SMOOTH TRAILS

User Documentation v1.4
1/1/2021

CONTENTS

WELCOME

Thanks for checking out **SmoothTrails**! Although there are numerous trail systems on the Asset Store, we think ours offers some unique features and usability that others do not. Both high performance and high quality smoothing are attainable with **SmoothTrails**.

If you like the asset, *please* do take a moment to write a positive review or give it a ★★★★★ rating. Be sure to recommend it to your friends. Building assets for the Unity developer community takes a lot of time and effort, so be sure to support the ones that make your life easier. Otherwise we won't be able to keep doing it for you.

The URL to leave a review is right here:

https://assetstore.unity.com/packages/slug/141291

FIRST STEPS (BORING, BUT RECOMMENDED)

This document is the best place to learn about the features and uses about this asset. Please read it thoroughly, as we want to make sure you get the most out of your investment.

1. Create a ***brand new project***. This may seem unnecessary, and while we go through great lengths to make sure there are no collisions with other assets, it is impossible to provide a guarantee for every combination of settings. Overall, this will give you the best possible experience to first figure out how to use the asset without introducing other variables.

2. Install **SmoothTrails** from the Asset Store.

3. Open the provided test scene. All our assets come with example data specifically for trying them in safe isolation. (Later, when you install it into your project, everything under the *testScenes* folder may be removed.)

4. Press Play. The mouse will control the camera, WASD will move the camera position, and backtick toggles the cursor on and off.

5. Select the ChaoticRotator object and expand its children.
    a. ChaoticRotator is rotating in fits and starts, as the name suggests, making for "jumpy" positions in the standard TrailRenderer on the UnitySword object. This is the blue trail.
    b. Notice the red trail is smoother? Under the SmoothTrailSword object, there is a Line Renderer driven by *SmoothTrails*.

6. Feel free to tweak the settings on the *SmoothTrails* component, as they all adjust dynamically (but are not intended to be modified at runtime, as there are visual artifacts).

7. Note also that there is an object called FrameRateKiller. It intentionally wastes time to force a low frame rate, so that the "jumpy" nature of ChaoticRotator is more pronounced. This is what users with lower end graphics cards will see with standard trail renderers, and it isn't pretty.

8. At this point, take a second to read through the description of the controls in more detail. Although it is relatively easy to use, the controls may not make sense without further explanation. The remaining sections will help you gain a deeper understanding of the effect shader and teach you how to use it effectively.

IMPORTING INTO YOUR PROJECT

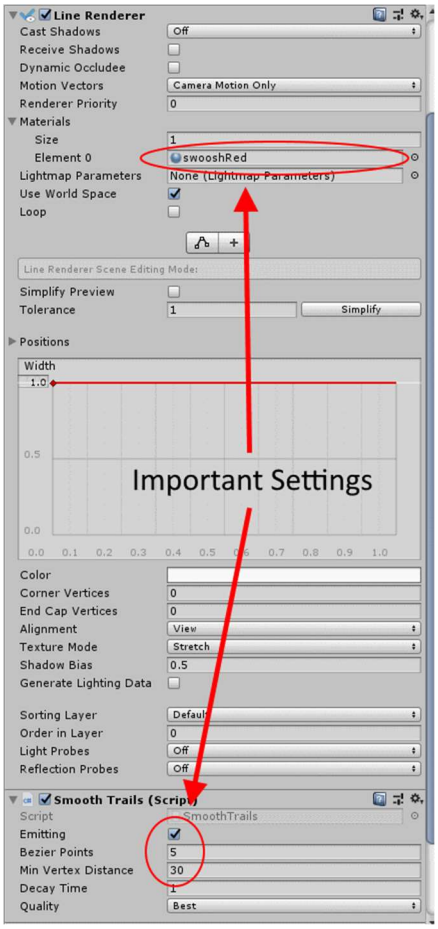
To import **SmoothTrails** into your project, please do the following:

1. Import **SmoothTrails** asset package

2. Select an object that you want to have a smooth trail following it

3. Add component… *SmoothTrails*
    a. This will automatically add a *LineRenderer* for you

4. Configure the *LineRenderer* so it is visible
    a. Choose a material for the trail
    b. Make sure Emitting is turned on
    c. Choose 0 or more Bezier Points
    d. Select an appropriate Min Vertex Distance, based on the speed the object will travel



FEATURES

The Unity *LineRenderer* and *TrailRenderer* are almost the same thing, except for a handful of properties. To make the *LineRenderer* work like a *TrailRenderer*, we force *Loop=off* and *WorldPosition=on*. **SmoothTrails** is essentially a reimplementation of the Unity *TrailRenderer*, except unlike the *TrailRenderer*, low frame rate or very fast movement does not result in low-grade, chunky trails. The performance is very close to that of a *TrailRenderer*, but slightly slower.

The *LineRenderer* is quite fast to render, but lacks important features. **SmoothTrails** gives you the high performance rendering of *LineRenderer*, plus the features of *TrailRenderer*, and helps take the jagged corners off trails so they look smoother by controlling the point set passed to the *LineRenderer* directly.

The devil is in the details, but suffice it to say that we dynamically recalculate the points nearest the emitter as if they were on a cubic Bezier spline. When points are sufficiently old that they are no longer changing, they lock into place until they decay out.

BEZIER POINTS

This is the number of additional points added through smoothing calculations. If you set this to 0, it is strictly a LineRenderer. Be careful of setting this number too high, as it can add a lot of points to your trails. A good starting point is a number between 3 and 7.

MIN VERTEX DISTANCE

Same as with a *TrailRenderer*, this is the distance the emitter must travel before it generates a new point in the trail. Until a new point is added, the "head" of the tail is constantly updating to the current position of the emitter. This keeps the trail connected to the emitter without showing any gaps, and also explains why the curve nearest the emitter seems to shift sometimes. That's because it is recomputing the Bezier curve segment every time the emitter moves.

DECAY TIME

Same as with a *TrailRenderer*, this is how long each point will take to fade out with alpha.

QUALITY

The choices are: Good, Better, Best. The quality increase is essentially how many segments away from the "head" are going to be recomputed. Best quality will recompute all three segments of the curve as the emitter moves. Better quality will recompute the two segments nearest the emitter. Good quality will only recompute the segment between the emitter and its most recently emitted vertex.

ALL SOURCE IS INCLUDED

If you want to change the way it works, knock yourself out! It's a great starting point for building your own curved trail system—provided you don't sell it or release it to the asset store, of course.

SUPPORT

Please read all the documentation. We put a lot of effort into it, and hope that it exceeds your expectations. In the event you have further questions, please check the following web pages for more details about this asset:

Our Website: https://reachablegames.com/unity-assets/

Unity Forum: https://forum.unity.com/threads/640408/

If you find that none of the above can answer your question, you may contact us at support@reachablegames.com, but know that we always handle support requests first that include:

1. **Invoice Number**
2. **Unity Version**
3. **Asset Version**
4. Links to screenshots or small sample projects on DropBox or similar sharing site describing the problem.
5. Kindness. If you are mean, rude, harassing, or hateful, do not expect a response.

As a matter of common sense, we do not offer support to free customers except as time permits.

Finally, if you are looking for a feature that is not currently supported, understand that we are a business and get many such requests. If you need a custom feature that is important enough to your product to pay for its development, contact us about it.

ABOUT US

Reachable Games is located in the beautiful hill country of Austin, Texas USA. It was founded by Jason Hughes, who has been a professional game developer since 1995, at Origin Systems. He has worked on many AAA games and with many recognizable companies. As a generalist, Hughes has worked on nearly every kind of game platform in every capacity—from graphics tools to AI to UI/UX to game design to shader writing to database management to networking and server development. It turns out this is the right skillset to help improve the Unity development experience for other developers.

We currently have several other assets on the Asset Store. For what it's worth, they are all built because we are working on projects of our own and both need and use them on a daily basis. If you think this one is great, chances are the others will help you out too.

RELEASE NOTES

TODO

1. Add a compute shader with more features?

V1.4 JANUARY 1, 2021

- Revised the error handling for UnityWebRequest now that the interface changed in 2020.2.

V1.3 DECEMBER 12, 2020

- Simplified version checking popup.
- Split out the code that handles the spline from the logic surrounding the sampling. This allows you to pump points in from somewhere else, if you like. One user was trying to do instant replays, and this change would have made doing so easier.

V1.2 JUNE 2, 2020

- Worked around an editor crash due to a bug in Unity 2019.3.
- Namespaced all the code.
- Backported package to run in any version of Unity from 2017.4LTS forward.

V1.1

- Added documentation
- Rearranged folder structure to avoid file conflicts with other assets from Reachable Games

V1.0

- Initial Release