EASY PATH FINDING SYSTEM

Basic manual

Overview

This system allows you to easily implement path finding ability to any game.
It's useful, as instance, for enemies to chase/seek for player or for some purposes where obstacles avoidance is needed.

The system is extremely easy to setup and tune - just in several clicks you'll implement it anywhere.
It really fast and performance safe (you can adjust it to get better accuracy or even higher performance).
The system doesn't require any specific actions or updates for your objects and scenes.
Moreover it allows avoiding even dynamic (unexpectedly appeared) objects/obstacles.

Please be aware: *unfortunately this system doesn't work perfectly yet (especially with complex geometries like terrain).*

This system works on all platforms supported by Unity3D.

How to use

To use this system – you should just:

1. Create game object (preferable with Rigidbody and Collider)
2. Assign *PathFinding* and *PathFollowing* components (scripts) to it
3. Assign any game object (or Transform) as Target in *PathFinding* component and assign this script to **pathFindingScript** property of *PathFollowing* component
4. System should works already, but you can tune/adjust parameters if you want.

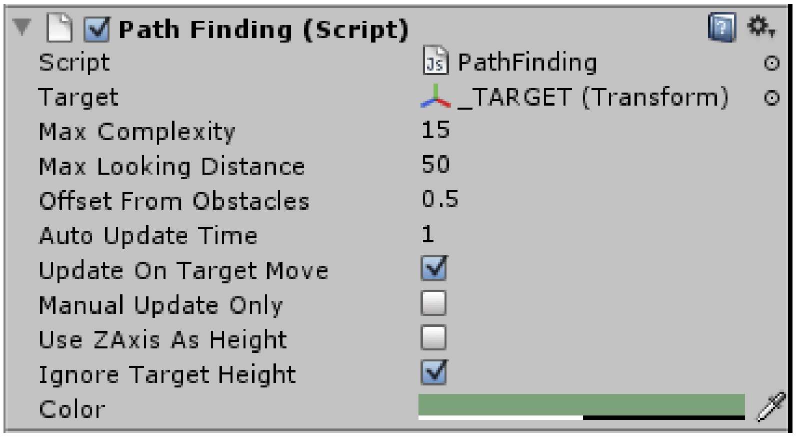
Please don't hesitate to contact me in any reason by mail: ***AllebiGames@gmail.com***

BASIC SCRIPTS DESCRIPTION

PathFinding script description

Main script of this Path finding system. Calculates (find) path automatically or according to specified rules Generates array of waypoints (around obstacles) until target will be reached

Average structure looks like:



☑ Path Finding (Script)

Script PathFinding

Target _TARGET (Transform)

Max Complexity 15

Max Looking Distance 50

Offset From Obstacles 0.5

Auto Update Time 1

Update On Target Move ☑

Manual Update Only □

Use ZAxis As Height □

Ignore Target Height ☑

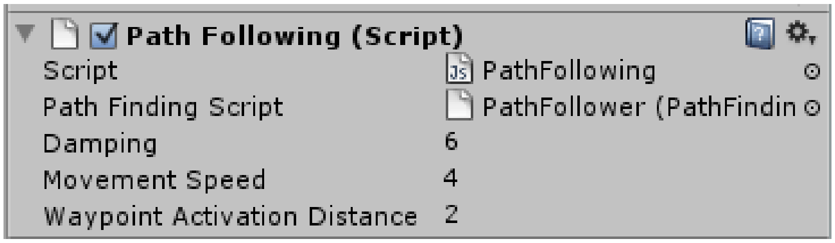
Color 🟩

- **target** - Target/final point to build path to
- **waypoints** - Array of generated waypoints
- **maxComplexity** - Max number of waypoins in path
- **maxLookingDistance** - Max distance of raycasts (looking for waypoint)
- **offsetFromObstacles** - Set additional offset between waypoints and obstacles
- **autoUpdateTime** - Delay to next path recalculation. Works automatically if updateOnTargetMove and manualUpdateOnly = false;
- **updateOnTargetMove** - Update only if new target position different from previous one
- **manualUpdateOnly** - Allow only manual updates by calling "FindPath" function
- **useZAxisAsHeight** - By default path calculates in XZ plane, set it to true to use XY plane
- **ignoreTargetHeight** - Ignore target Y (or Z) offset from this object
- **color** - Debug path-visualization color

PathFollowing script description

This is example script to follow path.
It manages waypointed path from pathFinding Script and move object along it.

Average structure looks like:



☑ **Path Following (Script)**

Script	PathFollowing
Path Finding Script	PathFollower (PathFindin
Damping	6
Movement Speed	4
Waypoint Activation Distance	2

- **pathFindingScript** - Path holder/generator script
- **damping** - Smooth facing/movement value
- **movementSpeed** - Speed of object movement along the path
- **waypointActivationDistance** - How far should object be to waypoint for its activation and choosing new
- **stuckDistance** - Max distance of move per regenTimeout that supposed to indicate stuck
- **stuckTimeout** - How fast should path be regenerated if player is stuck