Chapter 15

Torque Lighting

15.1 Torque Lighting (Kit) Basics

15.1.1 The Torque Lighting Kit

This chapter discusses lighting features added by the Torque Lighting Kit (TLK). This product (made by Synapse Gaming) became a standard part of the Torque Game Engine in versions 1.5 and beyond. If you have TGE version 1.42 (or before), this chapter does not apply to your copy of the engine, and you may stop reading now. However, if you have version 1.5 through 1.52, please continue reading.

Unfortunately, a few of the engine features discussed in this chapter are advanced topics that I don't talk about until my next book, *Multiplayer Gaming and Engine Coding for the Torque Game Engine*.

Throughout the rest of this chapter, I will refer to these new lighting features as the Torque Lighting Kit (TLK) instead of just Torque lighting, in order to clarify that I am in fact discussing the features provided by the TLK code changes.

15.1.2 Light Datablocks and Light Objects

TLK lights consist of two components: a light datablock and a light object. For example, below you can see a (partial) definition of an `sgUniversalStaticLightData` datablock, followed by an `sgLightObject` that uses it.

```
// A TLK Light Datablock
datablock sgUniversalStaticLightData(sgDefaultLightDataBlock) {
   className = "sgUniversalStaticLightData";
   LightOn = "1";
   Radius = "10";

   ...

   AdvancedLightingModel = "1";
   EffectsDTSObjects = "1";
};
```

```
// A TLK Light Object
new sgLightObject() {
   canSaveDynamicFields = "1";
   position = "-39.0195 83.2004 252.625";
   rotation = "1 0 0 0";
   scale = "1 1 1";
   dataBlock = "sgDefaultLightDataBlock";
   Enable = "1";
   IconSize = "1";
   ParticleColorAttenuation = "1";
};
```

The fact that we can share a single datablock between multiple instances of TLK light objects makes configuring and updating multiple objects extremely easy.

TLK light datablocks contain common lighting information, such as color, intensity, and type (omni/spotlight), which is shared between multiple light objects.

Meanwhile, TLK light objects are placed in a scene and provide the actual illumination.

Types of Light Objects
There are four TLK light object types:

1. **`sgLightObject`** – This is a point light source, derived from the `fxLight` object. It can be used standalone, or it can be linked to particle emitters to enable various effects.
2. **`volumeLight`** – This is a point light source, derived from the **`sgLightObject`** object. It is used to create volumetric (cones of) light.
3. **`sgMissionLightingFilter`** – This is an area lighting modifier and is used to control the lighting in a specific zone or zones.
4. **`sgDecalProjector`** – This light object is provided to allow us to "project" decals onto game surfaces (either interiors or the terrain).

Note that prior versions of TLK implemented an `sgUniversalStaticLight` object. This is the legacy name for the `sgLightObject` object and should not be used any longer (even though the engine will allow you to do so.)

Types of Light Datablocks
There are only two TLK light datablock types:

1. **`sgLightObjectData`** – This datablock is used to initialize either an `sgLightObject` or a `volumeLight` object.
2. **`sgMissionLightingFilterData`** – This datablock is used to initialize an `sgMissionLightingFilter` object.

Note that prior versions of TLK implemented an `sgUniversalStaticLightData` datablock. This is the legacy name for the `sgLightObjectData` datablock and should not be used any longer (even though the engine will allow you to do so.)

15.1.3 Creating Light Datablocks

The first step to creating a new light is to create any datablocks that the light may use, so let's discuss editing and creating light datablocks, then we'll talk about creating lights.

Finding Light Datablock Definitions

By default, all light datablocks are located in one of two places.

- **Light Datablocks** - "common/lighting/lights/"
- **Cinematic Filters** - "common/lighting/filters/"

Furthermore, the standard starter kits will all load any datablocks found in these two directories automatically. So, unlike other datablocks, you do not need to update loading scripts when you create new light datablocks. Simply place your new light datablock's files in either of these standard locations and they will be loaded for you.

Of course you can change the above default file locations. Simply modify one or both of these global variables (found in "~/common/server/lightingSystem.cs").

- **Light Datablocks Path** – `$sgLightEditor::lightDBPath;`
- **Cinematic Filters Path** – `$sgLightEditor::filterDBPath;`

Manually Editing / Creating Light Datablocks

Although you may manually edit the datablocks if you choose, you'll find that it is much easier to use the built-in TLK Light Editor.

Starting The TLK Light Editor

To start the TLK Light Editor, simply do the following:

1. Load the mission you wish to edit.
2. Start the World Editor (F11).
3. Toggle the TLK Light Editor (F12).

Once you follow these steps, you should something like Figure 15.1 below.

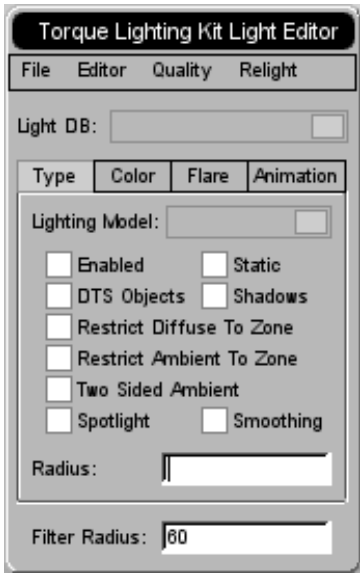

Figure 15.1 – TLK Light Editor.

Later, you can close the TLK Light Editor by pressing F12 again.

The TLK Light Editor Menu

As can be seen, this editor has four menu items in the top menu bar. These menu items supply the following choices.

- **File**
  - **New…** - Clicking this allows you to generate a new TLK datablock.
  - **Clone…** - Cliking this allows you to clone the current TLK datablock, creating a new datablock with all the same values but a new name.
  - **Save** – Clicking this will save any changes you made to the current TLK datablock.
  - **Restore** – Clicking this will restore any changes you made to the current TLK datablock since the last save.
  - **Close F12** – Closes (toggles) the TLK Light Editor.
- **Editor**
  - **Light Editor** – Enables the `sgLightObjectData` datablock editing features.
  - **Cinematic Filter Editor** – Enables the `sgMissionLightingFilterData` datablock editing features.
- **Quality**
  - **Production/Design/Draft** – You may choose any one of these and change the quality level of the lighting calculations, where production is the highest quality and draft is the lowest quality.
  - **Shadows** – Enables or disables shadows globally.
- **Relight**
  - **Filtered Relight** – Partial relight for area around camera (distance controlled by "Filter Radius" field; see Figure 15.1 above).
  - **Full Relight** – Completely relights entire mission.
  - **Reset All Light Animation** – Resets all light animations to their initial state.

The Light Editor Panes

In Light Editor (editing `sgLightObjectData` datablocks) mode, the editor presents a "Light DB" dropdown menu, four panes (tabbed pages), and a "Filter Radius" field.

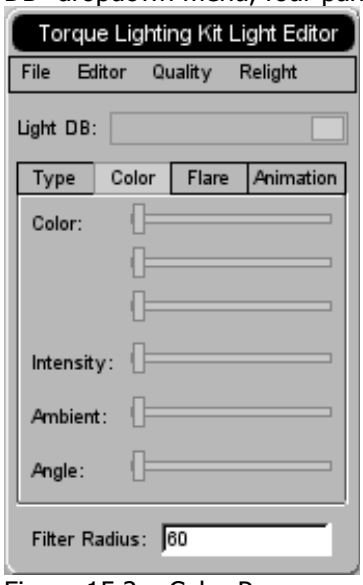

Figure 15.2 – Color Pane.

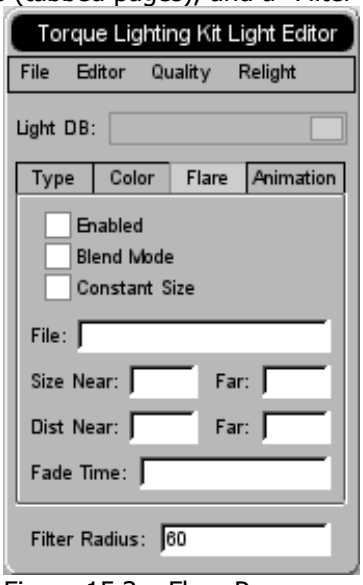

Figure 15.3 – Flare Pane.

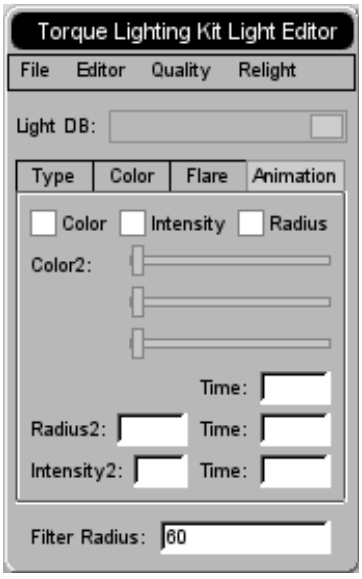

Figure 15.4 – Animation

Pane.

These controls each serve the following purposes.

- **Light DB Dropdown** – Clicking this dropdown allows you to select a datablock for editing. Simply choose the datablock you wish to edit from the list.
- **Filter Radus Field** – This field controls the distance (from the camera) for filtered (partial) relights. The smaller the radius value the faster the relight, and the fewer lights are recalculated.
- **Type Pane (Figure 15.1 above)** – This pane contains a "Lighting Model" dropdown menu, a series of option buttons, and a "Radius" field.
    - **Lighting Model Dropdown** – This menu allows us to select from one of six lighting models to use for lights using the current TLK datablock. (Seeee "15.2.7 Lighting Models" below for a full description of these models.)
    - **Lighting Options** – This list of options is actually a context-sensitive editor that allows you to configure the current TLK datablock. That is, based on which options you select, some other options may be enabled or disabled. (Read through "15.2 Advanced Lighting" to learn about each of these features/options.)
    - **Radius Field** – This field controls the radius to which the light using this TLK datablock will extend.
- **Color Pane (Figure 15.2 above)** – This pane allows you to adjust the color of the current TLK datablock. It includes three color sliders (red, green, and blue, from top to bottom), an intensity slider, an ambient (contribution) slider, and an angle slider (for spotlights).
- **Flare Pane (Figure 15.3 above)** – This pane allows you to edit the flare settings for the current TLK datablock. The options it presents are as follows.
    - **Enabled** – Enables or disables flare feature.
    - **Blend Mode** – Enables or disables blending.
    - **Constant Size** – If true, the flare size stays the same, regardless of the distance from which it is viewed. If false, the size will change based on the "Size Near", "Size Far", "Dist Near", and "Dist Far" fields.
    - **File** – Path and filename of flare graphic file to use for this flare.
    - **Size Near/Far** – Relative to render flare for camera near/far conditions.
    - **Dist Near/Far** – Distances (from light) at which camera is considered near or far.
    - **Fade Time** – Relative time (in seconds) over which flare will adjust size.
- **Animation Pane (Figure 15.4 above)** – This pane allows you to control whether the light using the current TLK datablock animates color, intensity, and/or radius. If one of the three check boxes at the top is selected, the specified feature will animate. In addition to the three checkboxes, this pane also has the following controls.
    - **Color2 Sliders** – Red, green, and blue settings for animated color. If color animation is enabled, the light will animate between the initial color (selected on color pane) and Color2.
    - **(Color2) Time Field** – The first field labeled "Time" below the Color2 sliders controls the time it takes for color animation to transition from Color to Color2 and back to Color.
    - **Radius2 / Time Fields** – Further down the pane are two fields for controlling the max radius (Radius2) and the time it takes for an animating radius to

transition from Radius to Radius2 and back to Radius. (Radius is specified on the Type pane.)

- **Intensity2 / Time Fields** – At the bottom of the Animation pane are two fields to control the maximum intensity and the duration of an intensity animation. (Remember that initial intensity is specified on the Color pane.)

The Cinematic Filter Editor Panes

In Cinematic Filter Editor (editing `sgMissionLightingFilterData` datablocks) mode, the editor presents a "Light DB" dropdown menu, two panes (tabbed pages), and a "Filter Radius" field.

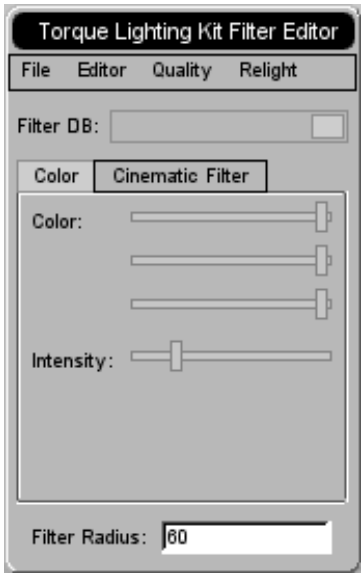

Figure 15.5 – (Default Filter) Color Pane.

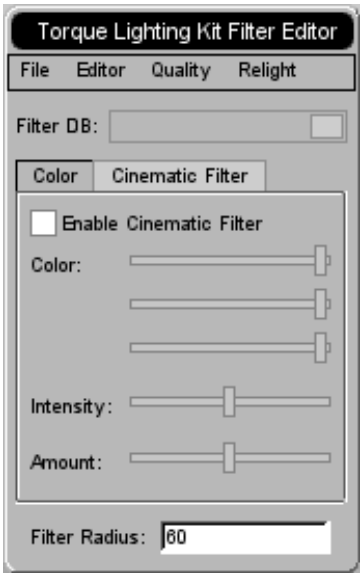

Figure 15.6 – Cinematic Filter Pane.

These controls serve the following purposes.

- **Light DB Dropdown** – Clicking this dropdown allows you to select a datablock for editing. Simply choose the datablock you wish to edit from the list.
- **Filter Radus Field** – This field controls the distance (from the camera) for filtered (partial) relights. The smaller the radius value the faster the relight, and the fewer lights are recalculated.
- **(Default Filter) Color Pane (Figure 15.5 above)** – This pane allows you to modify the default filter's red, green, blue, and intensity values for the current TLK cinematic filter datablock. (Read "15.3.1 Lighting Filters" to learn more about cinematic filters.)
- **Cinematic Filter Pane (Figure 15.5 above)** – This pane allows you to modify the cinematic filter's red, green, blue, intensity, and amount values for the current TLK cinematic filter datablock. These values only apply if the "Enable Cinematic Filter" checkbox is selected; otherwise, the default values from the Color Pane are in effect. (Read "15.3.1 Lighting Filters" to learn more about cinematic filters.)

15.1.4 Creating TLK Lights

Now that you know the basics about editing TLK datablocks, you're ready to create and place TLK lights.

Actually, this is the easy part. Creating and placing a light is just like creating and placing any other mission object. Simply do the following:

1. Open the mission you want to add TLK lights to.
2. Start the World Editor (F11)
3. Start the Creator (F4)
4. Open the Creator tree to "Mission Objects -> Environment" and select the TLK light object you want to add to your mission. (See Figure 15.7 below.)
5. Switch to the Inspector (F3) and modify any fields in the newly placed TLK light.
6. Save (CTRL + S).

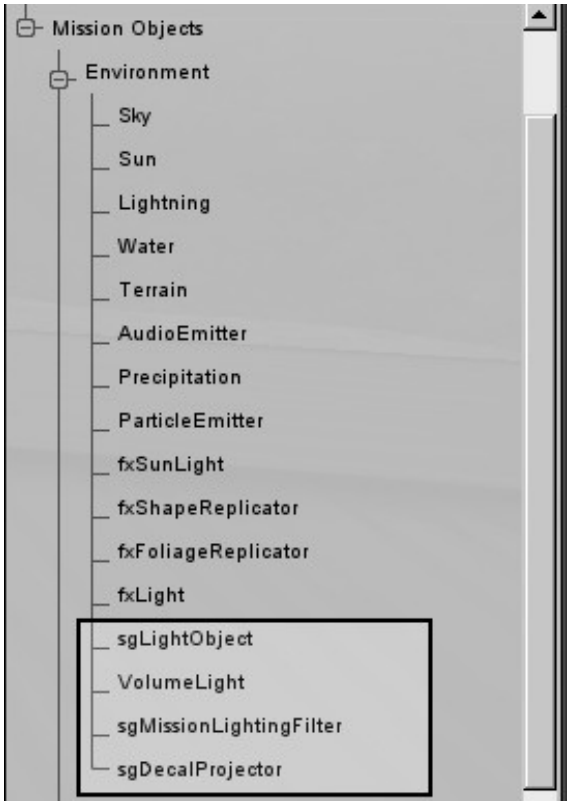

Figure 15.7 – TLK Light Objects (in Creator Menu).

15.2 Advanced Lighting

15.2.1 Static vs. Dynamic Lights

Unlike 3D modeling applications, most game engines differentiate between static and dynamic lights in order to optimize performance and still provide high-quality graphics. There are several distinct differences between static and dynamic TLK lights, and understanding the pros and cons of each is critical to keeping game performance and quality high.

Static TLK lights bake lighting into interior and terrain light maps (LMs). Because these light maps are always used, static lights are rendered "for-free." Static lights do, however, increase the time it takes to relight the scene. Because processing happens during relight and not while rendering, static lights can create far more realistic lighting and shadows but cannot move or be animated.

Dynamic TLK lights are calculated at render time, which creates overhead but allows the lights to move and be animated. Dynamic lights do not cast shadows and don't blend well on the terrain.

15.2.2 Scene Relighting

TLK provides a number of tools that speed up mission relighting and keep workflow as efficient as possible. These include filtered relighting, lighting profiles, and the ability to disable shadows.

Filtered relighting improves performance by only relighting interiors in the vicinity of the camera. Filtered relighting is controlled from the TLK Light Editor from where the filter radius is set and relighting is selected from the menu item "Relight -> Filtered Relight". While the Light Editor is open, interiors within the filter radius (interiors that will relight) are highlighted with a pink bounding box.

TLK's lighting profiles speed up performance by reducing lighting accuracy. This is perfect for calculating quick-and-dirty lighting on newly created interiors or for quickly tweaking light objects. TLK has three lighting profiles: Production, Design, and Draft, which are selected through the Light Editor using the "Quality" menu item and range from the highest to lowest quality. The selected profile is persistent, so lighting performance can be boosted during the relight that occurs on mission load.

TLK also allows shadow casting to be disabled globally, which increases performance considerably. Shadow casting is toggled through the Light Editor using the "Quality" menu item and is also persistent.

As an example of the performance gains accrued when using these options, Table 15.1 shows the relight times for an early and highly unoptimized version of the TGE *WarZone* demo (see Figure 15.8 below).

Profile	**Shadows**	**Time (full relight in seconds)**
Production	Enabled	115.7
Design	Enabled	45.6
Draft	Enabled	27.7
Production	Disabled	16.1
Design	Disabled	10.6
Draft	Disabled	9.6

Table 15.1 – TGE *WarZone* Lighting Times.

Figure 15.8 – TGE *WarZone* Demo.

15.2.3 Ambient Lighting

In addition to direct lighting, TLK lights can also contribute attenuated ambient lighting to the scene. This is particularly helpful in simulating radiosity without the enormous calculation times normally associated with it. Attenuated ambient lighting is different from classic ambient lighting in that it decreases over distance so it doesn't create flat lighting.

The ambient contribution of a light is set using the Ambient slider in the TLK Light Editor and fades between direct and ambient lighting. The ambient contribution can also be two-sided (illuminates surfaces facing away from the light) using the Double Sided property. Double-sided ambient lighting is helpful in simulating the indirect lighting that occurs when light reflects off of an object and onto the reverse side of another.

Ambient lighting does not cast shadows.

15.2.4 Zone Lighting

Zone lighting solves the issue of light bleeding into adjacent interior rooms when using dynamic lights and attenuated ambient lighting.

Because dynamic and ambient lighting does not cast shadows, the lighting passes through objects and walls. Zone lighting restricts illumination to the zone or zones (up to two) that contain the source light object, which can give the illusion that shadows are cast.

Zone lighting is controlled from the TLK Light Editor using the properties "Restrict Diffuse to Zone" and "Restrict Ambient to Zone", which restrict the direct (diffuse) and ambient lighting to the light's zones.

Restricting static lights to their zones also boosts relighting performance by limiting potential illuminated surfaces.

15.2.5 Mounting Lights

TLK lights actually attach to objects instead of mounting to them, which leaves the object's mount slot open. Lights are attached to objects in script by calling:

```
<light object>.attachToObject(<target object>);
```

The light determines the mount point and offset (both position and rotation) based on the datablock members MountPoint, MountPosition, and MountRotation.

Figure 15.9 – Mounted Light

15.2.6 Linking Lights to Particle Systems

Dynamic lights can be linked to particle emitters so the light's color and intensity is relative to that of the emitter's particles. This creates very realistic particle effects such as fire, plasma, and sparks.

Lights are linked to particle systems by setting the light object's "ParticleEmitterNode" property to the name of the target particle emitter node. The particle emitter node's name must be unique to all other mission objects, so the correct object can be found, and the target must be a particle emitter node (not just an emitter) otherwise the light will not link to the particle system.

When linked, the light object's color and intensity are tied to the particle system; however, the overall lighting intensity can be further adjusted using the light object's "ParticleColorAttenuation" property.

15.2.7 Lighting Models

Lighting models create an engine's distinctive look, feel, and style. To offer the most flexibility, TLK offers six built-in lighting models, as well as the ability to create your own. Getting a feel for TLK's lighting models makes it easier to achieve a specific look in your game.

The following discusses TLK's lighting models and explains how the lighting models interact with the light's color, intensity, and radius values.

- **Original Stock**- Introduced in the original Lighting Pack, this is the default Lighting Model. This model is unique in that it has no maximum intensity - the light's color and intensity attenuate its intensity, but they do not create a ceiling (normally the color and intensity act as an upper limit) - this makes the model easy to overexpose. The radius represents the distance at which the intensity starts to fall off.

The Stock model is extremely useful when trying to bring a lot of light into a limited space, using spotlights, or any application where intense lighting is needed to highlight objects and create focal points in the scene. Most models fill a scene with light, but the Stock model is designed to put lighting into focused areas.

- **Original Advanced** - introduced in the original Lighting Pack, it was the first alternative lighting model and provides radiosity-like lighting (very smooth and gradual). The color and intensity represent the maximum value the light can ever achieve. The radius represents the outer edge of the light's influence.

The Advanced model is useful for flooding large areas with light.

- **Inverse Square** - introduced in Lighting Pack version 1.3.5, this model is meant as an alternative to the Advanced model. The color and intensity represent the maximum value the light can ever achieve. The radius represents the distance at which the intensity falls to half, and the lighting continues to drop off gradually for a considerable distance.

The Inverse Square model was introduced as an alternative to the Advanced model that provides falloff characteristics that blend more smoothly into existing lighting. Generally speaking, the Inverse Square model is better suited for use in scenes with complex lighting, whereas the Advanced model provides a better single light solution.

- **Inverse Square Fast Falloff** - same as the Inverse Square model except the lighting intensity falls off to practically nothing at the radius.

- **Near Linear** - same as the Inverse Square model except the falloff is linear. Frankly this is the least flexible model. Although it works well in multi-light scenes, the lighting isn't very believable. Linear lighting does work well in conjunction with a high local ambient value and works very well for creating a background lighting to layer more complex lighting on top of.

- **Near Linear Fast Falloff** - same as the Near Linear model except the lighting intensity falls off to practically nothing at the radius.

15.2.8 Overexposure and Baked in Highlights

When adding TLK to an existing project, you'll notice that the existing levels, scenes, and assets look far too bright. TLK creates brighter more vibrant scenes by doubling Torque's lighting intensity, so restoring the pre-TLK lighting levels is as simple as cutting the color value of the sun and the interior light entities in half. From there you can go back and increase the level of individual lights to your liking.

Figure 15.10 – Overexposure.

TLK also overexposes self-illuminating materials, which can leave them far too bright. There are two ways to fix this: either disable overexposure on self-illuminated materials (talk to your development team regarding this compile-time option), or cut the texture intensity in half (assuming there are only one or two offending textures).

TLK's overexposure makes the lighting detail baked into diffuse textures, such as bump and specular highlights, really shine. Artificial or exaggerated highlights can be added to existing diffuse textures, but be careful not to overdo the effect. Generally try to avoid baking in large bump detail, and make sure that specular highlights face a visible light source when placed in levels, otherwise the effect looks faked.

15.2.9 DTS Object Shadows

TLK allows DTS objects to cast both static and dynamic shadows. Torque has many different types of DTS objects (TSStatic, Player, Items, ...), and each casts a different type of shadow. Table 15.2 shows the shadows cast by the base object types.

Object Type	Light Type	Shadow Type
TSStatic	Static	Static
	Dynamic	None
ScopeAlwaysShape	Static	Static
	Dynamic	Projected (Dynamic)
ShapeBase Derived	Static	Projected (Dynamic)
	Dynamic	Projected (Dynamic)

Table 15.2 – Shadows Type Results for Object vs. Light Type.

Each shadow type has its pros and cons that should be considered when placing static DTS objects in levels; deciding on the shadow type will dictate the object type to use.

Shadow Type	Pros	Cons
Static	1. No rendering overhead.	1. Fuzzy and inaccurate. 2. Based on collsion mesh.

Dynamic	1. Highly detailed.	1. High rendering overhead.

Table 15.3 – Shadow Types (Pros and Cons)

Static DTS objects have the option to disable animation and movement of shadows (optionally the shadow can be completely disabled) to minimize rendering overhead.

TLK's multiple dynamic DTS shadows can be disabled by setting the preference global `$pref::TS::sgMultiDynamicDTSShadows` to false. This reverts the shadows back to the older style single shadow.

The detail of dynamic DTS shadows can be changed by adjusting the pref `$pref::TS::sgShadowDetailSize`. TLK ships with seven shadow detail levels, which are chosen by using the detail index 0 - 6 (the lower the index the higher the detail).

These options are highly effective in tuning performance.

5.2.10 TLK Entities

TLK adds a new entity type to map2dif called `sgLightingScaleEntity`, which is a brush entity (similar to the portal entity, but it's applied to a brush) that changes the brush's light mapping scale (how large or small its light maps are) and can add self-illumination to the brush.

Changing the lighting scale per-brush can place as much or as little lighting detail right where it's needed. For instance, to place a highly detailed brush right where a shadow falls, or a very low detail brush where no lighting is visible.

15.3 Filtering Lights

15.3.1 Lighting Filters

TLK lighting filters can adjust the intensity and color of lights within an interior zone or the entire mission.

Once created, the filter is placed in the zone it's to control, or outside (zone zero) as the default filter. The filter will automatically start filtering light when created. Adjusting the filter's color and intensity controls will change the affected lights (lights in the controlled zone).

To make it easy to control lights without having to create a filter in every zone, lights look for a filter in their own zone first, and if not found, zone zero is checked, so placing a filter in zone zero creates a "default" filter.

15.3.2 Cinematic Filtering

TLK's filters can also create cinematic effects that simulate the "temperature" and "color balance" used in film. Temperature has little to do with heat and is instead similar to intensity. Color balance is achieved when the temperature of the scene lighting matches the temperature of the film. If the lighting temperature is highe,r the scene is overexposed and appears more blue in tint; if it's lower, then the scene is underexposed and appears more red/orange in tint.

TLK simulates this by using the color and intensity of the light and filter as the temperature of the light and film to calculate the color tinting to apply to each light.

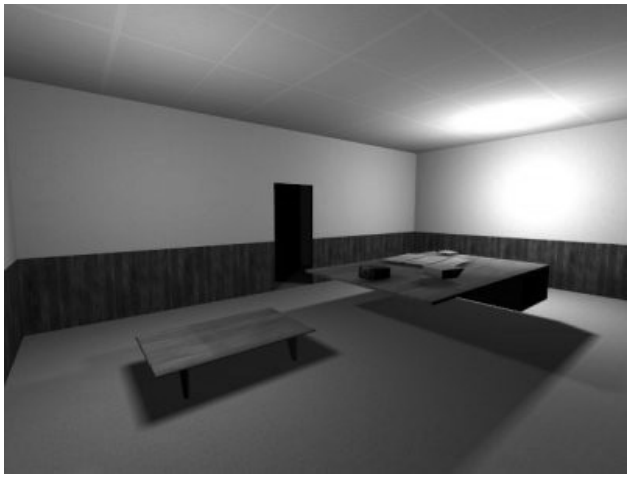
Figure 15.11 – Cinematic Lighting.

Cinematic filtering is enabled in the Filter Editor's Cinematic panel. Once it is enabled, the color and intensity affect the zone's temperature (equivalent to the film temperature) and the Amount property adjusts how heavy the exposure effect is.

15.4 DTS Lighting

15.4.1 DTS Lighting Sources

DTS objects have three potential light sources: diffuse, ambient, and self-illumination. Diffuse and ambient lighting can come from a combination of light objects and environmental lighting (from the interiors and the terrain), while self-illumination is applied during the modeling process.

TLK's advanced DTS lighting options allow different combinations of environmental and custom lighting to be used for the diffuse, ambient, and self-illumination sources. For instance, some of the more common environmental lighting sources are the sun and terrain/interior lighting. Table 15.4 is a list of the possible environmental lighting sources and their settings.

DTS Option											
Receive Sunlight	off	off	on	-	off	off	on	-	off	off	on
Receive Light Map (from terrain or interior LM)	off	on	-	-	off	on	-	-	off	on	-
Use Adaptive Self-Illumination	off	off	off	off	on	on	on	on	on	on	on
Use Custom Ambient	-	-	-	on	off	off	off	on	on	on	on
Custom Ambient Self-Illumination	-	-	-	off	-	-	-	off	on	on	on
Resulting Lighting											
Diffuse	none	none	sun color	none	none	none	sun color	none	none	none	sun color
Ambient	none	LM	sun ambient	custom	none	LM	sun ambient	custom	non	LM	sun ambient
Self-Illumination	full	full	full	full	none	LM	sun color	custom	custom	custom	custom

Table 15.4 – Lighting Sources and Resulting Lighting.

Stronger directional lighting can be achieved from environmental lighting by setting the pref `$pref::OpenGL::sgDynamicDTSVectorLighting` to true.

More stylized lighting can be achieved from environmental lighting by adjusting the pref `$pref::OpenGL::sgDynamicDTSShadingDiffuseAmount`. This value balances the ratio of diffuse to ambient lighting (this is disabled when using `$pref::OpenGL::sgDynamicDTSVectorLighting`).

15.4.2 Direct vs. Linked Lighting

In addition to environmental lighting, static DTS objects can receive lighting directly from light objects. There are two ways to achieve this: direct and linked lighting.

Direct lighting occurs when a light object's datablock is set to affect DTS objects. The light directly illuminates DTS objects and is blended with any lighting already affecting the object.

Linked lighting occurs when a light is not set to affect DTS objects and it, or its containing group, is added to a static DTS object's "lightGroupName" property, which links the DTS object to a light or lights, causing it to be illuminated by a light that otherwise would not illuminate it.

Light linking is especially helpful in situations where DTS lighting shines through obstructions (DTS lighting doesn't cast shadows) and zone lighting cannot be used. Instead, the lights are set to not affect DTS objects, and any object that should receive lighting is hand linked to the lights.

Multiple light objects and group names can be linked to a static DTS object by separating the name with semicolons (";").

15.5 Miscellaneous Lighting Features

15.5.1 Glowing Decals and Projectors

Decals can be configured to glow by setting their datablock property "SelfIlluminated" to true.

TLK also adds decal projectors that allow decals to be aimed and projected onto the first interior or terrain surface encountered. Decal projectors are found in Torque's Object Creator under "Mission Objects -> environment -> sgDecalProjector".

15.5.2 Illuminated Particle Systems

Particles can be illuminated by setting their datablock property "AllowLighting" to true. Particle illumination can be globally disabled by setting the pref `$pref::OpenGL::sgDynamicParticleSystemLighting` to false.


Figure 15.12 – Illuminated Particles.

15.5.3 Interior Detail Mapping

Detail mapping makes interior surfaces look highly detailed even when viewed up close without resorting to excessively large textures. Detail mapping works by blending a grayscale texture with baked-in lighting on top of interior surfaces. Though both the diffuse and detail textures are relatively small, the blending creates significantly more detail than either of the individual textures. Equally as important, detail textures can be shared among several diffuse textures, minimizing memory and disk usage.

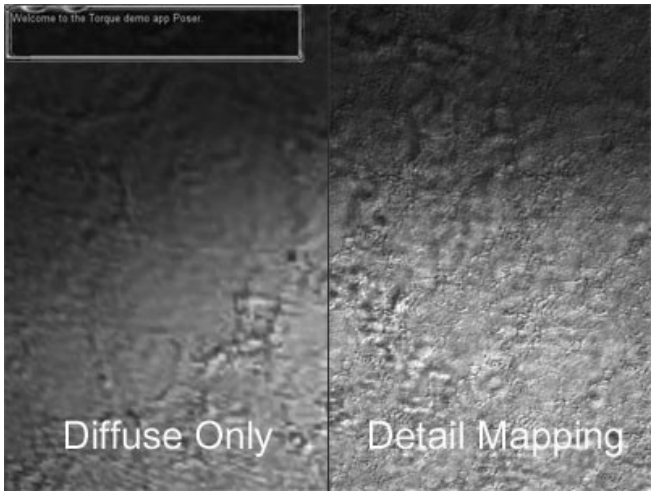

Figure 15.13 – Diffuse Only vs. Detail Mapping.

Detail textures are grayscale images containing baked-in bump data with an average intensity of 127 (out of 255). It's best to avoid distinct patterns and heavy detail, otherwise tiling can become apparent.

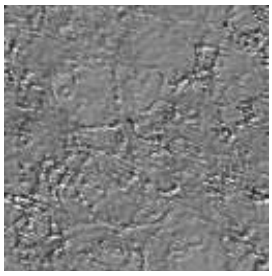
Figure 15.14 – Sample Detail Texture.

Detail textures are applied to interior surfaces with Torque's built-in material system using the following script call:

```
addMaterialMapping("<diffuse texture name>" , "detail: <full path to
detail texture>");
```

Which is usually called in "~/data/ini.cs" so it's mapped on the client and the server. This call maps the referenced detail texture to the material that contains the target diffuse texture. Whenever the diffuse texture is applied to an interior surface, the detail texture is also applied.

Detail mapping only works on video cards with four or more texture units (TGE's DirectX wrapper only supports two texture units, so detail mapping is unavailable in DirectX mode).

15.6 Performance Considerations

15.6.1 Dynamic DTS Shadows

Dynamic shadows create both rendering and CPU overhead, so it's a good idea to limit the usage of dynamic shadows. The easiest way to do this is to understand which object types cast dynamic shadows and to use them sparingly. Disabling animation and movement on static object shadows also helps a great deal.

DTS shadows can also be adjusted in-game to balance performance with graphic quality. The largest performance boost is achieved by disabling multiple DTS shadows, which significantly cuts down on the number of rendered shadows. Lowering the shadow detail level also helps, especially on older systems.

For more details on DTS object shadows, see the section "Advanced Lighting - DTS Object Shadows."

15.6.2 Dynamic Lights

Dynamic lights have a high rendering overhead especially when compared with static lights, which are rendered "for-free." Though dynamic lights are very usefully in many areas, use them sparingly, try to keep them spread out, and avoid large-radius dynamic lights at all costs.

15.6.3 Limit Dynamic DTS Lights

DTS objects are lit dynamically by all lights, even static lights (this has to do with the way DTS objects are rendered). As more and more lights illuminate a DTS object, the rendering performance decreases, especially on older cards. To avoid this, use Torque's pref

`$pref::OpenGL::maxHardwareLights` to set an upper limit on the number of lights that can affect each DTS object.

15.6.4 Mix TLK Map Lights

As more and more static lights are used, mission relight times increase. To minimize this, bake less important background lighting into the interiors using map light entities.

15.6.5 Ship Mission Lighting Files

To avoid the initial relight that occurs when running a mission for the first time, ship the mission lighting files with your demos and games.

15.6.6 Use a Release Build

Release builds of Torque run anywhere from 2 to 4 times faster than debug builds, so always try to use a release build.