MK Edge Detection

Quick Start

Contents

1.0 Setup / Install 1
- 1.1 Import 1
- 1.2 Legacy Image Effects (Built-in) 2
- 1.3 Post Processing Stack v2 (Built-in) 2
- 1.4 Integrated Post Processing API / Volumes (Universal) 2
- 1.5 Scriptable Renderer Feature Only / Global (Universal) 3
- 1.6 Integrated Post Processing API (High Definition) 3
- 1.7 Examples 3

2.0 Settings 4
- 2.1 Global Shader Features 5
- 2.2 Input 6
- 2.3 Appearance 8
- 2.4 Depth 9
- 2.5 Normal 10
- 2.6 Scene Color 11
- 2.7 Debug 12
- 2.8 Performance & optimization tips 13

3.0 How to get a good setup quickly 14

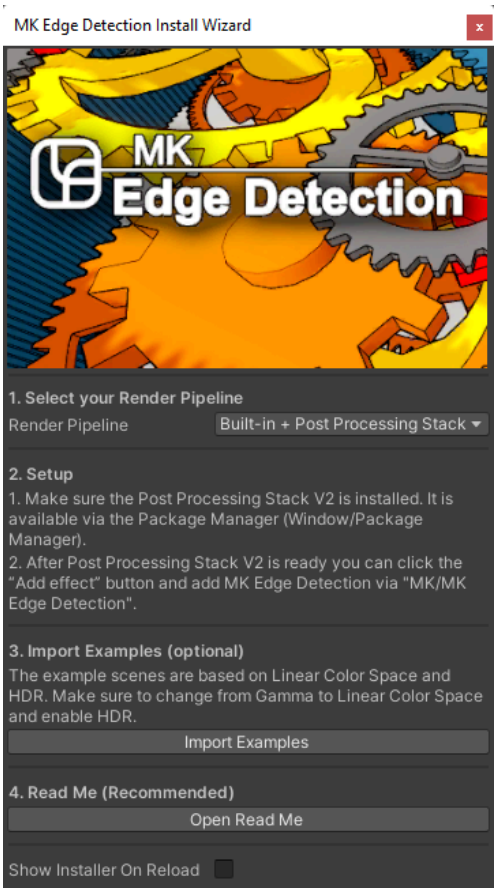
4.0 Scripting 15

5.0 Variant Stripping 16

6.0 Limitations 17

7.0 Feedback / Get in touch 18

1.0 Setup / Install



1.1 Import

Right after the import the Install Wizard should open ("*Window/MK/Edge Detection/Install Wizard*") and guide through the basic setup of the shader. All basic systems should be set ready automatically.

MK Edge Detection is supports the following post processing setups:

- **Built-In** Render Pipeline: Legacy Image Effects
- **Built-In** Render Pipeline: Post Processing Stack v2
- **Universal** Render Pipeline: Integrated Post Processing API via Scriptable Renderer Feature
- **Universal** Render Pipeline: Scriptable Renderer Feature Only (skip the Post Processing API entirely)
- **High Definition**: Integrated Post Processing API

1.2 Legacy Image Effects (Built-in)

The legacy image effects are the very old way to do post processing in Unity. You simply apply a script to your rendering camera.

1. On the game object of your rendering camera click the “Add Component” Button and apply the “MK Edge Detection” script.

1.3 Post Processing Stack v2 (Built-in)

1. Make sure the Post Processing Stack V2 is installed. It is available via the Package Manager (Window/Package Manager). More info about the package: Post Processing Stack V2.
2. After Post Processing Stack V2 is ready you can click the “Add effect” button and add MK Edge Detection via “MK/MK Edge Detection”.

1.4 Integrated Post Processing API / Volumes (Universal)

1. Select your Universal Render Pipeline Renderer Asset and add the custom Renderer Feature: MK Edge Detection. This pre-setup is required because otherwise custom post processing can’t be added to the URP.
2. On the added Renderer Feature set the Work Mode to Post-Process Volumes.
3. Make sure your Post Processing setup is working. More info: Post-processing in the Universal Render Pipeline
4. Apply MK Edge Detection on your Volume Component via “Post-processing/MK/MK Edge Detection”.

1.5 Scriptable Renderer Feature Only / Global (Universal)

The global setup will skip the Post Processing API entirely.

1. Select your Universal Render Pipeline Renderer Asset and add the custom Renderer Feature: MK Edge Detection. This pre-setup is required because otherwise custom post processing can't be added to the URP.
2. On the added Renderer Feature set the Work Mode to Global.

Please note that the URP 2D Renderer does not provide depth or normals. Therefore only "Scene Color" based lines can be detected.

1.6 Integrated Post Processing API (High Definition)

1. Add MK Edge Detection to your custom post processing under HDRP Default/Global Settings on the "After Opaque And Sky" List. Navigate to it via: Edit > Project Settings > Graphics > HDRP Default/Global Settings.
2. Make sure your Post Processing setup is working. More info: Post-processing in the High Definition Render Pipeline
3. Apply MK Edge Detection on your Volume Component via "Post-processing/MK/MK Edge Detection".

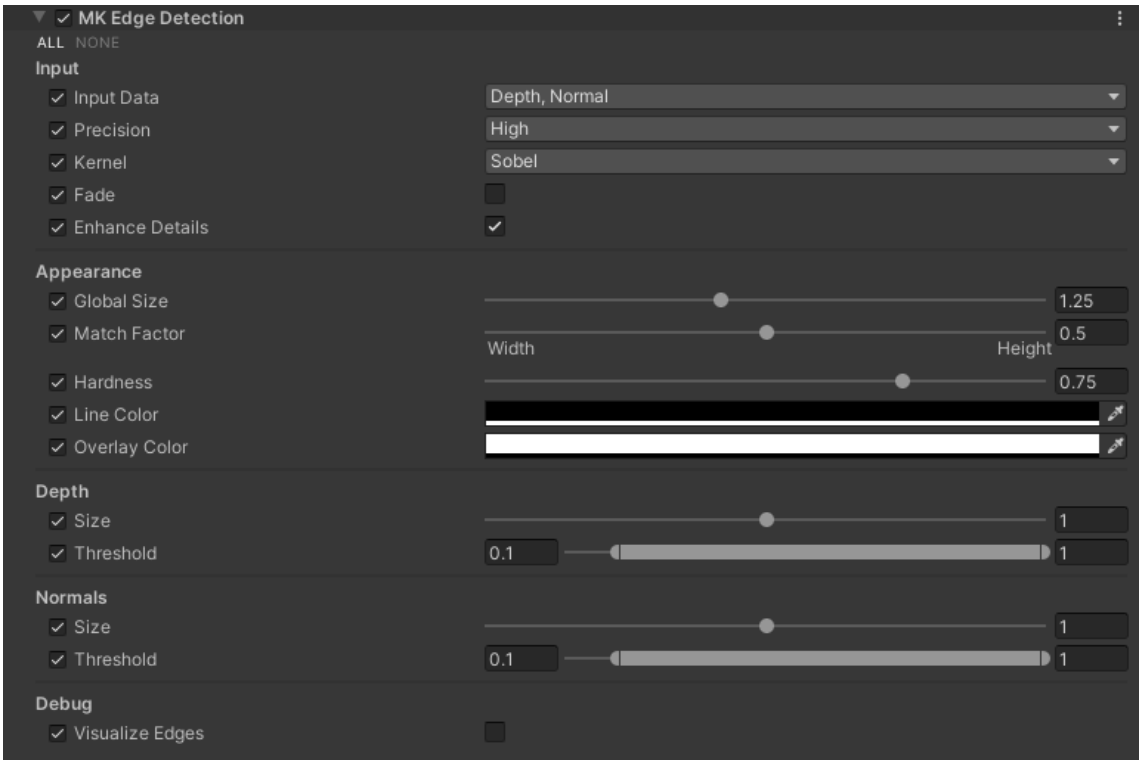
1.7 Examples

To get the examples ready you have to do the following using the install wizard:

1. Select your render pipeline. Examples will be imported based on your selection here. Make sure to set the correct render pipeline.
2. Click the "Import Examples" button.

The example scenes are based on Linear Color Space and HDR. Make sure to change from Gamma to Linear Color Space and enable HDR.

2.0 Settings


MK Edge Detection
ALL NONE
Input
Input Data
Depth, Normal
Precision
High
Kernel
Sobel
Fade
Enhance Details
Appearance
Global Size
1.25
Match Factor
0.5
Width
Height
Hardness
0.75
Line Color
Overlay Color
Depth
Size
1
Threshold
0.1
1
Normals
Size
1
Threshold
0.1
1
Debug
Visualize Edges

2.1 Global Shader Features

What are Global Shader Features?
Shader variants can grow exponentially as you add more features, which in turn increases compile times during builds. Global Shader Features allow you to enable or disable specific functionalities project-wide, reducing the overall number of variants and streamlining the build process.

Global Shader Features can be found on the install wizard (*Window/MK/Edge Detection/Install Wizard*).

Pixel Perfection	When enabled, rasterization is optimized for pixel precision. Edges increment exactly one pixel per size unit, achieving maximum granularity.

2.2 Input

Input Data	Sets up the input data for effect and combines the results in a final composition. It's generally recommended to enable depth and normal for best results. Scene Color is optional in most cases. If no input data is provided the effect will be disabled. **Depth**: Focuses on detecting the silhouette produced by the depth buffer. This option will force the depth texture from the render pipeline. **Normal**: Highlights inner edges based on camera normals, but also enhances them by the silhouette. This option will force the depth normals texture from the render pipeline. **Scene Colo**r: Shows small details based on the rendered image of the camera.
Precision	Determines the overall precision to detect edges. A higher precision detects edges more reliably. This property has a high impact on performance. **High**: Utilizes a full set of box lookups. Best quality. **Medium**: Utilizes a limited set of lookups. Balanced: in a good trade-off in terms of quality and performance. **Low**: Utilizes a minimum amount of lookups. Best performance.
Kernel	Determines the input kernel used for the detection based on the desired precision. **High**: Sobel: Gives smooth results and emphasizes edges of all kinds. Scharr: Provides a good rational symmetry between the edges. Prewitt: Emphasizes horizontal and vertical edges, but also takes diagonal edges into account. **Medium**: Roberts Cross (Diagonal): Focuses on diagonal edges. Roberts Cross (Axis): Focuses on horizontal and vertical edges.

	Low: Horizontal: Approximates horizontal edges only. Vertical: Approximates vertical edges only.
Enhance Details	Uses a combination of input data to calculate and detect edges (planar projected). This can greatly increase the quality of detected edges for difficult and complex geometry (Only available if Depth and Normal input data enabled).
Fade	Enables the fading feature. Edges fade out based on camera distance in world units. Each input data has its own fading setup. **Limits**: Ranges between 0 and 1 to either show edges at a distance or hide edges up close. A minimum limit higher than 0 will show edges far away from the camera, and a maximum limit lower than 1 will hide edges near the camera. **Begin**: Controls the start point where fading should start (world units). **End**: Controls the end point where fading should end (world units).

2.3 Appearance

Global Size	Multiplies onto the different line sizes for separate input data to influence them globally. (line size) * (global line size) = (final Line Size). Use this to globally increase the size of all edges based on the input data.
Match Factor	Determines if the line size uses width or height as reference, or a mix in between to adapt to different screen sizes. The line size is always automatically scaled for consistent behavior across different resolutions.
Hardness	Outputs softer or harder visually appearing edges and interpolates between them. 0: Edges are more artistic and projected softly. 1: Edges are very clear and strongly visually projected.
Line Color	Sets the color (RGBA) for generated lines/edges that appear on the screen. Alpha controls opacity.
Overlay Color	Sets an overlay color (RGBA). (A) Controls the opacity of the overlay and interpolates between the scene image and the overlay.

Sketch	Enables the sketch feature, giving detected edges a stylized, noisy look. This option may significantly impact performance.
Additional Noise	Applies an extra noise texture to further enhance the sketch effect.
Intensity	Adjusts the strength of the sketch effect. Higher values result in more pronounced, noisy edges.
Frequency	Sets the noise pattern frequency for the sketch effect. Higher values yield a more tightly tiled noise appearance.

2.4 Depth

The “Depth” flag on the input data has to be enabled. The depth input focuses on detecting the silhouette produced by the depth buffer. This option will force the depth texture from the render pipeline.

Size	Determines the thickness of the edges/lines created by the depth input. Use this to fine-tune the ratio between the different edge sizes based on the input data.
Threshold	Setups threshold for high pass and low pass values based on the input data to filter out unwanted edges/lines on detailed areas. Edges/lines are generated based on the filtered value range. **Left Handle**: Left Handle: Controls how much rough details are filtered out. A higher value will reduce the amount of detected edges on rough details. **Right Handle**: Controls how much fine details are filtered out. A higher value will reduce the amount of detected edges on noisy and small details.
Fade Limit	Ranges between 0 and 1 to either show edges on the distance or hide edges on a close lookup. A minimum limit higher than 0 will show edges far away from the camera and a maximum limit lower value than 1 will hide edges near to the camera.
Fade Start	Controls the start point where the fading should start (world units).
Fade End	Controls the end point where the fading should end (world units).

2.5 Normal

The “Normal” flag on the input data has to be enabled. The normal input highlights inner edges based on camera normals, but also enhances them by the silhouette. This option will force the depth normals texture from the render pipeline.

Size	Determines the thickness of the edges/lines created by the normal input. Use this to fine-tune the ratio between the different edge sizes based on the input data.
Threshold	Setups threshold for high pass and low pass values based on the input data to filter out unwanted edges/lines on detailed areas. Edges/lines are generated based on the filtered value range. **Left Handle**: Left Handle: Controls how much rough details are filtered out. A higher value will reduce the amount of detected edges on rough details. **Right Handle**: Controls how much fine details are filtered out. A higher value will reduce the amount of detected edges on noisy and small details.
Fade Limit	Ranges between 0 and 1 to either show edges on the distance or hide edges on a close lookup. A minimum limit higher than 0 will show edges far away from the camera and a maximum limit lower value than 1 will hide edges near to the camera.
Fade Start	Controls the start point where the fading should start (world units).
Fade End	Controls the end point where the fading should end (world units).

2.6 Scene Color

The “Scene Color” flag on the input data has to be enabled. Shows many details based on the rendered image of the camera.

Size	Determines the thickness of the edges/lines created by the scene color input. Use this to fine-tune the ratio between the different edge sizes based on the input data.
Threshold	Setups threshold for high pass and low pass values based on the input data to filter out unwanted edges/lines on detailed areas. Edges/lines are generated based on the filtered value range. **Left Handle**: Left Handle: Controls how much rough details are filtered out. A higher value will reduce the amount of detected edges on rough details. **Right Handle**: Controls how much fine details are filtered out. A higher value will reduce the amount of detected edges on noisy and small details.
Fade Limit	Ranges between 0 and 1 to either show edges on the distance or hide edges on a close lookup. A minimum limit higher than 0 will show edges far away from the camera and a maximum limit lower value than 1 will hide edges near to the camera.
Fade Start	Controls the start point where the fading should start (world units).
Fade End	Controls the end point where the fading should end (world units).

2.7 Debug

Visualize Edges	Visualizes edges as colors (RGB) based on the current view. Overlapping edges from different inputs will result in mixed colors. **(R)** - Depth edges **(G)** - Normal edges **(B)** - Scene color edges If the Enhanced Details feature is enabled existing data will be adjusted accordingly.

2.8 Performance & optimization tips

MK Edge Detection is designed for high performance, but some settings can significantly impact performance—especially on fill-rate limited devices like mobile.

Here is a quick overview over the most performance impactful settings:

Input Data	Enabling depth and normal data provides a solid foundation for high-quality edge detection. This is the default setup and generally recommended. Keep in mind, each additional input requires extra shader samples, so use them strategically.
Precision	**Medium Precision:** A great starting point, offering an optimal balance between quality and performance. **High Precision:** Delivers enhanced accuracy by increasing sample counts for each input, though at a higher performance cost. **Low Precision:** Maximizes performance but may miss some edge details—ideal for specific scene setups where speed is paramount.
Sketch	Enabling Sketch has a very high performance impact, as it requires rendering an additional full-screen quad.

By fine-tuning these settings, you can optimize MK Edge Detection to achieve the best balance between visual quality and performance for your target platform.

3.0 How to get a good setup quickly

By default if you enable MK Edge Detection a generic setup is used. This setup can cover a wide variety of scene setups and looks visually pleasing in many cases.

However if you want to fine-tune the effect its recommend to take the following steps:

1. Enable “Visualize Edges” to get a look where edges appear based on your input data.
2. For each input data try to optimize the threshold values to filter out unwanted edges and highlight the targeted ones. In most cases it's recommended to enable Depth and Normal on the input data.
3. In many cases the “Enhance Details” setting can increase the quality of detected edges (optional).
4. Adjust your desired precision. On pc and consoles you mostly want to use the high precision and on weaker devices medium. Use low only on very weak devices. Remember this has a high influence on performance.
5. Select your kernel. Based on the kernel the style of the edge detection varies.
6. Adjust your size for the edges using the separate size properties to get a good ratio between them.
7. Adjust the global size to further scale the edge sizes.
8. Enabled fading if wanted. Every input data has its own fading setup.
9. Disable “Visualize Edges” and set your properties for the appearance.

4.0 Scripting

You can change any setup of MK Edge Detection during runtime. The core assembly definition file is located under: “_MK/MKEdgeDetection/Components/MKEdgeDetectionComponents.asmdef”.

Based on your render setup a different component is used.

Components:

Image Effects (Legacy) (Built-in)	MK.EdgeDetection.ImageEffectsComponents.MKEdgeDetection
Post Processing Stack V2 (Built-in)	MK.EdgeDetection.PostProcessingStackV2VolumeComponents.MKEdgeDetection
Universal Volume Component	MK.EdgeDetection.UniversalVolumeComponents.MKEdgeDetection
High Definition Volume Component	MK.EdgeDetection.HighDefinitionVolumeComponents.MKEdgeDetection

Examples:

You can find scripting examples located under: “_MK/MKEdgeDetection/Examples/Scripting”.

5.0 Variant Stripping

By default MK Edge Detection includes all possible shader variants when building. Using the variant stripping asset you can adjust the variants that should be included into a build.

You can find the variant stripping asset via “MK/Edge Detection/Variant Stripping”.

Make sure to not strip away the variants you actually need. The names correlate to the settings you see on the MK Edge Detection components.

6.0 Limitations

- MK Edge Detection is injected right after opaque geometry is rendered. This means transparent objects can’t be supported.
- If automatic exposure is enabled, it can mess up the “Visualize Edges” feature, if the input data uses scene colors.
- If Scene Color input data is enabled, then high intensity parts (bright spots like specular for example) of the rendered image can be hard to balance to avoid artifacts.
- Using the Universal Render Pipeline 2D renderer only the scene color input is possible, because there is no depth or normal texture being rendered.
- Using Legacy Image Effects, there is no "Before Transparents" queue or a similar mechanism for effects to differentiate between opaque and transparent geometry. This means you will see transparent objects overwritten by the opaque edges.
- Using Post-processing Stack v2, only Multi Pass is supported for XR. The reason for this is that the Post-processing Stack v2 API does not work properly with Single Pass Instanced rendering (Once the API is getting fixed, MK ChromaFlow will be updated of course). Legacy Image Effects behave similar.

7.0 Feedback / Get in touch

Do you have some cool stuff to show?
I would love to see your results (High resolution screenshots / videos) using the shader!

Questions, bug reports, feature requests, feedback:
Feel free to get in touch via support@michaelkremmel.de.