visualscience
www.justwannaplay.com

CARMAGEDDONTV

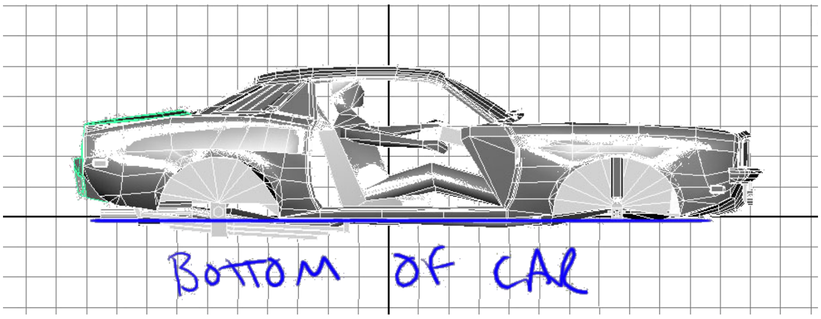
Vehicle Specification

This document shows how a vehicle model is setup and texture mapped.

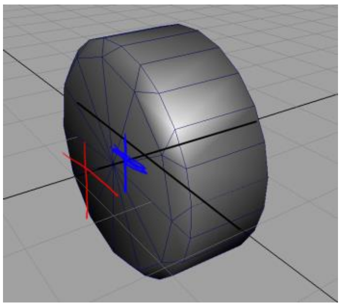
Scene Setup

Maya units are 1 unit = 1 cm. The scale is real world e.g. A car is 5 units long or 5 metres.

The vehicles face **–Z** (**Car faces up in the top view**.)



The vehicle or parts should be placed with X & Z at the origin (0,0,0)
The bottom of the vehicle (underside) should rest on the Y origin.
(With deleted history & freeze transformed.)

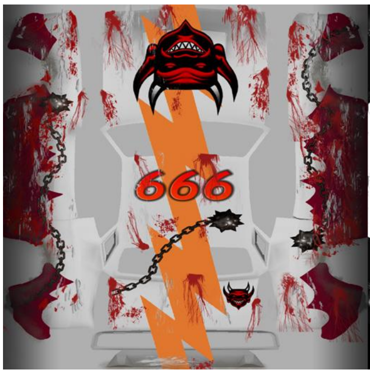


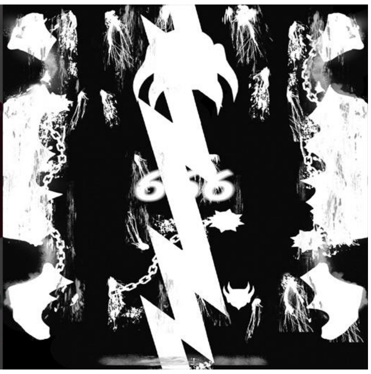
A single **left wheel** is exported. (Export left front & left rear if different.)
Centred on the origin in Z & Y. Pivot point placed where the suspension turning axis would be – blue cross.)   n.b. transform 0,0,0

All relevant models to be kept in one file e.g. mustang.mb

Vehicle Upgrades

Colour customisation


RGB


ALPHA

The colour is chosen in the front end with two slider bars. The grey in the texture map are the areas of the car that are colour corrected. The alpha (white areas) mask the anything which is not to be colour corrected. e.g. the decals.

Elements for the body texture kept separate for armour and decal variation generation. This will be combined to great the texture set for each decal. e.g.

4) bandit_body_dirt.tga - permanent - dirt, blood and scratches

3) bandit_body_stockcar_decal.tga - decal variation

2) bandit_body_armour_adv_right.tga - armour variations

1) bandit_body_base_GI.tga - base layer of car with self shading and tarnish

This information will be held in the master _body.psd file. (See the texturing section.)

Wheels

Four locators are placed in the scene to export the wheel offset positions.

Named :-

front_right_wheel
front_left_wheel
rear_left_wheel
rear_right_wheel

Additional upgrade wheels are exported in the same manner as a standard wheel with the pivot point (at 0,0,0) placed where turning pivot of the wheel / suspension is located. (Wheels can be tested by snapping them to the 4 wheel offset locators.)

Armour and Weapons

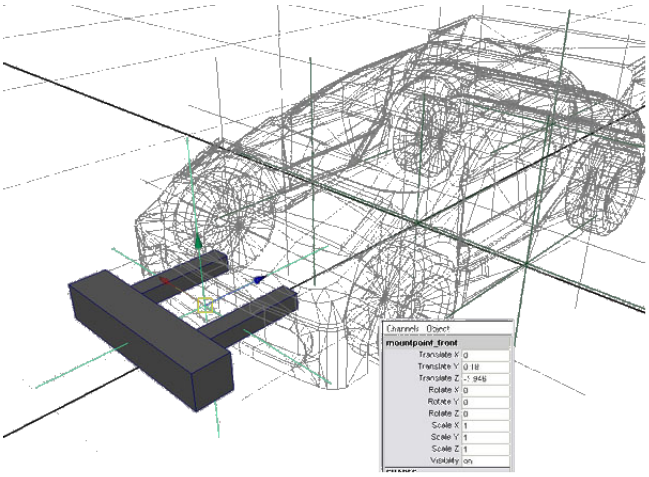
Each class of vehicle (e.g. small) has a range of weapons and armour that are interchangeable between the vehicles in that class.

There are five locators in the scene that control the offset positions for the weapons. Named :-

mountpoint_front
mountpoint_rear
mountpoint_left
mountpoint_right

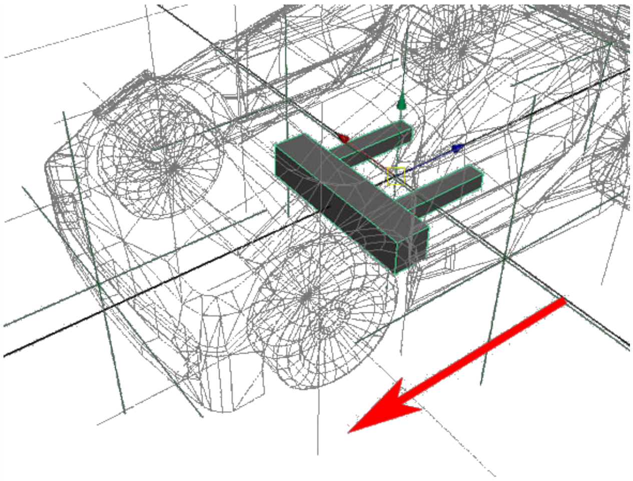
mountpoint_top - this is for the in game power-ups

The mountpoints need to be positioned for each vehicle so the weapons fit into the physical slots modelled on the vehicle. See Below.

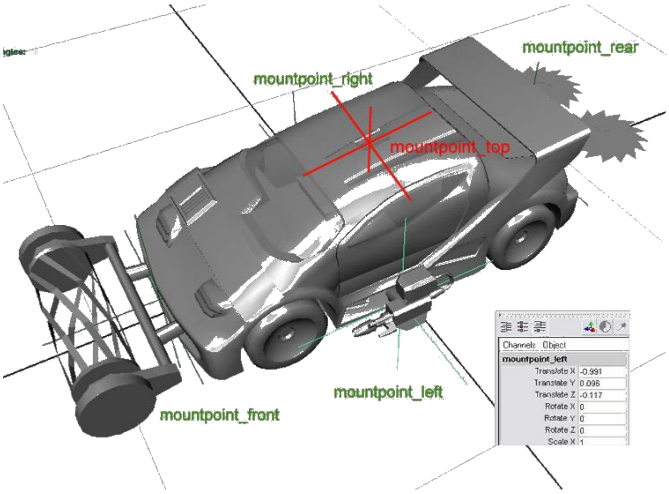


Channels Object

mountpoint_front	
Translate X	0
Translate Y	0.18
Translate Z	-1.946
Rotate X	0
Rotate Y	0
Rotate Z	0
Scale X	1
Scale Y	1
Scale Z	1
Visibility	on

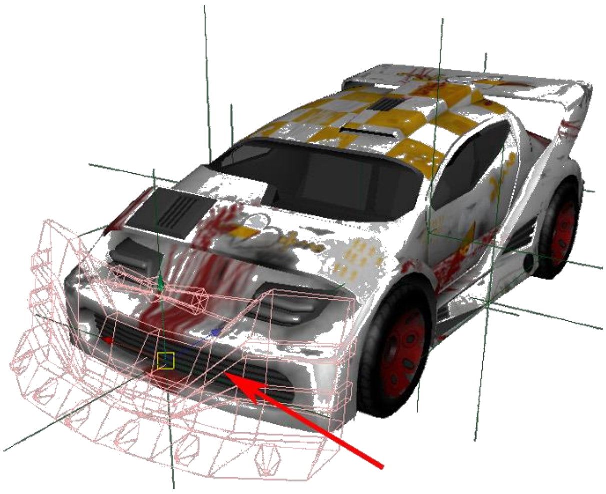


The weapons pivot point if fixed at 0,0,0. It is moved in game to the mountpoint locator position.



mountpoint_left
Translate X -0.991
Translate Y 0.096
Translate Z -0.117
Rotate X 0
Rotate Y 0
Rotate Z 0
Scale X 1

Mountpoints



Each designed with slot recesses
Front & Back

Model Specifications

As the game follows the car from behind more detail to the rear of the car. e.g. modelled panel lines and lights definition to add visual interest.

The underside of the vehicle needs to have some modelled and textured detail e.g. exhaust pipes to break up the shape and catch the light (specularity) when the vehicle rolls in the game.

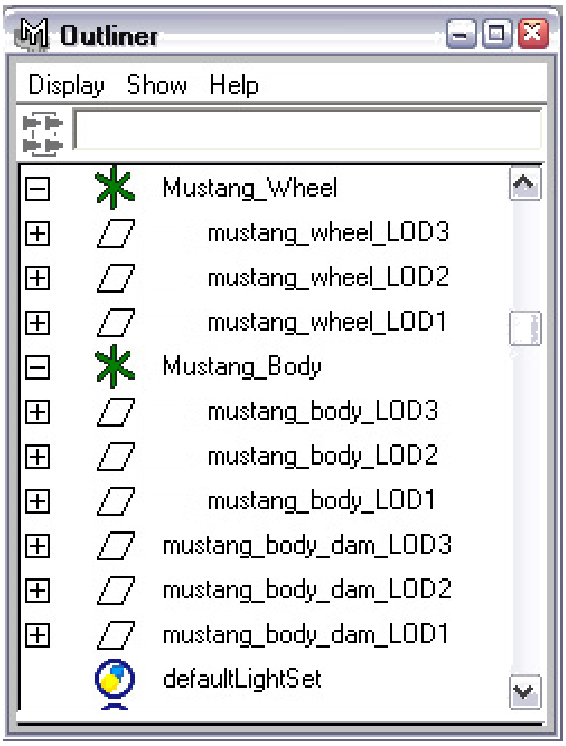
The poly counts represent a complete vehicle not including any of the upgrade components or wheels.

Standard Vehicle

• LOD 1	<6000
• LOD 2	<3000
• LOD 3	<1000

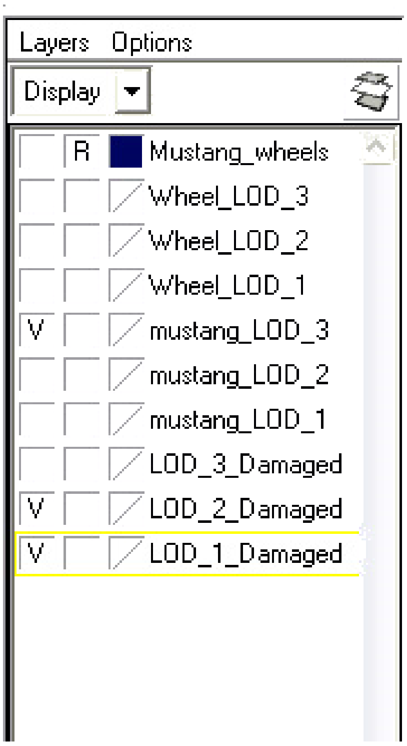
Wheels

• LOD 1	<550
• LOD 2	<350
• LOD 3	<100



- Mustang_Wheel
  - mustang_wheel_LOD3
  - mustang_wheel_LOD2
  - mustang_wheel_LOD1
- Mustang_Body
  - mustang_body_LOD3
  - mustang_body_LOD2
  - mustang_body_LOD1
- mustang_body_dam_LOD3
- mustang_body_dam_LOD2
- mustang_body_dam_LOD1
- defaultLightSet

LOD setup in groups and named as above.

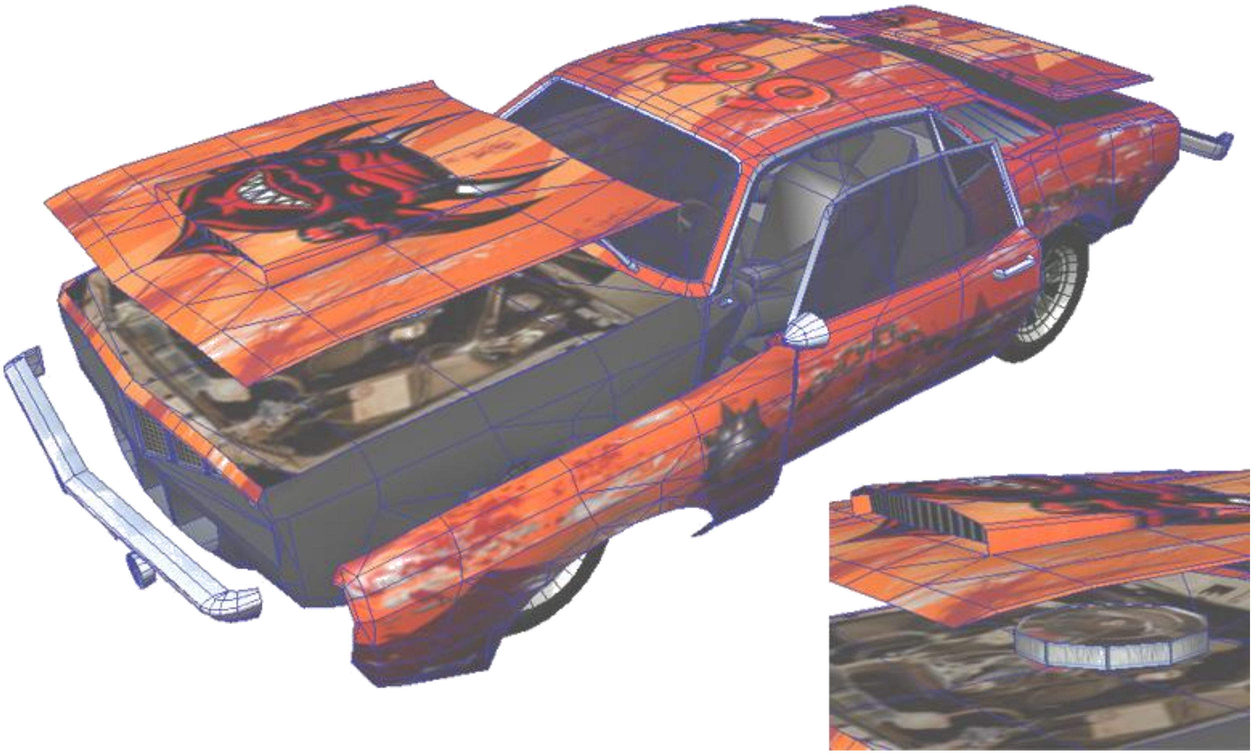


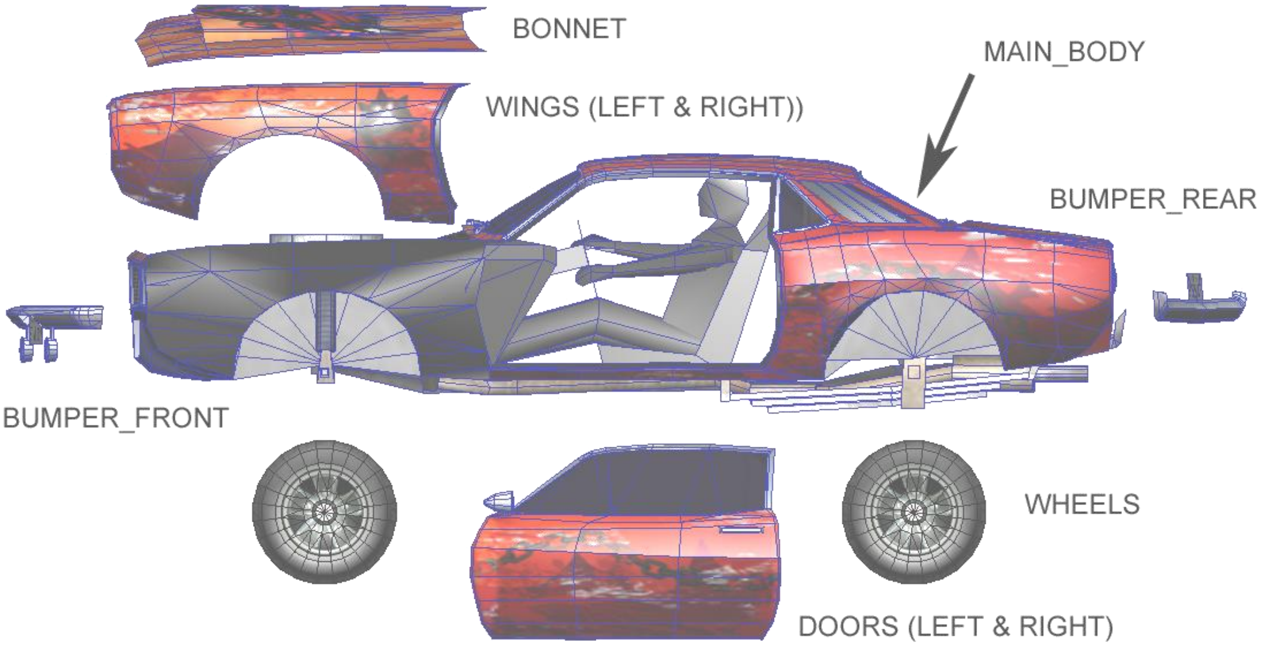
	R	Mustang_wheels
		Wheel_LOD_3
		Wheel_LOD_2
		Wheel_LOD_1
V		mustang_LOD_3
		mustang_LOD_2
		mustang_LOD_1
		LOD_3_Damaged
V		LOD_2_Damaged
V		LOD_1_Damaged

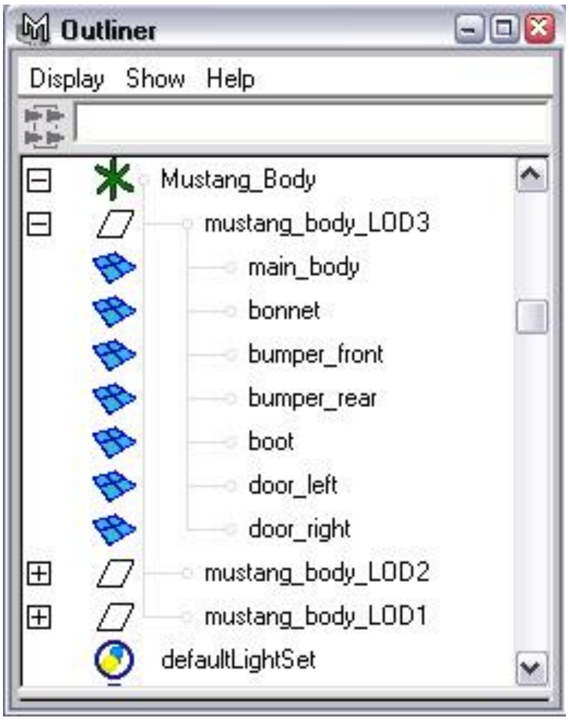
Use layers to keep the scene tidy.

Vehicle components

The vehicles need to be broken into components in the following way.




BONNET
MAIN_BODY
WINGS (LEFT & RIGHT))
BUMPER_REAR
BUMPER_FRONT
WHEELS
DOORS (LEFT & RIGHT)



Interiors & Windows

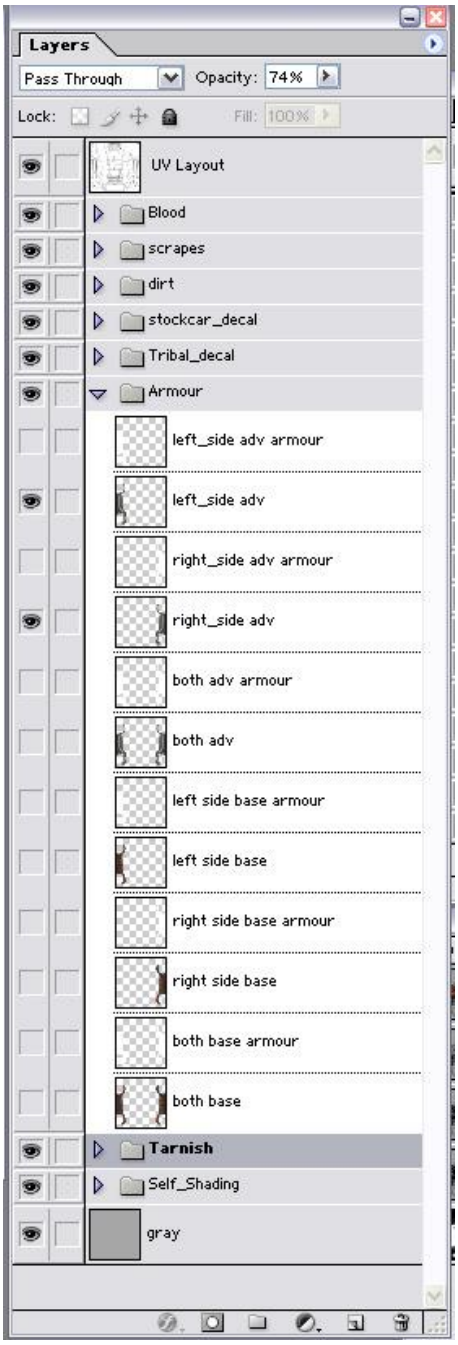
The vehicle has a basic interior and driver of approx <300 polys (included in the poly budget for the vehicle.) A low LOD player or ped character rig and model will be used as the driver with slightly transparent windows. The steering wheel is a single poly using alpha transparency for it shape. Texturing for the interior is placed in the _**detail** texture map.

**alpha** at the end of Maya shader name for anything with **transparency** e.g. windows. Transparent faces needs to be mapped with a separate Maya shader using the _detail texture with the transparency connected in the shader network.

Texturing and geometry need to be added behind the components. e.g. engine under the bonnet or the panel inside the front wings. In the damage system these components are to come away from the vehicle exposing what is underneath.

Texturing

The vehicle will have decal textures combined in code to create the main body texture map shown in the game. Therefore separate photoshop layers are required keep in the dirt, decals and armour etc in separate layers. (See below.)



Layers

Pass Through

Opacity: 74%

Lock:

Fill: 100%

UV Layout
Blood
scrapes
dirt
stockcar_decal
Tribal_decal
Armour
left_side adv armour
left_side adv
right_side adv armour
right_side adv
both adv armour
both adv
left side base armour
left side base
right side base armour
right side base
both base armour
both base
Tarnish
Self_Shading
gray

Two 1024 x 1024 **Photoshop psd** files will be created containing the texturing information for the vehicle. This will contain the decal layers, self shading / GI, dirt, scratches and damage in separate layers.

Named carname_**body**.psd & carname_**detail**.psd

This file can be flattened and named e.g. mustang_body.tga to display and map the finished vehicle.

Quota

1	1024x1024 map **_body.tga**	Used for the main body and any part of the vehicle that will be affected by palette changes. This map will be created in code to add blood, scratches and dirt.
1	1024x1024 map **_detail.tga**	Used for **all** the other parts of the vehicle e.g. underside, wheels etc. Alpha used for transparency. e.g. windows.

To allow glow to be added and control over the lights, e.g. break lights separate textures will be used e.g. **break_lights_bloom.**

Typical	**mustang_breaklights_bloom.tga**	@ 512 x 512
	mustang_headlights_bloom.tga	
	mustang_orangelights_bloom.tga	

_bloom.tga indicates bloom is to be added as effect.

alpha in a Maya shader name for anything with **transparency**
e.g. windows (Best mapped to only polys which are transparent.)

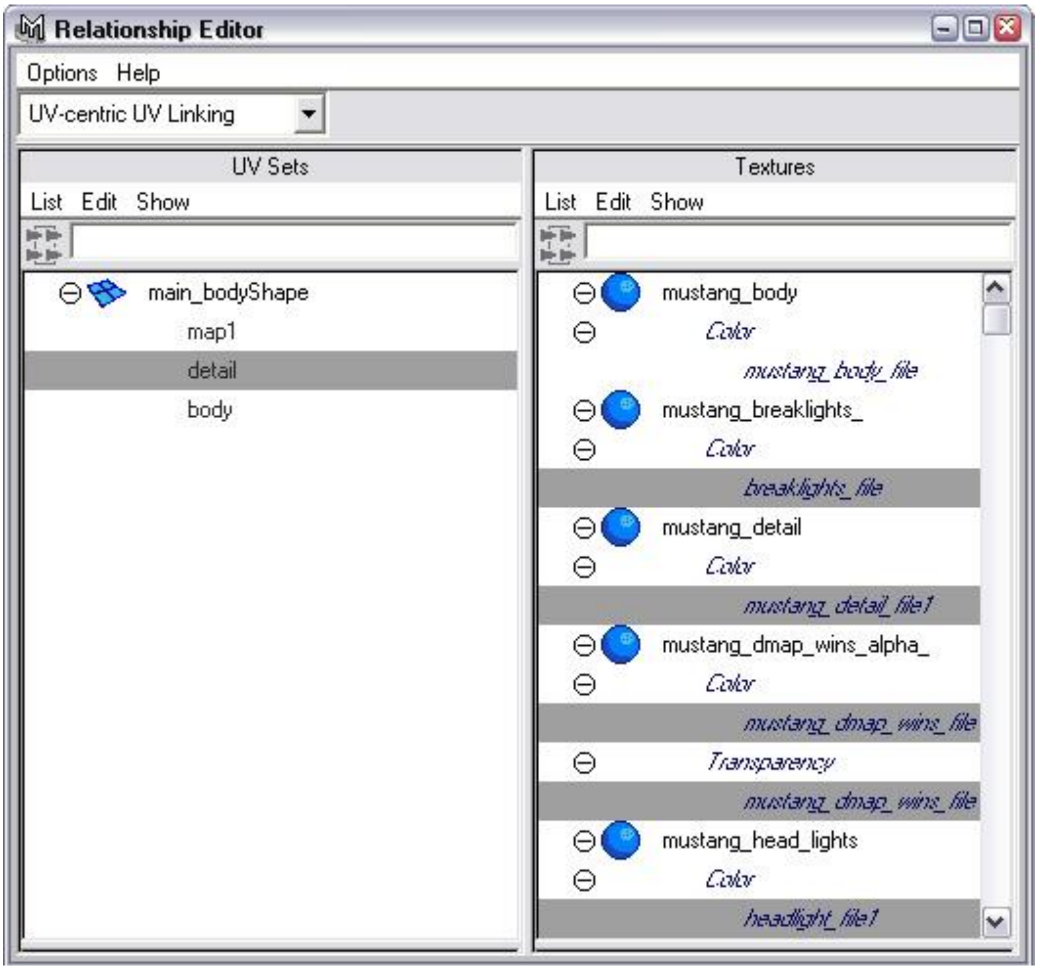
breaklights in the Maya shader name for break light control.

UV’s & Mapping

There are two UV sets are used for each vehicle.

body - for any polys or models with the _body.tga map applied.

detail - for all the other maps applied, _detail.tga, headlight.tga etc



UV-centric UV Linking

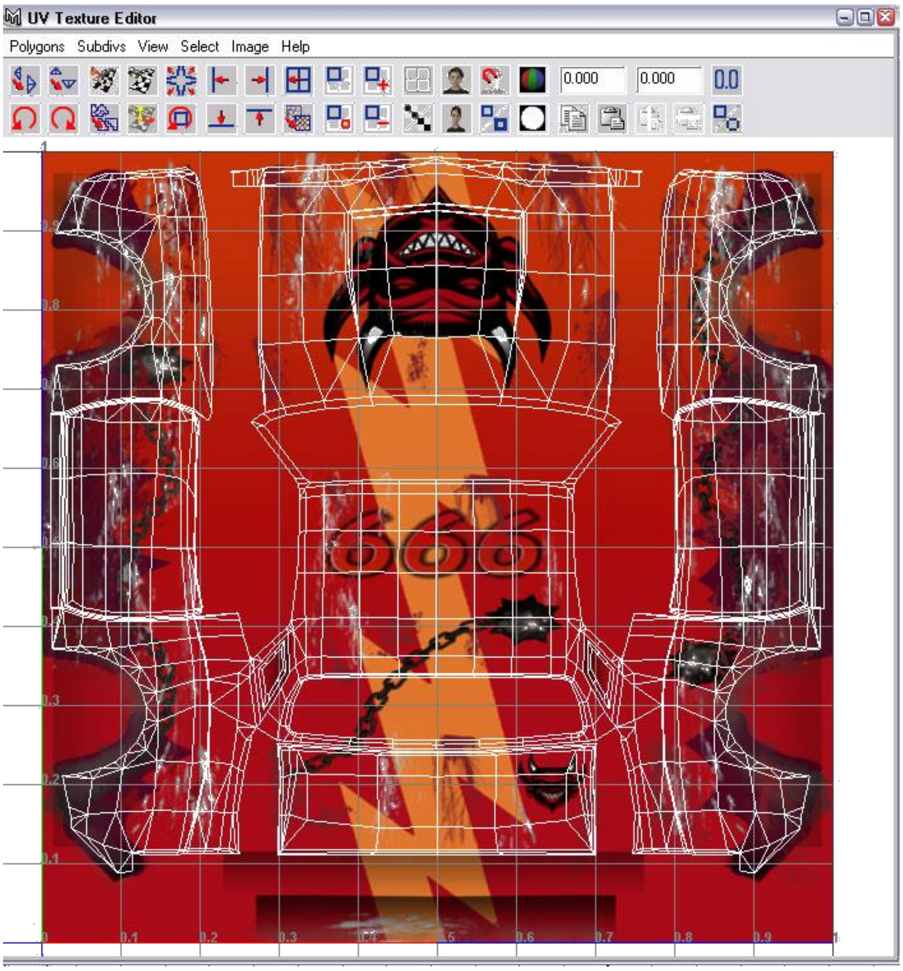
UV Sets:

```
main_bodyShape
    map1
    detail
    body
```

Textures:

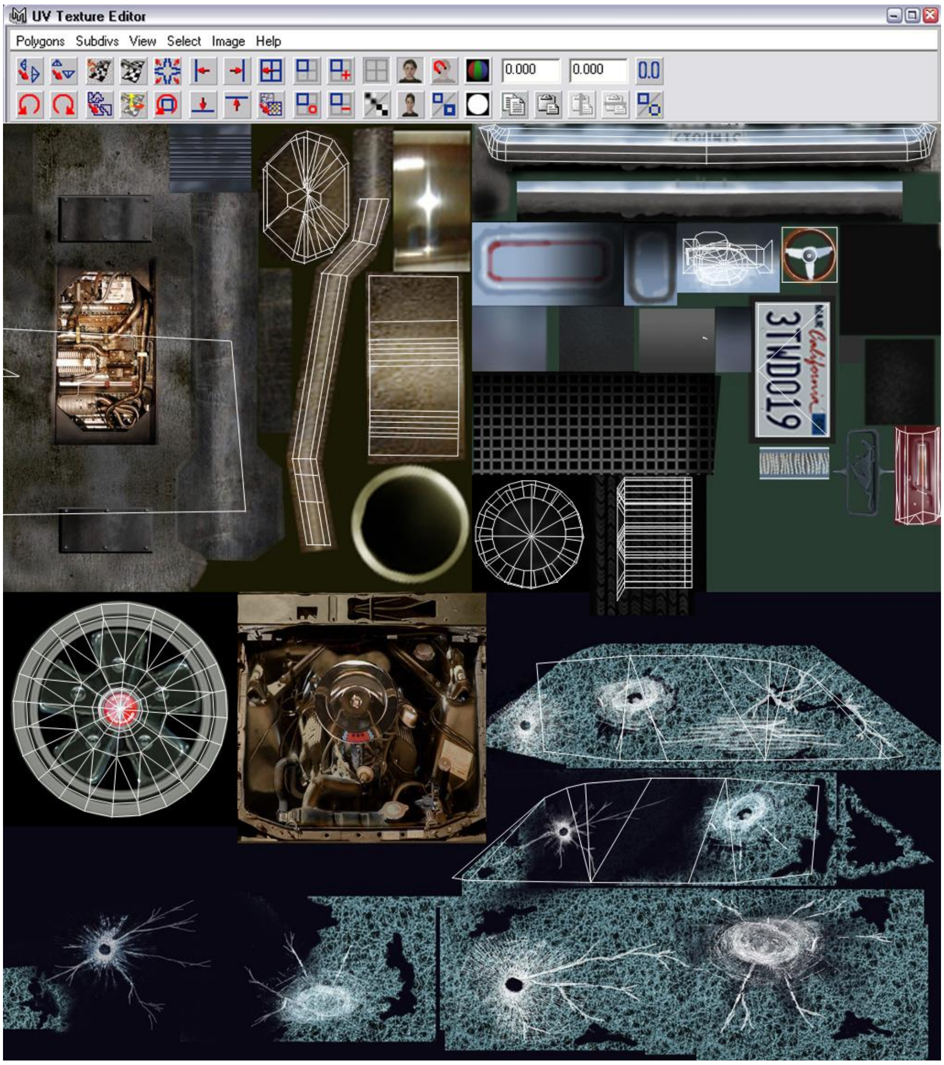
```
mustang_body
    Color
        mustang_body_file
mustang_breaklights_
    Color
        breaklights_file
mustang_detail
    Color
        mustang_detail_file1
mustang_dmap_wins_alpha_
    Color
        mustang_dmap_wins_file
    Transparency
        mustang_dmap_wins_file
mustang_head_lights
    Color
        headlight_file1
```

It is important to follow this convention as when the models are combined the texturing will not be correct otherwise.



The mustang body UV layout.

As the decals and particularly damage are a key aspect of the vehicles look both sides are shown in the UV layout.



Above is an example of a detail map. All the surfaces other than the cars body panels are included in this map including wheels and the interior.

The windows need to make up a reasonably part of this space to allow the damaged windows enough resolution.

Damage

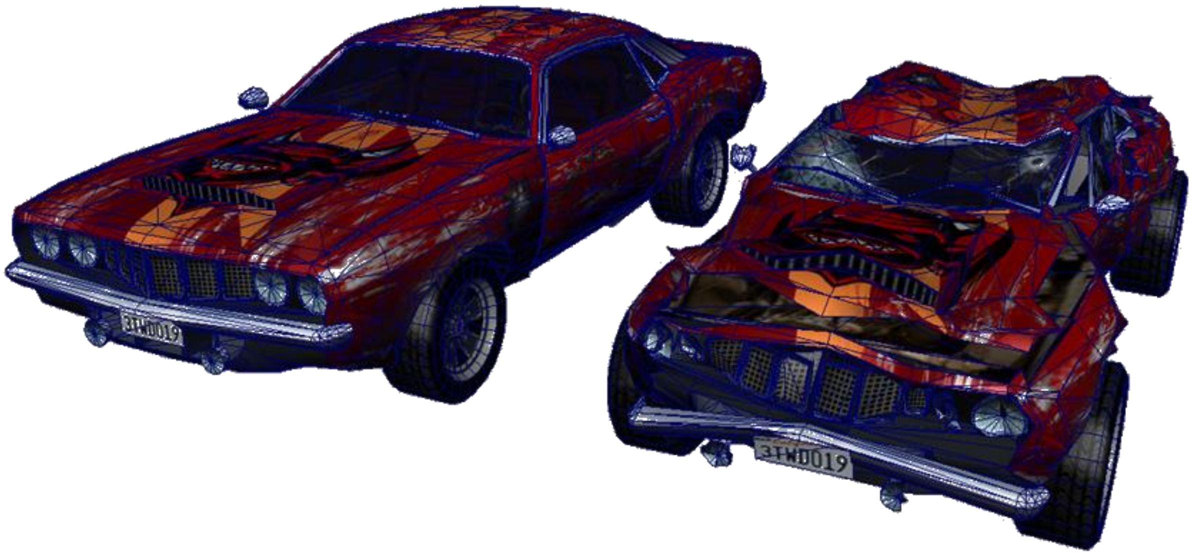
The final model **must be triangulated** before creating the damaged version.

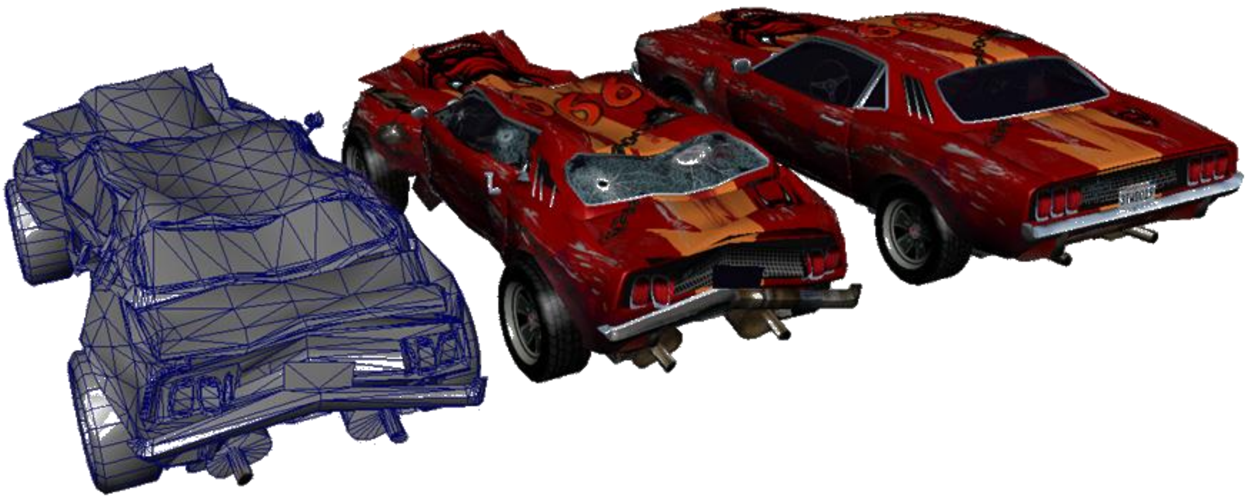
The damage system involves the creation of a damaged model **of the same vertex count** as the non damaged model. An exact copy deformed and manipulated to form the damaged version. In game the car is morphed between the undamaged and damaged local to the impact area.

The damage model needs to be created from the original mesh(s) via deforming and pulling / pushing vertices. This is so the vertex order remains the same for both versions for the morph to work correctly.

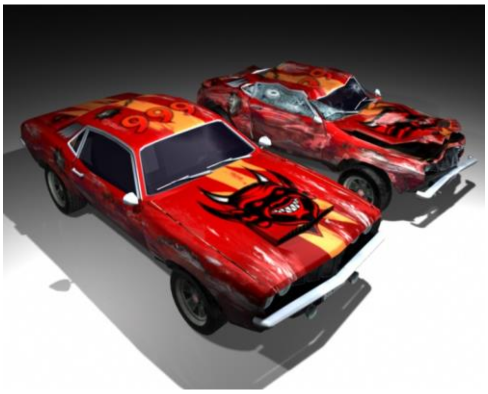
The model will show the complete vehicle damaged to its maximum all round.

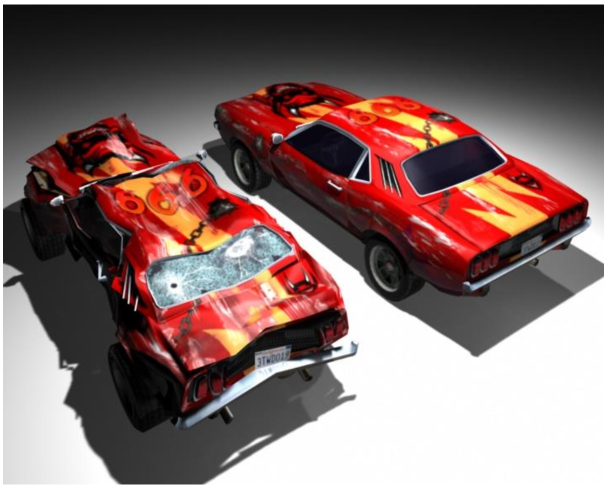
• LOD - There will be a damaged version for each LOD.





Damaged mesh with same vertex count as original.





The damaged LOD vehicle components will be named as the undamaged ones and grouped in a node named e.g. mustang_body_dam_LOD3

Photoshop layer(s) will be created for the damaged car and added the master _**body**.psd and _**detail**.psd photoshop files.