Games that feature destructible environments and how they do it.

Red Faction

Red Faction (2001) and Red faction 2 (2002) video games were created using the Geo-Mod (**Geo**metry **Mod**ification Technology) game engine developed by Volition. The engine featured the alteration of level geometry such as walls and ground by making holes in surfaces using certain weaponries. This was achieved by creating a new “empty space” object at the point of collision. This new object would then be subtracted from the surface it just hit in real time – occluded by visual effects such as dust and explosion animations to hide the sudden subtraction of the meshes. As a result, modifying the geometry by adding a hole to it. This approach was unique at the time as it was not replacing an object to its damaged state rather than altering its geometry altogether. This feature arose some game design issues as the game could not protect the player from himself because they could tunnel through everything in the level thus making some areas “unwinnable” if enough damage was done to the surroundings. Nevertheless, this was an important attempt to simulate a fully destructible environment in 3D regardless of the real-time constraints.

Red Faction: Guerrilla

Geo - mod 2 was later developed to create the next game in the Red Faction series, Red Faction: Guerrilla. This time the engine was focusing more on the destruction of buildings with a stress-based collapse model rather than the modification of level border geometry such as the walls and ground. Besides large or important structures, most buildings in Guerrilla are destructible and react to damage in real-time, eventually collapsing if enough structural support is lost.

<https://redfaction.fandom.com/wiki/Geo-Mod>

Crysis

I could not find information on how the CryEngine implemented destructible environments in the Crysis series for the reason that I assume is confidential, and the developer doesn’t want details of the game developed to be released to the public. However, I found the CryEngine documentation on breakable objects and in it there are descriptions of different ways of setting up breakable assets that the player can destroy.

Methods used are as follows:

* **Jointed breakable objects** are pre-broken objects that use virtual joints to hold the pieces together. The joints break when a certain amount of force such as a bullet is applied to it and the pieces start to fall by gravity. This method has a big impact on drawcalls, memory consumption and physics calculations. It is also usually followed by a particle effect based on the surface type of the object. A clever way to work with it, is to use as few pieces as possible by spawning more pieces when combined with the particle effect.
* **Destroyable objects/Assembled pieces** are objects that comprise of two models: the original object and the pre-created pieces that appear when the original object is destroyed. This is similar to the object replacement method.
* **Jointed destructible objects** can be set up to break into smaller pieces when detached from the joints. For example, a wooden signpost that will break into several planks when shot or a table that not only disassemble when shot but also brake into small wooden pieces.
* **Breakable glass** is glass objects with different levels of destructibility. First, they can break realistically by breaking into pieces that fall off and disappear. Second, they shatter into different patterns such as spiderweb or Roosevelt. Third and last, they don’t break or shatter, they just spawn a bullet shot.
* **Deformable objects** are like cloth objects skinned to a skeleton. Unlike cloth, they are not constantly moving, they instead “freeze” in their deformable position, so they can be used to create metal or other deformable materials. Deformable objects consist of two nodes: Render mesh, physics mesh and skeleton. The skeleton is the actual deforming object, the render mesh and physics mesh are following its deformations in a similar fashion similarly to a skinned character following bones. The positions of the skeleton mesh vertices control the deformation, vertices outside the render mesh will not be deformed, vertices inside the render mesh will be deformed the render mesh will be update based on those vertices.
* **Boolean operations** surpass the need of pre-made breakable objects and allow dynamic physical mesh damage to take place during runtime. This is achieved by subtracting a pre-modelled 3D shape from the original object on the exact point of impact. Because this is happening in run-time, the system dynamically re-triangulates the original model – by adding polygons as necessary - in order to maintain the original solid shape with the additional holes or missing pieces. The interior faces take the material and textures from the Boolean shape.
* **Pre-Baked Physics** simulation where the broken pieces can be detached from the animation and become physicalized. This would probably be the cheapest way to simulate large destruction that are the result of a cut scene or a predictable source.
* **Fractionalizing objects** in modelling software to be used for breakability
* **CryEngine** was developed by German developer Crytek in 2002 and was used to develop many AAA games including Crysis, FarCry, Sniper and Kindom Come: Deliverance. The engine kept updating to different versions up to current 5.4. Rummors say its licence was sold to Amazon for 50- 70 million dollars. Amazon changed the name to Amazon Lumberyard.

<https://docs.cryengine.com/display/SDKDOC2/Breakable+Objects>

<https://en.wikipedia.org/wiki/CryEngine>

Mercenaries 2

Minecraft

Worms

Fracture

Hulk: Ultimate Destruction

Just cause 2

RainBow six siege

7 days to die

Battlefied series