Overview

The Ghost Decoy System is a gameplay feature that allows players to leave behind a temporary afterimage (ghost version of themselves) that serves as a teleport anchor and a tool for misdirection. This mechanic introduces strategic repositioning and an extra layer of movement into Lyra's gameplay, making engagements more dynamic and skill-based.

Content

How It Works

- The player presses a button to spawn a Ghost Decoy at their current position.
- The decoy remains for X seconds (e.g., 3-5 seconds) before disappearing.
- The player can choose to teleport back to the decoy's position before it vanishes.
- If the clone is ignored, it simply fades out naturally after its timer expires.
- The clone does not move, attack, or interact with anything—it is purely an afterimage for misdirection and repositioning.

Visual Brief

Visual Description

- Decoys should have a semi-transparent, flickering appearance to distinguish them from real players.
- The decoys should spawn instantly with a subtle distortion effect.
- When disappearing, the decoy should dissolve, flicker out, or break apart in a visually appealing way.

Functional Overview

Priority	Function	Description	Comments
Must	Decoy Spawns on Activation	Players can press a button to leave behind a stationary afterimage (Ghost Decoy).	Completed
Must	Decoy Disappears After Set Time	The afterimage vanishes after a short duration (e.g., 3-5 seconds).	Completed
Must	Optional Teleportation to Decoy	Players can choose to teleport back to their decoy before it disappears.	Completed
Must	Visual & Audio Feedback	Decoys have a ghostly flickering effect and emit a unique sound when spawned and expired.	Completed
Should	Decoy Teleportation Window	Players should only be able to teleport while the decoy is active, preventing last-second abuse.	Completed
Could	Enemy Decoys Look Different	Players should be able to tell enemy decoys apart from real players.	Backlog

Technical Needs

- Blueprint Support: The feature should be designed with Blueprints to allow easy iteration and expansion.
- Replication Consideration: Since Lyra is multiplayer-focused, the Ghost Decoy System needs proper replication handling.
- Material & Shader Work: A ghost-like visual effect for decoys (transparency, flickering, or distortion) to make them visually distinct.
- Simple Animation Reuse: The decoy should reuse existing player animations, preventing the need for new animation assets.

Would need additional support from:

- VFX/Material Artist – Create visual effects for the ghostly flickering effect on spawn.
- Sound Designer – Design sound cues for clone spawn, existence, and expiration.
    - Would be needed only after the prototype is finished. Lyra lyra already has good placeholder sound cues.

Dual State Ability

Wait Input Press (Not part of Dual State): Ability is granted to players but not activated. Listening for Ghost Decoy Ability Input action Press and sends the information to the Gameplay Ability.

On Ability Active State #1: On First activation, check if a loose gameplay tag is added. If not then spawn GhostCharacter at player location, activate idle timer, send a cooldown message and end ability.

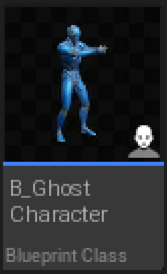
On Ability Active State #2: On Second activation, check if a loose gameplay tag is added. If True, remove the gameplay tag and Commit Ability. Teleports the player to the GhostCharacter, destroys GhostCharacter, sends a cooldown message and ends the ability.

Parameter Cheat Sheet

These are options available for Level Designers to customize the Ghost Decoy.

Folder Location

Where the files for the feature are located.

What and Where	Image Reference
GhostCharacter: Spawnable Decoy Character Location: /All/Plugins/ShooterCore/Game/Ghost	
Health: DT_GhostHealth for health values and a GE_GhostHealthTarget to apply health to the health component within B_GhostCharacter. Location: /All/Plugins/ShooterCore/Game/Ghost	DT_Ghost Health Data Table (Attrib... GE_Ghost HealthTarget Blueprint Class
Cooldown: Grants then Listens for (TagName="Ability.Ghost.Duration.Message"). If Ability is committed, a cooldown is initiated both for ability activation and UI Icon. Location: /All/Plugins/ShooterCore/Weapons/Ghost	GE_Ghost_ Cooldown Blueprint Class W_Ghost Cooldown Widget Blueprint
Gameplay Ability: Waits for input press to active ability. It's a Dual State Ability which determines the execution flow based on a first press and a second press depending on a loose gameplay tag. Location: /All/Plugins/ShooterCore/Input/Abilities	

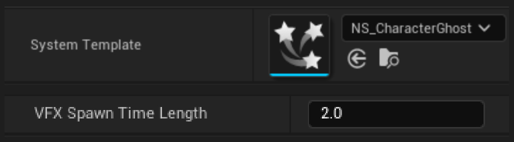
Ghost Input action: Used within Lyra's IMC_ShooterGame to set the Input for the Ghost ability. Also referenced in W_GhostCooldown as Associated Action.

Location: /All/Plugins/ShooterCore/Input/Actions

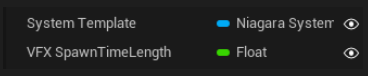
Ghost Character

Within the B_GhostCharacter, there are two designer variables that can be configured. The "VFX SpawnTimeLength" and "System Template"

- VFX SpawnTimeLength allows designers to set the time length for the System Template VFX when the GhostCharacter is spawned and destroyed.
- System Template allows designers and VFX artists to test and swap between different Niagara Systems Particle Effects for the characters on spawn and death VFX. Does not affect OnDeath gameplay message VFX, since it's a Lyra GCNL trigger and is used for all Lyra Death VFX Types.



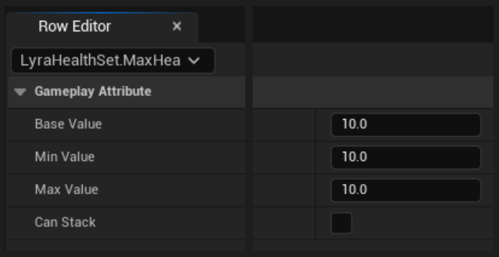
System Template	NS_CharacterGhost
VFX Spawn Time Length	2.0



System Template	Niagara System
VFX SpawnTimeLength	Float

Ghost Health

Within DT_GhostHealth, designers can configure the ghost health however they like. The ghost can have a minimum value of 1 or a value of 200 or more as health. The important part to remember is that the lower the value the health is set as, the less damage it takes to kill the ghost.



Row Editor

LyraHealthSet.MaxHea

Gameplay Attribute

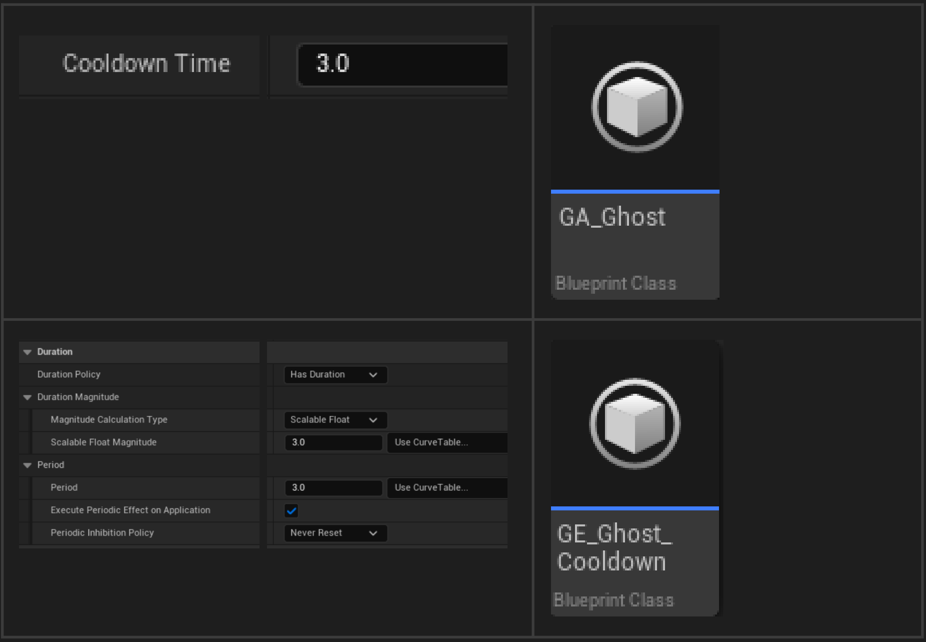
Base Value	10.0
Min Value	10.0
Max Value	10.0
Can Stack	☐

Cooldown

Changing cooldown is fairly straightforward. First the cooldown variable located in the details window/tab for "GA_Ghost", which handles the gameplay cooldown of the ability and drives the self-destruct timer for the "B_GhostCharacter".

- How long the player needs to wait before performing the action again.
- And how long to set the "B_GhostCharacter" self-destruct timer based on the cooldown value.
    - If Cooldown is set to 10 seconds. When the ability is active and a ghost is spawned, the ghost will stay alive for 10 seconds. Afterwards the cooldown activates for another 10 seconds and the ghost self-destructs. Total wait time becomes 20 seconds if the ghost stays idle/self-destructs.
    - Total wait time becomes less if the ghost is killed by an enemy or if the player teleports to it because the cooldown activation activates much earlier in this case.

"GE_Ghost_Cooldown" cooldown variables are purely for visual indications. It updates the visuals for the Ghost UI ability icon. Best practice whenever "GA_Ghost" Cooldown is updated, the same should be done for "GE_Ghost_Cooldown" cooldown variables for consistency between ability and UI.



Cooldown Time	3.0

Duration		
Duration Policy	Has Duration	
Duration Magnitude		
Magnitude Calculation Type	Scalable Float	
Scalable Float Magnitude	3.0	Use CurveTable...
Period		
Period	3.0	Use CurveTable...
Execute Periodic Effect on Application	✓	
Periodic Inhibition Policy	Never Reset	

Future Problems / Considerations

Problem	Possible Solution	Disciplines Required	Comments
Two of the same problems. Event Possessed. Didn't trigger since the ghost isn't possessed by a controller. Controller-based inventory setup. Broke due to lack of possession — Ghost isn't controlled.	There should be a new class of type LyraCharacterWithAbilities that has a valid non-playable controller agent. That way, we can script these none player characters in many ways we could not do before. Such as adding inventory items and other controller abilities.	C++ Programmer	It is known to be an issue when you create a Lyra character to be a non-controllable lyra character. Especially when assigning AI agents to the character. It breaks most, if not all of the blueprints functionality.
Ghost Ability Active State #1. Cooldown UI does not work for the purpose of the ability. Ability Icon directly buffers.wrong indication for players that may think it's on cooldown, but in fact, it's waiting for another input press or incoming damage.	Instead of a cooldown to handle it all. It would be better to use a resource that consumes and replenishes during the ability states and only apply the cooldown buffer once the ability has been used or ignored. Active State #1 ability active - waiting command/Waiting UI Icon - If ghost killed or not used, apply cooldown UI Buffer. Active State #2 ability teleport - apply cooldown UI Buffer	Gameplay Programmer and a UI Programmer.	This could be a good opportunity to create a resource and consume GameplayEffect for current and future Gameplay Abilities that would require it.
GA teleport rotation logic. Inconsistent / didn't affect network control rotation.	Perhaps the "Teleport" node doesn't support rotation within GA's. Either the logic setup is incorrect or the logic needs to support GA's.	Gameplay Programmer and C++ programmer.	
Lyra Team SubSystem. By default there is no way to set a player's team manually.	The solution that I found was to make the ChangeTeamForActor a C++ function blueprint callable within LyraTeamSubSystem.h	C++ programmer	We need to look into it a bit more if it should be exposed to blueprints or not before merging it into main. Otherwise we need to find another solution to the problem