July 01, 2016

# DYNAMICALLY CACHING VIDEO CONTENT BASED ON USER LIKELIHOOD OF INTERACTING WITH CONTENT ELEMENT

Richard Rapp

Justin Lewis

# DYNAMICALLY CACHING VIDEO CONTENT BASED ON LIKELIHOOD OF USER INTERACTION WITH CONTENT ELEMENT

When the video content of a webpage loads slowly, the loading of other content on the webpage may also be delayed. These delays may discourage the user from returning to the webpage and accessing other content on the webpage. This issue may be exacerbated if bandwidth and computing processing power is limited. For example, mobile devices usually have decreased bandwidth and less processing power than desktop and laptop computers. Similarly, emerging markets typically have lower quality networking infrastructure and older client-side hardware.

One presently implemented approach to mitigate slow loading video content may be to pre-cache all the video content on the webpage prior to playing the cached video content. However, not every video content element may be interacted with or played by the user. As such, some of the video content may have been pre-cached unnecessarily. Furthermore, pre-caching all the video content may lead to a degradation of the user's experience with the webpage. Pre-caching may consume computing resources. This may be especially problematic with mobile devices, in which data may be capped and power may be limited. In addition, pre-caching all the video content may still delay the loading of other content on the webpage.

To address these and other technical challenges, this paper proposes techniques for dynamically caching video content based on the likelihood of user interaction with a content element on a webpage. In general, two aspects of the user interaction may be used as proxies for the likelihood of user interaction with the content element: content visibility stability and hovering selection. The content visibility stability aspect concerns whether a specific content element remained within a viewport for a sufficient period of time. The hovering selection aspect concerns whether there is an interaction event on the content element. Both of these aspects may be taken into account to pre-cache video content determined to be within the viewport that are likely in the user's focus, instead of pre-caching all of the video content on the webpage.

The proposed techniques may make use of certain event listeners or handlers on the webpage or the constituent content element that is to contain video content to trigger the pre-caching of the first few seconds of the video content before playback. Examples of such types of events detected by event listeners or handlers include: (1) mouse down or press and hold, (2) hover, and (3) scroll stop. The duration of these events may be as short as a fraction of a second. This fraction of a second, however, may provide enough time to pre-cache some of the video content before the video content is played, and may significantly reduce latency of the video content playback, thereby improving the user's experience of the webpage. These events may have a variable degree of correlation with the likelihood of the user interacting with the content element. Furthermore, the content elements themselves may have various estimated click through rates and view rates, independent of these events. Which event listener is selected and applied may be based on a variety of factors, such as computing device type, application profile, network bandwidth, storage on computing device, number of videos on the webpage, etc.

One of the event listeners that may be used to determine whether to pre-cache video content may be the mouse down event, corresponding to the "onmousedown()," "onTouch()," or "onLongClick()" event listener functions in HyperText Markup Language (HTML) scripts. The pre-caching functionality may be embedded in the mouse down event listener of the video content element of the webpage, as depicted in Fig. 1A. Further details of the pre-caching functionality of the mouse down event listener are illustrated in Fig. 1B.

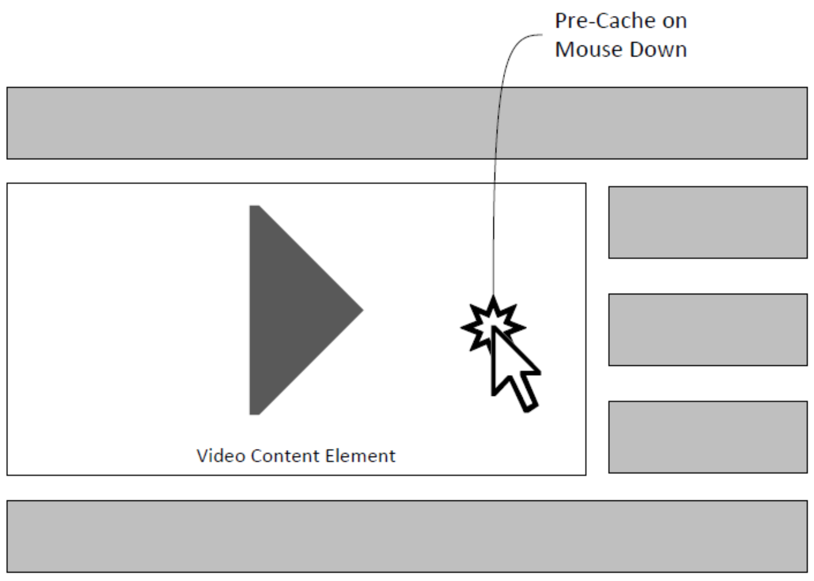

FIG. 1A

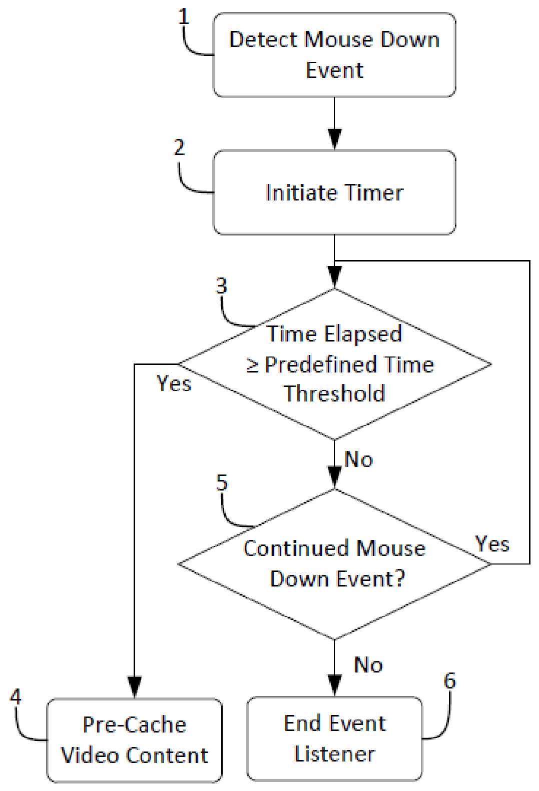

FIG. 1B

At step 1, the computing device can detect a mouse down event on the content element. The content element may be a video content element, specifying a video source. The mouse down event may correspond to a pressing of a mouse button or the display screen over the content element. The triggering of this event may correspond to user interest in the video content.

At step 2, the computing device can initiate a timer in response to detecting the mouse down event. The event listener may make use of the mouse down timer (e.g., “downTimer”) to track the amount of time the mouse button or the display screen is pressed over the content element. At step 3, the computing device can determine whether the time elapsed is greater than or equal to a predefined time threshold. The predefined time threshold may be fractions of a second, and may be pre-set based on a variety of factors, such as the computing device type, application profile, network bandwidth, storage on computing device, or the number of videos on the webpage.

At step 4, if the time elapsed is greater than or equal to the predefined time threshold, the computing device can pre-cache the video content of the content element. The computing device may request the server identified in the content element for the first few seconds or chunks of the video content. At step 5, if the time elapsed is less the predefined time threshold, the computing device can determine whether the mouse down event is continued. If the mouse down event is still triggered, the computing device can repeat the functionality of step 3. At step 6, if the mouse down event is terminated, the computing device can terminate the event listener. In such instances, the mouse down event may be too short to warrant pre-caching of the video content.

Another event listener that may be used to trigger the pre-caching of video content may be the hover event, corresponding to the "mouseover()" or "onHover()" event listener functions of HTML scripts. The pre-caching functionality may be embedded in the hover event listener of the video content element of the webpage, as shown in Fig. 2A. Further details of the pre-caching functionality of the hover listener are depicted in Fig. 2B.

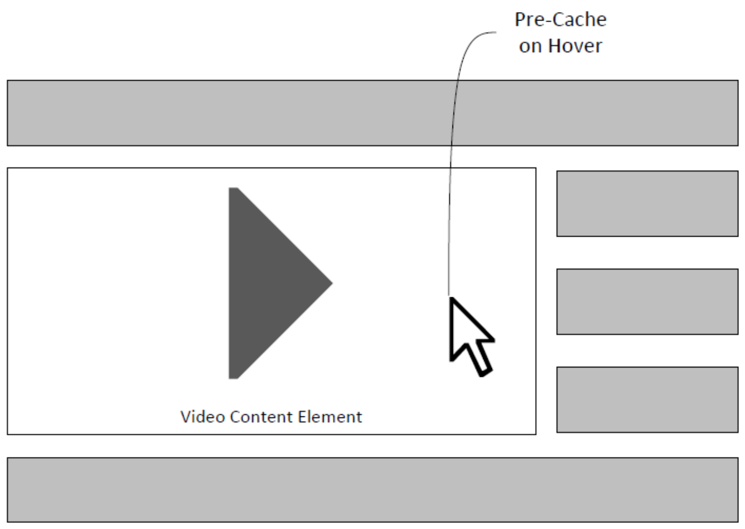

FIG. 2A

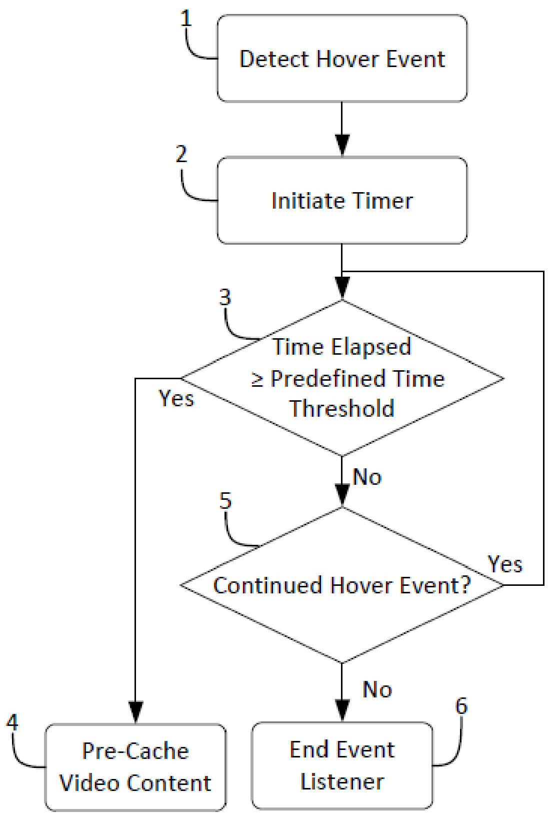

FIG. 2B

At step 1, the computing device can detect a hover event on the content element. The hover event may correspond to moving a mouse selection over the content element. The event may also correspond to the moving of a user's finger over the area of the content element on the display screen. Similar to the mouse down event, the triggering of the hover event may correspond to user interest in the video content.

At step 2, the computing device can initiate a timer in response to detecting the hover event. At step 3, the computing device can determine whether the time elapsed is greater than or equal to a predefined time threshold. For example, the computing device can set a timeout (e.g., "setTimeout") as the predefined time threshold for the amount of time prior to invoking the event listener again or other functions (e.g., "callback"). Similar to the case of the mouse down event, the predefined time threshold may be fractions of a second, and may be pre-set based on a variety of factors, such as the computing device type, application profile, network bandwidth, storage on computing device, or the number of videos on the webpage.

At step 4, if the time elapsed is greater than or equal to the predefined time threshold, the computing device can pre-cache the video content of the content element. The computing device may request the server identified in the content element for the first few seconds or chunks of the video content. At step 5, if the time elapsed is less the predefined time threshold, the computing device can determine whether the hover event is continued. If the hover event is still triggered, the computing device can repeat the functionality of step 3. At step 6, if the hover event is terminated, the computing device can terminate the event listener. In such instances, the hover event down event may be too short to warrant pre-caching of the video content.

The scroll stop event listener (e.g., "scrollstop" in HTML scripts) may also be used to trigger the pre-caching of video content in a hover event. The pre-caching functionality may be embedded in the scroll down event listener of the webpage. Further details of the pre-caching functionality of this scenario are depicted in Figs. 3A and 3B.

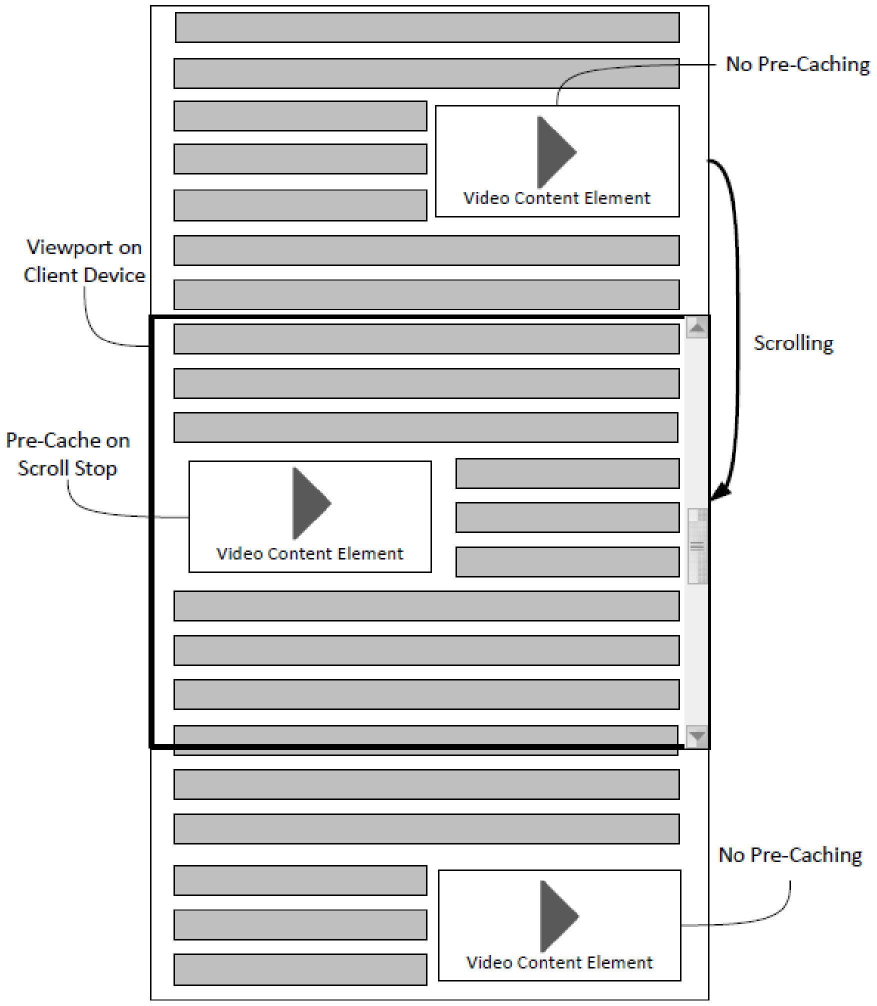

FIG. 3A

In brief overview, all the video content of the content elements on the webpage may initially be not pre-cached. Upon detecting an on scroll event (e.g., "onscroll," "onTouch," etc.) on the webpage, the computing device can monitor for a scroll stop event. Once the scroll stop event is detected on the webpage, the computing device can then instantiate a timer and compare the time to a predefined threshold to determine whether the user is interested in the content element now within the viewport of the web browser in the display of the computing device. If there is no further scroll event prior to the elapsing of the predefined threshold, the computing device can pre-cache the video content of the content element within the viewport while the other content elements outside the viewport remain un-cached.

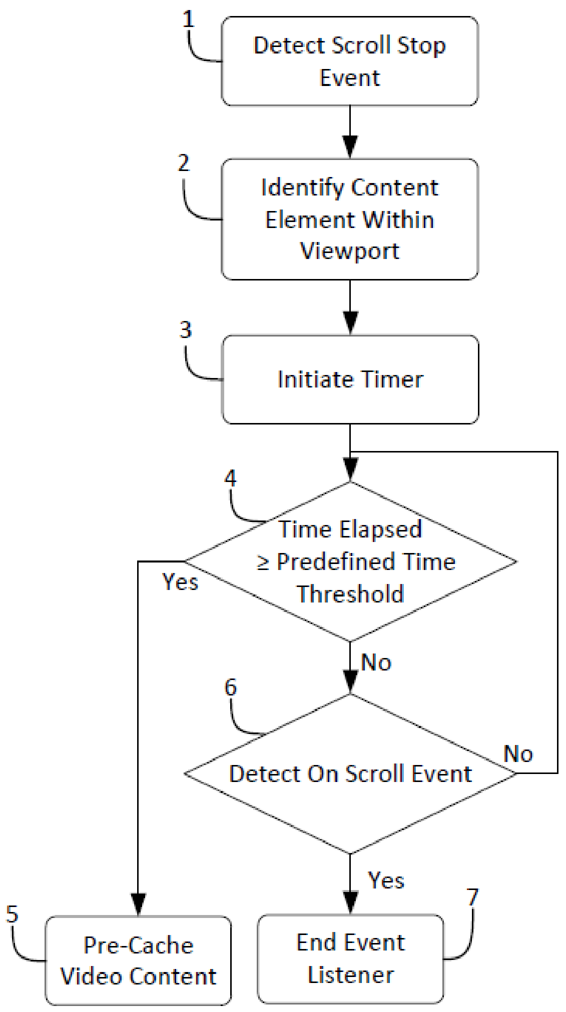

FIG. 3B

In further detail, at step 1, the computing device can detect a scroll stop event on the webpage subsequent to an on scroll event. The scroll stop event may correspond to a lack of any other scroll events on the webpage. The scroll stop event may also correspond to the lack of a user's finger touching or over the display of the computing device. Similar to the mouse down and hover events, the triggering of the scroll stop event may be a proxy to user interest in the video content on the content element within the viewport.

At step 2, the computing device can identify the content elements that may include video content within the viewport. The computing device can retrieve coordinates of all the content elements on the webpage and coordinates of the viewport. Based on comparing the coordinates of the content elements and the viewport, the computing device can ascertain which content elements that may include video content are within the viewport of the computing device.

At step 3, the computing device can initiate a timer in response to detecting the hover event. At step 4, the computing device can determine whether the time elapsed is greater than or equal to a predefined time threshold. For example, the computing device can set a timeout (e.g., "setTimeout") as the predefined time threshold for the amount of time prior to invoking the event listener again or other functions (e.g., "callback"). The predefined time threshold may be fractions of a second, and may be pre-set based on the same variety of factors as the mouse down and hover events.

At step 5, if the time elapsed is greater than or equal to the predefined time threshold, the computing device can pre-cache the video content of the content element within the viewport. The video content of the content elements outside the viewport can remain un-cached. The computing device may request the server identified in the content element for the first few seconds or chunks of the video content. At step 6, if the time elapsed is less the predefined time threshold, the computing device can determine whether an on scroll event is detected. If no on scroll event is detected, the computing device can repeat the functionality of step 4. At step 7, if the scroll stop event is terminated and the on scroll event is detected, the computing device can terminate the scroll stop event listener. In such instances, the scroll stop down event may be too short to warrant pre-caching of the video content.

By utilizing certain event listeners to pre-cache video content prior to playback, the techniques proposed herein may reduce latency of the playing of the video content and loading of other content on the webpage, thereby improving the user's experience of the webpage and the computing device.

## ABSTRACT

We propose techniques of dynamically caching video content of content elements of webpages. The techniques may leverage certain event listeners, such as mouse down, hover, and scroll stop, among others, to dynamically trigger the pre-caching of video content on the webpage. These events may correlate with the likelihood of user interaction with the content element containing the video content. Once one of these event listeners is detected, the computing device may initiate a timer for the duration of the event. If the timer exceeds a predefined threshold, first few seconds the video content of the content element may be pre-cached. These techniques may enhance loading of other content on the webpage and reduce latency of the playing of the video content, resulting in an improvement to the user's overall experience with the webpage.