

community forums

Xbox LIVE Indie Games Forums » Interests » Game Design » Rendering top-down maze game with edge wrapping (game on a torus)

My Discussions Active Discussions Not Read

Search Forums

Advanced

Sort Discussions: Oldest to newest

Previous Discussion Next Discussion


Page 1 of 1 (6 posts)

# rendering top-down maze game with edge wrapping (game on a torus)

Last post 1/29/2008 3:43 PM by Darkside. 5 replies.

- 1/15/2008 12:43 AM

Hampoke



Rendering top-down maze game with edge wrapping (game on a torus)

Reply Quote

toroidal-shaped world. When the player (or an enemy) hits the max-y or max-x value, the player is transported back to the min-x or min-y value instead. If the world is flattened and viewed from above, this would mean that players move out of the left edge and reappear on the right, and vice-versa (and top to bottom.) I've seen a few solutions for handling the wrapping of the world objects (bullets, people) I have fewer solutions for rendering. Think "top-down maze game" or "pac-man" with a non-whole-map view" as examples of similar endeavours.

My question is vastly larger than the player's view port will be (my plan is to put the camera above the plane, looking almost straight down). I can't quite get my head around what happens when the viewing area starts showing that "edge" of the map. In a simpler world, the 2d sprite-based map rendering step would say "oh, rendering the map only filled up 300 pixels, I had better start over with the left edge and render the remainder." I'm fine with either using an orthographic or a tweaked perspective, the map 'ground' will be a textured terrain with walls that will stand out.

I'd like to get this same effect. When the camera moves such that the viewport shows the a map edge, the corresponding other edge is displayed as well. A few possibilities come to mind.


Generate the map as a series of pre-rendered textured quads. Near the logical map edge, tile more of the map pieces from the wrapping edge, replicating the game objects as necessary; when the camera crosses the edge, flip it back to the other edge and keep going. Depending on the FOV and height of the camera, this could be a few extra quads or a lot. Seems pretty easy to do.

When generating the walls, render the whole thing into a huge texture. Create the map surface as nine textured quads (one large enough to hold the entire map, the others wide enough to prevent the camera from seeing beyond as it approaches the edges) in a single mesh. Position the huge texture over the map-sized quad on the mesh and set the "wrap" flag for texturing. Replicate game objects as necessary when they are near an edge. Seems memory intensive as the size of the map will be ~16x wider than the viewport.

Thoughts? Anyone else done/doing a maze-game-with-wrapping?

How about something similar with either a spherical map or with polygonal shapes? I'd really like to keep the appearance of an overhead-view of a maze in those cases as well, but I have a harder time wrapping my head around the math and how characters will appear to move in the maze.
- 1/16/2008 4:20 PM

Subsonicz




Re: Rendering top-down maze game with edge wrapping (game on a torus)

Reply Quote

is just me, but I'm having trouble visualizing what you're trying to do. It might help if you throw up a screen cap or something to give us a better perspective of the problem.
- 1/19/2008 12:51 AM

Hampoke



Re: Rendering top-down maze game with edge wrapping (game on a torus)

Reply Quote

if that doesn't show, see also <http://cid-df933786e9eee3be.skydrive.live.com/self.aspx/xnacontent/XnaExample.gif>

here's a grossly simplified example. The player is the red dot. They can move anywhere x-y on the map in the usual way. When the player crosses the east edge of "F", they enter the west edge of "D". When they head north from "F" to "C" and out the north edge of "C" they enter by the south edge of "J". (And so on.)


I want the camera positioned over the player looking straight down. The red box is the outline of what the player can see on their screen. (For this simple example, the viewport is approximately 2 tiles wide.) In the illustration, note that the player is close to the east edge of F. At that position, they can see all of F, the south part of C, the north part of J, the east part of E, the northeast part of H, the southeast part of B, the southwest part of A, the northwest part of G, and the west part of D.

As the player moves across that east edge of F, the camera follows them into D and eventually shows part the south part of C, north part of J, more southwest of A, more northwest of G, and the west of D.

One way to render this is to replicate the geometry based on where the player is. Since the viewport is 2 tiles wide, if the player is in "F", then render "D" (and its contents) east of F, A east of C, and G east of J. If the player were in the southeast corner of J, then I'd render C south of J and G east of J.

Does that help?
- 1/19/2008 6:43 AM

Stainless



Re: Rendering top-down maze game with edge wrapping (game on a torus)

Reply Quote

The way I would approach this problem is to break down the world into patches. Each patch is rendered with the same textures using a multitexturing shader, look at any of the good terrain tutorials for an example. This will keep your texture size down to a manageable level.

Then I would use the same approach as has been used for 20+ years. Work out which patches are visible, then render them at a calculated x,y,z.

This would allow you to do any shaped world depending on how you store the patch relationships.


On your graphic you would work out that the player is in F. For ease of explanation lets change it to a 2d map so A = {0,0} b={1,0} c={2,0}, d={0,1}, etc.

```
F would then be {2,1} which I will call xmap,ymap
in pseudo code you would then
    yworld=negative_start
    for y=ymap-1 to ymap+1
        xworld= negative_start
        for x=xmap-1, to xmap +1
            calculate z world based on xworld,yworld
            draw patch x,y at xworld,yworld,zworld
            xworld+=step_size
        endloop
    yworld+=step_size
endloop
```

In the patch draw you convert x,y to a patch number using any technique you like. For a simple 2D map all you would have to do is check to see if either coordinate is off the map and wrap it around. This is as trivial as  
if (x>=mapsize) x-=mapsize;  
For more complicated structures you would use more complicated relationships. It would even be feasible to define an equation that maps x,y to patchnumber. I hope that's what you were looking for.


1/24/2008 5:56 PM

PumpkinGames



1/29/2008 3:43 PM

Darkside



Re: Rendering top-down maze game with edge wrapping (game on a torus)

Reply Quote


e just written a wrap-around 2d scrolling shooter and my approach was to never move the player! They were always located at 0,0 (with the camera directly above them) but whenever player input was detected I'd apply a force in the opposite direction to all the bad guys and scenery.

Re: Rendering top-down maze game with edge wrapping (game on a torus)

Reply Quote

If your quick (and I mean really quick) you can check out Kurt Jaeger's Tile Game Engine series over on [XNAResources](#). Kurt's been a great community help with his site and it's a damn shame, but he's shutting down the site soon. Check out the tutorial Series [here](#). Also check out his other tutorials on Colour Key Mapping and Photoshop basics.   
\*\*Note, the tutorials are still in Beta 2 code but it should be no problem to quickly upgrade them to Ver 2

Page 1 of 1 (6 posts) [Previous Discussion](#) [Next Discussion](#)

 [RSS Available](#)

Need Help?

[Terms of Use](#)

[Privacy & Cookies](#)

[Code of Conduct](#)

© 2015 Microsoft Corporation

All rights reserved

http://xboxforums.create.msdn.com/forums/p/7881/42342.aspx

Page 2 of 2