Project Hollow Point

Team Number: May1741

Advisor: Prof. Moni Mina

Team: Travis Evers

What is Project Hollow Point

Project Hollow Point is an adaptor to give Unity 3D access to all of the Bullet Physics Library. Giving Unity 3D the same physics calculation power of the best open source physics engine available.

Overview

- What is Unity 3D and Bullet Physics
- What problem is this project trying to solve
- The deliverables of this project
- Challenges
- What is done and what is left to do

Unity 3D

- Unity 3D is a cross platform game development engine
  - Currently supports 21 different platforms
- Uses C# and Javascript
- Use a modified version of NVidia’s PhsX physics engine
  - Designed for portability
  - Lighter weight
  - Uses multi threaded commands to CPU instead of the GPU

Bullet Physics

- Bullet Physics is an open source Physics engine
- Written C++
- Supports rigid body physics, soft body physics, inverse kinematics, and collision detection
- Used by major developers in the gaming industry such as Rockstar Games
- Used by the scientific community for research

Why This Project Matters

- More scientific research and development is being done in Unity 3D
- With the increasing demand for VR more realistic Physics are necessary to maintain immersion
- There are no available plugins for Unity 3D that successfully integrate a better physics engine into Unity 3D without timing conflicts

Functional Requirements

- Integration of Bullet Physics Operations
  - Rigid body
  - Soft Body
  - Inverse Kinematic
- Integrated Unity editor Interface
- Ability to create 3D primitives and load 3D models
- Handle collision detection on the GPU

Non Functional Requirements

- Maintain Unity 3D’s portability
- Increase Unity 3D’s ability to process physics based objects by a minimum of 50%
- Have a user interface that feels familiar to Unity 3D developers
- No timing conflicts between Unity 3D and Bullet Physics

Physics Types

- Rigid Body Physics integration
  - Transform movement
  - Transform Rotations
  - Collision Detection
- Soft Body Physics integration
  - Fluid simulation
  - Cloth simulation
  - Voxel based deformable objects
- Inverse Kinematic integration
  - Joint constraint movement

User Interface

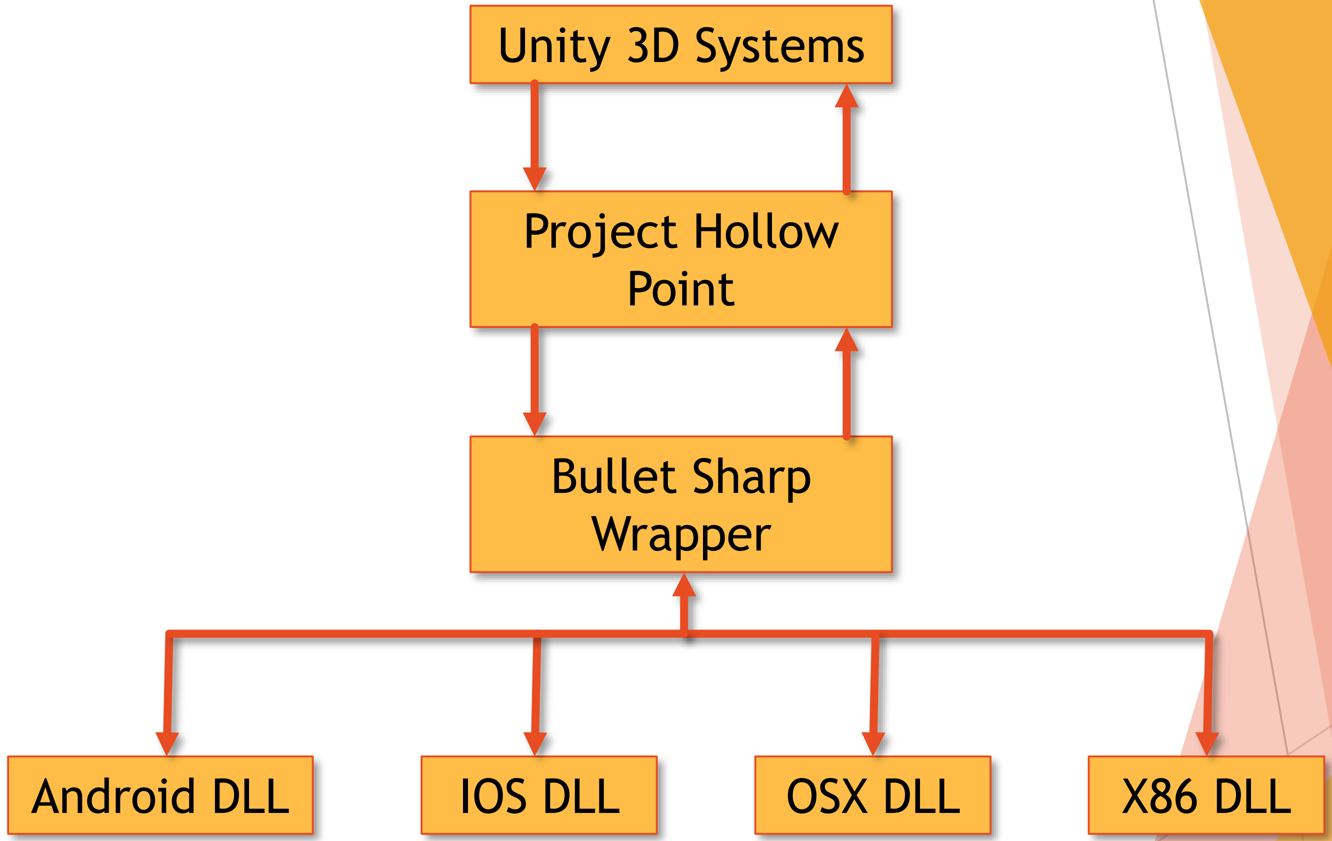
- Customized Unity editor extensions to make it easier to create and modify objects
  - Editor menus
  - Adding a Bullet Physics drop down menu for object creation
  - In scene Editing widgets
- The ability to create and load in 3D models into the scene with optimized mesh structures, UV maps, and normal mappings
- Visualization of colliders in scene

Collision Detection Overview

- Multiple Broad phase Collection methods
  - Dynamic Axis Aligned Bounding Box
  - Axis Sweeping
  - Simple
- Only one type of Narrow Phase collision detection

Handling Portability

- Using the Bullet Sharp DLLs
- Using a modified version of Bullet Sharp's wrapper
- Active DLL selectable within Build settings

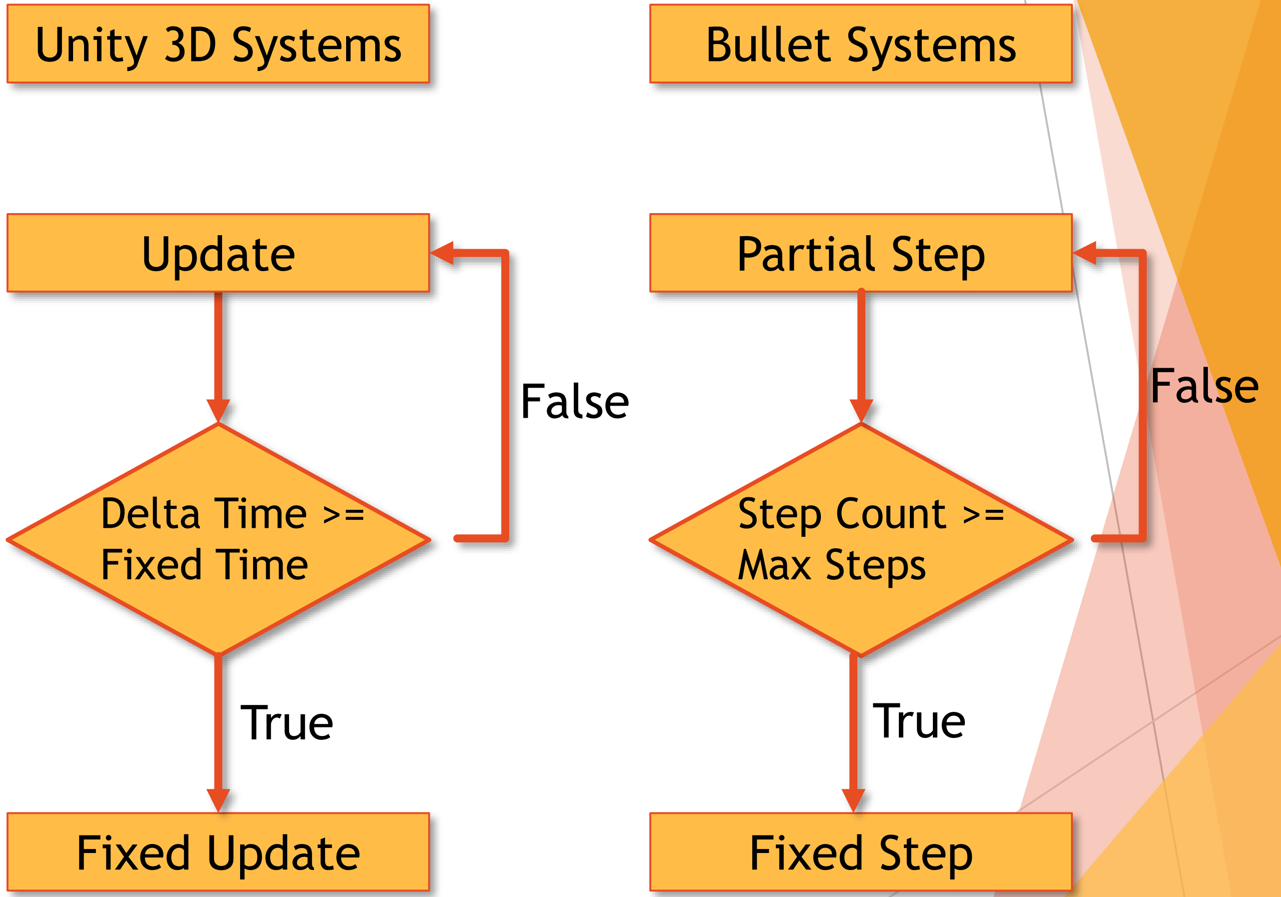


Challenges

- Scale and scope of this project is huge
- Keeping two update cycles from going out of synch
- Automated testing is nearly impossible to do since the majority of concerns are timing related

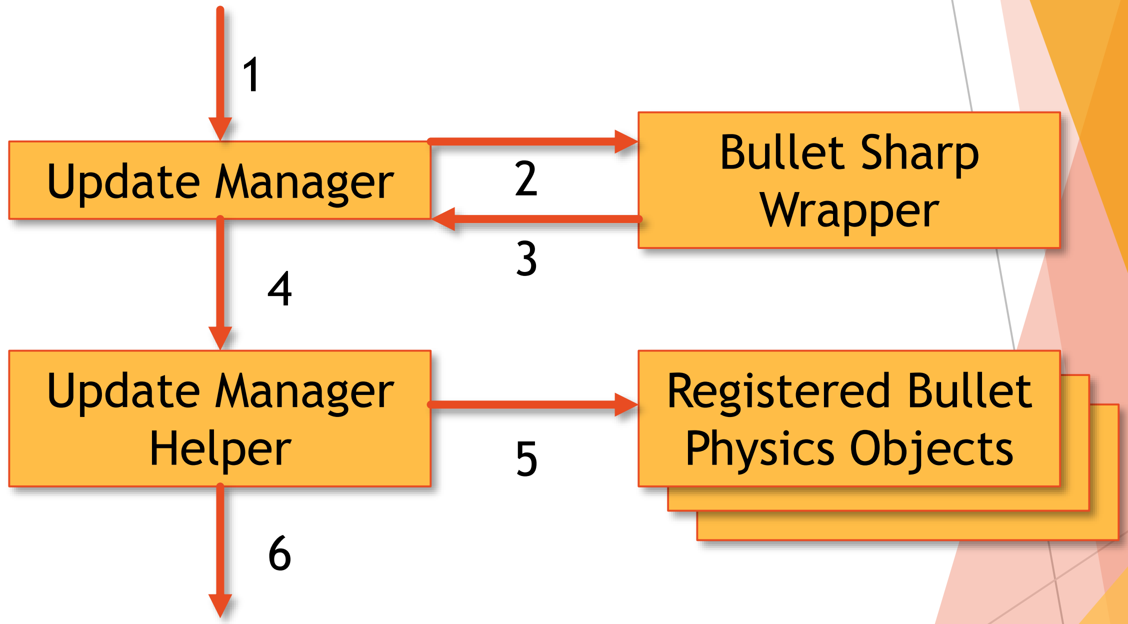
Issues with Timing

- Unity 3D's update cycle is automatic
- Order in which game objects update is unpredictable
- Physics updates may not be complete by the end of a frame
  - State conflicts
  - Erratic motion
  - Ignored user input



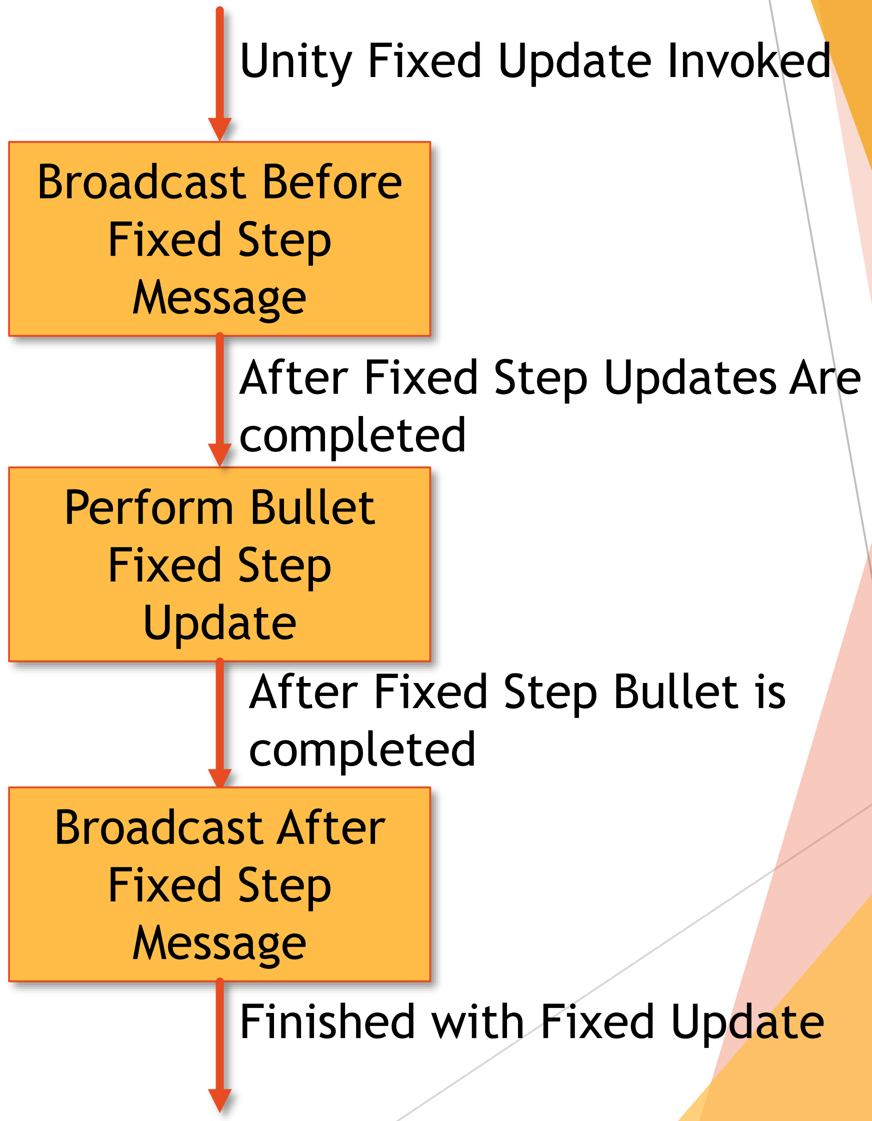
Timing Solution For Partial Step

1. Unity Update Invoked
2. Execute Partial step calculations
3. Return When completed
4. Invoke Message Broadcast on completion
5. Message received by registered listeners invoking an after partial step function
6. return



Timing Solution For Fixed Step

- Fixed Step updates use yield breaks
- Frame rate is locked and can’t be dynamic
- Does not interfere with existing Unity systems
- Controls timing reliably


Unity Fixed Update Invoked
Broadcast Before Fixed Step Message
After Fixed Step Updates Are completed
Perform Bullet Fixed Step Update
After Fixed Step Bullet is completed
Broadcast After Fixed Step Message
Finished with Fixed Update

What is Done

- Mesh generation for primitive objects and 3D model optimization
  - Vertex mapping, normal mapping, UV mapping, and triangle mapping
- Collision object creation and management
  - Support for sweeping, AABB, and simple static area broad phase collision
  - Collison groups and filters
- Message board system for update management
- Optimized Bullet Sharp wrapper

What is Left

- Finish update manager
- Soft body physics support
  - Fluid systems
  - Voxel based mutable meshes
- Inverse Kinematic Constraints
- Character controllers
- User interface

Questions