Dear Angelica: Breathing Life into VR Illustrations

Oculus VR, LLC
Menlo Park, CA 94025

ABSTRACT

In many ways, Oculus Story Studio's VR experience, "Dear Angelica" as shown in Figure 1, is a unique project. We wanted to immerse the viewers inside a series of hand-drawn illustrations and tell a story by artfully transitioning between the drawings, and breathe life into these drawings by adding various animations and visual effects. In trying to create this ground-breaking experience, we had to create a brand-new pipeline: from inventing the tools to create these 3D illustrations, to processing and animating the drawings, to rendering them in virtual reality.

CCS CONCEPTS

• **Software and its engineering → Object oriented development**;

KEYWORDS

Virtual reality, real-time rendering, 3D illustration

ACM Reference format:
Oculus VR, LLC. 2017. Dear Angelica: Breathing Life into VR Illustrations. In *Proceedings of SIGGRAPH '17 Talks, Los Angeles, CA, USA, July 30 - August 03, 2017,* 2 pages.
https://doi.org/10.1145/3084363.3085073

1 PAINTING IN VR

Figure 1: A screen shot from Dear Angelica.

SIGGRAPH '17 Talks, July 30 - August 03, 2017, Los Angeles, CA, USA

ACM ISBN 978-1-4503-5008-2/17/07.
https://doi.org/10.1145/3084363.3085073

In the pre-production phase, we experimented on many techniques to bring hand-drawn illustrations into VR. However, methods such as projecting paintings as textures on proxy models or manually modeling out the drawings in traditional 3D programs quickly proved to be too time consuming with mediocre results. Eventually, we created a program called Quill that let our artist, Wesley Allsbrook, paint directly in VR with Oculus Touch controllers, as depicted in Figure 2. The artist can finally use 3-dimensional space as a canvas to create truly immersive drawings.

Figure 2: Illustrator Wesley Allsbrook demoing Quill.

2 PROCESSING AND ANIMATION

The drawings are then brought into Houdini as 3D lines, along with all the attributes defined by the artist such as color, opacity, width, line direction and drawing orders. We then use the information to process every single stroke. For example, when we need an illustration to "draw in" as if there is an invisible hand painting, we calculate the timing and direction attributes on the strokes, attach them onto the vertices, and animate the vertex opacity by comparing these attributes against the global time, as shown in Figure 3.

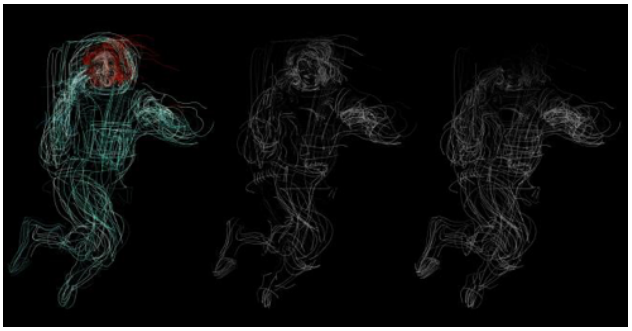

Figure 3: (Left-Right): original line data from Quill, original draw- in attributes visualized as colors, modified draw-in attributes visualized as colors.

Furthermore, when we create more advanced effects such as lines that fly around the viewer or character animations, we convert all the animation data into positional textures, and import them into Unreal shaders and move the lines via vertex position offsets. We also optimize the drawing data so they can be rendered in real time: we resample the stroke intelligently to reduce the vertices on the strokes without losing the curvatures, and use various other methods such as LOD reduction, back face culling, and procedurally remove nested lines.

3 RENDERING AND SHADING

After the illustrations are processed, they are then exported into Unreal Engine as lines for real time rendering. The engineers developed a tessellation shader that takes curve data points as inputs and produce polygons at render time. Since no lighting is required, we developed a custom forward rendering engine for Unreal that can handle lots of transparencies with 8X MSAA super sampling to render the illustrations with the highest possible quality.