# Using MPEG DASH SRD for zoomable and navigable video

Lucia D'Acunto
TNO
Anna van Buerenplein 1
The Hague, The Netherlands
lucia.dacunto@tno.nl

Jorrit van den Berg
TNO
Anna van Buerenplein 1
The Hague, The Netherlands
jorrit.vandenberg@tno.nl

Emmanuel Thomas
TNO
Anna van Buerenplein 1
The Hague, The Netherlands
emmanuel.thomas@tno.nl

Omar Niamut
TNO
Anna van Buerenplein 1
The Hague, The Netherlands
omar.niamut@tno.nl

## ABSTRACT

This paper presents a video streaming client implementation that makes use of the Spatial Relationship Description (SRD) feature of the MPEG-DASH standard, to provide a zoomable and navigable video to an end user. SRD allows a video streaming client to request spatial subparts of a particular video stream, which might be available in multiple resolutions.

The paper outlines the design choices enabling the player to render DASH content supporting the SRD feature, such as (i) identifying the total amount of resolution layers and selecting the most appropriate one for the user's current selection (pan or zoom), and (ii) enabling a seamless switch between spatial subparts. Doing so, we provide practical implementation guidelines for applications and services that may want to use the SRD feature of MPEG DASH to provide zoomable and navigable video to end users.

The video streaming client is implemented in JavaScript and extends *dash.js*, an MPEG DASH reference client implementation.

## CCS Concepts

•**Information systems** → **Multimedia streaming;**

## Keywords

video, mobile video, streaming, javascript, dash.js, MPEG-DASH, SRD, tiled streaming, standards

## 1. INTRODUCTION

Triggered by the advent of higher resolution video (4K, 8K and beyond) and the increase in usage of mobile devices for video consumption, a new feature of the MPEG DASH standard, referred to as Spatial Relationship Description (SRD) [4], has recently been published. The feature extends the Media Presentation Description (MPD) specified in part 1 of MPEG DASH by describing spatial relationships between associated pieces of video content. This provides DASH clients with the possibility to select and retrieve only those video streams at those resolutions that are relevant for the users watching the video.

However, as it holds in general, MPEG DASH does not provide information on client logic implementation of the SRD feature. For example, it does not explicitly indicate how many multiple resolution layers of the video stream are offered, and how many spatial subparts are available at each resolution layer. Similarly, there are no specific instructions on how to enable a seamless switch among the related pieces of video content.

In this paper, we provide practical guidelines on how a client application can effectively use the SRD feature of the MPEG DASH standard to provide zoomable and navigable video to end users. On the basis of these guidelines, a video streaming client implementation of the SRD feature is proposed. The video streaming client is implemented in JavaScript and extends *dash.js*, an MPEG DASH reference client implementation [1]. The source code of the video streaming client, referred to as the *SRD video player*, is available at `https://github.com/tnomedialab/dash-srd.js`. By providing implementation guidelines and source code, we aim at aiding the adoption of this technology within the media industry. As such, we also expect that our work will provide beneficial input for industry bodies supporting the adoption of the MPEG DASH standard.

## 2. BACKGROUND

In this section, we first describe the concept behind zoomable and navigable video. Then, we outline the salient characteristics of the SRD feature. Finally, we touch upon *dash.js*, a reference implementation of the MPEG DASH player.

### 2.1 Tiled Streaming

The application of zoomable and navigable video sees its roots in the concept of interactive *region-of-interest (ROI) video streaming*. While several approaches to interactive ROI streaming exist, the approach known as *tiled streaming* [6] offers the best compromise between bandwidth, storage, processing and device requirements, in particular for large-



*MMSys'16 May 10-13, 2016, Klagenfurt, Austria*



ACM ISBN 978-1-4503-4297-1/16/05.

DOI: http://dx.doi.org/10.1145/2910017.2910634

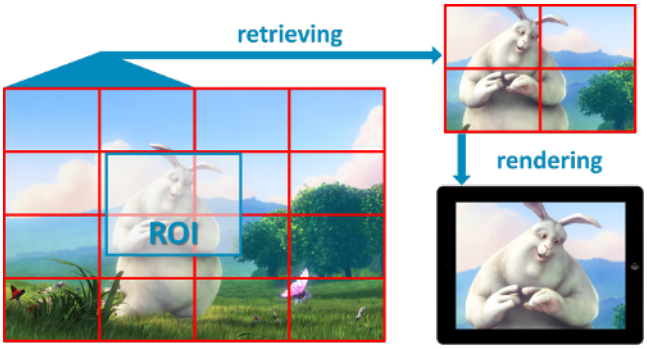


**Figure 1: Tiled streaming: retrieving and rendering the ROI.**

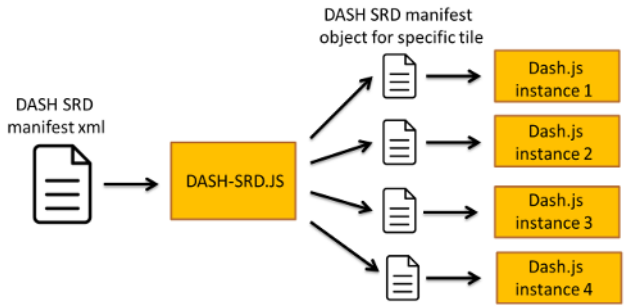


**Figure 2: Architecture of the *SRD player*. The SRD player is implemented as a wrapper around *dash.js*: for each spatial subpart that needs to be displayed to the user, a new instance of the *dash.js* player is created.**

scale deployments. Essentially, "tiling" refers to partitioning a video stream into a number of independently decodable video streams or "tiles". The idea behind tiled streaming is that, if a user is only interested in viewing a certain ROI, the video player does not need to retrieve the full view of the video stream, but only those "tiles" corresponding to the ROI selected by the user, hence saving bandwidth in the process (Figure 1). For zoomable video, multiple resolution layers are created from the original source video. Each additional layer originates from a lower resolution version of the original video frame, tiled into a grid with fewer tiles. This multi-resolution tiling increases the quality of user-defined zooming on tiles. The nature of tiled streaming makes it particularly useful within the context of *Adaptive Bitrate Streaming*, where the content is typically made available at multiple bitrates and resolutions. When a user selects a certain ROI in the video, e.g. with zoom or navigation commands, the player can decide to download the tiles making up the ROI at a higher resolution, therefore increasing the level of detail in the video as shown to the user.

## 2.2 MPEG-DASH SRD

DASH is the MPEG standard for Adaptive Streaming over HTTP [5]. The standard defines a Media Presentation Description (MPD), namely an XML manifest file that describes the media content available (*AdaptationSets*) including its alternate versions (*Representations*) exhibiting different bitrates, resolutions, or codecs.

MPEG has recently standardised a new feature of the MPD, called Spatial Relationship Description (SRD) [4], which associates spatial information with the media content contained in an MPD. Specifically, each content item included in the MPD might be enhanced with an SRD, which is composed of the following parameters: `source_id`, `object_x`, `object_y`, `object_width`, `object_height`, `total_width`, `total_height` and `spatial_set_id`. The `source_id` identifies a particular content item in the stream (e.g. one TV program); `object_x`, `object_y`, `object_width`, `object_height` provide the position and size of a subpart of this particular content (i.e. a tile for tiled streaming) within a coordinate system; `total_width` and `total_height` represent the size of this coordinate system; the `spatial_set_id` is an identifier for a (sub)group of the subparts relating to the particular content (e.g. all spatial subparts belonging to a given resolution layer). The first five SRD parameters are mandatory, `total_width` and `total_height` are conditional mandatory (meaning that they are required to be present only under certain circumstances) and `spatial_set_id` is optional. For more details see [4].

## 2.3 dash.js

*dash.js* is a MPEG DASH reference client implementation in JavaScript. Its development was initiated by the DASH Industry Forum (DASH-IF) with the aim of establishing "a production quality framework for building video and audio players that play back MPEG-DASH content" [1]. The client is OpenSource and publicly available on GitHub under a BSD-3 license. Furthermore, this client also implements best practices in the playback of MPEG DASH content provided by DASH-IF [3]. Given the features of the player and the goal of the *dash.js* project, we have chosen to implement our SRD video player as a wrapper around *dash.js* (Figure 2).

# 3. DESIGN AND IMPLEMENTATION OF THE SRD PLAYER

## 3.1 Motivation and Objectives

SRD was designed to enable a variety of use cases and as such it has a flexible blueprint. However, the MPEG DASH specification does not provide reference client logic implementation guidelines for the features it contains, and this also holds true for SRD. In this section, we present the design choices we have made to use SRD in the context of zoomable and navigable video.

## 3.2 Identifying Resolution Layers

In order to handle video content offered at multiple resolutions, the first step is to determine how many resolution layers there are. Then, the spatial subparts belonging to each resolution layer need to be identified. The number of subparts per resolution layers is not explicitly provided in the SRD feature and the client needs to infer it from the SRD parameters. However, if only the mandatory parameters are present, the client will need to parse the entire MPD, compare the sizes and positions of all the spatial subparts, and then infer the total number of resolution layers and the resolution layer to which a specific subpart belongs to. A basic assumption is that all the subparts at a given resolution layers have the same size; in an operational environment,

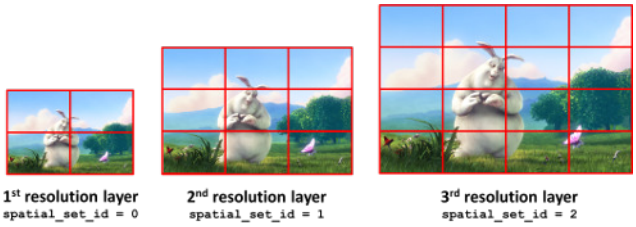


**Figure 3: Identifying resolution layers using the parameter `spatial_set_id`.**

content providers may deviate from this assumption. The process of determining the number of subparts per resolution layer can be readily simplified by providing the optional and conditional mandatory SRD parameters to *all* content items in the MPD. These parameters specify the total size of the coordinate system and allow for grouping all the spatial subparts belonging to the same resolution layer under the same `spatial_set_id`. Furthermore, a special semantic can be given to the value of the `spatial_set_id` in order to order the different resolution layers. Specifically, the spatial (sub)parts belonging to the lowest resolution layer will have the value of the `spatial_set_id` set to `0`, those belonging to the second resolution layer will have value `1` and so on (Figure 3). The total number of resolution layers can then be easily computed, as it corresponds to `1 +` the highest `spatial_set_id` for a given content (identified by the parameter `source_id`).

To summarize, the following design choices have been made with regards to identifying resolution layers:

- Both mandatory and optional SRD parameters are always used;
- The parameter `spatial_set_id` identifies all subparts belonging to the same resolution layer, and is used to order the resolution layers from the lowest to the highest, the lowest starting from value 0.

It is important to notice that these design choices need to be supported by the content provider as well as by the DASH client for the mechanism described above to work.

## 3.3 Switching Among Spatial Subparts

Zoomable and navigable video applications typically will require the client to frequently switch among spatial subparts. To guarantee a good user experience when performing these actions, the player needs to implement a *seamless switch*. A seamless switch means a gap-less transition in both space and time from the originally presented spatial content to the new one.

### *3.3.1 Seamless Switch in Space*

To enable a seamless switch in space, the client must ensure that during the switch, some content keeps being displayed to the user.

For a zoom-in action, a new chunk of content needs to be fetched corresponding to the selected region of interest at a higher resolution. In our design, the client will upscale the current stream and display it to the user, until the new chunck of content has been retrieved, decoded, and made available for rendering. This way, once the chunk of content

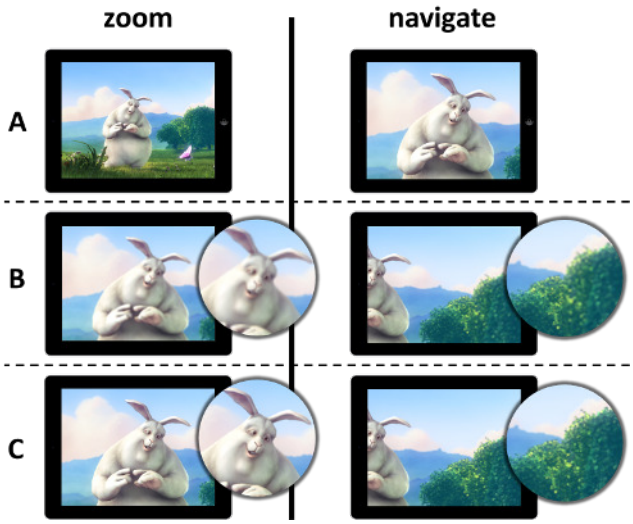


**Figure 4: Seamless switch in space: the left column shows a zoom action, while the right column a navigation action. *A* corresponds to the situation before the switch. *B* pictures the temporary state when the original stream (in the right case combined with the fallback layer) is displayed to the user in the meantime that the new stream is being fetched. *C* represents the situation after the switch has been completed and the new stream is displayed to the user.**

corresponding to the higher resolution is fetched, it can be readily displayed to the user with minimal disturbance (Figure 4). Note that this switch is visually similar to a regular DASH switch between two Representations.

Enabling a similar seamless switch when the user navigates to other regions of interest (panning) requires some additional effort. In particular, these regions of interest might be partially or completely outside of the current view being displayed to the user. To support this scenario, the SRD player always downloads the content in the first resolution layer (which contains the full view of the scene) and downloads in parallel the content in the resolution layer currently displayed to the user. In this way, a version of the content from the first resolution layer scaled to the resolution of the requested content can be temporarily displayed if needed, hence avoiding a black screen effect. In the background, the client still fetches the chunk of content corresponding to the new region of interest requested by the user (Figure 4).

Always downloading the full view also serves the purpose of bandwidth adaptation: when the available bandwidth is not sufficient to stream the higher resolution content, the client can immediately "fall back" to that resolution layer. For this reason the first resolution layer is also referred to as *fallback layer*.

Recapitulating, the design choices regarding the seamless switch in space are:

- the "fallback layer" is always downloaded in parallel with the resolution layer to be displayed; and
- the "fallback layer", resized according to the resolution layer of the content to be displayed, is being presented

to the user until the requested chunk of content has been fetched.

In contrast to the case of the identification of the resolution layers, the above design choices only apply to the DASH client's behavior.

#### 3.3.2 Seamless Switch in Time

When a spatial subpart is replaced by another one (or more) spatial subpart(s), the client must ensure that the timing of presentation of these new subparts is (i) aligned with the time of the presentation of the current spatial subpart, as well as (ii) within the new subparts themselves. DASH already offers support for the first aspect, by introducing so-called "Stream Access Points" in the video streams [5]. The second aspect is intrinsic to tiled streaming application and it requires the player to have frame-level control of the video stream in order to guarantee that the playback of the different spatial subparts is tightly synchronised. Currently, the HTML 5 specification does not explicitly offer this functionality, but only specifies that the granularity of access to the video stream shall be between the 15ms and 250ms [7]. Most browsers have chosen for 250ms, with the exception of Mozilla Firefox which offers frame-level access for videos with framerate not higher than 66frames/s [2]. This value defines the level of accuracy that synchronization among video streams can achieve. With these limitations, the player currently focuses on ensuring that:

- the functions starting the playback of the different spatial subparts are called in parallel at the same moment;
- periodically, the different spatial subparts being displayed (if any) are resynced with the fallback layer as reference.

As for the case of seamless switch in space, these design choices only apply to the DASH client behavior.

## 4. DEMONSTRATION

In this section we describe the details of the demo that will be given to the MMSys conference visitors.

### 4.1 Setup

Prior to the conference, tiled video content will be made available at a webserver and at a local machine. Then, on the basis of the video content, a DASH manifest file will be created where the different video subparts are signalled throught the SRD feature of DASH, as described in Section 3. A webpage with the SRD player will also be setup. This webpage accepts a DASH manifest file as input and feeds it to the SRD player.

At the conference, a laptop and several iPads will be made available for visitors to interact with the zoomable and navigable video content.

### 4.2 Content

For the purpose of the demo, we have collected our own video material, tiled it and encoded at different qualities.

### 4.3 Interaction Opportunities for Visitors

After a brief introduction of the application and its architecture, visitors will be given the possibility to interact with the tiled video content through the SRD player on the laptop and/or iPads. We aim at collecting visitors' feedback on their experience with the demo, in order to further improve the SRD player.

## 5. CONCLUSIONS AND FUTURE WORK

In this paper, an implementation of a DASH client supporting the SRD feature is presented. The specific use case of zoomable and navigable video is addressed, and the design choices made to effectively use SRD for this application are illustrated. In particular, a specific semantic for the SRD parameter `spatial_set_id` has been introduced to enable an optimal management of the various resolution layers, and the presence of all SRD parameters is advocated. In order to handle a seamless switch, the player makes use of the "fallback layer", i.e. the resolution layer relative to the content representing the entire view of the video scene. This paper makes the recommendation to have the client always download the fallback layer while operating, to allow seamless switches between spatial subparts of the content. Finally, we do notice that frame-level access to the video streams should be provided in the web browser API to improve seamless switching in time.

Future work will be focused on raising awareness within relevant standardization bodies (e.g. W3C) regarding frame-level video controls as well as aiding the adpotion of MPEG DASH and its SRD feature in the media industry (by means of demonstrations to the public, for example).

## 6. ACKNOWLEDGMENTS

The authors would like to acknowledge the contributions of their colleagues, in particular Ray van Brandenburg, Oskar van Deventer and Arjen Veenhuizen, as well as the many colleagues in MPEG DASH for the collaboration on the matter and their contributions to a hopefully successful and widely deployed standard.

## 7. REFERENCES

[1] dash.js: A reference client implementation for the playback of mpeg dash via javascript and compliant browsers. https://github.com/Dash-Industry-Forum/dash.js/wiki Accessed: 2016-02-04.

[2] Fire timeupdate event less frequently than once per frame. https://bugzilla.mozilla.org/show_bug.cgi?id=571822 Accessed: 2016-02-12.

[3] DASH Industry Forum. "Guidelines for Implementation: DASH-IF Interoperability Points", 2015.

[4] ISO/IEC 23009-1:2014/Amd 2. "Spatial relationship description, generalized URL parameters and other extensions", 2015.

[5] ISO/IEC JTC1/SC29/WG11 W13533. "MPEG DASH: The Standard for Multimedia Streaming over the Internet", 2012.

[6] Niamut O., Prins M., van Brandenburg R., and Havekes A. "Spatial tiling and streaming in an immersive media delivery network". *Adjunct Proceedings of EuroITV*, 2011.

[7] W3C. "HTML5.1, Editor's Draft, 17 February 2016", 2016.