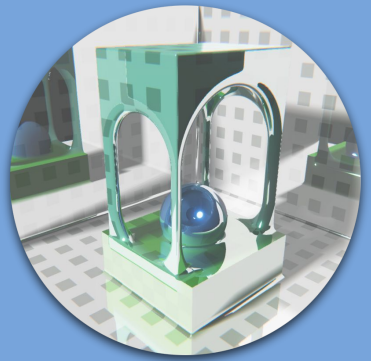
Raymarcher v2.1.0

Official Documentation



Official Documentation Video

Official Overview Video

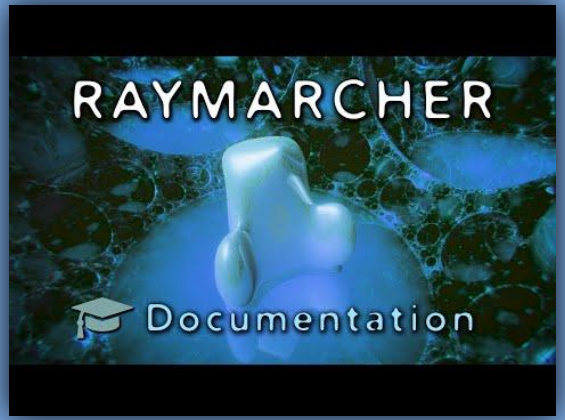
Change-log v2.1.0

15.06.2021 (dd/mm/yyyy)

- Added mobile support (iOS, Android)
- Added mobile-VR support (Oculus Quest, GO...)
- Added WebGL support (High-End, Low-End)
- Major rendering optimizations
- Added toon shading
- Added lambert shading
- Added outline shading
- Added new collection of example scenes
- Fixed editor UI minors
- Updated UI & icons
- Major source-code refactor
- Unity 2019, 2020 & 2021 support

Older media

Raymarcher documentation 1.0.0 (2020)



Raymarcher first VR overview 1.1.0 (2020)

Raymarcher overview 1.0.0 (2020)

Still related to the actual version

Content

01 Introduction
02 Brief Features
03 Supported Platforms
04 Set Up (in general)

05 Platform Specifications

06 PC/Consoles
07 PC-VR

08 Mobiles
09 Mobiles-VR

10 WebGL

Rendering Options 11

Lambert Shading 12
Flat Shading 13
Toon Shading 14
Outline 15

Objects 16
Fractals 17
Operations 18
API 19

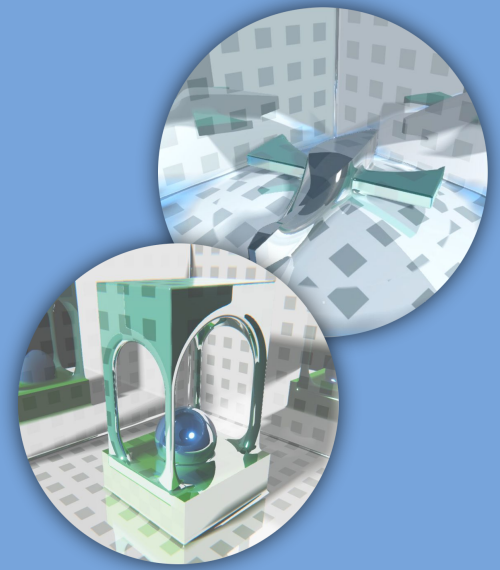
Tips & Tricks 20
Warnings 21
FAQ 22
Example Content 23
Commercial Products 24

Introduction

Raymarcher is a complete package of rendering technique called Raymarching. Huge and modular package that offers user-friendly tools to create unique scenes full of fractals and organic objects in just a few seconds.

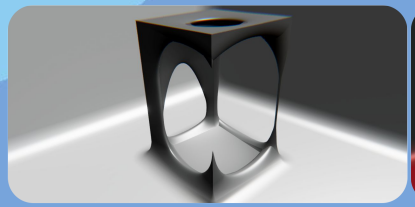
I started working on this package 3 years ago as I was very inspired by raymarching rendering technique and I could see great potential in it. So I did some research about it's functionality, roots and compatibility. I found many great resources about this algorithm and in general, it wasn't really that hard to implement the single raymarcher renderer. The hardest part was the 'convertor' that makes all the 'hard writing' instead of you. It took me some time writing custom convertor that translates user interaction (creating objects, operations etc) into shader language (at runtime).

The general feature of the package is 'runtime-change' which means, that once you set up the scene, you can *create/destroy/change/operate* any raymarcher object at runtime without touching the shader language or even the code itself. This makes the package accessible almost for every Unity user without any further knowledge and opens new opportunities to the realtime rendering. The package is ready for beginners & non-programmers.

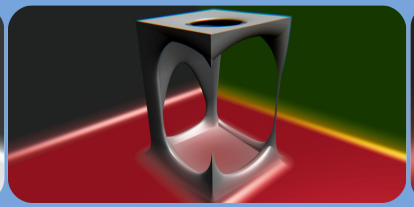


Brief Features

Simple Renderer



Colored Materials

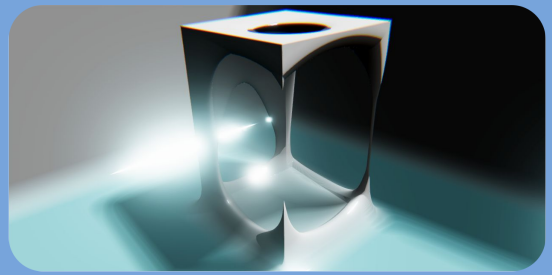


Global Smoothness

Flat, Toon & Outline Shading

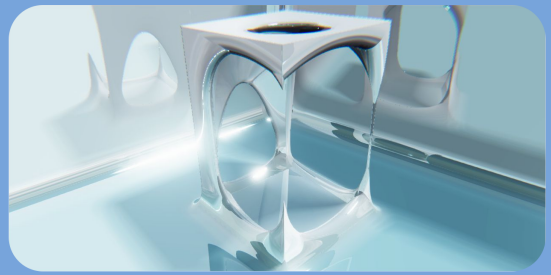
Lambert Shading Model

- Blinn-Phong specular
- Hard/Soft shadows
- Custom Normal jitter



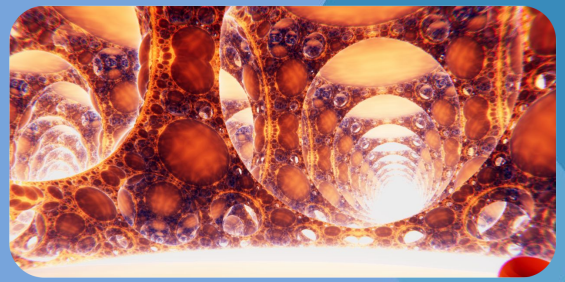
Reflections & PBR

- Cubemap reflection
- Physically based reflections
- Custom reflection samples



Fractals, Operations and more...

- Custom shapes
- Modular operations
- VR, Mobile & WebGL support



Supported Platforms

Raymarcher supports all possible platforms in Unity engine. However, each platform has strict & specific limitations due to the heavy rendering calculations. More about each platform in next slides.

OSX

iOS

Windows

android

Linux

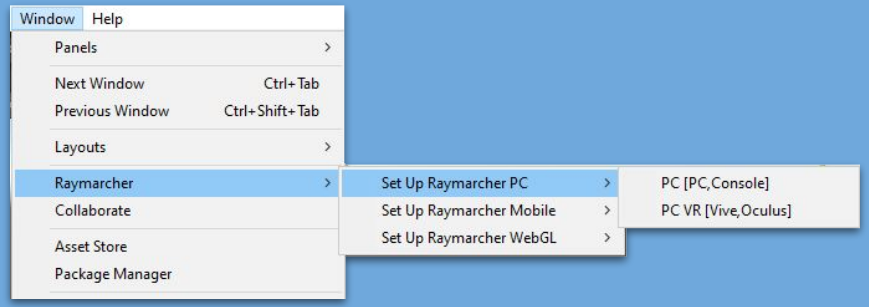
WebGL

oculus quest

VIVE

Set Up (in general)

It's very simple to set up raymarcher renderer in your scene. Just go to ***Windows/Raymarcher/*** and choose your desired platform.



This will prepare your whole scene for raymarching features. All you need is at least one Main Camera tagged as 'MainCamera'.

The following slides will teach you specific features and limitations for each platform setup.

The 'Set Up' process will create **3 essential** components that manage whole system.

- **RM_Master**
  Main system that handles rendering & generic values
- **RM_Mapping Master**
  System that handles objects & scene
- **RM_XConvertor**
  System that handles shader conversion and all the 'magic'

If you create any raymarching object, the object will contain **RM_Object** component.

Platform Specifications

Raymarcher supports all possible platforms. The following slides contain brief information about each platform's features & limitations. Also each slide contains panel with features that mostly differ in other platforms. All other features (that are not mentioned in the panel) are available in all platforms.

Does target platform supports realtime shadows in raymarching?
(Would that be possible to enable realtime shadows with soft/hard feature to raymarching objects?)

Does target platform supports outline feature in raymarching?
(Would that be possible to use outline feature to raymarcher objects?)

Does target platform supports realtime reflections in raymarching?
(Would that be possible to enable physically-based reflections?)

Does target platform supports global texture in raymarching?
(Would it be possible to apply global texture to the raymarching renderer?)

Does target platform supports depth-buffer in raymarching?
(Generic Unity objects will react with Raymarcher objects)

Does target platform supports rotation & re-scaling for raymarcher objects?
(Would it be possible to rotate & rescale raymarching objects in editor/ at realtime?)

Does target platform supports colored material for raymarcher objects?
(Would it be possible to change color for each raymarcher object? Mobiles use ramp textures)

Does target platform supports fragment operation for raymarcher objects?
(Would it be possible to enable internal operation - fragment for each raymarcher object?)

Global Rendering
Shadows
Outline
Reflections
Texture
Depth Buffer
Object Transform
Rotation & Scale
Color
Fragment Operation

Raymarcher - PC/Consoles

Windows

OSX

Linux

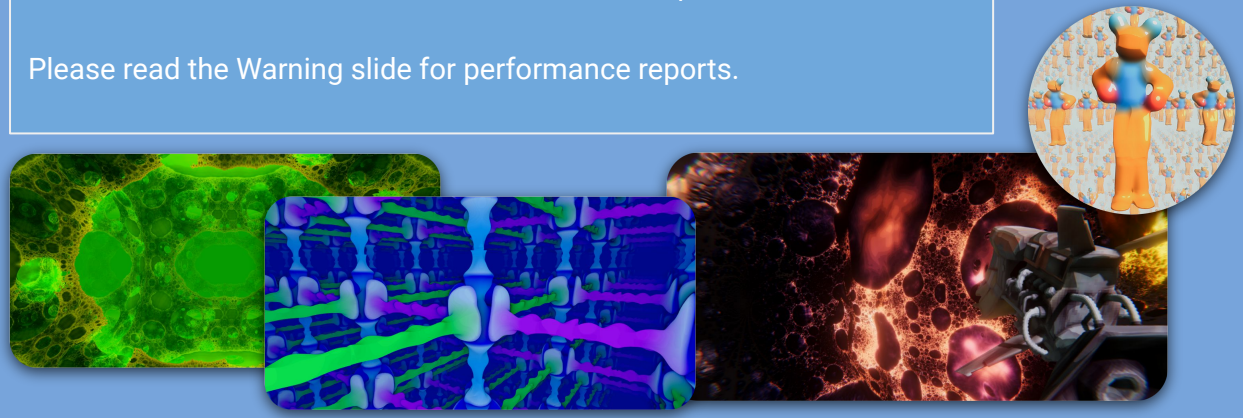
Raymarcher was initially built for high-end PCs with full feature list. However as the mobile support came, the feature list had split to optimize rendering performance. Please read the specified feature list on the right side of the slide.

To set up the raymarcher for PC/Consoles, choose: ***Windows/Raymarcher/Raymarcher Set Up PC/PC [PC,Console]***.

The specified feature list on the right tells you what are the main differences between mobile raymarcher and PC raymarcher. It's because the less data shader contains, the faster GPU computes.

Please read the Warning slide for performance reports.

Global Rendering	
Shadows	Yes
Outline	No
Reflections	Yes
Texture	Yes
Depth Buffer	Yes
Object Transform	
Rotation & Scale	Yes
Color	Yes
Fragment Operation	Yes



Create advanced objects with organic behaviour, connect external operations and use breathtaking fractals. Upgrade your project to the next level.

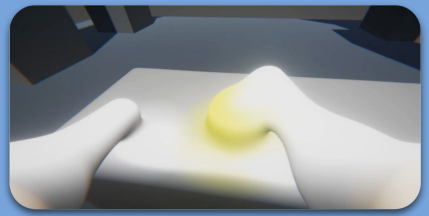
Raymarcher - PC VR

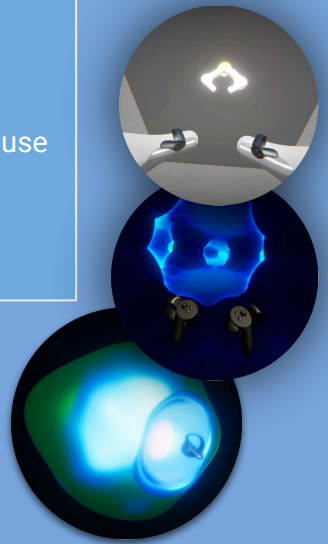
Raymarcher for PC-VR works just like for PC non-VR. However, there are a few differences: the performance is 2x higher as the renderer calculates 2 eyes. It's recommended to own RTX series to run raymarcher with massive scenes smoothly. Also raymarcher for VR doesn't contain Depth Buffer, which means the generic Unity objects will not react with raymarcher objects.

To set up the raymarcher for PC VR, choose:
Windows/Raymarcher/Raymarcher Set Up PC/PC VR [Vive,Oculus].

The specified feature list on the right tells you what are the main differences between mobile raymarcher and PC raymarcher. It's because the less data shader contains, the faster GPU computes.

Please read the Warning slide for performance reports.





Global Rendering	
Shadows	Yes
Outline	No
Reflections	Yes
Texture	Yes
Depth Buffer	No
Object Transform	
Rotation & Scale	Yes
Color	Yes
Fragment Operation	Yes

Create advanced objects with organic behaviour, connect external operations and use incredible fractals in VR. Upgrade your app to the next level.

Raymarcher - Mobiles

Raymarcher for mobile is a little bit different and contains less features than Raymarcher for PC. It's due to heavy rendering calculations as the raymarcher is a small 'brother' of ray tracing. Please read the specified feature list on the right side of the slide.

To set up the raymarcher for mobile, choose:
Windows/Raymarcher/Raymarcher Set Up Mobile/Mobile [iOS,Android].

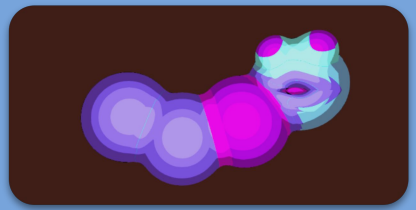
The specified feature list on the right tells you what are the main differences between default raymarcher (for PC) and mobile raymarcher. It's because the less data shader contains, the faster GPU computes. Mobiles are not very performant, but technology advances very quickly and realtime renderers (like this) may become a norm in the future.

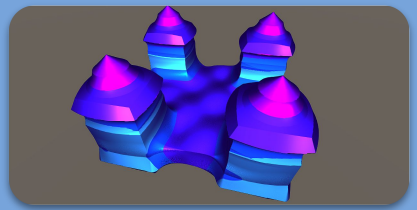
Please read the Warning slide for performance reports.

iOS

android

Global Rendering	
Shadows	No
Outline	Yes
Reflections	No
Texture	No
Depth Buffer	Yes
Object Transform	
Rotation & Scale	No
Color	Coords only
Fragment Operation	Global only





Create simple objects with organic behaviour, connect external operations and use optimized fractals.

Raymarcher - Mobiles VR

oculus quest

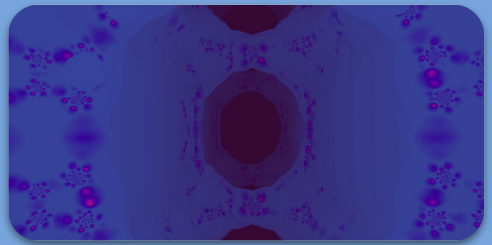
pico neo2

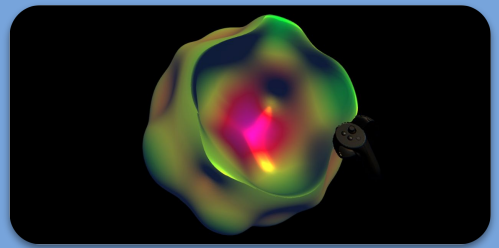
Raymarcher for mobile in VR has the same limitations as raymarcher for mobile non-VR, except depth buffer - generic Unity objects will not react with raymarcher objects. The Raymarcher for mobile in VR is specially designed for mobile-based platforms such as Oculus Quest or Pico Neo. Please read the specified feature list on the right side of the slide.

To set up the raymarcher for mobile-VR, choose:
Windows/Raymarcher/Raymarcher Set Up Mobile/Mobile VR [Oculus Go,Oculus Quest].

The specified feature list on the right tells you what are the main differences between default raymarcher (for PC) and mobile raymarcher. It's because the less data shader contains, the faster GPU computes.

You are very free to try the example content in Web here!





Global Rendering	
Shadows	No
Outline	Yes
Reflections	No
Texture	No
Depth Buffer	No
Object Transform	
Rotation & Scale	No
Color	Coords only
Fragment Operation	Global only

Create simple objects with organic behaviour, connect external operations and use optimized fractal trips in VR.

Raymarcher - WebGL

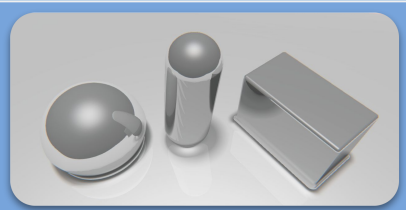
Raymarcher for WebGL allows you to choose between two types - High-End and Low-End. High-End contains all the essential raymarcher features, it is equal to the PC-version. On the other hand, the Low-End is equal to the mobile version. Use High-End if you focus on quality, use Low-End if you focus on performance. Please read the specified feature list on the right side of the slide.

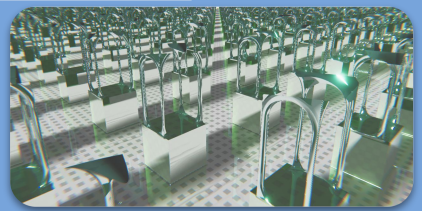
To set up the raymarcher for WebGL, choose:
Windows/Raymarcher/Raymarcher Set Up WebGL

The specified feature list on the right tells you what are the main differences between default raymarcher (for PC) and mobile raymarcher. It's because the less data shader contains, the faster GPU computes.

You are very free to try the example content in Web here!

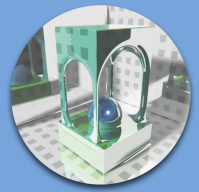
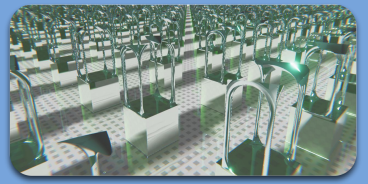
Global Rendering	High/Low-end
Shadows	Yes/No
Outline	No/Yes
Reflections	Yes/No
Texture	Yes/No
Depth Buffer	Yes
Object Transform	
Rotation & Scale	Yes/No
Color	Yes/Coords
Fragment Operation	Yes/Global





Create simple/advanced objects with organic behaviour, connect external operations and use fractal trips. Build and play your application directly in browser!

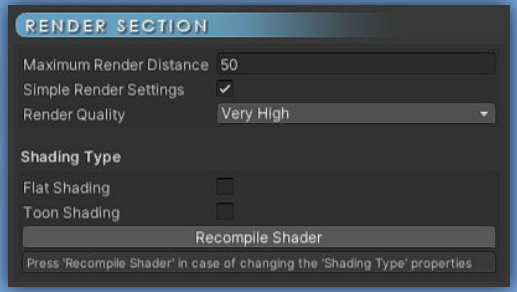
Rendering Options

Raymarcher package contains many options of rendering and CG visualizing. The main Master component (which handles all the rendering work) is divided into 4 'Sections': **Render Section**, **Visual Section**, **Shading Section**, **Operations**. The default options give you the most realistic results. There are 3 shading types available: **Flat shading**, **Lambert shading** (Default) & **Toon shading**.

Maximum render distance <
(The higher the value, the more performance)
Render quality <
(The higher the quality, the more performance)

Various shading types <
(Options differ for each target platform)



Main global color & additional emission <

Additional-triplanar texture & texture tiling <

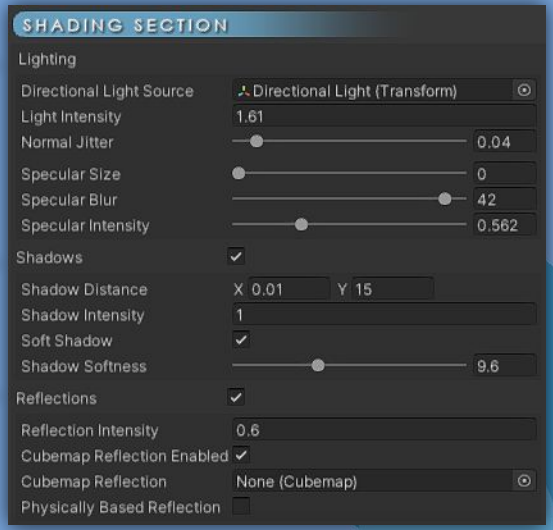
Distance fog with custom density & color <

Global object & color smoothness <
(The higher the value, the more organically will objects behave)

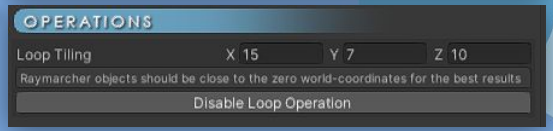
Lighting settings <

Shadow settings <

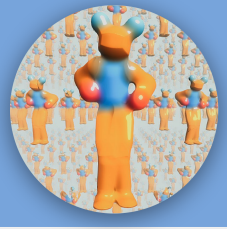
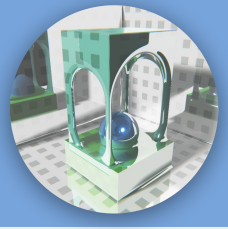
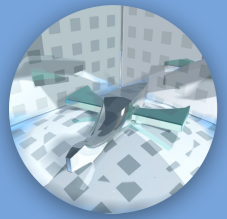
Reflection settings <



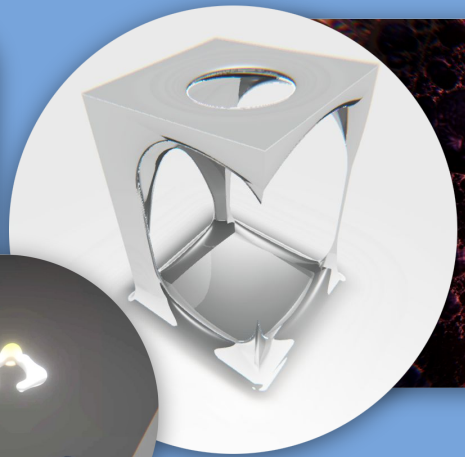
Global operation - Loop <



Lambert Shading

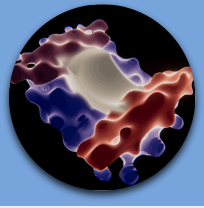
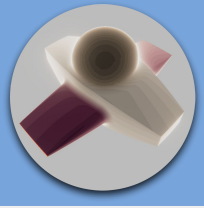




Lambert Shading is a default shading in Raymarcher which evaluates fully shaded image with specular highlights, shadows and reflections. The shading section for lambert shading is already described in the previous slide (which opens defaultly). If you are targeting to mobile platform, lambert shading is an option as it takes less performance. Create astonishing scenes with realtime shadows (soft/hard) and realtime reflections (with custom sample count) in the highest & unique rendering quality. Explore new possibilities in another way, in another sub-category of ray tracing = in raymarching.

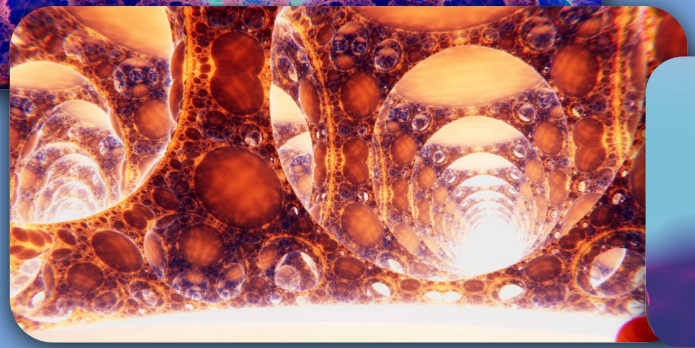
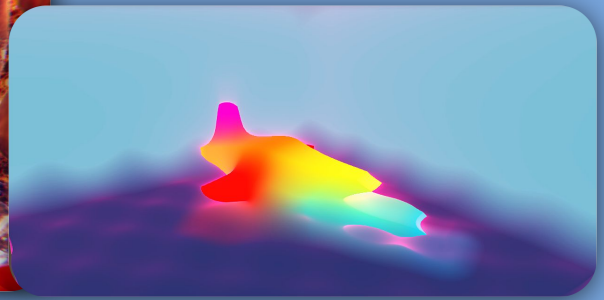



Great for realistic & unique scenes!

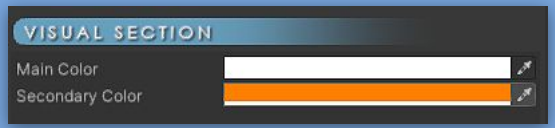
Flat Shading


Flat Shading evaluates plain image without shadows or additional, specular highlights. If you enable flat shading in **Shading Section**, you will get a few more options which allow you to change its look and functionality. If you are targeting to mobile platform, the flat shading is enabled by default (as it takes less performance than lambert shading).

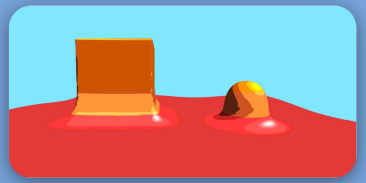
General color of flat shading <
Secondary color of flat shading <
(Mostly comes from the edges of objects)

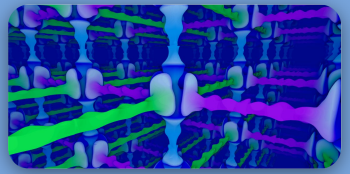


Fresnel Density <
(How much outer color affects the inner color)
Fresnel Multiplier <
(Overall fresnel multiplier)

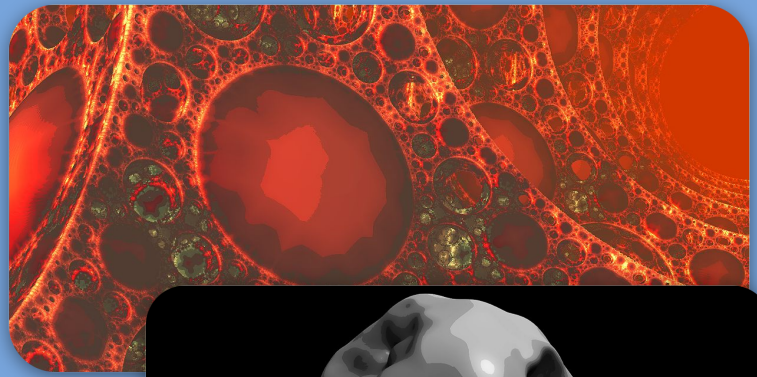
Great for fractal trips & self-lit objects!

Toon Shading

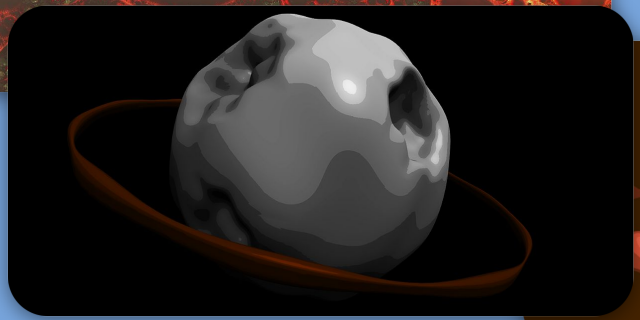


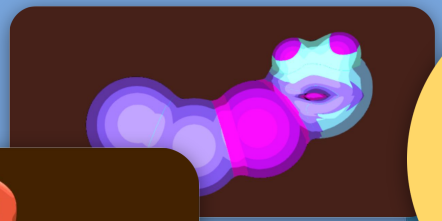


Toon Shading evaluates compressed image with cascading highlights. The image remains the same until '**Toon Threshold**' & '**Toon Density**' are increased. These two essential parameters change the final image with cascading highlights & additional, global emission. The results are basically similar to stylized/cartoon movies or posterized colors.



Toon density <
(Toon intensity & additional emission multiplier)
Toon Threshold <
(Amount of highlight cascades)

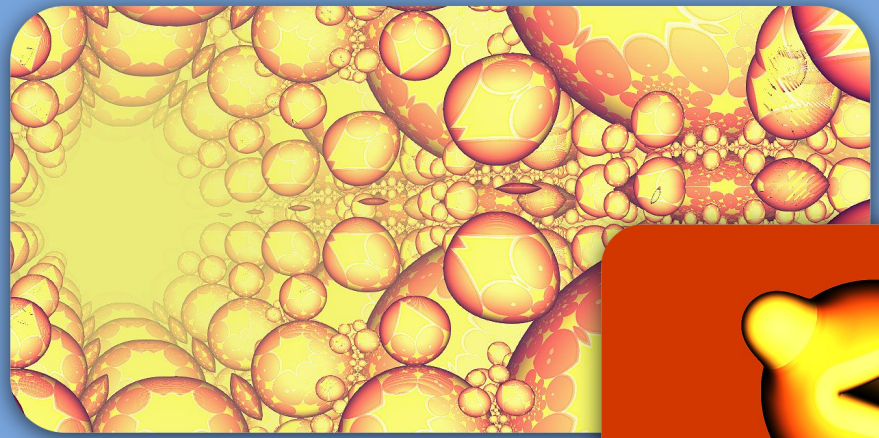




Great for cartoon characters!

Outline (mobile only)

Outline is an additional feature for mobile platforms only. The outline adds outer, soft color for each raymarcher object. The outer color can be customized as well as the softness. The outline feature is available for mobile platforms only (yet).



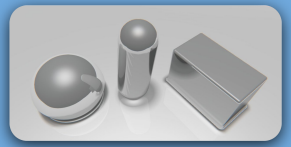
Outline Density <
(Outline spread intensit)
Outline Softness <
(Possible to use reversed softness)

Great for stylized sceneries!

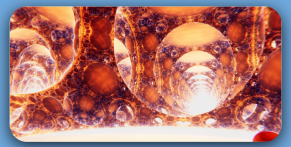
Objects

Raymarcher allows you to create 3 types of objects:

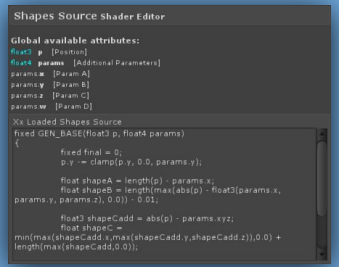
- ***Built-in Shapes***



- ***Fractals***

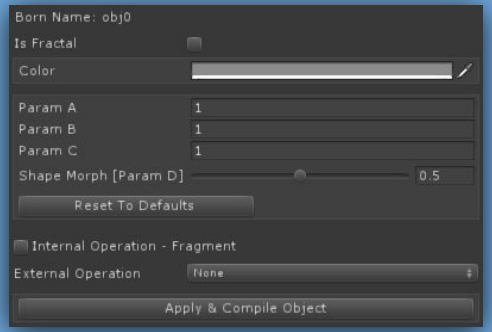


- ***Custom Shapes***



Built-In Shapes

Each raymarcher object contains **RM_Object** component which contains ***'Is Fractal'*** *option*, **4 parameters** (A,B,C and D), **internal operations** (Fragment) and **external operations** (Subtraction & Intersection with other objects). Each parameter corresponds to the specific object value such as size, length, depth and morph. Built-in shapes are **Sphere**, **Capsule** and **Box**. All these three shapes can be morpher and mixed together as well.



Fractals

Fractals change the single object to the complete fractal formula. Each fractal option contains specific parameter to control fractal size, approach, bend, color mix etc. More about this in **Fractals** slide...

Custom Shapes

You can also customize your own shapes in **Shape Editor** if you know correct formulas. More about this in **API** slide...

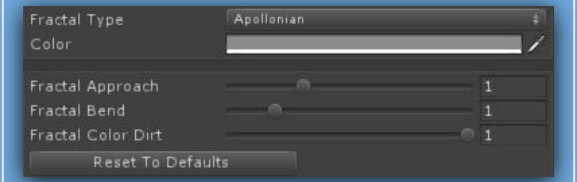
If **'Is Fractal'** or any **'External Operation'** has some changes, the 're-compilation' must be proceeded. It can be done by pressing the **Apply & Compile Object** button. Otherwise there won't be any changes.

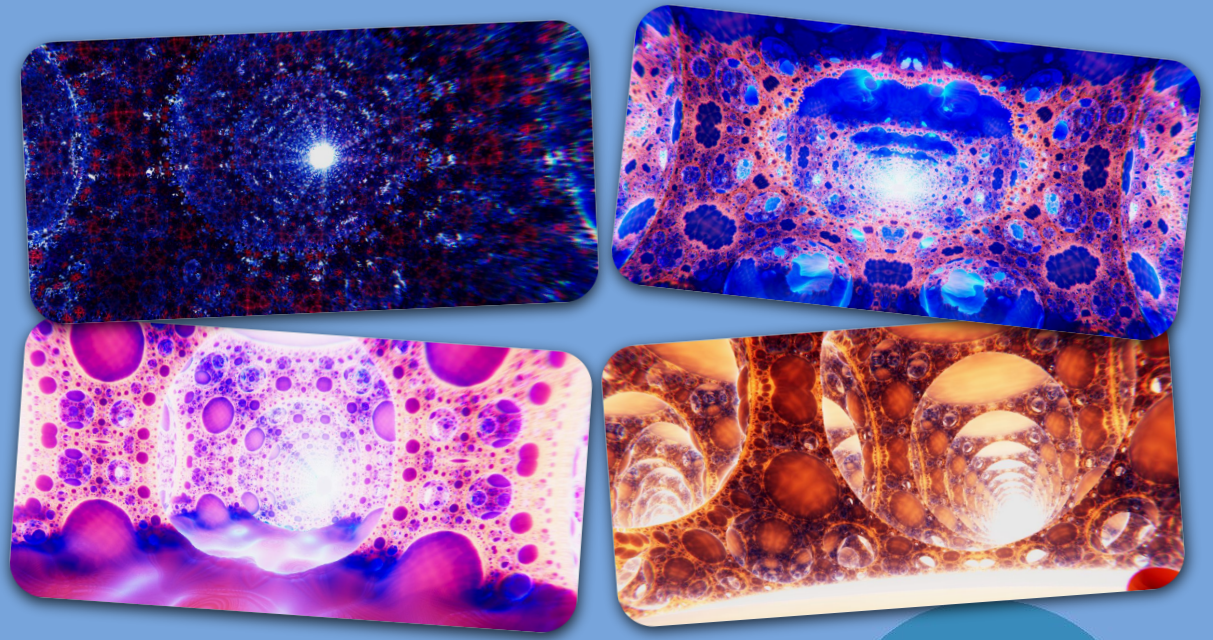
Fractals

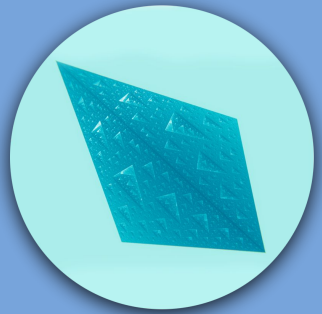
Raymarcher allows you to create astonishing worlds with fractals. Actual version contains 4 fractals:

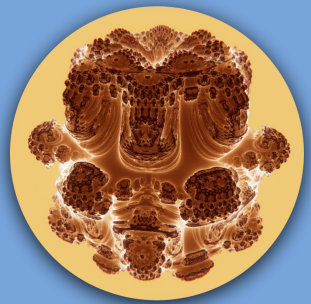
1. ***Apollonian***
2. ***Kleinian***
3. ***Mandelbulb***
4. ***Tetrahedron***

Each fractal contains clear parameters of its size, approach, burst, color mix and more. Create as many fractals as your GPU can take and play with infinite potential!









Even better in VR!

Operations

Raymarcher allows you to make advanced operations between raymarcher objects. There are 3 types of operations:

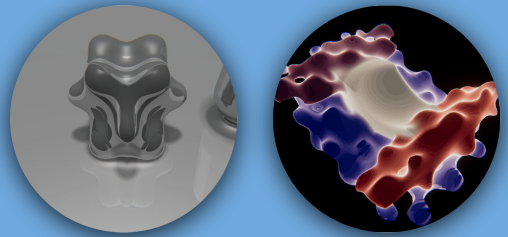
1. ***Master Operation [Loop]***
2. ***Internal Operations***
3. ***External Operations***

As it's mentioned in the previous slides, the **RM_Master** component contains its own 'main' operation called **Loop** which will loop all your raymarcher object in specific tiling.

Internal & **external** operations can be found on every object with **RM_Object** component.

Internal Operations

There is just one internal operation called **Fragment**. Fragment makes your **RM_Object** look 'blobby-like'. You can set the fragment evolution, size and even animation. Results are interesting!



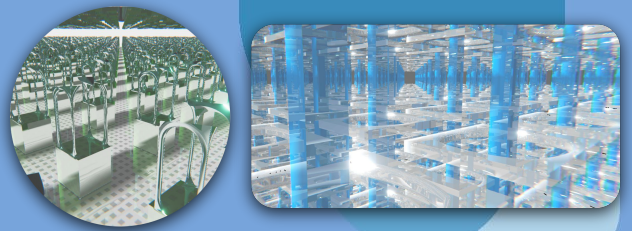
Master Operation - Loop

Master loop operation allows you to **loop** all raymarcher objects in your scene with specific tiling. The results are really interesting ->

External Operations

External operations require 'second operand'. External operations allow you to **Subtract** or **Intersect** with specific object. *Object is able to have just <u>one</u> external operation!* Any changes made in external operations must be refreshed by pressing the **'Compile'** button.

Subtraction with fractal



API

Raymarcher package contains public API with custom classes and functions that can be used in any case.

The following API will be described from these scripts:

- **RM_Master**
- **RM_XConvertor**
- **RM_Mapping Master**
- **RM_Object**

Global namespace: ***RaymarcherPackage***
Scripts Folder: ***/Core***
Shaders Folder: ***/Core/CG***

General Description

RM_Master is a general and main system of the whole package. It manages rendering, visualization, shading and operations. The script is highly required for the other components to make the raymarcher work as intended.

The **RM_Master** inspector content is described in ***Rendering Options*** slide.

API Description

```
public void MASrenderDistance(Slider v/ float v)
- Change render distance (by float or UI slider)
public void MASrenderQuality(Slider v/ float v/ int v)
- Change render quality (by integer, float or UI slider)
public void MASchangeGlobalSmoothness(Slider v/ float v)
- Change global object smoothness transition (float or UI slider)
public void MasterRewriteMapping(bool clearAllObjects = true)
- Rewrite & recompile current mapping of the raymarcher - Editor Only
public void MasterRecompileMaster(string currentSceneName)
- Recompile & rewrite current master shader of the raymarcher - Editor Only
```

API

General Description

RM_Mapping Master is one of the essential components that manages all existing raymarching objects in scene. The system refreshes all their values & transforms.

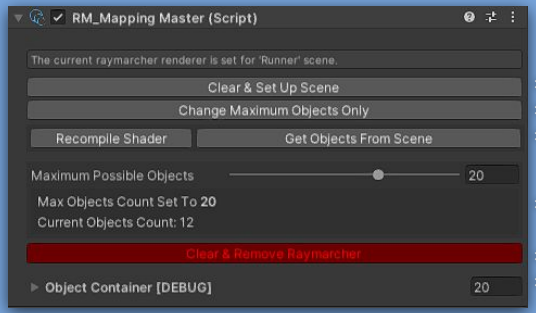
Scene set up, scene refresh and objects conversion from scene to shader happens here in this system.

Mapping Master contains a few options that allow you to control your scene. *If you change the **Maximum Possible Objects** value, it's required to recompile the shader (warning message will appear).*

API Description

public static GameObject MAPcreateRaymarcherObject()
- Create raymarcher object - returns complete raymarcher object
public void MAPaddObject(RM_Object sender)
- Add new raymarcher object to the mapping master database 'manually'
public void MAPrefreshContainer()
- Refresh raymarcher objects to master shader
public void MAPclearAllObjects(bool includeContainerElement = true)
- Clear all raymarcher objects manually. If included parameter is true, the whole list will be cleared (otherwise the list's values will be cleared and list count will remain)
public void MAPrefreshRaymarcherScene()
- Refresh current raymarching scene and sum-up all available RM Objects

Inspector Description



> Setup scene again (clears whole raymarcher with all its objects)
> Change maximum object count only (recompile shader & set max obj count)
> Recompile shader only & Get raymarcher objects from scene

> Maximum possible objects & currently created raymarcher objects

> Clear current raymarcher renderer with all its files & shader sources
> Current object container (as debug - in case of advanced changes)

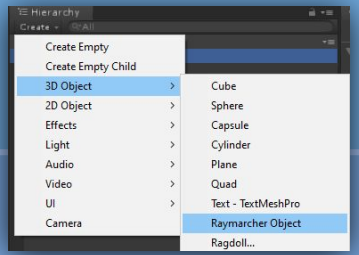
API

General Description

RM_Object is a required component for every raymarching object. The script contains morpher feature, shape generator, fractals and additional parameters marked as A,B,C and D.

Each raymarching object has a unique ID and 'Born Name' which are defined right after their creation.

To create any raymarcher object, go to ***Hierarchy/Create/3D Object/Raymarcher Object***.



API Description

public void ObjRefreshDefaultSize(int preset, bool isFract)
- Refresh object to the default settings by specific preset (preset 4 is default)

public void ObjchangeParamA(float v/ Slider v)
- Change value of the parameter A (by UI slider or Float)
public void ObjchangeParamB(float v/ Slider v)
- Change value of the parameter B (by UI slider or Float)
public void ObjchangeParamC(float v/ Slider v)
- Change value of the parameter C (by UI slider or Float)
public void ObjchangeParamD(float v/ Slider v)
- Change value of the parameter D (by UI slider or Float)

public void ObjchangeSmoothness(float v/ Slider v)
- Change value of objects smoothness (by UI slider or Float)

public void ObjOP_Fragment(bool v/ Togglev)
- Enable/ Disable fragment effect at runtime (by UI toggle or Bool)
public void ObjOP_changeFragmentSize(float v/ Slider v)
- Change fragment size (by UI slider or Float)
public void ObjOP_changeFragmentEvol(float v/ Slider v)
- Change fragment evolution (by UI slider or Float)
public void ObjOP_FragmentAnim(bool v/ Toggle v)
- Enable/ Disable fragment animation effect (by UI toggle or Bool)
public void ObjOP_changeFragmentDir(float v/ Slider v)
- Change fragment animation direction (by UI slider or Float)

API

General Description

RM_XConvertor is one of the essential systems that converts user interaction into shader language [Specifically to the ObjectBuffer.cginc]. Once the user hit **Refresh** or **Recompile button**, the **XConvertor** will look for all raymarcher objects and re-calculate their values. After all the required processes, the code will be written into **ObjectBuffer** file which the **core shader** uses as a source of all raymarching shapes. Each scene has its own & unique **ObjectBuffer** file which is stored in the **Core** directory of the **Raymarcher** system.

XConvertor doesn't contain any practical API as it is not necessary for developer. It manages overall compilation from user language to the shader code.

Inspector Description

'Object Buffer' editor

• *Object Buffer is a shader source file that contains all created raymarching objects in your current scene [each scene has its unique shader instance]. You can check the source code anytime. Changing the source code is not recommended, because the XConvertor manages its data on any change.*

Shapes Source editor

• *Shapes Source is a shader source file that contains general shapes that raymarching objects use. For example the built-in shapes [Sphere, Capsule & Box] are written here. You can change it any time or even make a custom shape formula.*

Tips & Tricks

UI Events

Raymarcher allows you to create very quick interaction between user and RM_Object at runtime by changing its size, length, height, color and more. This can be easily achieved by creating simple UI Elements and assigning them to the RM_Objects.

1. Create any UI Slider or UI Toggle
2. Create a new event and assign any RM_Object
3. Select any value you would like to change [for example **ParamA**]
4. Done!

Or you can check this example in the official documentation here.

Subtraction

Raymarcher allows you to create interesting operations such as Intersection or Subtraction. Both operations require two operands - Object A and Object B. Once the object is affected by the operation, it can't be apply another operation. The operations can't be changed at runtime!

1. Create two raymarcher objects
2. Select one of the raymarcher objects and set the external operation to Subtract. [And hit Compile]
3. Assign the first object to the Target.
4. Done!

Or you can check this example in the official documentation here.

Fractal Planet

Now let's create something more interesting. A fractal that will affect a single object. But the fractal will contain subtract operation! Fractals can be very powerful in many cases and their potential is endless. You can choose any fractal, in this example we will use Apollonian.

1. Create two raymarcher objects
2. Select one of the raymarcher objects and set the external operation to Subtract and enable Is Fractal. [And hit Compile]
3. Assign the first object to the Target.
4. Done! Now play with parameters.

Or you can check this example in the official documentation here.

Warnings

Technical Warnings

In general

- Raymarcher is not compatible with **URP** or **HDRP** render pipelines. **Standard** only.

- Raymarcher was tested in
  • **Unity 2018.4.31f1**
  • **Unity 2019.4.23f1**
  • **Unity 2020.3.2f1** and
  • **Unity 2021.1.9f1**.
  Although it's recommended to use LTS versions.

PC/Consoles/ PC-VR
Raymarcher was tested on GPU Nvidia RTX 2070 & Nintendo Switch

Mobile
Raymarcher was tested on **iPhone XR** (iOS) and **Samsung Galaxy A51** (Android). It's highly recommended to use latest models as the Raymarcher is GPU-very heavy

Mobile-VR
Raymarcher was tested on Oculus Quest 2

Package Warnings

The current package version contains some restrictions and limitations. Please read them carefully below.

1. **Maximum raymarcher** object count is set to **30** due to the performance. You can increase this value on your own risk inside the code.

2. In some cases the raymarcher **might not work properly** due to **NaN** values or **compilation** errors (Depends on GPU type). Try to recompile the shader & refresh scene.

3. Raymarcher **doesn't work** with **transparency & transparent** materials. Objects with **Transparent** shaders will be **behind** the raymarching objects.

4. Raymarcher **doesn't support two** or **more light sources**. Currently there is just **one** light source - **directional light**. This is still in development.

5. VR raymarcher **doesn't work** with **Unity's default geometry** - it doesn't contain **depth buffer**. The VR raymarcher will be always above the default geometry.

FAQ

- **What computer do I need to run Raymarcher?**
  Raymarcher is a subcategory of ray tracing and uses similar algorithm, which requires quite good performance. But in very small scenes and low-quality rendering results there is no need for high-end computer. Try the example scenes & verified download content!

- **Does the Post Processing Stack work with Raymarcher?**
  Yes, it does work. But I highly recommend to use PPS v1 as I found it much more effective and less problematic. However the Ambient Occlusion doesn't work!

- **Do the Unity regular geometry interact with Raymarcher?**
  Yes, they do! The raymarcher contains depth camera which checks if the ray hits any regular geometry. On the other hand, if you are targeting to VR, the raymarcher will ignore all the other objects and will be prioritized in render queue.

- **Are there any plans to update Raymarcher in the future?**
  Yes, there are. I would like to update VR support, add points lights support and update overall raymarcher workflow.

- **Does the Raymarcher reflections work with regular objects?**
  No, it doesn't work. The Raymarcher objects are created in another renderer and use other rendering techniques.

- **Does the Raymarcher objects receive shadows from regular objects?**
  No, they don't as they use another rendering technique in another pass.

- **Does the Raymarcher works with Forward rendering path?**
  Yes, Raymarcher works with both Deferred & Forward. However if you are targeting to mobile platform, the Deferred rendering path is the only supported.

- **Does the Raymarcher works with Unity fog?**
  *No, Raymarcher doesn't work with Unity fog. Raymarcher has its own fog feature which is called **Distance Fog** and can be found in Visual Section.*

FAQ

- **How can I avoid using deferred rendering path for mobiles?** ...
  If you would like to use forward rendering path for mobiles, simply setup the Raymarcher for Mobile-VR. However please keep in mind that the Z Buffer won't work as it's all projected on the quad (Unity generic objects will not react with Raymarcher objects).

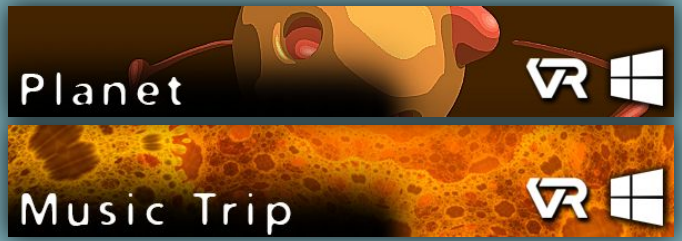
- **Does the Raymarcher contain 'Music Reactor' script?**
  Yes, the Raymarcher does contain very simple 'music reactor' script that can be found in example assets.

Example Content

If you would like to test your performance or try the raymarcher package, you can play/download example demos for free.
Available for PC-Windows, PC-VR, WebGL & Oculus Quest. ***Just click the image.***

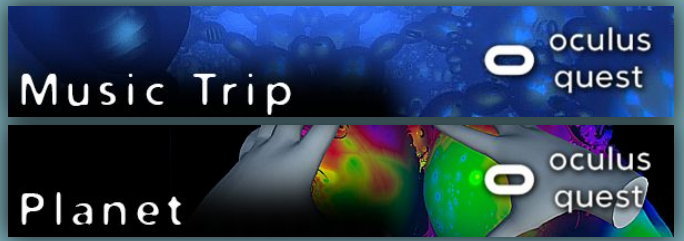
PC
Executable
Win-only

WebGL
Play in
Browser



PC-VR
Executable
Win-only
Unity XR

Mobile-VR
Play in
Oculus Quest

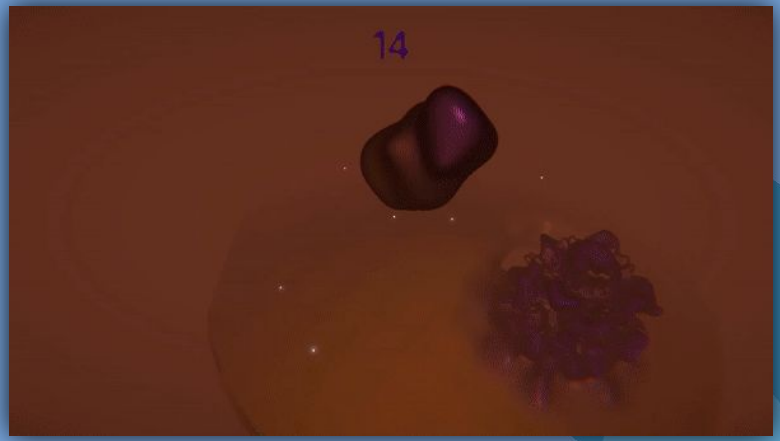


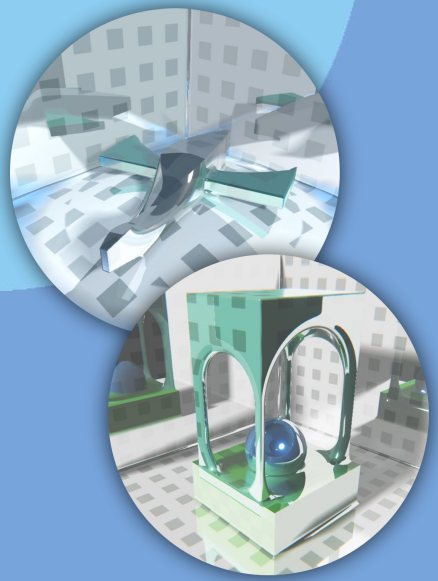
Commercial Products

There are some products on the market that use Raymarcher package. Feel free to explore the Raymarcher possibilities!
Just click the gif.

Cellings [PC-Win] *(2022)*

(2021) [Mobile-iOS] Refractor Dilemma





Thank you!

If you have any questions, issues or suggestions, you are very welcome to join my official Discord channel!
(just click the image below)

DISCORD

If you don't like Discord, you can contact me directly here
https://matejvanco.com/contact