M-011

ARIB-TTML と IMSC により記述される字幕の相互変換
Mutual Conversion of Captions Described by ARIB-TTML and IMSC

阿部 晋矢† 藤井 翔子† 小松 佑人† 藤津 智† 藤沢 寛†
Shinya Abe Shoko Fujii Yuto Komatsu Satoshi Fujitsu Hiroshi Fujisawa

1. はじめに

テレビ番組やインターネットの動画配信では，マルチメディアサービスを提供しており，映像，音声の他に字幕データが含まれる場合がある．字幕データは様々な方式が提案され運用されている．日本の放送方式の標準規格である ARIB-TTML[1]は新 4K8K 衛星放送に，インターネットの標準規格である TTML Profiles for Internet Media Subtitles and Captions (IMSC)[2]は OTT 事業者などに，それぞれ広く利用が進んでいる．近年，放送サービスや通信サービスを問わないコンテンツの多メディア展開のために字幕の互換性が求められるが，ARIB-TTML と IMSC には非互換の機能や，互換があっても記述方法が異なる機能など課題がある．本稿では，ARIB-TTML と IMSC で記述される字幕データの相互変換手法を提案する．各方式の互換，非互換の機能を整理し，互換のある機能について記述を変換する．提案手法により ARIB-TTML と IMSC の相互変換結果を確認し，それぞれを字幕レンダラーで再生することにより，表示結果を確認した．

2. ARIB-TTML と IMSC の字幕相互変換

ARIB-TTML と IMSC により記述される字幕の相互変換について紹介する．図 1 に ARIB-TTML と IMSC の相互変換アルゴリズムを示す．ARIB-TTML と IMSC では変換内容が異なる．そのため，入力ファイルに応じて ARIB-TTML から IMSC へ変換するアルゴリズム，IMSC から ARIB-TTML へ変換するアルゴリズムを切り替える．ARIB-TTML と IMSC は互換のある機能と非互換の機能がある．非互換の機能は変換が不可能なため，変換の際にファイル内の記述自体を削除する．互換のある機能については，ARIB-TTML と IMSC で記述方法が共通の箇所と共通でない箇所がある．記述方法が共通でない箇所について記述を変換し，ファイルの相互変換を実現する．

2.1 ARIB-TTML から IMSC への変換

ARIB-TTML から IMSC への変換について紹介する．図 1 のアルゴリズムの通り，非互換の機能に関する記述を削除し，互換のある機能のうち共通でない記述を変換する．ARIB-TTML について，IMSC と非互換の機能を表 1，互換があるが記述が共通でない機能と変換方法の概要を表 2 に示す．

2.2 IMSC から ARIB-TTML への変換

IMSC から ARIB-TTML への変換について紹介する．図 1 のアルゴリズムの通り，非互換の機能に関する記述を削除し，互換のある機能のうち共通でない記述を変換する．IMSC について，ARIB-TTML と非互換の機能を表 3，互換があるが記述が共通でない機能と変換方法の概要を表 4 に示す．

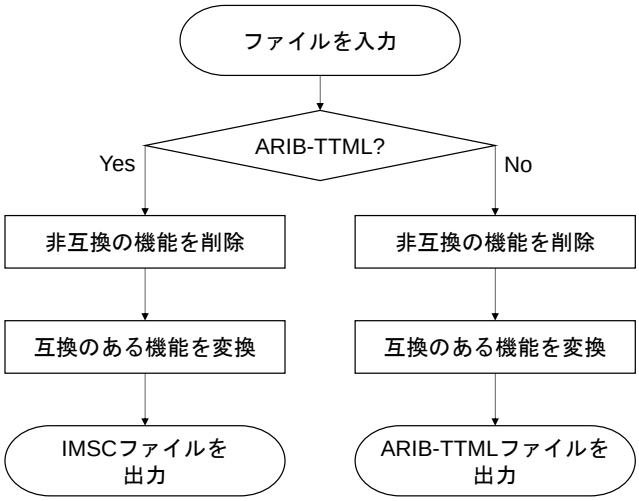

図 1 ARIB-TTML と IMSC の相互変換アルゴリズム

表 1 ARIB-TTML について IMSC と非互換の機能

機能	処理
arib-tt:audio arib-tt:border arib-tt:letter-spacing arib-tt:marquee	該当の要素，属性を削除

表 2 ARIB-TTML について IMSC と互換がある機能の変換方法

機能	変換方法
arib-tt:font-face	IMSC の font 要素と source 要素に置換
arib-tt:keyframes arib-tt:animation	IMSC の set 要素へ置換
arib-tt:text-shadow	tts:textShadow 属性へ置換
arib-tt:ruby	属性を削除

表 3 IMSC について ARIB-TTML と非互換の機能

機能	処理
#contentProfiles #disparity #displayAspectRatio #luminanceGain #metadata-item #shear #activeArea #altText #aspectRatio #forcedDisplay	該当の要素，属性を削除

† 日本放送協会 放送技術研究所
NHK Science & Technology Research Laboratories

表 4　IMSC について ARIB-TTML と互換がある機能の変換方法

機能	変換方法
#font	arib-tt:font-face へ置換
#initial	新しい style 要素を挿入し参照
#textShadow	arib-tt:text-shadow へ置換
#position	対応する位置を算出し region 要素の tts:origin に該当する x 座標，y 座標を指定
#multiRowAlign	表示位置を算出し style，region 要素により指定
#linePadding	行幅を算出し，region 要素の追加，tts:extent 属性を指定
#fillLineGap	行幅を算出し，region 要素の追加，tts:extent 属性を指定
#ruby	ふりがな用の style，region，p 要素を作成し，表示位置，文字サイズを指定
#textEmphasis	強調文字用の style，region，p 要素を作成し，表示位置，文字サイズを指定
#textCombine	表示位置を算出し，該当の文字のみ横書きの style，region を適用

3. 実験結果

提案手法を Node.js によりソフトウェア実装し，動作を検証した．

ARIB-TTML から IMSC への変換として，arib-tt:keyframes, arib-tt:animation のアニメーション機能に関する変換を紹介する．図 2 に ARIB-TTML ファイル，図 3 に IMSC ファイルを示す．IMSC ではアニメーション機能に set 要素を用いるため，図 3 の通り変換される．

IMSC から ARIB-TTML への変換として，#ruby によるふりがなを表示する機能に関する変換を紹介する．図 4 に IMSC ファイル，図 5 に ARIB-TTML ファイルを示す．ARIB-TTML ではふりがなを表示する機能は無いため，ふりがな用に行を新たに用意し，文字サイズと位置を調整することで，図 5 の通り変換される．

図 2，図 3，図 4，図 5 のファイルを Web ブラウザ上で動作する JavaScript による字幕レンダラーで再生し，表示可能なことを確認した．

4. おわりに

本稿では，ARIB-TTML と IMSC の方式の互換，非互換の機能などを整理し，ARIB-TTML と IMSC で記述される字幕の相互変換手法を提案した．また，提案手法をソフトウェアで実装した上で，相互変換結果を確認し，再生することで動作を確認した．今後，互換機能における記述の統一化や非互換な機能の共通化へ向け活動していく．

参考文献

[1] ARIB STD-B62 2.2 版: デジタル放送におけるマルチメディア符号化方式（第 2 世代）, 2019 年 7 月.
[2] "TTML Profiles for Internet Media Subtitles and Captions 1.2," https://www.w3.org/TR/ttml-imsc1.2/.

```
...
<arib-tt:keyframes animationName="blink">
<arib-tt:keyframe position="0%"
tts:color="black"/>
<arib-tt:keyframe position="50%"
tts:color="white"/>
<arib-tt:keyframe position="100%"
tts:color="black"/>
</arib-tt:keyframes>
<style xml:id="s1" arib-tt:animation="blink
2000ms step-end 0ms 2 normal"/>
...
<div xml:id="d0001" begin="00:00:01"
end="00:00:10">
<p xml:id="p0001" region="r1">フラッシングテスト</p>
</div>
...
```

図 2　arib-tt:keyframes，arib-tt:animation 機能を持つ ARIB-TTML ファイル

```
...
<div xml:id="d0001" begin="00:00:01"
end="00:00:10">
<p xml:id="p0001" region="r1" begin="0ms"
tts:color="black">
フラッシングテスト
<set begin="1000ms" tts:color="white"/>
<set begin="2000ms" tts:color="black"/>
<set begin="3000ms" tts:color="white"/>
<set begin="4000ms" tts:color="black"/>
</p>
</div>
...
```

図 3　set 要素へ変換し出力される IMSC ファイル

```
...
<style xml:id="s1" tts:fontSize="30px"/>
...
<region xml:id="r1" style="s1" tts:origin="0px
0px"/>
...
<div xml:id="d001" begin="00:00:10"
end="00:00:20">
<p xml:id="p001" region="r1">
<span tts:ruby="container">
<span tts:ruby="base">漢字</span>
<span tts:ruby="text">ルビ</span>
</span>
</p>
</div>
...
```

図 4　#ruby 機能を持つ IMSC ファイル

```
...
<style xml:id="s1" tts:fontSize="30px"/>
<style xml:id="s1_ruby" tts:fontSize="16px"/>
...
<region xml:id="r1" style="s1" tts:origin="0px
9px"/>
<region xml:id="r1_0" style="s1_ruby" arib-
tt:letter-spacing="10px" tts:origin="10px 0px"/>
...
<div xml:id="d001" begin="00:00:01"
end="00:00:05">
<p xml:id="p001" region="r1">漢字</p>
<p xml:id="p001_0" region="r1_0">ルビ</p>
</div>
...
```

図 5　変換し出力される ARIB-TTML ファイル