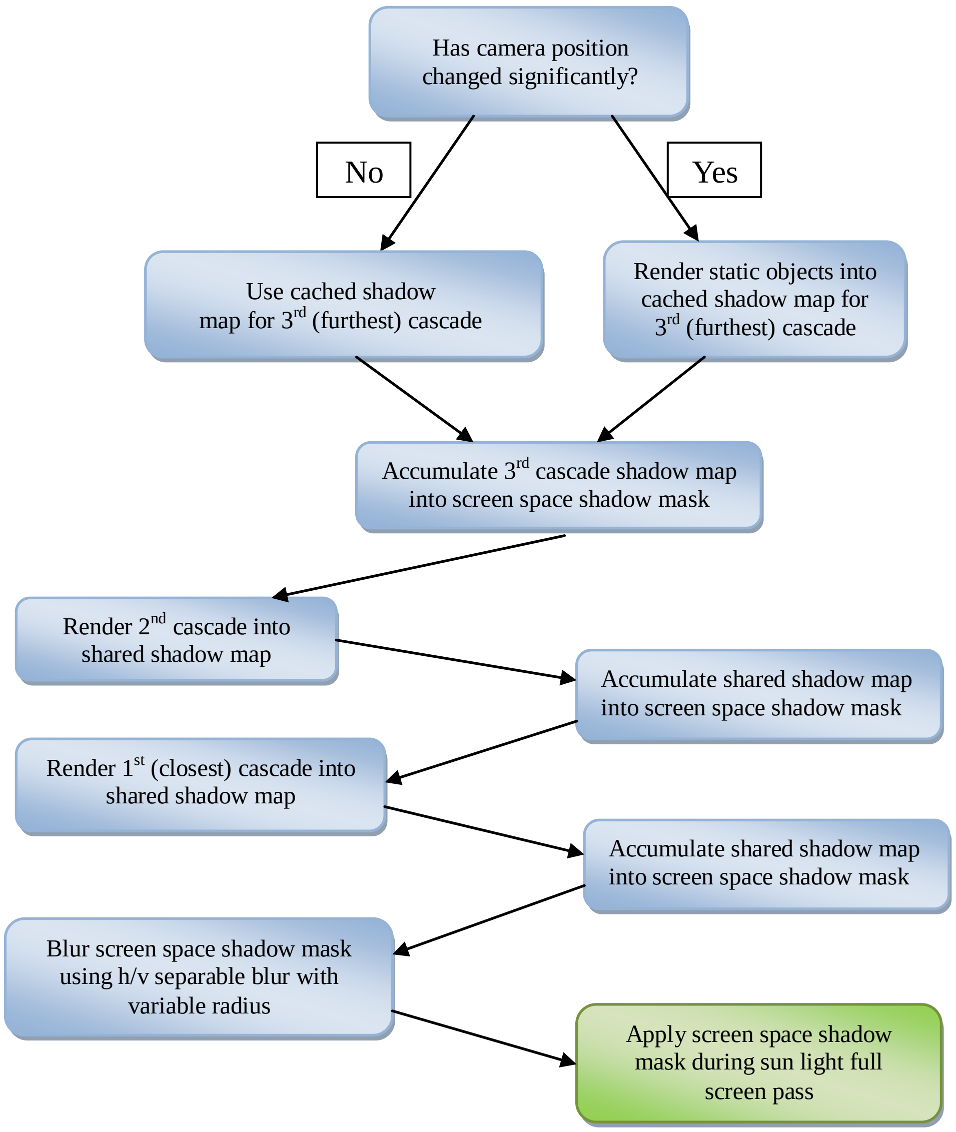
Shadows

Description

Sun Shadows

For sun shadows, the engine utilizes cascaded shadow mapping. Cascade area is optimized with respect to shadow visibility to ensure maximum possible resolution. The number of active cascades depends on quality settings. The furthest cascade is cached and is updated only when camera position changes significantly, ensuring excellent shadowing performance. On low shadow quality settings, which don't require advanced blurring, hardware PCF is used. On medium quality settings, advanced hardware PCF driven filtering is used. On high shadow quality settings, custom screen space blurring filter is applied. This filter calculates penumbra softening based on distance from the caster to the occluded object among other things.

High quality sun shadow pipeline looks like this:


Has camera position changed significantly?
No
Yes
Use cached shadow map for 3rd (furthest) cascade
Render static objects into cached shadow map for 3rd (furthest) cascade
Accumulate 3rd cascade shadow map into screen space shadow mask
Render 2nd cascade into shared shadow map
Accumulate shared shadow map into screen space shadow mask
Render 1st (closest) cascade into shared shadow map
Accumulate shared shadow map into screen space shadow mask
Blur screen space shadow mask using h/v separable blur with variable radius
Apply screen space shadow mask during sun light full screen pass

Spot light shadows
Spot light shadows are calculated using shadow mapping. Hardware PCF feature is applied on low quality settings. It is possible to setup screen space shadow blurring with penumbra softening individually for each light for high quality settings.

Spot light shadows have freeze option, which renders shadow depth into L16 texture and uses it for shadowing instead of dynamically generated shadow maps. This option is only applicable for the lights which position and orientation doesn't change over time. Only static objects are cast into the frozen shadowmap.

Point light shadows
Point light shadows are rendered into a cubemap texture of R32F format. It is possible to setup screen space shadow blurring with penumbra softening individually for each light for high quality settings.

Spot light shadows have freeze option, which renders shadow depth into L16 cube texture and uses it for shadowing instead of dynamically generated shadow maps. This option is only applicable for the lights which position doesn't change over time. Only static objects are cast into the frozen shadowmap.

Shadows for transparent primitives.

Transparent primitives (e.g. particles) are rendered after deferred lighting has been applied to opaque scene components. Thus, it is shadowed separately. Last, cached shadow slice is used for shadowing, insuring optimal perfomance. This means that only static geometry's shadows affect the particles.

Aliasing artifacts

Aliasing artifacts are fought with using 2 techniques. First. where it is possible, inverse culling is used for shadow casting. Second, sample is being displaced based on the direction of the normal of the mesh. For normals, direction of which is close to being perpendicular to light's direction, greater displacement is used.

Temporal aliasing artifacts

Temporal aliasing artifacts occur from frame to frame and are caused by camera position and orientation changes in relation to the light source. A sub pixel displacement is applied to shadow map projection matrix in order to counter these artifacts.

Sun shadow visibility

Visibility of the shadow casters for sun shadows is determined in the following manner. Corners of object's bounding box are projected on the plane perpendicular to light's direction. Bounding rectangle is build around this points. Then, this rectangle gets “extruded” in the direction of the light. The length of the extrusion is either determined by the intersection with level's terrain, or with levels “shadow extrusion limit”, which is configurable in the editor. The “shadow extrusion limit” defines a plane, further than which the boxes are not extruded. The resulting box is checked against main frustum for visibility. Another limitation for extrusion is defined via *r_shadowcull_extrude* variable. Shadow extrusion cannot exceed this length.

Associated code

RenderShadowMap(..)	Function that performs shadow rendering. Several overloads exist for different lighting types.

BlurShadowMap(..)	Function that performs screen space shadow blurring with penumbra distance based softening.
RenderShadowScheme(..)	Renders shadow slice distribution scheme for debug purposes.
DS_AccumShadows_ps.hls	Accumulates shadow cascades into a screen space shadow mask with extra data for distance based penumbra blurring.
DS_BlurShadows_ps.hls	Blurs screen space shadow mask with distance based penumbra blur radius.
DS_Dir_ps.hls	Sun (directional) light shader. Applies screen space shadow mask, constructed using *DS_AccumShadows_ps.hls*
DS_DirL_ps.hls	Low quality variant of *DS_Dir_ps.hls*
DS_Spot_ps.hls	Spot light shader. In certain configurations it applies the shadows. If advanced blurring is enabled for the shadows, screen space shadow mask is applied. Otherwise, the shader compares samples with shadow map to account for shadows.
DS_Point_ps.hls	Point light shader. In certain configurations it applies the shadows. If advanced blurring is enabled for the shadows, screen space shadow mask is applied. Otherwise, the shader compares samples with shadow map to account for shadows.
r_shadows	Config variable that defines whether shadow rendering is enabled or not. This variable affects all shadow casting: sun shadows, point light shadows etc.
r_particle_shadows	Config variable that defines whether shadow particles are allowed to receive shadows.
r_shadowcull	Config variable that defines whether to perform shadow visibility culling as described in **“Sun shadow visibility”** topic above
r_shadowcull_extrude	Config variable that defines maximum allowed distance of object's box extrusion for shadow visibility culling.
r_shadow_extrusion_limit	Config variable that defines the plane on the level, below which extrusion of object's boxes doesn't occur for shadow visibility culling
r_do_dyn_shadow_blur	Config variable that defines whether to allow dynamic branching in shadow blur shader. On highest quality settings this branching is disallowed forcibly because it produces shadowing artifacts in some cases.
r_shadows_blur_phys_range	Config variable that defines the distance between the least and most amount of blurring applied to penumbra.
r_shadows_blur_bias	Config variable that defines the minimal amount of blurring applied to penumbra
r_shadows_blur_sense	Config variable that defines depth sensitivity of shadow blurring. When this value is high, sharp depth transitions prevent shadows from leaking into “unrelated geometry”. However, steep planes with respect to camera will also get blurred less.
r_shadows_blur_radius	Config variable that defines base shadow blur radius.
r_active_shadow_slices	Config variable that defines the number of active shadow slices. This number is 1 for the lowest shadow quality settings, 2 for medium, 3 for high and ultra.
r_precalc_shadowcull	Config variable which controls whether object's box extrusion precalculation is allowed. At cost of some extra memory, a lot of CPU can be saved because most of shadow casters do not move.

r_inst_precalc_shadowcull	Config variable which controls whether object's box extrusion precalculation is allowed for instanced objects
r_show_shadow_scheme	Config variable which controls whether debug shadow scheme should be drawn(*RenderShadowScheme)*
r_optimize_shadow_map	Config variable which controls whether shadow map space optimization is allowed to accomplish better shadow map resolution usage.
r_shadow_low_size_coef	Config variable which controls how much is the shadow rendering distance is expanded for the first and only slice for low shadow quality settings

Associated Source Files

RenderDeffered.cpp	Contains definitions of *RenderShadowMap(..), BlurShadowMap(..), RenderShadowScheme(..)*
DS_AccumShadows_ps.hls	Source of the shader that accumulates shadow maps into screen space shadow mask
DS_BlurShadows_ps.hls	Source of the shader that blurs screen space shadow map.
DS_Dir_ps.hls	Source of the sun lighting shader.
DS_DirL_ps.hls	Source of the sun lighting low quality shader.
DS_Spot_ps.hls	Source of spot lighting shader
DS_Point_ps.hls	Source of point lighting shader
Vars.h	Among other things, contains declarations of shadow related config variables

struct ShadowSlice

Summary:

Describes shadow slice (cascade).

This structure has the following fields:

index	*int*	Index of the shadow slice.
depthBias	*float*	Depth bias to apply for the shadow slice to reduce aliasing artifacts
depthBias_HW	*float*	Separate depth bias settings for hardware shadow mapping(advanced normal based biasing is not supported with this method)
shadowMapSize	*float*	Shadow map size in pixels
lightView	*D3DXMATRIX*	Light's view matrix
lightProj	*D3DXMATRIX*	Light's projection matrix.
camPos	*r3dPoint3D*	Camera position
sphereRadius	*float*	Radius of the sphere that encompasses the shadow map in world units
worldSizeX	*float*	Size of the shadow map along light space X axis
worldSizeY	*float*	Size of the shadow map along light space Y axis

pixelDiameter	*float*	Diameter of the of the shadow map pixel in world units

Some of these variables are updated every frame (*lightView, lightProj, camPos, sphereRadius, worldSizeX, worldSizeY, pixelDiameter*), while others do not change within the same level (*index, depthBias, depthBias_HW, shadowMapSize*)

---

void RenderShadowMap(ShadowSlice& slice)

Summary:

Renders shadow map for sun shadows and for shadow slice (cascade) specified as *slice.*

Parameters:

slice	- slice structure to render shadows for.

---

void RenderShadowMap(const r3dCamera& lightCam)

Summary:

Renders shadow map for light camera *lightCam*. This function is used for rendering spot light shadows, and for rendering individual faces of point light shadow cubemap.

Parameters:

lightCam	- light camera to use for shadow map rendering.

```
void FillSliceForSplitDistances(    ShadowSlice* ioSlice, r3dPoint3D* oLightSource, r3dPoint3D* oLightTarget,
                                    float fNear, float fFar, bool allow_optimize )
```

Summary:

Updates dynamic fields of the slice pointed to by *ioSlice* with new values. This function is called from *RenderShadowMap(ShadowSlice& slice)* function.

Parameters:

ioSlice	- pointer *ShadowSlice* structure to fill
oLightSource	- pointer to the vector to fill with light source point
lightTarget	- pointer to the vector to fill with light destination point
fNear	- light camera near plane
fFar	- light camera output plane
allow_optimize	- if this parameter is true, optimizes cascade frustum taking actual shadow visibility into account.