

Mirrors with realtime reflections

Table Of Contents

- [1 General information](#)
- [2 Execution](#)
 - [2.1 Blender](#)
 - [2.2 Script](#)
 - [2.3 ContentTool](#)

1 General information

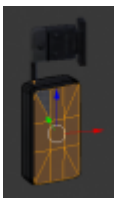
Regarding the real-time reflections for (rear) [mirrors](#), the following things should be noted:

- For each rear-view [mirror](#), a separate entry must be created under "Real-time [mirrors](#)". However, each entry there costs performance as long as the corresponding [mirror](#) is in the field of view, since it represents a separate rendering process. If two [mirrors](#) are close to each other and have a similar orientation, you can "trick" them by "splitting" a texture (by adapting the mapping of the individual [mirror](#) objects).
- The [mirror](#) camera is placed in the middle of the [mirror](#) and aligned perpendicular to the [mirror](#) surface. The calculation of the final angle based on the position of the user camera is done by LOTUS.
- Each [mirror](#) camera is rendered only if the center point (plus the specified radius) is within the field of view of the user camera. [Mirror](#) cameras are currently only rendered if the user is in an interior view.
- The created texture is explicitly used as "normal" texture! Not as reflection texture or similar! The simulation of the reflection is already done while rendering the texture.
- Likewise, all lighting effects of the [mirror](#) surface are deactivated, because the current lighting is directly derived from the rendered texture.

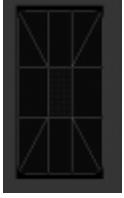
2 Execution

2.1 Blender

A typical rear view [mirror](#) can look like this:



The [mirror](#) surface must be given its own "dummy" texture. It should have a favorable aspect ratio (here 1:2), so that it can be mapped onto the [mirror](#) surface as full as possible:



Each [mirror](#) should have its own dummy texture so that separate materials can be created in the ContentTool.

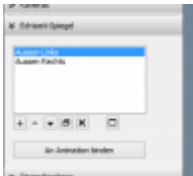
2.2 Script

In the script, an integer public variable named `mirrortex_#` (`#` = index of the [mirror](#), starting from 0) must be created for each [mirror](#) texture, into which LOTUS can write the texture ID and which can then be accessed by the [material properties](#). So if the vehicle has three [mirrors](#):

```
mirrortex_0: integer;  
mirrortex_1: integer;  
mirrortex_2: integer;
```

2.3 ContentTool

Now you have to create an entry for each [mirror](#) in the "Real-time [mirror](#)" section on the left:



Each [mirror](#) has the following parameters:

Eigenschaften Echtzeit-Reflexion

OK Abbrechen

Name:

Ursprung:

X:

Y:

Z:

Richtungs-Winkel:

Höhen-Winkel:

Öffnungswinkel (vertikal gemessen):

Ist eine Reflexion:

Radius des Spiegel-Objektes:

Textur-Breite (Pixel):

Textur-Höhe (Pixel):

- Name: Only for the list
- Origin: Center of the [mirror](#) surface and point from which the [mirror](#) camera "looks":



- Direction and elevation angle: The orientation of the [mirror](#) surface normals - in the following picture the blue arrow:



- Opening angle: How large is the field of view of the [mirror](#)?
- Is a reflection: No monitors are supported yet - but later this parameter will help to distinguish between [mirrors](#) and monitors (e.g. the perspective of the surveillance camera does not change when the perspective of the user camera changes 😊)
- Radius of [mirror](#) object: If the center of the [mirror](#) surface moves out of the field of view, LOTUS would immediately stop rendering. The rest of the [mirror](#) could still be in view and freeze visibly. To avoid this effect, this radius is specified as a tolerance limit.
- Texture width and height: The resolution of the texture into which the rendering is to be done. This automatically results in the aspect ratio with which the [mirror](#) image is to be rendered.

Finally, the material must be configured accordingly:



- "Direct Texture" is recommended as material type, because all light effects are already realized directly via the reflection texture. If it gets dark outside, the [mirror](#) image will darken by itself 😊
- For the texture, the check mark is set at the back and the corresponding "mirrortex_#" variable created in the script is selected

Done!