SENSES COMPONENT

The **Senses Component** is your AI's way of understanding the world around it. Once added to an actor, it starts paying attention to anything nearby that has a **TargetSenses** component. Depending on the senses you enable—Sight, Hearing, or Smell—your character will naturally begin noticing others through vision cones, noises, scent trails, and overall awareness that rises and fades over time.

The goal of Senses 2026 is to give designers a simple, intuitive perception system that "just works." Instead of writing detection scripts or chaining raycasts, you only adjust a few familiar sliders like vision angle, hearing sensitivity, or smell radius. From there, the Senses Component quietly handles all the heavy lifting: checking line-of-sight, updating awareness, remembering targets, and forgetting them when they're no longer detectable.

Whether you're making a stealth enemy, a guard who responds to noise, or a creature that follows a scent trail, the Senses Component provides clean, reliable perception data that your gameplay logic can easily use.

Awareness

Everything inside the Senses Component revolves around a single idea: **Awareness**.
Awareness is how strongly each Sense—Sight, Hearing, or Smell—believes a target is actually there. Each Sense builds its *own* awareness value, always ranging from **0 to 100**, depending on how well it detects the target.

- **0** means the sense has no clue the target exists.
- **100** means the sense is completely sure.

Awareness rises smoothly when a Sense receives strong information (clear vision, loud noise, fresh scent), and falls gradually when that information fades away—based on your forgetting settings. This creates believable detection behavior: an enemy doesn't instantly snap into alert mode the moment you appear; instead, their awareness climbs naturally as they gather more evidence.

A target might be faintly audible (low Hearing Awareness) or clearly visible (high Sight Awareness), and each Sense contributes independently. Later, during each update, the change in Awareness is called **Delta Awareness**—it's simply how much Awareness goes up or down based on what the senses perceive at that moment.

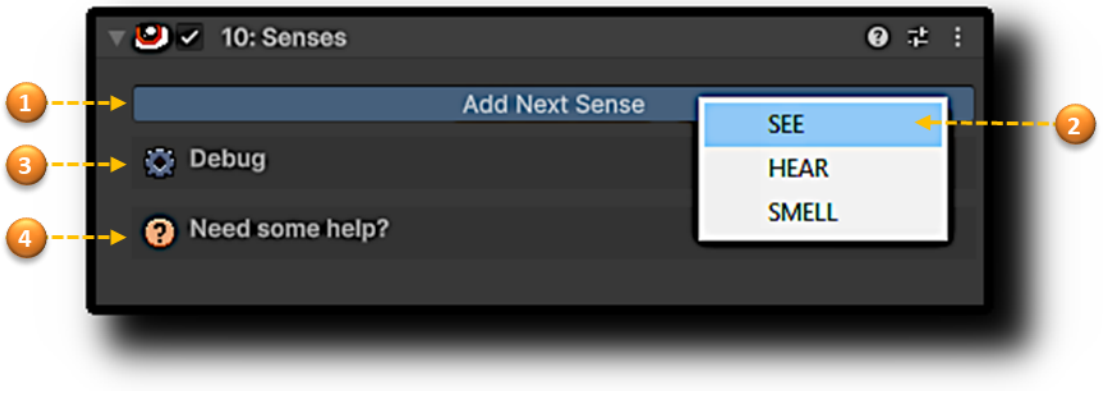
See Inspector Tab

When you add the **Senses Component** to any GameObject, you'll see a clean inspector divided into collapsible sections. This layout helps you focus on the senses you want to work with while hiding everything else until you need it. At the top, you'll find the main controls that let you enable or disable each sense independently—Sight, Hearing, or Smell—along with tools for adding new senses, configuring awareness, debugging, and tweaking general settings.

Each sense you enable expands into its own detailed section. For example, turning on Sight reveals settings for field-of-view angles, ranges, line-of-sight behavior, peripheral vision, and more. Enabling Hearing lets you tune noise sensitivity. Adding Smell exposes scent radius and refresh-rate options. The inspector is designed so you only see the parts of the system your AI actually uses.

If this is your first time setting up an AI with perception, don't worry—every sense has a short description and familiar sliders like radius, angle, or sensitivity. You can begin with defaults, experiment freely, and open more advanced foldouts only when you need them. The entire Senses Component is built around discoverability: you'll understand how things work simply by exploring the inspector.

Adding Your First Sense



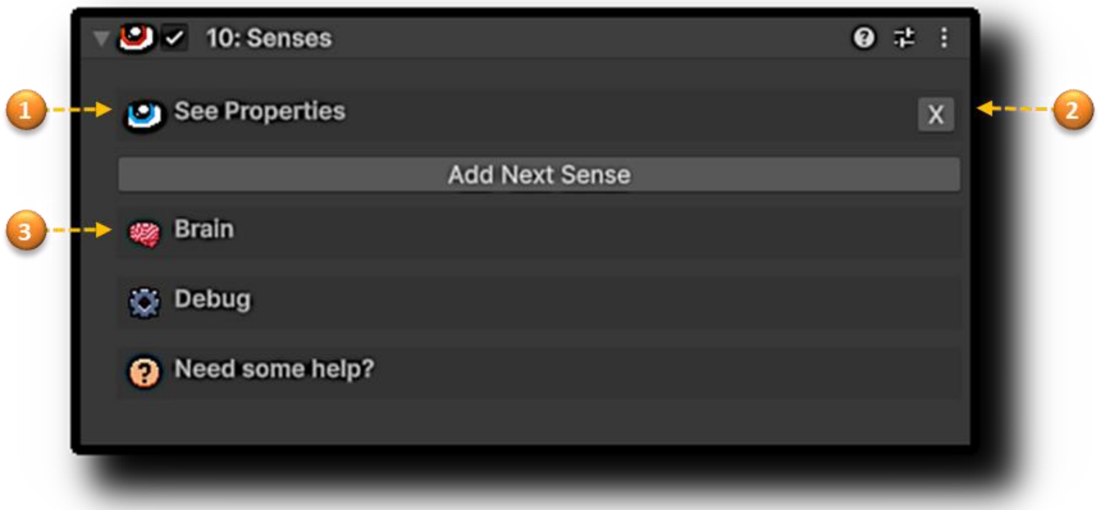
When you first add the **Senses Component**, the inspector intentionally keeps things clean and simple. Instead of overwhelming you with dozens of options, the custom inspector only shows what you actually need at the start. The most important control is already front-and-center: the **Add Next Sense** button **(1)**.
Clicking this button opens a small menu **(2)** with all available senses:

- **SEE**
- **HEAR**
- **SMELL**

You can enable just one sense, or add multiple—Senses 2026 is designed to work with any combination. As soon as you choose a sense, its dedicated settings tab will appear below, ready to configure.

You'll also notice the **Debug tab (3)** and the **Help section (4)**. The Debug tab provides a quick way to visualize what your AI is currently detecting, and the Help section links you to explanations and tips. We'll explore both later in this document, once you've added your first sense.

After Adding Your First Sense

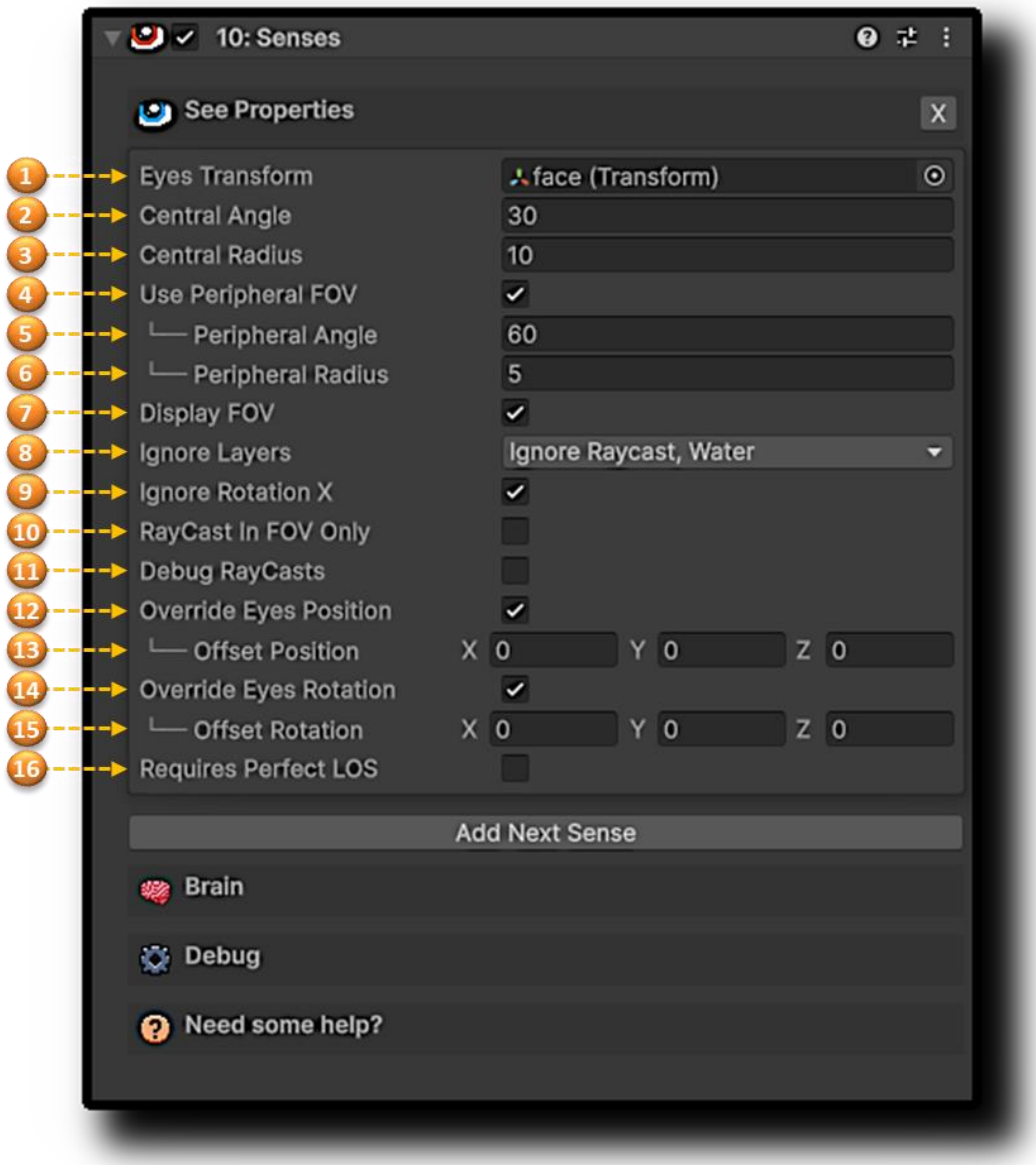


Once you add your first sense from the **Add Next Sense** menu, a brand-new foldout appears in the inspector: **See Properties (1)**. Each sense you enable creates its own foldout like this—easy to spot, easy to expand, and neatly separated from the others. Clicking the foldout reveals all configuration options for that specific sense.

If you later decide a sense is no longer needed, simply use the small **X** button **(2)** on the right side of its header. Removing a sense instantly hides its settings and disables its logic, keeping your inspector clean and focused.

You'll also notice the **Brain** foldout **(3)** showing up as soon as any sense is active. The Brain handles shared behavior such as awareness and memory, so it only becomes visible when at least one sense is enabled. We'll cover this in more detail later—once you've seen how individual senses work.

Configuring Your First Sense


10: Senses

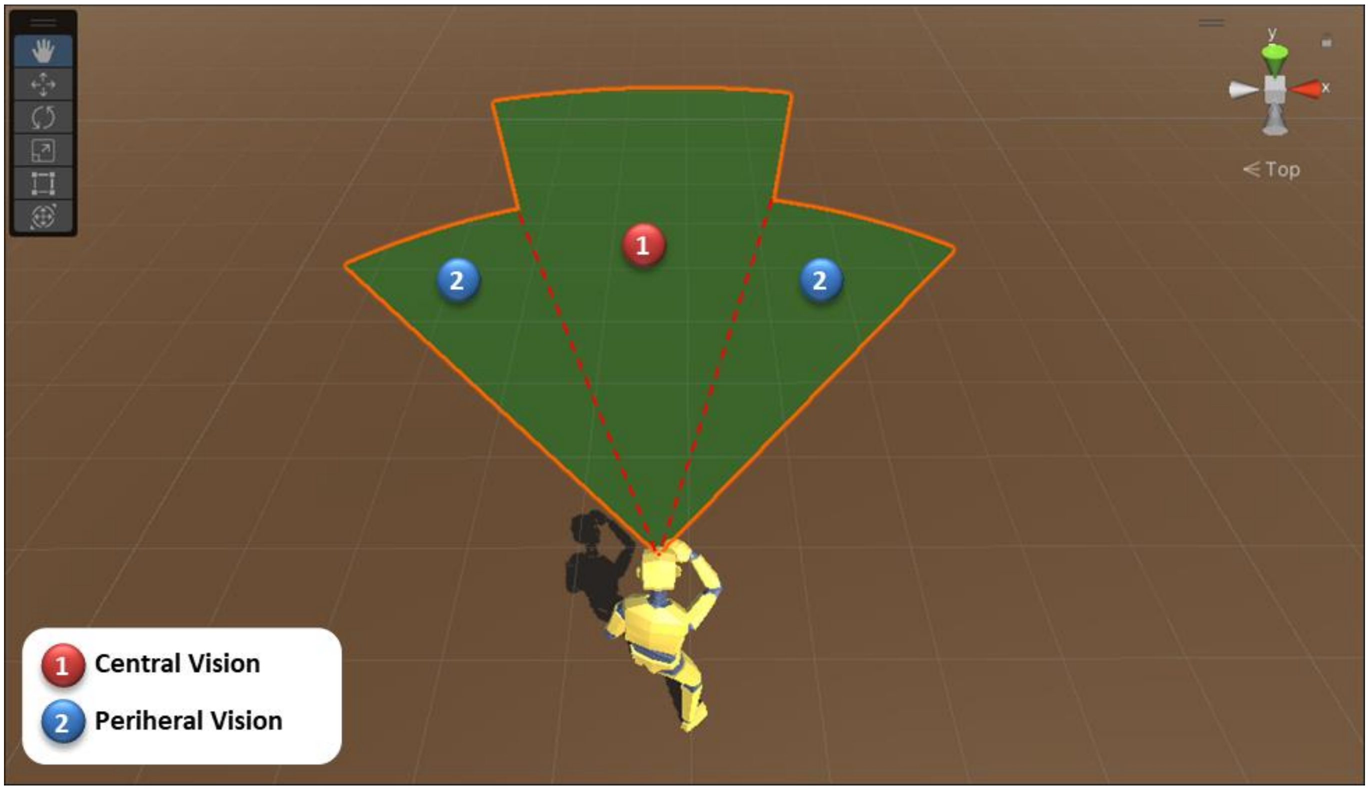
See Properties

1 Eyes Transform	face (Transform)
2 Central Angle	30
3 Central Radius	10
4 Use Peripheral FOV	✓
5 └── Peripheral Angle	60
6 └── Peripheral Radius	5
7 Display FOV	✓
8 Ignore Layers	Ignore Raycast, Water
9 Ignore Rotation X	✓
10 RayCast In FOV Only	
11 Debug RayCasts	
12 Override Eyes Position	✓
13 └── Offset Position	X 0 Y 0 Z 0
14 Override Eyes Rotation	✓
15 └── Offset Rotation	X 0 Y 0 Z 0
16 Requires Perfect LOS	

#	Field	Description
1	**Eyes Transform**	The transform used as the AI's "eyes." Its position becomes the origin for all vision checks and line-of-sight raycasts.
2	**Central Angle**	The main forward-facing vision cone angle (in degrees). Determines how wide the AI can see straight ahead.
3	**Central Radius**	The maximum distance of the central vision zone. Targets inside this area are detected with full reliability.
4	**Use Peripheral FOV**	Enables a wider secondary field of view for weaker side vision.

5	**Peripheral Angle**	The angular width of the peripheral vision zone. A wider value increases side awareness.
6	**Peripheral Radius**	How far peripheral vision reaches. Usually shorter than the central radius.
7	**Display FOV**	Displays central and peripheral vision cones in the Scene View for easy visual debugging.
8	**Ignore Layers**	Layers excluded from detection and line-of-sight checks.
9	**Ignore Rotation X**	Keeps the field of view level with the ground, ignoring vertical tilt. Useful when head animations tilt up/down but vision should stay horizontal.
10	**RayCast In FOV Only**	Restricts line-of-sight raycasts to targets inside the 2D FOV triangle. Objects far above or below the cone are ignored.
11	**Debug RayCasts**	Draws debug lines for every LOS raycast: **Green** = clear, **Red** = blocked. Great for tuning vision behavior.
12	**Override Eyes Position**	Enables a custom positional offset for the eye origin point.
13	**Offset Position**	A Vector3 offset applied to the eye position when **Override Eyes Position** is active. Useful for shifting the eyes up/down/forward.
14	**Override Eyes Rotation**	Enables a custom eye rotation instead of using the GameObject's rotation.
15	**Offset Rotation**	A Vector3 rotation offset applied when **Override Eyes Rotation** is enabled. Helps fine-tune where the AI is "looking."
16	**Requires Perfect LOS**	If enabled, characters with TargetSenses can block each other's view. If disabled, the AI can see through actors in a crowd so overlapping characters never hide what's behind them. Obstacles still block or reduce LOS normally.

Central Vision & Peripheral Vision



The See Sense uses two different vision zones to create more natural and believable AI behavior: Central Vision and Peripheral Vision.

Central Vision

Central Vision is the AI’s strongest and most reliable visual zone—the bright forward cone right in front of the character. Targets inside this zone are detected quickly and produce the highest Delta Awareness, especially when they’re close and clearly visible. This is where the AI is most alert, similar to how humans focus straight ahead.

Peripheral Vision

Peripheral Vision is a wider, weaker vision area on the sides. Targets seen here still raise Awareness, but much slower and with less strength. This zone is perfect for “I noticed something moving over there…” moments, making the AI feel more alive without instantly detecting everything.

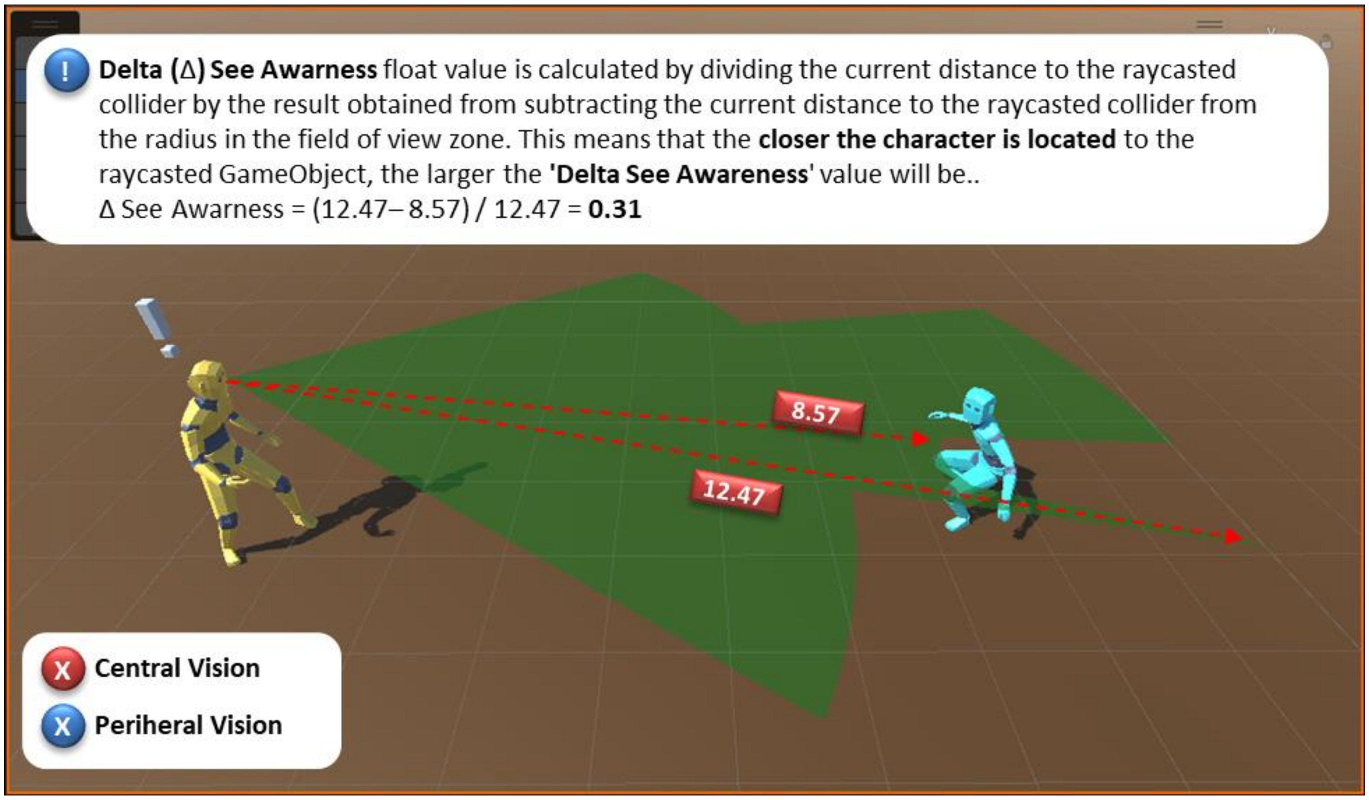
Distance Matters

Both vision zones are affected by distance—the closer a target is, the stronger the visual signal and the faster Awareness grows. A distant target in Peripheral Vision barely nudges the AI, while a nearby target in Central Vision increases Awareness much faster.

The combination of zones and distance is what drives Delta Awareness, which you'll learn more about soon. For now, just remember:

Where the target stands- and how close it is -defines how quickly the AI becomes aware of it.

Delta Awareness – Distance and Field of View Zones Influence



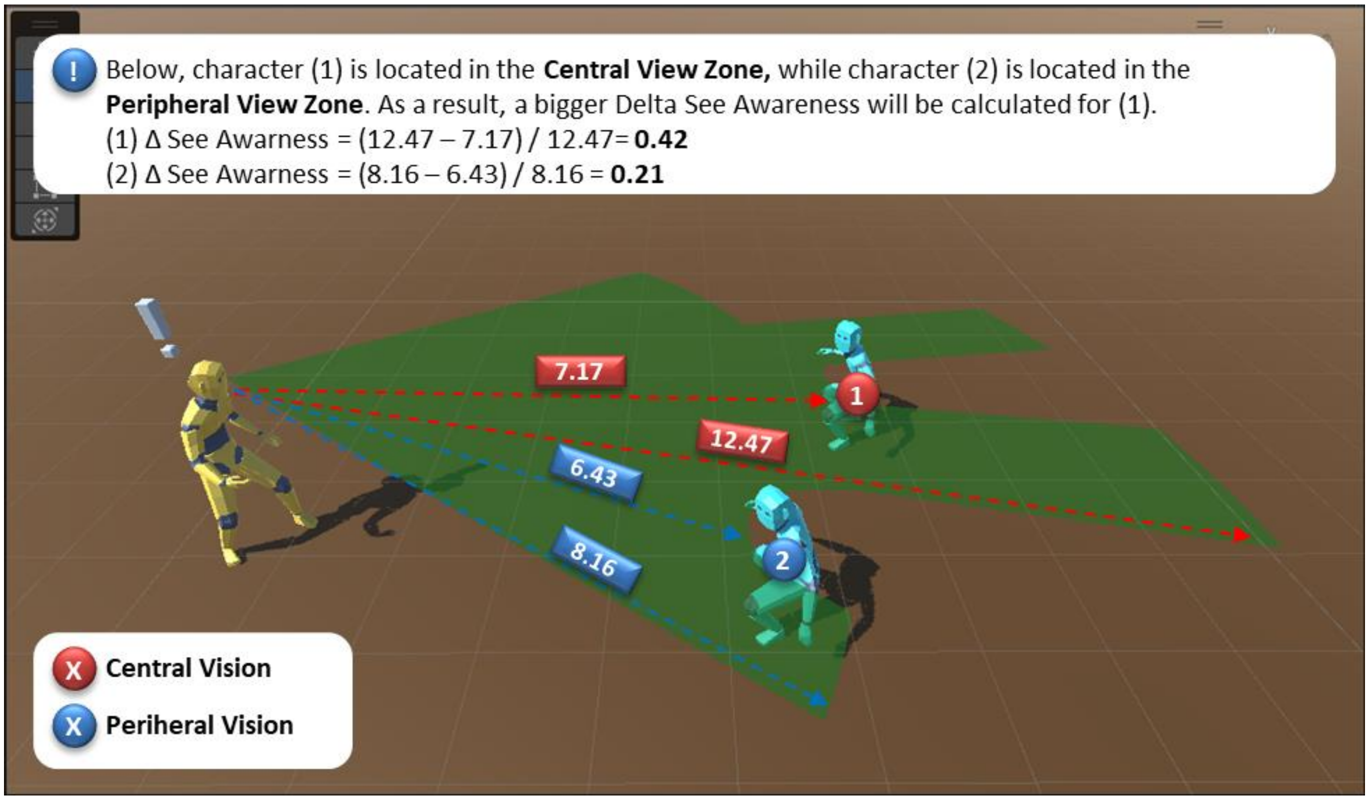
Delta (Δ) See Awarness float value is calculated by dividing the current distance to the raycasted collider by the result obtained from subtracting the current distance to the raycasted collider from the radius in the field of view zone. This means that the **closer the character is located** to the raycasted GameObject, the larger the **'Delta See Awareness'** value will be..
Δ See Awarness = (12.47– 8.57) / 12.47 = **0.31**

Delta Awareness for sight is determined by how close the target is to the **maximum vision radius**. The closer the target stands to the AI, the stronger the Delta Awareness value will be.

In simple terms:

- Close to the AI → strong Delta Awareness
- Far from the AI → weak Delta Awareness

This makes the AI react more strongly to nearby targets and more softly to distant ones, creating smooth and realistic awareness growth.


Below, character (1) is located in the **Central View Zone,** while character (2) is located in the **Peripheral View Zone**. As a result, a bigger Delta See Awareness will be calculated for (1).
(1) Δ See Awarness = (12.47 – 7.17) / 12.47= **0.42**
(2) Δ See Awarness = (8.16 – 6.43) / 8.16 = **0.21**

In this example, both targets stand at different distances and different parts of the field of view:

- **Target 2** is only **6.43 units** away, but inside the **Peripheral Vision** zone.
- **Target 1** is **7.17 units** away, farther than Target 2, but inside the **Central Vision** zone.

Even though Target 2 is physically closer, the AI will assign a **higher Delta Awareness** to Target 1. This is because:

Central Vision is stronger than Peripheral Vision.

Central Vision represents sharp, focused sight — the area where the AI reacts the fastest. Peripheral Vision is weaker and picks up motion slowly, even when the target is near.

Ignore Rotation X

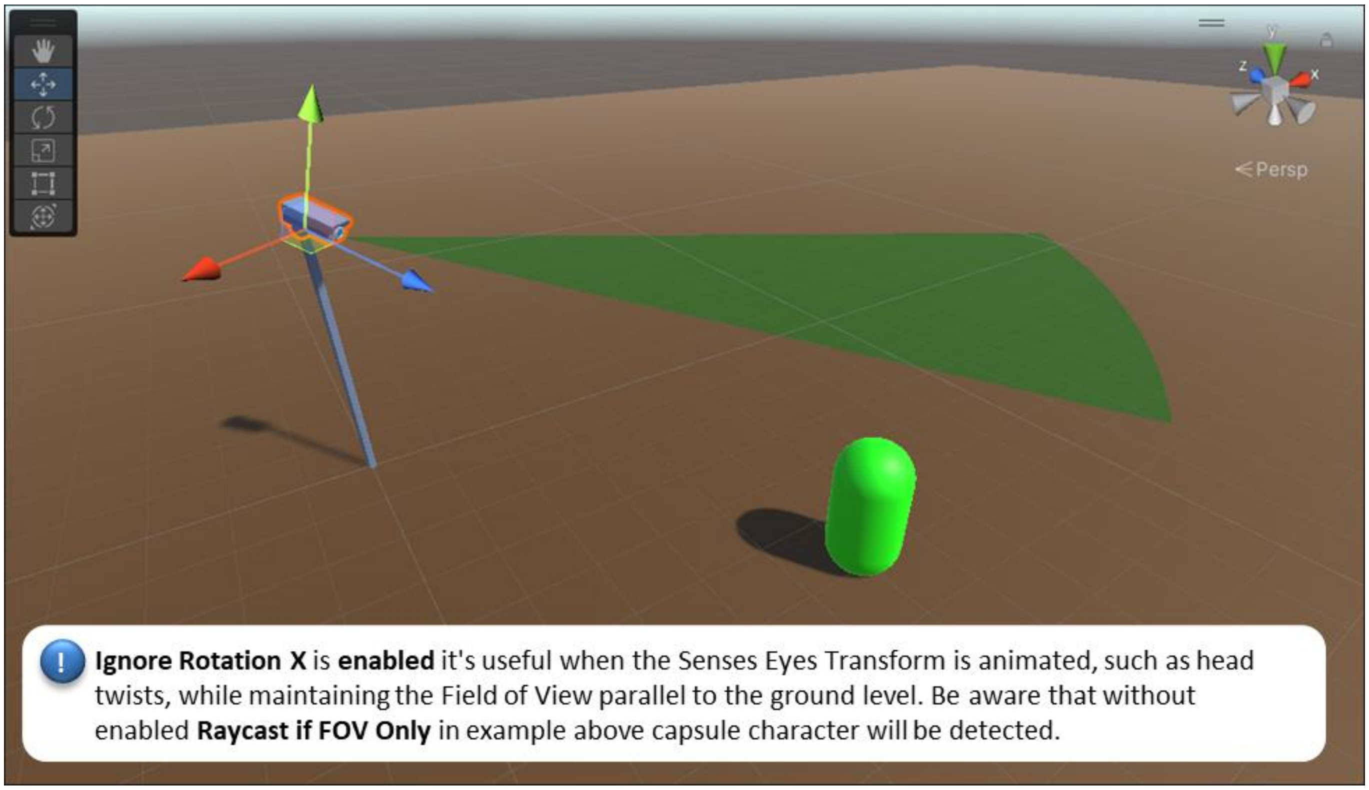
Ignore Rotation X controls whether the AI's field of view tilts up and down with the Eyes Transform. When it is **enabled**, the vision cone stays perfectly level with the ground, even if the character's head or body tilts because of animations. This is extremely useful for animated characters whose head nods, bends, or twists — they keep a stable, horizontal vision plane that won't accidentally detect objects above or below the intended viewing level.

When Ignore Rotation X is **disabled**, the vision cone follows the exact rotation of the Eyes Transform. This gives you fully precise and realistic detection based on the visible FOV shape. If the character looks down, their field of view points down; if they look up, the FOV tilts up.

In short:

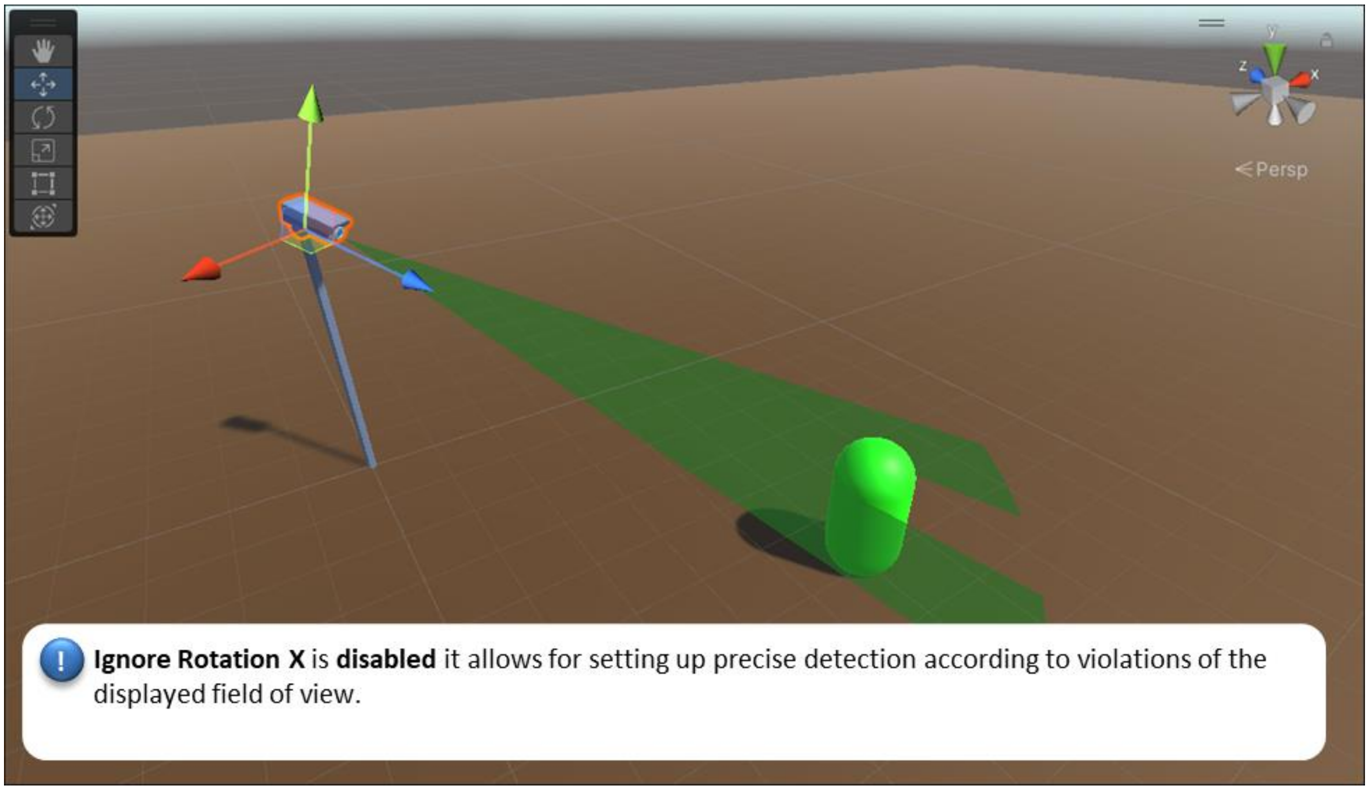
- **Enabled** → Stable, horizontal FOV. Great for animated heads, idle motions, or uneven terrain.
- **Disabled** → Exact FOV tracking. Great for precision aiming, snipers, turrets, or characters that must detect based strictly on what they look at.

Ignore Rotation X (enabled)



Ignore Rotation X is **enabled** it's useful when the Senses Eyes Transform is animated, such as head twists, while maintaining the Field of View parallel to the ground level. Be aware that without enabled **Raycast if FOV Only** in example above capsule character will be detected.

Ignore Rotation X (disabled)



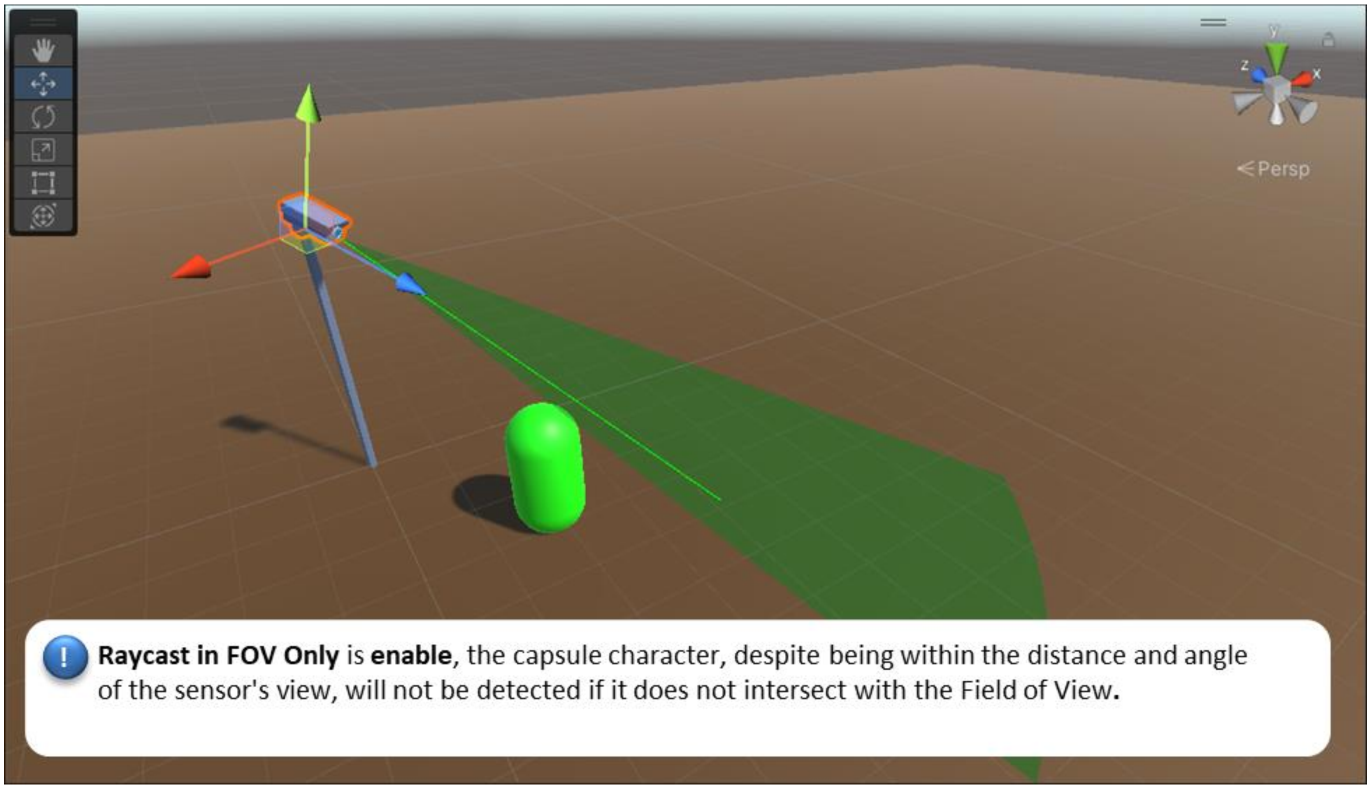
Ignore Rotation X is **disabled** it allows for setting up precise detection according to violations of the displayed field of view.

Raycast in FOV Only

Raycast In FOV Only decides whether a target must physically **touch the Field of View cone** to contribute any Awareness.

Raycast in FOV Only (enabled)

A target **only gathers Awareness** if its collider actually **intersects the visible FOV cone**. If a character is crouching, crawling, or positioned *just below* the cone, Awareness will **not** increase — even if they are directly in front of the sensor and within range.

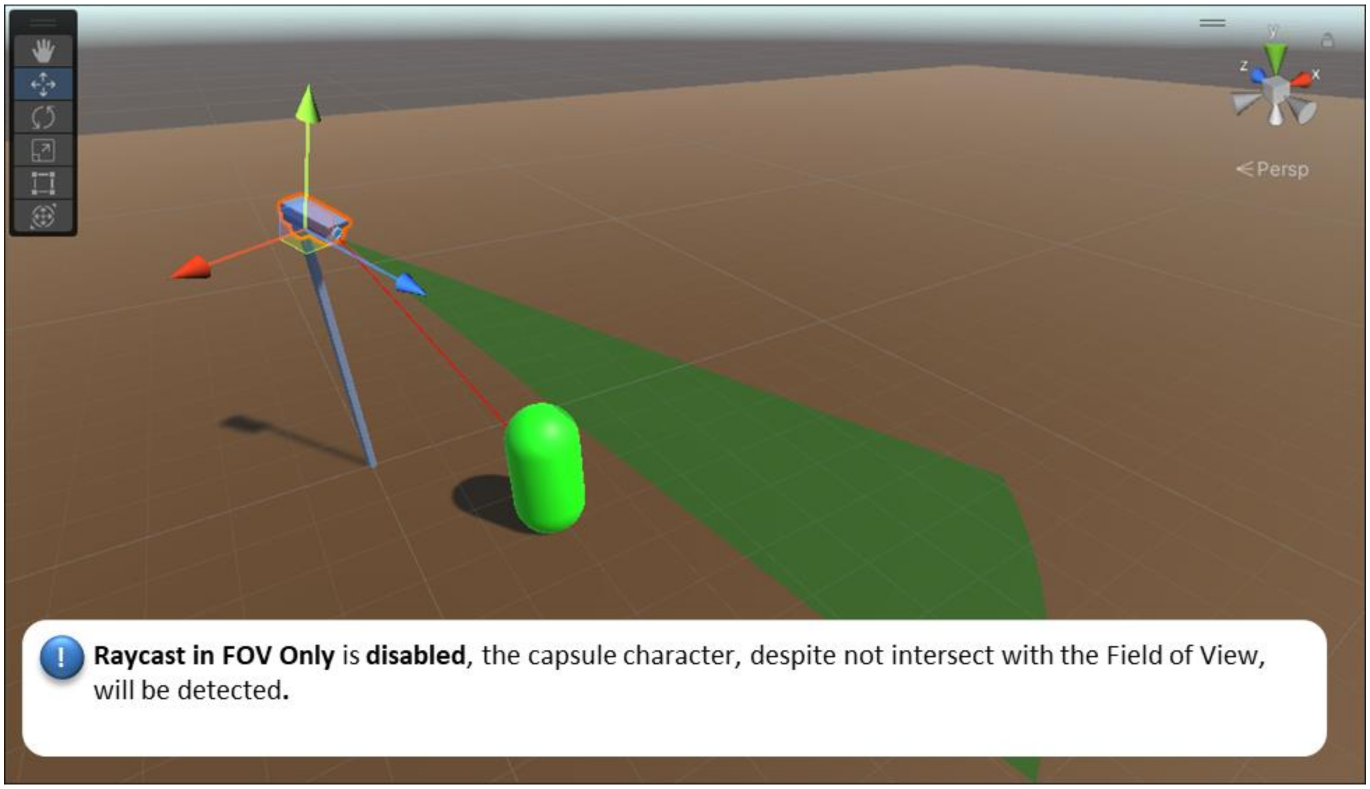


Raycast in FOV Only is **enable**, the capsule character, despite being within the distance and angle of the sensor's view, will not be detected if it does not intersect with the Field of View**.**

Example:
A crawling character stays below the cone → **Awareness remains at 0**, because the cone does not touch them. This mimics realistic vision: if the cone doesn't reach the target, no Awareness is gathered.

Raycast in FOV Only (disabled)

The target does **not** need to intersect the cone. As long as a raycast from the Eyes Transform reaches the target, Awareness **will** increase.



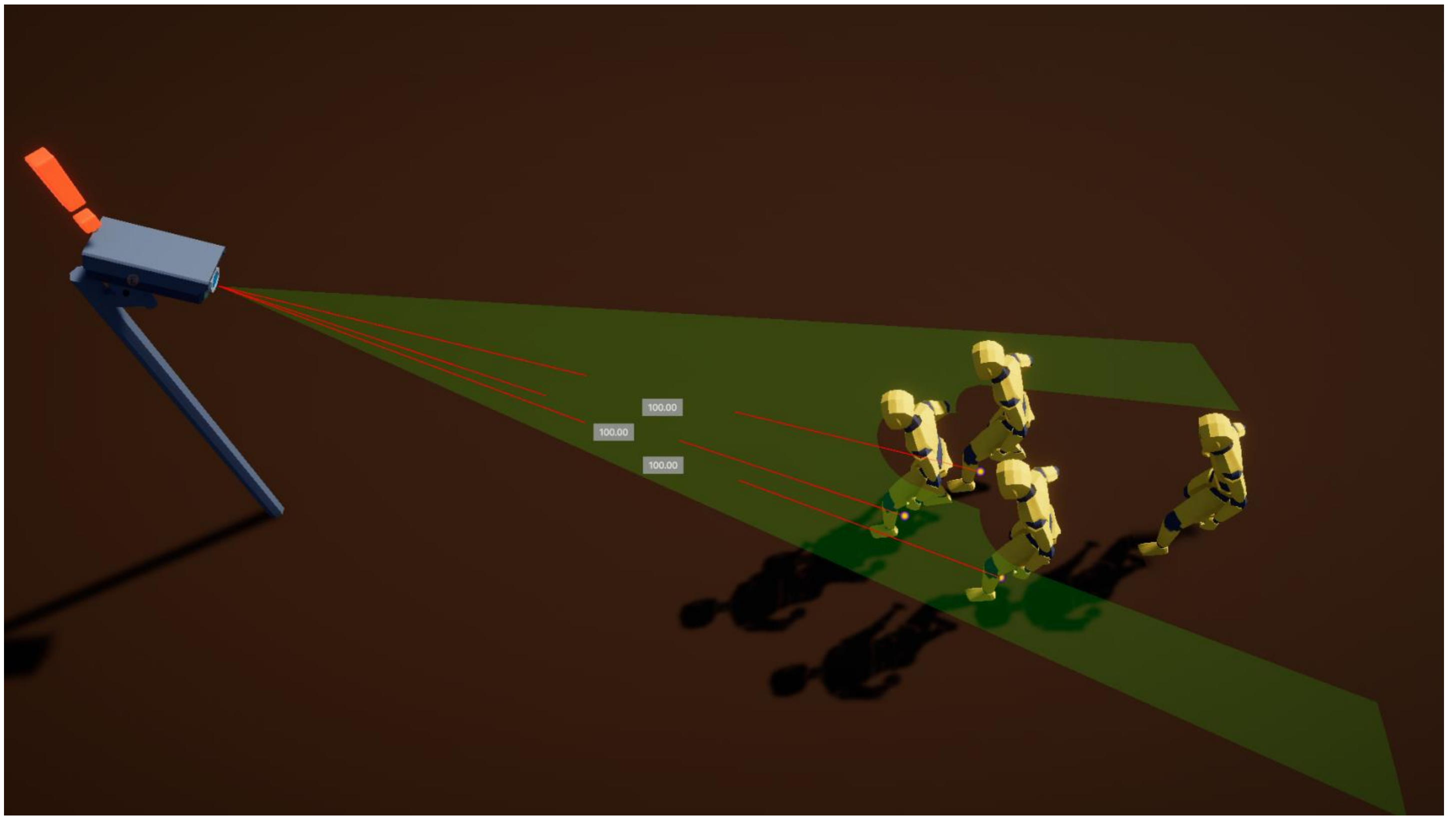
Raycast in FOV Only is **disabled**, the capsule character, despite not intersect with the Field of View, will be detected**.**

Example:
The same crawling character below the cone → Awareness **does** rise, because the ray reaches him even though the cone does not.

Require Perfect LOS

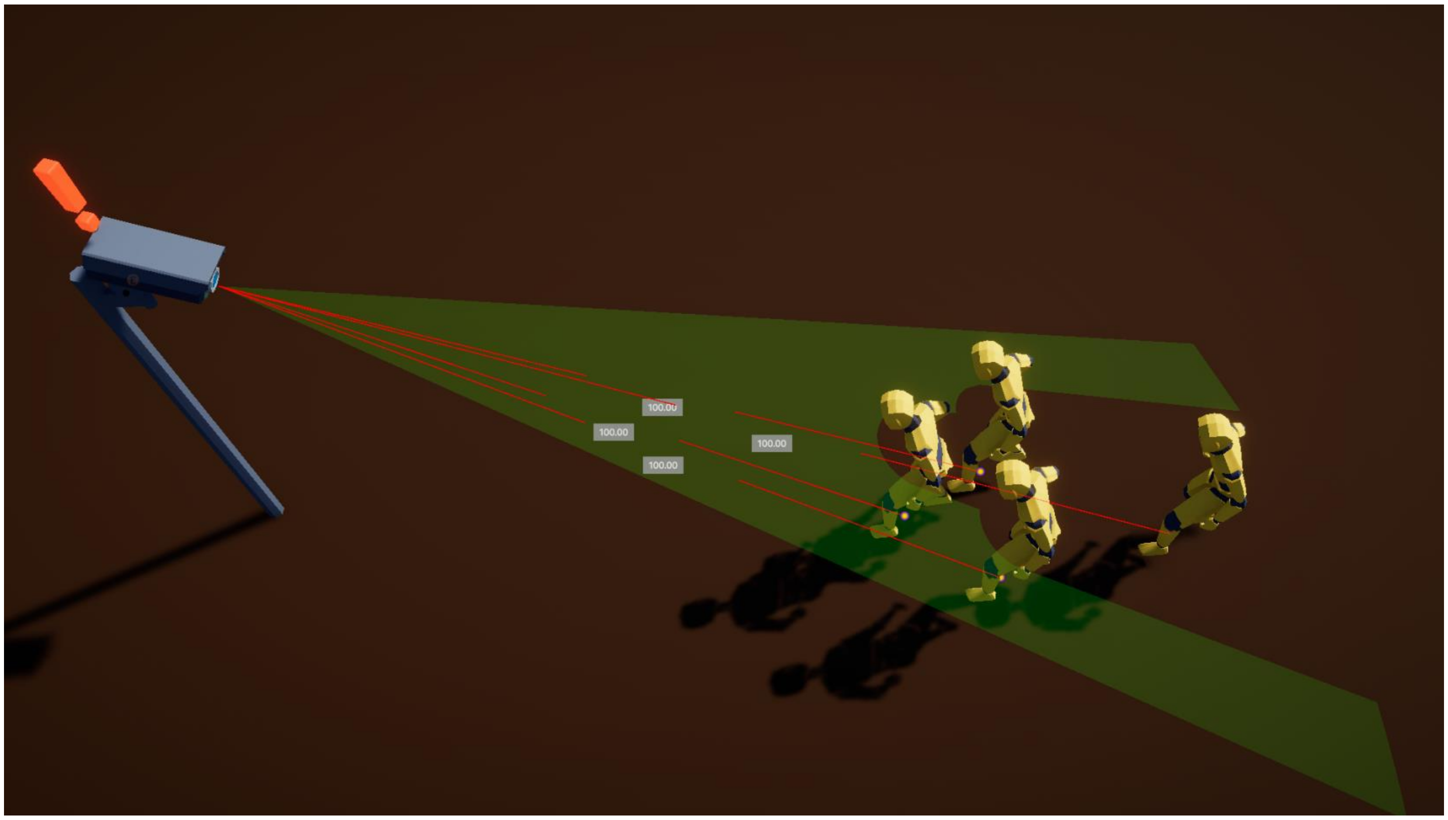
Requires Perfect LOS determines whether actors can visually obstruct one another when the sensor gathers Awareness. When enabled, a character standing in front can hide those behind; when disabled, the sensor can gather Awareness through crowds without obstruction.

Require Perfect LOS (enabled)



With Requires Perfect LOS enabled, the sensor uses strict line-of-sight. Actors standing in front visually obstruct those behind them, preventing the hidden characters from contributing any Awareness. In this example, the camera gathers Awareness only from the actors directly visible in the front row, while the actor in the back receives none.

Require Perfect LOS (disabled)



With Requires Perfect LOS disabled, TargetSenses actors no longer obstruct each other. The sensor can gather Awareness through overlapping characters, allowing it to detect every actor in the group regardless of their position. In this example, the camera successfully gathers Awareness from all four actors, even though some stand behind others.

Hear Sense

The Hear Sense lets your AI react to sounds happening in the game world—not by analyzing audio files, but through gameplay noise you choose to emit. Footsteps, impacts, alarms, traps, machinery, or anything you decide can generate noise using a Noise Component or Noise Trap. When a noise is created, the Hear Sense simply checks how strong it is at the AI's position and raises Hear Awareness when the AI is close enough to notice.

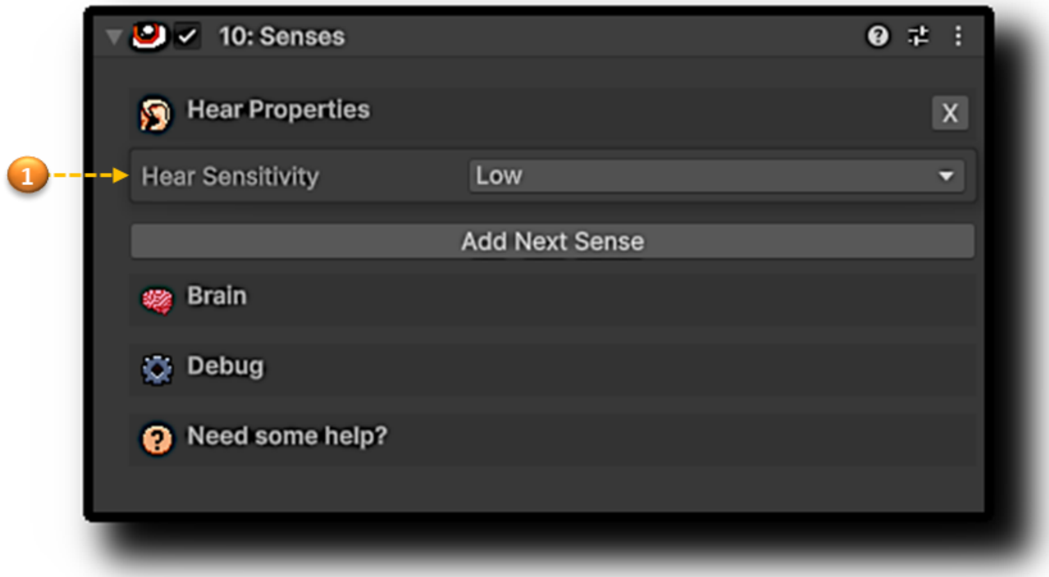
Loud, nearby noises boost awareness quickly, while distant or subtle noises build awareness slowly, creating natural "did I hear something?" moments without any complicated setup. You don't need audio systems or mixers—just trigger the noise when something important happens, and the Hear Sense handles the rest.

This opens up gameplay possibilities that vision alone cannot offer:

- Distract enemies by throwing an object or triggering a noise trap.
- Guide AI behavior using machinery, alarms, or rhythmic sound sources.
- Let guards investigate suspicious noises instead of instantly detecting the player.
- Create stealth gameplay where players must avoid making sound on certain surfaces.
- Shape difficulty by adjusting how sensitive characters are to noise.

Once Hear Sense is enabled, all of this happens automatically. Your AI responds to the noises you choose to emit, creating dynamic, reactive behavior with only a few settings.

Hear Sense Inspector Tab



Hear Sensitivity: Low

#	Field	Description
1	**Hear Sensitivity**	How loud a sound must be before the AI detects it. Higher sensitivity detects quieter noises.

Noise Component

To really understand how the Hear Sense behaves, it helps to take a quick look at the Noise Component and Noise Traps. These are the tools responsible for creating the gameplay noises that the Hear Sense reacts to—things like footsteps, impacts, alerts, machinery, thrown objects, and environmental traps.

Both systems are included in their own dedicated documentation files, where you'll find simple explanations and examples for creating noise sources, controlling how far they travel, and shaping how your AI responds to them. If you want to build stealth mechanics, distractions, or reactive environments, these two components are essential companions to the Hear Sense.

Smell Sense

The Smell Sense lets your AI detect characters based on the scent they leave behind as they move through the world. Instead of reacting only to what's visible or audible, your AI can pick up lingering traces of someone who recently passed by—perfect for creatures, monsters, animals, trackers, or advanced stealth enemies.

As an actor moves, its scent gradually spreads through the environment and fades over time. If another character with the Smell Sense enters the area, it begins raising Smell Awareness depending on how close and how fresh that scent is. Fresh scents create strong awareness quickly, while older scents fade and become harder to detect.

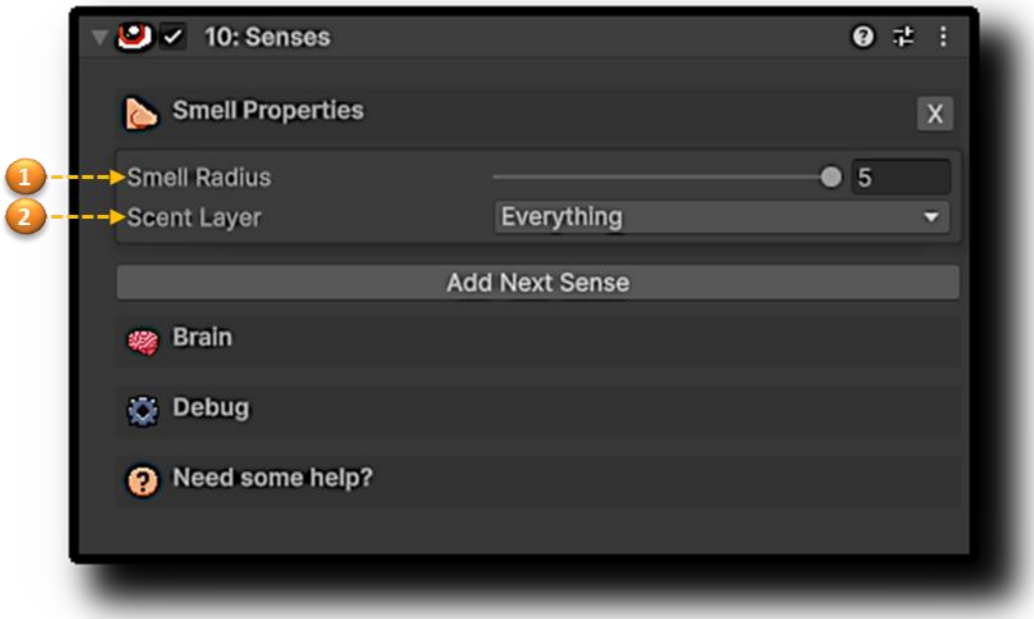
This opens up gameplay possibilities that sight and hearing cannot offer:

- An enemy can follow a player's path into a room even after the player has already left.
- A creature might investigate where a scent trail ends.
- Designers can place Scent Distractor Zones that mask or weaken scent in certain areas, letting players hide their trail.
- Or they can boost scent in specific regions to guide AI naturally along paths.

All you do is enable the Smell Sense, set the detection radius, and choose which layers contain scent sources.

The system handles the rest—providing smooth, natural-feeling smell-based detection with almost no setup.

Smell Sense Inspector Tab



Smell Radius 5
Scent Layer Everything

#	Field	Description
1	**Smell Radius**	Maximum distance at which scent nodes can be detected.
2	**Scent Layer**	Layers containing scent nodes the AI is allowed to detect.

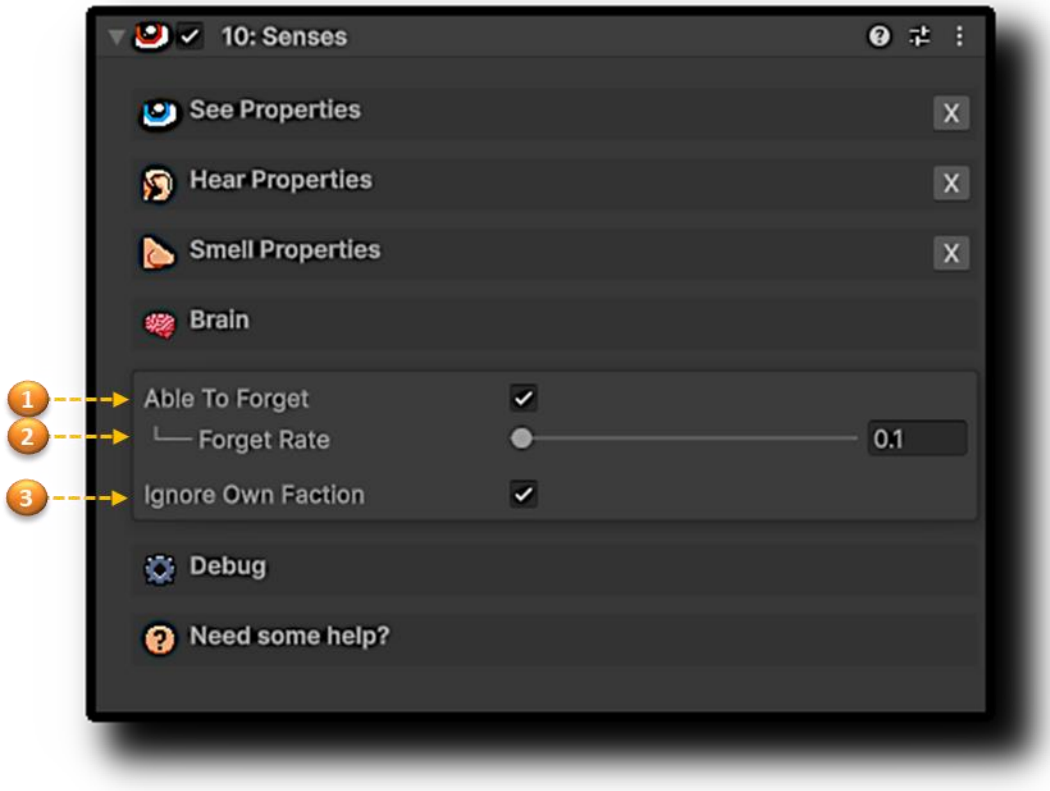
Scent Component

To fully understand how the Smell Sense behaves, it's helpful to check out the Scent Component and Scent Distraction Zones. These systems are responsible for creating and shaping the scent trails that your AI can detect—letting characters leave traces behind as they move, or allowing you to modify how strong or weak scent becomes in specific areas of your level.

Both the Scent Component and the Scent Distractor Zone have their own dedicated documentation files, where you'll find simple explanations, examples, and setup tips. If you want to build creatures that track players, design areas where scent fades quickly, or create zones that confuse or mask trails, these two systems are essential companions to the Smell Sense.

Brain

The Brain foldout controls how your AI thinks about the targets it detects and how that Awareness changes over time. While the senses themselves determine how a target is detected, the Brain determines what happens after that detection. It contains two important settings that shape how your AI behaves in moment-to-moment gameplay.



Able To Forget ✓
Forget Rate 0.1
Ignore Own Faction ✓

Able To Forget

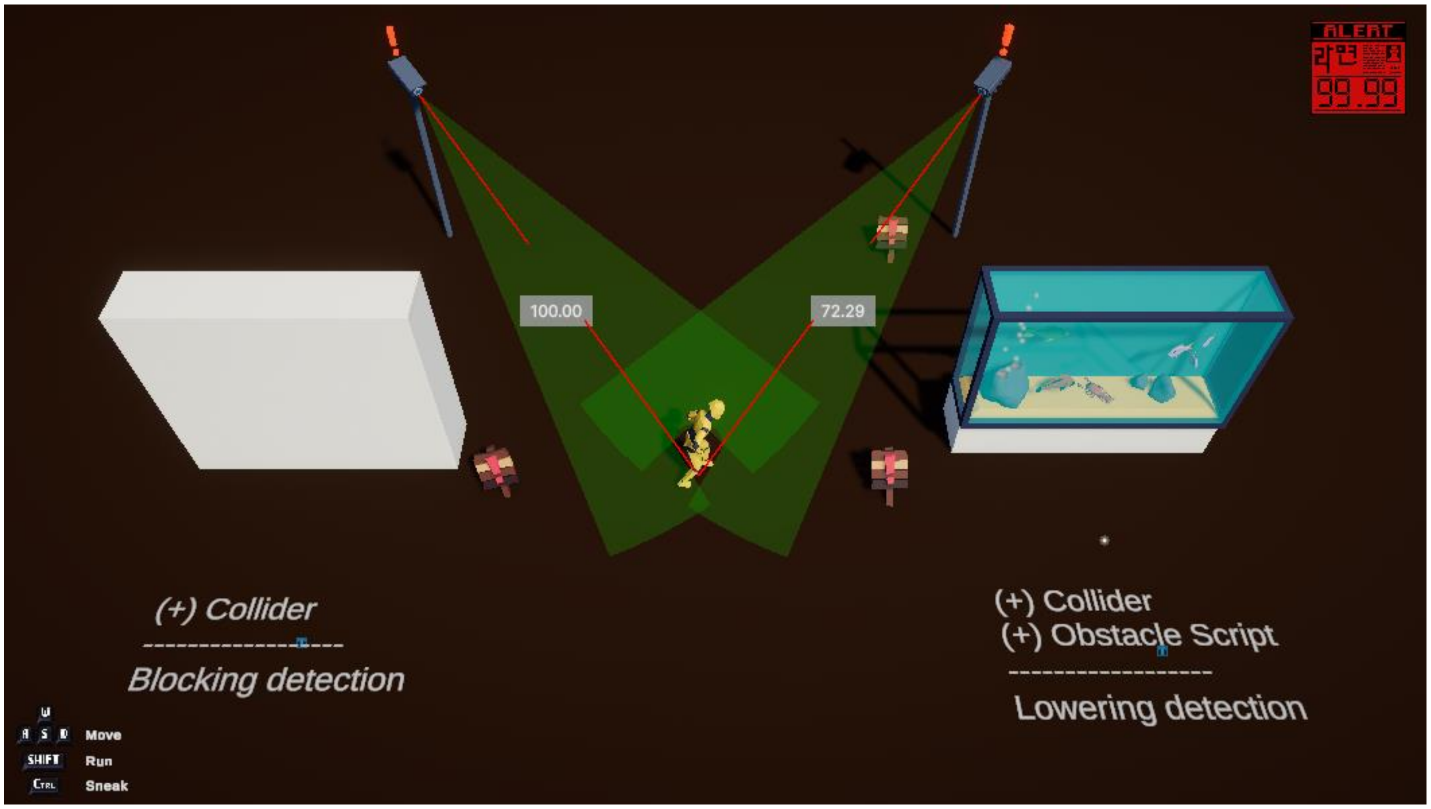
When Able To Forget is enabled, the AI gradually loses Awareness of any target that hasn't been detected during the latest update.
This allows characters to naturally calm down after losing sight, sound, or smell of someone. Awareness fades step by step, creating smooth transitions from "alert" back to "idle," rather than snapping instantly.

Ignore Own Faction

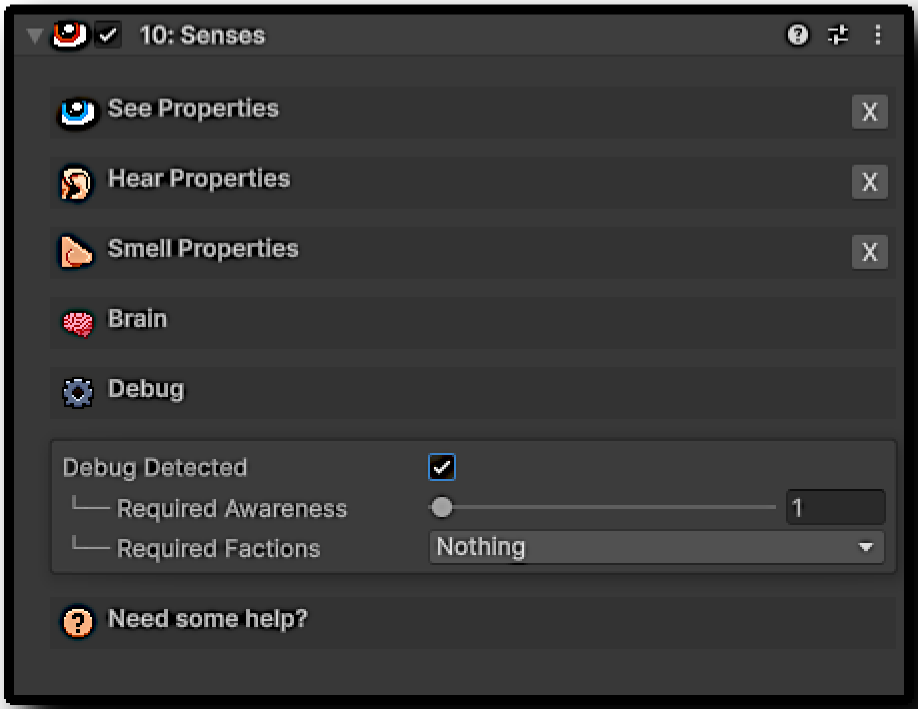
When Ignore Own Faction is enabled, the AI completely ignores characters from the same faction. It won't gather Awareness for allied TargetSenses, meaning teammates, friendly units, or neutral NPCs will never trigger detection or suspicion. This keeps friendly groups from distracting or confusing the AI and makes faction-based behavior simple and clean.

Debug

The **Debug** foldout was added based on user feedback, giving you a simple and powerful way to *see* what your AI senses in real time. When Debug is enabled, each detected TargetSenses actor is visualized with a **red line** drawn from the sensing actor to the target, along with a **numeric label** showing the exact amount of Awareness that sense has gathered.



This makes it incredibly easy to understand why an AI is reacting—or not reacting—in any moment. You can instantly see which characters are being detected, how strong the Awareness value is, and how quickly it changes over time.


10: Senses

See Properties

Hear Properties

Smell Properties

Brain

Debug

Debug Detected ✓

└── Required Awareness 1

└── Required Factions Nothing

Need some help?

To keep things clear, the Debug panel includes two helpful filters:

Required Awareness

Lets you choose the minimum Awareness a target must reach before it appears in the debug view. This prevents clutter and helps focus only on meaningful detections.

Required Factions

Allows you to select which factions will be displayed. Since this is a mask option, you can enable or disable any combination of factions for precise, selective debugging.

Together, these options make debugging and learning Senses far easier. You can watch Awareness rise and fall, understand how zones and distance influence detection, verify that Noise and Scent systems are working, and visually see how settings like Perfect LOS, Ignore Own Faction, and Forgetting affect behavior. It's one of the fastest ways to tune and master the entire Senses system.