March 31, 2016

SYSTEMS AND METHODS FOR DYNAMICALLY MANAGING RENDERING OF ANIMATION CONTENT

Eugenio Marchiori

SYSTEMS AND METHODS FOR DYNAMICALLY MANAGING RENDERING OF ANIMATION CONTENT

In FLASH-based animation rendering, each frame of the animation can take as long as the frame needs to render, and if the time the frame takes to render is more than the time allotted to each frame, the animation renders slower. In some situations, for instance, in online advertising, the animation can be allotted a fixed duration of time (for example, a 30 second ad). When the animation is rendered slower, the animation uses all available resources thereby introducing latencies in the rendering of other content. In FLASH-based online advertisements, rendering ads slowly on a content publisher’s webpage can result in a bad user experience and as such, this paper aims to address some of the concerns related to the rendering of FLASH-based animations of online ads on webpages.

FLASH-based animations may include a plurality of frames, each of which is dependent on a previous frame for generation. As such, to render any frame of a FLASH-based animation, the previous frame has to be generated. This is different from keyframe based animations, where some frames are designated as keyframes (i.e., a frame in which a complete image is stored in the data stream). In such animations, one or more non-keyframes may be dropped when the animation is rendering slow without affecting the ability of the animation to render. In video games and other applications that utilize keyframe-based animations or can perform keyframe-interpolation, one or more frames between keyframes may be dropped to reduce the total amount of time to render the animation. In such implementations, the frames that are dropped are not even generated.

At present, some types of internet display advertisements, and in particular those that use vector graphics, may generate and render frames on the client device. However, generating and rendering (i.e. drawing the frames into the browser) can be costly to computer resources of the client device and can therefore adversely affect performance, quality, load times, etc. when there is a shortage of available computer resources. Animations in online advertisements typically have a fixed frame rate (e.g., 24 frames per second), which results in these advertisements taking longer than desired to load if the rendering is slow, which can cause latency and degrade the overall user experience. Solutions utilized in other video rendering applications are not

4831-2586-9357.2

applicable, as online advertisements do not use keyframes. Each frame of the online advertisement is generated prior to rendering, and the frame rate is fixed.

To address at least the problems described above, including others, this paper discusses methods and systems for dynamically managing rendering of animation content. The system can prevent some subset of frames from being rendered if not doing so would consume more computing resources than desired. In one implementation, if the render time (“render”) plus the frame generation time (“tick”) is less than the frame time (based on the fixed frame rate) itself, the content will be animated normally. However, if the render time plus the tick time is greater than the frame time itself, meaning that there is now some latency, the system will calculate the excess time to generate and render the current frame and not render some of the succeeding frames based on the calculated excess time.

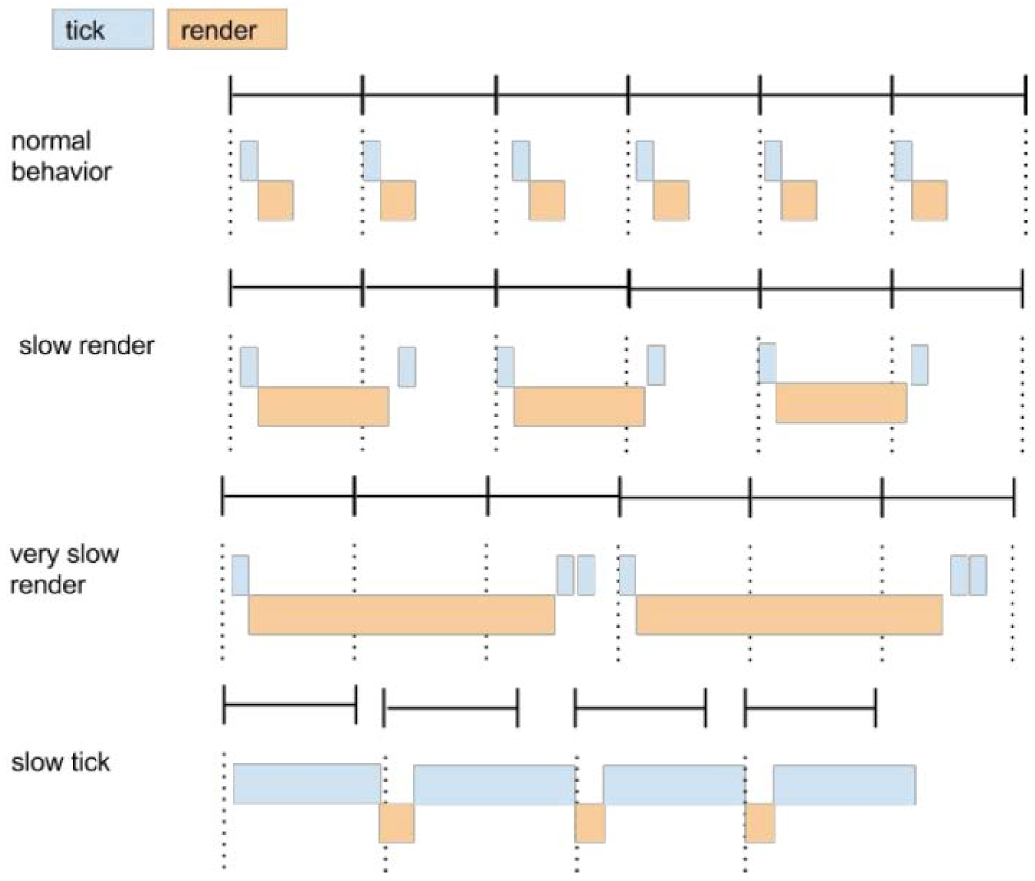

FIG. 1

This paper generally describes methods and systems for dynamically managing rendering of animation content that allow ticking to behave normally (e.g., without the need for catching up), and for dropping of frames by not rendering some ticked frames when bottlenecking of frames occurs, as needed. Referring to FIG. 1 above, a graph illustrates how the present solution functions under four different scenarios: 1) normal behavior; 2) slow render; 3) very slow render; and 4) slow tick. As shown in FIG. 1, under the normal behavior scenario, no dropping of frames is necessary, and ticking and rendering are performed normally. Each tick is performed on time at the beginning of each frame period (which is based on the fixed frame rate). Under the slow render scenario, the rendering of the first frame takes a long enough amount of time to extend into the second frame's time period. As such, the ticking of the second frame occurs as soon as the rendering of the first frame is complete, but the rendering of the second frame is skipped so that the ticking and rendering of the third frame may begin on time. In the slow render scenario, every second frame is dropped (e.g., not rendered).

Under the very slow render scenario, the rendering takes enough time to extend into the third frame's time period. After rendering of the first frame is complete, ticking of the second and third frames occurs immediately thereafter without rendering of the second and third frames, such that the fourth frame is ticked and rendered thereafter and on time. In the very slow render scenario, each second and third frame is dropped. The system may also set a maximum number of frames that is allowed to be skipped such that too many frames are not dropped such that animation quality is significantly degraded. Under the slow tick scenario, the system can tick again as soon as possible after rendering. However, the rendering of some of the ticked frames is skipped proportional to the whole frame time.

Referring to FIG. 2 below, a flowchart illustrating a method of implementing the present solution is shown. The method of FIG. 2 is based on the slow render and very slow render scenarios described above and shown in FIG. 1. The method begins with receiving content including frames. Each of the frames can include a frame time, which can be based on the frame rate (e.g., the frame time can be the maximum amount of time it should take to tick and render that frame within a frame period). Each of subsequent frames can be dependent on a previous frame for generation or ticking, which is how the animation of advertisements functions (i.e., without keyframes), and which is illustrated in FIG. 1 by each frame sequentially ticking without rendering until the processing of the frames is caught up.

Next, the method includes generating a first visual content based on a first frame of the frames. Next, the method includes rendering the first visual content for display. The rendering of the first visual content can occur in response to the generating the first visual content step.

Next, the method can include determining a sum of a generation time and a render time of the first visual content. The sum refers to the amount of time that it takes to tick and render the first visual content (e.g., the total length of a pair of the tick and render bars shown in FIG. 1). This step of determining the sum can be performed subsequent to the step of rendering the first visual content for display.

Next, the method includes determining a number of subsequent frames to skip rendering based on the sum of the generation time and render time of the current generated and rendered frame. The number of subsequent frames to skip rendering can be based on a comparison of the sum of the generation time and the render time to the frame time of the current generated and rendered frame. In other words, if the generation time and render time of the current generated and rendered frame extends into any number of subsequent frame periods, frames corresponding to those breached frame periods can be designated as being skipped (e.g., as shown in the slow render and very slow render scenarios of FIG. 1). For example, if the sum of the generation time and the render time is greater than the frame time, then at least one subsequent frame will be skipped (not rendered).

Next, the method includes generating a number of frames equivalent to the number of subsequent frames to skip rendering. This generation of the skipped frames refers to the ticking of sequential frames, without the rendering of the skipped frames. For example, in the slow render scenario of FIG. 1, the second frame is generated or ticked as the skipped frame after the rendering of the first frame. This generating step can be performed responsive to the determining the number of subsequent frames to skip rendering step. Next, the method includes generating a corresponding visual content based on a frame of the plurality of frames subsequent to a last skipped frame. The frame subsequent the last skipped frame is the frame that will be the first frame ticked and rendered after the one or more frames that were skipped are ticked, but not rendered. For example, in the slow render scenario of FIG. 1, the frame subsequent the last skipped frame is the third frame that is ticked (i.e., the frame that begins in the third frame period), and the last skipped frame is the second frame that is ticked. Finally, the method includes rendering the corresponding visual content subsequent to the last skipped frame. For example, in the slow render scenario of FIG. 1, the third frame (or the frame subsequent the last skipped frame) is shown to be rendered.

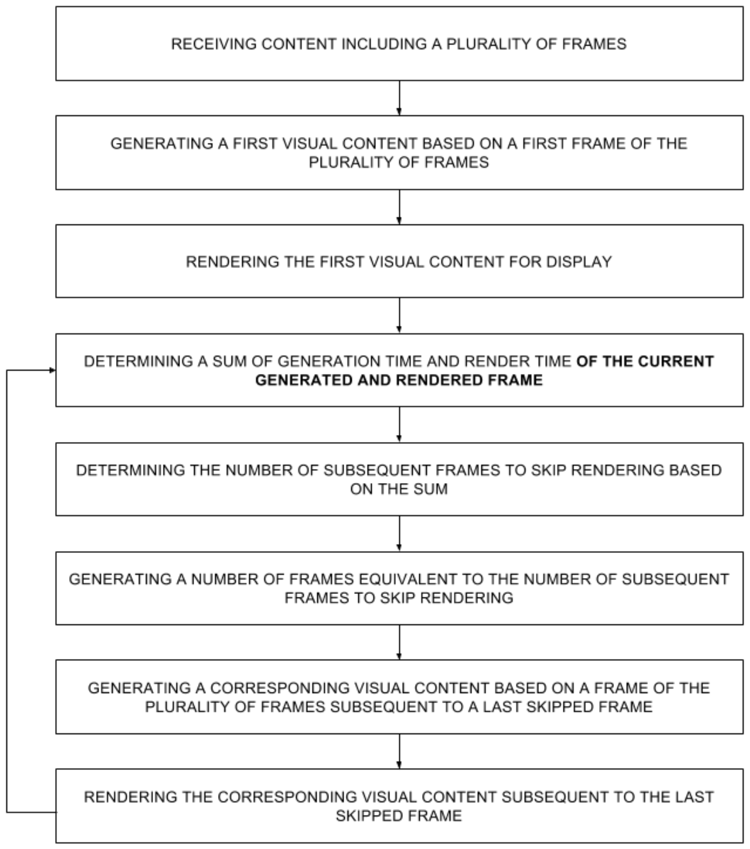
RECEIVING CONTENT INCLUDING A PLURALITY OF FRAMES
GENERATING A FIRST VISUAL CONTENT BASED ON A FIRST FRAME OF THE PLURALITY OF FRAMES
RENDERING THE FIRST VISUAL CONTENT FOR DISPLAY
DETERMINING A SUM OF GENERATION TIME AND RENDER TIME OF THE CURRENT GENERATED AND RENDERED FRAME
DETERMINING THE NUMBER OF SUBSEQUENT FRAMES TO SKIP RENDERING BASED ON THE SUM
GENERATING A NUMBER OF FRAMES EQUIVALENT TO THE NUMBER OF SUBSEQUENT FRAMES TO SKIP RENDERING
GENERATING A CORRESPONDING VISUAL CONTENT BASED ON A FRAME OF THE PLURALITY OF FRAMES SUBSEQUENT TO A LAST SKIPPED FRAME
RENDERING THE CORRESPONDING VISUAL CONTENT SUBSEQUENT TO THE LAST SKIPPED FRAME

FIG. 2

ABSTRACT

This paper describes methods and systems for dynamically managing rendering of animation content. At present, some types of internet display advertisements require animation to occur in the client (e.g., browser) to maintain the appropriate quality. However, rendering can be costly to computer resources and therefore to performance, quality, load times, etc. Animations typically have a fixed frame rate (e.g., 24 frames per second), which results in advertisements taking longer than desired to load if the rendering is slow, which can cause latency and degradation of user experience. This paper discusses methods and systems for dynamically managing rendering of animation content. The system can drop some select frames from being rendered if not doing so would consume more computing resources than desired. In one implementation, if the render time plus the frame generation time is less than the frame time (based on the fixed frame rate) itself, the content will be animated normally. However, if the render time plus the tick time is greater than the frame time itself, meaning that there is now

some latency, the system will calculate the excess time to generate and render the current frame and not render some of the succeeding frames based on the calculated excess time.