Geometric modelling of ships for real-time 3D ship simulators

J.M Varela & C. Guedes Soares
Centre for Marine Technology and Engineering (CENTEC), Instituto Superior Técnico, Universidade de Lisboa, Portugal

ABSTRACT: The paper presents the sequence of procedures and the main techniques used currently to build a digital geometric model of a ship that fulfils the high demanding requirements for realism and performance of modern 3D Ship Simulators. The work presented shows that some particular characteristics of a ship, often define how to apply these techniques. It also highlights the importance of a well-defined workflow, and discusses the impact that it may have on the modelling time and quality of the final product. The description of the procedures and subsequent discussion are supported by a real example of 3D modelling of a free fall lifeboat and a tugboat currently used in a ship manoeuvring simulator.

1 INTRODUCTION

Modern real-time simulation systems for training in general and Ship Simulators in particular, are highly demanding tools from the graphical and physical points of view. The training of pilots and masters, using this type of systems, requires the user to be immersed in a very realistic Virtual Environment (VE). The VE must create a sense of believe, by stimulating actions and responses as if the user was facing a real situation. The realism and the level of immersion depend not only of the VE itself, but also of the overall surrounding physical environment, such as the equipment and structures of the simulator, the size and quality of the display device, the surrounding sound, or even the ambient lighting (Gruchalla, 2004). However, despite the importance of all these factors, the levels of display and interaction fidelity of the Virtual World, presented in the display device, are still the most significant factors in determining the performance, presence, engagement and usability of the simulator (McMahan et al. 2012). Therefore, it is fair to say that in this type of systems, the Virtual World plays a decisive role on the user’s behaviour. Moreover, for Desktop Ship Simulators, the surrounding environment may be quite “poor”, and therefore, these must rely almost only on the quality of the Virtual World.

In Virtual Reality systems, the world is entirely virtual and must be entirely modelled and programmed. This raises many production issues because the time and efforts spent in creating these supposedly realistic Virtual Worlds may easily become very large, and even derail the development of the simulation system. Therefore, the strategy, procedures and techniques used to create the Virtual World, which means modelling geometrically and physically all the virtual objects, is of crucial importance to reduce the time spent on this task. For the case of Ship Simulators, two main types of objects play a fundamental role in the simulation: the water surface (Varela and Guedes Soares, 2014), and the floating objects (Varela and Guedes Soares, 2011), which can be ships, platforms, etc. Often, coastal entities such as port structures, headlands or capes among others, also play important roles on typical operations simulated by these systems.

Considering these aspects of the development of Ship Simulators, the current paper addresses the creation of geometric models of ships (or other floating structures) that are suitable for Virtual Worlds. The overall workflow adopted to the development of the model is described, from the generation of the mesh, to the use of the so-called next-generation texture techniques to improve the realism of the model. The main techniques that allow the well-known fragile but crucial balance between performance and graphics quality are described.

Many aspects, technics, small tricks and tips about geometric modelling of ships exist, and should be mentioned and described. However, due to obvious restrictions on the paper length, it is not possible to present them all. Therefore, the paper will focus mainly on what the authors believe are the core methodology and techniques to achieve a high quality model. The reader should also be aware that the accuracy and execution of the techniques described
2 RELATED WORK

Geometric modelling, based on either solids, NURBS (Non-uniform Rational B-Spline) surfaces or triangular meshes, comprises a set of procedures that have been developed and improved over the last decades. Botsch et al. (2007) highlights the following study fields in the geometry-processing pipeline based on triangle meshes: data acquisition, removal of topological and geometrical errors, analysis of surface quality, surface smoothing, parameterization, simplification for complexity reduction, remeshing for improving mesh quality, freeform and multi-resolution modelling.

In the maritime industry, 3D models of ships are mainly CAD models used for design and production. Hull, structures and piping systems are the main targets for 3D geometric modelling of ships, where many research work has been done during the latest years, namely on ship hull modelling.

NURBS provide a set of efficient and accurate techniques (Dimas and Briassoulis, 1999) for modelling freeform shapes and they are normally preferred to obtain accurate and smooth hull shapes for engineering computation purposes (Ventura and Guedes Soares 1998, Pérez-Arribas et al. 2006, Shamsuddin et al. 2006). Triangular mesh models can be efficiently generated from NURBS using appropriated triangulation algorithms as described in Varela et al. (2011).

Polygonal mesh based models for VEs may also be obtained through 3D scanning, which is currently a common method used in a variety of application fields. However, scanning technology still has some drawbacks such as the expensive equipment and the need of post-processing to reduce the point data (Lee et al., 2001) and correct defects such as noise, outliers, holes or ghost-geometry (Weyrich, 2004). Moreover, 3D scanning is not physically practical for most the ships due to their dimensions and number of components.

Therefore, geometric modelling of polygonal meshes from sketch is normally applied for almost all the components of the ship model.

For animation models, subdivision surfaces (Zorin, et al. 2000) are normally used to obtain smooth surfaces. This type of geometry, initially developed by Catmull and Clark (1978), is controlled by a coarse control mesh normally modelled from a sketch, to which successive refinements are applied to obtain smooth surfaces.

3 MODEL REQUIREMENTS

The geometry of a 3D model, either a ship or any other object to be used in an interactive VE with a high level of graphical realism, must comply with a set of requirements regarding its size, topology or complexity.

3.1 Level-of-Detail

In 3D geometric modelling for real-time applications, the main indicator of the level-of-detail is the number of polygons that compose the model, referred to as the polycount.

For the case of ship simulators, the virtual world is usually an exterior large scenario where ships may be quite distant from the viewer. When this happens, there is no need to display a very detailed model of the ship, which consumes additional resources with no visual benefits. A single model of a ship, even if composed by some hundred thousands of polygons, will certainly not compromise the performance of the system that uses modern processors. However, if dozens of ship models with such level of detail exist in the VE, then performance issues may start to arise. Because of this, considerable efforts have been spent to find new methods to create simplified versions (with fewer polygons) of geometric models maintaining an acceptable level of visual realism (Pouderoux and Marvie, 2005, Ho et al. 2006, Bittner et al. 2009). Hence, different versions of the ship model are created with different levels-of-detail. A minimum of two versions are normally created, one for close and other for distant views. However, more versions may be created for intermediate distances if time and available efforts allow it.

3.1.1 Near distance model

The near distance model is the most detailed model and it is created first. The other models are then generated from this one by removing primitives, redefining the mesh topology or deleting objects appropriately.

The number of primitives of this model, which characterizes its level-of-detail, depends mostly of the size of the ship, its complexity (normally proportional to the quantity of relevant objects that must be modelled) and the refinement of the mesh.

Therefore, in order to obtain an appropriate model for 3D ship simulators, the number of objects modelled for each ship and the number of polygons of each object should be reduced as much as possible.

The major components of the ship models are the hull, decks and superstructures. However, these do not contain the majority of the polygons of the model because, despite their size, they can be modelled with large primitives. Therefore, normally these objects do not consist of a problem when a maximum limit of primitives is established.
Small objects with rounded shapes or multiple corners and edges that exist in high number are normally more problematic. The main factors that influence the decision of modelling them or not, are their size, visibility and role in the simulation. Typical objects that are not geometrically modelled, in most of the cases due to their size, are bolts and nuts, buttons and small handles. This does not mean that they are not represented in the final model; however, their geometry is normally replaced by adequate textures.

The visibility is an obvious factor to decide if an object is modelled or not: if the user does not see it, then there is no reason to model it. The same applies to primitives that are not visible within an object, even if the object is almost all visible. One common mistake at this level are the bottom primitives of objects on decks that are not visible and often are not deleted. Objects that are temporarily invisible but, due to their role in the simulation, may become visible at some instant, should also be considered. An example of this is the chain cable, which is most of the time inside the chain locker and in the hawse pipe, and only a small part of it is visible. However, if anchoring is to be simulated, then the entire chain must be modelled because it will become visible during the operation.

Finally, the simplification of the mesh is also a crucial factor to reduce the number of primitives of the model. The basic rule for mesh refinement is that the number of primitives should be proportional to the curvature of the surfaces. Fortunately, the use of normal and opacity maps (Section 6), allows applying some tricks to reduce the primitives and maintain the realistic appearance of curved surfaces. Cylindrical shapes, such as pipes and rails, are critical to increase the number of primitives and especial attention should be given to this type of objects. Figure 1 shows the low-poly version of the ‘Svitzer Pembroke’ tugboat. It gives an idea of the mesh refinement used in the near distance model for this ship.

One last concern is that the final model should not have long stretch triangles because this can cause lighting issues, namely on specular reflections. Once more, here the common sense should prevail and experience of the modeller is a key factor.

3.1.2 Far distances models
Far distance versions of the model are created by deleting small objects of the main model and reducing the primitives of the remaining ones. Ultimately, for the farthest distance model, only the hull and super-structure should be kept, and even those only with the minimum number of primitives enough to maintain the general silhouette of the ship, as shown in Figure 2 for the Free Fall Life Boat.

The simplification is obtained by removing small objects first and collapsing primitives then. From LOD 1 to LOD 2, only small and thin objects, such as handles, supports and cables were removed and the number of polygons decreased substantially to approximately 1/4 of the original. Then, for the subsequent LODs, the remaining objects started to be simplified by removing small polygons. In LOD 4, only the main objects are maintained and even those are quite simplified. In the last LOD, the only concern is to maintain a believable silhouette of the ship because this is to be used with distances to which none of the details is no longer perceptible.

Geometric algorithms such as described in Tarini et. al (2010), generate automatically low-poly models from the main model and these may be used to obtain the first approach of the far distance models. However, normally it is possible and even desirable to optimize the mesh automatically generated.
3.2 Interior

The interior of the ship, namely the navigation bridge, must also be modelled. In a manoeuvring simulator, the user will spend most of the time in the bridge, and therefore special attention should be given to this part of the model. Moreover, this is where the manoeuvring control equipment is located, with which the user will interact. Within the scope of the manoeuvring simulator, other interior compartments are normally not modelled.

One of the most important objects in the bridge is the control console. In modern ships, this can be very complex with dozens of buttons, small handles, clock meters or screen displays, as shown in Figure 3 (top) for the case of the ‘Svitzer Pembroke’ tugboat.

![Figure 3 - Interior bridge of the Svitzer Pembroke tugboat: real photo (top), virtual model without textures (middle), virtual model with textures (bottom).](image)

However, most of these objects are not used in the majority of operations simulated and they are replaced by textures mapped into the console. Figure 3 (middle) presents the virtual model of the console without textures, where most of the buttons are not geometrically modelled. In Figure 3 (bottom), images are mapped in the geometry to represent visually the buttons and meters.

Images should be used carefully when replacing geometry. They are adequate for objects that do not raise up too much from the top of the console. Buttons, clock meters and screen display are the ideal candidates, however for handles, joysticks, the results are not so realistic, and modelling their geometry should be considered.

3.3 Textures

Ship models for modern ship simulators must take advantage of the so-called next generation texturing methods. This means that textures are no longer used only for displaying images mapped on the meshes, but they also to orient surface normal vectors (Olano and Baker, 2010), to create lighting effects, or even to change the geometry of the model (Bunuel, 2005). Much of these features are directly related with the advances of the programming features implemented on modern GPUs, and therefore, they are only efficient when programmable vertex and fragment shaders are used.

Texturing the model requires the creation of at least three types of maps: the diffuse map, the normal map and the specular map. Depending of the size and complexity of the ship, more than one texture of each type may exist. However, the lowest possible number of textures should be used, since these will also consume computational resources.

The number and contents of each texture is defined during the unwrapping phase of the modelling process, which will be described ahead. However, in general term, if more than one texture (of each type) is necessary to map all the meshes, then the contents of each texture should be organized taking into account as far as possible, the following rules:

- Parts of single objects should be kept in the same texture.
- Objects belonging to the same functional system should be kept in the same texture.
- Systems with the same material characteristics (reflection type, roughness, specularity) should be kept in the same texture.

The first two rules promote reusability of the objects and texture or part of the texture in other models. This is important for objects that may be reused in various ship models, such as hatches, mooring bollards, deck shocks, etc. The same applies to more complex systems, such as cranes, towing winches, mooring equipment, among others, which may also be reused in various ship models only with small changes.

The third rule aims the reduction of the number of different materials, which reduces the number and the computational work required by fragment shaders.

4 THE GEOMETRY OF THE MODEL

The creation of the geometry is divided into two main parts: the high poly model and the low-poly model derived from the first one. In this case, the low-poly model corresponds to the near distance model described in previous section.

4.1 High-poly geometric model

Ultimately, the overall goal of the high-poly model of the ship is to be used to generate the normal maps that will provide a detailed appearance to the low-poly model. Figure 4 presents three examples of high-poly version of ships with the corresponding number of primitives and objects.

The high-poly version of the model should be built without restrictions into the number of primi-
tives or objects used. It should be as realistic as possible containing all the objects and details of the real ship. The only constraint applied to the modelling of the high-poly version should be the schedule to have the model ready to be used. Therefore, in high-poly models there is no relation between the size or type of ship with the number of objects or polygons.

As can be seen in Figure 4, the number of polygons is not proportional to the size of the ship and does not depend of its type. Typically, high-poly models are normally made of subdivision surfaces. These are generated automatically from coarse meshes that can then be used to create the low-poly versions of the models. Figure 5 presents the control meshes and the subdivision surfaces of the Free Fall Life Boat model.

Hard edges should be avoided in high-poly models and curved silhouettes should have rounded appearance as presented in Figure 6 for the hawser winch and anchor windlass of the ‘Svitzer Pembroke’ at fore deck in high-poly version. Bolts, nuts, buttons and small handles are all modelled independently of their size as also shown in Figure 6. In order to maintain the knuckles of the control mesh when the subdivision surface is generated (without adding hard edges), it is required to add flow edges near the knuckles as presented in the engine model of Figure 7 (detailed window). One should be aware that this technique increases significantly the complexity of the geometry and the number of primitives.

Contrary to low-poly models, the existence of n-gons (polygons with more than four edges) on high-poly models is acceptable as long as it does not create lighting defects. However, one should be aware that tessellation executed by subdivision algorithms to obtain the high-poly version, sometimes originates strange and degenerated polygons. If this happens, then the n-gon(s) should be removed by defining new mesh topologies.

4.2 Low-poly geometric model

The low-poly model of the ship is created from its high-poly version, which normally contains many cloned objects, with excessive geometric detail for a real-time model. Therefore, it is recommended to do some work planning before start modelling the low-poly version.

N-gons should be avoided at all costs for real-time models. The graphics engine, which processes only triangles, will try to triangulate n-gons and it could create odd shading. Moreover, the existence of n-gons can also create small defects on the process of baking out the high-poly object. Most of the times, it is faster to create the low-poly version of the model from the control mesh of the high-poly model. In fact, subdivision algorithms normally generate surfaces that are quite close to the control mesh. Therefore, a copy of the original control mesh may be created and shaped to the high-poly subdivision surface with just a few adaptions. Moreover, the subdivision process shrinks slightly the control mesh generating a subdivision surface that is completely inside the control mesh. As will be described ahead, this is an important characteristic for the baking process. Adaptions to the control mesh include mainly
the remove of flow edges near the knuckles and n-gons that may exist.

The control mesh, high-poly and low-poly versions of the external part of the hawser winch engine shown in Figure 6 are presented in Figure 7.

Sometimes the control mesh is quite complex and it is easier and faster to create the low-poly version of the object from scratch. This is the case of the object in Figure 7. In order to obtain the main body of the low-poly version of the object, it is faster to create a new cylinder with 12 sides aligned with the main body of the control mesh, than to remove flow edges and small holes or buckles. As can be seen, the number of primitives of the subdivision surface is extremely high, only suitable for static renderings. On the other hand, the low-poly version contains only a few hundred polygons, mostly quads and no n-gons. It can also be seen that all the objects of the control mesh were attached into a single object in the low-poly version, but with fewer elements than the control mesh. This means that some smaller objects were simply deleted from the geometry and are represented in the low-poly version by textures.

5 THE UNWRAPPING AND BAKING PROCESSES

Unwrapping and baking of the model aims the generation of the textures that will provide the visual realism to the low-poly version. The need for these techniques is a consequence of having a geometry defined in the 3D world whose surfaces exhibit images defined in a 2D space such as the diffuse, normal, occlusion and specular maps.

5.1 Unwrapping

The process of unwrapping consists in unrolling the 3D mesh into the 2D space. Figure 8 shows the one of the unwrapped maps of the low-poly version of the ‘Svitzer Pembroke’ ship model. A few important rules must be taken into consideration in order to obtain an efficient result:

- The deformation of the primitives of the unrolled mesh (mapped into the texture) should be avoided as much as possible (naturally, this is not possible for undevelopable surfaces).
- Hard edges in the low-poly geometry, typically with an angle higher than 45 degrees, should originate edge seams in the unwrapped map.
- Whenever possible, (long) edge seams in the unwrapped map should be orthogonally aligned.
- The unused areas of the texture should be reduced to the minimum possible.

![Unwrapped mesh](image)

Figure 7 – The control mesh, which is the base geometry to obtain the high-poly and low-poly version of the model.

Figure 8 – Unrolled mesh mapped into a 2D texture from which the normal, diffuse (bottom-left image) and specular maps are generated.

Typically, the area that is more subjected to deformations is the hull, and therefore edge seams are created (A) during the unwrapping process to avoid excessive deformation. It can be seen that the step between the upper forward deck and the lower working deck (B), is broken into a separate parts in the UV map, due to the hard edges in the 3D mesh. Cylindrical objects such as pipes, rails, etc. are unrolled orthogonally (C) and empty spaces denote a waste of memory that should be avoided (D), even if small.
5.2 Baking

The baking process aims to map the normal vectors of the high-poly surface into the unwrapped map of the low-poly model. This is achieved by defining a projection cage with the same topology of the low-poly mesh and project it into the high-poly model. Figure 9 presents an example of the projection cage around the high-poly model.

![Projection cage of the Svitzer Pembroke hull](image)

Projection vectors are defined at the cage vertices by the normal vector at each vertex pointing to the high-poly model. At the faces of the cage, projection vectors correspond to the interpolation between vertex normals. The polygons of the cage have a direct correspondence with the unwrapped map because they were both generated from the low-poly model. Each projection reveals the normal vector in the high-poly model, and the result is stored in the unwrapped map through a predetermined colour code. This way, the low-poly model exhibits the normal vectors of the high-poly version.

In order to obtain a consistent normal map without errors, the projection cage should be as close as possible to the high-poly model. However, for the projection of the cage to be possible in all extent, the model must be completely inside the cage.

As a starting point, the projection cage should coincide with the low-poly version of model (Figure 9, top). Normally, the final cage is then obtained by moving the vertices along the surface normal direction until the high-poly version is completely inside the cage (Figure 9, bottom). The cage should not be excessively large as shown in Figure 9 (center).

6 TEXTURING

Texturing should be the final task of the process and aims to assign a realistic appearance to the surfaces of the model. This is achieved with a 2D image editing software and a real time shader that updates and renders the textures in the model when they are modified.

6.1 Normal map

The normal map modifies the normal vector of the surface according to the colour of the pixel in the texture (mapped into the 3D surface). This will affect the view angle in the shading equation and therefore change the appearance of the surface.

Figure 10 presents the normal map assigned to the interior model of the ‘Svitzer Pembroke’ tugboat.

![The normal map of the control panel of the Svitzer Pembroke tugboat](image)

It can be seen that most of the buttons of the control console (top-left part of the image) are represented by means of changing the normal vectors of the flat surface panels the panels. The normal map is automatically generated from the high-poly version of the model applying the baking technique described before. However, small corrections are normally required, namely on rounded hard-edges. Contrast correction is also commonly applied to increase the apparent elevation or roughness of the surface.

6.2 Diffuse map

The diffuse map will affect the diffuse colour of the surface. It is the map that most influences the appearance of the surface. Material colours, rust, scratches, dirt, signs or texts are all included in this map as shown in Figure 11 (left).

Although the final texture is a flat image, it is advisable to create and maintain a multi-layered version of this texture separately. Typical layers in the texture are: Base material colour (paint, wood texture, metal appearance, etc.), Dirt and scratches, Rust, Signs and Text. The use of layers allows to easily
change each of the listed items without affecting the others.

This map is normally created manually in a 2D modelling software package. The base image to start creating the texture is the unwrapped UV map generated by the unwrapping process. It is also common to use material photos, like the wood planking in the working deck area of Figure 11 (left), to increase the realism of the texture.

6.3 Occlusion map

Ambient occlusion is a non-physical based method to control the exposure of each point in the surface to the ambient light. The computation algorithm assumes that each object occludes to a certain extent the ambient light from the other objects that are nearby, generating a sort of soft shadows on their surfaces. The result is a diffuse lighting throughout the scene, casting soft shadows, where enclosed and sheltered areas are darkened. This type of map is automatically generated by the 3D software package. The result for the unwrapped UV map presented in Figure 8, is shown in Figure 11 (right).

Figure 11 – Diffuse and occlusion maps of the Svitzer Pembroke tugboat.

In order to reduce the computational work and memory used during the real-time rendering, it is usual to blend the occlusion into the diffuse map, generating a single diffuse-occlusion map.

7 RESULTS AND DISCUSSION

The ‘Svitzer Pembroke’ tugboat was modelled using the techniques and the sequence of tasks described in the paper. The hardware used in the modelling and rendering process is presented in Table 1.

<table>
<thead>
<tr>
<th>Table 1 – Hardware specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
</tr>
<tr>
<td>RAM</td>
</tr>
<tr>
<td>GPU</td>
</tr>
</tbody>
</table>

The chart in Figure 12 presents the partial times spent in each task.

Figure 12 – Time spent in each of the modelling.

The most time consuming task is the creation of the high-poly version of the model. It corresponds to the creation of the control mesh, since the final high-poly version is then automatically generated by subdivision algorithms. The generation of the low-poly version takes less than one third of the time spent in the high-poly modelling. It is also interesting to note that the total working time is divided approximately by half in building the geometric mesh (about 51%) and creating the textures (about 49%). Therefore, after finishing the geometry, the time spent is approximately half of the total time. The texturing process also consumes a significant part of the modelling time mainly on the generation of the diffuse map.

Although the primary purpose is the interactive simulation, the model of the ship is also detailed enough to be used in static animations, as long as a certain distance is kept from the finest details and appropriate production shaders are used. Figure 13 presents the same model rendered with a real-time shader (left) for interactive simulations and a production shader (right) for static animations.

Figure 13 – The low-poly model rendered with real-time shader (left image) and production shader for static animations (right image).

The realism is notoriously higher in the right image, which applies direct and indirect lighting, hard and soft shadows, color correction and anti-aliasing. However, this image takes about 35 minutes to render with the hardware specified in Table 1. Nevertheless, even if this realism is not available for real-time, the right image shows that the low-poly textured version model can also produce very realistic images for static animations.

Figure 14 shows the model in the Virtual Environment of the ship simulator under development in
Figure 14 – The model of the Switzer Pembroke tugboat in the Virtual Environment of the simulation system.

8 CONCLUSIONS

The paper presented the methodology to create the geometry of a ship suitable for modern interactive 3D ship simulators. The modelling process can be divided in two main tasks concerning the working time and the skills required to perform them: the geometry modelling and the generation of the textures.

Although the only geometry used by the simulator is the low-poly version, it is clear that the control mesh and the high-poly version are essential to obtain the expected results. Moreover, the use of textures, not only to ‘paint’ surfaces but also to modify the geometry and create lighting effects is nowadays a requirement for interactive 3D models.

Using the methodology and techniques described, low-poly version of the model is also suitable for static animations.

Naturally, the modelling process depends of the modeller skills. However, the sequence described is independent of this factor and increases the efficiency of the process.

Finally it is important the highlight that although the model may have the potential to present a very realistic scene, including appropriate and detailed geometry, fully textured using next-generation maps, it is the shading equation that will define the final rendered image. Therefore, in order to obtain the final realistic image, the shader must have the ability to harness the full potential of the model.

9 ACKNOWLEDGEMENT

The study was carried out within the framework of the research project “Towing Dynamics of Ships in Harbour Areas” (PTDC/ECM/100686/2008) financed by the Portuguese Foundation for Science and Technology (FCT), Portugal. The first author has been funded by the Portuguese Foundation for Science and Technology (FCT) under grant nr. SFRH/BD/39312/2007.

10 REFERENCES


