X-FRAME FPS ACCELERATOR

Kronnect

Contents

What's X-Frame FPS Accelerator?

X-Frame FPS Accelerator is a camera script plus some shader effects that improves the FPS of your game.

The technique used by X-Frame FPS Accelerator works in two steps:

1) Makes the camera render to an off-screen surface with reduced resolution which obviously makes the render faster, but it also reduces any post-image effects time.
2) Upscales the rendered frame to window size, optionally applying MSAA antialias + our custom fast sharpen algorithm, which reduces blur and improves result vs a normal upscale operation.

X-Frame is aimed mainly to mobile and HDPI screens (high dot-per-inch screens). As mobile devices pack more and more pixels per inch, you may afford to lose some pixels in exchange for performance. **It's important to test the effect of X-Frame on the mobile device itself, since usual monitors don't have such DPI and the quality reduction will be more noticeable on the big screens than on mobile.**

X-Frame also controls dynamically both pixel light count and quality of shadows per light in the scene depending on current FPS. It will reduce/restore pixel light count as well as reduce or disable shadows if FPS can't be achieved. These options can be enabled/disabled in X-Frame.

X-Frame allows you to choose the proper balance between performance and image quality, providing full control regarding the FPS and quality range you want in your game.

With X-Frame you can make your game playable in devices that currently can't tackle your rendering stack or even run "heavy" image effects on mobile that takes too much FPS due to their complexity.

Algorithms included in X-Frame

X-Frame FPS Accelerator includes 3 main algorithms:

- **Horizontal Downsampling**: the resolution reduction is applied horizontally. For example, if you choose a 0.5 quality factor, the resolution will be Screen Width / 2 x Screen Height (remember that after rendering the frame, it will be upscaled to full screen resolution). You can choose different ratios, for example, a 0.75 will just reduce the image size by a quarter. This reduction is always applied regardless of current FPS.

- **Quad Downsampling**: the resolution reduction is applied horizontally and vertically. For example, if you choose a 0.5 quality factor, the resolution will be Screen Width / 2 x Screen Height / 2 (remember that after rendering the frame, it will be upscaled to full screen resolution). This reduction is also always applied regardless of current FPS.

- **Adaptative Downsampling**: works like Quad Downsampling but you can choose the desired FPS and a quality range and. X-Frame will dynamically use different off-screen resolutions to match the best FPS according to the quality range specified.

Important: you always control the minimum quality factor! For example, if you want to maximize quality vs FPS, you can set a quality factor high, like 0.7 or 0.8 – or if you want to ensure the best smooth experience in your game in terms of FPS in exchange for quality you could reduce the minimum quality to 0.5 or even 0.3.

X-Frame brings you an opportunity to address less powerful devices and make your game available to more players. If current device is powerful enough and your FPS requirements are met, with adapatative downsampling X-Frame won't affect the image quality of your game.

NEW RENDERING PIPELINES NOTICE!
X-Frame FPS Accelerator will detect if Universal Rendering Pipeline is enabled. In this case, the adaptative downsampling is always used. Some other options won't apply in this scenario.

Quick Start

To use X-Frame:

1- Import X-Frame package into your project.
2- Select your main camera and select "X-Frame FPS Accelerator" from the component menu:
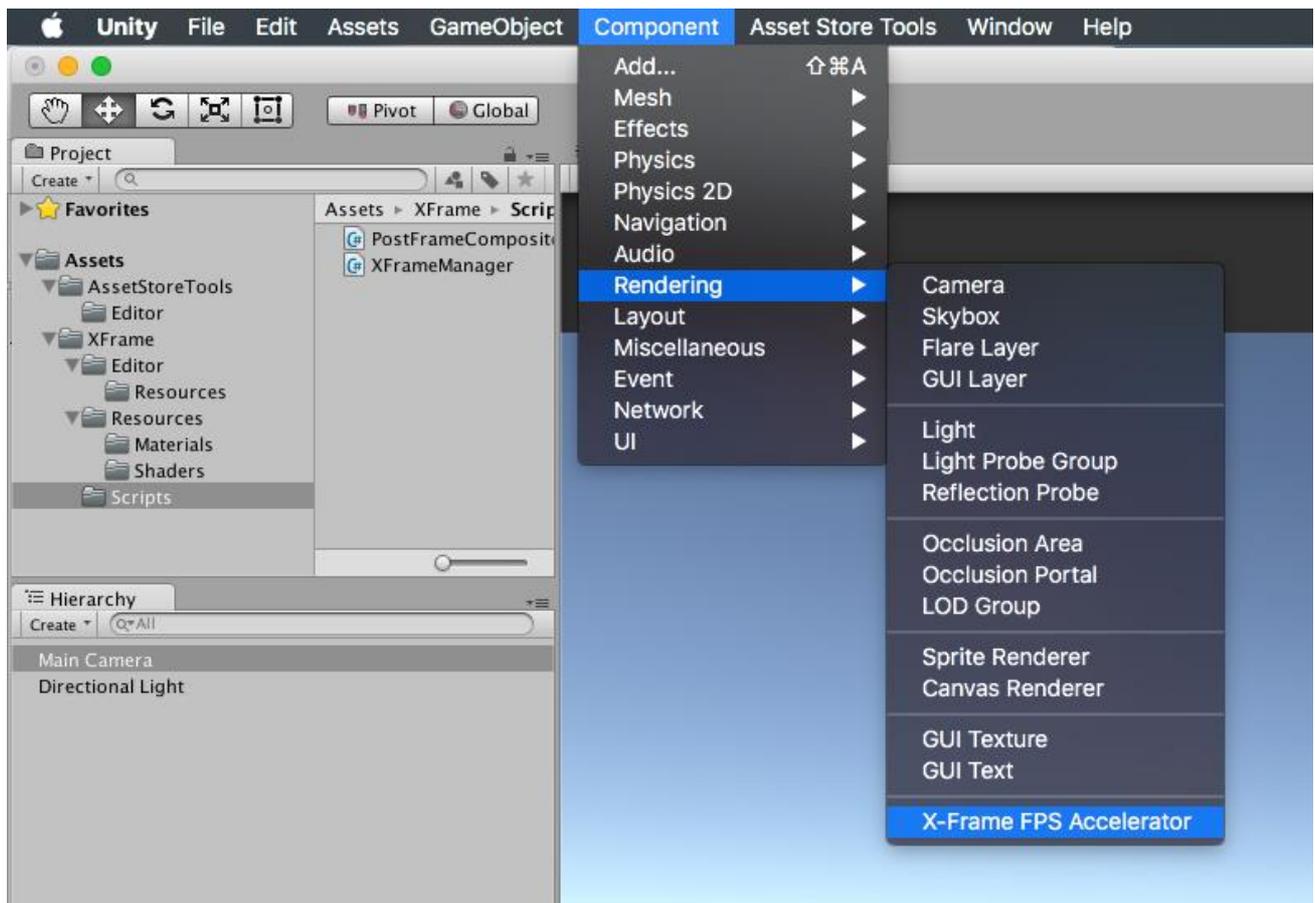
You can also add the script to your camera, simply clicking on the "Add component" button in the inspector and entering the name "X-Frame" (as you press X, it will be listed in the result list):

Once you add X-Frame to your camera, click "Help" button on the inspector for more details and guidance.

Description of X-Frame parameters

General Settings

❖ **Method**: choose one of the three operation modes as described in the section "Algorithms included in X-Frame". Note that when Universal Rendering Pipeline is used, it will always use adaptative downscaling internally.

❖ **Minimum FPS**: your desired target FPS. If current FPS are greater than this value, no image reduction will be applied, ie. X-Frame won't affect your image.

❖ **Preserve Nice FPS**: if FPS exceeds this value, X-Frame will temporarily be disabled so it does not affect at all to your game performance. For example, if your game already runs on 60 fps, X-Frame should not execute. However, if FPS drops below this value, X-Frame will reactivate automatically.

❖ **Show FPS**: display current FPS (frames per second) and optionally the current quality level (measured as a percentage where 100% = full quality or resolution).

Quality Management Settings

❖ **Adapt Speed Up**: in adaptative downsampling, this parameter controls the speed of change between different image quality (ie. image resolution). When FPS drops, X-Frame will reduce the image quality in 10%. When FPS increase, X-Frame will increase the image quality in 10%.

This parameter controls the speed of quality increase, when FPS are above the Minimum FPS parameter. If you want X-Frame to increase quality as soon as possible increase this value.

❖ **Adapt Speed Down**: controls the speed of quality decrease, which occurs when FPS are below the Minimum FPS parameter.

❖ **Minimum Quality**: this setting allows you to decide the minimum image resolution that you can afford to use in exchange for the extra FPS. This setting puts you in control of the impact of X-Frame in the resulting image quality. You can choose a high value and the image quality won't be much affected, but the gain in FPS will be less. Experiment with different values always on the mobile device itself, never on screen!

❖ **Static Camera Quality**: this setting is similar to minimum quality and will be used instead when the camera is not moving or rotating. You may set this value equal than the minimum quality parameter or increase it a little bit, so when the player is static in the scene, the image quality can get a little better. It's known that when you need "action", you will want the best FPS but if there's no "action" you will want the best image quality for the best viewing experience – this setting allows you to control that situation.

❖ **Reduce Pixel Count**: this setting will reduce pixel count in the scene until 1 if desired minimum FPS is not achieved. It will revert to original value as soon as minimum FPS is obtained. The threshold setting let you specify the minimum resolution fall before applying this optimization.

❖ **Manage Shadows**: X-Frame will monitor all lights in the scene and will swtich the quality of its shadows, from soft to hard shadows or even disabling shadows. The control is done per light depending on current FPS goal.

❖ **Manage LOD bias**: the LOD bias is a global multiplier for the distance LOD that affects all LOD groups. It can be used to reduce the distance required to switch to a higher LOD level thus reducing triangle count earlier to improve performance.

Rendering Settings

❖ **Filtering**: specifies the sampling filter used when upscaling the frame, either nearest neighbour (faster, pixelates) or bilinear (smoother, blurs).

❖ **Antialias**: this slider controls the MSAA quality level. Setting this slider to 2 or more will enable antialias and will dramatically improve the result. Note that this setting doesn't change the main camera or project settings, it only affects the internal rendering performed by X-Frame.

❖ **Clear Flags**: the clear flags for the X-Frame camera. By default it's set to Nothing, but if you experience ghosting artifacts, set this to Color or Solid Color.

❖ **Blend with Background**: when enabled, X-Frame won't clear the background of the frame buffer when upscaling the frame buffer. This option allows certain combinations of cameras where some camera render at different depths.

❖ **Render Method**: choose a final compositing method.
  o "Second Camera Billboard World Space": uses a second camera to draw the upscale frame inside a world space quad. Default and recommended setting.
  o "Second Camera Blit": uses a second camera and a post-effect to upscale the frame optionally adding a sharpening pass. This mode is slower on older devices.

❖ **Sharpen Image**: optionally sharpen the upscaled image which brings more details to the resulting image (only available with Second Camera Blit).

❖ **Prewarm**: forces X-Frame to create the offscreen render buffers during start-up, instead of creating them on-demand. Most of the time you won't notice any difference although checking it should reduce any hiccup when changing resolution in Adaptative Downsampling on some devices.

❖ **GameObject Clicks:** when enabled, X-Frame will perform raycasting and bubble events like OnMouseDown, OnMouseUp and OnPointerClick to the gameobjects in the scene.

❖ **GameObject Clicks:** when enabled, X-Frame will perform raycasting and bubble events like OnMouseDown, OnMouseUp and OnPointerClick to the gameobjects in the scene.

❖ **Boost Frame Rate**: when enabled, X-Frame will set the maximum frame rate and v-sync values at optimal values to obtain the best FPS possible when application is running.

Canvas and Ray Casting considerations

X-Frame will automatically manage ray casting for Canvas set to Screen Space Overlay or World Space.

If you experience issues with Camera.ScreenPointToRay function, you should use XFrameManager.instance.ScreenPointToRay function instead, since it takes into account any active downsampling factor applied to the camera target.

Custom APIs for dealing with UI and GameObject interactions

X-Frame provides a set of useful APIs to adjust the tap or mouse positions due to downscaling:

To access the API use this code:

using XFrameEffect;
…
XFrameManager xframe = XFrameManager.instance;

Then the following methods are available:

public void UpdateUICanvases()
Call only if you create Canvas objects at runtime. This method will force X-Frame to integrate with new canvas UI objects.

public Vector3 AdjustScreenPosition(Vector3 position, bool targetIsDownscaled)
Transforms the input position into a proper screen coordinate that takes into account downsampling. If the target position is aimed to a screen space overlay canvas, set targetIsDownscaled to false.

public Ray ScreenPointToRay(Camera camera, Vector3 position);
Returns a Ray object from a camera and a screen position.

public RaycastResult Raycast(Camera camera, Vector3 pointerPosition)
Performs a raycast on the camera from the pointer position (ie. screen mouse position or Input.mousePosition)

public bool IsPointerOverUIElement(Vector3 pointerPosition)
Returns true if the pointer is over any UI element of any Canvas (only canvas set to Screen Space – Camera or World Space).

public bool IsPointerOverGameObject(Camera camera, Vector3 pointerPosition)
Returns true if the pointer is over any gameobject.

Additional Support & Questions

Remember to test your settings on the real mobile device!

If you have any issue using X-Frame, please use our support forum on https://kronnect.com/support.