Span background height in EBU-TT-D

Chris Bass · BBC Research & Development

1. Problem description

1.1 Content Provider requirements

Content Providers like the BBC will need to use spans with opaque backgrounds in their EBU-TT-D subtitles. This is particularly important for live programmes, where subtitles are delivered word-by-word (Figure 1), since the alternative of setting a background colour on the containing paragraph would result in the background behind each line extending to the entire region width (minus padding), even for lines that are incomplete. This would unnecessarily obscure parts of the underlying video, as demonstrated in Figure 2.

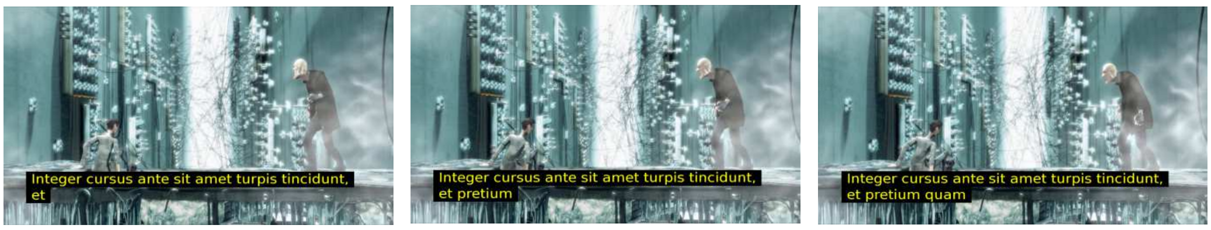
Integer cursus ante sit amet turpis tincidunt,
et
Integer cursus ante sit amet turpis tincidunt,
et pretium
Integer cursus ante sit amet turpis tincidunt,
et pretium quam

Figure 1 Live subtitles appearing word-by-word.

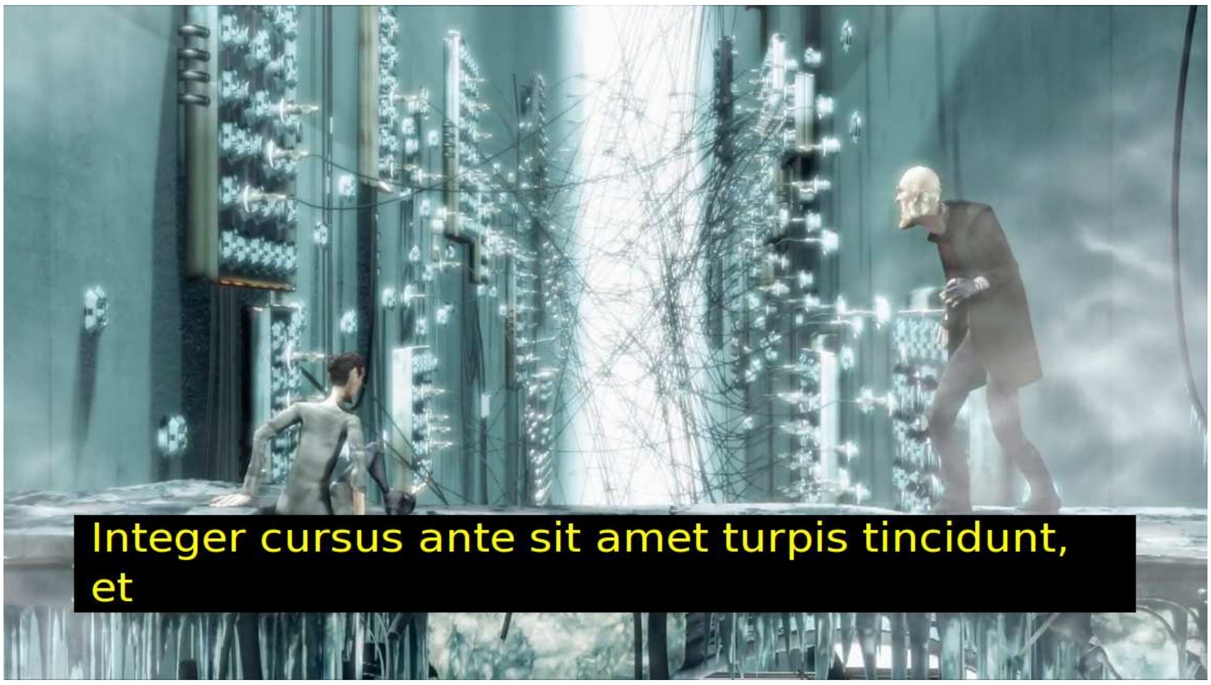

Figure 2 Live subtitles with `tts:backgroundColor` applied to paragraph.

For subtitles delivered in spans with background colours, it is important that the height of the rendered backgrounds extend to fill the full height of each line, leaving no vertical gaps between the backgrounds of consecutive lines. This is vital because having vertical gaps between lines of subtitles through which moving video is displayed reduces the readability of those subtitles for viewers who rely upon them (not to mention the fact that it simply looks bad).

1.2 TTML/EBU-TT-D behaviour

Unfortunately, it appears that a correct implementation of the TTML/EBU-TT-D specs would result in vertical gaps between lines. In a strict implementation of the specs, the area filled by the background colour behind a span would not extend beyond the text height[1]; therefore, (a) whenever `tts:lineHeight` is greater than 100% of `tts:fontSize`, there will be gaps between lines, and (b) the background would be virtually flush with the top and bottom of the font glyphs.

1.2.1 XSL-FO *rendering*

Figure 3 below shows the rendering of a XSL-FO representation of a TTML span (as rendered by the Apache FOP XSL-FO renderer). As expected, the span backgrounds do not extend beyond the content, leaving gaps between the lines.

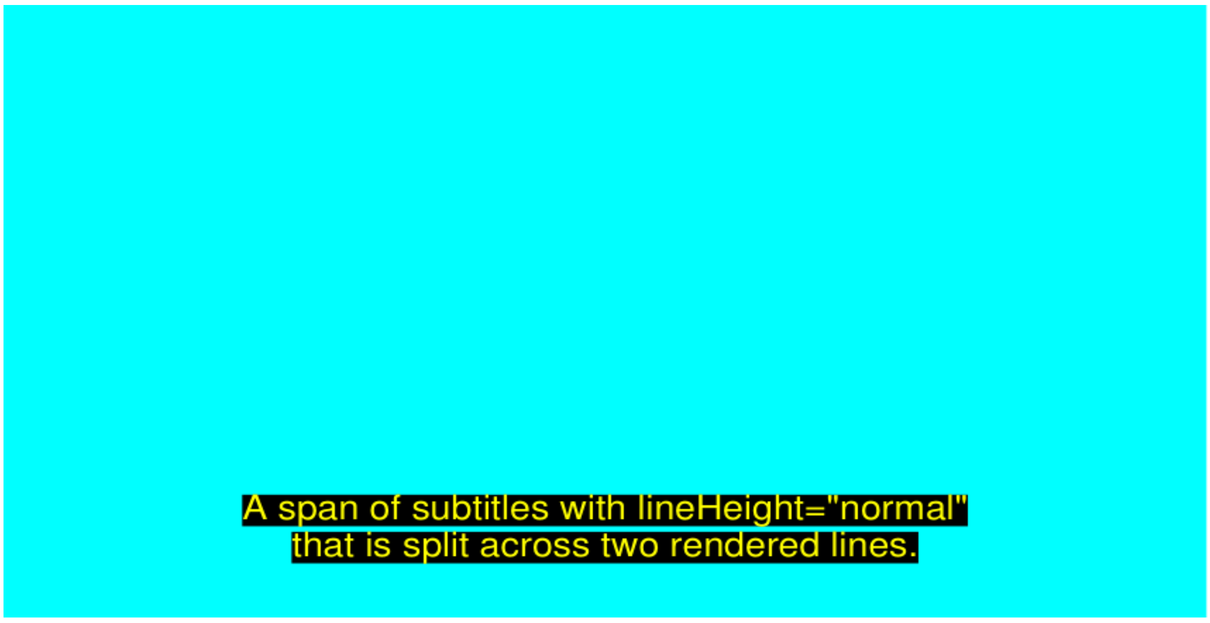
A span of subtitles with lineHeight="normal"
that is split across two rendered lines.

Figure 3 Rendering of a TTML span as represented by XSL-FO objects.

1.2.2 HTML/CSS *rendering*

One common approach to implementing a TTML/EBU-TT-D client is by converting subtitles into HTML form and then using an HTML/CSS browser to render that content.

[1] TTML is defined in terms of XSL-FO objects. According to the XSL spec, '*The background, if any, is rendered in the padding-rectangle, in accordance with the background-image, background-color, background-repeat, background-position-vertical, and background-position-horizontal traits*' (4.9.4). The spec defines the default values of the padding-top and padding-bottom traits to be 0pt (7.8.35 and 7.8.36, respectively), and the TTML spec does not override these default values; therefore, the padding rectangle in the case of spans is coincident with the content rectangle.

The default behaviour of HTML/CSS (as implemented in current browsers) is similar but not identical to XSL-FO: as with XSL-FO, it will *not* extend span backgrounds to the full line height, but unlike XSL-FO it will extend the background a short distance above and below the glyph ascenders and descenders, respectively. The distance by which HTML/CSS implementations extend span backgrounds depends upon the font that is being used, with the result that with some fonts there will be no gaps between lines when `tts:lineHeight="normal"`, whereas for other fonts there will be gaps between lines when `tts:lineHeight="normal"` (see Figure 4 below).

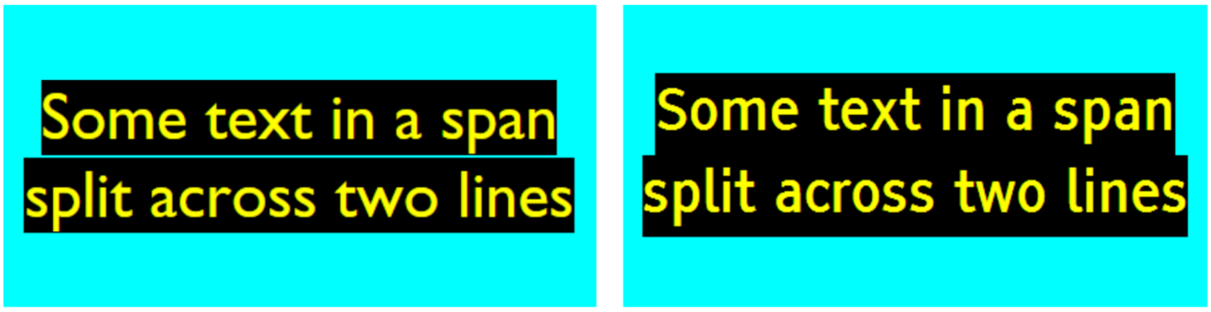

Figure 4 Browser rendering of spans with `tts:lineHeight="normal"` using Gill Sans (left) and Tiresias Screenfont (right) (rendered using Google Chrome).

1.3 Conclusion

The net result of the above is that TTML/EBU-TT-D as it stands does not support BBC's subtitling requirements, as it does not provide a means to author subtitles using arbitrary fonts and line heights with the guarantee that they will be rendered on all compliant clients without inter-line gaps.

2. Proposed solution

2.1 `backgroundHeight` attribute

To rectify this situation, we propose that a new attribute, `backgroundHeight`, be added to the EBU-TT styling namespace. This attribute would allow authors to specify whether the background of spans should (a) extend no further than the content rectangle (as currently), or (b) extend to the full line height.

The `ebutts:backgroundHeight` attribute would be defined as follows:

Values:	`line | content`
Initial:	`line`
Applies to:	`span`
Inherited:	Yes

A value of `line` means that if the background height of a span is less than the calculated line height of its parent paragraph, it should be extended vertically to the calculated line height (with the result that there are no gaps between lines); a value of `content` means that span background should extend to the height of the content within the span (as currently).

2.2 Implementation

There are two broad approaches to implementing an EBU-TT-D renderer. The first is one in which the rendering of EBU-TT-D elements is implemented by the client software itself using low level graphics libraries; the second is where the rendering of elements is done by a third party HTML/CSS browser engine, in order to leverage the similarities between TTML and HTML rendering.

2.2.1 Low-level implementation
It is envisaged that implementing the proposed new behaviour in a client that takes a low-level approach would not be problematic, as by nature such an implementation will have direct control over the height of rendered backgrounds[2]. All that would be needed is some extra code to render the height of backgrounds differently based upon the value of `backgroundHeight`.

2.2.2 HTML/CSS-*based implementation*
Implementing the proposed new behaviour in HTML/CSS would be reasonably straightforward. CSS allows padding to be added to the top and bottom of HTML elements (via the `padding-top` and `padding-bottom` style properties), and since the background colour applies to the text area plus padding, adding extra padding above and below an element will result in an increased background height.

Thus, to avoid gaps between lines, a HTML/CSS-based implementation must find out the rendered height of the spans[3] and add padding to the top and bottom of those spans until the height of their backgrounds match (or slightly exceed) the computed line height.

2.3 Interaction with `tts:padding` in TTML2

The work-in-progress TTML2 specification extends the `tts:padding` attribute to apply to all elements, rather than only to regions; this makes it possible for authors to apply vertical padding to spans, which would result in expanded background heights. Thus, in a hypothetical future version of EBU-TT-D that references TTML2 and exposes this new padding behaviour, there would be the possibility of `tts:padding` and `ebutts:backgroundHeight` competing to control the rendered height of a span background.

Although it might appear that the new padding abilities enabled by TTML2 would render `ebutts:backgroundHeight` obsolete, that would not be the case: for an author to be able to know how much vertical padding should be applied to a span to ensure no inter-line gaps, he or

[2] The BBC GStreamer TTML plugin (https://github.com/bbc/gst-ttml-subtitles) is an example of a low-level client that extends span backgrounds to the calculated line height.
[3] The actual rendered dimensions of lines in a span can, for instance, be retrieved by JavaScript code using the `getClientRects()` method. The code can then calculate the required `padding-top` and `padding-bottom` and apply these values to the span.

she must know the height at which that span will be rendered; but the rendered height of a span will vary across different clients and (as demonstrated above) may depend on the font that is used. So it is still necessary to have an option that instructs the presenting client to extend a span's background to the full line height, based on the actual rendered height of the span that only it knows.

The `ebutts:backgroundHeight` attribute could be made to co-exist with the `tts:padding` capabilities of TTML2 by careful definition of expected behaviour, e.g., by defining `ebutts:backgroundHeight="line"` behaviour such that the if the rendered height of a span *including padding* is less than the calculated line height, then the client should extend the background to the full line height.