Reel Breakdown

Cloud Voyage
I was responsible for creating the clouds, lighting and rendering.

In order to achieve this animation, I first found some geometry on-line to use. After importing it, I then then scattered points along the geometry of each piece and then spawned a sphere of random size in these locations. This was done in-order to increase the randomness of the silhouette of the geometry. I then merged this geometry and converted it to a VDB fog density so I could manipulate the it using a Volume VOP & VEX. Finally I added some cloud noise to the volume and lit it with a volume light. This was done for every piece of geometry. To animate the geometry I just translated the geometry before the VDB conversion.

Disposable
I was responsible for rigid body and particulate simulations, surfacing various environment pieces, lighting, compositing half of the shots, modeling various environment objects, pipeline, layout, and small animation.

To do the rigid body simulations, I took the trash cube, reduced the polycount of it to around 25k. I then RBD Packed the cube and spawned it from randomly seeded points on a grid. I then ran a basic RBD simulation using the Packed Geo and a RBD solver. To create the bits and pieces, I took the reduced cube and cut out corners, RBD packed the geo, randomly spawned the packed geo, and then ran a RBD simulation using the original cube simulation as collision geometry.

To do the particulate simulations, I created a volume that filled the camera frustum to the wall. I then multiplied the density of this uniform volume with coarse perlin noise and fine curl noise. I then used that volume to spawn particles for the simulation. I then took those particles and moved them around by adding random drag, torque, and acceleration using POPs and Attribute Wrangles. After that I instanced thin grid geometry that was randomly twisted.

Bug Spray
I was responsible for water simulations, surfacing various environmental models, compositing half of the shots, modeling various environment objects, pipeline, and layout.

To do the rising water for the title, I created the font geometry and made it collision geometry. I then took a plane and animated it to pass through the font. While the plane passed through the font I used an intersection test to keep only the geometry that was part of the plane and the font. I then used this intersection geometry as an emitter for a FLIP fluid and ran the simulation.

To do the main water simulation I created the collision geometry (pipe, hero character, room, objects on the floor) and created an emitter for the fluid inside the pipe. I then ran a FLIP fluid simulation using the emitter and collision geometry. To achieve a sense of surface tension with the fluid I had to modify the FLIP particles using GAS DOPs to make sure the particles wouldn't separate to fast

Underwater Dust
To do the underwater dust I created a pyro simulation and modified the buoyancy, temperature, and lift. I added some low frequency turbulent noise for some rough movement, and some high frequency to get some wisps. I modified the arnold volume shader to make the light scatter through the volume and turn a burnt brown look.

Particle Portal
To do the particle portal I animated a torus and scattered emission points for the particles along the geometry. I then used the velocity of the animation to drive the initial POP simulation and then used animated values in POPs to add random drag, torque, and decay.

Generic Explosion
To achieve this generic explosion I first started out with a sphere and added animated alligator noise on it. This was done in-order to achieve change during the pyro simulation so that it wasn't a static emitter. From there I used the shelf tool to get a quick start on the pyro simulation. I then modified the simulation settings (such as burn rate, turbulence, buoyancy, etc.) to what I want the explosion to look like. After that was done I used a volume light to help light up the simulation. For the shader I decided to allow an artistic influence in the sense that an artist can modify the heat color of the simulation by using a color ramp. This was chosen over physical option solely because I could control the look of the explosion far better.

Garage Explosion
I was responsible for the explosion, fracturing, smoke trails, lighting, and rendering.

I imported the geometry from an alembic cache. I then selected which parts of the geometry would be affected by the explosion and isolated them. After that I painted on the geometry for where I would like the geometry to fracture the most. After this was done I scattered points based on the density of the painted geometry. I then did a voronoi fractured based around the scattered points. I then glued the fractures together so that they could be driven by the velocity of the pyro simulation.

In order to do the pyro simulation I imported the simulation file from my generic explosion asset. I then used the velocity data in the simulation to drive and fracture the building breaking apart. The smoke trails were done by doing a cluster pyro simulation on the fractured geometry and spawning points based off of their velocities. Due to the pyro

simulation being the same as the generic explosion it has the same artistic custom-ability as the generic explosion.

Procedural Paintings

To do the procedural paintings I procured some images of paintings that I liked from the internet. I then made a grid to match the size of the image exactly by making one division equivalent to one pixel. Next, I used a custom image kernel written in VEX to detect the edges of the inputted image. I then projected the original image onto the grid and changed the color value of the grid vertices. Next, I used the edges found by the kernel to add and delete vertices of the grid based on color intensity. I then used a polypipe and attribute wrangle to find the nearest point and make an edge between the two points. I then made the geometry emit the color of the point.