USER MANUAL
Runtime
Mesh
Operator
Ultra Fast Mesh Tools

RUNTIME
MESH
OPERATOR
MANUAL
35.6%
2.0%
29.0%
3.8%
21.1%
18.7%

Contents

1. Introduction

This package provides the core functionality of mesh subdivision and decimation. The Unity renderer recognizes triangular meshes , these mesh modification processes offers changing number of triangles per object.

The core of Runtime Mesh Operator implements the fundamentals of level of detail generation :

- https://en.wikipedia.org/wiki/Level_of_detail

The mesh decimation and subdivision operations are heavy operations but this asset made this dream true.

There are many different types of approaches to manipulate the object’s triangulation structure

The Project has been inspired by the tecniques in the following paper :

http://graphics.stanford.edu/courses/cs468-10-fall/LectureSlides/08_Simplification. pdf

2. FAQs

Q: Can I save the prefabs that I manipulate during run time?

A: You can not save with Runtime Mesh Operator System.It is not designed to do that. But there are tools in the asset store about serialization of runtime instantiated gameobjects.

Q: Why is the performance is absolutely wicked fast ?

A: Because of the threads .Carrying out the algorithm to thread is completely frees Unity’s main thread so there will be no performance loss.

B: Carrying unsafe code to c# have a little bit help too.

Q: Why the source code of core functionality has been provided as a DLL.

A: The CagLib DLL contains unsafe context codes to gain extra speed. For this reason modification of core codes will cause problems.

Q: Is this multi-threaded?

A: Yes. You can even set the thread count .

Q: What is the difference between subdivision and decimation?

A:The decimation operation on mesh decreases the level of detail , vice versa the subdivision increases the level of detail.

Q: Do i have chance to use the functionality without runtime ?

A:Yes. You can do it before the game starts. Please investigate the setup instructions below.

Q: Does it have mesh editor script ?

A:Yes, but you can not change the gameobject before the gamestarts.

3. Demos & Videos

Demos:

Included with the package are two demo scenes displaying some of the capabilities of this plug in. Some details to get you started are presented here:

WhenGameStarts.

This demo demonstrates the opeartions in Unity’s Start() functions. So the gameobjects will be operated in the game initialization process.

Please note that there is RuntimeMeshOperatorInitializerscript attached to the RuntimeMeshOperatorInitializer gameobject. The catlast gameobject has GameObjectOperatorscript attached to it.

You can attach your all game objects GameObjectOperator script and just hit play in this mode.

There is no scripting required with this mode. Just hit play and observe.

RealTime

During game play if you click the OperateMesh!The batch game object operation on 100 gameobjects will kick off.

Please note that there is RuntimeMeshOperatorController sufficient and required to asynchroneously calling theRuntimeMeshOperator .Operate(… function.

This demo how shows how fast Runtime Mesh Operator is , you can observe 100 gameobjects batch process in Real Time.

4. Package Contents & Setup

Since tool has two different types of implementation, it has different entry points and different related scripts.

There are two different entry points into the code:

WhenGameStarts implementation enters the code flow with this code.

```
meshAdapter.Build(newMesh,(int)OperationType, UsePooling,(int
)Level,destroyScripts,Fast);
```

RealTime implementation enters the code flow with this code.

```
RuntimeMeshOperator.Operate(rmrNewMesh,selectedOpera tion,optimizedMemory,rmrFast,rmrColor,
                            destroyGeneratedScripts,factor,processObjectsByTag,rmrTag,processObjectsByLayer,
                            rmrLayer,processObjectsThread,rmrThread );
```

4A **WhenGameStarts**Operations related scripts are:

- RuntimeMeshOperatorInitializer.cs
- GameObjectOperator.cs

4B **RealTime**Operations related scripts are :

- RuntimeMeshOperatorController.cs

MeshAdapter.cs

- RuntimeMeshOperator.cs

4.a **WhenGameStarts**:

Mesh reduce and mesh subdivide before the game starts.

Implementation Steps:

First of all there is no scripting required for this implementation.

1- Add gameobject operator component to the related gameobject.

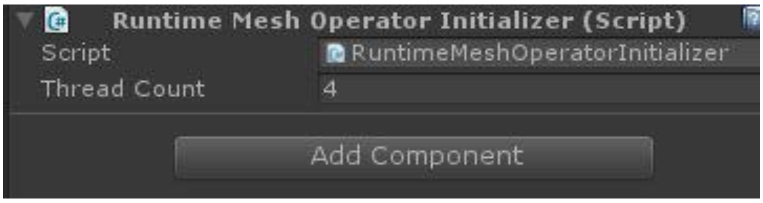
2- There must be a runtimemeshoperatorinitializer prefab in the scene to use eager operations. as shown below. Prefabs are located at the

demo/resources/prefabs location

3- Hit play.

- Runtime mesh operator is a multithreaded software you can redefine the thread count.

➔ RuntimeMeshOperatorInitializer.cs

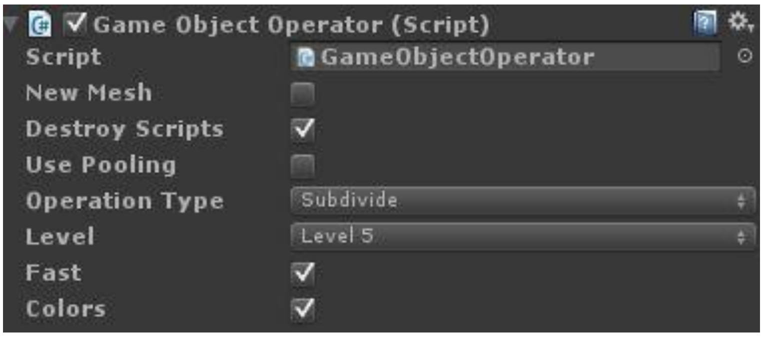


Runtime Mesh Operator Initializer (Script)

Script: RuntimeMeshOperatorInitializer

Thread Count: 4

➔ GameObjectOperator.cs



☑ Game Object Operator (Script)

Script: GameObjectOperator

New Mesh: □

Destroy Scripts: ☑

Use Pooling: □

Operation Type: Subdivide

Level: Level 5

Fast: ☑

Colors: ☑

Features on the component:

*New mesh option is about decision of using a new mesh or using current on object.

*Destroy scripts feature is about whether destroying or not destroying script after operation completed.

*Use pooling feature is about loading resources before the operation starts . If you have many objects this option will reduce memory consumption. And if you are using run time mesh generations systems like marching cubes dual contouring marching tetrahedrons this is what you are looking for.

*Operation type choose list is about selecting mesh operation. There are two types of operations including subdiving or reducing polygons.

*Level parameter is the operation coefficient of operations. You can find your best level coeeficient.

*Fast checkbox disables workload of checking duplicates , checking unused vertices and empty triangle checkback.

*Colors checkbox is about whether disabling the colors in the mesh or not.

4.b RealTime

Implementation steps:

1- Runtime mesh operator controller prefab must be added to scene.

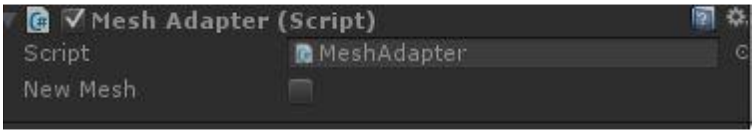
2-To use this approach you just need to call Operate(); function in your script side and you are done!

```
RuntimeMeshOperatorController.instance.Operate();
```

Note:

Please note that if you disable destroy generated scripts options you will notice that each operated object will have meshdata component attached to it.

➔MeshAdapter.cs

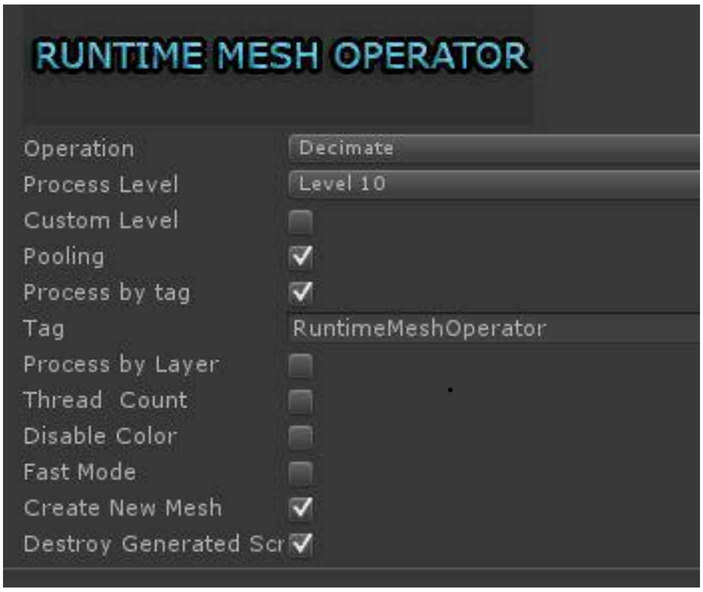


☑ Mesh Adapter (Script)

Script	MeshAdapter
New Mesh	□

If you enable destroy generated scripts option all scripts will be destroyed.

➔RuntimeMeshOperatorController.cs



RUNTIME MESH OPERATOR

Operation: Decimate
Process Level: Level 10
Custom Level: □
Pooling: ☑
Process by tag: ☑
Tag: RuntimeMeshOperator
Process by Layer: □
Thread Count: □
Disable Color: □
Fast Mode: □
Create New Mesh: ☑
Destroy Generated Scr: ☑

Features on the component:

*Custom level parameter is assigning custom operation coefficient. You can find the best floating point number with the textbox appears.

*Process by tag feature lets you making batch operations gameobjects have same tag.

*Process by layer feature lets you making batch operations gameobjects have same layer

*Thread count options id predefining number of threads to trigger .

*Remaining features or properties has been told in the previous section. you can use them safely as stated.

RUNTIME MESH OPERATOR
MANUAL