

Performance Optimization

Optimizing Graphics Performance

Optimizing Graphics Performance

- Why is graphics performance low?

Optimizing Graphics Performance

- Why is graphics performance low?
- Fillrate? (screen pixels * overdraw)

Optimizing Graphics Performance

- Why is graphics performance low?
- Fillrate? (screen pixels * overdraw)
- Shader computations?

Optimizing Graphics Performance

- Why is graphics performance low?
- Fillrate? (screen pixels * overdraw)
- Shader computations?
- Vertex count? (Intel GMA950)

Optimizing Graphics Performance

- Why is graphics performance low?
- Fillrate? (screen pixels * overdraw)
- Shader computations?
- Vertex count? (Intel GMA950)
- Draw calls?

Draw calls

Draw calls

- Lots of objects tax the CPU. On the other hand lots of Triangles tax the GPU.

Draw calls

- Lots of objects tax the CPU. On the other hand lots of Triangles tax the GPU.
- You still need the CPU for game code

Draw calls

- Lots of objects tax the CPU. On the other hand lots of Triangles tax the GPU.
- You still need the CPU for game code
- On a modern machine you can use around 500 draw calls per frame

Draw calls

- Lots of objects tax the CPU. On the other hand lots of Triangles tax the GPU.
- You still need the CPU for game code
- On a modern machine you can use around 500 draw calls per frame
- Each pixel light renders a mesh one more time. Don't use too many pixel lights!

Combining

Combining

- Lots of objects in the scene - bad.

Combining

- Lots of objects in the scene - bad.
- Spatially close objects combined - good.

Combining

- Lots of objects in the scene - bad.
- Spatially close objects combined - good.
- “Overcombined” objects - bad again.

No Combining

No Combining

- Lots of objects

No Combining

- Lots of objects
- High draw count

No Combining

- Lots of objects
- High draw count
- Bad on CPU and GPU

Good Combining

Good Combining

- Combined objects are close to each other and use the same material.

Good Combining

- Combined objects are close to each other and use the same material.
- E.g. combine inside “rooms” of the house.

Good Combining

- Combined objects are close to each other and use the same material.
- E.g. combine inside “rooms” of the house.
- Often requires using texture sheets!

Good Combining

- Combined objects are close to each other and use the same material.
- E.g. combine inside “rooms” of the house.
- Often requires using texture sheets!
- Rule of thumb: object size and point/spot light size about the same.

Bad Combining

Bad Combining

- Never combine objects that are far apart!

Bad Combining

- Never combine objects that are far apart!
- If light shines only on part of object, whole object needs to be drawn.

Bad Combining

- Never combine objects that are far apart!
- If light shines only on part of object, whole object needs to be drawn.
- If camera sees only a tiny corner of object, whole object needs to be drawn.

Combining Demo

Combining Demo

- How much do I actually gain?

Combining Demo

- How much do I actually gain?
- Can your game afford not combining? Can you afford a factor 30x in the worst case?

Combining Demo

- How much do I actually gain?
- Can your game afford not combining? Can you afford a factor 30x in the worst case?
- Geometry matches lighting

Lights are for?

Lights are for?

- Dynamic lights!

Lights are for?

- Dynamic lights!
- Lightmapping to the rescue

Lights are for?

- Dynamic lights!
- Lightmapping to the rescue
- Combine the two. Lightmap your scene to get rid of all static lights. It's faster and looks better!

Lights are for?

- Dynamic lights!
- Lightmapping to the rescue
- Combine the two. Lightmap your scene to get rid of all static lights. It's faster and looks better!
- When to use lights? Rockets, torches and dynamic shadows

Lights are for?

- Dynamic lights!
- Lightmapping to the rescue
- Combine the two. Lightmap your scene to get rid of all static lights. It's faster and looks better!
- When to use lights? Rockets, torches and dynamic shadows
- Culling masks!

How to lightmap?

How to lightmap?

- Lightmapping is done in Maya or other rendering Tools.

How to lightmap?

- Lightmapping is done in Maya or other rendering Tools.
- Easily imported into Unity, simply switch to any lightmapped shader

How to lightmap?

- Lightmapping is done in Maya or other rendering Tools.
- Easily imported into Unity, simply switch to any lightmapped shader
- Ask Ethan, our Maya expert! He'll happily get you up to speed with Lightmapping in Maya. Use the sign up sheet!

Avert Fate Demo

Optimizing Script code

Be aware of expensive
functions

Be aware of expensive functions

- A lot of Utility functions may be slow.

Be aware of expensive functions

- A lot of Utility functions may be slow.
- So while useful and sometimes necessary. avoid it.

Raycasting

Raycasting

- Raycasting against Meshes is slow. Don't do a lot of raycasts each frame

Raycasting

- Raycasting against Meshes is slow. Don't do a lot of raycasts each frame
- Use culling mask to early out raycasting

Raycasting

- Raycasting against Meshes is slow. Don't do a lot of raycasts each frame
- Use culling mask to early out raycasting
- Raycast only every third frame?

Finding objects

Finding objects

- FindObjectsOfType, FindGameObjectWithTag, GameObject.Find

Finding objects

- FindObjectsOfType, FindGameObjectWithTag, GameObject.Find
- Avoid doing it too much. It's convenient but those calls can quickly add up

Finding objects

- FindObjectsOfType, FindGameObjectWithTag, GameObject.Find
- Avoid doing it too much. It's convenient but those calls can quickly add up
- Use static variable

Finding objects

- FindObjectsOfType, FindGameObjectWithTag, GameObject.Find
- Avoid doing it too much. It's convenient but those calls can quickly add up
- Use static variable
- Store all active components of a type using a static ArrayList and OnEnable / OnDisable

Only Update nearby
objects

Only Update nearby objects

- Don't run scripts at all, when far away

Only Update nearby objects

- Don't run scripts at all, when far away
- `function Update () { if (far away) return; }`
does NOT count. Use `enabled = false;`

Only Update nearby objects

- Don't run scripts at all, when far away
- `function Update () { if (far away) return; }` does NOT count. Use `enabled = false;`
- Use triggers for finding out when player is close by or use coroutine that updates every 5 seconds

Math

Math

- Add, subtract, multiply: a couple cycles

Math

- Add, subtract, multiply: a couple cycles
- Divide: 30-40 cycles

Math

- Add, subtract, multiply: a couple cycles
- Divide: 30-40 cycles
- Square root, sin, cos: 60-100 cycles

Sqrt and Normalize

Sqrt and Normalize

- In inner loops avoid Sqrt and Normalize. They are fairly expensive. (Normalize internally uses Sqrt)

Sqrt and Normalize

- In inner loops avoid Sqrt and Normalize. They are fairly expensive. (Normalize internally uses Sqrt)
- $\text{Normalize} = \text{vec} / \text{Sqrt}(\text{vec.x}^2 + \text{vec.y}^2 + \text{vec.z}^2)$; Use `sqrMagnitude` instead.

Sqrt and Normalize

- In inner loops avoid Sqrt and Normalize. They are fairly expensive. (Normalize internally uses Sqrt)
- $\text{Normalize} = \text{vec} / \text{Sqrt}(\text{vec.x}^2 + \text{vec.y}^2 + \text{vec.z}^2)$; Use `sqrMagnitude` instead.
- `if (distance > dif.magnitude) Bad!`

Sqrt and Normalize

- In inner loops avoid Sqrt and Normalize. They are fairly expensive. (Normalize internally uses Sqrt)
- $\text{Normalize} = \text{vec} / \text{Sqrt}(\text{vec.x}^2 + \text{vec.y}^2 + \text{vec.z}^2)$; Use `sqrMagnitude` instead.
- `if (distance > dif.magnitude) Bad!`
- `if (distance * distance > dif.sqrMagnitude)`

Cache

Cache

- Cache hit: a couple cycles

Cache

- Cache hit: a couple cycles
- Cache miss: 40-hundreds of cycles

Cache

- Cache hit: a couple cycles
- Cache miss: 40-hundreds of cycles
- What does a simple transform.position do internally?

Cache

- Cache hit: a couple cycles
- Cache miss: 40-hundreds of cycles
- What does a simple transform.position do internally?
- Walk memory linearly (Structs)

JavaScript

JavaScript

- Dynamic typing is slow!

JavaScript

- Dynamic typing is slow!
- Static typing gives you the same performance as C#

JavaScript

- Dynamic typing is slow!
- Static typing gives you the same performance as C#
- 50% the performance of optimized C++ code. 20x faster than Mozilla JavaScript

JavaScript

- Dynamic typing is slow!
- Static typing gives you the same performance as C#
- 50% the performance of optimized C++ code. 20x faster than Mozilla JavaScript
- `#pragma strict` to the rescue

Demo Boids

Questions?