OMT (Object Motion Tracker)

Thank you for buying this Object Motion Tracker component. I hope you find the component useful and please do not hesitate to give me feedback or feature suggestions. If you have any support queries you can raise them here: **support@hexagonneuron.com**

Index

Overview

Tracks the movement of a gameObject, and records it's position, rotation and time at user defined distance or time based intervals. Each category of data is stored in easy to access Generic Lists.

- Each position stored is called a **Waypoint.**
- Waypoints make up a **Waypoint Group.**
- There can be multiple Waypoint Groups.
- The waypoint group that is having new waypoints added to it is called the **Active Waypoint Group.**
- All other waypoint groups are called **Non-Active Waypoint Groups.**

There are 3 types of waypoint:

- **Absolute Waypoint** (*Optional*)

Creates an additional waypoint at the start (index position [0] in the active waypoint group), and uses the position of the GameObject plus any offset being applied. This is useful to tweak the waypoints local to the gameObject. The Absolute waypoint is not stored as the waypoint list grows. It is destroyed and re-created each time a new waypoint is added.

- **Offset Waypoint**

This waypoint is where all new waypoints are created. It uses the position of the gameObject, and the results can be offset both manually, percentage based, or track another object!

- **Waypoint**

All waypoints after the offset waypoint are simply referred to as waypoints. They are the previously created offset waypoints.

When combined the absolute waypoint and the offset waypoint allow for complete control of the start of a waypoint group. So for example you may want to use the waypoint position data for particle FX from an engine on a spaceship. This means you can position the waypoints to generate from the engine location. Or maybe track a power up spinning round the player. You get the idea..

Data Collection Types

- **Waypoint Position**

The minimum waypoint information that can be recorded is the gameObject position, as this is obviously required to store the objects movement. Additional information such as the gameObject rotation, and the time since scene start, are optional, and if selected are stored in their own generic lists, and their index numbers will match that of the waypoint position index number.

- **Waypoint Rotation**(*Optional*)

The current rotation of the gameObject being tracked, when the waypoint position was stored. Waypoint rotations are stored in their own generic lists within the waypoint group The index number of the waypoint rotation will match that of the relevant waypoint position.

- **Waypoint Time Stamp**(Optional)

The time elapsed since the start of the scene when the waypoint position was stored. Waypoint time stamps are stored in their own generic lists within the waypoint group. The index number of the waypoint time stamp will match that of the relevant waypoint position.

Data Structure

Waypoint data is made up of a generic list which uses a class to populate it. The class holds all the data for the waypoint group. The class is made up of various data types and generic lists.

```
//The List of waypoints created
	var waypointGroups = new List.<WayPointsClass>();

class WayPointsClass
{
	//Unique Name description for each group
	var id : String;

	//Is this group currently having waypoints added to it
	var activeWaypointGroup : boolean;

	//Current number of waypoints in the group
	var waypointTotal : int;

	//Position of the Waypoint
	var waypointPosition : List.<Vector3>;

	//gameObject Rotation at waypoint position
	var waypointRotation : List.<Quaternion>;

	//Time since start of scene when waypoint was created
	var waypointTimeStamp : List.<float>;
}
```

Setting up...

Drag and drop the OMT_CS.cs (C# Version) or OMT_JS.js (Unity Script Version) component onto the object you wish to track. That's it!

OMT Inspector Options

Waypoint Plotting Setup

- ***Tracking Active***

Enable / Disable Tracking

- ***Track This gameObject***

If no gameObject is specified, the gameObject that OMT is attached to will be used by default. You can place this script on a different object to the one being tracked.

- ***Waypoint Plotting Modes***

Groups: Multiple waypoint groups will be created. When the command is given to stop tracking, the waypoint group will no longer be classed as active and will become it's own separate non-active waypoint group.

Continuous: Only one waypoint group will exist at any time. So when the command to stop tracking is given, no further waypoints will be created. However when tracking resumes, the next waypoint will still be linked to the previous way point. A continuous waypoint group is never destroyed.

- ***Waypoint Interval Modes***

Distance: Waypoint will be plotting after moving a certain distance.

Seconds: Waypoint will be plotted at set amount of seconds.

DistanceAndSeconds*:* Both distance and seconds will plot waypoints.

Frames: Set a frame interval for plotting waypoints. For example 1=Every Frame or 10 = Every 10 frames.

- ***Distance interval for waypoint plotting***

The amount of Unity units to move before plotting a waypoint. Please keep in mind that sudden velocity increases or very high velocity rates will mean that the actual waypoint distance will 'drift'.

- ***Time interval for waypoint plotting***

The amount of seconds to pass before plotting a waypoint.

- **Waypoint Removal Method**

None: Waypoint removal is Off

Active: Only limits the active waypoint group. It is also the default method when using continuous waypoint plotting.

Extreme: When using group waypoint plotting, the oldest waypoints from the oldest non-active waypoint group will be removed.

•**Max Number Of Waypoints**
The Maximum number of waypoints at any time. Note: Includes active AND non-active waypoint groups. The minimum allowed value in this field is 3.

•**Track: Rotation**
Store the gameObject rotation at point of waypoint creation.

•**Track: Time**
Store the time elapsed since the start of the scene at point of waypoint creation.

•**Show Gizmos in Scene Window**
Draw Gizmos in the Scene window to visualize the current waypoints being plotted.

Absolute Waypoint

•***Add Absolute Waypoint***
Creates an additional waypoint at the start (index position [0] in the active waypoint group), and uses the position of the gameObject plus any offset being applied. This is useful to tweak the waypoints local to the gameObject.

To tweak even further the next waypoint after that is the offset waypoint (index position [1]), which is where the actual plotting of waypoints takes place, this can also have offsets applied to it.

On the next plotting cycle the previous absolute waypoint is removed, then the new offset waypoint is added, then a new absolute waypoint is added. This is to maintain waypoint history continuity.

•***Absolute Waypoint Offset***
The amount to offset the absolute waypoint by.

Offset Waypoint

•***Method Of Offset***
Manual: Specify the amount of offset relative to the tracker object.

Percentage: Adjust the amount of distance to offset using a percentage of a min and max Vector3 values. This allows you to associate the offset with another value, for example object speed, power-ups, etc.

Actual: Use a GameObject worldspace position for the offset waypoint position.

•***Manual Offset Amount***
Offset the current waypoint plot relative to the gameObject Z axis. You could use this value to follow another object, or position on a certain point of a player avatar.

This offset is not applied to the percentage offset option below.

•
•**Offset %**

The percentage of offset to apply. For example if 50% then the offset will be between the min and max offset positions.

- ***Min % Offset Distance***

Min: The minimum distance of offset to apply.

- ***Max % Offset Distance***

Max: The maximum distance of offset to apply.

- ***Offset using Object Position***

Offset using an object position. For example there may be a rotating element to the player avatar. This will allow the offset to simply follow it. But keeping the absolute waypoint relevant to the tracked object.

Waypoint Merging

- ***Merge Active Waypoint Group***

Enabled: The current active waypoint group will gradually merge using the merge speed settings, starting with the oldest and moving on to the next waypoint in line.

Disabled: No merging will take place and the active waypoint list will continue to grow (Unless you are using Max Waypoints setting).

Merging can also be disabled by setting Merge Speed to 0.

- ***Merge Speed: Active Waypoint Group***

This value applies to only the Active Waypoint group.

The speed at which the oldest waypoints are merged to the next waypoint in line.

Smaller numbers result in slower merging.

Setting this to 0 will stop waypoint merging, however there is the option above to enable/disable waypoint merging.

If merging cannot keep up due to settings, the last waypoint in the group will be removed each frame as a last resort.

Merging will cause the position information for the oldest waypoint to be updated as it moves closer to the next waypoint.

- ***Merge Active Speed % Adjust***

Enabled: The Merge speed for the active waypoint group can be increased and decreased using a percentage. This allows you to associate the offset with another value, for example object speed, power-ups, etc.

For example if the active waypoint group merge speed is 1 and the percentage is 10% then merging speed will be 0.1.

Disabled: Merging will use the active waypoint group merge speed value.

- ***Merge Rate %***

The percentage of the active waypoint group merge speed to use.

- ***Non-Active Waypoint Group Merging Modes***

Simultaneous: All non-active waypoint groups will merge at the same time.

Sequential: All non-active waypoint groups will merge in order, starting with the oldest waypoint group.

- ***Merge non-active Waypoint Groups***

Enabled: The non-active waypoint groups will all gradually merge at the same speed, starting with the oldest waypoint and moving on to the next waypoint in it's group.

Disabled: No merging will take place and the non-active waypoint groups will persist.

Merging can also be disabled by setting Merge Speed to 0.

- ***Merge Speed: Non-active Waypoint Groups***

This value applies to all non-active waypoint groups.

How quickly the oldest waypoints are merged to the next waypoint in line.

Smaller numbers result in slower merging.

Setting this to 0 will stop waypoint merging, however there is the option above to enable/disable non-active waypoint group merging.

If merging cannot keep up due to settings, the last waypoint in the group will be removed each frame as a last resort.

Merging will cause the position information for the oldest waypoint to be updated as it moves closer to the next waypoint.

Notes

Keep notes of your waypoint set-ups etc.

Debug

General behind the scenes information including the waypoints list.

Code Examples

Included in the project file are a number of example files. They are all documented and should help to get you started. Below is an overview of each file.

- **OMTReadDataFromWaypointGroupExample**
Demonstrates on how to access waypoint data from an external script. You are shown how to access a waypoint group and the index in that waypoint group.

- **OMTChangeSettingsExample**
Example script showing how to change various settings at the start. Kind of like a configurator.

- **OMTSendMessageExample**
Some functions in OMT can be called remotely, either by referencing them directly or by using SendMessage. This script shows you how to use those functions.

- **OMTOffsetFollowObjectExample**
Example showing how to use an objects position as the Offset Waypoint position.

- **OMTTimeTravel**
Example showing how to move an object back in time. For example at a certain pint in time, or an offset of seconds from the current time.

- **OMTMiscExamples**
Other script examples on how to use OMT.

- **OMTFollowWaypoints**
Example showing how to get an object to follow the path in a waypoint group.

- **OMTParticles**
Example showing how to generate particles at waypoint positions.

- **OMTRaycast**
Example showing how to generate a line cast along the length of a waypoint group.

- **OMTUnityLineRendererTrail**
Example showing how to use Unity's Line Renderer to generate motion trails.

- **OMTVectrosityEngineTrail**
Example showing how to use the excellent Vecrosity to generate motion trails.

NOTE: *The* **OMTVectrosityEngineTrail** *scripts have been commented out to avoid errors for users who do not have Vectrosity installed. Just remove the comments and the scripts will work fine. This commenting out will cause a warning for the C# Version of the script which will say: "The class defined in script file named 'OMTVectrosityEngineTrail_CS' does not match the file name!".*