Programming a mirror in 3D scenery

by Graham Relf, April 2021

In thinking about an online challenge on a theme simply stated as "Mirror" I have developed a method of displaying a vertical mirror placed in 3-dimensional game scenery. This may be useful to others, so I am writing a detailed description here.

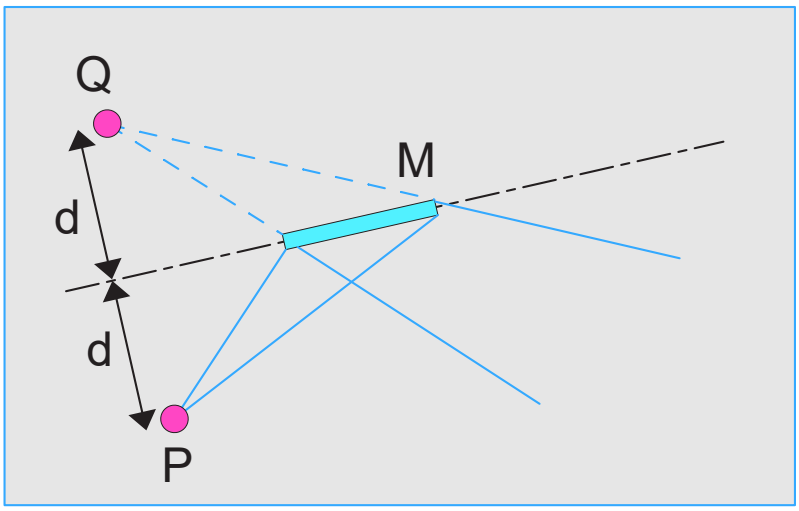
My starting point was a framework I had already written in plain JavaScript using a 2D canvas in an HTML5 page. A previous demonstration game, called "The Green", used my framework and can be seen at https://grelf.io/the-green.

Part of the challenge for me was adding the mirror without having to change any of my existing framework files.

First we need to consider the optical arrangement of a vertical mirror, for which a plan view may help.

An observer standing at P looks at mirror M. What is seen in the mirror is what an observer at Q would see beyond the mirror if it were not there.

Point Q is the same distance behind the plane of the mirror as point P is in front of it, by a vector which is normal to the plane of the mirror.



Step 1:

Find the unit vector normal to the plane of the mirror.

Calculate distance d, the shortest distance from P to the mirror plane.

Get point Q by offsetting P by a vector which is -2d x the unit normal.

Step 2:

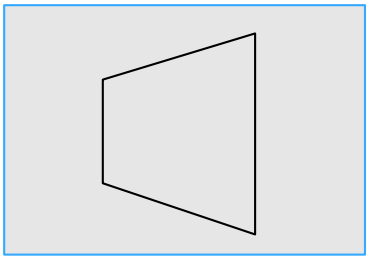
Draw the scene as viewed from Q in the direction of the centre of the mirror.

It is important not to include in the scene any objects nearer to Q than the mirror because any such objects would not be visible in the reflection.

Find the outline shape of the mirror in this view.

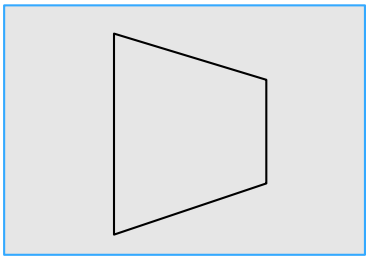
Save the pixel data within this outline, for use later in P's view of the mirror.

In Q's perspective view the outline of the mirror will be something like this but the mirror itself is not drawn in the scene.



It is important to note that if the observer at P turns around on the spot it will not change the view from Q towards the centre of the mirror.

The perspective view from P will show the mirror outline flipped horizontally, as seen here.



If the observer at P turns on the spot the exact shape of this will change. It is not always an exact reversal of the shape seen from Q. That would only be the case when P was looking at the centre of the mirror.

Step 3:

> Draw the scene as viewed from P.
>
> When showing the mirror use the pixel data saved from Q's view but flipped horizontally and scaled to fit P's view of the shape.

My framework is very much object-oriented. For this new development I created a JavaScript type called Mirror which inherits from an existing Polygon type which in turn implements an abstract type called Shape. I use a separate script file for each object type, so the new file is Mirror0.js, where the 0 suffix is a version number.

A drawback is that the scene has to be computed twice, as seen from positions Q and P, and then the contents of the mirror have to be copied from an off-screen view to the visible one. The result is much less responsive than my original program (The Green). The slow-down becomes more apparent as you approach the mirror because the area to be copied and scaled into the mirror becomes bigger.

The protagonist is not seen in the mirror: an invisible person. This is partly because revealing their appearance would spoil something else that can happen in the game.

It is possible to go through the mirror, in a certain way. The world then changes. There are some objects which are intended only be visible in the through-mirror world, but their switchable visibility and their subsequent behaviour have not yet been programmed.

The program with mirror can be run at https://grelf.itch.io/mirror