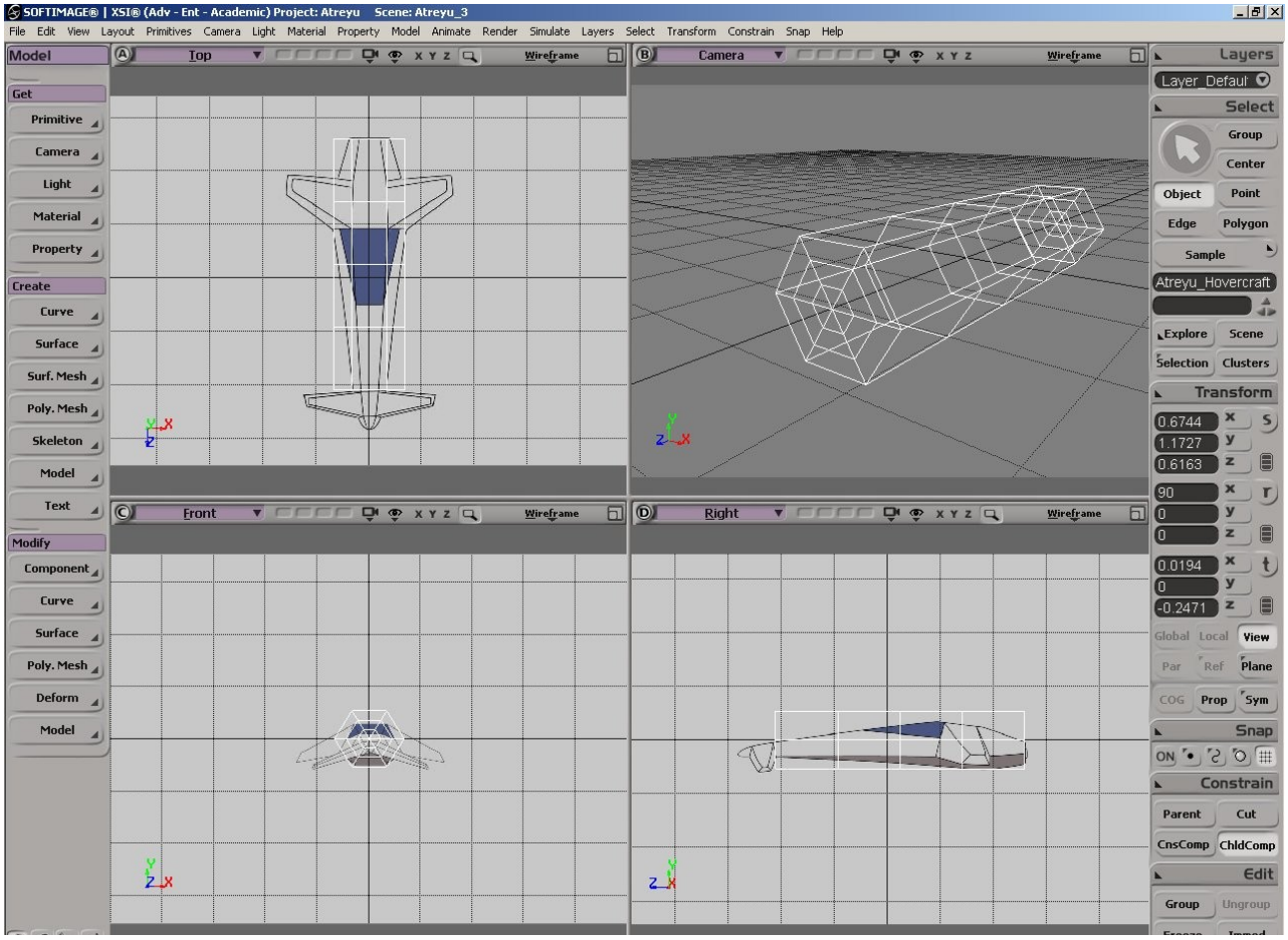
Modeling

Let start with a cylinder. Click on **Primitives -> Poly mesh -> Cylinder**. Use these following settings;

- Radius = 1; Height = 4
- U subdivision = 6; V subdivision = 4 and Base subdivision = 3
- Give the object a name, something you like



Rotate the cylinder 90° around x-axis and scale it so that the cylinder covers the basic shape of the hovercraft. Don't worry about covering the tip of the hovercraft. We'll shape the base-subdivision on the cylinder to shape the tip of the hovercraft.



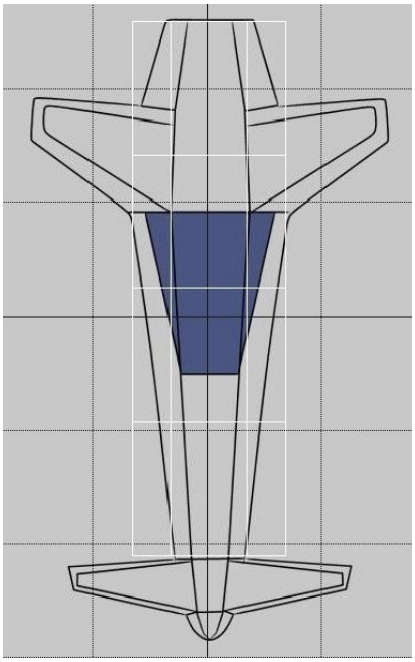
You might notice that the side view seems a little bit shorter than the top view, this is fine. You may ignore this. If you want to fix this, release the camera tracking, zoom and pan the view so that the cylinder sits exactly on the image plane then activate the camera tracking again.

I started out with a cylinder because it resembles the shape of the hovercraft. I could've started with a sphere or a cube or even a grid, but it would take longer to get the shape of the hovercraft. As I mentioned from the previous tutorial, clicking Cube button is not the only way to have a cube. You can create a cube from a sphere too, however, it takes more time and not productive.

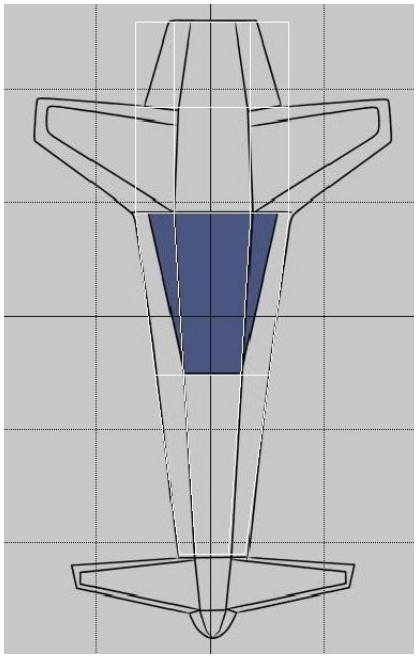
Here's the breakdown of what we are going to do next;

- ➔ Move the vertices to shape the basic hovercraft shape from the top view.
- ➔ Move the vertices to shape from side view.
- ➔ Form the tip of Hovercraft by using base-subdivision on the cylinder.
- ➔ Extrude 4 faces on the model to create the fins.
- ➔ Refine the model to your liking, or improve it.
- ➔ Setting up UV mapping for texture placement.
- ➔ Creating the collision boundary mesh.
- ➔ Freezing up the construction history.
- ➔ Export out to .OBJ format which Blender is able to Import in.

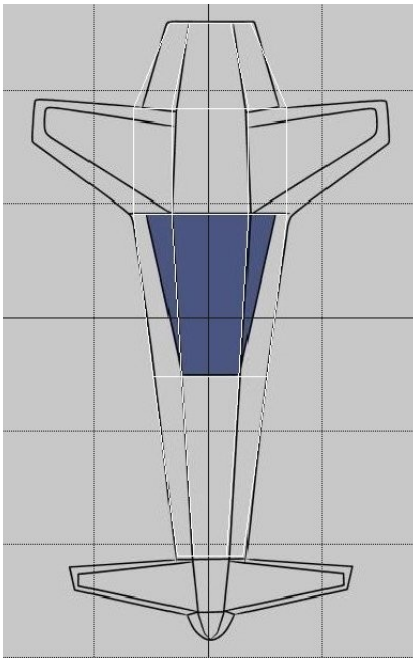
Maximise your top view and let's start building the hovercraft.



From here...

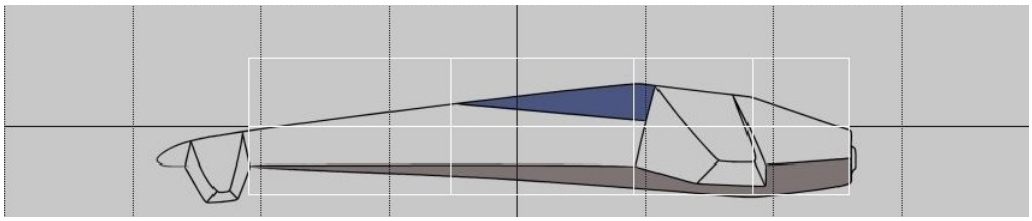


...move vertices by translating along z axis and scaling vertices along x local axis...

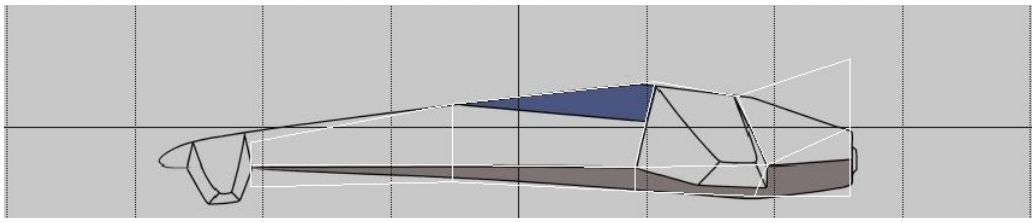


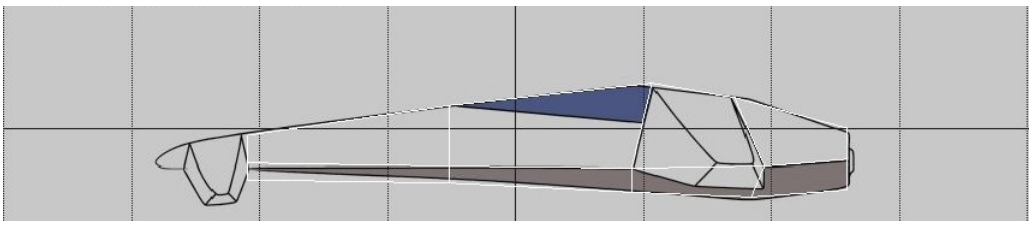
...to here. And we're done with top view.

Maximise your side viewport and start moving the vertices to get the shape of the hovercraft from side view.



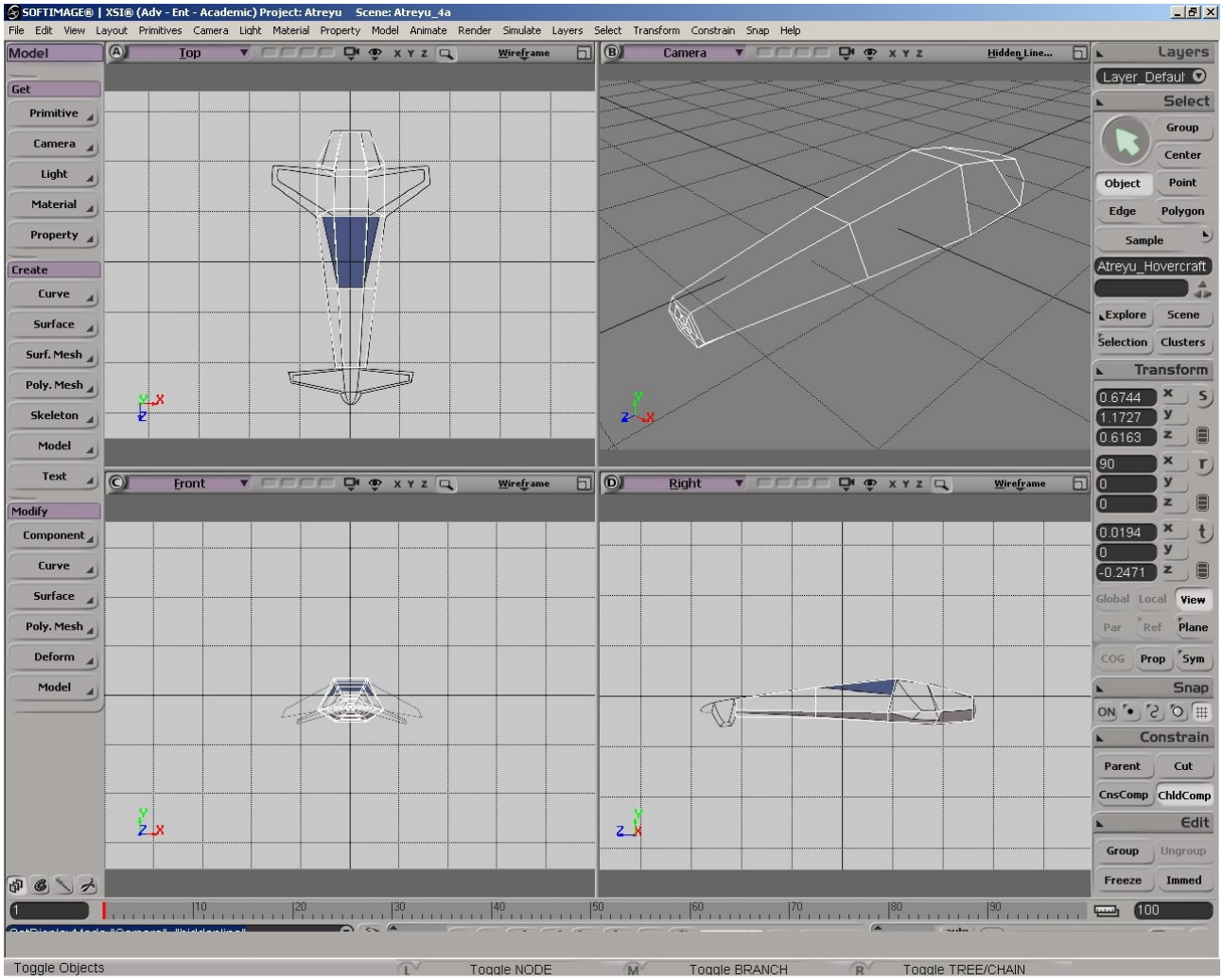
From here...



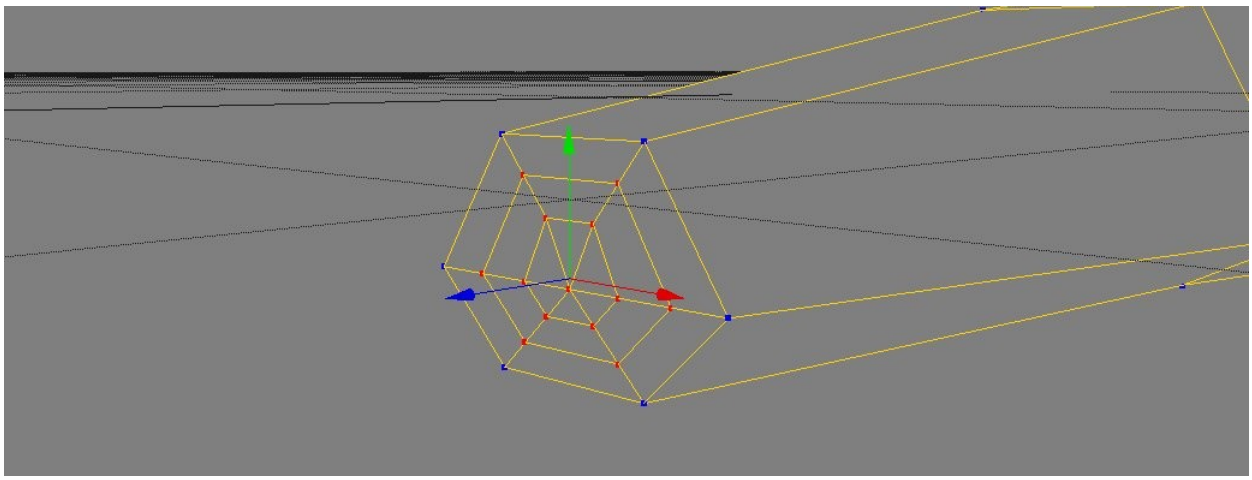


...to here and we have the basic shape of the hovercraft.

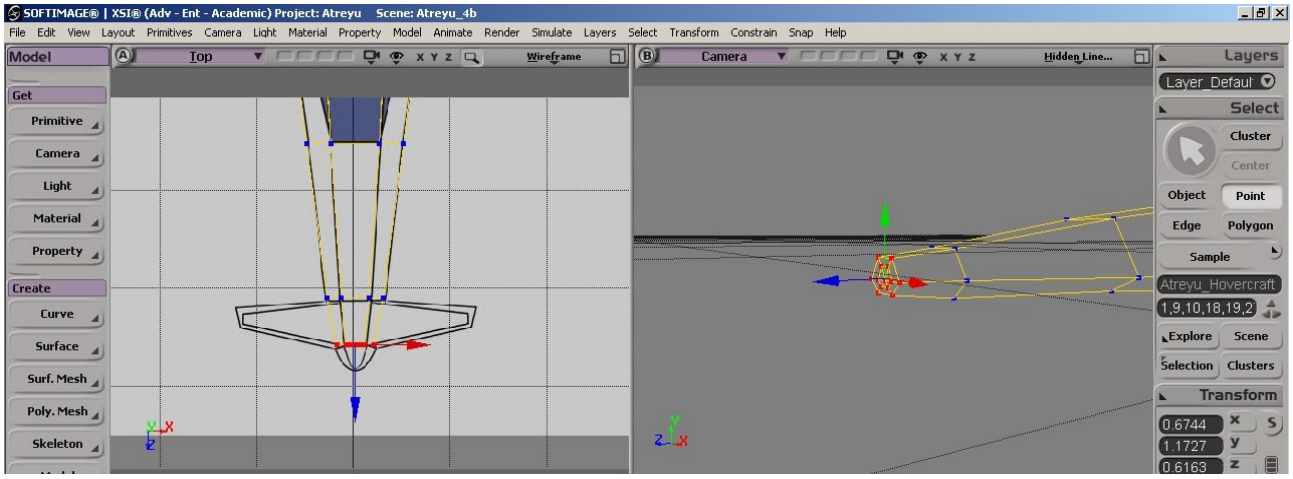
Up to this point, you would see something like this illustration below. Shaping the tip of the hovercraft would be easy from the perspective viewport with occasionally referring to side and top images.



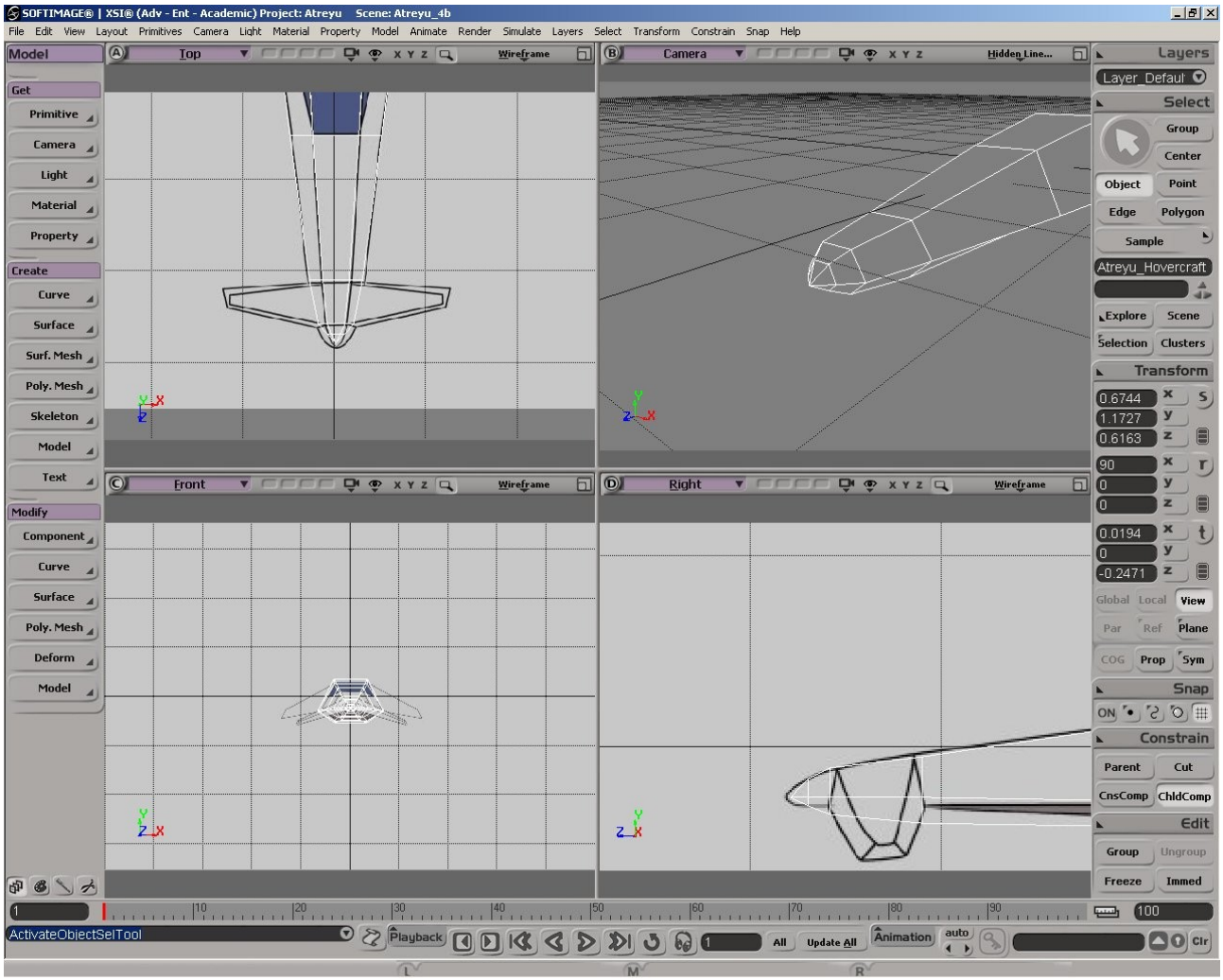
We will now shape the front tip of the hovercraft. Select all the points of the base subdivision on the front...



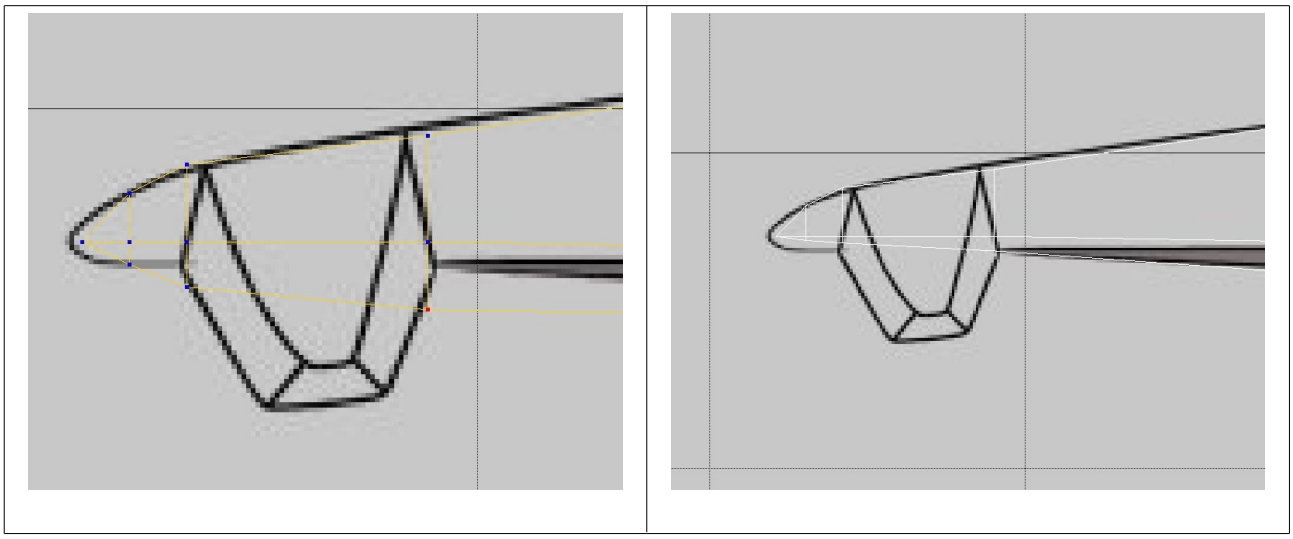
...then move them forward to the front side of the front fin.



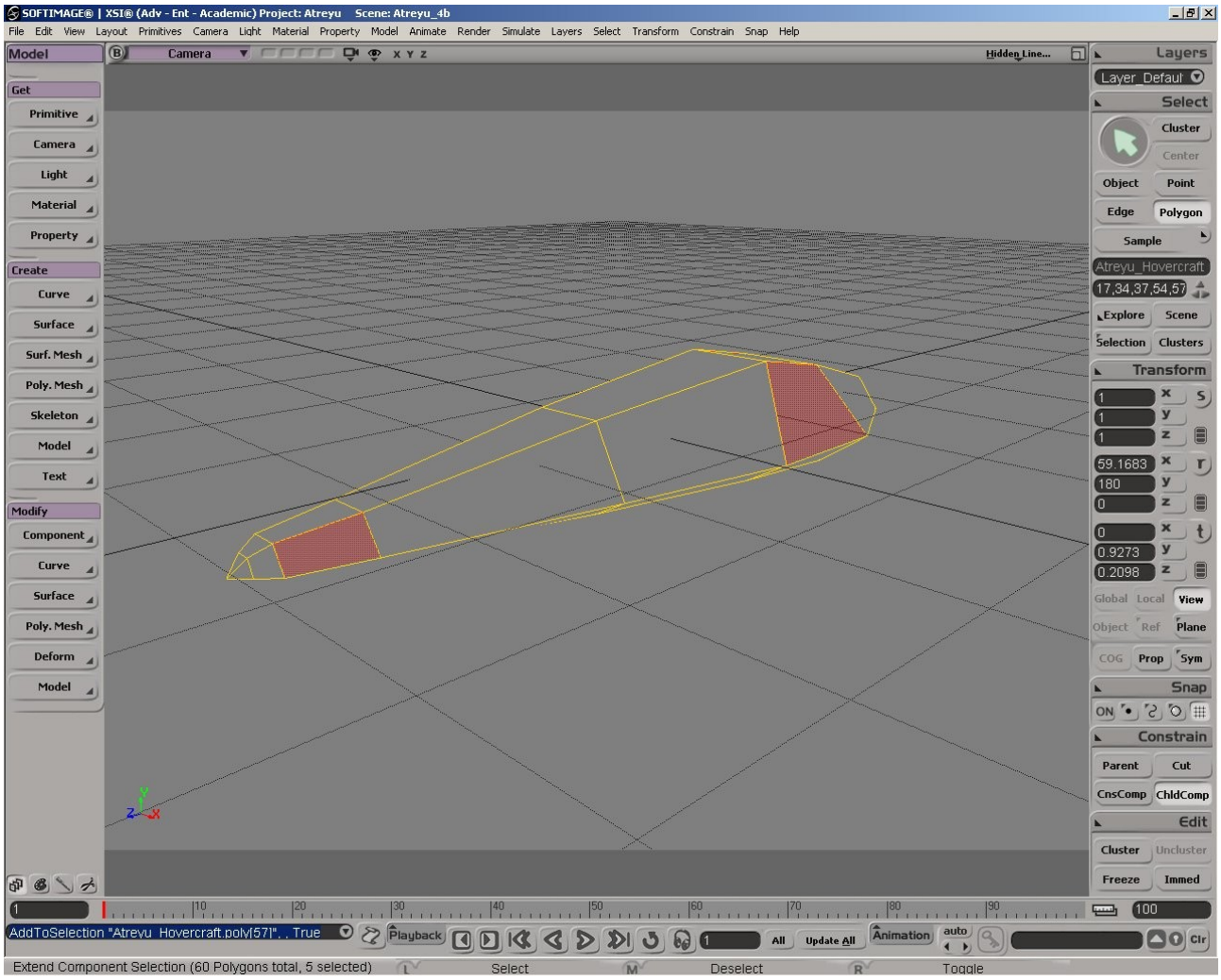
Shape the inner vertices of the base-subdivision so that it looks like the sketch of the hovercraft..



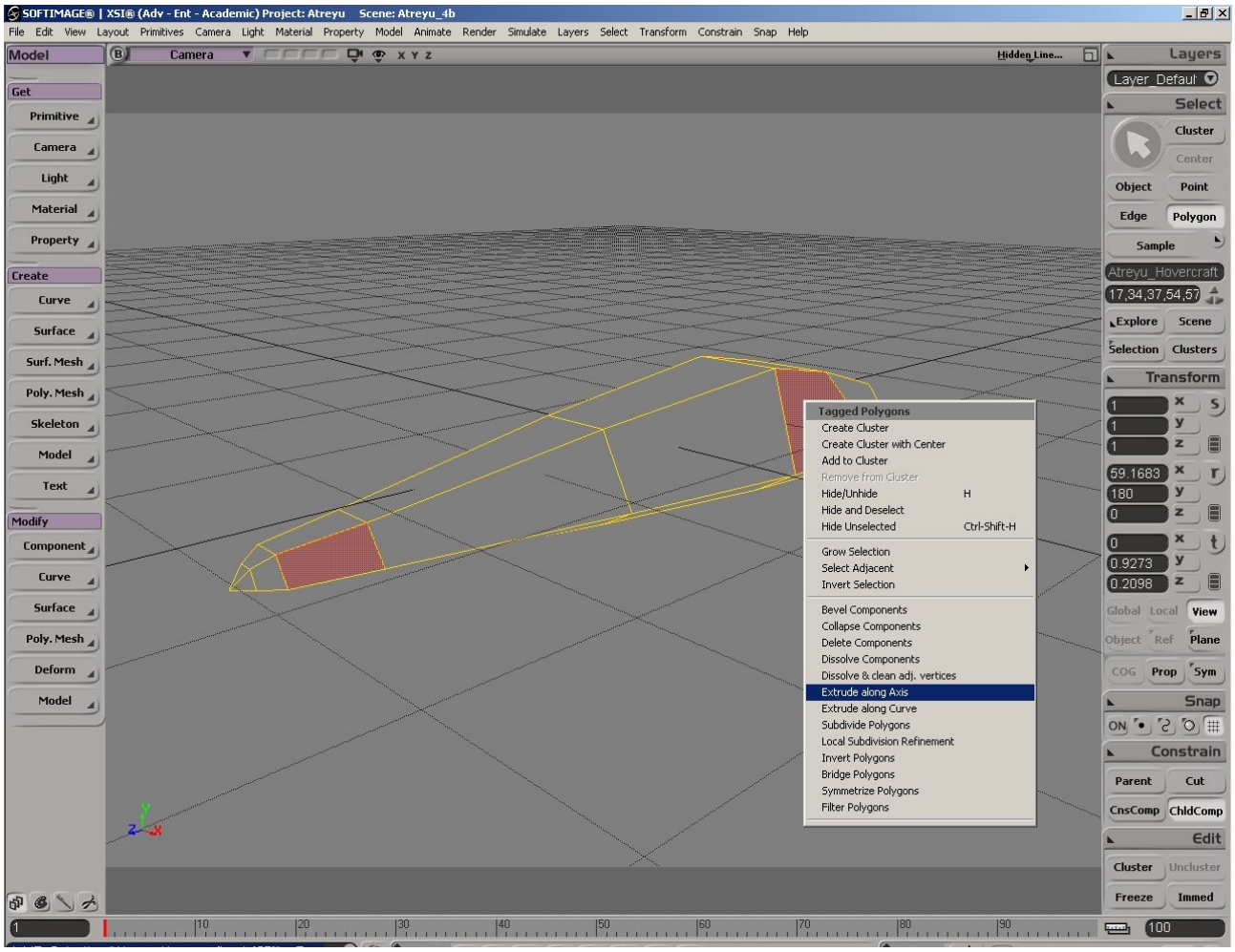
Go back to side view, and move the vertices to refine the bottom shape of the hovercraft (if you haven't already done this).

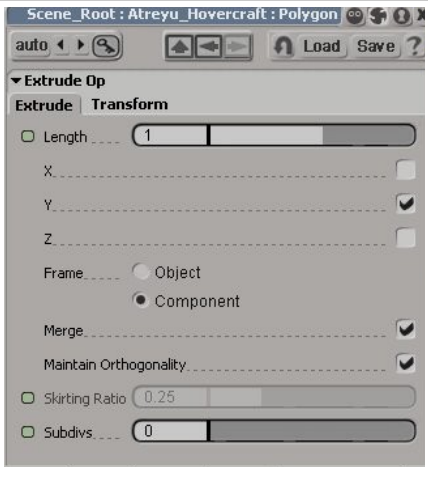


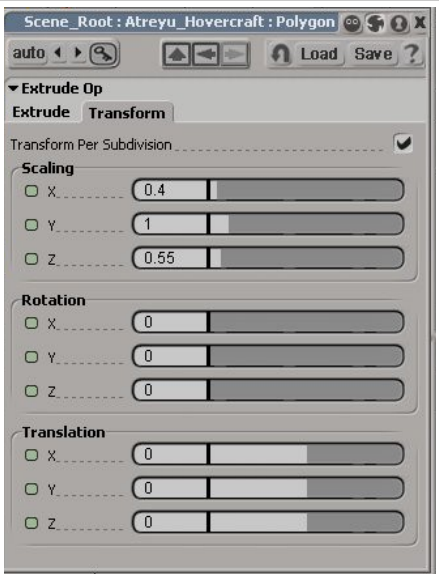
Let's create the fins now. Select the faces (4 faces in total, 2 on each side) on the model which the fin will be extruded from. Use U key to ray-cast select polygon.



Move the cursor on one of the highlighted face, do RMB click **-> Extrude along Axis**. A PPG will pop up on the screen

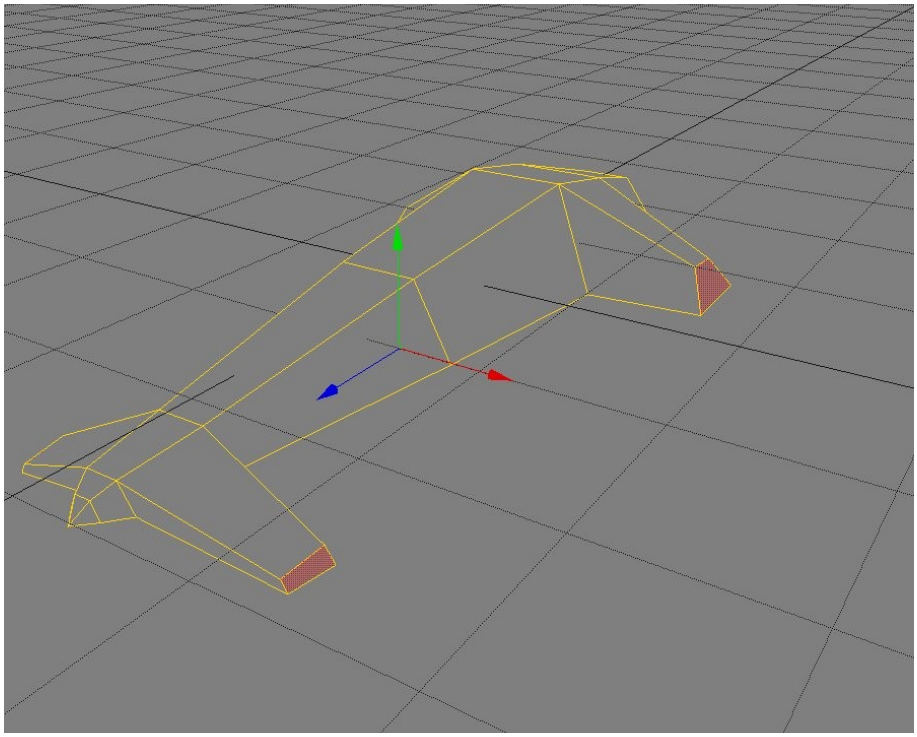




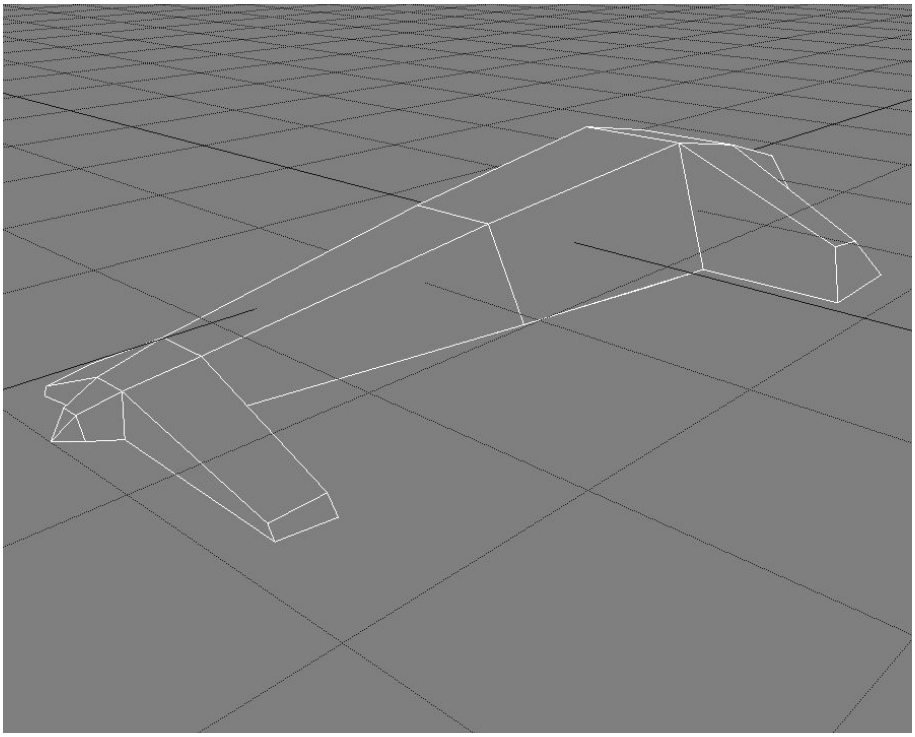


Under *Extrude* tab; set length to 1, enable *y*-axis, Frame component, set merge and set Maintain Orthogonality. Under *Transform* tab; set the x, y and z scaling to 0.4, 1 and 0.55 respectively. Close the PPG when you're done.

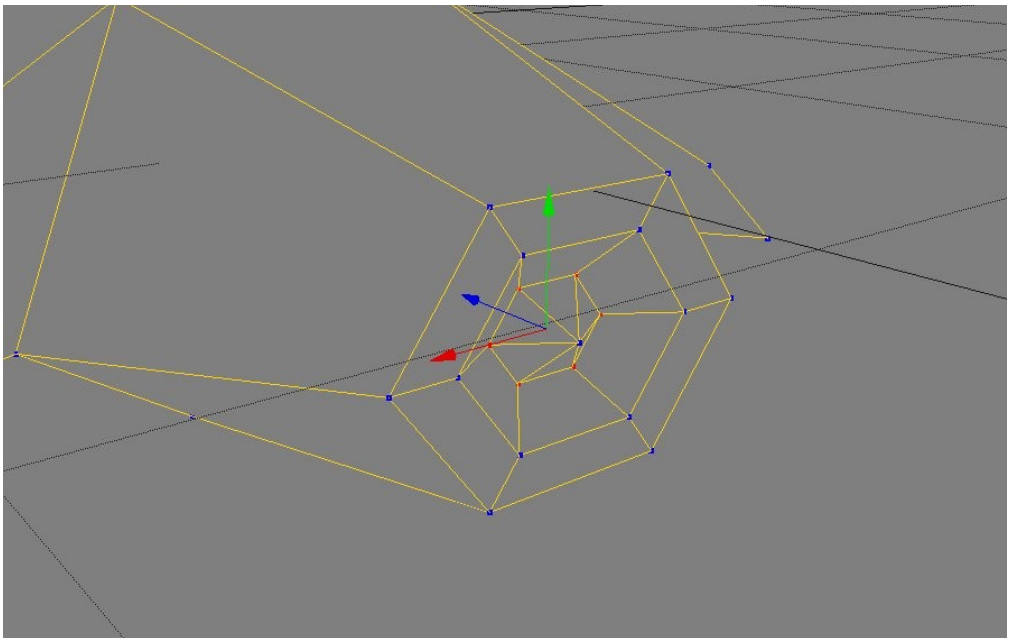
With the 4 faces still highlighted, translate them downward and backward a bit.



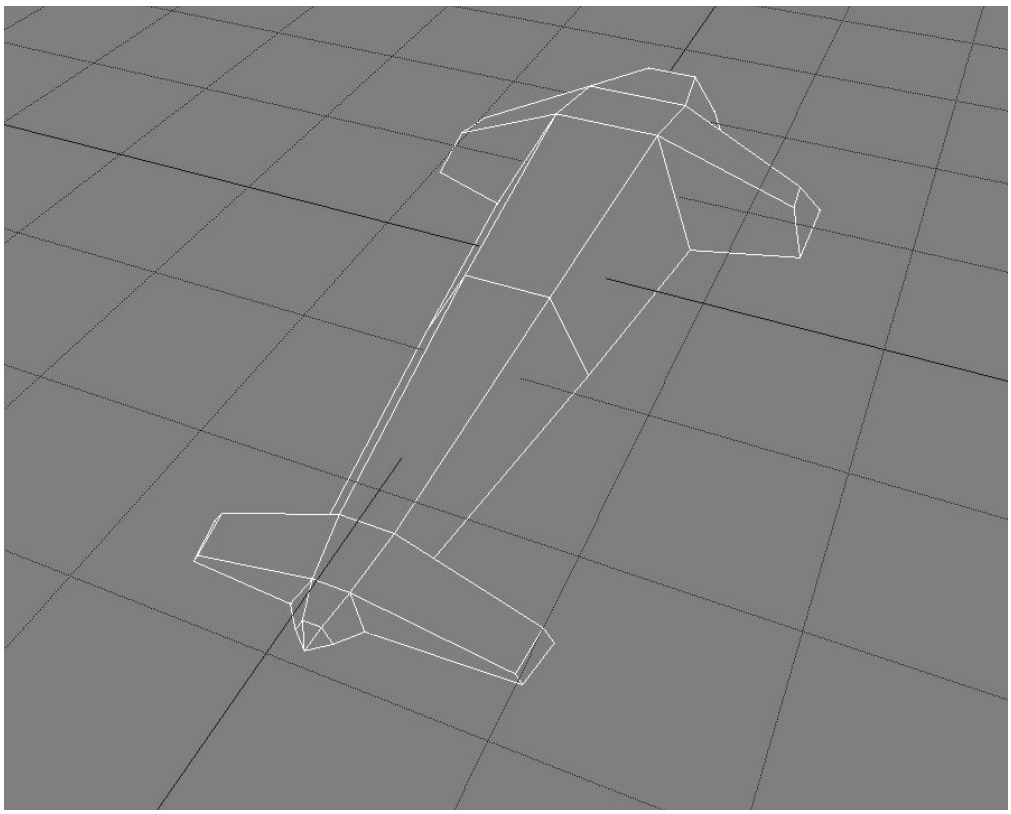
At this point you would get something like the illustration below.



To shape the exhaust, simply select the middle vertices of the base subdivision at the back of the hovercraft, and move them inwards. See Illustration below.



You would probably get something like the illustration below. Don't forget to save your scene.



That's all folks! We're just finished the modeling process of the hovercraft. If you wish to add more detail or modify the shape of the hovercraft go ahead. When you're ready, we'll be setting up UV coordinates of the hovercraft.

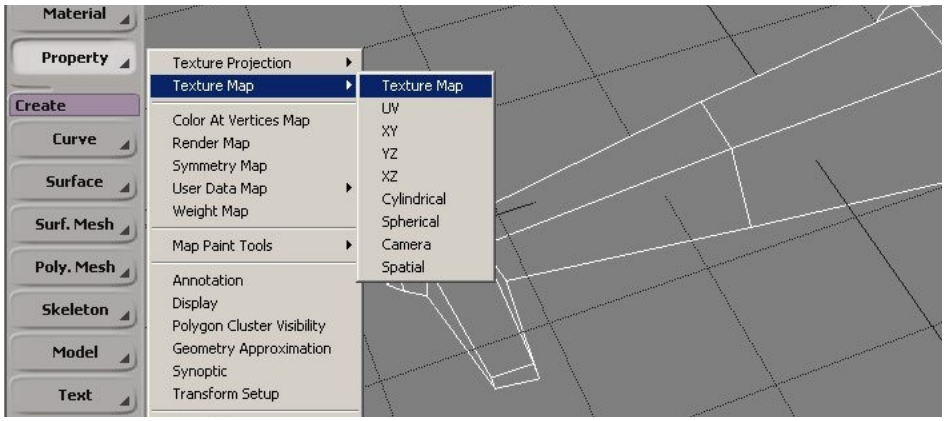
Setting up UV Coordinates

Before putting textures on the object we need to specify the UV coordinate; basically it's like which part of image will be projected to which part of the model.

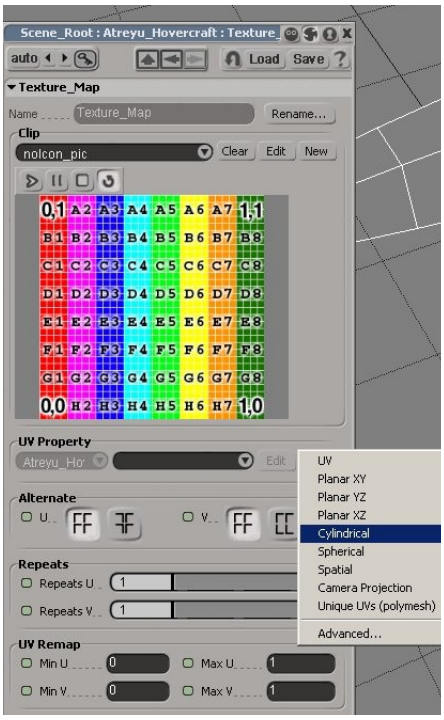
Here's the breakdown of what we are going to do to set up the UV coordinates of the hovercraft;

- Apply a full cylindrical projection to the hovercraft.
- Apply sub-projections on the hovercraft (top, bottom and back exhaust).
- Stamp out the UV coordinates. (to help us in texturing)

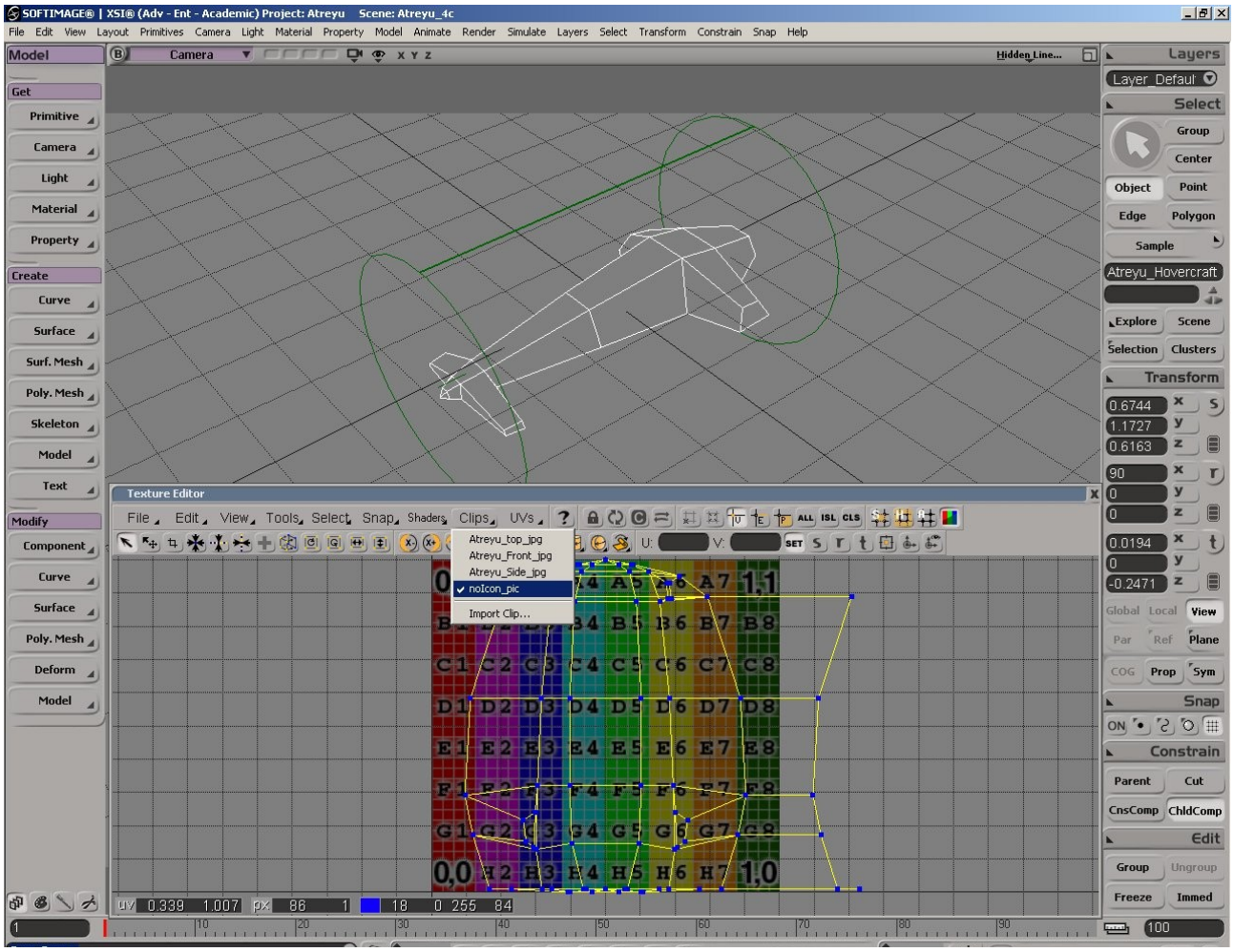
Select your hovercraft, apply cylindrical texture projection from *Model Toolbar* **-> Property -> Texture Map -> Texture Map**. A PPG will pop up.



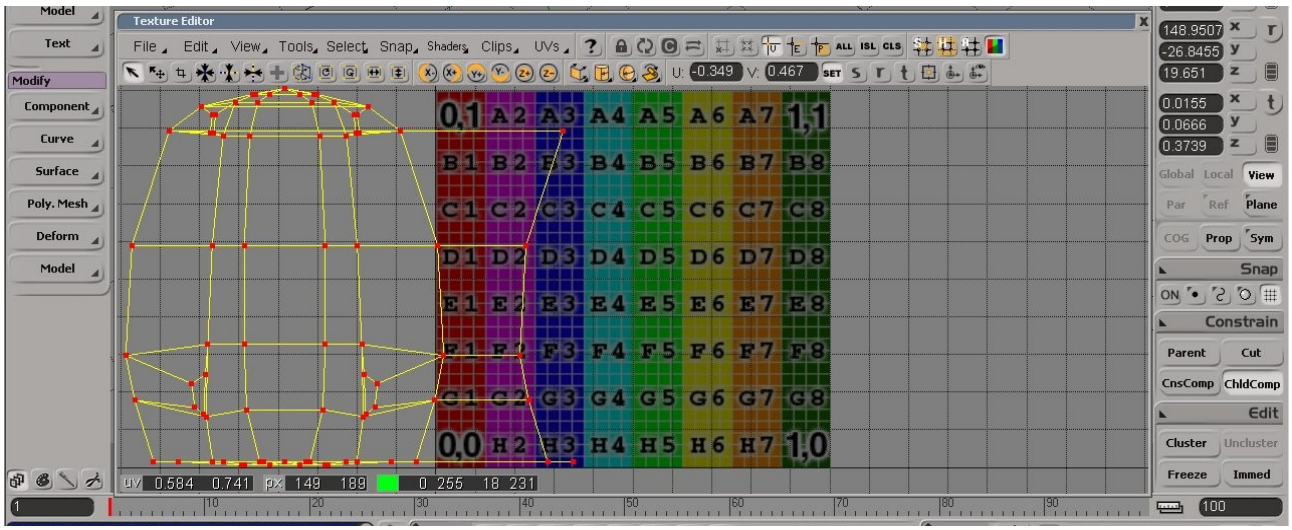
Leave the current image to *noicon_pic* and set the UV property to cylindrical by clicking **New -> Cylindrical**. Once you applied the cylindrical projection, close the PPG.



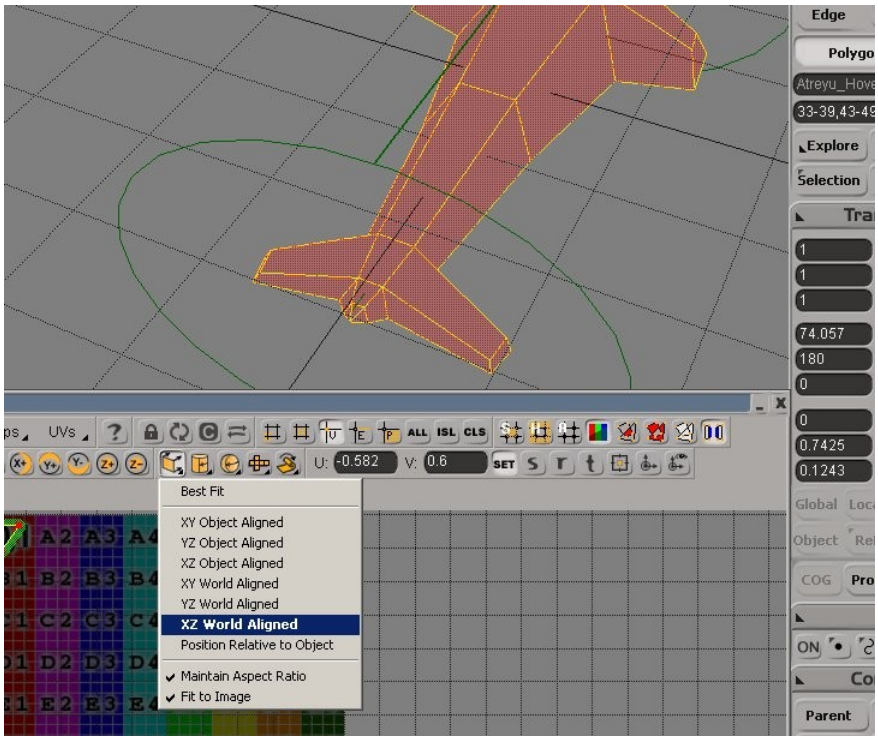
Open up Texture editor by pressing Alt + 7, and set the background image to *noicon_pic* by clicking **Clips** (on texture editor toolbar) **-> noicon_pic**.



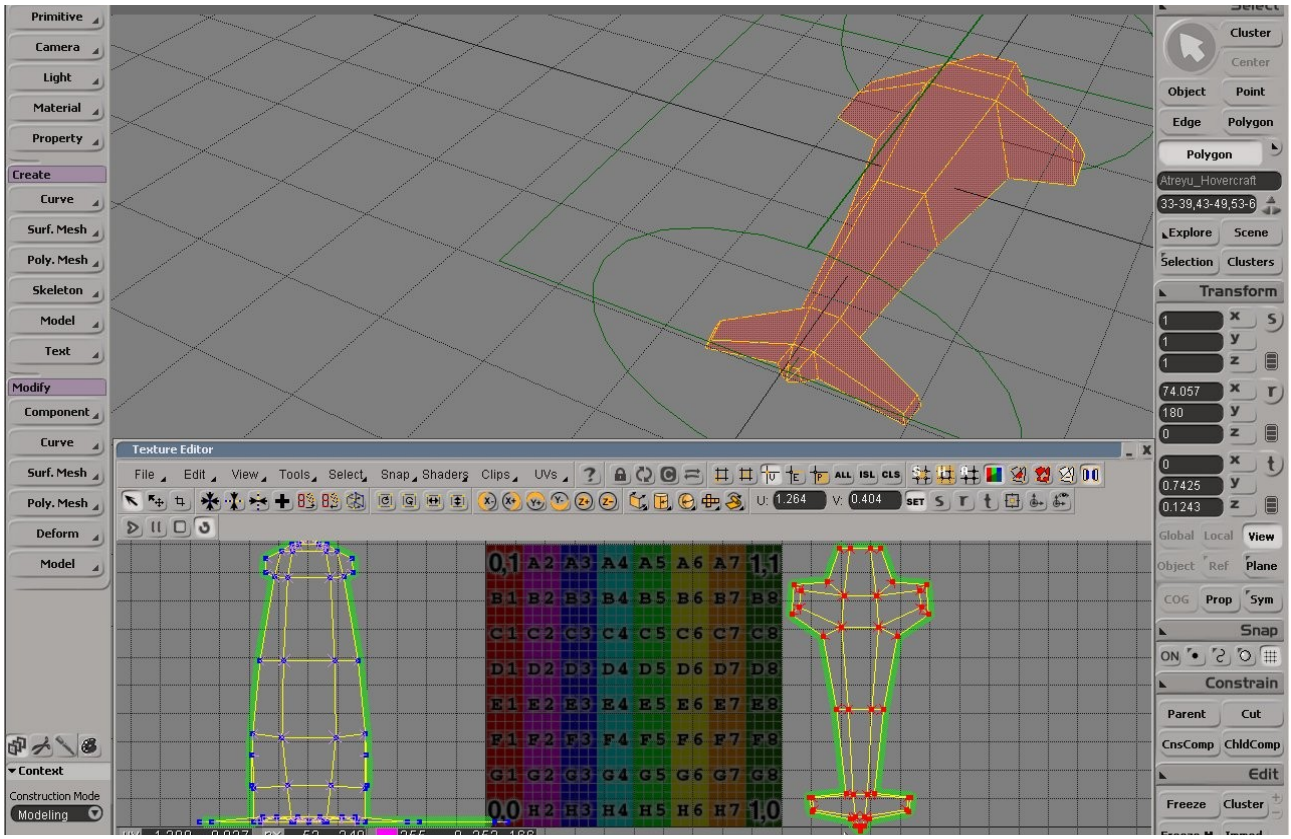
If your cylinder projection looks different from the picture above that's alright. It might be caused by the way you did the modeling process. Let's move all the points in Texture editor to one side before we apply the sub projections. Use T key to tag all the points and V key to translate it.



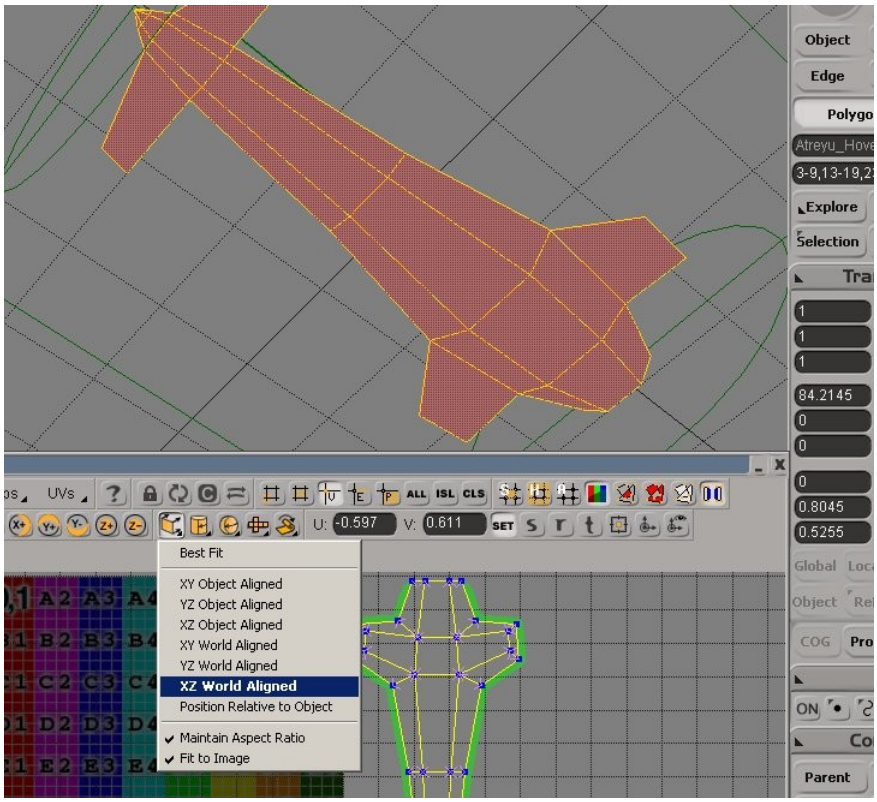
Select only the polygons that are visible from the top, then click *Planar Sub-projection* icon and select **XZ World Aligned**.



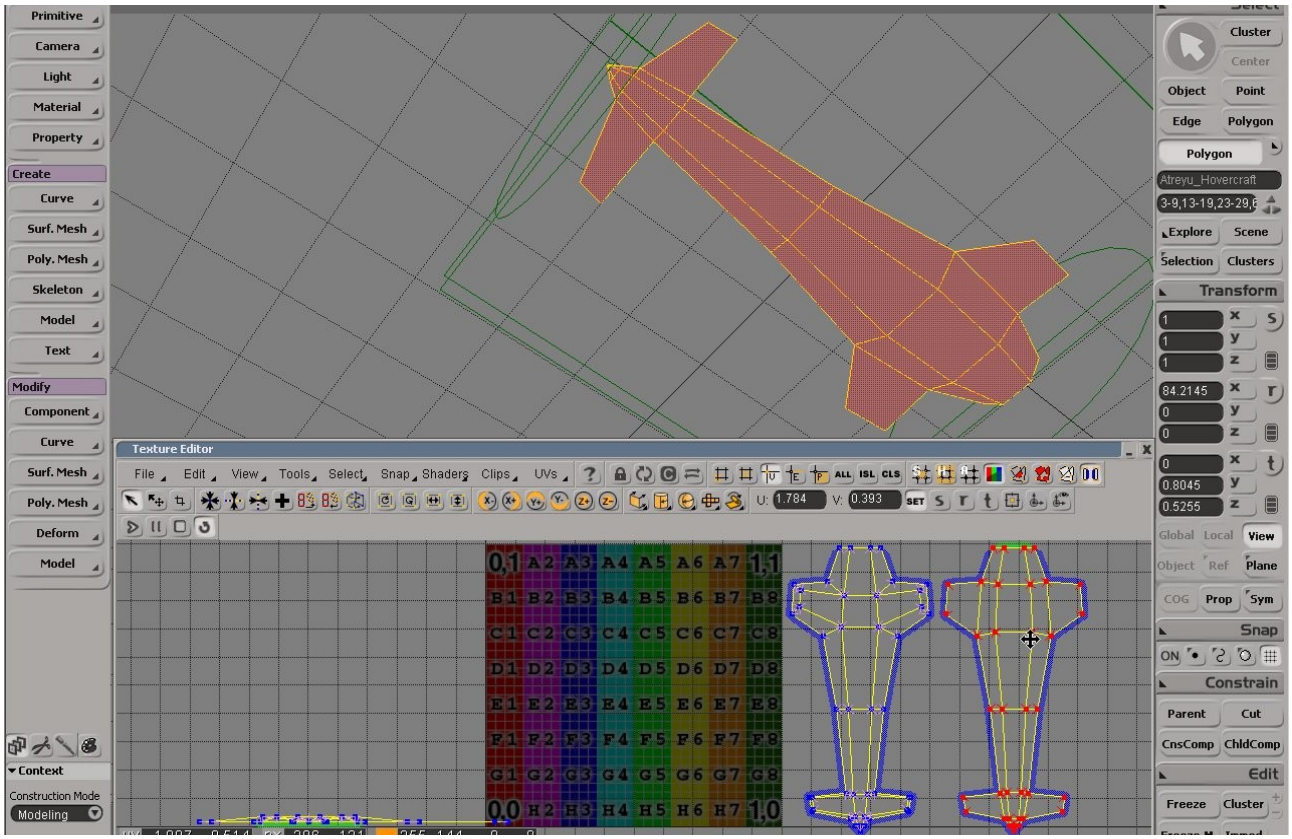
Move the top sub-projection of the hovercraft to the right side of the screen.



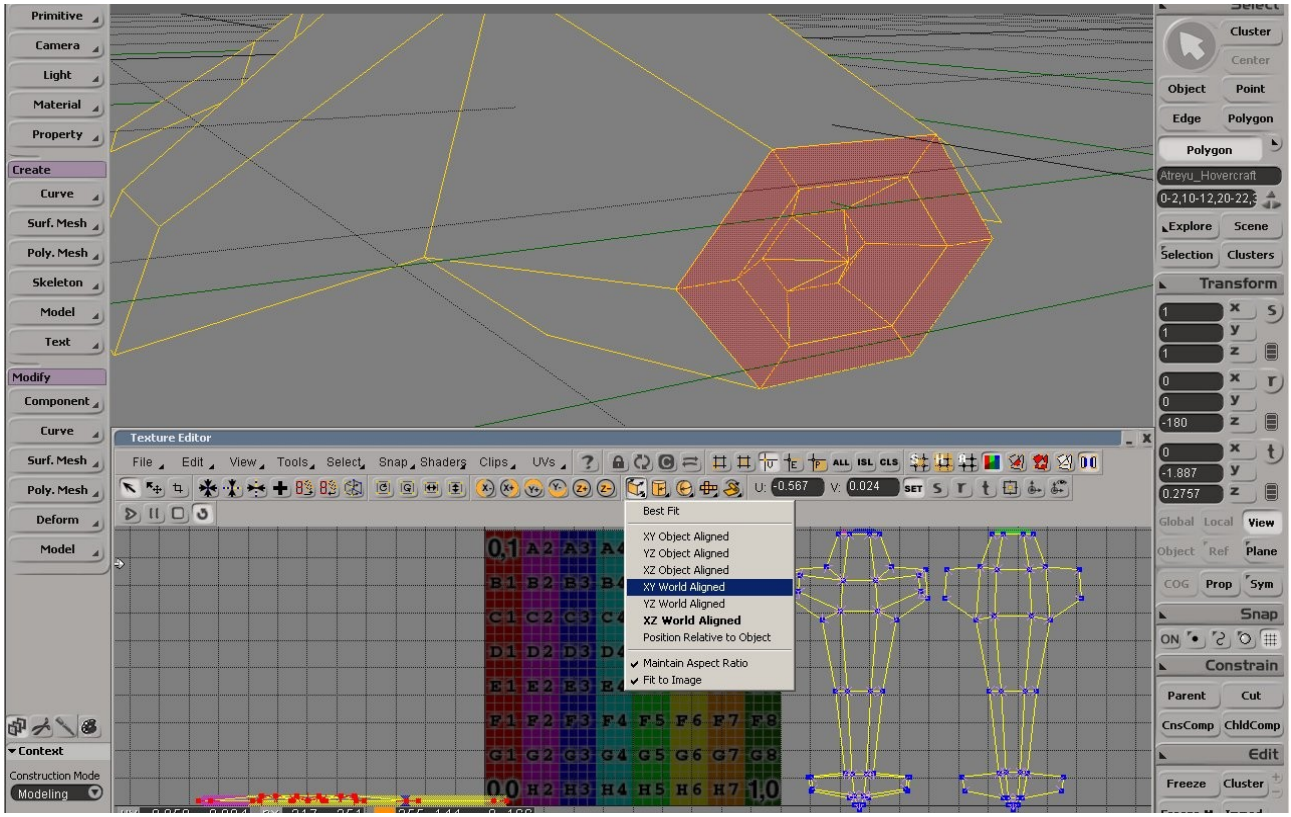
Now, select only the polygons that are located at the bottom of the hovercraft, then click *Planar Sub-projection* icon and select **XZ World Aligned**.



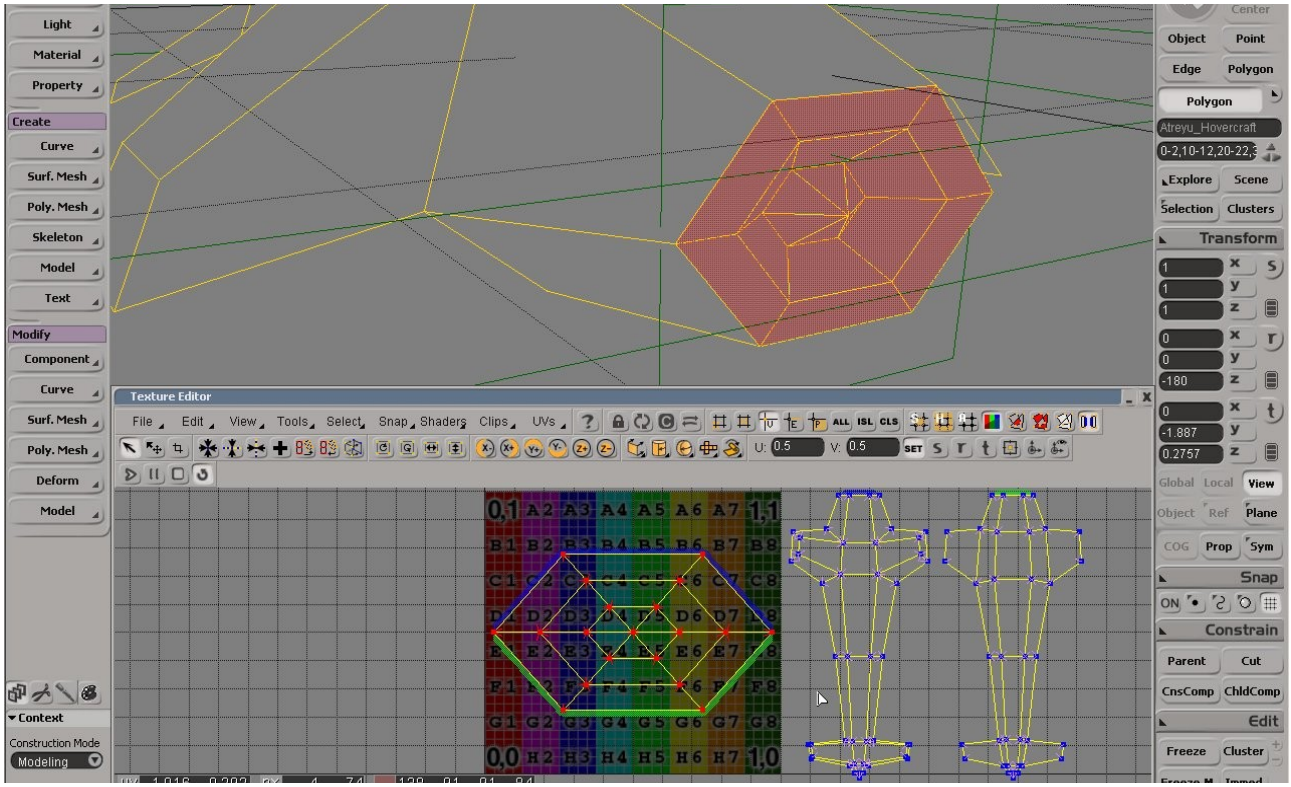
Move the bottom sub-projection of the hovercraft to the right side of the screen.



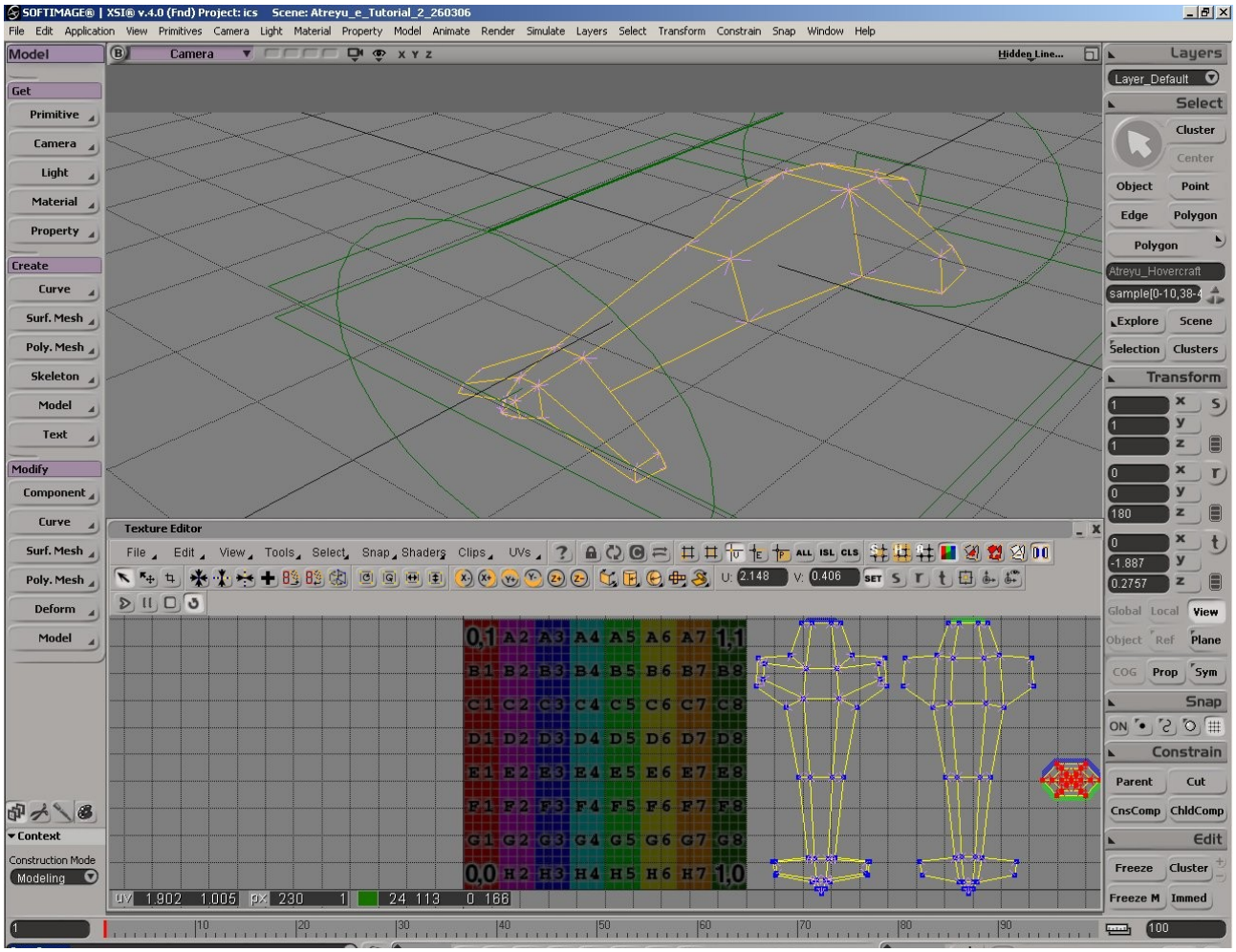
The only polygons left to be UV mapped are the polygons which located the back, the exhaust part. Now, select only the polygons at the back of the hovercraft that form the exhaust. Click *Planar Sub-projection* icon and select **XY World Aligned**.



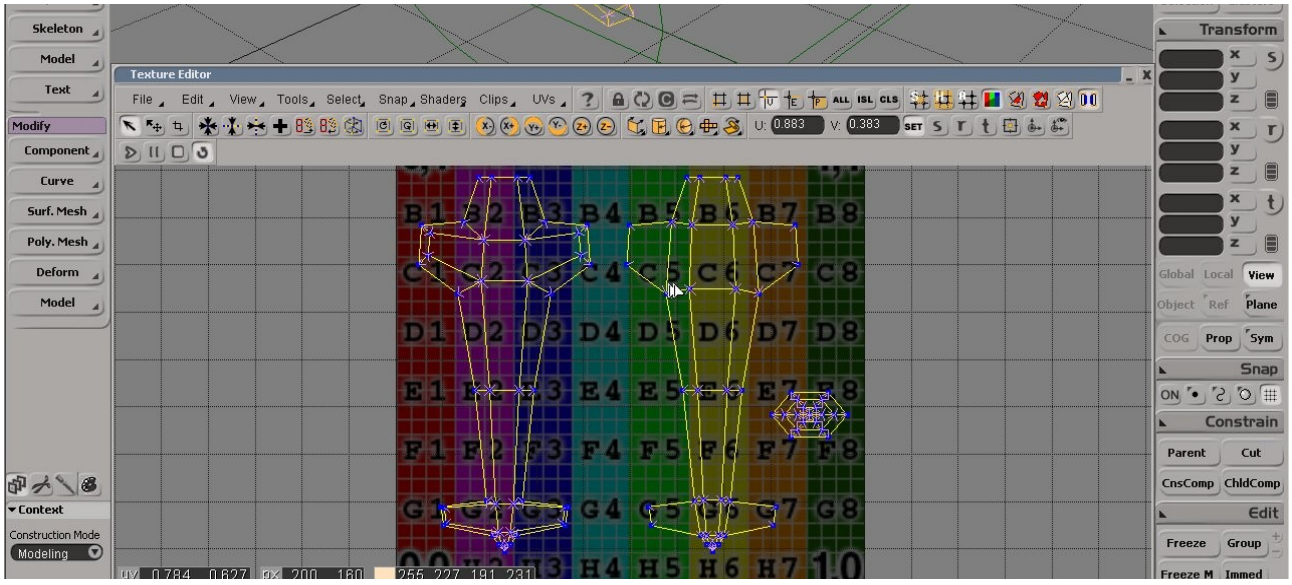
You would get something like the picture below.



Move the sub-projection of the exhaust to the right side of the screen. Scale it smaller to reflect the size of the exhaust from the top/bottom sub- projection (see picture below). If there are no more polygons left on the right side of the *noicon_pic* image, that means we have mapped every single polygons on the hovercraft.

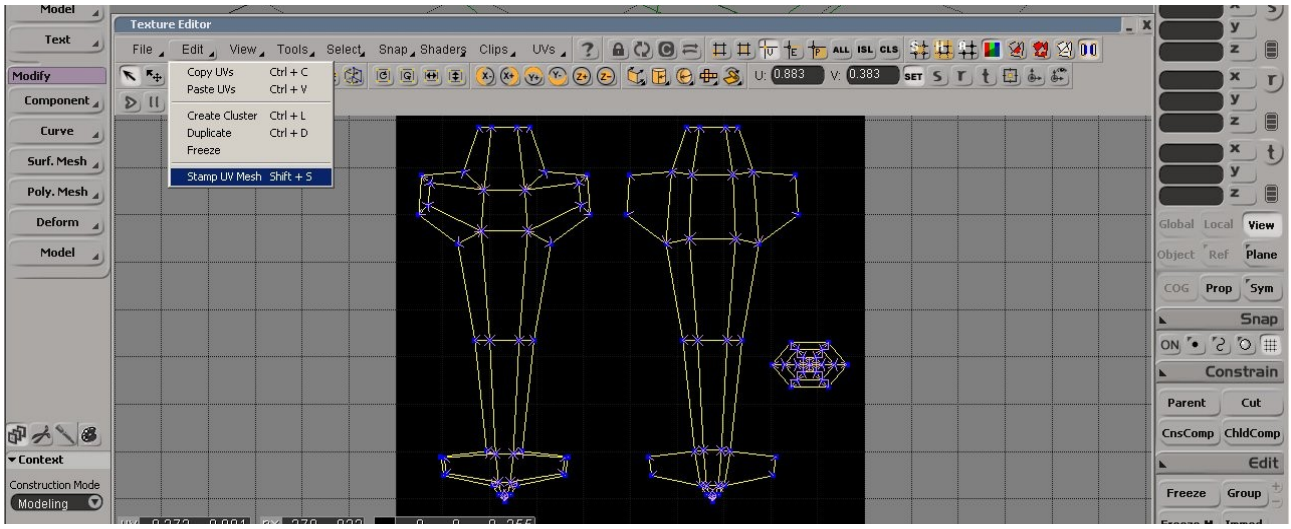


Now Arranged the UV's so that they sit right inside the *noicon_pic* image. My arrangement looks like this picture below. Use Z key inside Texture Editor window to pan or zoom view.

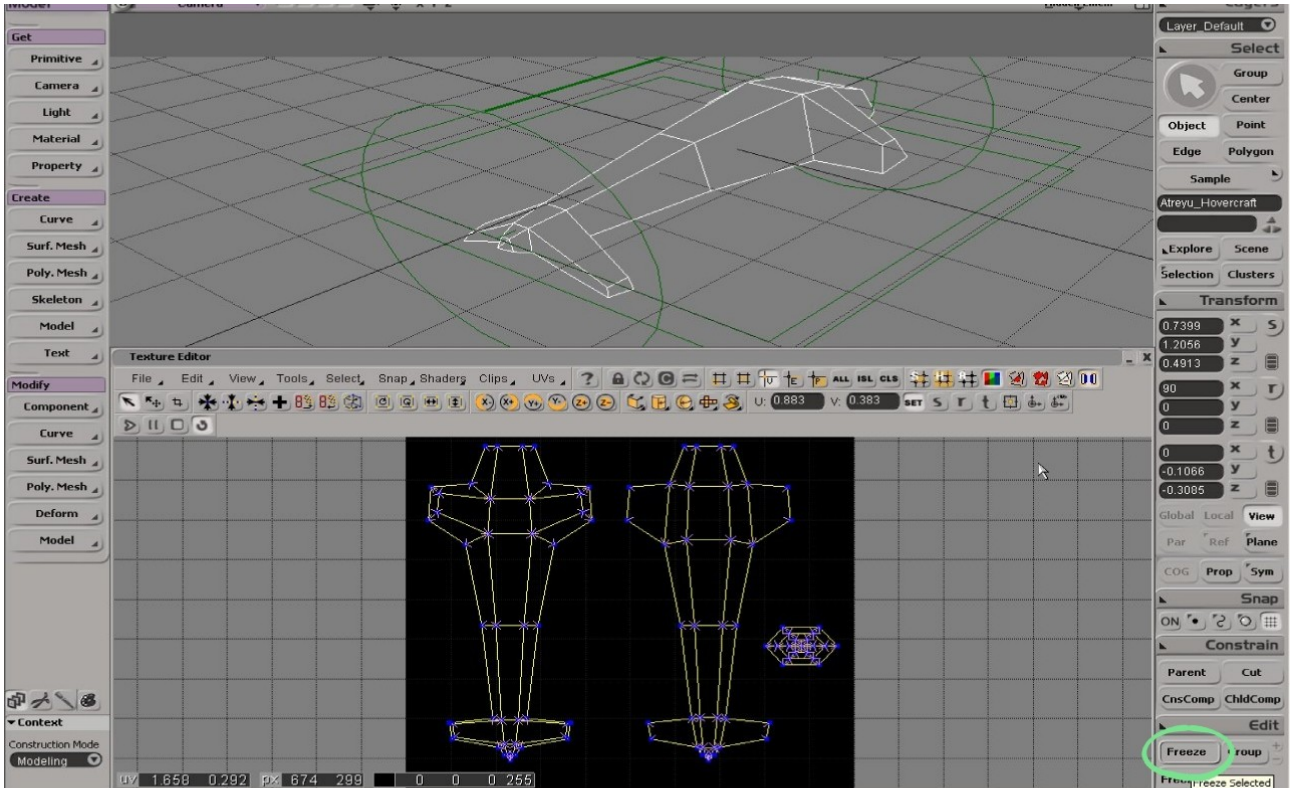


Stamp out the UV mesh (**Edit->Stamp UV Mesh**) to your own working directory as BMP or JPG file, this will be the texture file for the vehicle. You'll be given 256x256 by default. If you need higher resolution UV, before stamping out, click **Clips -> Import...** in Texture Editor to load a blank 512x512 or 1024x1024 picture. Actually 512 x 512 should give you enough space to create visible details on the texture.

In the picture below, I loaded my own blank 1024 x 1024 image filled with black. Seeing yellow lines of UV's on black would be much easier. I'm going to use a high resolution texture because I'll be putting lot of details in it. To make small details visible, you might need higher resolution than 256 x 256.



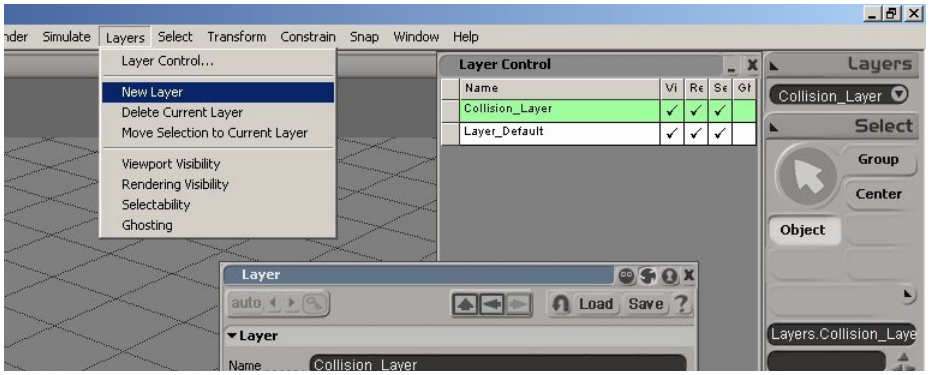
Don't forget to freeze transformation of the projections which we did earlier. This can be done by selecting all the hovercraft model and click freeze button at the bottom of MCP bar.



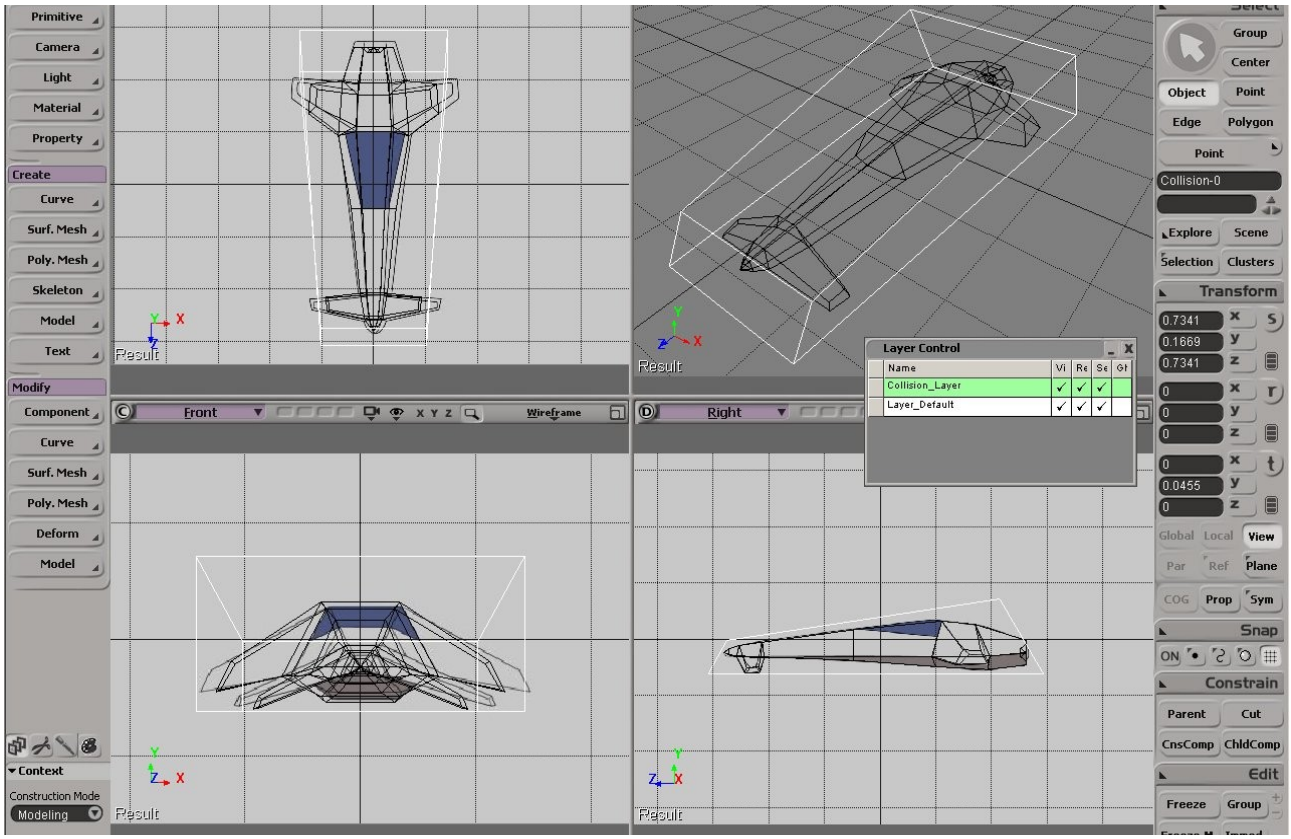
Once you hit the **Freeze** button, the texture projection will be frozen as well. So, at this point we're done with setting up UV Coordinates for texture placement. Don't forget to save your work. There are many ways to set up and to arrange UV Coordinates for Texture placements, that was just one of the method to lay out UV Coordinates.

Setting up Collision Box for Hovercraft

Create a new layer, the new layer should be activated by default. Name the new layer "*Collision layer*".



Create a cube from *Model Toolbar* -> **Primitives -> poly mesh -> cube**. Name it "*Collision-0*" Shape the cube so that it looks like the cube in the picture below.



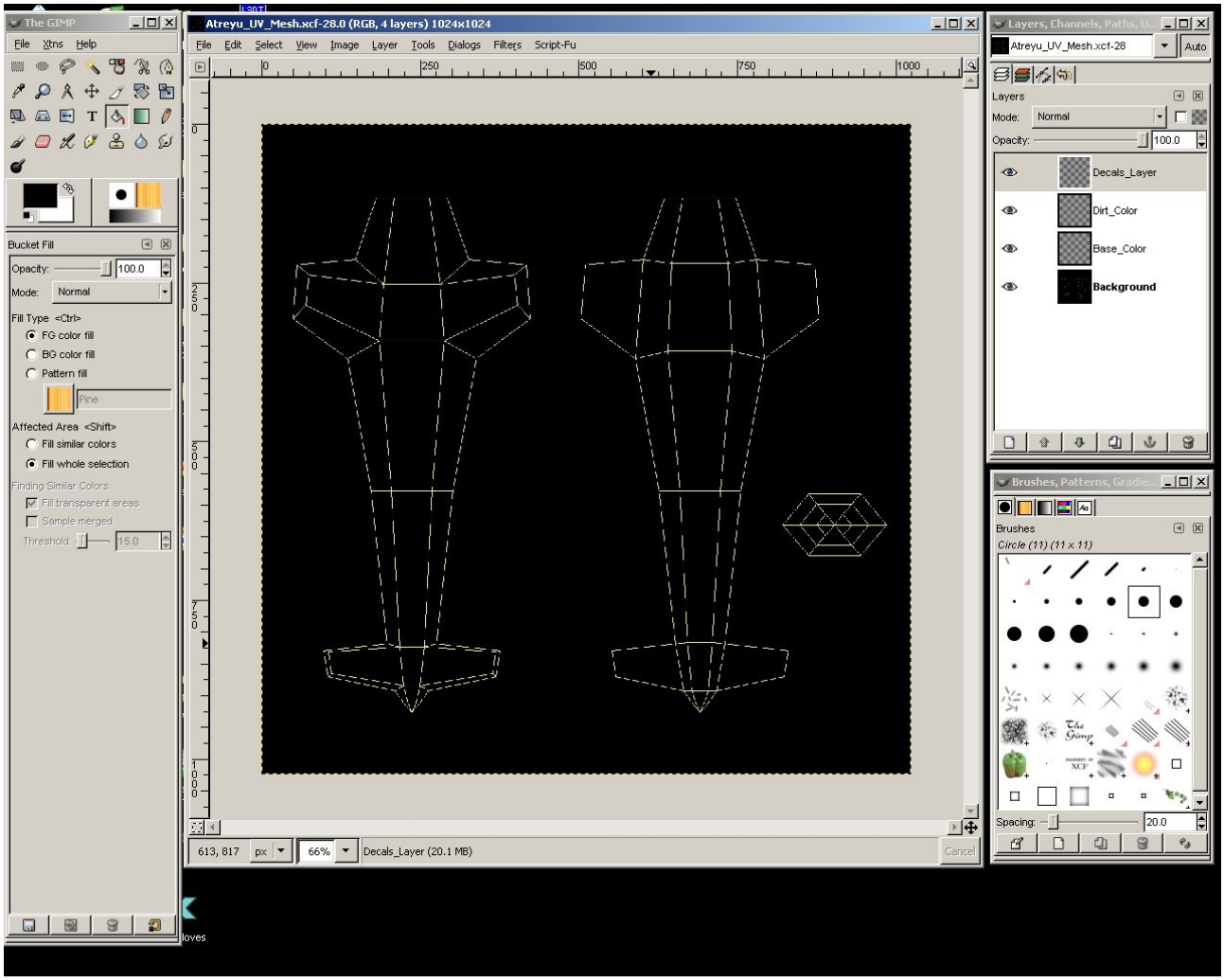
That's it. Save your scene. We're done here with XSI. We can set the collision box in Blender too, but doing it in XSI would be lot easier. Next stop we'll use Gimp to paint the texture.

Section 4
- Texturing the Hovercraft with Gimp

A note before we begin, if you're experienced in Adobe Photoshop, Corel Painter or Microsoft Acrylic, go ahead and use your favorite image editor. I'm using Gimp here because it's free! Download the latest version from www.gimp.org

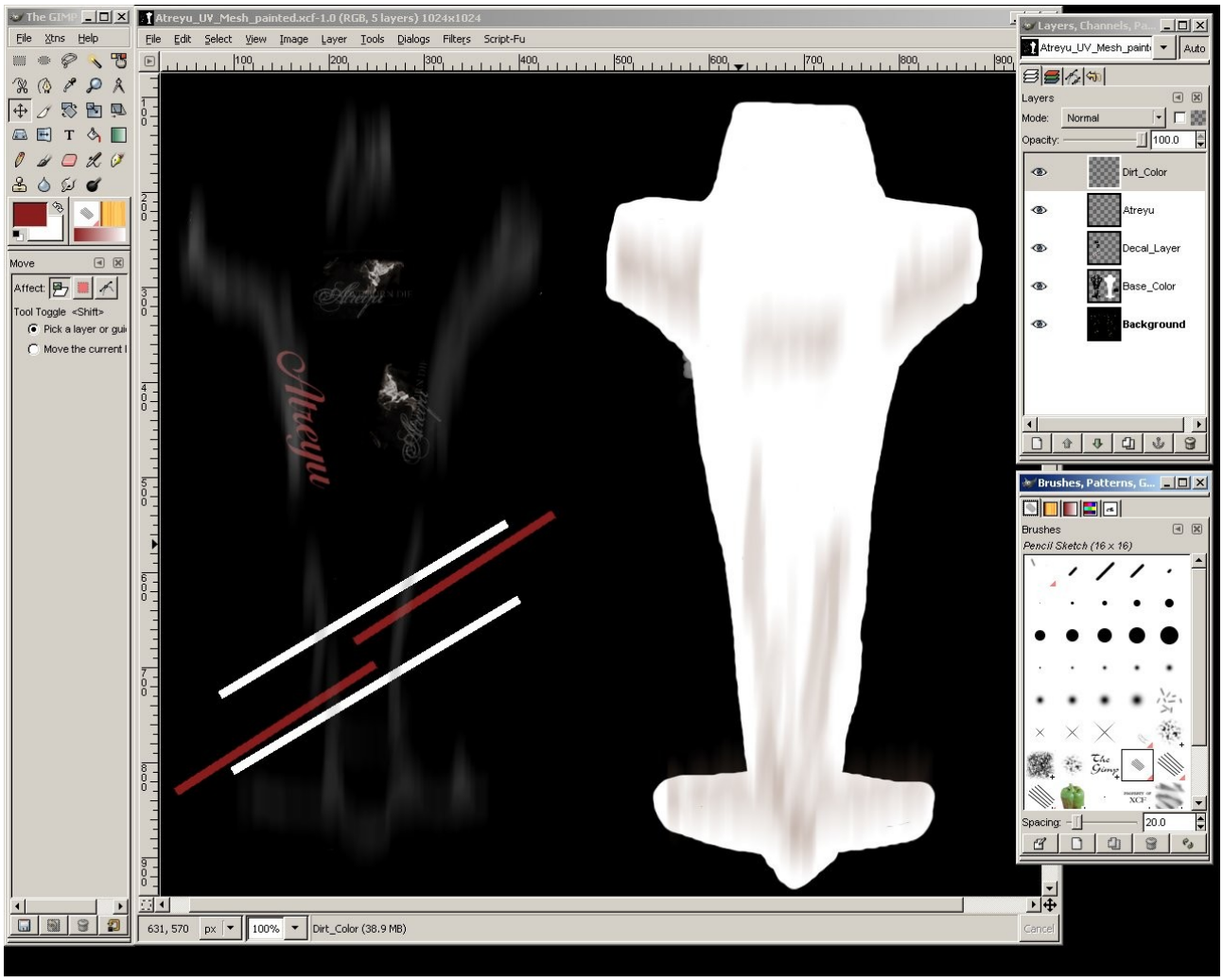
I'm not going to explain how to create a life-like texture here, many books have been written about this texturing subject. Search amazon Bookstore or Borders bookstore for books in digital texturing and lighting.

Launch Gimp, load up the stamped UV Mesh image file, and start painting. Because UVs Coordinates has been laid out properly, we can see which area to paint to give texture on the hovercraft.



It is recommended to work in layers. Create layers for different type of colors. In the picture shown above, I created 3 layers; base_color, dirt_color and decals_layer. Pick a brush and pick a color and paint away!!!

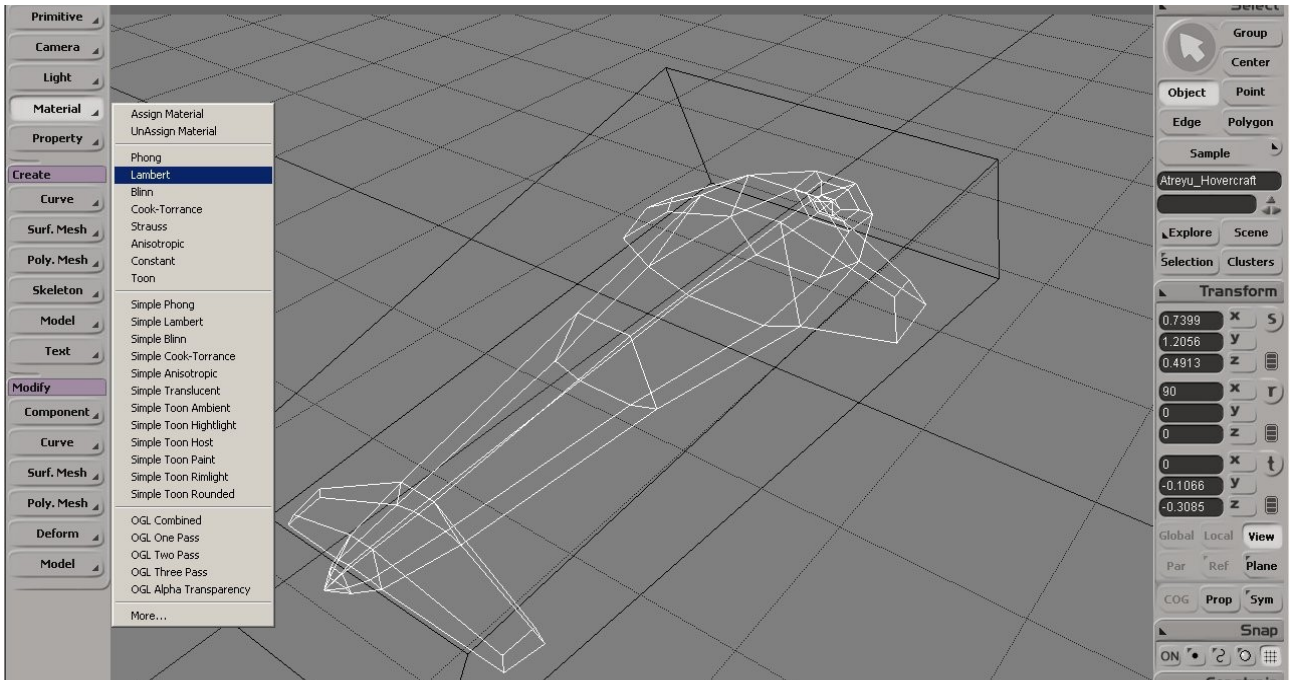
I painted the bottom of the vehicle to be white with brown dust, black color on the top, couple of decals and some stripes on the top.



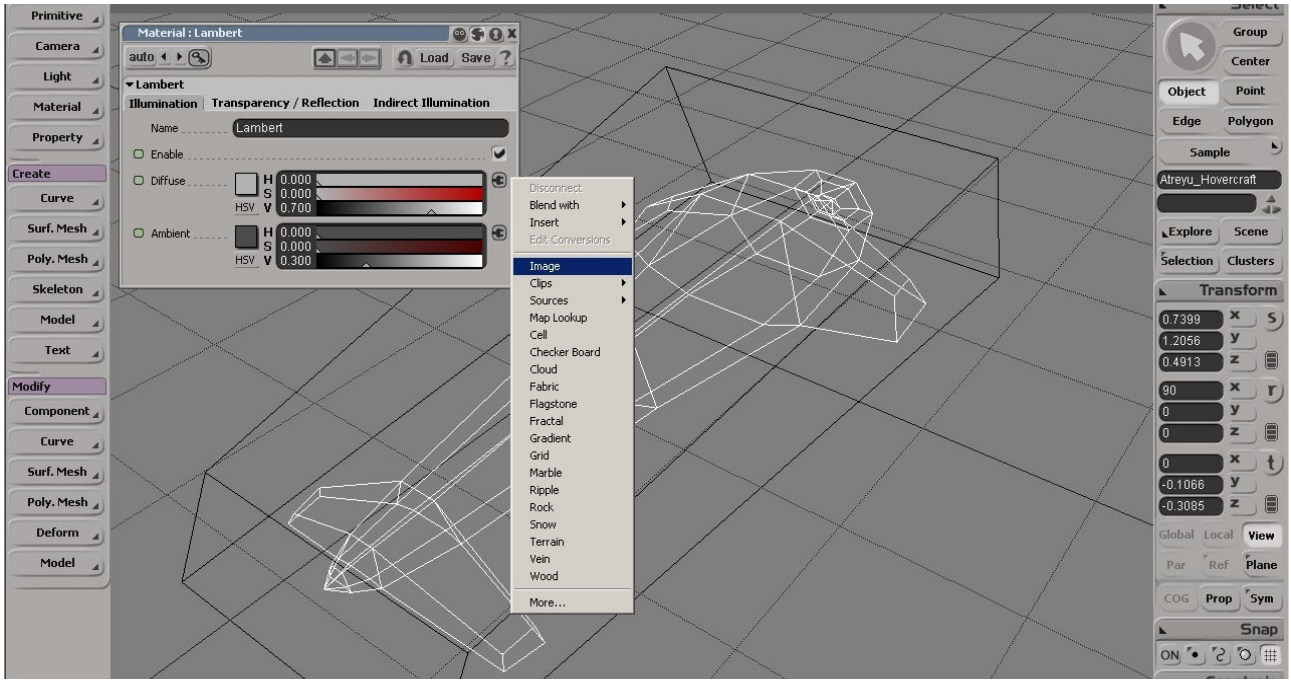
When you're done and happy with painting your hovercraft, save your work as .XCF before exporting out to .JPG. The reason because you might want to modify the colors. XCF is Gimp image format. Well attach this texture to the model in Softimage before exporting the model to OBJ format for Blender.

Attaching texture with the Hovercraft

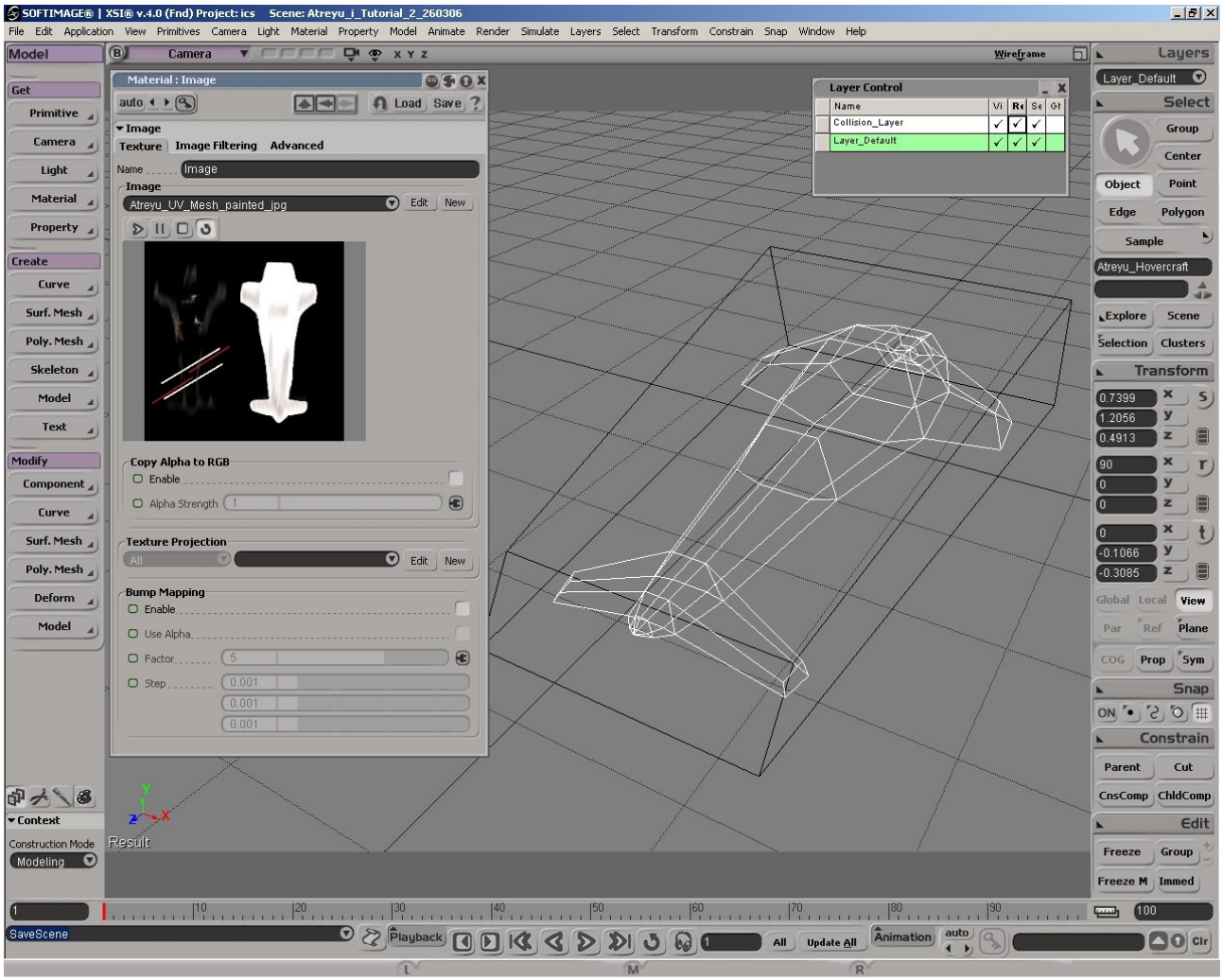
Back to XSI, Load your hovercraft which you did earlier. First we need to apply Lambert Material on it. Select the model, Click on **Material -> Lambert**.



A Lambert material PPG will pop up. Click on *Connect* icon next to Diffuse color, and click on **Image** to attach an Image to model.



Another PPG will appear. Set the Image to your texture file and leave out the projection option. See picture below.



You should be able to see the texture on the model (in low res) if the setting of the perspective viewport is set to **textured** or **texture decals**. If the collision box is getting in your way turn off the collision layer visibility.

To display the actual texture resolution on the object, do the following;

- Select Hovercraft, press 7 key to bring up Render tree window.
- There is a node connected to the left of Image node.
- Double click, a PPG will appear. Go to *Texturing* Tab, set OpenGL Texture Settings: Maximum width or Height to 1024 or 2048.

Save your work! The last step would be to export the model to OBJ format which Blender can import. We'll then later export from OBJ to Torque DTS format.

Have a break before we move to the next section to prepare this Hovercraft model for Torque Game Engine.

Section 5
- Importing and Exporting in Blender.

Here's blender. Blender is an open source software for 3D modeling, animation, rendering, post-production, interactive creation and playback. Available for all major operating systems under the GNU General Public License. I quoted this from the website. It does almost everything and it's free.

Blender uses left-hand coordinate system with vertical z axis. While on the other hand, Softimage uses right-hand coordinate system with vertical y axis. When you import an object built in XSI into blender the object orientation would be different. To rectify this, simply rotate the object by tapping R key towards +y direction. Use Shift key to do incremental rotation by 15°. These are the commonly used hot keys in Blender;

G	Translate tool. (G, X constrain movement in X axis. G, Y constrain movement in Y axis. G, Z constrain movement in Z axis. Alternatively use MMB click to constrain translation to a particular axis.
R	Rotate tool. (tap X, Y or Z after tapping R to constrain rotation to a particular axis)
S	Scale tool. (tap X, Y or Z after tapping S to constrain scaling to a particular axis)
N	Transform properties box.
LMB click on viewport	Set position of 3D cursor.
LMB drag on SRT manipulator icon	will translate, rotate or scale object.
MMB drag on viewport	Rotate view.
Ctrl + MMB drag (or mouse wheel)	Zoom view.
Shift + MMB drag	Pan view.
RMB	Select an Object or Sub-component (face, vertex and edge)
Numpad 7	Switch to top view.
Numpad 1	Switch to front view.
Numpad 3	Switch to side view.
Space bar	Pop up menu.

Blender Installation

Please download the latest version from www.gimp.org. Install it and please read "Blender Quickstart" pdf document before you proceed further.

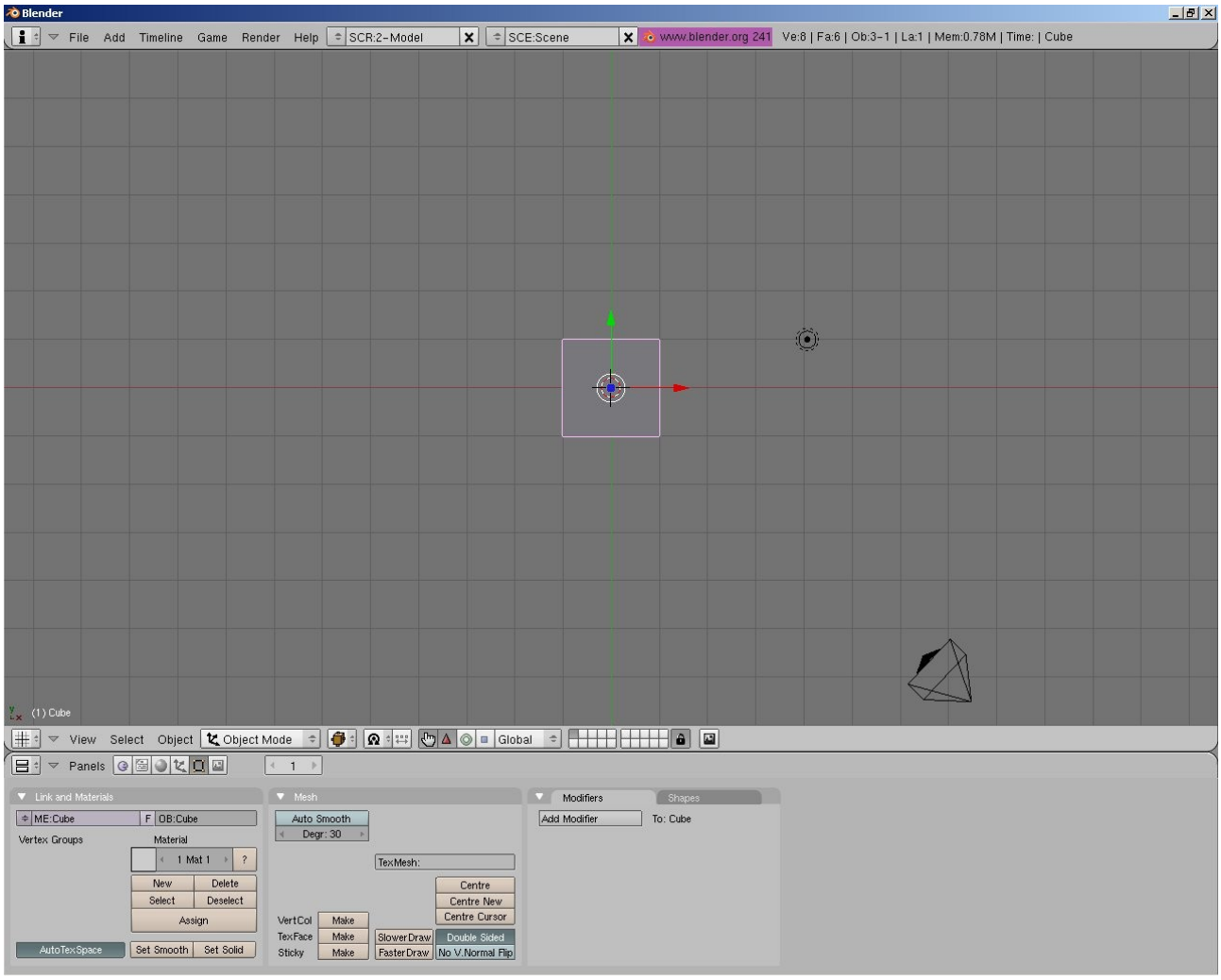
DTS Exporter Plug-in Installation

Get the latest of DTS Exporter for Blender from http://projects.blender.org/frs/?group_id=95
The installation of this DTS exporter is fairly easy. Unzip the files to a temporary folder, copy *DTSPython* folder, *Dts_Blender.py, Common_Gui.py, DtsShape_Blender.py and DtsMesh_Blender.py* to your blender scripts directory. The script folder would be in [*your blender installation folder*]\.blender\scripts.

When you did it correctly, under **File -> Export** menu, you would see *'Torque shape (.dts)'* listed.

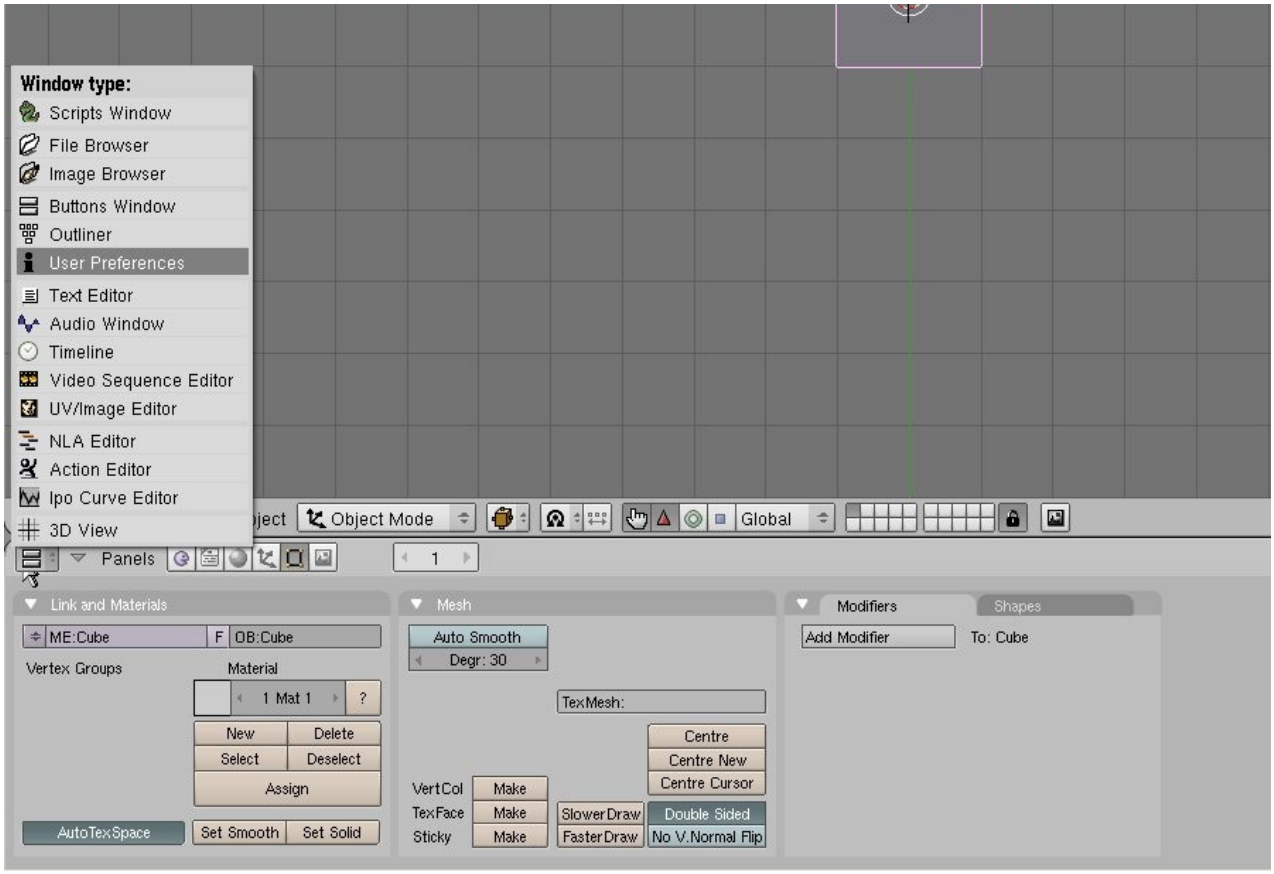
Setting up Blender

Launch Blender. By default, you will start with a cube, a lamp and a render camera, looking from top viewport. If you prefer to work in orbit mode, you may skip this part and continue on with importing object into Blender. Some people work extremely well in orbit mode, while I find it very irritating. If you don't like to work in orbit mode, read on.

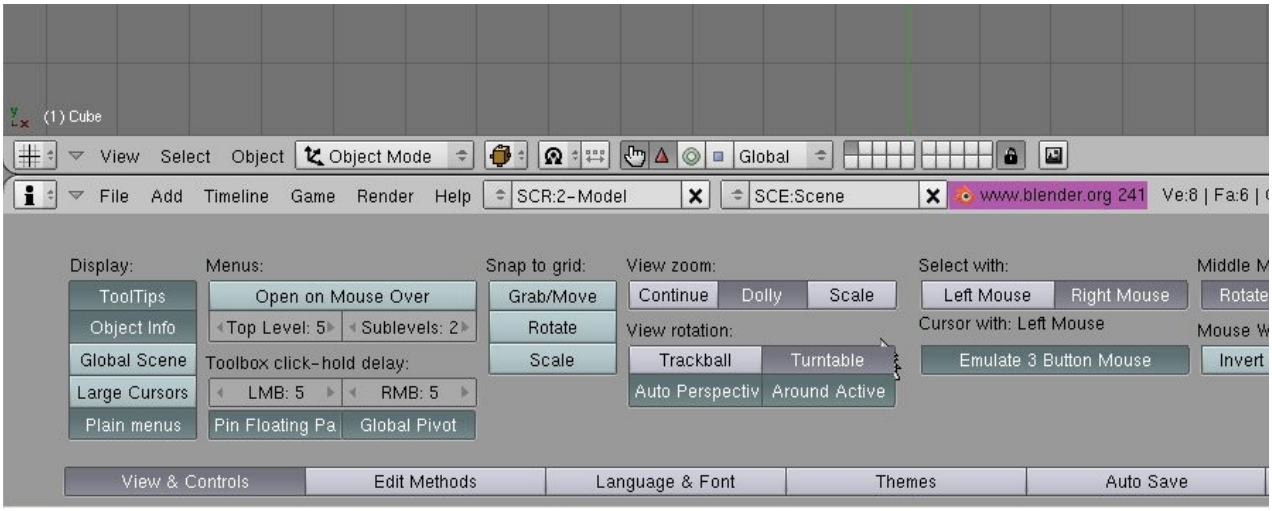


The default rotation mode in blender is Orbit mode, let's switch the rotation mode to turntable so that it behaves like XSI or Maya.

Switch the bottom window type to '*User Preferences*' window. This can be done by clicking the icon on the window toolbar on the far left, and click on '**User Preferences**'.



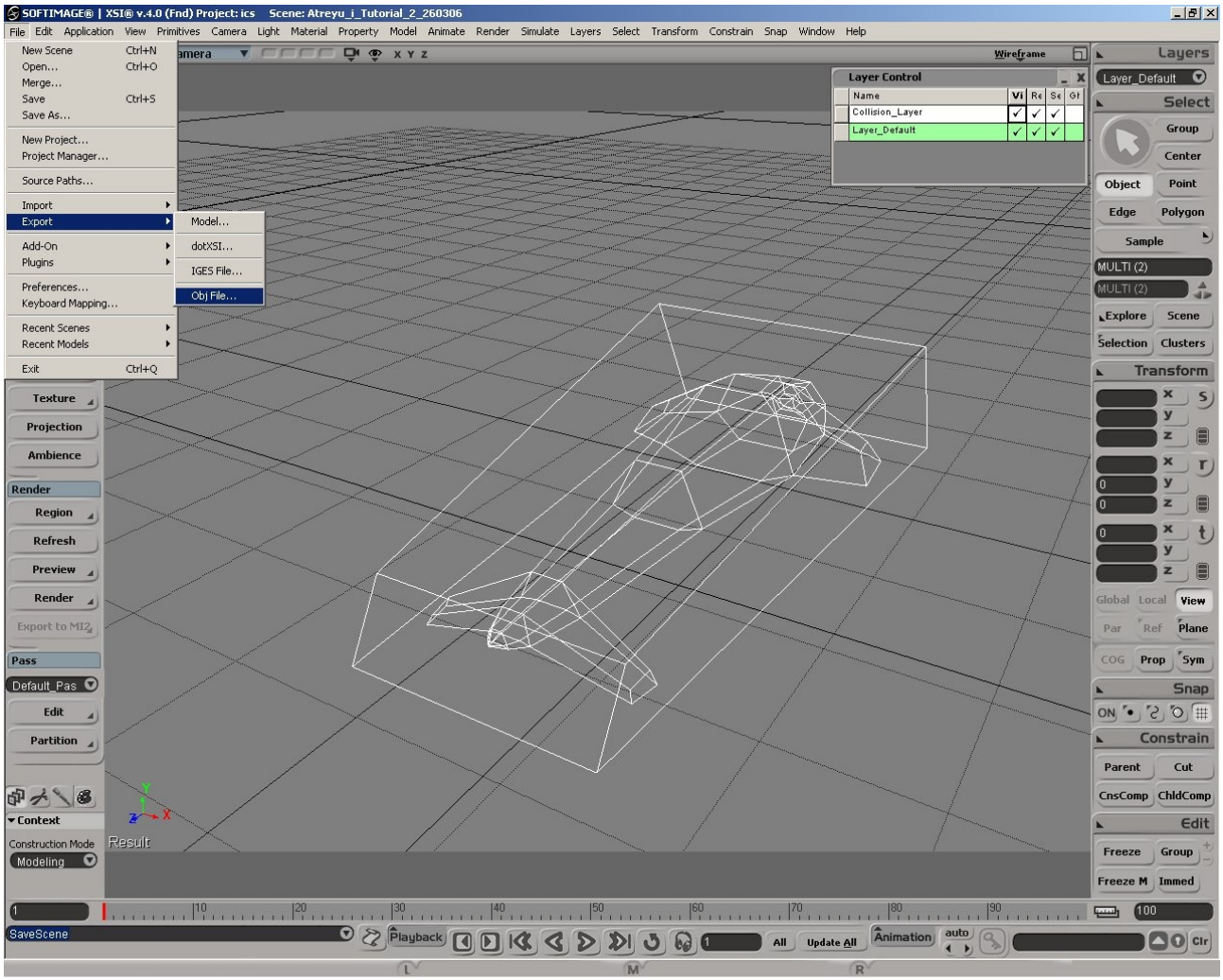
Click on '**View and Controls**' button to bring up option for view rotation setting. Click **Turntable**, **Auto Perspective** and **Around Active** buttons.



We're done. Save your setting with **File -> Save Default Settings** or hit Ctrl + U. When you restart Blender you will have the turntable mode activated instead of orbit mode.

Exporting Hovercraft from XSI to .OBJ format

Select the collision object and the the hovercraft, click **File -> Export -> Obj File...** Specify a folder and a filename with .obj extension and click OK to export.

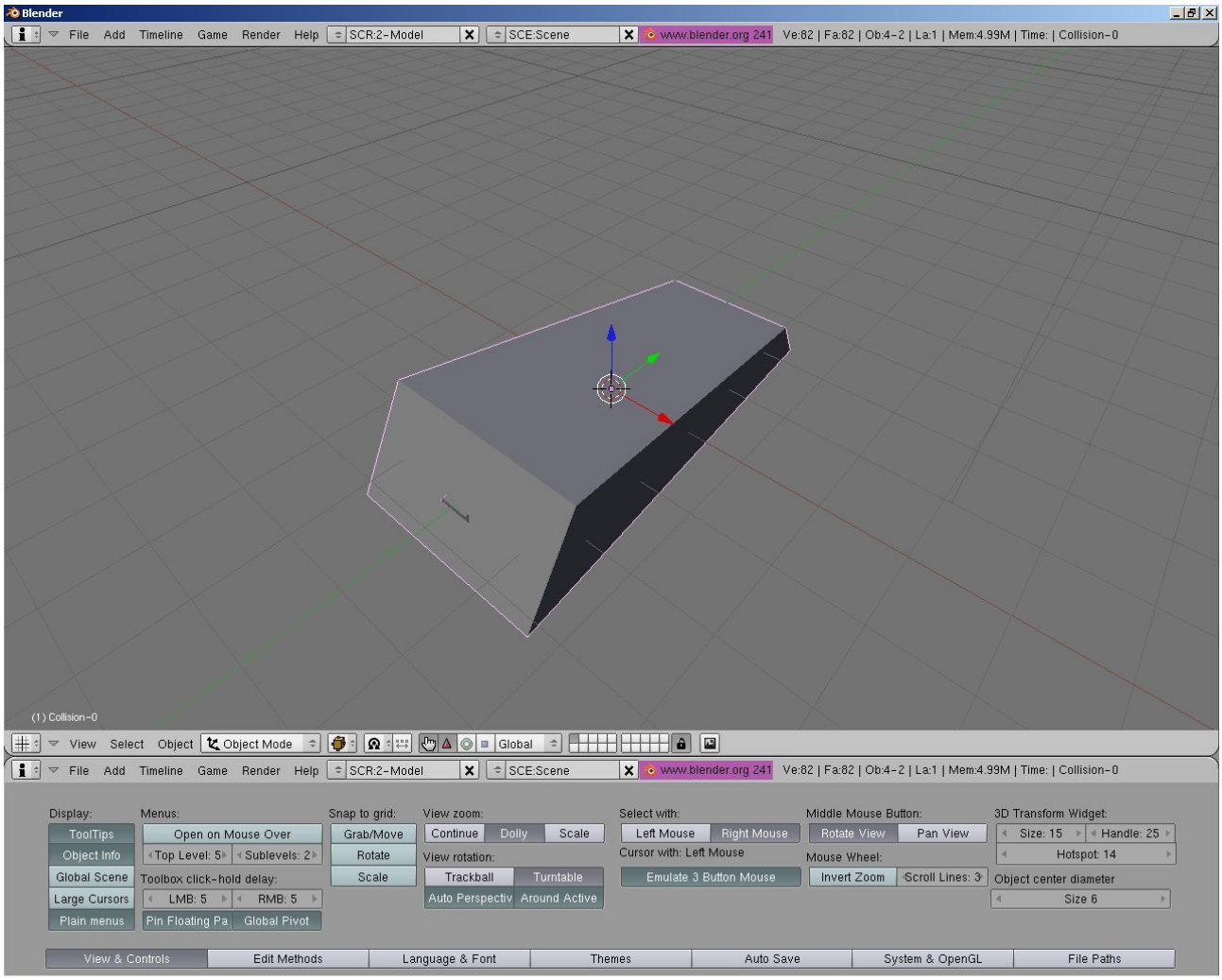


Save your scene, we're leaving XSI. Next part onwards we;ll be using blender.

Importing Object

First of all, launch your Blender and delete the cube from the scene. Click on **File -> Import -> Wavefront (.obj)...** to import the hovercraft which we did earlier.

Rotate the hovercraft and the collision box so that both face +y direction. Rotate around Z axis from top view and rotate along X axis from side view.



Before we export these objects to DTS there are a couple of things we need to do. Firstly we need to organise the object in a way which the exporter could understand and we need to define camera points for first and third person view.

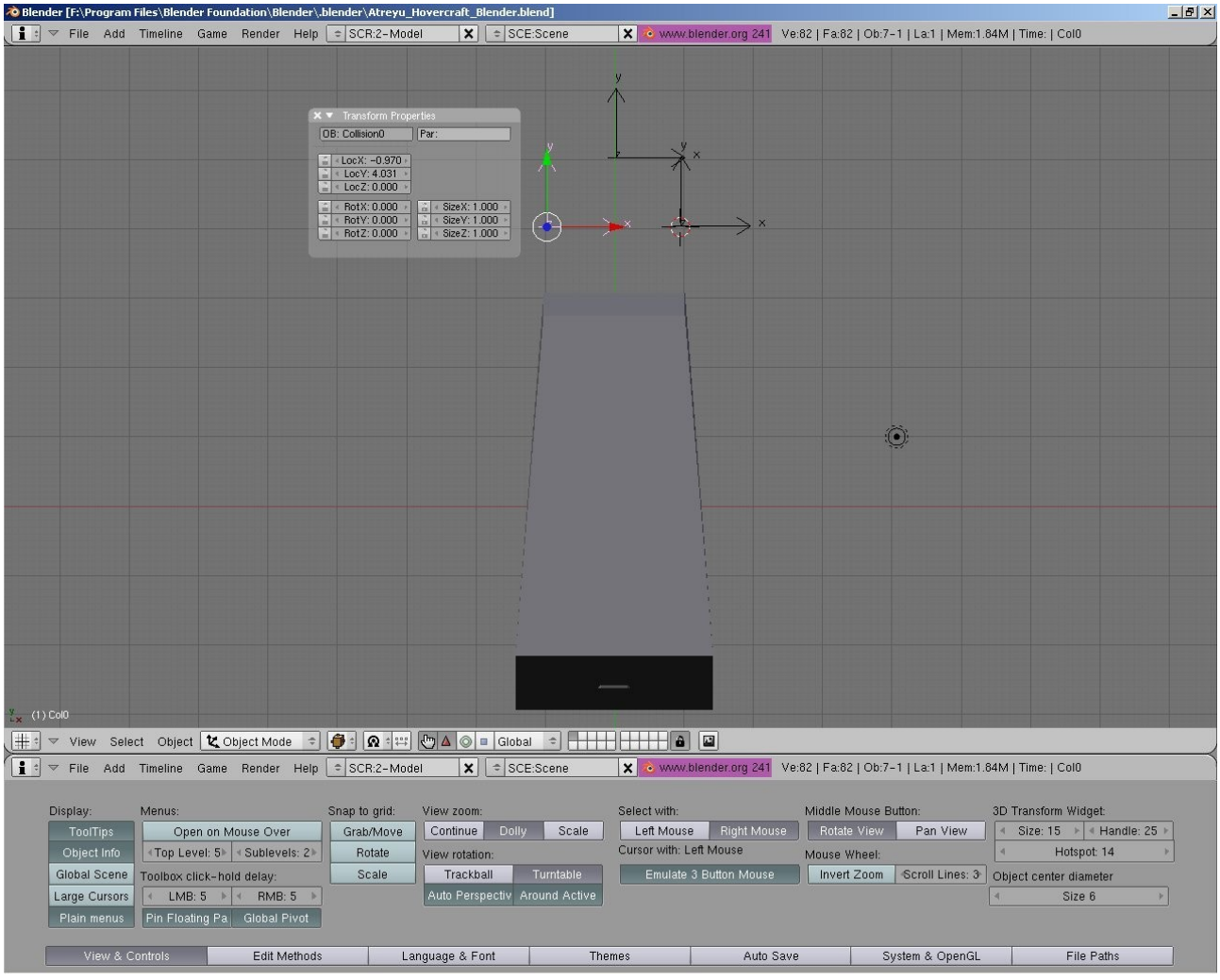
From the top view we'll create 3 empties. An empty is equal to a null in XSI. Where you place them is not an important as long as the hovercraft is facing in +y direction of the empties.

Follow these steps to create empties;

- Move the 3D cursor to a space in front of the models with LMB click.
- Hold Space bar to bring up add menu, click on **Add -> Empty**.
- While the new empty is still highlighted, tap N key to bring up Transformation property box and name it *Shape*.
- Repeat the process to create 2 more empties (see picture below for reference).

Name the 1st empty as *Shape*, the 2nd one as *Collision0*, and the third one as *Detail32*.

Proper naming is very important here because this is a way of telling the exporter which object is which.

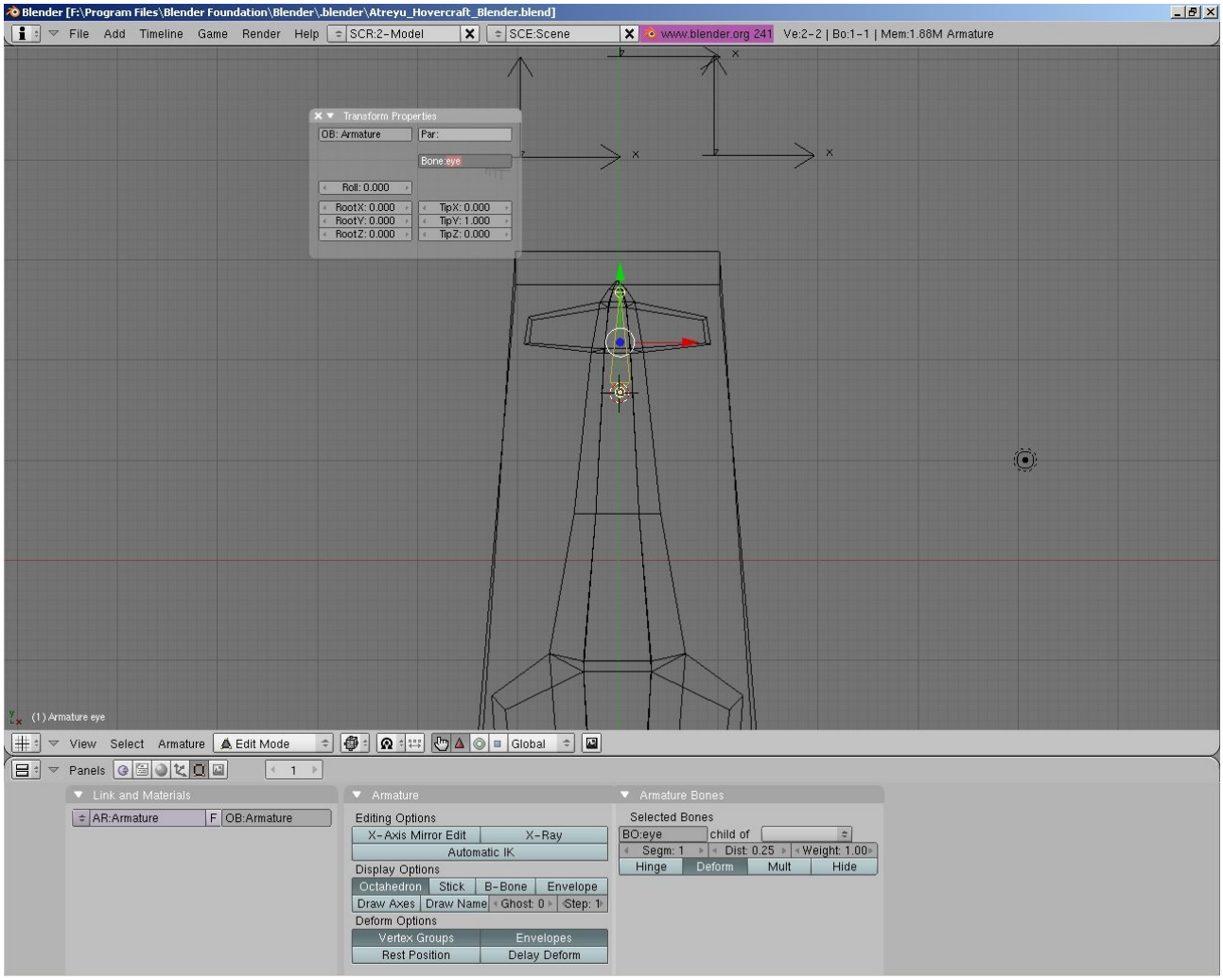


Next, we'll create 2 armatures (bones) from the top view. This is a way of telling the exporter where the camera points are.

Follow these steps to create armatures for camera points (refer to the pictures below);

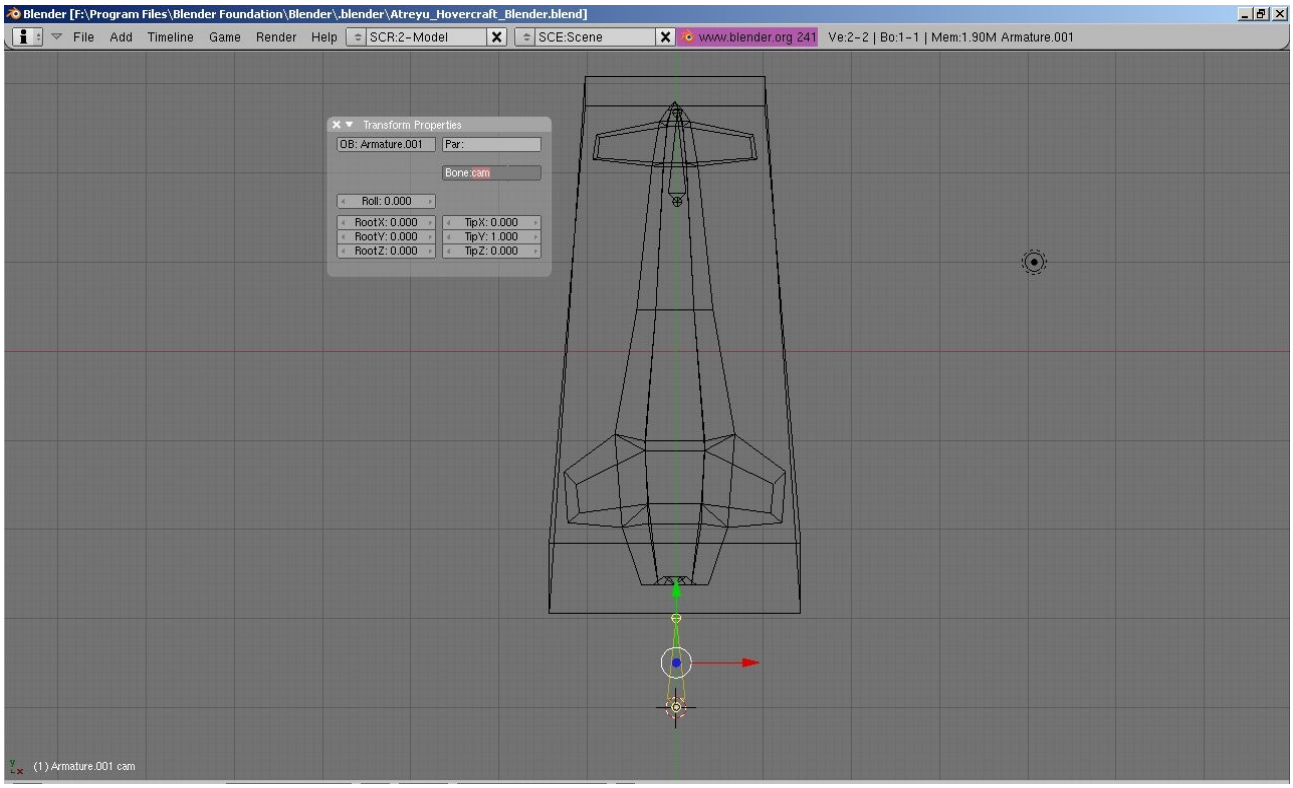
- Move the 3D cursor to a space in front of the models with LMB click.
- Hold Space bar to bring up add menu, click on **Add -> Armature**.
- The Joint will be highlighted. Select the actual bone with RMB click, tap N key to bring up Transformation property box and name the bone as *eye*.
- Hit Tab to exit edit mode to object mode.
- Repeat the process to create another armature at the back of the hovercraft.

Name the 1st bone as eye and the 2nd one as *cam*.

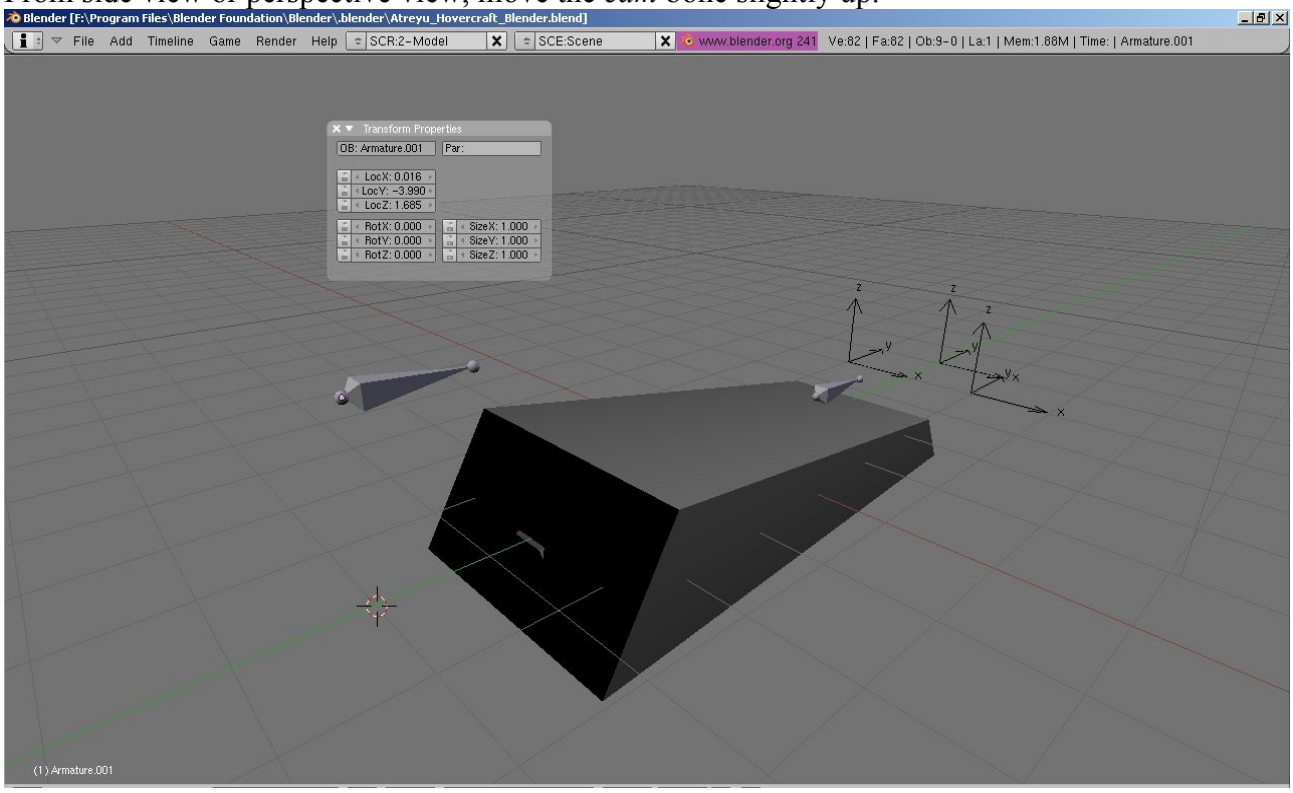


Eye -> placement point for first person view. Cam -> placements point for third person view.

Place the second armature somewhere down the back of the hovercraft. Don't forget to name the bone as *cam*.



From side view or perspective view, move the *cam* bone slightly up.

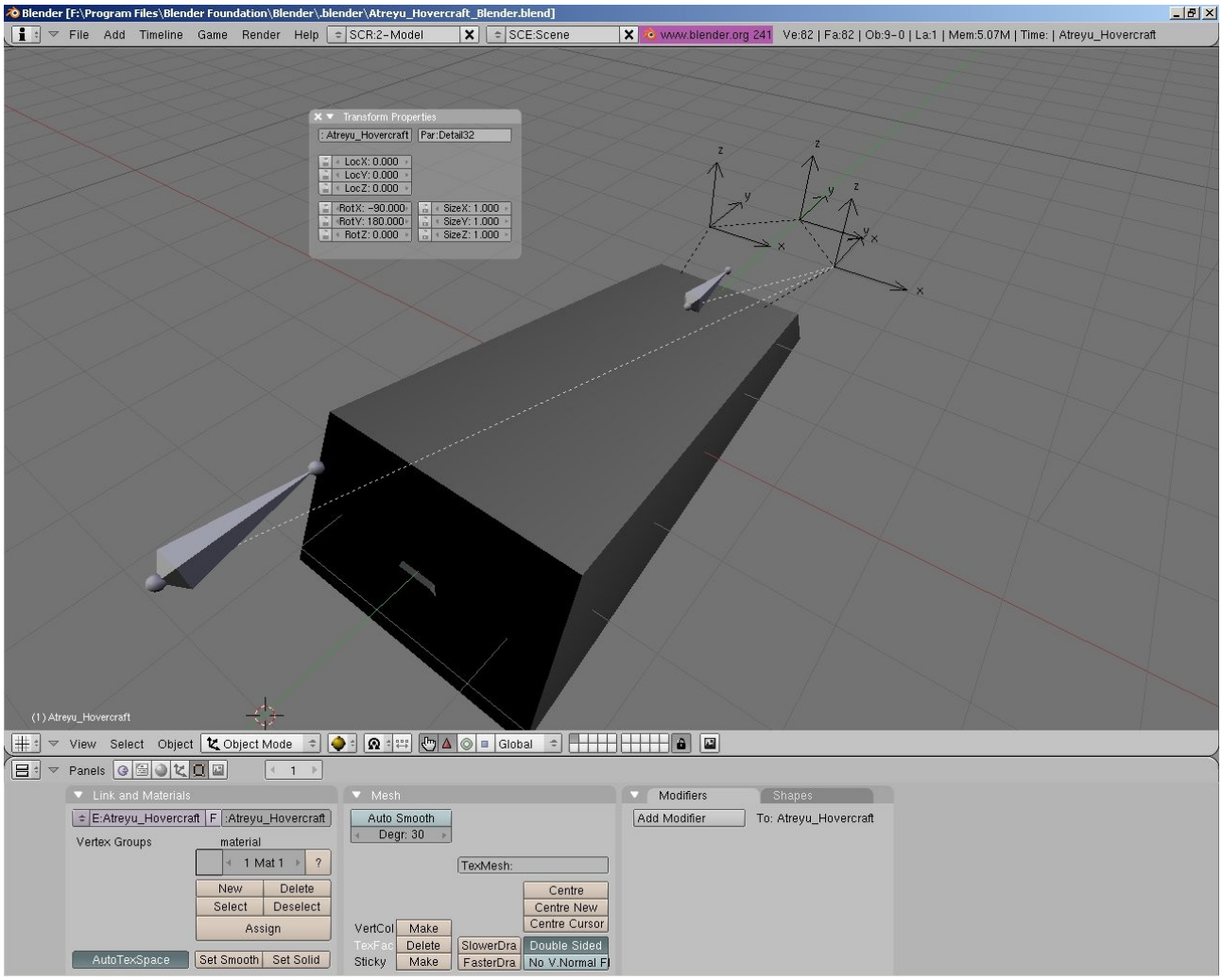


So at this point, everything is has been laid out. Before we export this model to DTS we need to do some parenting. To do parenting, first pick the child object, Shift + select the parent object, hit Ctrl +P (with 2 objects selected), and click Make Parent. (Note: This doesn't work in real life!!!)

Whenever you're ready, do the followings;

- Parent Collision-0 box to *Collision0* empty.
- Parent *eye* bone to *Detail32* empty.
- Parent *cam* bone to *Detail32* empty.
- Parent Hovercraft model to *Detail3*2 empty.
- Parent *Detail32* to *Shap*e empty.
- Parent *Collision0* to *Shape* empty.

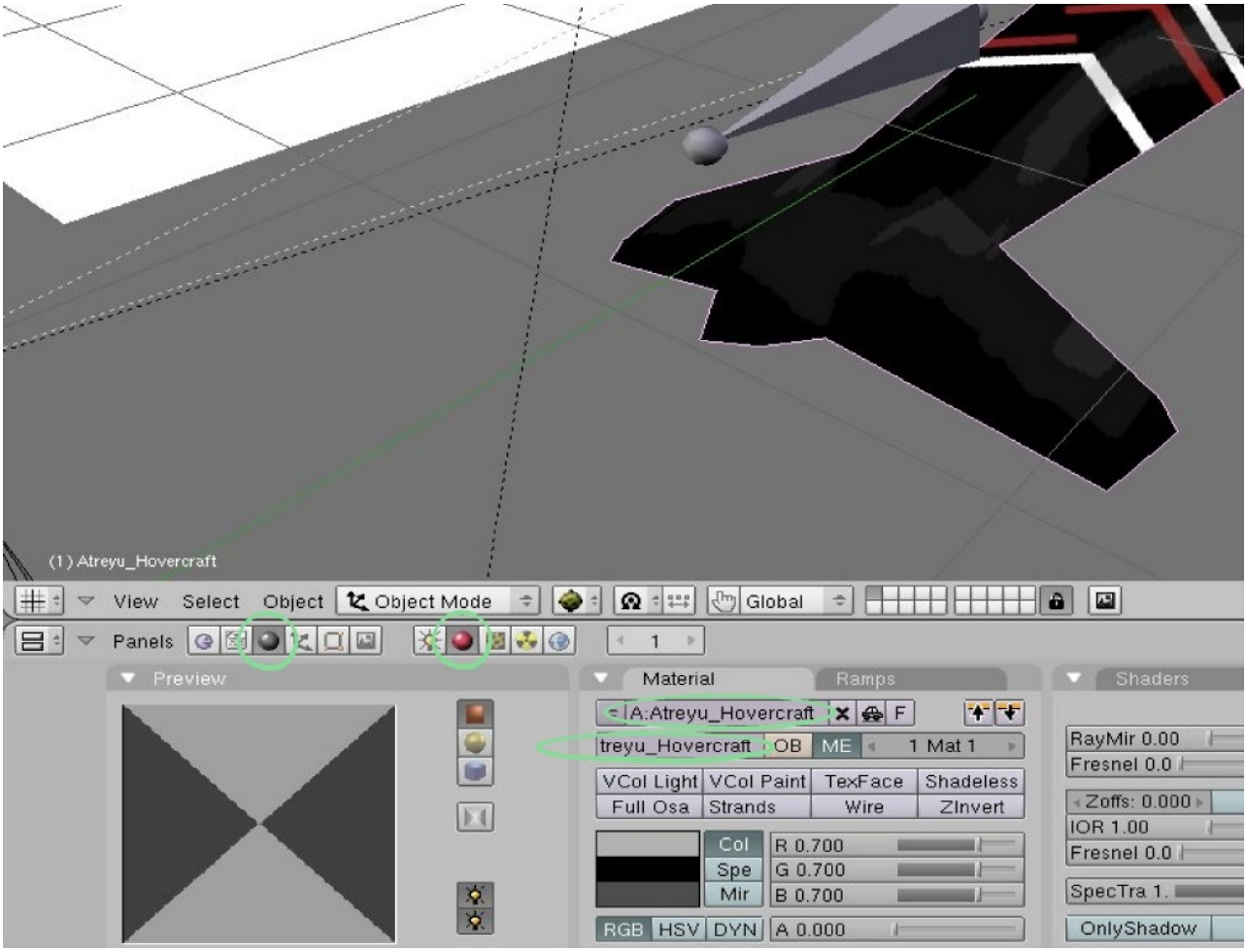
Once you've done the parenting you'll see a dotted lines between objects in the scene.



Save your Blender scene. Now we are almost ready to export the objects into Torque's DTS format. The last step would be to name the texture material name so that texture material name = object name. Lastly when we export we must make sure the .DTS filename = texture material name = object name.

In the picture below I moved away the collision box temporarily, so that I could see the texture on the hovercraft. Change the bottom window type to *buttons window* by moving the mouse over the window and hit Shift+F7, hit F5 to enable the shading window and click on material button.

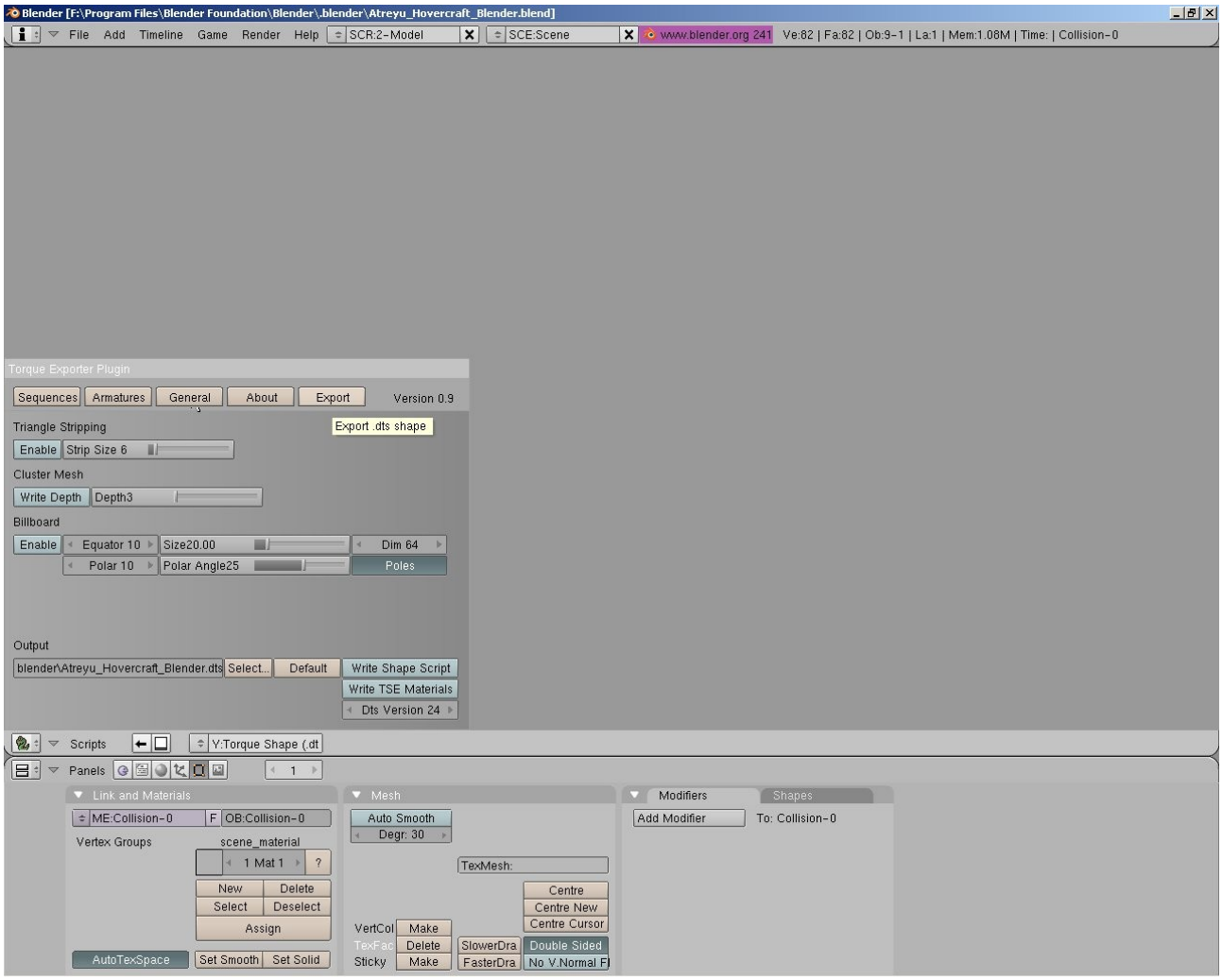
Change the material name to match with the object name.



Note on working with hot keys

To increase productivity in modeling or animation or even texturing it is recommended to work with one hand on keyboard and the other on mouse. Using hot keys significantly increases your speed, and as you become more proficient, you're likely to use hot keys rather than locating the buttons on the user interface.

Click on **File -> Export -> Torque Shape (.dts)...** A dialogue box will open. Under **General** tab, you can specify where the .DTS file will be written. Change the output file name to reflect the object name and the material name. Leave the other settings the settings and click **EXPORT**.
By default, the DTS file will be written in <Root>\Blender Foundation\Blender\.blender\



Note: The Exporter will export 1 unit length in blender as 1 unit length in Torque which is equivalent to 1 meter.

Proper naming is very important here because this is a way of telling the exporter which object is which. Always check readme.html that comes with the latest DTS exporter to see any update or changes in the naming convention.

Save the DTS file and the texture file in one folder (they must stay in the same folder), for now keep them in your own working folder. Rename the texture file to match with the name of .DTS file.

We're done with blender. Next stop we'll be working on Torque scripting to accommodate this hovercraft in Torque Game Engine.

Section 6
- Torque Scripting

Now we have the model and the texture done, what we are going to do in this section is to make the Hovercraft playable in Torque Game Engine.

Putting 3D model into Torque's folder
Create a '*Hovercraft*' folder under...starter.racing\data\shape\. Copy the DTS file and the texture in this '*Hovercraft*' folder.

Creating a Datablock for The Hovercraft
A datablock define the physical properties of an object in Torque game environment. Get into ...starter.racing\server\scripts\, then create a file for Hovercraft's datablock, in this example I created a file called Hovercraft.cs

Type these following lines in Hovercraft.cs;

```
//--------------------------------------------------------------------------------------------------------------------
datablock HoverVehicleData(Atreyu)
{
//Specify category and Model
catagory = "Vehicles";
shapeFile = "~/data/shapes/Hovercraft/Atreyu_Hovercraft.dts";
emap = true;

//Camera setup for 3rd person view of Hovercraft
cameraRoll = true;              //This setting enables the camera to roll when the vehicle rolls
cameraMaxDist = 4.0;            //Camera Max Distance from vehicle
cameraOffset = 1.5;             //vertical offset of camera
cameraLag = 0.04;               //Velocity lag of camera
cameraDecay = 0.85;             //Decay per sec. rate of velocity lag

//Physics Properties of Hovercraft
integration = 4;                //Physics integration: TickSec/Rate
collisionTol = 0.1;             //Collision distance tolerance
contactTol = 0.1;               //Contact velocity tolerance
mass = 0.70;                    //Simulated Mass of the Vehicle
floatingGravMag = 5.5;                  //Gravity applied when the Hover-vehicle is airborne
bodyFriction = 0.05;
bodyRestitution = 0.9;
dragForce = 3.0;                //Constant Drag force for the vehicle
floatingThrustFactor = 0.8;     //Scalar applied to Thrust force while floating
mainThrustForce = 180;                  //Thrust power vehicle
reverseThrustForce = 0.12;
strafeThrustForce = 1;
brakingForce = 10;              //Force generated by braking
brakingActivationSpeed = 50;
stabLenMin = 1.5;               //min Dist of vehicle from ground when traveling forward??
stabLenMax = 2.8;               //max dist vehicle from the ground when traveling forward??
stabSpringConstant = 50;
stabDampingConstant = 30;
```

```
gyroDrag = 15;          //Drag which is scaled against vehicle angular momentum
normalForce = 50;
steeringForce = 10;             //Force applied to steering, higher number spins faster
rollForce = 4;          //Force applied to roll the vehicle, smaller number harder to roll
pitchForce = 4;         //Force applied to pitch...
};
//--------------------------------------------------------------------------------------------------------------------------
```

for more detailed HoverVehicleData parameters please check garagames's website. Lastly, we need to modify game.cs under server folder. We want to modify the existing code to load our Hovercraft datablock.
So, open up game.cs, scroll down to find...

GameConnection::createCar(%this, %spawnPoint)...
..
Change...

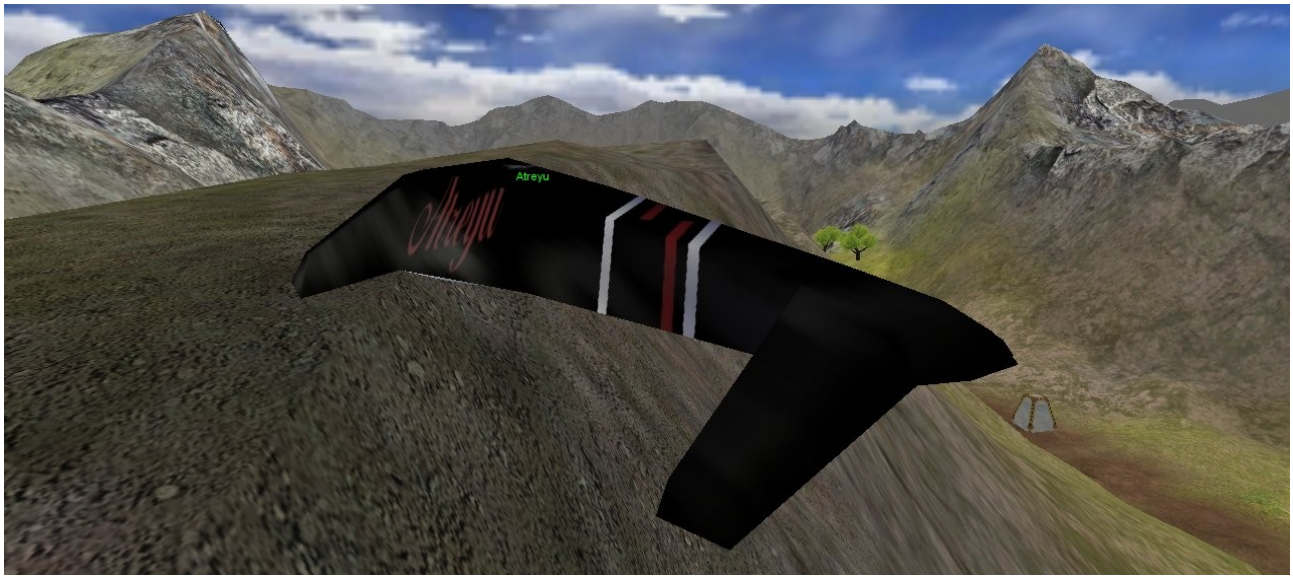
```
// Create the player object
   %car = new wheeledVehicle() {
      dataBlock = DefaultCar;
      client = %this;
   };
```

To...

```
// Create the player object
   %car = new HoverVehicle() {
      dataBlock = Atreyu;
      client = %this;
   };
```

Also, at the beginning of game.cs, find; exec("./car.cs"); and
replace it with; exec("./Hovercraft.cs");

That's all, save your files, run your Torque Game Engine and enjoy the game.



Section 7
- Adding more checkpoints

(Adding more checkpoints into the starter_racing kit for TGE v1.3/1.4)

Some of us would like to have a huge racing track with many checkpoints to avoid players taking shortcuts (and also making sure that the player would go to a certain predestined path; a molten lava, a deep-dark-thick forest or a huge jump between canyons (well... that's a great idea).

The Starter_Racing kit that comes with Torque Game Engine is already loaded with 4 checkpoints. In this section, I'll show how to add another checkpoint in the starter_racing kit.

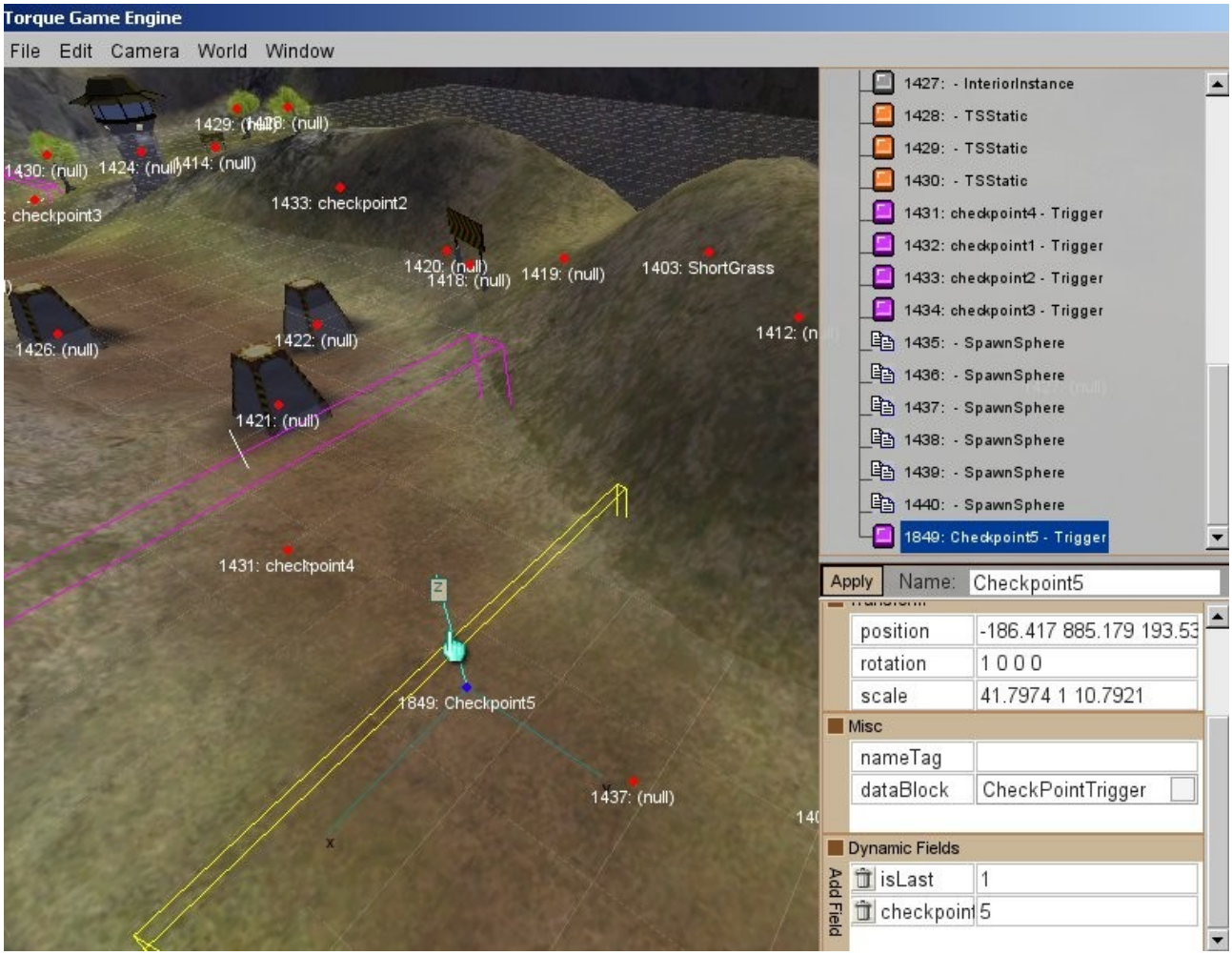
- Launch TGE with starter_Racing, when the game starts, enter camera fly mode by hitting Alt+c. In camera fly mode, use W, S, A, D and RMB to move around.
- Hit F11 [Mission Editor],
- Hit F4 [World Editor Creator], The Creator Frame is Located at the Lower right Corner of the Window
- In the Creator frame, click [+] to expand *Mission Objects* tree, Click on Mission and Click Trigger.

- You will be given a new Trigger properties Window
  -Enter *Object Name* as **Checkpoint5**
  -*datablock* = <pick> **CheckPointTrigger**
  -Then click OK

  We need a trigger in game world if we want TGE to detect if a player enters or leaves a certain area.

- Translate (LMB on axis), Rotate (Alt + LMB on axis) and Scale (Ctrl + Alt + LMB on axis) the new CheckPointTrigger to a new position after checkpoint4.
- Switch to Window-World Editor Inspector by hitting F3
- Under MissionGroup or on the game window , Click on checkpoint4 to select it..
  On the dynamic field (lower right screen) **delete** entry "*isLast = 1*" by clicking the bin icon to the left of "isLast", then Click **Apply** button.

- Click on the new checkpoint5 trigger under MissionGroup window or on the game window.
  Add 2 entries in the dynamic field for this trigger, by clicking **add field**.
  - (dynamic field entry) **checkpoint** (with value of) **5**
  - (dynamic field entry) **isLast** (with value of) **1**



Always click **Apply** button to update the information for this checkpoint trigger.
These two properties; "*checkpoint*" and "*isLast*", are accessed within the server side of program code, checkpoint.cs.

- Save your map and done!
- Re-Launch TGE, after you passed those 5 checkpoints the lap counter will increase by 1.

Ken Finney has explained about this triggering-events nicely in his book "3D game programming All in one". If you're really interested in Torque, go get this book.

Section 8
- Loading up a heightmap into Terrain Editor

Basically a heightmap file is a black and white fractal/noise-like image which can be obtained from fractal, noise generator or DEM(Digital Elevation Model)/SDTS (Spatial Data Tansfer Standards) of real-geographical site. This heightmap later can be used to generate a terrain based on the color level. Black represents the lowest part of terrain and white represents the highest point in terrain.

If you need to get a heightmap of a real geographical site on the world, goto www.geocomm.com or other earth mapping websites. Some of them will provide free download of certain SDTS or DEMs. There are also free tools to convert these formats to heightmap file. I'm not going to give another 30 pages explanation here regarding SDTS and DEM, so head to www.geocomm.com for more information. This SDTS/DEM formatting and standards stuff are pretty complicated.

There is also a free and good terrain generator program called L3DT (http://www.bundysoft.com/L3DT/) which allows you to create a terrain with lots of options (details). This program allows you to export the terrain to a heightmap file.

Before loading your heightmap to TGE, resize it to 256 x 256 pixel first.

These are the steps to load a heightmap into a new mission;

- Launch starter.fps, when game starts switch to flying camera mode by pressing Alt+C
- Press F11 to get into World Editor
- Goto File-New Mission to create a new World
- Press F7 to get into Terraform Editor
- Click the Box next to Operation and Choose Bitmap. Locate your Heightmap file (which is in PNG) and click Load and Click Apply. You should now see the new terrain based on the heightmap we prepared earlier.

If you don't like the way it looks, simply sculpt the terrain or quickly generate another one.

A quick note on Torque Mission Editor

Before saving a **new** mission file (a new map), go to World Editor Inspector by pressing F3
Click MissionInfo under MissionGroup-SimGroup
Give the Mission Map a proper name and a proper description under the Dynamic Fields.
Save the New Map Mission under \starter.racing\data\missions\

Section 9
- Applying Textures to Terrain

This is another short section. Actually, placing textures on a terrain is very straightforward. TGE has a powerful built-in texture painter to place texture on the terrain.
There are 2 ways of doing this; you can Paint directly the texture on the Terrain or using functions to distribute the textures on the terrain.

Terrain Texture Editor

- If you're still in World Editor, press F8 to open up Terrain Texture Editor.
- Click Add Material, and load all the texture that you need to use for the Terrain.
- Once you loaded all the textures up, you will then need to specify how the texture to be placed on the terrain. Highlight the texture filename then click on Placement Operation then play around with some settings and click Apply when you're done.
- Under Placement Operation, there are 4 ways to place texture on the terrain.
  - Place by Fractal
  - Place by Slope
  - Place by Height
  - Place by WaterLevel

Don't forget to click Apply button every time you made changes.

Terrain Texture Painter

This one is very straight-forward. simply load all the textures that you need to use and paint it directly on the Terrain. You may change the brush effect as you paint.

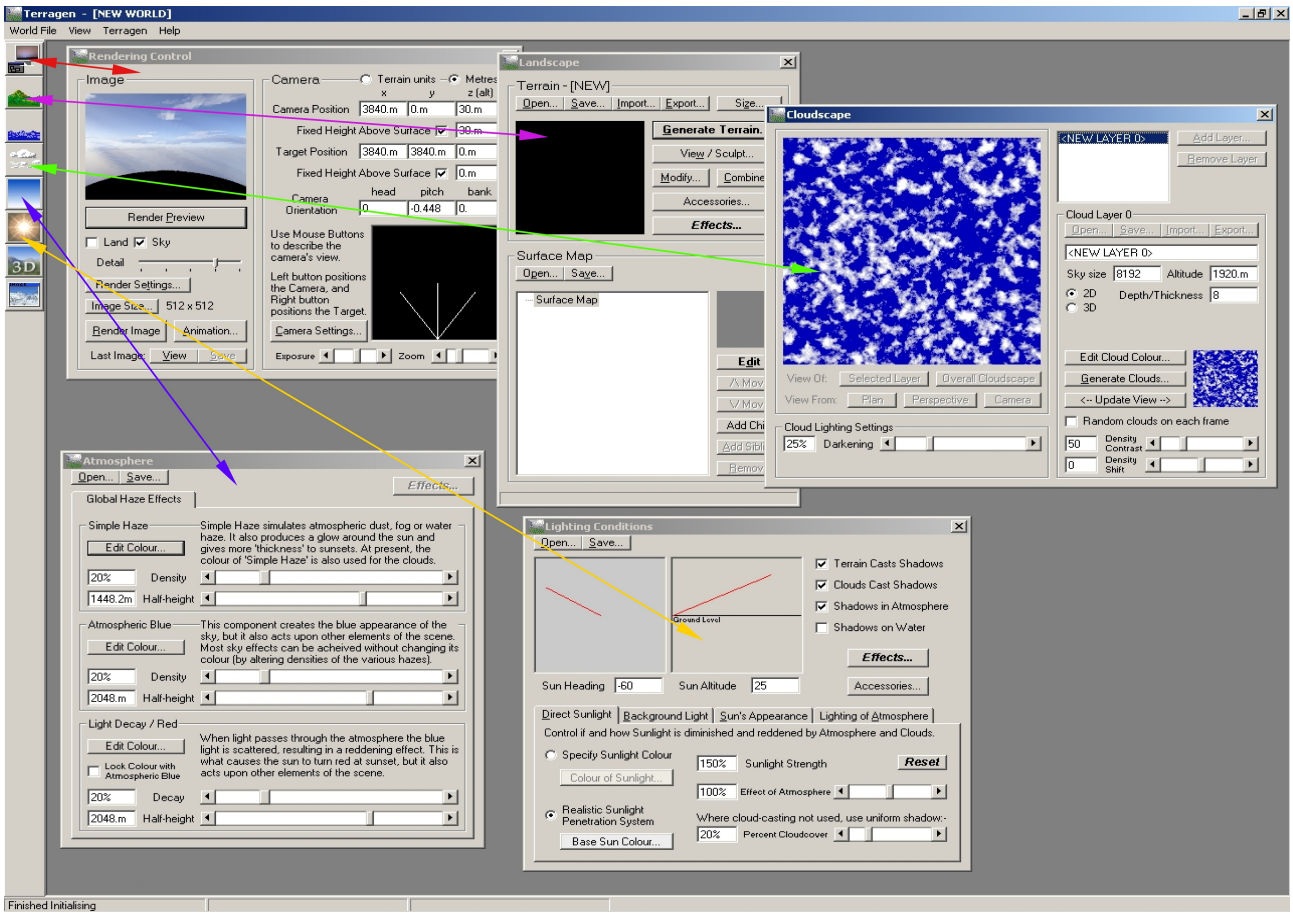
A quick note on Torque Mission Editor

Before saving a **new** mission file (a new map), go to World Editor Inspector by pressing F3
Click MissionInfo under MissionGroup-SimGroup
Give the Mission Map a proper name and a proper description under the Dynamic Fields.
Save the New Map Mission under \starter.racing\data\missions\

Section 10
- Skyboxes with Terragen.

If you do have experience with Bryce, MojoWorld or Vue, go ahead and use your favorite program to create a masterpiece. I will describe steps that need to be taken to create a basic sky and prepare it for Torque Game Engine. I'm not going to give detail explanations about Terragen's bit and pieces in this tutorial, but feel free to check the website.

Have a go with Terragen to get familiar with the interface, before doing this section.



Creating images for skybox in Terragen

To create a skybox using Terragen, follow these steps;

- Under Rendering Option window - uncheck the Land checkbox (unless if you're making skybox with land on it).
- Click on **Camera Setting** button and change the *zoom* value to *1*, which equal to FOV 90.
- Increase the preview detail to max so you could get better visual of how the sky looks like.
- Click on **Render settings** button; set *accuracy* for Atmosphere and Cloud Shading to *max*, and set image render size to *512* x *512*.
- Open up Cloud Window and create one of your own. Set the Sky Size, altitude and Thickness to your choice. Always press **Render Preview** button to see the changes that you made.

- Open up Atmosphere window and create one to your liking. Again, always press **Render Preview** button to see the changes that you made.
- Open up Lighting Window (Sun Icon), set the position and heading for your sun. Take note of this because we'll need to match the sun position of the skybox with the sun positions in the game. We need to subtract 90 from Terragen's Heading value to be used in Torque. Otherwise, the lighting won't match.

  e.g. Terragen
  heading: -60 and altitude: 25

  in Torque (via world editor) this translates to
  heading: (360-60-90) 220 and altitude: 25

  The direction of 0 heading is different in both programs.
  You may enable *Cloud Cast Shadow* option to greatly enhance the visual look at the cost of rendering time.

- Do another render preview, make changes as you wish before we do the final render. Use **Render Image** button to render your sky at 512 x 512.

- Point camera at Head: 0 Pitch: 0 - Render, save as <namefile for front image>.bmp
- Point camera at Head: 90 Pitch: 0 - Render, save as <namefile for right image>.bmp
- Point camera at Head: 180 Pitch: 0 - Render, save as <namefile for back image>.bmp
- Point camera at Head: 270 Pitch: 0 - Render, save as <namefile for left image>.bmp
- Point camera at Head: 0 Pitch: 90 - Render, save as <namefile for up image>.bmp
- Point camera at Head: 0 Pitch: -90 - Render, save as <namefile for bottom image.bmp>

Use Gimp image editor to convert them all to .jpg format, and store them under \starter.racing\data\skies\ folder.

Importing sky images into Torque

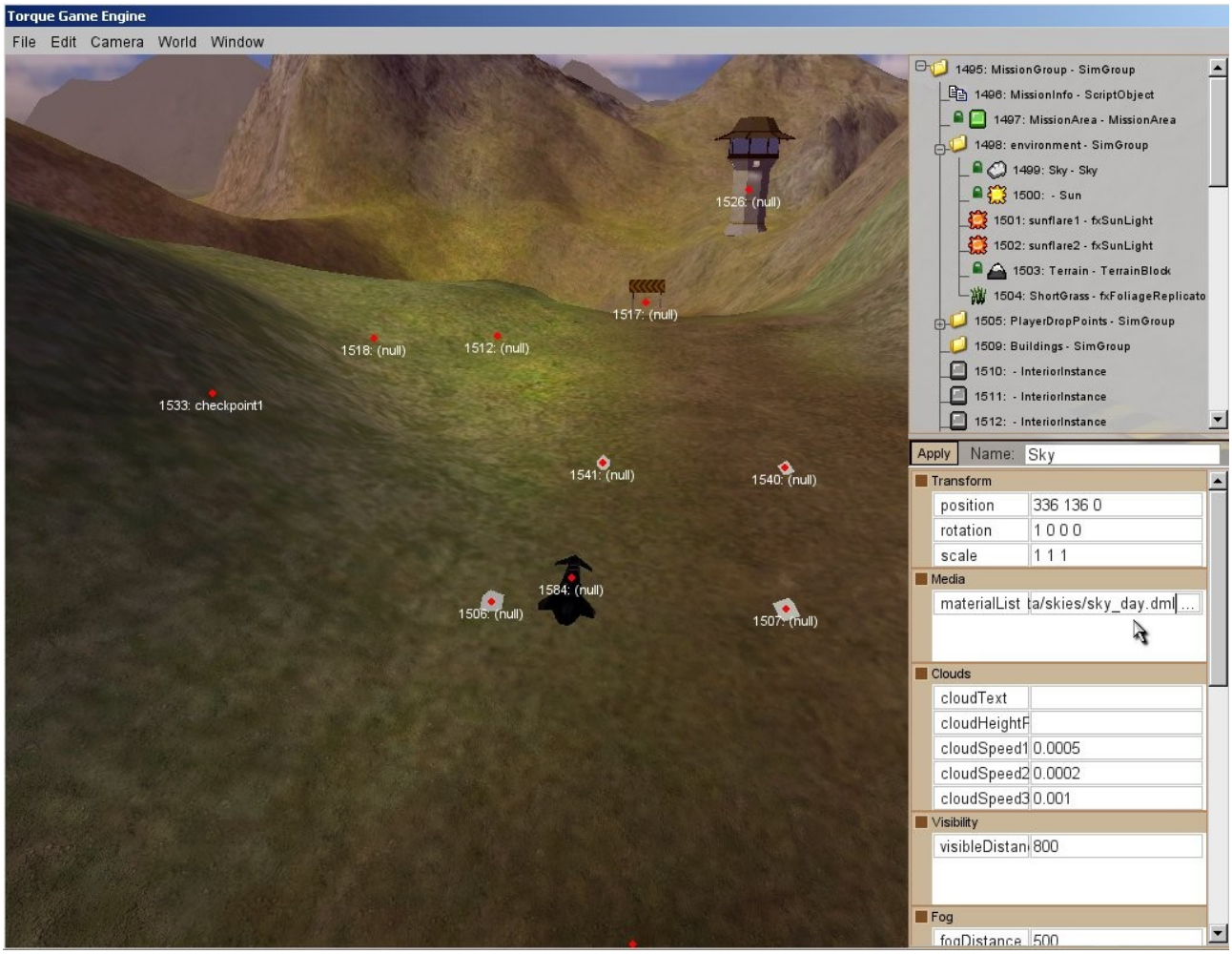
Skybox definition for starter.racing is stored in \starter.racing\data\skies\sky_day.dml. This .dml file may contains up to 10 lines;

- Line 1 to 4 - refer to your image of front, right, back, left
- Line 5 - refers to top image of skybox
- Line 6 - refers to bottom image of skybox
- Line 7 - refers to image used for reflections
- Line 8 to 10 - refers to layers of clouds

One thing to remember though, the .jpg or .png files that listed in a .dml file must be in the same directory of .dml file.

To Change the Skybox with your own, press F11 while in game to get into World Editor Inspector. Select *Sky* node under *MissionGroup – SimGroup -> environment – SimGroup* tree. Under *Media*, change the value of *materialList* from sky_day.dml to your own (.mdl) skybox. See picture on next page for reference.

Save the Mission Map, and you're done.



A quick note on Torque Mission Editor

Before saving a **new** mission file (a new map), go to World Editor Inspector by pressing F3
Click MissionInfo under MissionGroup-SimGroup
Give the Mission Map a proper name and a proper description under the Dynamic Fields.
Save the New Map Mission under \starter.racing\data\missions\

Troubleshooting

Torque loaded up and crashed before gameplay!
So here's the thing, torque would still load up even there's an error in one of the scripts, and sometimes this leads to unwanted crashes. Always check the console log if you made any changes and no changes took place within the game.

Things to look at;
- Have the Nodes for camera placement been placed into the Model before exporting to DTS?
- Have you defined the collision box?
- Check you code, look for missing (;) or other anomalies, then save it.
- Try to delete Hovercraft.cs.dso, and re-launch the game. Hovercraft.cs.dso is the compiled version of Hovercraft.cs, by deleting .cs.dso we are asking or forcing Torque to recompile it again.

My Hovercraft has no texture!
Make sure DTS file and it's texture are in the same folder. And they both have the same filename.

My Hovercraft has gaps in it!
When you did the modeling perhaps some of the faces were facing inside. Load up Softimage or Blender, make sure all the Normals of the polygons are facing outward, then export it again to DTS file format.

My Hovercraft is stuck to the ground and cannot move!
Open up the world editor in Torque by pressing F11, smooth the land around the start area, also you might want to move some of the spawnpoints up a little bit.

Alternatively, set *stabLenMin* and *stabLenMax* in *hovercraft.cs* higher then current values.

There is no sound!
True, sound datablocks were not defined in hovercraft.cs. Fell free to add one in Hovercraft.cs.

There's is no contrail effect!
True, I didn't set up a particle emitter for the hovercraft. Fell free to add one. Refer to Torque Developer Network or Garagegames website for more explanation on setting up emitter.

Where's my Hovercraft? I can see only Buggy car!
Make sure hovercraft.cs is loaded in game.cs and disable the line that load car.cs. Car.cs is the datablock file for the Buggy.

I want to know more about game design!
Head your way to building E6A 372 to Dr. Manolya Kavakli's Office.