Atomic Bomb in Unity

Christopher Dungo  
 University of California, Santa Cruz  
 Santa Cruz CA U.S.A.  
 cdungo@ucsc.edu

Alfred Lam  
 University of California, Santa Cruz  
 Santa Cruz CA U.S.A.  
aylam@ucsc.edu

ABSTRACT

The objective of this project is to recreate an atomic bomb explosion similar to that of the explosion in Call of Duty: Modern Warfare Remastered. We were able to achieve this using particle systems, light, and bloom shaders.

1 Smoke

The smoke of an atomic explosion can be divided into two particle systems: one for the “stem” and one for the “ring.” Both systems are transformed so that they’re in the same position.

Smoke contains a countably infinite number of small particles, so spawning individual smoke particles would require an overwhelming amount of memory. To fix this problem, particles will spawn using an 8x8 animated texture sheet that makes it look like rapidly moving clouds of smoke. The Texture Sheet Animation Module is checked for this case.

To support this, the size of each particle is rather large (greater than 30) and the emission rate is at least 100 so that it looks like clouds of smoke are clumped together well and the scene loads faster. The lifetime of each particle is around 10 seconds so that one could view the rushing speed of the smoke at its fullest.

To make each cloud seem different, they exhibit a random rotation between 0 to 360 degrees.

The material used for the smoke uses Unity’s legacy animated alpha blend shader (Legacy Shaders/Particles/Anim Alpha Blended) so that when the Color Over Lifetime module is set, the smoke fades in as the explosion happens and fades out when the clouds near the end of their lifetime.

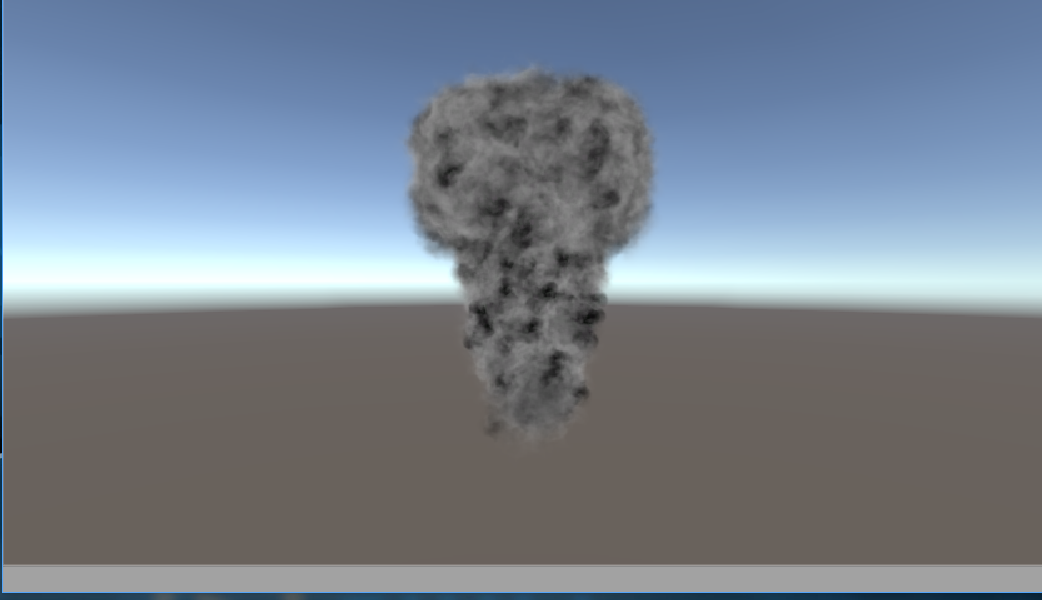


Figure 1.1: The Smoke Stem.

*1.1 The Stem.* The stem is generated with an upward cone-shaped particle system. Gravity and drag are used to create the “puff” at the top of the stem, where the particles start to slow down. These values change according to particle speed. In this case, the speed of each particle ranges from 100-200, the drag is 0.3 and gravity is ~0.2. It behaves almost like a fountain of smoke.

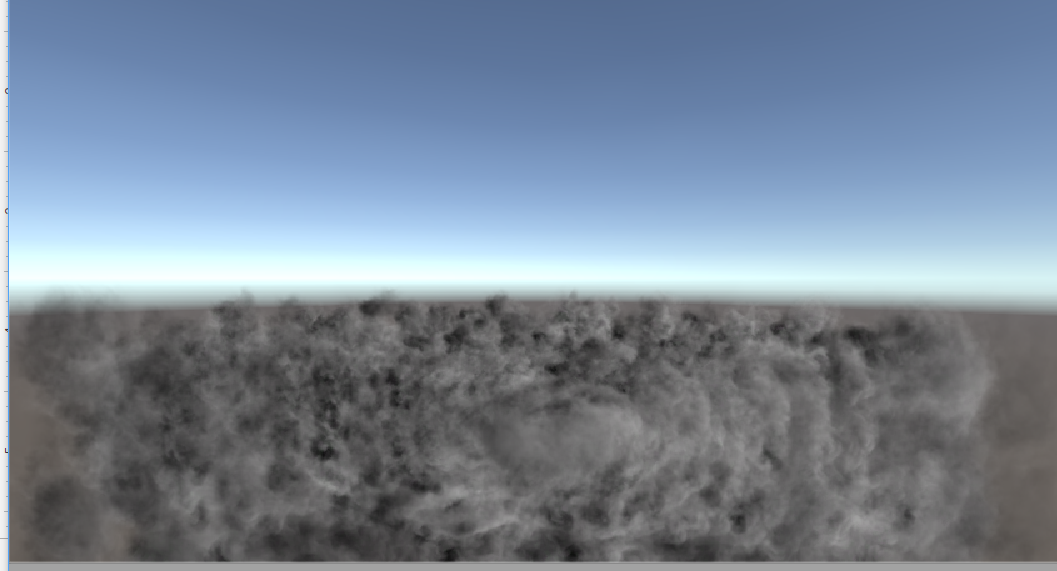


Figure 1.2: The Smoke Ring.

*1.2 The Ring.* The ring is generated with a circular particle system that spawns particles at the edge of the circle (i.e. radius thickness = 0). Some of the particles move slightly upward, due to their velocity over their lifetime being random. The speed is actually ~10 since the camera is close enough that the clouds look like they are moving rapidly.

2 Light

As part of the scene, we wanted to recreate the flash of blinding light that is released with the massive burst of energy from an atomic bomb, as seen in the clip from Modern Warfare I. Our initial approach involved using the Lens Flare component built in to Unity - however, we quickly realize that this greatly limited the range of effect we could produce, specifically restricting the size and brightness of the light, since we needed a special “Flare” asset. Because of this, we decided to use a bloom shader to recreate the blinding light effect, which gave us much more control over the various aspects of the effect. The way that this was implemented involved 3 main parts - rendering to a temporary texture, downsampling and upsampling to “blur” the image, and then combining the blurred results with the original image. The main choices we had to make here were how to downsample/upsample, and to what extent. We found that 10 iterations was the most effective for our blinding light effect, as this would create enough blur in the image to cover the majority of the screen, with high enough intensity of light.

*2.1 Implementation* The downsampling is done with a box filter that samples the surrounding pixels around each pixel, and takes the average of all pixels within the box. This process is repeated for a certain number of iterations (configurable in the Unity Editor), and then we upsample to recreate the blurred image. Upsampling is done in a similar manner, but with a smaller kernel for the box filter, which recreates a more focused and accurate image than with the larger kenel used in downsampling. Finally, this blurred image is combined with the original image, to create a bloom effect for the parts of the image that are above a certain threshold brightness (also configurable in the Editor). By increasing the intensity of the light, we can then increase the bloom effect to cover the entire screen, and slowly fade it away as well.



Figure 2.1: The Bloom Effect.

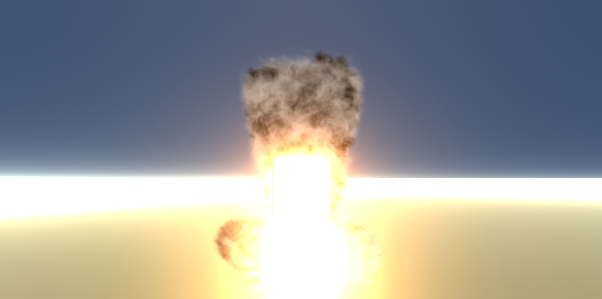


Figure 2.2: The Initial Explosion.

In order to create the effect we wanted, the light had to spawn as the bomb hit the ground. We implemented this by instantiating a new object as the object collided with the ground, and turning the light intensity and bloom factor up to maximum. After this, the intensity would smoothly drop down over time and reveal the explosion behind it, until finally the light object would be destroyed, and the light would fade away, exactly the effect we wanted.

3 Fire

While atomic bombs are well-known for their massive mushroom clouds of smoke, we also wanted to recreate the fiery inferno that lies within the center of the cloud, which adds a hellish tint of red and orange to the already frightening explosion. Our plan was to add a secondary particle system to the smoke, which would be similar in shape and size, but would be slightly smaller and kept within the smoke. That being said, when we tried this, we found issues with flickering between the fire and the smoke particles, which were fighting to get in front of the other. We couldn’t find a way to solve this in time, and so we decided instead to use a simpler particle system, keeping a ball of fire at the center of the explosion. With more time, we believe that it would’ve been better implemented as a combined texture of fire *and* smoke particles, using noise functions to create random variation in the effect.

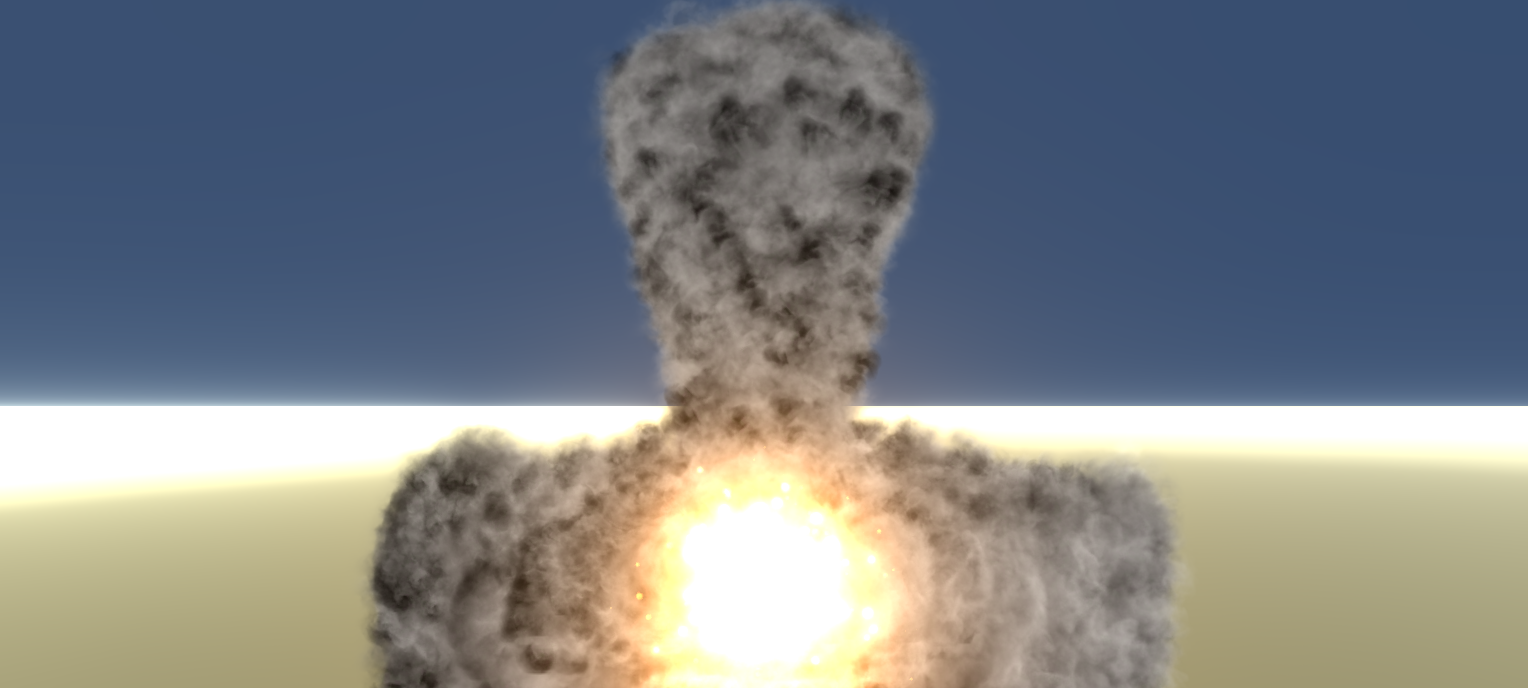


Figure 2.3 A fireball remains in the center of the particle system.

ACKNOWLEDGMENTS

YouTube users Brackeys and Mirza helped inspire the smoke design and texture sheet animations respectively. Brackeys’s video provided the texture sheet used in this project in his video description.

REFERENCES

[1] Brackeys Smoke Tutorial: <https://www.youtube.com/watch?v=R6D1b7zZHHA&t=323s>

[2] Mirza Texture Sheet Animation Tutorial: <https://www.youtube.com/watch?v=8lhMZH6SmIY>

[3] Call of Duty Nuke Scene: <https://www.youtube.com/watch?v=4r-YtrYGAIQ&t=166s>

Conference Name:ACM Woodstock conference

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00