intel GAME DEVELOPERS CONFERENCE

CREATING A SCALABLE AND DESTRUCTIBLE WORLD IN HITMAN* 2

21st March 2019

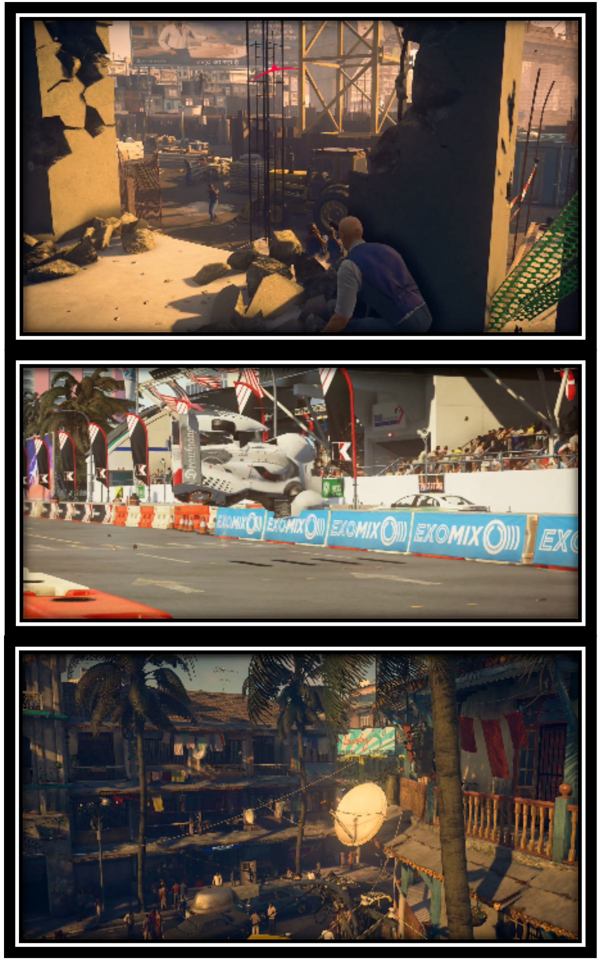
Introduction

- Leigh Davies **Intel**
- *Leigh.Davies@Intel.com*
- Senior Game / Graphics Application Engineer, UK.

- James Vango-Brown **IOI**
- *jamesv@ioi.dk*
- Senior Gameplay / Physics Programmer, Hitman Series

Agenda

- Part 1: Introduction
- Part 2: Creating Realism
  - Crowd System
  - Rendering
  - Audio
- Part 3: Why Destruction?
- Part 4: The Destruction Runtime
- Part 5: The Content Pipeline
- Part 7: Conclusion/Q&A

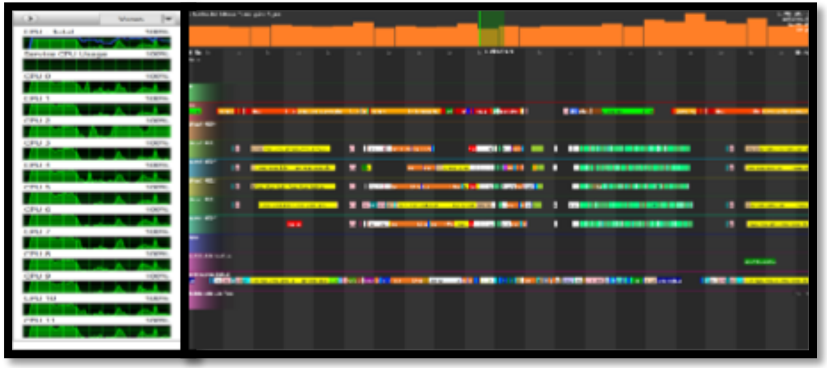


INTRODUCTION

Why are we here

The World of Hitman

- **Third-person stealth video game**
- **Unique varied levels, cities to jungles**
- **Why was HITMAN a good fit for CPU work?**



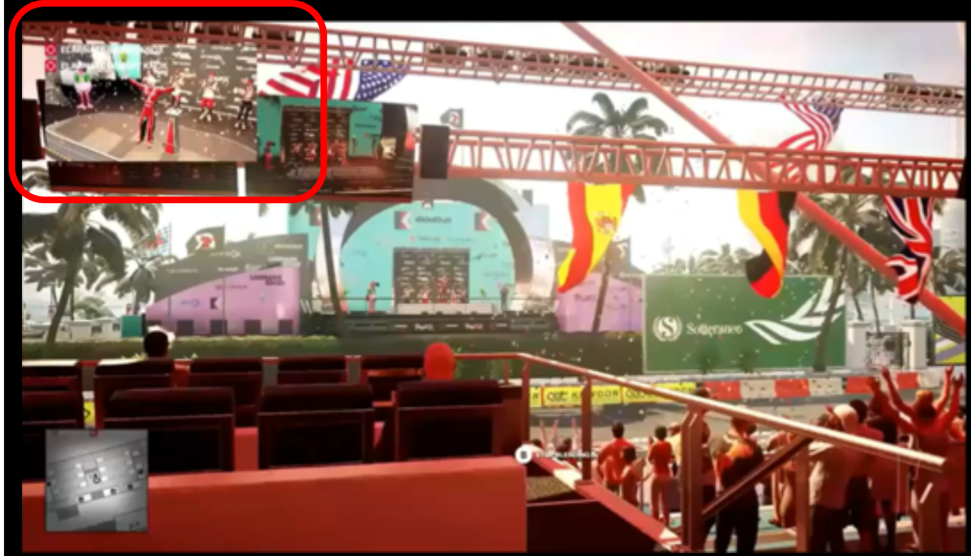
- **HITMAN was already well threaded**
- **Newer processors provided more cores**

Game design objectives:

- Create larger living levels
  - Picture in Picture
  - More complex environments
  - Larger draw distances
- Make the crowd an integral part of the experience
- More opportunity to interact with the world:-

  “Make the world your weapon”

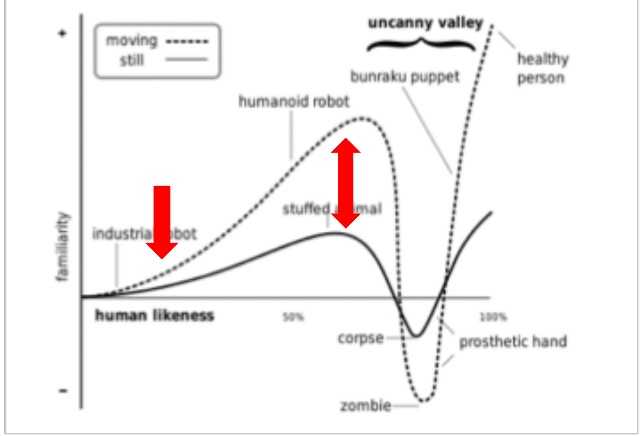
Scale on CPU
Rendering
AI/ Animation
Physics



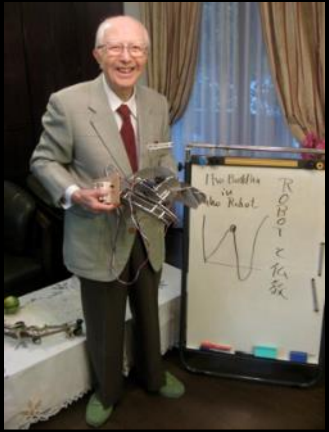
”It’s not an eSport. It’s a sandbox. You’re supposed to have fun with it.”:
Christian Elverdam

Realism is more than just Pixel Quality

Uncanny Valley

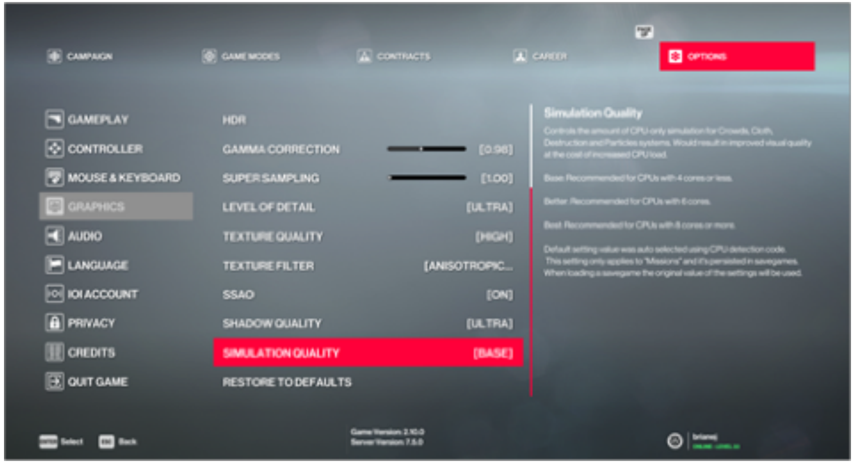


The common unsettling feeling people experience when androids closely resemble humans but are not quite convincingly realistic



Masahiro Mori

Realism is more than just Pixel Quality



HDR
GAMMA CORRECTION [0.98]
SUPER SAMPLING [1.00]
LEVEL OF DETAIL [ULTRA]
TEXTURE QUALITY [HIGH]
TEXTURE FILTER [ANISOTROPIC...
SSAO [ON]
SHADOW QUALITY [ULTRA]
SIMULATION QUALITY [BASE]
RESTORE TO DEFAULTS

Simulation Quality

Controls the amount of CPU-only simulation for Crowds, Cloth, Destruction and Particles systems. Would result in improved visual quality at the cost of increased CPU load.

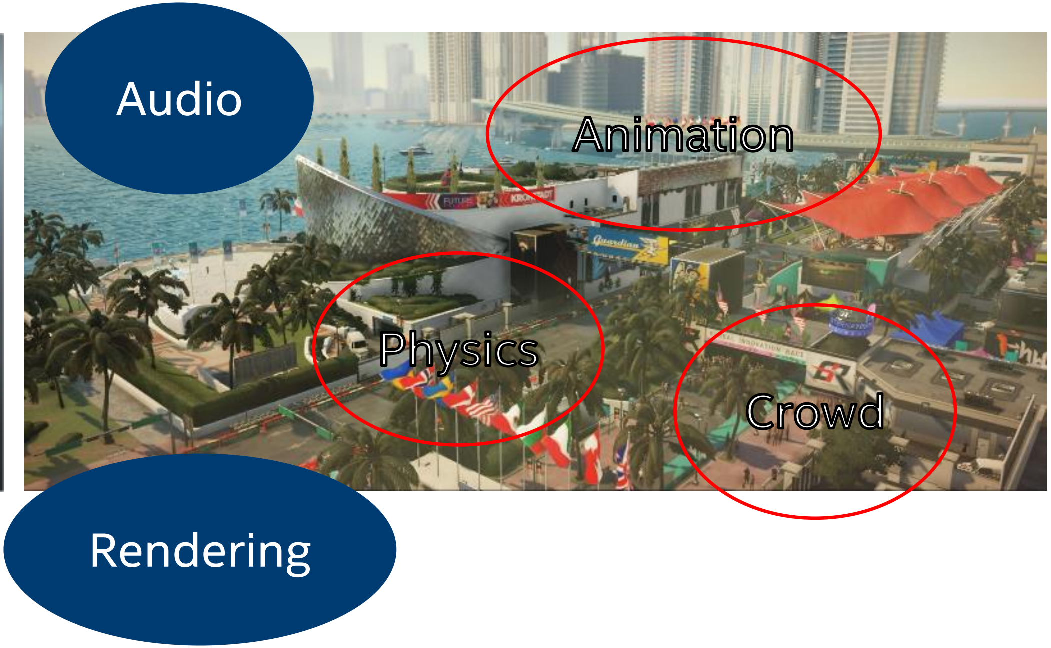
Base: Recommended for CPUs with 4 cores or less.

Better: Recommended for CPUs with 6 cores.

Best: Recommended for CPUs with 8 cores or more.

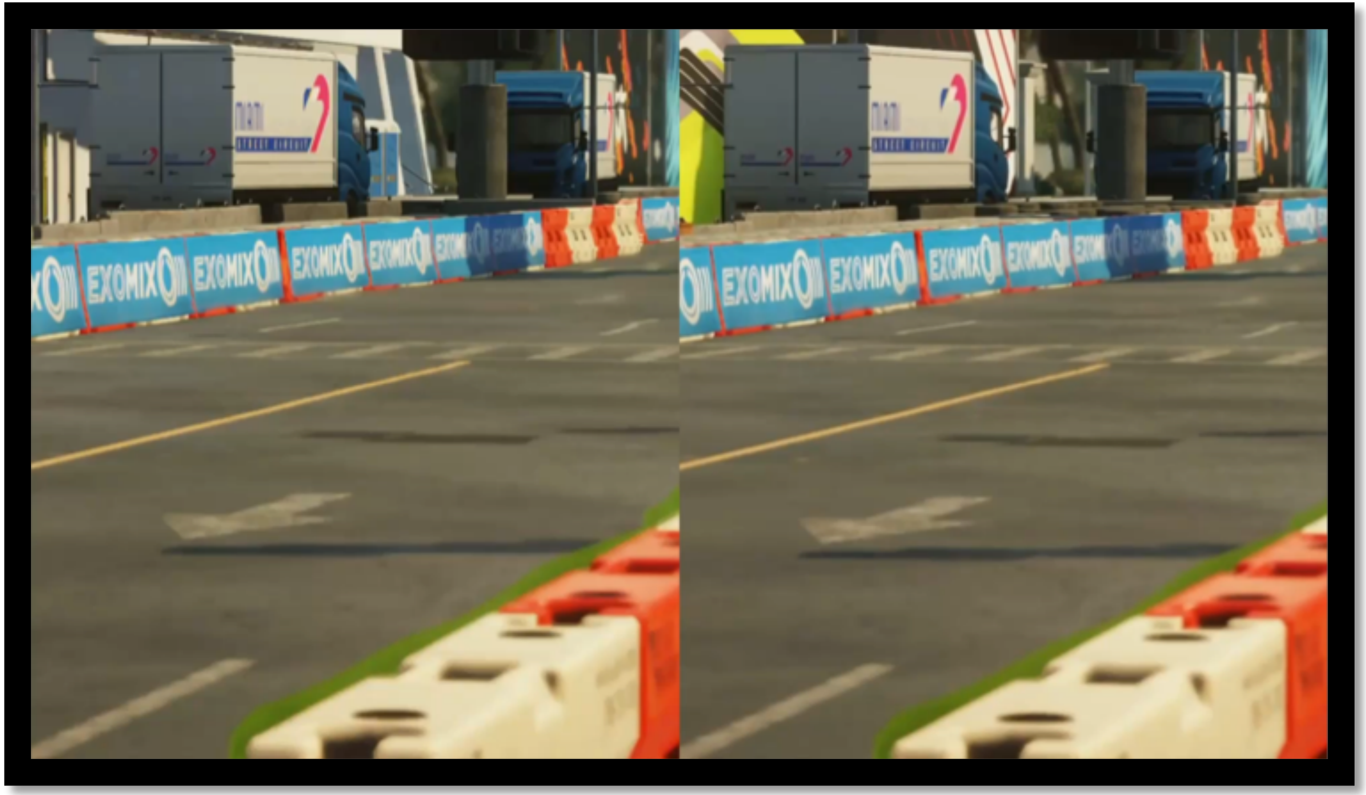
Default setting value was auto selected using CPU detection code. This setting only applies to "Missions" and it's persisted in savegames. When loading a savegame the original value of the settings will be used.

Game Version: 2.10.0
Server Version: 7.5.0



In gaming the Uncanny Valley includes the whole world:
The better it looks, the more important your interaction with it

Miami



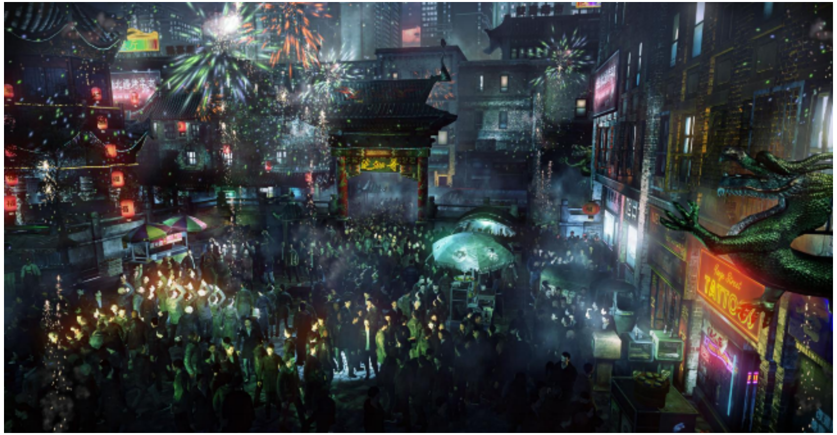
Default Simulation Settings

Best Simulation Settings

CREATING REALISM

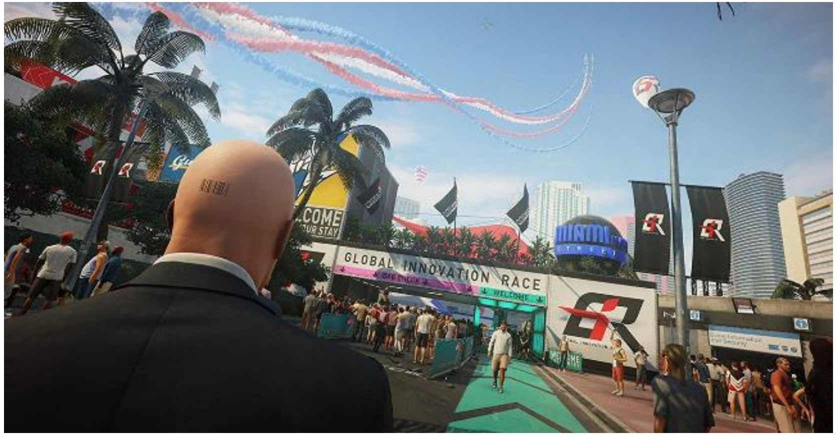
How individual systems were scaled

Evolving role of the Crowd in Hitman



Hitman: Absolution

- 1200 Characters per zone:
  - 500 max onscreen
- AI limited to per Zone

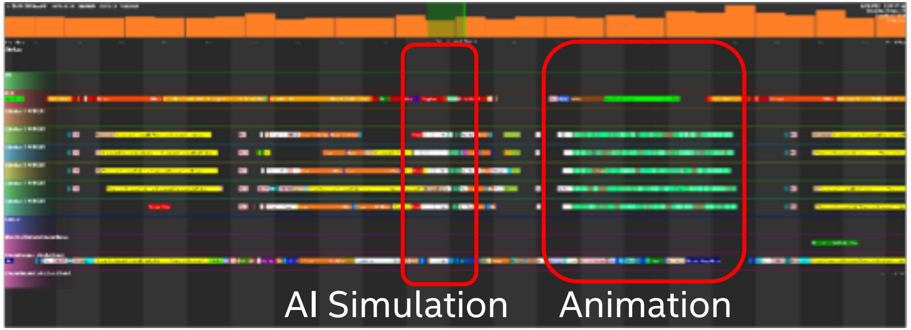


HITMAN Season 2

- 1700 (Default) Characters per level:
  - 700 Max onscreen
- Hitman can hide in the crowd
- AI simulation even when off screen

https://www.gdcvault.com/play/1015315/Crowds-in-Hitman

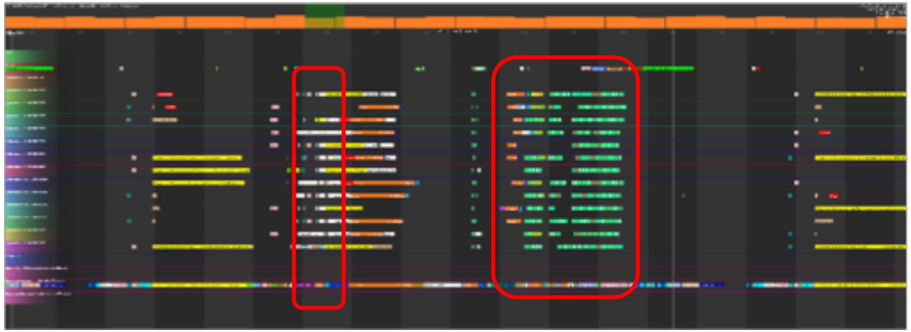
Crowd is CPU intensive



4 Core Micro Profiler capture

- Heavily parallelized
- Mix of systems executed in job system;
  - AI Simulation: Done every frame
  - Animation: Asynchronous:
    - Time sliced over multiple frames

Crowd Scaling



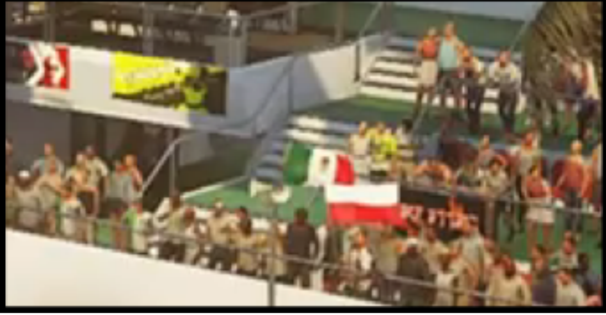
8 Core Micro Profiler capture

- Characters animating further into the scene
- Higher quality animations
- Reduction in time slicing

HESS
SECURITY

Crowd vs NPC

- Possession system, On-demand upgrade crowd agent to full NPC AI
- Allocates small pool of invisible NPCs
- Supports advanced gameplay mechanics
  - Interaction with Agent 47
  - Allows for scripted crowd acts
    - Talk on phone, smoke, sit on bench
    - Couples holding hands
    - Wave flags
- Spawns randomly near player
- Larger pool == more unique animations



NPC Pool Size	
Default build	Pool Size
Default build	30
6 cores	60
8 cores	90

Crowds: Improving Realism

- Per level crowd limits affect level design in none game play areas.

	Default build	6 cores	8 cores
Total amount of Crowd:	1797		
Max visible Crowd	700		
Level Of Detail 1	Distance: 7.0 Count: 25		
Level Of Detail 2	Distance: 15.0 Count: 75		
Level Of Detail 3	Distance: 22.0 Count: 300		

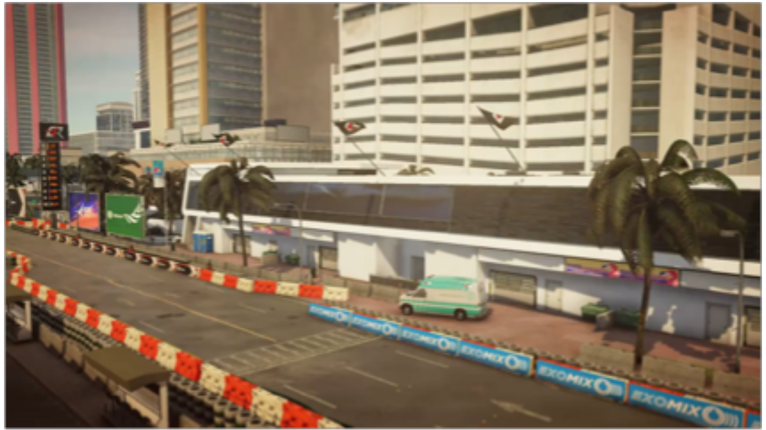
Base

Crowds: Improving Realism

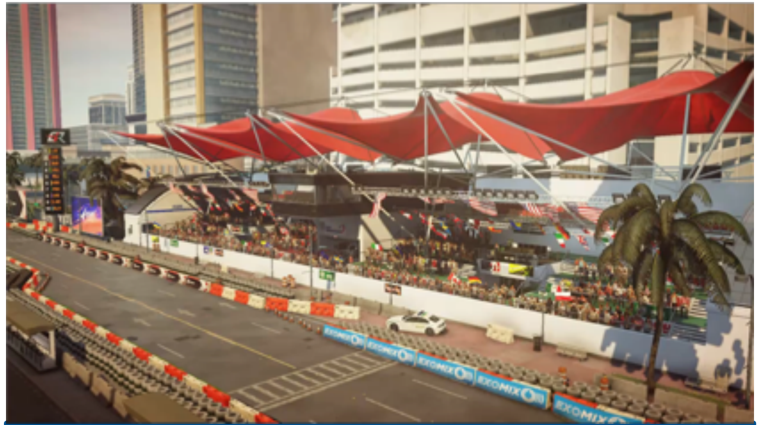
- Per level crowd limits affect level design in none game play areas.

	Default build	6 cores	8 cores
Total amount of Crowd:	1797	2923	2923
Max visible Crowd	700	1000	1400
Level Of Detail 1	Distance: 7.0 Count: 25	Distance: 10.0 Count: 50	Distance: 15.0 Count: 100
Level Of Detail 2	Distance: 15.0 Count: 75	Distance: 20.0 Count: 120	Distance: 25.0 Count: 200
Level Of Detail 3	Distance: 22.0 Count: 300	Distance: 25.0 Count: 400	Distance: 30.0 Count: 500

- Higher limits allow crowd usage in other none game play areas.
- Crowds still interact with the events



Base

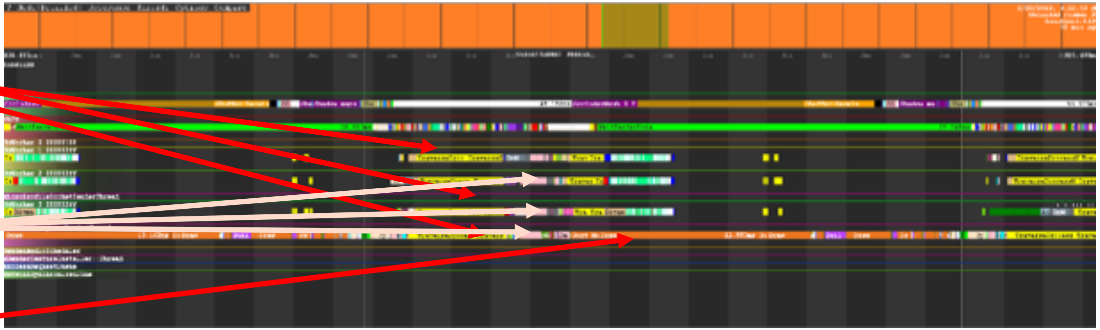


Better/Best

Rendering Multi-threading

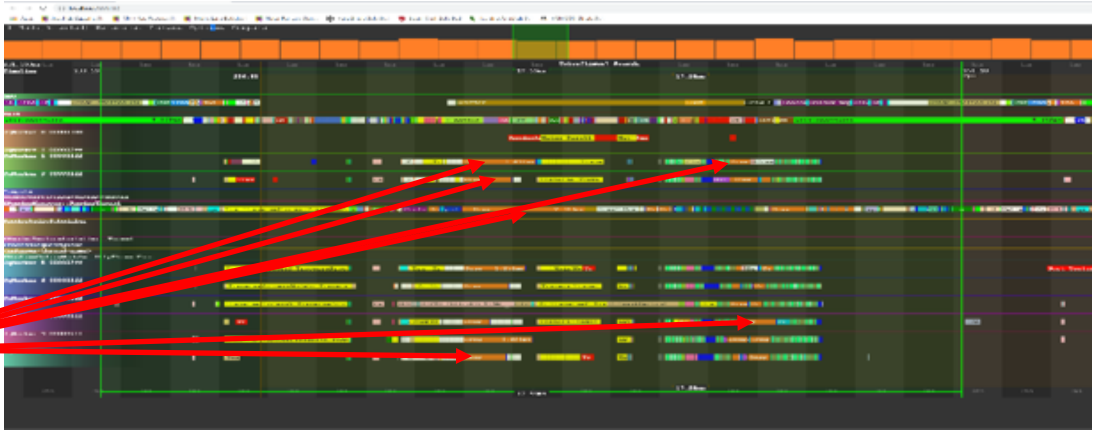
General DX11 Threading

- Scene traversal multi-threaded.
- Software visibility culling runs across up to 5 threads.
- Highest CPU load on Render submission thread.



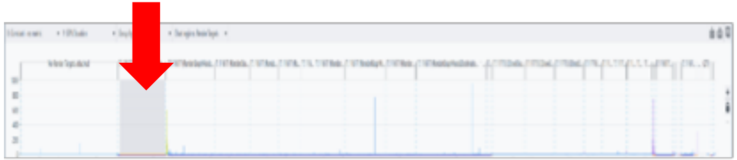
DX12 Improvements

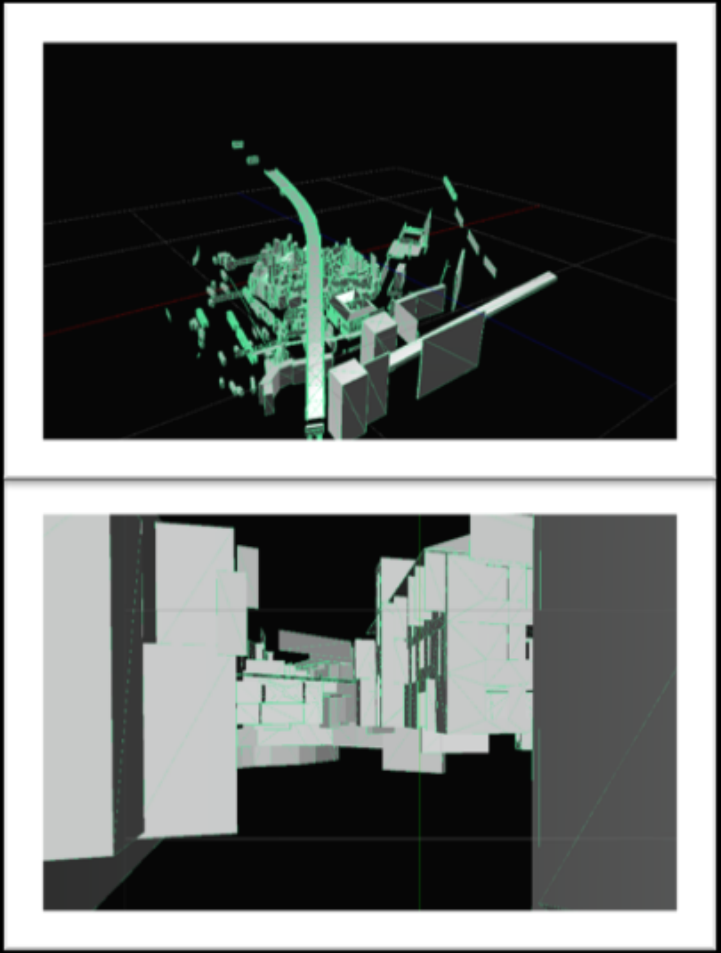
- Commandlist Generation split across worker threads shown as "DRAW"
- 30+ "DRAW"'s Per frame, submission on Render Thread



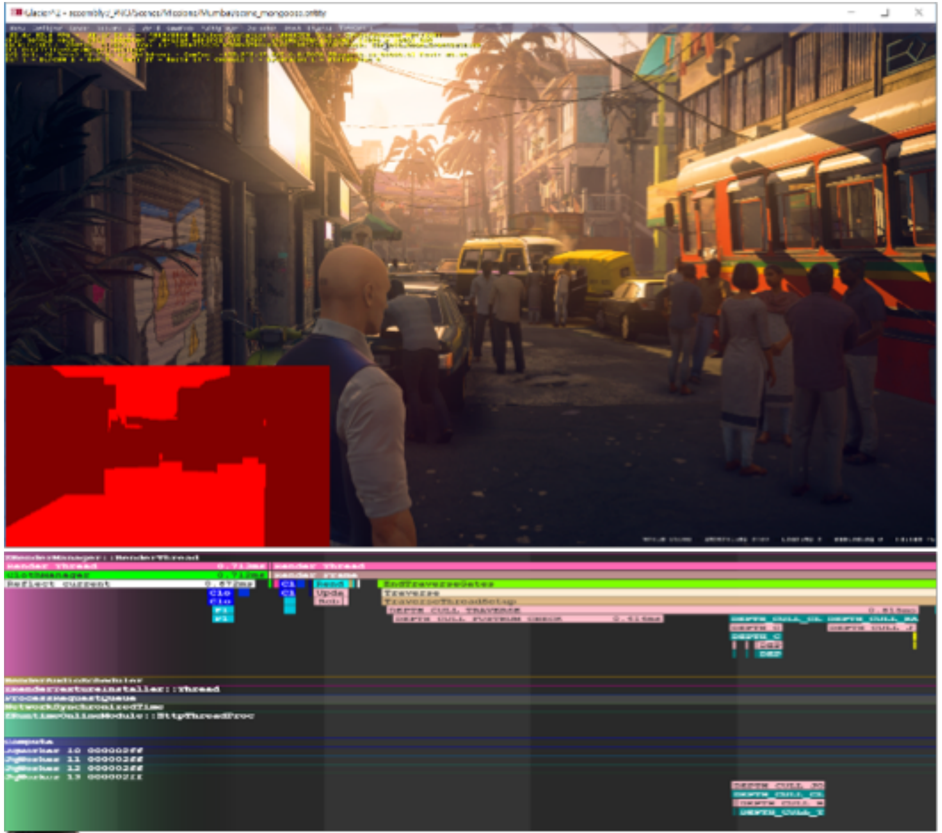
Software Occlusion

- Based on Intel Software Occlusion Culling (https://github.com/GameTechDev/OcclusionCulling)
- Optimisations taken from MSOC https://github.com/GameTechDev/MaskedOcclusionCulling
- Used for Main scene and shadow cascades.
- Used as simplified Depth pre-pass
- 5% of total draw calls in the scene
- 15K out of 3,000K+ vertices





Occlusion Performance

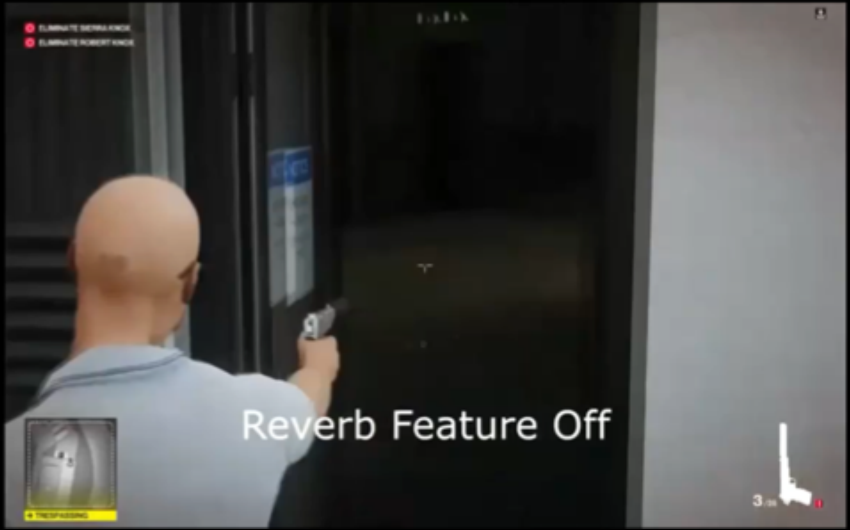


	Mumbai	
	No Depth Culling	Culling
Occluders	210	
Tests	21642	
Draws/Instances	5540/12840	4870/8900
Occlusion cost CPU Time@3Ghz	0ms	0.815ms
VS Invocations	4.54M	3.44M
PS Invocations	20.6M	19.2M
MS/Frame	11.9	10.7

*Core i7-6900X, 32GB Ram, NVIDIA GTX 1080

Audio: Simulation Quality

- Wwise multi-core audio enables audio tasks to execute as part of the job system.
- BASE:
  - Reverb is based the listener position.
  - Crossfade with reverb from adjoining room when approaching doors, windows etc..
- BETTER and BEST
  - Each playing audio emitter uses reverb of the room that it is located in.
  - Increase active reverb busses.
- For BEST
  - Use higher-quality reverb presets:



DEFINING DESTRUCTION

What is it and why do we need it?

First… A Demo!

Why destroy things?

- So much fun!
- Environment interaction improves player immersion
- More interesting opportunities for ‘accidents’
- More iconic destructive moments

What we need

- Very specific destruction patterns
- Animated destructible objects
- Scalable destruction
- Support existing assets

Dynamic vs Pre-Defined

Dynamic

- Simple cases are scalable (glass etc.)
- Need lots of tooling for complex cases
- Harder to scale at runtime
- Lots of edge cases to deal with when dynamically slicing assets

Pre-Defined

- Not quite as realistic
- Easier to scale at runtime
- No need to worry about complex edge cases when slicing assets
- Content creators have full power over how objects break

SIMULATING DESTRUCTION

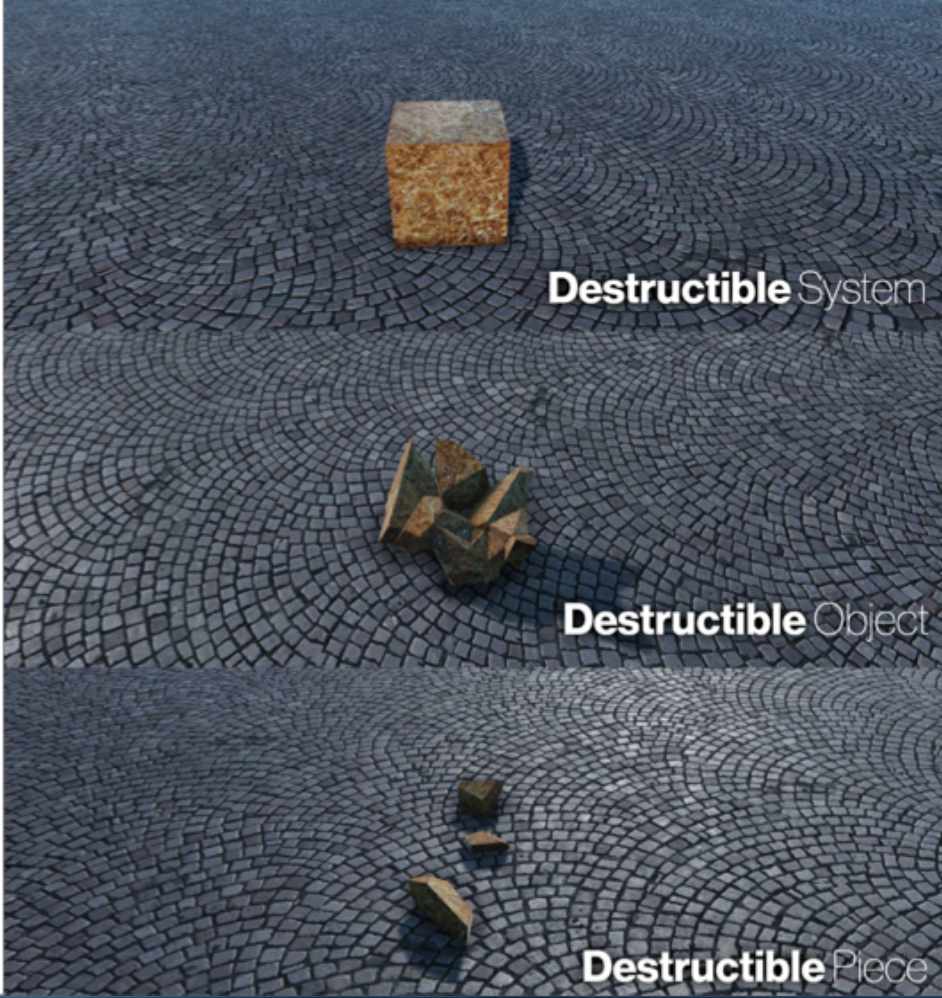
The destruction runtime

Types of destruction

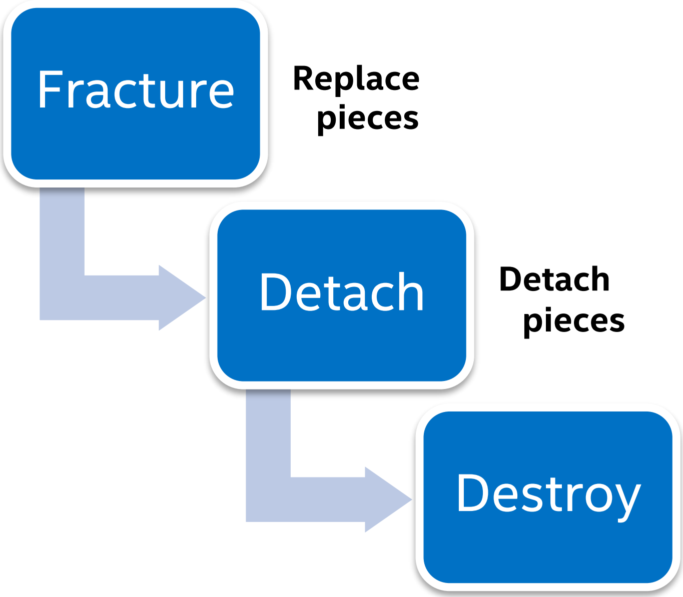
- Gameplay Effecting
  - Affects gameplay in a big way
  - Usually affects AI
  - May need to persist in the world between saves
- Cosmetic
  - Doesn’t effect gameplay in a big way
  - Has no major affect on AI
  - Does not usually need to be saved

Some Definitions

- **Destruction System**
  - Manages destruction of a single asset
- **Destructible Object**
  - A simulated physics object owned by a destruction system
- **Destructible Piece**
  - A single piece of a destructible object
  - Objects can be made of more than one piece



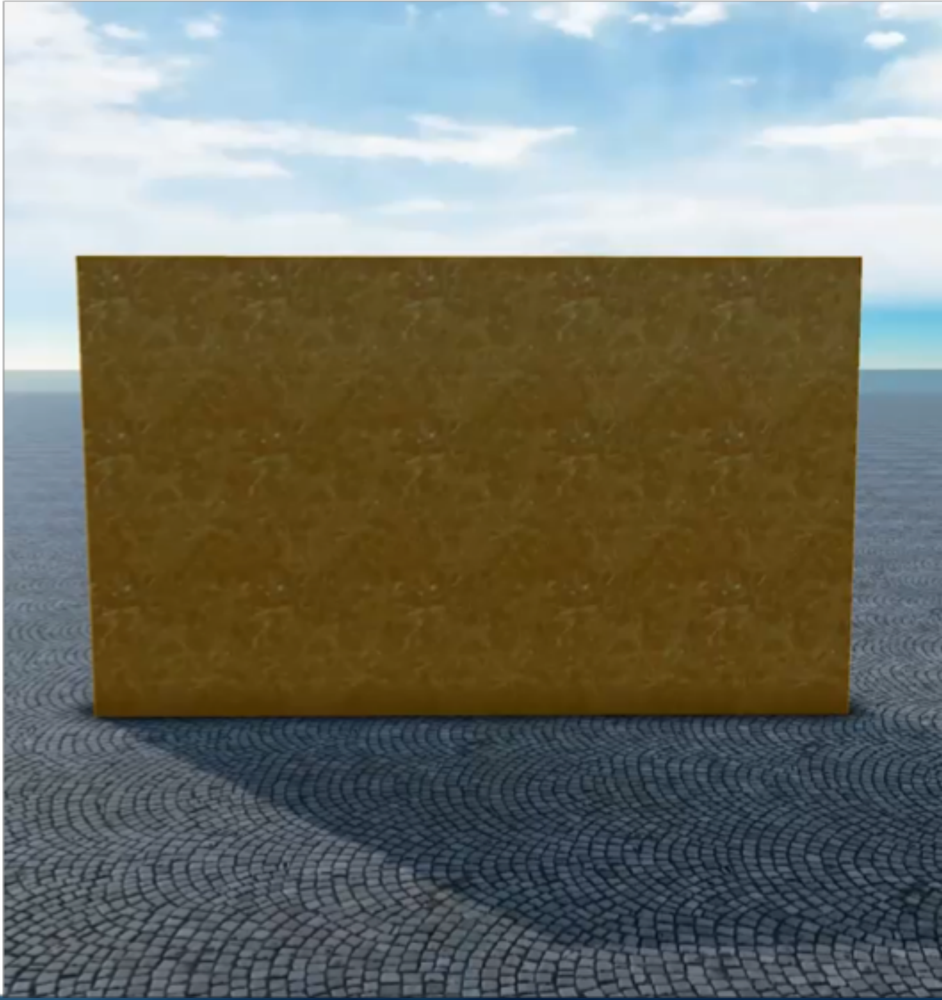
Destruction Phases



Replace pieces

Detach pieces

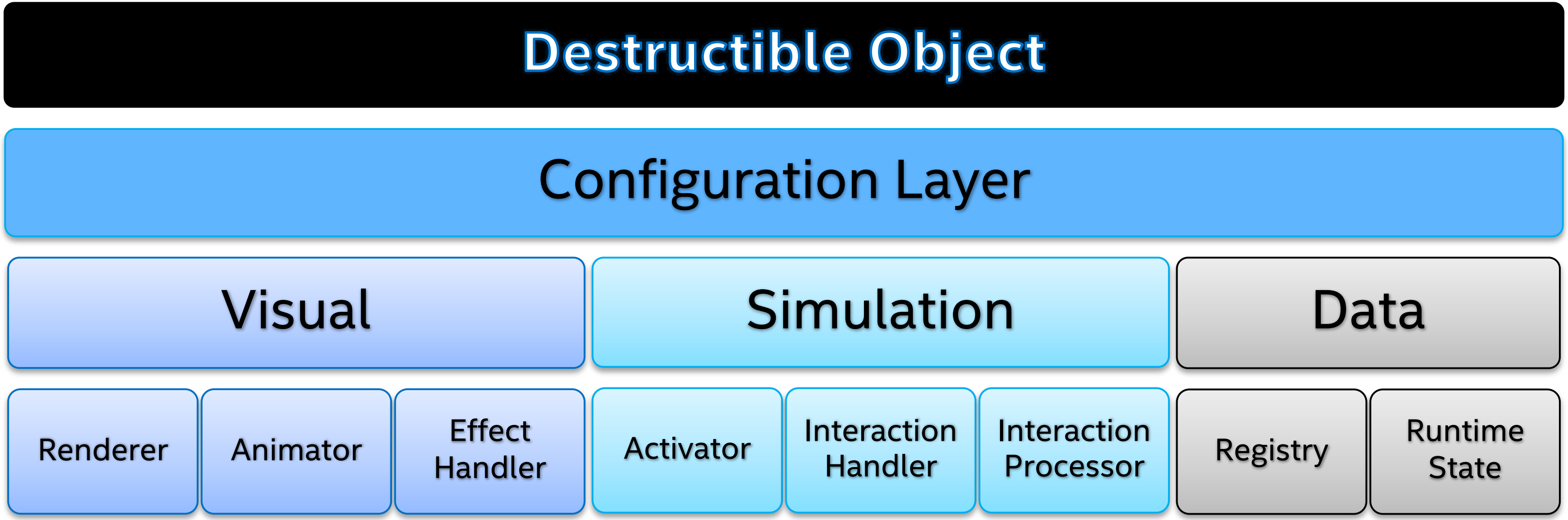
Remove pieces



Destruction Runtime

- Two classes of system
  - Instanced sub systems
  - Global systems
- Runs alongside our physics engine as a ‘plugin’
- Highly configurable

Instance Specific Systems



Destructible Object

Configuration Layer

Visual

Simulation

Data

Renderer

Animator

Effect Handler

Activator

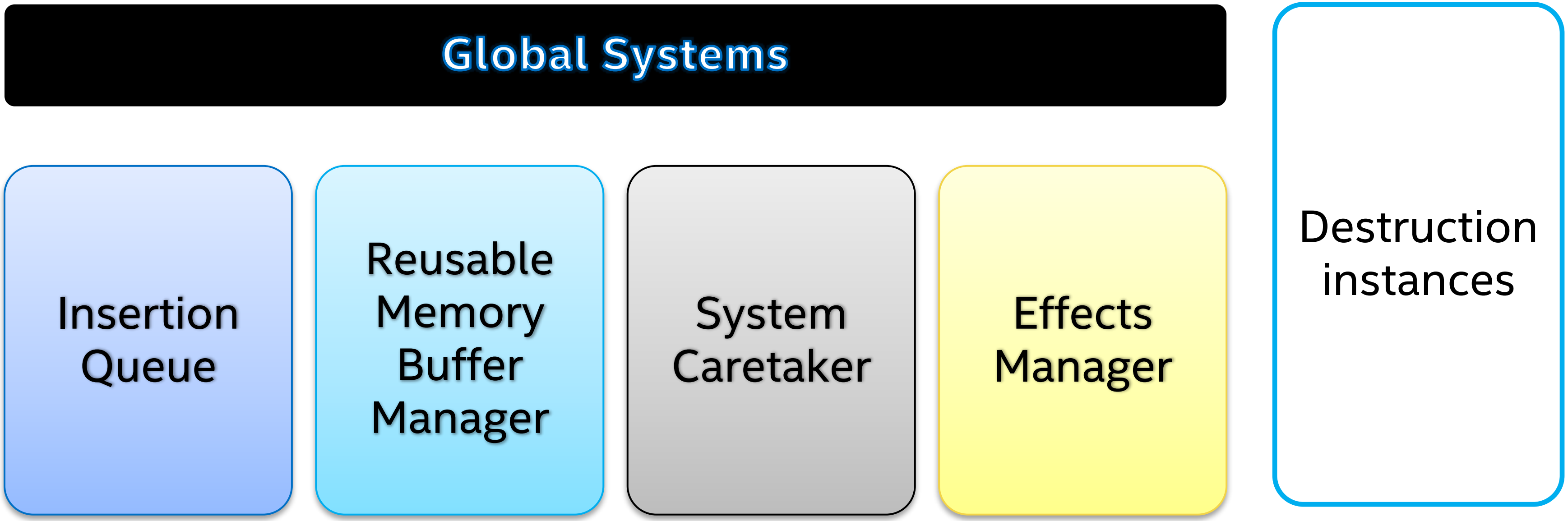
Interaction Handler

Interaction Processor

Registry

Runtime State

Global Systems



Global Systems

Insertion Queue

Reusable Memory Buffer Manager

System Caretaker

Effects Manager

Destruction instances

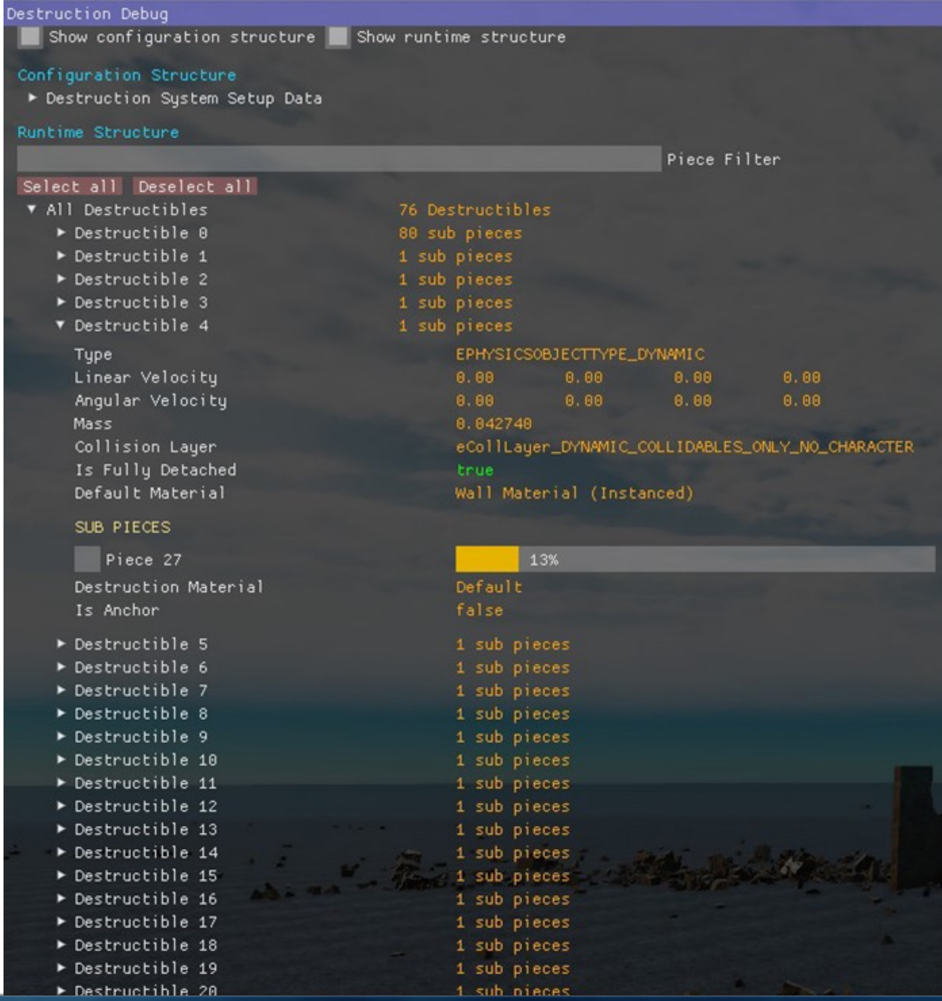
Instance Runtime

Storing the Data

Storing the Data

The Registry

- Stores references to all active physics objects for this destruction system
- Maintains a link between the physics object and its configuration data
  - Piece Connections
  - Materials
  - Strength

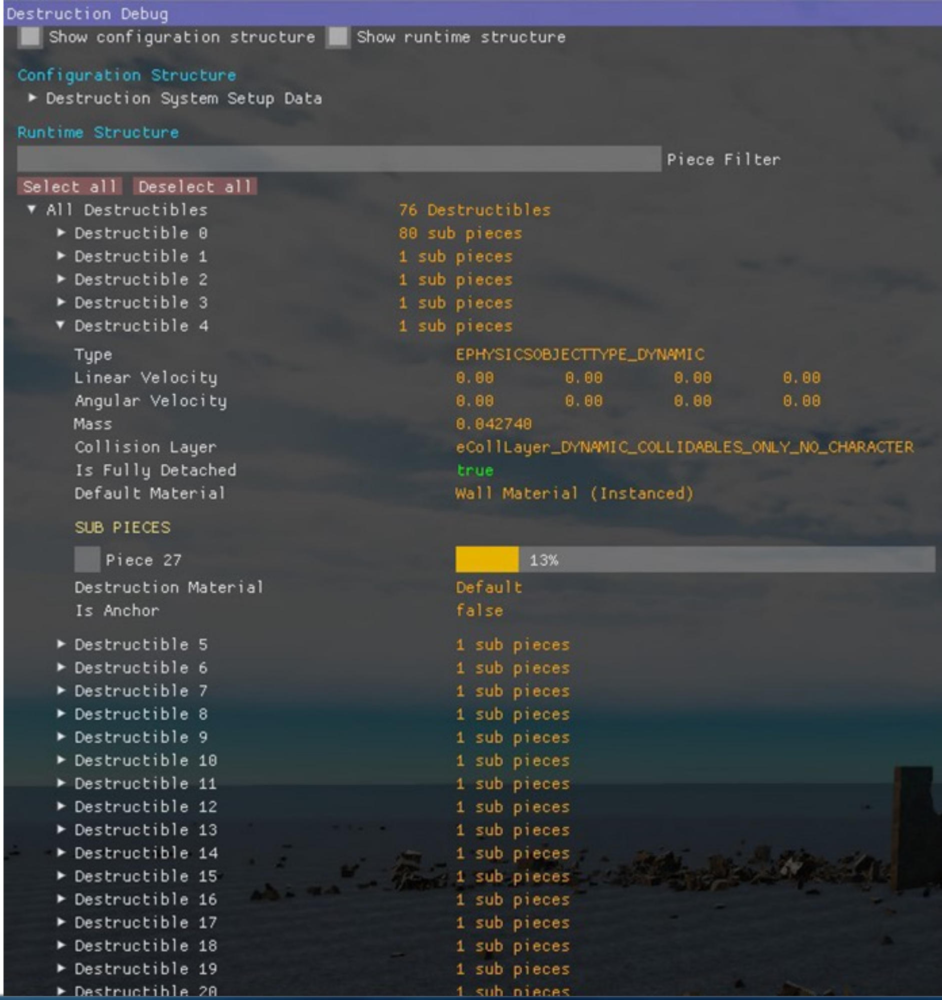


```
Destruction Debug
□ Show configuration structure  □ Show runtime structure

Configuration Structure
 ► Destruction System Setup Data

Runtime Structure
                                                  Piece Filter
Select all  Deselect all
 ▼ All Destructibles                  76 Destructibles
   ► Destructible 0                   80 sub pieces
   ► Destructible 1                   1 sub pieces
   ► Destructible 2                   1 sub pieces
   ► Destructible 3                   1 sub pieces
   ▼ Destructible 4                   1 sub pieces

     Type                                EPHYSICSOBJECTTYPE_DYNAMIC
     Linear Velocity                     0.00      0.00      0.00      0.00
     Angular Velocity                    0.00      0.00      0.00      0.00
     Mass                                0.042740
     Collision Layer                     eCollLayer_DYNAMIC_COLLIDABLES_ONLY_NO_CHARACTER
     Is Fully Detached                   true
     Default Material                    Wall Material (Instanced)

     SUB PIECES

     □ Piece 27                          13%
     Destruction Material                Default
     Is Anchor                           false

   ► Destructible 5                      1 sub pieces
   ► Destructible 6                      1 sub pieces
   ► Destructible 7                      1 sub pieces
   ► Destructible 8                      1 sub pieces
   ► Destructible 9                      1 sub pieces
   ► Destructible 10                     1 sub pieces
   ► Destructible 11                     1 sub pieces
   ► Destructible 12                     1 sub pieces
   ► Destructible 13                     1 sub pieces
   ► Destructible 14                     1 sub pieces
   ► Destructible 15                     1 sub pieces
   ► Destructible 16                     1 sub pieces
   ► Destructible 17                     1 sub pieces
   ► Destructible 18                     1 sub pieces
   ► Destructible 19                     1 sub pieces
   ► Destructible 20                     1 sub pieces
```

Storing the Data

The Runtime State

- Stores all changes to the static configuration data
- This includes
  - Damage applied to each piece
  - Active connections for each piece
- High level system state change flags
  - Has been fractured, detached or destroyed
  - Contains anchors etc.



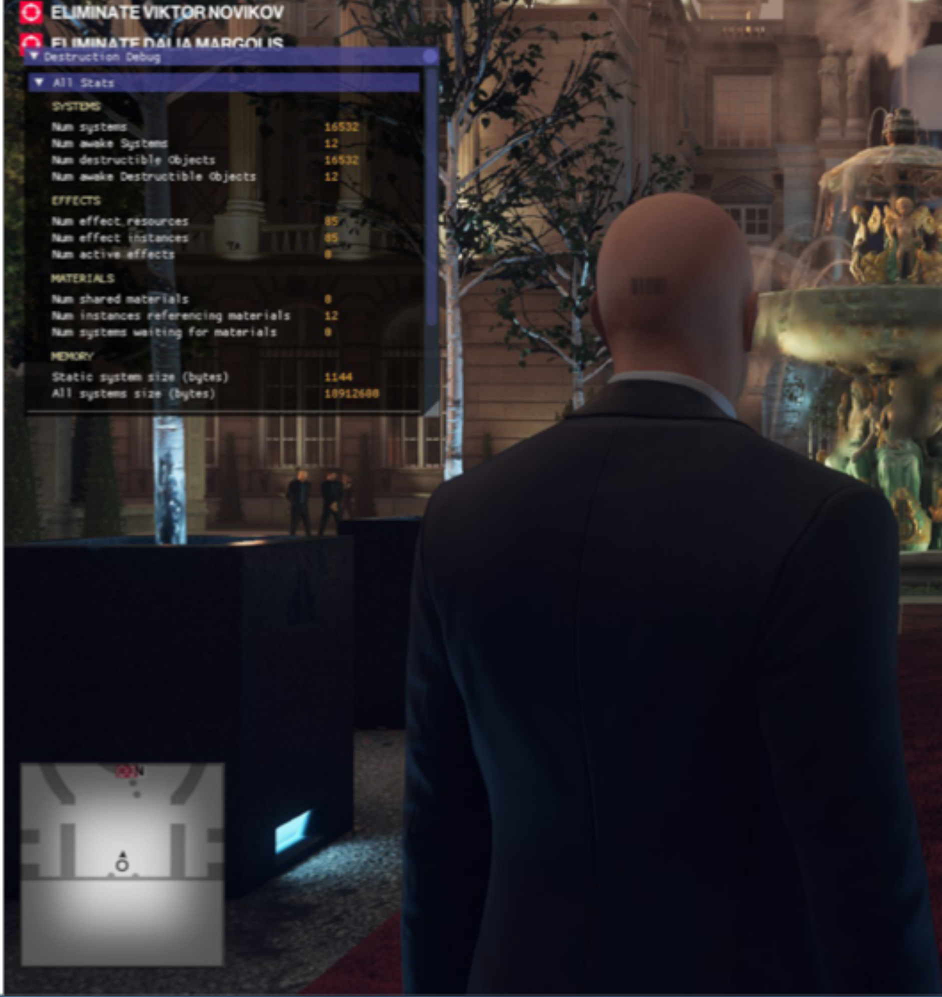
```
Destruction Debug
 □ Show configuration structure  □ Show runtime structure

 Configuration Structure
  ▶ Destruction System Setup Data

 Runtime Structure
                                                       Piece Filter
 Select all  Deselect all
  ▼ All Destructibles                 76 Destructibles
     ▶ Destructible 0                 80 sub pieces
     ▶ Destructible 1                 1 sub pieces
     ▶ Destructible 2                 1 sub pieces
     ▶ Destructible 3                 1 sub pieces
     ▼ Destructible 4                 1 sub pieces

       Type                             EPHYSICSOBJECTTYPE_DYNAMIC
       Linear Velocity                  0.00      0.00      0.00      0.00
       Angular Velocity                 0.00      0.00      0.00      0.00
       Mass                             0.042740
       Collision Layer                  eCollLayer_DYNAMIC_COLLIDABLES_ONLY_NO_CHARACTER
       Is Fully Detached                true
       Default Material                 Wall Material (Instanced)

       SUB PIECES

       □ Piece 27                       13%
       Destruction Material             Default
       Is Anchor                        false

     ▶ Destructible 5                   1 sub pieces
     ▶ Destructible 6                   1 sub pieces
     ▶ Destructible 7                   1 sub pieces
     ▶ Destructible 8                   1 sub pieces
     ▶ Destructible 9                   1 sub pieces
     ▶ Destructible 10                  1 sub pieces
     ▶ Destructible 11                  1 sub pieces
     ▶ Destructible 12                  1 sub pieces
     ▶ Destructible 13                  1 sub pieces
     ▶ Destructible 14                  1 sub pieces
     ▶ Destructible 15                  1 sub pieces
     ▶ Destructible 16                  1 sub pieces
     ▶ Destructible 17                  1 sub pieces
     ▶ Destructible 18                  1 sub pieces
     ▶ Destructible 19                  1 sub pieces
     ▶ Destructible 20                  1 sub pieces
```

Instance Runtime

Controlling Simulation

The Activator

- Controls when a system should be marked as ‘active’ or ‘inactive’
- Causes the animator to run and synchronise positions
- Means that interactions will be processed
- Allows gameplay systems to hook into when a destructible system is being interacted with
- Provides feedback to the debug systems for displaying stats



```
▼ Destruction Debug
  ▼ All Stats
    SYSTEMS
    Num systems                                16532
    Num awake Systems                          12
    Num destructible Objects                   16532
    Num awake Destructible Objects             12
    EFFECTS
    Num effect resources                       85
    Num effect instances                       85
    Num active effects                         0
    MATERIALS
    Num shared materials                       0
    Num instances referencing materials        12
    Num systems waiting for materials          0
    MEMORY
    Static system size (bytes)                 1144
    All systems size (bytes)                   18912600
```

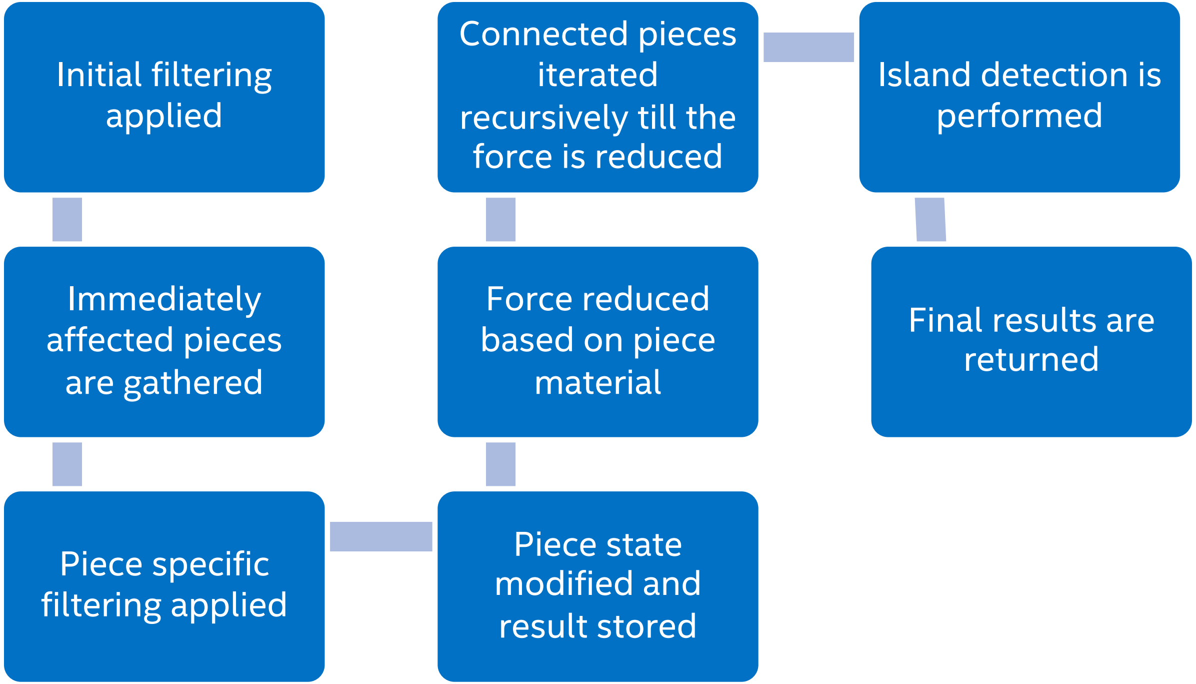
The Interaction Handler

- Hooks into external game systems to listen for destructible events
  - Shots
  - Explosions
  - Collisions
  - Out of world
- Unifies external events into a common destruction force structure
- Passes destruction forces to the interaction processor

The Interaction Processor

- Stores a queue of pending interactions and processes them each frame
- Propagates damage through the system for each interaction using a damage propagator
- Applies world state changes based on the propagation results (fracturing, detaching and destroying)
- Passes all system results out for a frame to the effect processor

The Damage Propagator



Initial filtering applied

Immediately affected pieces are gathered

Piece specific filtering applied

Piece state modified and result stored

Force reduced based on piece material

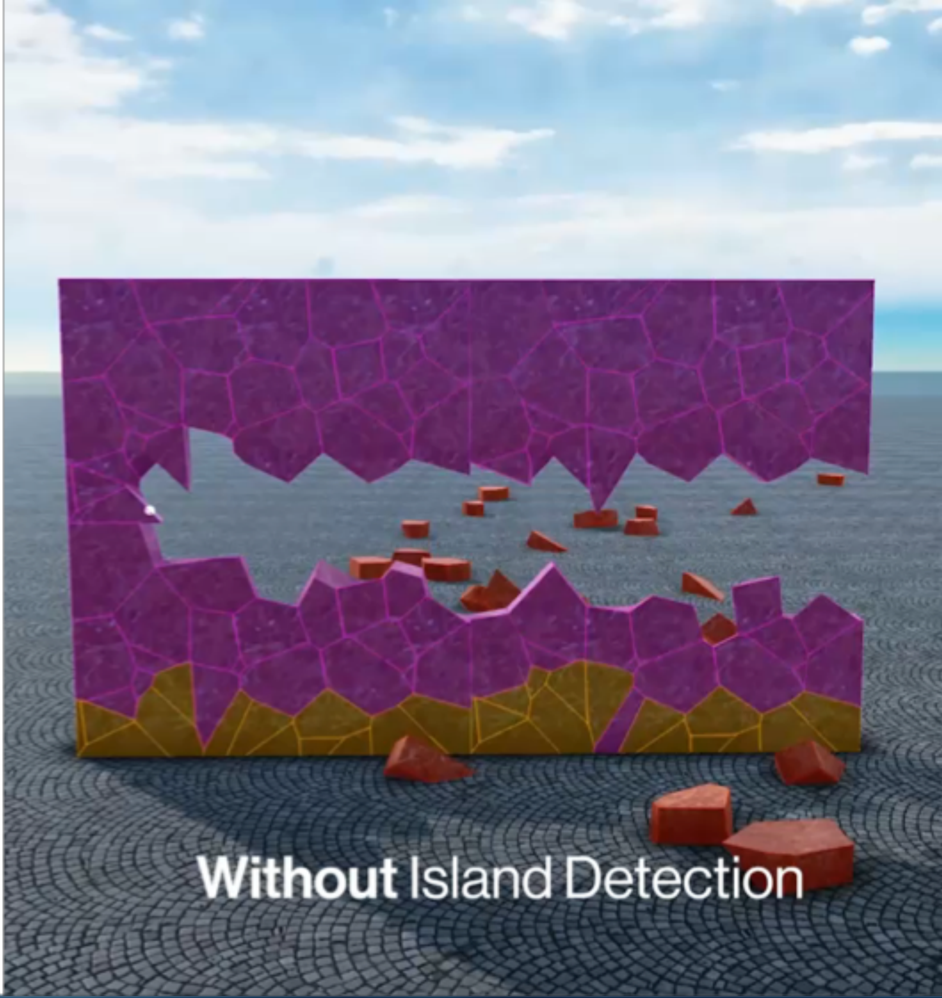
Connected pieces iterated recursively till the force is reduced

Island detection is performed

Final results are returned

Island Detection

- When connecting pieces are detached, other pieces need to fall
- Island detection is run when a piece or set of pieces is detached from a system
- Any ‘islands’ are then detached from the system



Island Detection

- Get island candidates
- For each island candidate
  - Assign a group id to the candidate and mark as visited
  - Assign the same group id to each unvisited connection and mark as visited
  - Do the same for each connection
- Find all unique group ids and detach each of the smallest groups

Global Systems

Modifying The World

The Insertion Queue

- Controls asynchronous creation and insertion of physics objects in to the scene
- Allows throttling and discarding of requests under heavy loads
- Provides callbacks to the destruction system instances when their requests have been dealt with

The Effects Manager

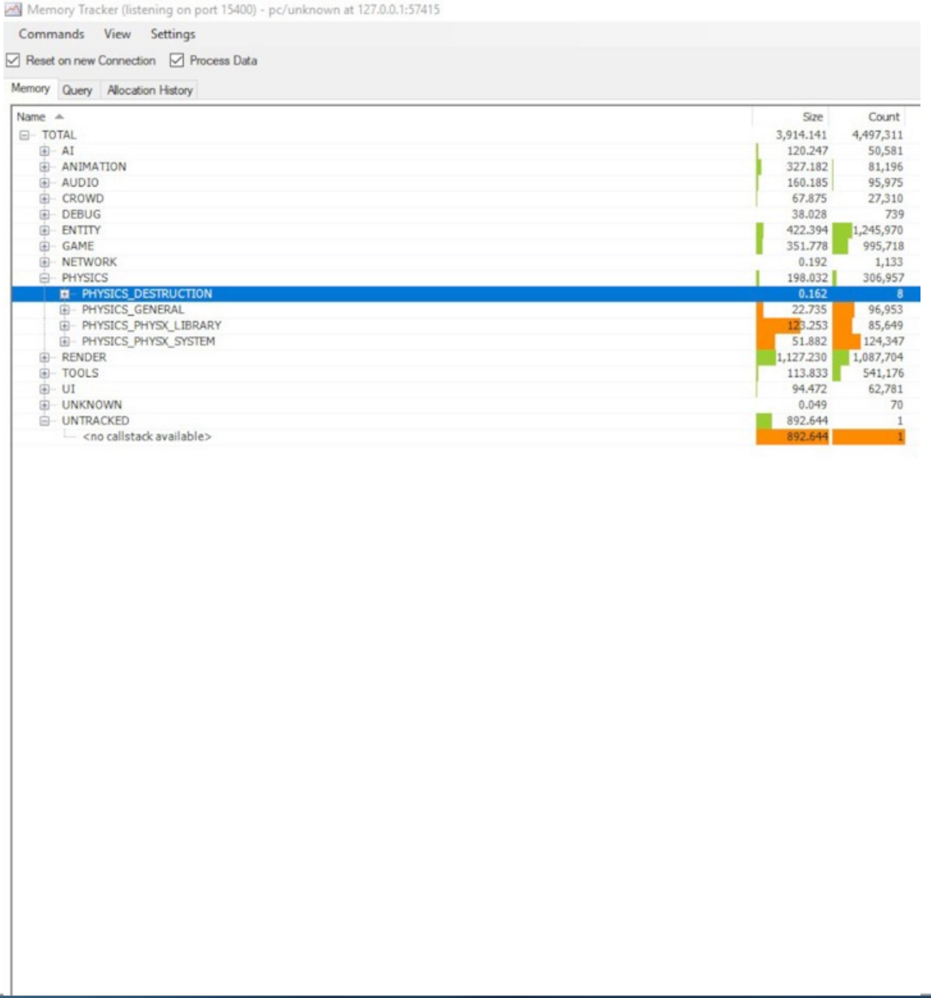
- Responds to requests for showing an effect from a system effect handler
- Responds to four different events
  - Collided
  - Fractured
  - Detached
  - Destroyed
- Pools effect resource instances to allow them to be reused
- Controls the synchronisation of active effect positions

Global Systems

Managing Resources

The Reusable Memory Buffer Manager

- Provides thread safe access to reusable memory buffers
- Massively reduces the overhead of runtime memory allocations during destruction processing



Memory Tracker (listening on port 15400) - pc/unknown at 127.0.0.1:57415
Commands
View
Settings
Memory
Query
Allocation History

☑ Reset on new Connection ☑ Process Data

Name	Size	Count
TOTAL	3,914.141	4,497,311
AI	120.247	50,581
ANIMATION	327.182	81,196
AUDIO	160.185	95,975
CROWD	67.875	27,310
DEBUG	38.028	739
ENTITY	422.394	1,245,970
GAME	351.778	995,718
NETWORK	0.192	1,133
PHYSICS	198.032	306,957
PHYSICS_DESTRUCTION	0.162	8
PHYSICS_GENERAL	22.735	96,953
PHYSICS_PHYSX_LIBRARY	123.253	85,649
PHYSICS_PHYSX_SYSTEM	51.882	124,347
RENDER	1,127.230	1,087,704
TOOLS	113.833	541,176
UI	94.472	62,781
UNKNOWN	0.049	70
UNTRACKED	892.644	1
<no callstack available>	892.644	1

The System Caretaker

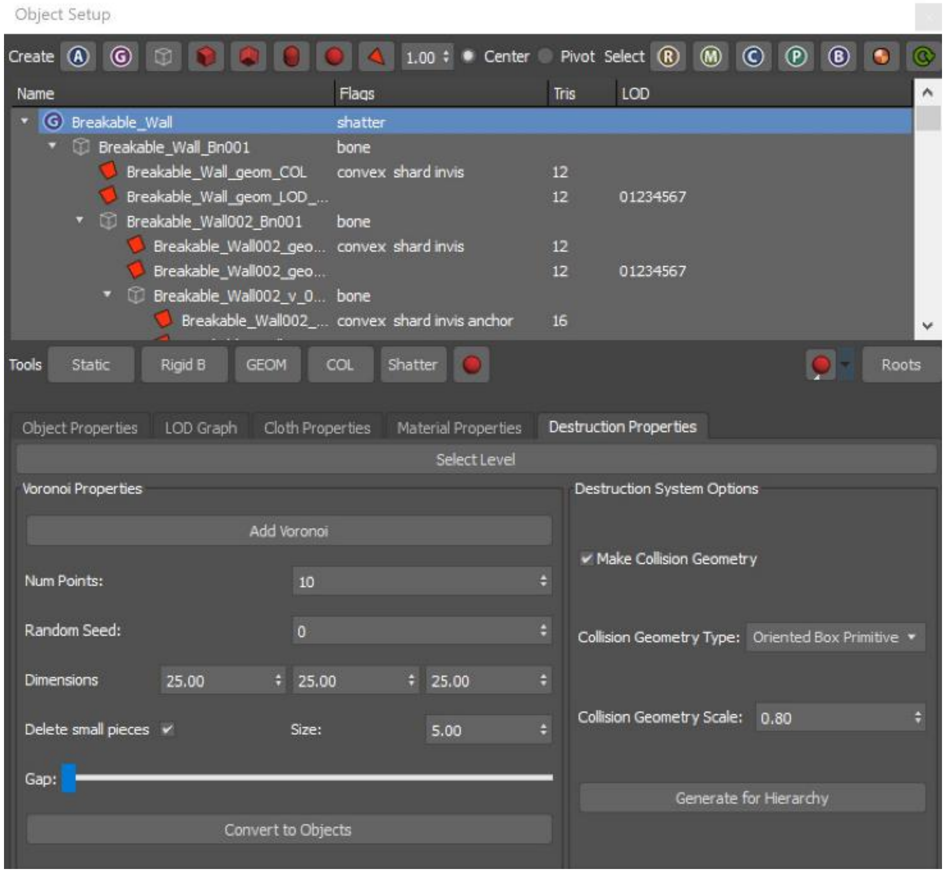
- Periodically performs the following tasks
  - Removes objects from the scene that meet the required criteria
  - Deallocates runtime memory that objects no longer need
  - Reduces particle effect pools based on a heuristic that estimates current particle effect requirements in the level

DESIGNING DESTRUCTION

How to make something destructible?

Content Authoring

- Content is created in 3DS Max
- Voronoi partitions are used to split assets
- Physics shapes are automatically generated from the geometry
- The object hierarchy is used to define the fracture levels



Name	Flags	Tris	LOD
Breakable_Wall	shatter		
Breakable_Wall_Bn001	bone		
Breakable_Wall_geom_COL	convex shard invis	12	
Breakable_Wall_geom_LOD_...		12	01234567
Breakable_Wall002_Bn001	bone		
Breakable_Wall002_geo...	convex shard invis	12	
Breakable_Wall002_geo...		12	01234567
Breakable_Wall002_v_0...	bone		
Breakable_Wall002_...	convex shard invis anchor	16	

Voronoi Properties

Num Points: 10

Random Seed: 0

Dimensions 25.00 25.00 25.00

Delete small pieces ☑ Size: 5.00

Gap:

Destruction System Options

☑ Make Collision Geometry

Collision Geometry Type: Oriented Box Primitive

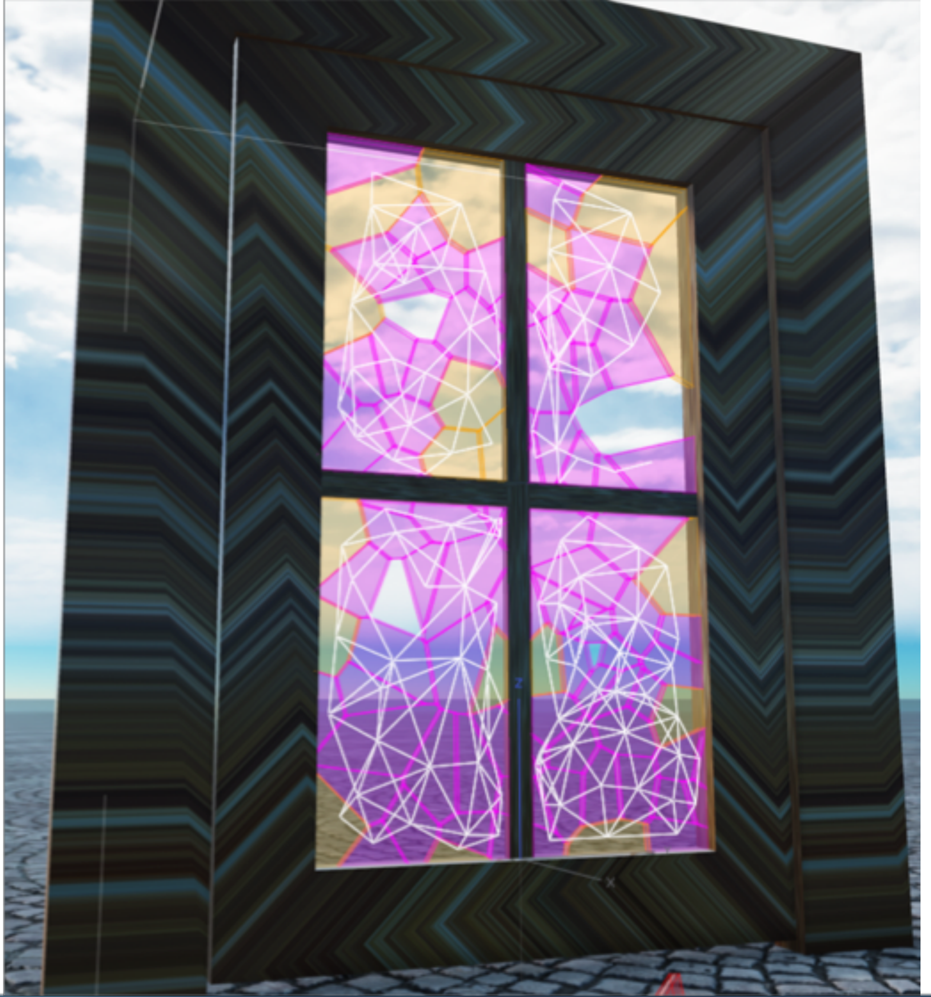
Collision Geometry Scale: 0.80

Destruction materials

- A material controls the response to destructible forces
- Response is configured in two parts
  - **Strength**
    - How resistant to damage is an object
  - **Shock absorption**
    - How much force is absorbed by an object
- Different pieces can have different materials

Connections

- Connections are used to connect two or more pieces together
- Connections are automatically generated between touching geometry
- Attributes can be used to disable connection generation



Anchor Pieces

- Pieces connected directly or indirectly will match the transform of the main system when it is animated
- Used to control kinematic objects and objects with constraints

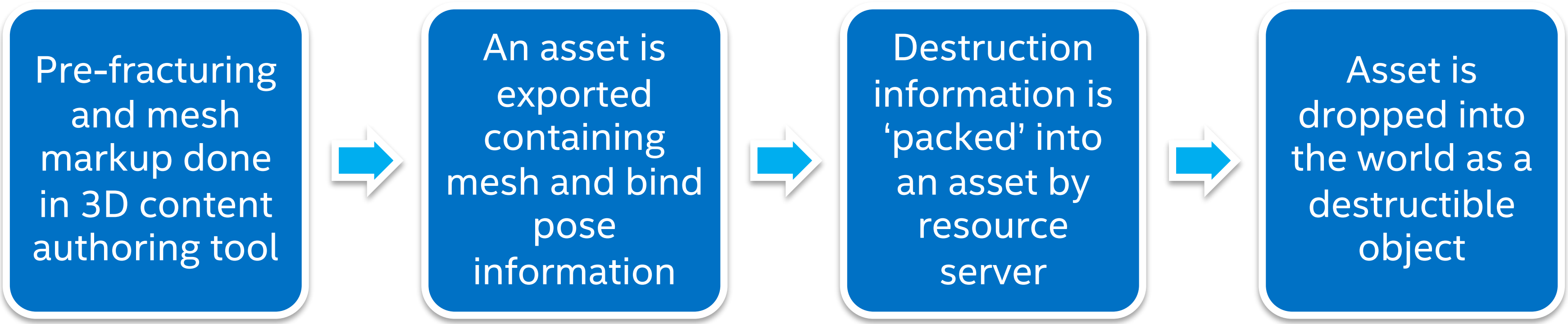
Without Anchors

Piece Attributes

Attributes that control behaviour

- **Remain** : Pieces with this attribute will not be removed from the world or destroyed
- **Orphaned** : Pieces with this attribute will never allow connections to be generated to other pieces

The Destruction Content Pipeline



Pre-fracturing and mesh markup done in 3D content authoring tool

An asset is exported containing mesh and bind pose information

Destruction information is ‘packed’ into an asset by resource server

Asset is dropped into the world as a destructible object

Conclusion

- Realism is about more than pixel quality
- Scale simulation quality as CPU power increases
  - Consider gameplay implications
  - Expand Sandbox without effecting core gameplay mechanics
  - Add more diversity to scene through physics and animation LOD’s
- Environment interaction improves player immersion
  - More interesting opportunities for ‘accidents’
  - Consider gameplay implications again ☺
- Utilize DX12 and advanced culling system to support ever expanding content sets
- Audio adds to immersion, look at multi-threaded solutions

Legal Notices and Disclaimers

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at [intel.com].

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.   For more complete information visit www.intel.com/benchmarks.

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Intel, Core and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others

QUESTIONS?

intel®