Simple Jobs

Table of Contents

Script Reference

SimpleJobs

Demos

Scripts

HashCountJobs.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

HashCountJob

```
[BurstCompile]
public struct HashCountJob
: IJob
```

Part of the Spatial Hash process. Counts the number of objects in each hash-cell.

Variables

CellPositions

```
[ReadOnly]
public NativeArray<int3> CellPositions
```

HashSize

```
[ReadOnly]
public int HashSize
```

Counts

```
public NativeArray<int> Counts
```

Methods

Execute

```
public void Execute()
```

C# SpatialHashDemo.cs

Namespaces

SimpleJobs.Demos

```
namespace SimpleJobs.Demos
```

Classes

SpatialHashDemo

```
public class SpatialHashDemo
: MonoBehaviour
```

Demonstration of a Spatial Hash algorithm, which makes use of some the jobs in SimpleJobs.

Classes

Particle

```
[System.Serializable]
public struct Particle
```

Variables

Position

```
public Vector3 Position
```

CurrentCell

```
public int3 CurrentCell
```

Velocity

```
public Vector3 Velocity
```

Radius

```
public float Radius
```

LastCollision

```
public float LastCollision
```

CollisionTimer

```
public float CollisionTimer
```

Variables

NewParticlesBttn

```
[Header("Particles")]
public bool NewParticlesBttn
```

NewParticlesCount

```
public int NewParticlesCount
```

NewParticlesSlider

```
public Slider NewParticlesSlider
```

ParticleBounds

```
public Bounds ParticleBounds
```

Particles

```
public Particle[] Particles
```

ParticlePrefab

```
public GameObject ParticlePrefab
```

maxRadius

```
float maxRadius
```

ShowHashCountsBttn

```
[Header("Hash Performance")]
public bool ShowHashCountsBttn
```

HashCounts

```
public int[] HashCounts
```

FindOverlapsType

```
[Header("Performance Test Controls")]
public int FindOverlapsType
```

EnableFindOverlaps

```
public bool EnableFindOverlaps
```

UpdateParticles

```
public bool UpdateParticles
```

CheckToggle

```
public Toggle CheckToggle
```

UpdateToggle

```
public Toggle UpdateToggle
```

CheckSizeSlider

```
public Slider CheckSizeSlider
```

HashCellSizeSlider

```
public Slider HashCellSizeSlider
```

UpdateTypeSlider

```
public Slider UpdateTypeSlider
```

CheckRadius

```
public float CheckRadius
```

CheckSphere

```
public Transform CheckSphere
```

HashCellSize

```
[Range(.01f, 100f, order = -1)]
public float HashCellSize
```

HashCellMarkers

```
public Transform HashCellMarkers
```

PerformanceResults

```
[Header("Performance Test Results\n (10000 Ticks = 1
Millisecond)")]
[TextArea(3, 10)]
public string PerformanceResults
```

PerformanceText

```
public Text PerformanceText
```

ClearPerformanceData

```
public Button ClearPerformanceData
```

queryData

```
public List<long> queryData
```

qdPC

```
public int qdPC
```

qdHS

```
public int qdHS
```

qdUT

```
public int qdUT
```

qdCS

```
public float qdCS
```

qdCR

```
public float qdCR
```

queryDataMean

```
public float queryDataMean
```

queryDataVariance

```
public float queryDataVariance
```

Methods

Start

```
private void Start()
```

Update

```
private void Update()
```

SavePerformanceData

```
void SavePerformanceData(
     string results,
     long queryTicks,
     int pCount,
     int hSize,
     int updateType,
     float cellSize,
     float cRadius)
```

FindOverlaps1

```
void FindOverlaps1()
```

In this variation, the spatial hash job modifies the array of particles.

FindOverlaps2

```
void FindOverlaps2()
```

In this variation, the job is divided up into many jobs to be multi-threaded. Each hash-cell is given its own job.

FindOverlaps3

```
void FindOverlaps3()
```

In this variation, the spatial hash job returns a list of indicies. Each index in the list points to a particle.

FindOverlapsBruteForce

```
void FindOverlapsBruteForce()
```

OnDrawGizmosSelected

```
private void OnDrawGizmosSelected()
```

C# SpatialHashParticle.cs

Namespaces

SimpleJobs.Demos

```
namespace SimpleJobs.Demos
```

Classes

SpatialHashParticle

```
public class SpatialHashParticle
: MonoBehaviour
```

Particles in the SpatialHash demo are GameObjects, controlled by this MonoBehaviour.

Variables

demo

```
SpatialHashDemo demo
```

Renderer

```
public SpriteRenderer Renderer
```

CollisionGradient

```
public Gradient CollisionGradient
```

Scale

```
public Vector3 Scale
```

Methods

Update

```
private void Update()
```

C# SpatialHashQueryJob.cs

Namespaces

SimpleJobs.Demos

```
namespace SimpleJobs.Demos
```

Classes

SpatialHashQueryIndiciesJob

```
public struct SpatialHashQueryIndiciesJob
: IJob
```

Finds a list of pointers (int index) to objects that are in the queried cells.

Variables

QueriedIndicies

```
[ReadOnly]
public NativeArray<int> QueriedIndicies
```

InvalidCellValue

```
public int InvalidCellValue
```

SortedPointers

```
[ReadOnly]
public NativeArray<int> SortedPointers
```

Sums

```
[ReadOnly]
public NativeArray<int> Sums
```

FoundPointers

```
[WriteOnly]
public NativeList<int> FoundPointers
```

Methods

Execute

```
public void Execute()
```

SpatialHashQuerySphereJob

```
public struct SpatialHashQuerySphereJob
: IJob
```

Finds a list of pointers (int index) to objects that are in the queried cells.

Variables

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckRadius

```
[ReadOnly]
public float CheckRadius
```

CellSize

```
[ReadOnly]
public float CellSize
```

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

Radii

```
[ReadOnly]
public NativeArray<float> Radii
```

SortedPointers

```
[ReadOnly]
public NativeArray<int> SortedPointers
```

Sums

```
[ReadOnly]
public NativeArray<int> Sums
```

FoundPointers

```
[WriteOnly]
public NativeList<int> FoundPointers
```

Each value in this list is: the index of an object found by this query, that object can be found in the Positions array.

CheckedIndicies

```
public NativeHashSet<int> CheckedIndicies
```

Methods

Execute

```
public void Execute()
```

SpatialHashQueryParticlesSphereJob

```
[BurstCompile]
public struct SpatialHashQueryParticlesSphereJob
: IJob
```

Finds a list of pointers (int index) to objects that are in the queried cells.

Variables

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckRadius

```
[ReadOnly]
public float CheckRadius
```

CellSize

```
[ReadOnly]
public float CellSize
```

MaxParticleRadius

```
[ReadOnly]
public float MaxParticleRadius
```

SortedPointers

```
[ReadOnly]
public NativeArray<int> SortedPointers
```

Sums

```
[ReadOnly]
public NativeArray<int> Sums
```

Time

```
[ReadOnly]
public float Time
```

Particles

```
public NativeArray<SpatialHashDemo.Particle> Particles
```

CheckedIndicies

```
public NativeHashSet<int> CheckedIndicies
```

Methods

Execute

```
public void Execute()
```

SpatialHashQueryParticlesCellJob

```
[BurstCompile]
public struct SpatialHashQueryParticlesCellJob
: IJob
```

Finds a list of pointers (int index) to objects that are in the queried cells.

Variables

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckRadius

```
[ReadOnly]
public float CheckRadius
```

HashIndex

```
[ReadOnly]
public int HashIndex
```

SortedPointers

```
[ReadOnly]
public NativeArray<int> SortedPointers
```

Sums

```
[ReadOnly]
public NativeArray<int> Sums
```

Time

```
[ReadOnly]
public float Time
```

Particles

```
[NativeDisableContainerSafetyRestriction]
public NativeArray<SpatialHashDemo.Particle> Particles
```

Methods

Execute

```
public void Execute()
```

SpatialHashQueryBoolCellJob

```
[BurstCompile]
public struct SpatialHashQueryBoolCellJob
: IJob
```

Finds a list of pointers (int index) to objects that are in the queried cells.

Variables

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckRadius

```
[ReadOnly]
public float CheckRadius
```

CellSize

```
[ReadOnly]
public float CellSize
```

MaxParticleRadius

```
[ReadOnly]
public float MaxParticleRadius
```

Cell

```
[ReadOnly]
public int3 Cell
```

SortedPointers

```
[ReadOnly]
public NativeArray<int> SortedPointers
```

Sums

```
[ReadOnly]
public NativeArray<int> Sums
```

Time

```
[ReadOnly]
public float Time
```

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

Radii

```
[ReadOnly]
public NativeArray<float> Radii
```

IsFoundArray

```
public NativeArray<bool> IsFoundArray
```

Methods

Execute

```
public void Execute()
```

C# SpatialHashSortJob.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

SpatialHashSortJob

```
[BurstCompile]
public struct SpatialHashSortJob
: IJob
```

Part of the Spatial Hash process. Takes a list of objects, creates and sorts their pointers (int index), such that all the objects in each hash-cell are together in the list.

Variables

CellPositions

```
[ReadOnly]
public NativeArray<int3> CellPositions
```

Sums

```
public NativeArray<int> Sums
```

SortedPointers

```
[WriteOnly]
public NativeArray<int> SortedPointers
```

Methods

Execute

```
public void Execute()
```

StatisticsDemo.cs

Namespaces

SimpleJobs.Demos

```
namespace SimpleJobs.Demos
```

Classes

StatisticsDemo

```
[ExecuteInEditMode]
public class StatisticsDemo
: MonoBehaviour
```

Demonstration of how to use Mean, Variance, and Covariance jobs.

Variables

ValuesA

```
[Header("Use the 'Bttn' buttons to calculate the desired
values")]
public float[] ValuesA
```

ValuesB

```
public float[] ValuesB
```

MeanA

```
public float MeanA
```

VarianceA

```
public float VarianceA
```

MeanB

```
public float MeanB
```

VarianceB

```
public float VarianceB
```

Covariance

```
public float Covariance
```

SampleVarianceOrPop

```
[Tooltip("Sample Variance? or if not, Population
Variance")]
public bool SampleVarianceOrPop
```

FindVarianceBttn

```
public bool FindVarianceBttn
```

FindCovarianceBttn

```
public bool FindCovarianceBttn
```

MinA

```
public float MinA
```

MaxA

```
public float MaxA
```

MinB

```
public float MinB
```

MaxB

```
public float MaxB
```

FindMinMaxBttn

```
public bool FindMinMaxBttn
```

Methods

Update

```
private void Update()
```

FindMeanAndVariance

```
public static void FindMeanAndVariance(
     float[] values,
     ref float mean,
     ref float variance,
     bool sampleVariance)
```

FindMeansAndCovariance

```
public static void FindMeansAndCovariance(
     float[] valuesA,
     float[] valuesB,
     ref float meanA,
     ref float meanB,
     ref float covariance,
     bool sampleCovariance)
```

FindMinMax

```
public static void FindMinMax(
     float[] values,
     ref float min,
     ref float max)
```

OnDrawGizmos

```
private void OnDrawGizmos()
```

Scripts

Hash.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

Hash

```
public static class Hash
```

Methods

Cell

```
public static Vector3Int Cell(
     Transform transform,
     float cellSize)
```

Find the cell the given transform is in.

cellSize: Size of each cell.

Cell

```
public static Vector3Int Cell(
     Vector3 position,
     float cellSize)
```

Find the cell the given transform is in.

cellSize: Size of each cell.

Index

```
public static int Index(
     Transform transform,
     int hashLength,
     float cellSize)
```

Finds the index of the cell the given transform is in.

hashLength: Length of the hash arrays.
cellSize: Size of each cell.

Index

```
public static int Index(
     Vector3 position,
     int hashLength,
     float cellSize)
```

Finds the index of the cell the given position is in.

hashLength: Length of the hash arrays.
cellSize: Size of each cell.

Index

```
public static int Index(
     Vector3Int cell,
     int hashLength)
```

Finds the hash-index of the given cell.

hashLength: Length of the hash arrays.

Cell

```
public static int3 Cell(
     float3 position,
     float cellSize)
```

Find the cell the given transform is in.

cellSize: Size of each cell.

Index

```
public static int Index(
     float3 position,
     int hashLength,
     float cellSize)
```

Finds the index of the cell the given position is in.

hashLength: Length of the hash arrays.
cellSize: Size of each cell.

Index

```
public static int Index(
     int3 cell,
     int hashLength)
```

Finds the hash-index of the given cell.

hashLength: Length of the hash arrays.

Jobs

BasicStatisticsJobs.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

MeanJobFloat

```
[BurstCompile]
public struct MeanJobFloat
: IJob
```

Variables

Given

```
[ReadOnly]
public NativeArray<float> Given
```

Result

```
[WriteOnly]
public NativeArray<float> Result
```

Methods

Execute

```
public void Execute()
```

MeanJobFloat2

```
[BurstCompile]
public struct MeanJobFloat2
: IJob
```

Variables

Given

```
[ReadOnly]
public NativeArray<float2> Given
```

Result

```
[WriteOnly]
public NativeArray<float2> Result
```

Methods

Execute

```
public void Execute()
```

MeanJobFloat3

```
[BurstCompile]
public struct MeanJobFloat3
: IJob
```

Variables

Given

```
[ReadOnly]
public NativeArray<float3> Given
```

Result

```
[WriteOnly]
public NativeArray<float3> Result
```

Methods

Execute

```
public void Execute()
```

MeanJobFloat4

```
[BurstCompile]
public struct MeanJobFloat4
: IJob
```

Variables

Given

```
[ReadOnly]
public NativeArray<float4> Given
```

Result

```
[WriteOnly]
public NativeArray<float4> Result
```

Methods

Execute

```
public void Execute()
```

VarianceJobFloat

```
[BurstCompile]
public struct VarianceJobFloat
: IJob
```

Finds the variance of a set data. To set up given data: use MeanJobFloat, AddUniformFloat, and SquaresJobFloat.

Variables

MeanDiffSquares

```
[ReadOnly]
public NativeArray<float> MeanDiffSquares
```

To create this array: Subtract the mean from each value, and then square each value.

Result

```
[WriteOnly]
public NativeArray<float> Result
```

SampleOrPopulation

```
[WriteOnly]
public bool SampleOrPopulation
```

IsSample

```
public bool IsSample
```

IsPopulation

```
public bool IsPopulation
```

Methods

Execute

```
public void Execute()
```

VarianceJobFloat2

```
[BurstCompile]
public struct VarianceJobFloat2
: IJob
```

Finds the variance of a set data. To set up given data: use MeanJobFloat2, AddUniformFloat2, and SquaresJobFloat2.

Variables

MeanDiffSquares

```
[ReadOnly]
public NativeArray<float2> MeanDiffSquares
```

To create this array: Subtract the mean from each value, and then square each value.

Result

```
[WriteOnly]
public NativeArray<float2> Result
```

SampleOrPopulation

```
[WriteOnly]
public bool SampleOrPopulation
```

IsSample

```
public bool IsSample
```

IsPopulation

```
public bool IsPopulation
```

Methods

Execute

```
public void Execute()
```

VarianceJobFloat3

```
[BurstCompile]
public struct VarianceJobFloat3
: IJob
```

Finds the variance of a set data. To set up given data: use MeanJobFloat3, AddUniformFloat3, and SquaresJobFloat3.

Variables

MeanDiffSquares

```
[ReadOnly]
public NativeArray<float3> MeanDiffSquares
```

To create this array: Subtract the mean from each value, and then square each value.

Result

```
[WriteOnly]
public NativeArray<float3> Result
```

SampleOrPopulation

```
[WriteOnly]
public bool SampleOrPopulation
```

IsSample

```
public bool IsSample
```

IsPopulation

```
public bool IsPopulation
```

Methods

Execute

```
public void Execute()
```

VarianceJobFloat4

```
[BurstCompile]
public struct VarianceJobFloat4
: IJob
```

Finds the variance of a set data. To set up given data: use MeanJobFloat4, AddUniformFloat4, and SquaresJobFloat4.

Variables

MeanDiffSquares

```
[ReadOnly]
public NativeArray<float4> MeanDiffSquares
```

To create this array: Subtract the mean from each value, and then square each value.

Result

```
[WriteOnly]
public NativeArray<float4> Result
```

SampleOrPopulation

```
[WriteOnly]
public bool SampleOrPopulation
```

IsSample

```
public bool IsSample
```

IsPopulation

```
public bool IsPopulation
```

Methods

Execute

```
public void Execute()
```

CovarianceJobFloat

```
[BurstCompile]
public struct CovarianceJobFloat
: IJob
```

Finds the covariance of a set data. To set up given data sets: use MeanJobFloat, and AddUniformFloat.

Variables

MeanDiffA

```
[ReadOnly]
public NativeArray<float> MeanDiffA
```

To create these arrays: Subtract the mean from each value.

MeanDiffB

```
[ReadOnly]
public NativeArray<float> MeanDiffB
```

To create these arrays: Subtract the mean from each value.

Result

```
[WriteOnly]
public NativeArray<float> Result
```

SampleOrPopulation

```
[WriteOnly]
public bool SampleOrPopulation
```

IsSample

```
public bool IsSample
```

IsPopulation

```
public bool IsPopulation
```

Methods

Execute

```
public void Execute()
```

C# CheckDistanceJobs.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

CheckSpheresDistanceJobIndexList

```
[BurstCompile]
public struct CheckSpheresDistanceJobIndexList
: IJob
```

Creates a list of indecies (int), where each index corresponds to a given sphere. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

Radii

```
[ReadOnly]
public NativeArray<float> Radii
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckRadius

```
[ReadOnly]
public float CheckRadius
```

FoundPointers

```
[WriteOnly]
public NativeList<int> FoundPointers
```

Methods

Execute

```
public void Execute()
```

CheckUniformSpheresDistanceJobIndexList

```
[BurstCompile]
public struct CheckUniformSpheresDistanceJobIndexList
: IJob
```

Creates a list of indecies (int), where each index corresponds to a given sphere. Faster than non-uniform, uses square magnitudes and does not need square roots. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckRadius

```
[ReadOnly]
public float CheckRadius
```

Radius of the check sphere + Radius of the spheres. Uniform: given spheres are all the same radius.

FoundPointers

```
[WriteOnly]
public NativeList<int> FoundPointers
```

Methods

Execute

```
public void Execute()
```

CheckSpheresDistanceJobTrim

```
[BurstCompile]
public struct CheckSpheresDistanceJobTrim
: IJob
```

Takes the given arrays, and creates new ones were all of the spheres are within the checked distance. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

Radii

```
[ReadOnly]
public NativeArray<float> Radii
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckRadius

```
[ReadOnly]
public float CheckRadius
```

TrimmedPositions

```
[WriteOnly]
public NativeArray<float3> TrimmedPositions
```

TrimmedRadii

```
[WriteOnly]
public NativeArray<float> TrimmedRadii
```

Methods

Execute

```
public void Execute()
```

CheckUniformSpheresDistanceJobTrim

```
[BurstCompile]
public struct CheckUniformSpheresDistanceJobTrim
: IJob
```

Takes the given arrays, and creates new ones were all of the spheres are within the checked distance. Faster than non-uniform, uses square magnitudes and does not need square roots. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckRadius

```
[ReadOnly]
public float CheckRadius
```

Radius of the check sphere + Radius of the spheres. Uniform: given spheres are all the same radius.

TrimmedPositions

```
[WriteOnly]
public NativeArray<float3> TrimmedPositions
```

Methods

Execute

```
public void Execute()
```

CheckAABBsDistanceJobIndexList

```
[BurstCompile]
public struct CheckAABBsDistanceJobIndexList
: IJob
```

Creates a list of indecies (int), where each index corresponds to a given axis-aligned bounding box. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

Sizes

```
[ReadOnly]
public NativeArray<float3> Sizes
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckSize

```
[ReadOnly]
public float3 CheckSize
```

FoundPointers

```
[WriteOnly]
public NativeList<int> FoundPointers
```

Methods

Execute

```
public void Execute()
```

CheckUniformAABBsDistanceJobIndexList

```
[BurstCompile]
public struct CheckUniformAABBsDistanceJobIndexList
: IJob
```

Creates a list of indecies (int), where each index corresponds to a given axis-aligned bounding box. Faster than non-uniform. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckSize

```
[ReadOnly]
public float3 CheckSize
```

Size of the check box + Size of the boxes. Uniform: given boxes are all the same size.

FoundPointers

```
[WriteOnly]
public NativeList<int> FoundPointers
```

Methods

Execute

```
public void Execute()
```

CheckAABBsDistanceJobTrim

```
[BurstCompile]
public struct CheckAABBsDistanceJobTrim
: IJob
```

Takes the given arrays, and creates new ones were all of the AABB are within the checked box. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

Sizes

```
[ReadOnly]
public NativeArray<float3> Sizes
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckSize

```
[ReadOnly]
public float3 CheckSize
```

TrimmedPositions

```
[WriteOnly]
public NativeArray<float3> TrimmedPositions
```

TrimmedSizes

```
[WriteOnly]
public NativeArray<float3> TrimmedSizes
```

Methods

Execute

```
public void Execute()
```

CheckUniformAABBsDistanceTrim

```
[BurstCompile]
public struct CheckUniformAABBsDistanceTrim
: IJob
```

Takes the given arrays, and creates new ones were all of the AABB are within the checked box. Faster than non-uniform. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckSize

```
[ReadOnly]
public float3 CheckSize
```

Size of the check box + Size of the boxes. Uniform: given boxes are all the same size.

TrimmedPositions

```
[WriteOnly]
public NativeArray<float3> TrimmedPositions
```

Methods

Execute

```
public void Execute()
```

CheckAABBsDistanceJobBool

```
[BurstCompile]
public struct CheckAABBsDistanceJobBool
: IJobParallelFor
```

Creates a new bool array, where each value is true if the corresponding box is within the check box. Faster than non-uniform. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

Sizes

```
[ReadOnly]
public NativeArray<float3> Sizes
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckSize

```
[ReadOnly]
public float3 CheckSize
```

Results

```
[WriteOnly]
public NativeArray<bool> Results
```

Methods

Execute

```
public void Execute(
     int i)
```

CheckAABBsDistanceJobBool3

```
[BurstCompile]
public struct CheckAABBsDistanceJobBool3
: IJobParallelFor
```

Creates a new bool array, where each value is true if the corresponding box is within the check box. Faster than non-uniform. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

Sizes

```
[ReadOnly]
public NativeArray<float3> Sizes
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckSize

```
[ReadOnly]
public float3 CheckSize
```

Results

```
[WriteOnly]
public NativeArray<bool3> Results
```

Methods

Execute

```
public void Execute(
     int i)
```

CheckUniformAABBsDistanceJobBool

```
[BurstCompile]
public struct CheckUniformAABBsDistanceJobBool
: IJobParallelFor
```

Creates a new bool array, where each value is true if the corresponding box is within the check box. Faster than non-uniform. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckSize

```
[ReadOnly]
public float3 CheckSize
```

Results

```
[WriteOnly]
public NativeArray<bool> Results
```

Methods

Execute

```
public void Execute(
     int i)
```

CheckUniformAABBsDistanceJobBool3

```
[BurstCompile]
public struct CheckUniformAABBsDistanceJobBool3
: IJobParallelFor
```

Creates a new bool array, where each value is true if the corresponding box is within the check box. Faster than non-uniform. Brute Force : Used for performance comparisons.

Variables

Positions

```
[ReadOnly]
public NativeArray<float3> Positions
```

CheckCenter

```
[ReadOnly]
public float3 CheckCenter
```

CheckSize

```
[ReadOnly]
public float3 CheckSize
```

Results

```
[WriteOnly]
public NativeArray<bool3> Results
```

Methods

Execute

```
public void Execute(
     int i)
```

C# IfSetJobs.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

IfSetJobFloat

```
[BurstCompile]
public struct IfSetJobFloat
: IJobParallelFor
```

Writes the Value to each index where the given Checks are true.

Variables

Checks

```
[ReadOnly]
NativeArray<bool> Checks
```

Value

```
[ReadOnly]
NativeArray<float> Value
```

Must be length 1.

Results

```
[WriteOnly]
NativeArray<float> Results
```

Should be same length as Checks.

Methods

Execute

```
public void Execute(
     int index)
```

IfSetJobFloat2

```
[BurstCompile]
public struct IfSetJobFloat2
: IJobParallelFor
```

Variables

Checks

```
[ReadOnly]
NativeArray<bool> Checks
```

Value

```
[ReadOnly]
NativeArray<float2> Value
```

Must be length 1.

Results

```
[WriteOnly]
NativeArray<float2> Results
```

Should be same length as Checks.

Methods

Execute

```
public void Execute(
     int index)
```

IfSetJobFloat3

```
[BurstCompile]
public struct IfSetJobFloat3
: IJobParallelFor
```

Variables

Checks

```
[ReadOnly]
NativeArray<bool> Checks
```

Value

```
[ReadOnly]
NativeArray<float3> Value
```

Must be length 1.

Results

```
[WriteOnly]
NativeArray<float3> Results
```

Should be same length as Checks.

Methods

Execute

```
public void Execute(
     int index)
```

IfSetJobFloat4

```
[BurstCompile]
public struct IfSetJobFloat4
: IJobParallelFor
```

Variables

Checks

```
[ReadOnly]
NativeArray<bool> Checks
```

Value

```
[ReadOnly]
NativeArray<float4> Value
```

Must be length 1.

Results

```
[WriteOnly]
NativeArray<float4> Results
```

Should be same length as Checks.

Methods

Execute

```
public void Execute(
     int index)
```

IfSetJobInt

```
[BurstCompile]
public struct IfSetJobInt
: IJobParallelFor
```

Variables

Checks

```
[ReadOnly]
NativeArray<bool> Checks
```

Value

```
[ReadOnly]
NativeArray<int> Value
```

Must be length 1.

Results

```
[WriteOnly]
NativeArray<int> Results
```

Should be same length as Checks.

Methods

Execute

```
public void Execute(
     int index)
```

IfSetJobInt2

```
[BurstCompile]
public struct IfSetJobInt2
: IJobParallelFor
```

Variables

Checks

```
[ReadOnly]
NativeArray<bool> Checks
```

Value

```
[ReadOnly]
NativeArray<int2> Value
```

Must be length 1.

Results

```
[WriteOnly]
NativeArray<int2> Results
```

Should be same length as Checks.

Methods

Execute

```
public void Execute(
     int index)
```

IfSetJobInt3

```
[BurstCompile]
public struct IfSetJobInt3
: IJobParallelFor
```

Variables

Checks

```
[ReadOnly]
NativeArray<bool> Checks
```

Value

```
[ReadOnly]
NativeArray<int3> Value
```

Must be length 1.

Results

```
[WriteOnly]
NativeArray<int3> Results
```

Should be same length as Checks.

Methods

Execute

```
public void Execute(
     int index)
```

IfSetJobInt4

```
[BurstCompile]
public struct IfSetJobInt4
: IJobParallelFor
```

Variables

Checks

```
[ReadOnly]
NativeArray<bool> Checks
```

Value

```
[ReadOnly]
NativeArray<int4> Value
```

Must be length 1.

Results

```
[WriteOnly]
NativeArray<int4> Results
```

Should be same length as Checks.

Methods

Execute

```
public void Execute(
     int index)
```

C# MinMaxJobs.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

MinJobFloat

```
[BurstCompile]
public struct MinJobFloat
: IJob
```

Finds the minimum value in the given array

Variables

Given

```
[ReadOnly]
public NativeArray<float> Given
```

Result

```
public float Result
```

Methods

Execute

```
public void Execute()
```

MaxJobFloat

```
[BurstCompile]
public struct MaxJobFloat
: IJob
```

Finds the minimum value in the given array

Variables

Given

```
[ReadOnly]
public NativeArray<float> Given
```

Result

```
public float Result
```

Methods

Execute

```
public void Execute()
```

ParallelOperationJobs.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

AddUniformJobFloat

```
[BurstCompile]
public struct AddUniformJobFloat
: IJobParallelFor
```

Variables

Given

```
[ReadOnly]
public NativeArray<float> Given
```

Value

```
[ReadOnly]
public NativeArray<float> Value
```

Only reads value at index = 0. With burst, this is faster than a a simple float

Results

```
[WriteOnly]
public NativeArray<float> Results
```

Methods

Execute

```
public void Execute(
     int index)
```

AddUniformJobFloat2

```
[BurstCompile]
public struct AddUniformJobFloat2
: IJobParallelFor
```

Variables

Given

```
[ReadOnly]
public NativeArray<float2> Given
```

Value

```
[ReadOnly]
public NativeArray<float2> Value
```

Only reads value at index = 0. With burst, this is faster than a a simple float

Results

```
[WriteOnly]
public NativeArray<float2> Results
```

Methods

Execute

```
public void Execute(
     int index)
```

AddUniformJobFloat3

```
[BurstCompile]
public struct AddUniformJobFloat3
: IJobParallelFor
```

Variables

Given

```
[ReadOnly]
public NativeArray<float3> Given
```

Value

```
[ReadOnly]
public NativeArray<float3> Value
```

Only reads value at index = 0. With burst, this is faster than a a simple float

Results

```
[WriteOnly]
public NativeArray<float3> Results
```

Methods

Execute

```
public void Execute(
     int index)
```

AddUniformJobFloat4

```
[BurstCompile]
public struct AddUniformJobFloat4
: IJobParallelFor
```

Variables

Given

```
[ReadOnly]
public NativeArray<float4> Given
```

Value

```
[ReadOnly]
public NativeArray<float4> Value
```

Only reads value at index = 0. With burst, this is faster than a a simple float

Results

```
[WriteOnly]
public NativeArray<float4> Results
```

Methods

Execute

```
public void Execute(
     int index)
```

AddPairJobFloat

```
[BurstCompile]
public struct AddPairJobFloat
: IJobParallelFor
```

Variables

GivenA

```
[ReadOnly]
public NativeArray<float> GivenA
```

Must be same length

GivenB

```
[ReadOnly]
public NativeArray<float> GivenB
```

Must be same length

Results

```
[WriteOnly]
public NativeArray<float> Results
```

Methods

Execute

```
public void Execute(
     int index)
```

AddPairJobFloat2

```
[BurstCompile]
public struct AddPairJobFloat2
: IJobParallelFor
```

Variables

GivenA

```
[ReadOnly]
public NativeArray<float2> GivenA
```

Must be same length

GivenB

```
[ReadOnly]
public NativeArray<float2> GivenB
```

Must be same length

Results

```
[WriteOnly]
public NativeArray<float2> Results
```

Methods

Execute

```
public void Execute(
     int index)
```

AddPairJobFloat3

```
[BurstCompile]
public struct AddPairJobFloat3
: IJobParallelFor
```

Variables

GivenA

```
[ReadOnly]
public NativeArray<float3> GivenA
```

Must be same length

GivenB

```
[ReadOnly]
public NativeArray<float3> GivenB
```

Must be same length

Results

```
[WriteOnly]
public NativeArray<float3> Results
```

Methods

Execute

```
public void Execute(
     int index)
```

AddPairJobFloat4

```
[BurstCompile]
public struct AddPairJobFloat4
: IJobParallelFor
```

Variables

GivenA

```
[ReadOnly]
public NativeArray<float4> GivenA
```

Must be same length

GivenB

```
[ReadOnly]
public NativeArray<float4> GivenB
```

Must be same length

Results

```
[WriteOnly]
public NativeArray<float4> Results
```

Methods

Execute

```
public void Execute(
     int index)
```

SquaresJobFloat

```
[BurstCompile]
public struct SquaresJobFloat
: IJobParallelFor
```

Variables

Given

```
[ReadOnly]
public NativeArray<float> Given
```

Must be same length

Squares

```
[WriteOnly]
public NativeArray<float> Squares
```

Methods

Execute

```
public void Execute(
     int i)
```

SquaresJobFloat2

```
[BurstCompile]
public struct SquaresJobFloat2
: IJobParallelFor
```

Variables

Given

```
[ReadOnly]
public NativeArray<float2> Given
```

Must be same length

Squares

```
[WriteOnly]
public NativeArray<float2> Squares
```

Methods

Execute

```
public void Execute(
     int i)
```

SquaresJobFloat3

```
[BurstCompile]
public struct SquaresJobFloat3
: IJobParallelFor
```

Variables

Given

```
[ReadOnly]
public NativeArray<float3> Given
```

Must be same length

Squares

```
[WriteOnly]
public NativeArray<float3> Squares
```

Methods

Execute

```
public void Execute(
     int i)
```

SquaresJobFloat4

```
[BurstCompile]
public struct SquaresJobFloat4
: IJobParallelFor
```

Variables

Given

```
[ReadOnly]
public NativeArray<float4> Given
```

Must be same length

Squares

```
[WriteOnly]
public NativeArray<float4> Squares
```

Methods

Execute

```
public void Execute(
     int i)
```

C# PartialSumJobs.cs

Namespaces

SimpleJobs

```
namespace SimpleJobs
```

Classes

PartialSumJobInt

```
[BurstCompile]
public struct PartialSumJobInt
: IJob
```

Computes partial sums of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<int> Given
```

Sums

```
[WriteOnly]
public NativeArray<int> Sums
```

Methods

Execute

```
public void Execute()
```

PartialSumJobInt2

```
[BurstCompile]
public struct PartialSumJobInt2
: IJob
```

Computes partial sums of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<int2> Given
```

Sums

```
[WriteOnly]
public NativeArray<int2> Sums
```

Methods

Execute

```
public void Execute()
```

PartialSumJobInt3

```
[BurstCompile]
public struct PartialSumJobInt3
: IJob
```

Computes partial sums of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<int3> Given
```

Sums

```
[WriteOnly]
public NativeArray<int3> Sums
```

Methods

Execute

```
public void Execute()
```

PartialSumJobInt4

```
[BurstCompile]
public struct PartialSumJobInt4
: IJob
```

Computes partial sums of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<int4> Given
```

Sums

```
[WriteOnly]
public NativeArray<int4> Sums
```

Methods

Execute

```
public void Execute()
```

PartialSumJobFloat

```
[BurstCompile]
public struct PartialSumJobFloat
: IJob
```

Computes partial sums of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<float> Given
```

Sums

```
[WriteOnly]
public NativeArray<float> Sums
```

Methods

Execute

```
public void Execute()
```

PartialSumJobFloat2

```
[BurstCompile]
public struct PartialSumJobFloat2
: IJob
```

Computes partial sums of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<float2> Given
```

Sums

```
[WriteOnly]
public NativeArray<float2> Sums
```

Methods

Execute

```
public void Execute()
```

PartialSumJobFloat3

```
[BurstCompile]
public struct PartialSumJobFloat3
: IJob
```

Computes partial sums of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<float3> Given
```

Sums

```
[WriteOnly]
public NativeArray<float3> Sums
```

Methods

Execute

```
public void Execute()
```

PartialSumJobFloat4

```
[BurstCompile]
public struct PartialSumJobFloat4
: IJob
```

Computes partial sums of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<float4> Given
```

Sums

```
[WriteOnly]
public NativeArray<float4> Sums
```

Methods

Execute

```
public void Execute()
```

PartialCountJobBool

```
[BurstCompile]
public struct PartialCountJobBool
: IJob
```

Computes partial counts of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<bool> Given
```

Counts

```
[WriteOnly]
public NativeArray<int> Counts
```

Methods

Execute

```
public void Execute()
```

PartialSumJobBool2

```
[BurstCompile]
public struct PartialSumJobBool2
: IJob
```

Computes partial counts of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<bool2> Given
```

Counts

```
[WriteOnly]
public NativeArray<int2> Counts
```

Methods

Execute

```
public void Execute()
```

PartialCountJobBool3

```
[BurstCompile]
public struct PartialCountJobBool3
: IJob
```

Computes partial counts of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<bool3> Given
```

Counts

```
[WriteOnly]
public NativeArray<int3> Counts
```

Methods

Execute

```
public void Execute()
```

PartialCountJobBool4

```
[BurstCompile]
public struct PartialCountJobBool4
: IJob
```

Computes partial counts of a given list.

Variables

Given

```
[ReadOnly]
public NativeArray<bool4> Given
```

Counts

```
[WriteOnly]
public NativeArray<int4> Counts
```

Methods

Execute

```
public void Execute()
```