MAXIS

Potential Character Rendering Strategies

MDL, January 25, 1996

Here's a list of potential character rendering strategies that are proposed for the CTG Character Rendering project. I envisioned the following plan of attack for this, the second phase of this project (please let me know if I left anything out).

- We'll get together and expand, prioritize (or throw out) the list of strategies
- I'll work on demoing, mocking up, estimating development time, or profiling the top rendering strategies so that we can determine whether a technique is feasible.
- We'll get together to review the results, weighing rendering quality, art work required, RAM requirements, and CPU requirements.
- The group will decide which approach to use in Project Y.
- I will continue to work until artwork and integration requirements are done.
- I'll finish the development of tools for artwork integrated, if needed.
- I'll complete the final schedule, and move on to development phase.

Here's the list:

1. Full polygon rendering: Visualized, this is similar to the "Alone in the Dark ", "Fade to Black", etc. games, where animated characters are rendered polygons and consist of anywhere from 150 polys and on up. For this to work there would have to be some variation in the descriptions of the figures (beyond color), so we wouldn't have a city full of clones. This technique is a more traditional approach, and would let the Project Y renderer do all of the polygon work.

Demo Path: During Project Y initialization phase, load in all body parts as separate objects. From Will's engine convert theta, phi directions into rotations applied to each separate body part during each frame. Let Project Y perform all polygon rendering. Switch between rendering techniques at distances, by manipulating object flags for the renderer. The rendering callback will need to be modified so one call is made per person.

Integration Path: Unknown, however, grouping of objects, and real time manipulation of the character rendering engine for different distances may be required—as well as optimization of animation coordinate operations. If decals are to be applied to the face (or other parts of the body), then some additional modification of the rendering engine will be required.

Predicted impact on game: This technique is potentially the safest path to travel. It will result in better looking characters—additionally, the number of polygons (rendering quality) can be easily reduced or increased to test frame rates. The real unknown is the tradeoff between game performance and quality of rendering.

Art Requirements: Some 3D models will be required before I can investigate this technique. This will require the art department to produce some variations on 3D human models, that will depend on the number of polygons, as well as aesthetics of the model. For implementation this will not be a big art development hit, requiring perhaps several variations on the 3D figure.

Risks: Joints of segments, performance, transitions of rendering techniques, quality of rendering.

Scaleability: This technique would work for near to middle ranges—it is unneeded beyond that. It might be possible to switch to a simpler polygon model in the middle range.

2. Doom-like all sprite rendering: Actually there really is no 3D rendering here, the rendering is done by the artist off line, and this solution will be fast (very fast). Because the artwork is done off-line, each potential position the character could assume, would have to exist in RAM, or be fast enough to yank from the disk. As a matter of fact, for this to work, serious analysis of the character animation requirements would need to be done, in order to determine how many possible positions the character would actually have to be in.

Demo Path: Build transparent blitting routine, or use the renderers transparent blitting routine, to render in the character images. Fast scaling techniques will be required for smooth distance transitions. The technique to load these images into memory will need to be built (optimization of image loading will occur in the integration stage). This method will require decoupling, or limiting the character animation engine.

Integration Path: Work with Project Y on memory issues, imaging loading issues, and potentially porting of scaling code to assembly for speed.

Predicted impact on game: Will look good, but movements and views will be limited.

Art Requirements: Several character animations sequences will need to be rendered from off-line 3D models. Some artwork will need to be created up front, in the demo stage, for this technique to be evaluated.

Risks: Large memory requirements, limitation of character activities.

Scaleability: Highly scaleable, from near to far.

3. Partial line draw and sprite rendering: This technique use sprites for the torso and the head, while leaving the limbs in the current line drawn mode. Because the head and torso do not generally move with respect to the body, this would only require artwork for the 3D to 2D transformation (and not a range of motions). This technique would suffer from the limbs looking somewhat crude compared to the rest of the torso and the head.

Demo Path: Pretty much the same requirements as the previous technique—with the exception of no animation sprite matrix.

Integration Path: Pretty much the same requirements as the previous technique.

Predicted impact on game: Less of a memory hit than the previous method—but still may eat up RAM. This may or may not turn out to look good enough.

Art Requirements: Quite a bit for all viewing positions of torso and head.

Risks: Near distance rendering quality will not be great, RAM usage.

Scaleability: Same as the previous technique.

4. Characters using ellipsoids as primitives: Ellipsoids can be used to model body segments with the head represented as a sprite or 3D model. Only a few ellipsoids need to be used to model the segments of a body. This technique might be useful in creating a smooth, somewhat featureless body.

Demo Path: Artwork is needed to mock up off line some ellipsoid rendered figures. Build ellipsoid rendering routine in C.

Integration Path: Most likely move the ellipsoid into the assembly rendering code.

Predicted impact on game: This might provide an incremental improvement over the current rendering quality. If the ellipsoid renderer is fast then I suspect that we could really populate the city with this technique.

Art Requirements: Some time is needed up front to mock up the characters in order to get a handle on the potential quality. Additional work on the polygon/sprite head will be needed if this technique is demoed/integrated into Project Y.

Risks: Rendering ellipsoids is slow, quality of character body.

Scaleability: Good for near and middle distances.

5. Characters using spheres as primitives: There are some games on the market that use this technique (Relentless comes to mind). Basically this technique works by using spheres as rendering primitives for the character. Rendering and quality of rendering is fairly unknown. Another question is whether this technique will fit in the environment, in other words will the characters look like Sims?

Demo Path: Implement new primitive. Work with Ed to get this primitive dropped down into the renderer. Again some work and conversion of character animation positions to sphere movements will be required.

Integration Path: Unknown.

Predicted impact on game: May still suck up close—some games that use this technique should be checked.

Art Requirements: Not much.

Risks: Look of characters do not fit into environment, rendering speed.

Scaleability: Probably good for near, middle and far distances.

6. 3D face with decal, line drawn body: This would require a head shaped spheroid to have a 2D flat stretched face wrapped to the sphere. This method would be used in conjunction with line drawing and some rendering technique for the torso.

Demo Path: The technology to decal the face onto the 3D object needs to be built—unknown without more research into the current structure of the Project Y renderer.

Integration Path: Unknown.

Predicted impact on game: This would be a much needed upgrade to the current rendering technique. However, there might be too large a contrast between the line drawing output and the facial image in terms of aesthetics.

Art Requirements: Not great—some work with artist is developing tool to view face decal in 3D.

Risks: Quality of character.

Scaleability: All ranges, except in the far distance the 3D and decal are not needed.

7. Voxel character rendering: Or solid modeling of characters. The character would be represented as 3D particles. Each 3D particle would contain it's location and color. The collection of these particles are the 3D character. In addition to representing the character this method makes manipulation and deformation of the character's shape possible. The major unknown for this technique is rendering speed.

Demo Path: Unknown—survey of voxels is required, as well as voxel rendering strategies.

Integration Path: Unknown.

Predicted impact on game: Could be cool, plus it might be possible to do some things not possible with other techniques.

Art Requirements: Some development of 3D models to convert to voxel representation.

Risks: Too many unknowns, may be too slow.

Scaleability: Probably very scaleable.