August 24, 2016

ASSISTED FRAME INTERPOLATION FOR ANIMATED 2D/3D GRAPHICS

Simon Smith

Anthony Mantler

ASSISTED FRAME INTERPOLATION FOR ANIMATED 2D/3D GRAPHICS

ABSTRACT

Disclosed are a system and method to overcome dropped frame problems encountered during displaying of animated 3D (or accelerated 2D) graphics. The 3D rendering hardware generates a vector field describing the objects being displayed, which is achieved by the 3D software program. The motion vector information in a frame is used to displace the rendered values to create a future interpolated frame. A similar technique is used to create a future interpolated vector buffer. The interpolating frames technique provide more accurate motion estimates without significant impact on display quality, adding load to CPU or GPU, or increasing latency. The method can be used on hardware with limited power.

BACKGROUND

Dropped frames are a commonly encountered problem during display of animated 3D (or accelerated 2D) graphics. It is often a challenge to avoid dropped frames in devices with processing power limitations. There are some existing solutions for the dropped frame problem. Video encoding commonly uses motion estimation to generate interpolated frames. Some TV sets use interpolation between two frames to increase effective frame rate. But both of these techniques use consecutive frames to estimate motion and create the interpolated frame. They do not use any additional metadata that would provide more accurate motion estimates. A system and method are disclosed to interpolate frames which provide more accurate motion estimates without significant impact on display quality, adding to CPU or GPU load, or increasing latency.

DESCRIPTION

A system and method are proposed to overcome dropped frame problems encountered during displaying of animated 3D (or accelerated 2D) graphics. The proposed idea is to make use of the 3D rendering hardware to generate a vector field describing the motion of objects being displayed, as shown in FIG. 1. This is achieved by the 3D software program adding a motion vector to each vertex, which would, in turn be rendered into a 3-value buffer (V1) alongside the actual displayed content (P1).

The 3D rendering hardware would then use the motion vector information in V1 to displace the rendered values in P1 to create a future interpolated frame P1' and to displace the rendered values in V1 to create a future interpolated vector buffer V1'. If frame P2 is not rendered in time for the next display refresh, the pixel values in P1' are used instead, and the displacement buffer V1' is used to create P1" and V1". When frame P2/V2 is rendered, the buffers derived from P1 are discarded and P2 is displayed on the screen, serving as the basis for P2', P2" and so on.

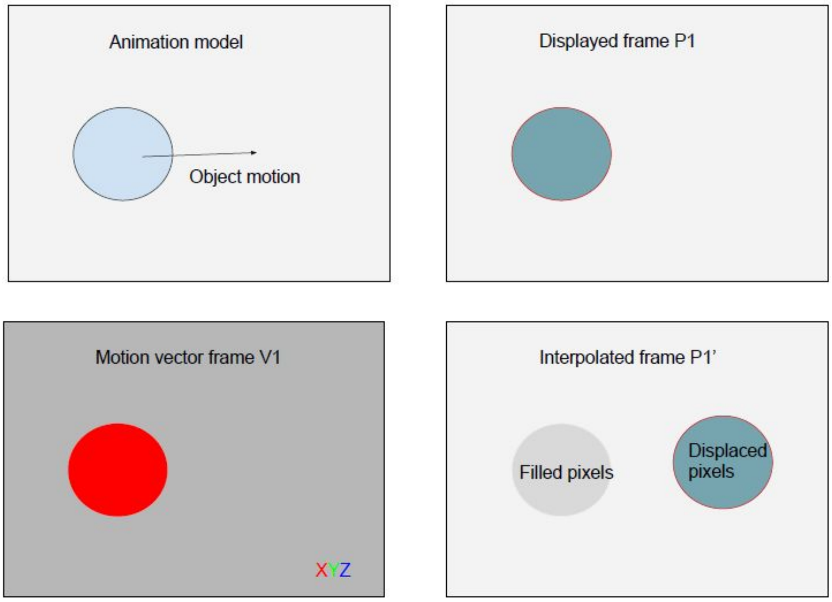

FIG. 1: 3D rendering hardware generated vector field describing the motion of objects being displayed

The system and method requires much less processing than the classic motion estimation and interpolation algorithms as it does not have to analyze a sequence of frames. It is also not limited to interpolating between existing frames, which would result in added latency. Since the exact scene motions are known to the animation engine, no computational estimation of motion is required. This makes the technique viable for use on hardware with limited power.