LiquidVRLateLatch
SDK Sample

Disclaimer

The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information.

Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale.

AMD, the AMD Arrow logo, ATI Radeon™, CrossFireX™, LiquidVR™and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Windows™, Visual Studio and DirectX are trademark of Microsoft Corp.

Copyright Notice

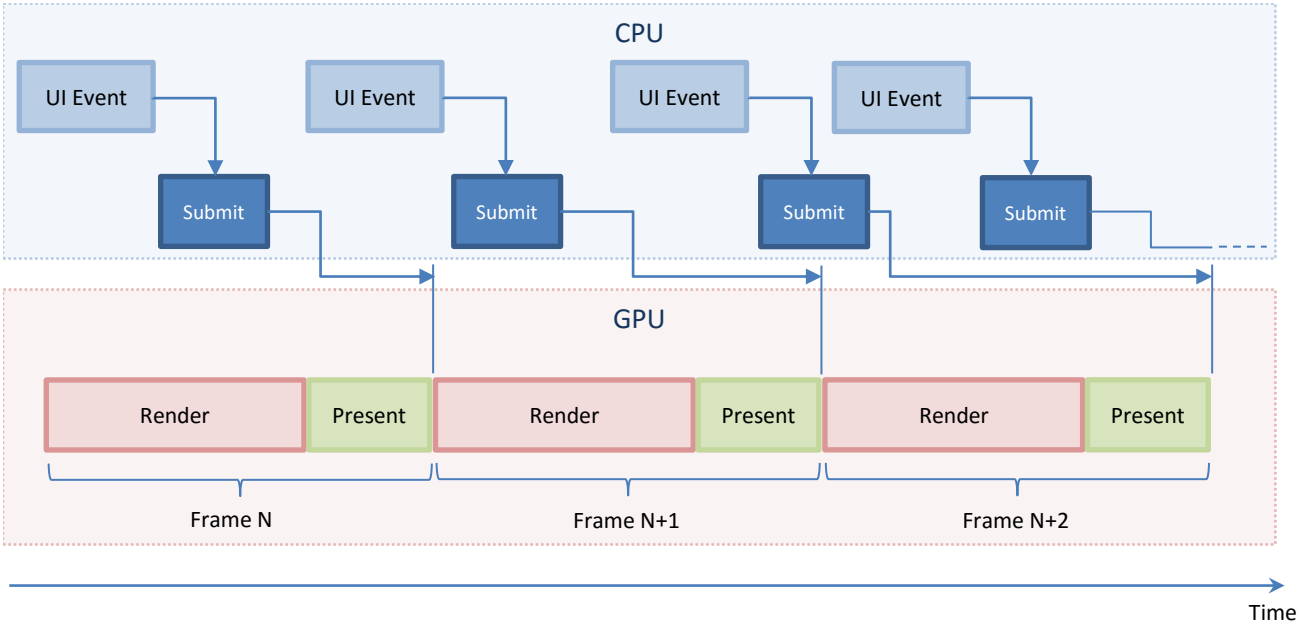
Advanced Micro Devices
One AMD Place
P.O. Box 3453
www.amd.com

Overview

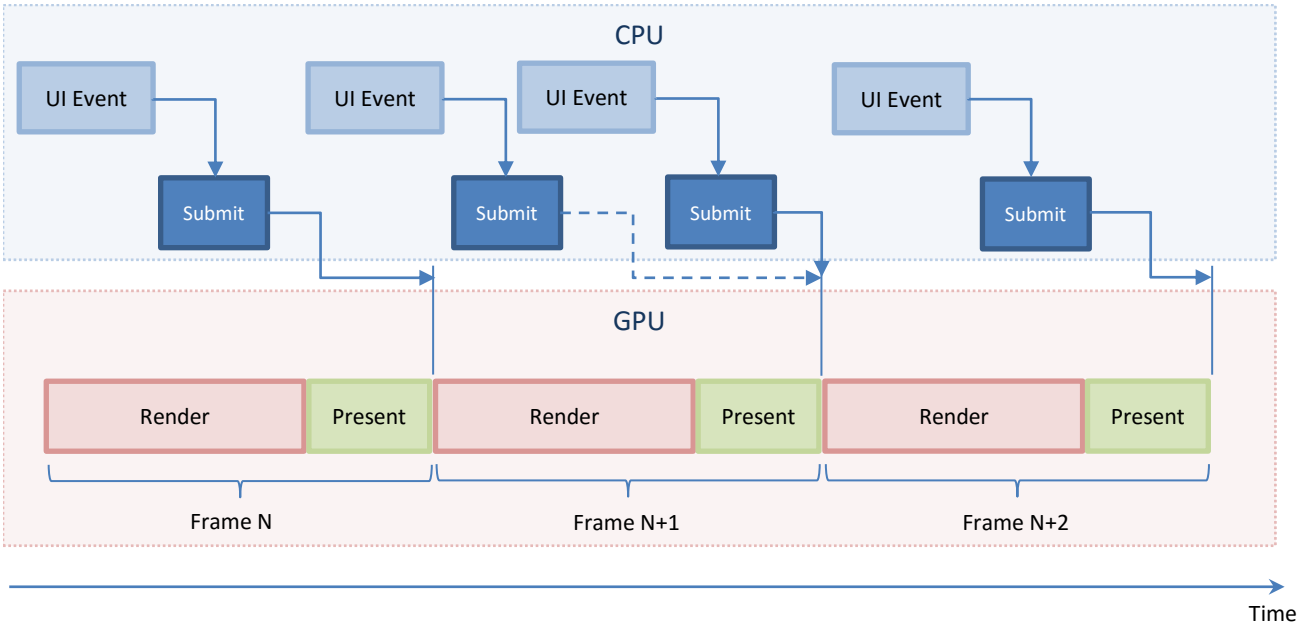
The LateLatch sample demonstrates how games and VR applications using LiquidVR™ SDK on AMD Radeon™ graphics could increase the responsiveness and reduce the lag of their graphics to user's actions by using a special programming technique called "late latching".

In a typical gaming or a VR application frames are rendered based to user's actions coming from various controls, such as mice, joysticks, gamepads, etc, submitted by the CPU to the GPU. Once a render task is submitted to the GPU, it is not possible to modify its parameters. Since CPU and GPU time domains are not synchronized, there is some lag between the moment when a frame is submitted to the GPU pipeline for rendering and the moment when the rendered frame is presented on the screen. This lag depends on the amount of time need to render a scene and in certain cases can be significant, especially when the frame rates are low. At the same time the user continues to control the application unaware that the image currently being visible on the screen has been rendered according to the transformation matrix recorded at the moment when the render job was submitted to the GPU. This lag makes the application appear less responsive to user's actions, therefore it is very desirable to reduce it as much as possible.

The example below illustrates how this lag is introduced:



LiquidVR deals with this problem by using the late latching approach which allows the application to change the transformation matrix up to the moment when the frame is about to be rendered:



With late latching UI events occurring before the frame has started being rendered can still be submitted to the GPU and applied as rendering parameters, overwriting previously submitted parameters. This allows for adjustment of the position of a moving object according to the most recent input from the user, minimizing the response lag.

It is important to note that late latching cannot increase the application's frame rate, but it can make the application appear more responsive to user's input.

The positive effect of late latching on application's responsiveness is most noticeable at low frame rates.

Pre-requisites

To run the sample, the following hardware and software requirements need to be fulfilled:

- A PC with an AMD Radeon™ graphics card. Please note that Late Latch is not supported when CrossFire is enabled.
- A 64-bit Microsoft Windows 7, Windows 8.1 or Windows 10 with the Radeon Crimson graphics driver.
- **Optional:** to compile the sample, Microsoft Visual Studio 2013 is required.

Building the LateLatch Sample

Start Microsoft Visual Studio 2013 and open the *LiquidVR_SDK_1.0\samples\LiquidVR_Samples_VS2013.sln* workspace. Navigate to the *Solution*

Explorer pane in Visual Studio, right-click on the *LateLatch* project and choose the *Set as Startup Project* option from the pop-up menu. Choose *Build Solution* from the *BUILD* menu.

Running the LateLatch Sample

Run the executable built in the previous step or run a pre-built binary located in *LiquidVR SDK\LiquidVR_SDK_1.0\bin_x64\Debug\LateLatch.exe.* Once started, you should see the following image on your monitor:

The sample emulates a video game displaying a stereoscopic 3D scene with a low frame rate. On a regular non-stereoscopic display the left and the right sides of the image represent what the left and the right eye sees.

The scene can be navigated by dragging it with a mouse with the left button pressed.

The human figure in the scene can be moved over the scene's background by dragging it with a mouse with the left button down.

A user interface allowing control of various image rendering parameters is displayed on the right in a semi-transparent pane. It can be turned on or off using the F1 function key on the keyboard.

The user interface pane has the following controls:

- The *Toggle full screen* button switches between a windowed and a full-screen mode.
- The *Change device* button allows you to select the graphics card on which the sample is run in case your system is equipped with multiple GPUs.
- The *Number of Draw Calls* slider controls the number of times the game character is drawn over the background.
- The *Shift TinyMesh(s)* check box offsets the image of the game character by a few pixels each time it is drawn.
- The *Late Latch* check box toggles the late latching mode on and off.

The *Number of Draw Calls* and *Shift TinyMesh(s)* controls affect the amount of time needed to perform the 3D rendering operation.

Note: If your computer is equipped with multiple GPUs, ensure that the demo is running on an AMD GPU by clicking on the *Change Device* button and selecting the appropriate device from the menu.

The current device, format and the frame rate are displayed in the overlay in the top left corner over the main scene:

To observe the effects of the late latching technique perform the following steps:

1. Adjust the *Number of Draw Calls* slider to achieve a low frame rate of several (2-7) frames per second.
2. Turn **off** the late latching mode by unchecking the *Late Latch* check box.
3. Quickly drag the main game character with the mouse while holding the left mouse button down. Observe the lag between the mouse pointer and the movements of the main game character.
4. Turn **on** the late latching mode by checking the *Late Latch* check box.
5. Quickly drag the main game character with the mouse again, trying to maintain the same speed, direction and the trajectory of the motion as in step 3 above. Observe the lag between the mouse pointer and the movements of the main game character. It is normal for the motion to be jerky because of the low frame rate, but the delay between

the movement of the mouse and the movement of the game character in response to it should be noticeably shorter than when late latching was off.

6. Experiement with the position of the *Number of Draw Calls* slider and *Shift TinyMesh(s)* check box, as well as mouse motion to observe the effects of the late latching technique on responsiveness of the application.