32. VIDEO GAME OPTIMISATION

Novikov P.A.

Belarusian State University of Informatics and Radioelectronics
Minsk, Republic of Belarus

Liakh Y.V. – Senior Lecturer

400

This paper studies and analyses various aspects of video game optimisation, such as common graphics settings and optimisation techniques, and provides brief explanations to all of them. There are also insights on importance of video game industry and optimisations as a whole.

Video game industry started more than 50 years ago, and it led to rapid development of both software and hardware due to high demand. Often game developers face many issues regarding technology limitations, so it is important to adapt and find ways around them, push available tools to the limits.

FPS (frames per second) is one of the most common metrics for evaluating performance of video games, because it affects the end user's game performance as well. FPS describes how many times per second the screen refreshes to render the image.

A low number of frames means that it is difficult for someone to play the game, while a higher number means that the game runs smoothly. Most video games today are developed to hit a frame rate of 60 FPS, but the rate between 30 FPS and 60 FPS is considered acceptable [1].

Many modern games include advanced graphics settings, that can affect performance depending on hardware the game is being played on. They can be adjusted automatically or manually to find balance between aesthetics and performance.

Most common graphics settings include:

1. Rendering resolution that describes the relative screen size the game is being projected on. It can be written as two numbers: length and width. For example, 1920x1080. Bigger resolution means higher image quality, but resources required to maintain it are drastic, considering that increasing resolution by 2 times also requires 4 times more resources.

2. Anti-aliasing is a method of smoothing sharp pixel edges by adding smoothing pixels. There are multiple methods for anti-aliasing in modern games (listed in ascending order for their expected performance impact): pure screen-space methods like FXAA or SMAA; temporal accumulation; multisampling (MSAA); supersampling (SSAA); combinations of the above (TXAA) [2].

3. Lighting and shadowing are methods of simulating light and shadows in games. Generally, their impact on performance can be vital depending on the amount of lighting in a scene. There are two main options for projecting light and shadows: real-time and baking. Real-time makes calculations during runtime, which affects the performance. Baking light/shadows is a process of calculating them beforehand, "baking" them into textures.

4. Ambient occlusion is a shadowing technique used to make 3D objects look more realistic by simulating the soft shadows that should naturally occur when indirect or ambient lighting is cast out onto the scene. This is an expensive effect that also has more advanced variants: voxel ambient occlusion, or occlusion based on distance fields.

5. The draw distance (or view distance) and field of view determine what the player will see on the screen. These settings are relevant to first and third-person shooters. The draw distance setting determines how far one sees into the distance, while the field of view refers to the peripheral view of a character. Their impact is minimal.

6. Anisotropic filtering is a method of using lower quality models and textures for objects depending on how far away they are from the view.

Optimisation techniques are not uncommon in the video game industry, as many projects tend to push software and hardware to the current limits. In the last decade, mobile gaming industry began growing rapidly, rivaling home consoles and PC branches. The problem with mobile devices, however, is that they do not all have the same capabilities. It is the result of the constant rising of technology and the emergence of more powerful systems. Furthermore, for older models, optimisation techniques can prove to be demanding or meaningless because the device does not have enough processing power to meet the requirements of a modern video game. Therefore, the developer should choose from the beginning the range of devices which they want the game to run. This is also the case for studies on the effects of various optimisation techniques, which always define the devices that were utilised in the experiments [3].

Most common optimisation techniques are the following:

1. Code optimisation. They can be planned from the beginning, by defining the project architecture, like MVC or ECS, and code writing rules, like SOLID, but they are usually done iteratively, by updating pieces of code when needed. Even small changes can impact the performance, since most of the code runs every frame, which is about 60 times per second.

2. Using low polygon 3D models. 3D models are made of triangles or polygons. Greater polygon count makes the model look more detailed, but it also requires greater processing power from the GPU or the CPU. For weaker devices, especially mobile ones, it is recommended to reduce the number of polygons. For example, by merging several polygons/vertices into one. It is also important to note that a GPU can only render triangles, so all polygons of a model should be converted into triangles beforehand.

3. Batching is a method of reducing the amount of draw calls sent to the GPU, by giving several models the same material, so that they can be combined into one batch.

4. Occlusion culling increases rendering performance simply by not rendering geometry that is outside the view frustum or hidden by objects located closer to the camera. Two common types of occlusion culling are occlusion query and early-z rejection [4].

5. Object pooling is a method of creating a set of objects to be used in the game. They are created at the beginning of the game. When the game needs an object to be created, it takes one of the available ones in the pool, and when it needs to be removed, the object is put back in the pool, ready to be reused. It is a very important technique for games with many objects being quickly added or removed in a scene.

6. Profilers are tools that track a program's usage of the CPU, GPU, memory allocation, frame rate, and other key metrics [5]. They are useful for developers to find performance issues and their causes. Most game engines like Unity have built-in solutions (Figure 1).

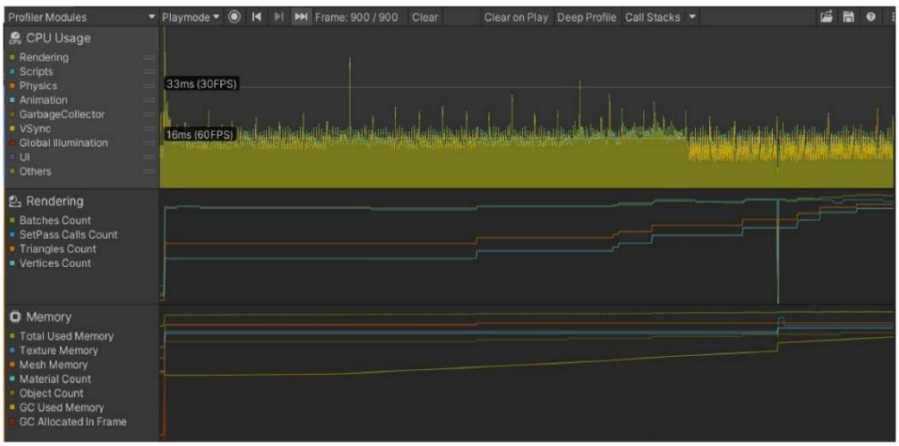

Figure 1 – Unity Profiler

Game engines like Unity or Unreal Engine also abstract a lot of routine work for developers, like memory managing via built-in garbage collector, graphics rendering, options for asset optimisations like changing formats or different options, as well as provide graphic editors and simple solutions for quick game prototyping.

In conclusion it should be mentioned that the video game industry is an environment where technology is constantly being pushed to the limit, which leads to its rapid development. The video game market is growing and evolving all the time. New trends are emerging, like mobile gaming, virtual and augmented realities (VR and AR). But in order to achieve such feats, developers must not only use available tools, but optimise them in various ways for specific needs.

References:

1. Understanding and optimizing video game frame rates [Electronic resource]. – Mode of access: https://www.lifewire.com/optimizing-video-game-frame-rates-811784. – Date of access: 16.03.2023

2. What 'optimization' really means in games [Electronic resource]. – Mode of access: https://www.pcgamer.com/what-optimization-really-means-in-games/2. – Date of access: 14.03.2023

3. Improving Mobile Game Performance with Basic Optimization Techniques in Unity [Electronic resource]. – Mode of access: https://www.mdpi.com/2673-3951/3/2/14. – Date of access: 19.03.2023

4. Efficient Occlusion Culling [Electronic resource]. – Mode of access: https://developer.nvidia.com/gpugems/gpugems/part-v-performance-and-practicalities/chapter-29-efficient-occlusion-culling#:~:text=Occlusion%20culling%20increases%20rendering%20perfor mance,query%20and%20early%2Dz%20rejection. – Date of access: 21.03.2023

5. Basic Game Optimization Techniques [Electronic resource]. – Mode of access: https://jarlowrey.com/blog/game-optimizations. – Date of access: 18.03.2023