W3C/慶應 デジタルサイネージとHTML5セミナー

Web-based Signageテストスペック

2015年3月26日

Webプラットフォーム性能ベンチマーク検討会

テストスペックの構成

- 「Web-based Signage性能要件」の各項目毎
- 評価試験実施のための仕様を記載

試験手順について

- 性能評価試験：自動化が望ましい
- 組込系端末が対象⇒目視など手動に頼らざるを得ないケースも想定(現時点)
- 今後、ブラウザ実装の進化により、手順は改訂

実装タイプと適合要件

1. 静止画・スライドショー型

小規模な小売店舗で店の情報などを表示する場合

企業オフィスで従業員向けの情報を伝える場合

- 動画は多用せず、静止画のアニメーションやテキストを主とするケース

実装タイプと適合要件

2. 動画型

静止画・スライドショー型に加え、ビデオ再生を多用したコンテンツにも対応するケース

イメージ映像や環境映像などを表示する場合

実装タイプと適合要件

3. 複合型

全評価項目で最高クラスに属し、様々な表現が可能

高品質なハイビジョン映像や高精細静止画などで構成されるコンテンツも表示するケース

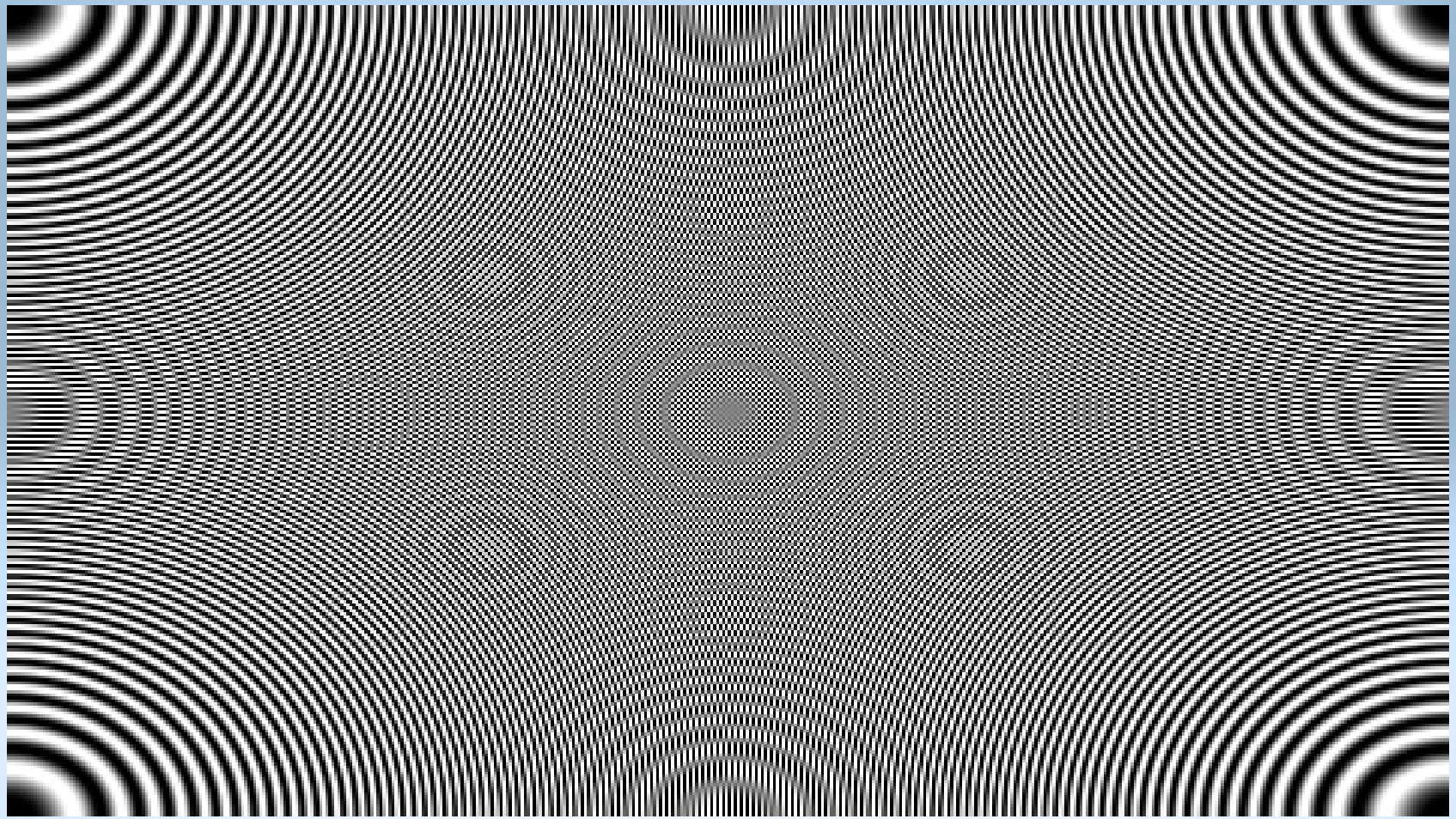
試験結果例

#	機能		申請値	試験判定	適用要件（複合型）
1	解像度	普及型	-	-	-
		高密度型	-	-	-
		超高密度型	**ピクセル	**Pass** **＊＊ピクセル**	**M**
2	ストレージ	キャッシュ型	-	-	-
		蓄積型	-	-	-
		大容量蓄積型	**MB	**Pass** **＊＊MB**	**M**
3	描画速度	普及型	-	-	-
		高速型	-	Pass **fps	-
		超高速型	**fps	**Fail**	**M**
4	並行処理	シングル型	-	-	-
		マルチ対応型	有	**Pass**	**M**
5	ロード容量	普及型	-	-	-
		ロング型	**枚	**Pass** **＊＊枚**	**M**
6	ビデオ再生	限定型		-	-
		普及型	-	-	-
		高機能型	有	**Pass**	**M**
7	オーディオ	普及型	-	-	-
		高機能型	有	**Pass**	**M**
8	再生開始遅延	普及型	-	-	-
		高性能型	300msec	Pass ＊＊msec	**M**

1. 解像度のテスト

- 小画面型(1280x720以下)、高密度型(1920x1080以下)、超高密度型(1920x1080超)
- ゾーンプレートなどのテストチャートを準備
- 実際に読み込み表示してビューポート解像度を確認する

ゾーンプレートの例



2. ストレージのテスト

- 128MBの画像ファイル8個(計1024MB)を用意
- Indexed Database APIまたはFile API: Directories and Systemを利用、端末に保存
- 保存機能なし(キャッシュ型)、8個とも保存可能(大容量蓄積型)、それ以外(蓄積型)

3. 描画速度のテスト(1)

- ティッカー・全画面移動のいずれかでフレームレートが20fps未満：普及型
- ティッカー・全画面移動の両方でフレームレートが20fps以上となり、かついずれかで40fps未満：高速型
- ティッカー・全画面移動の両方でフレームレートが40fps以上：超高速型

3. 描画速度のテスト(2)

a)ティッカー

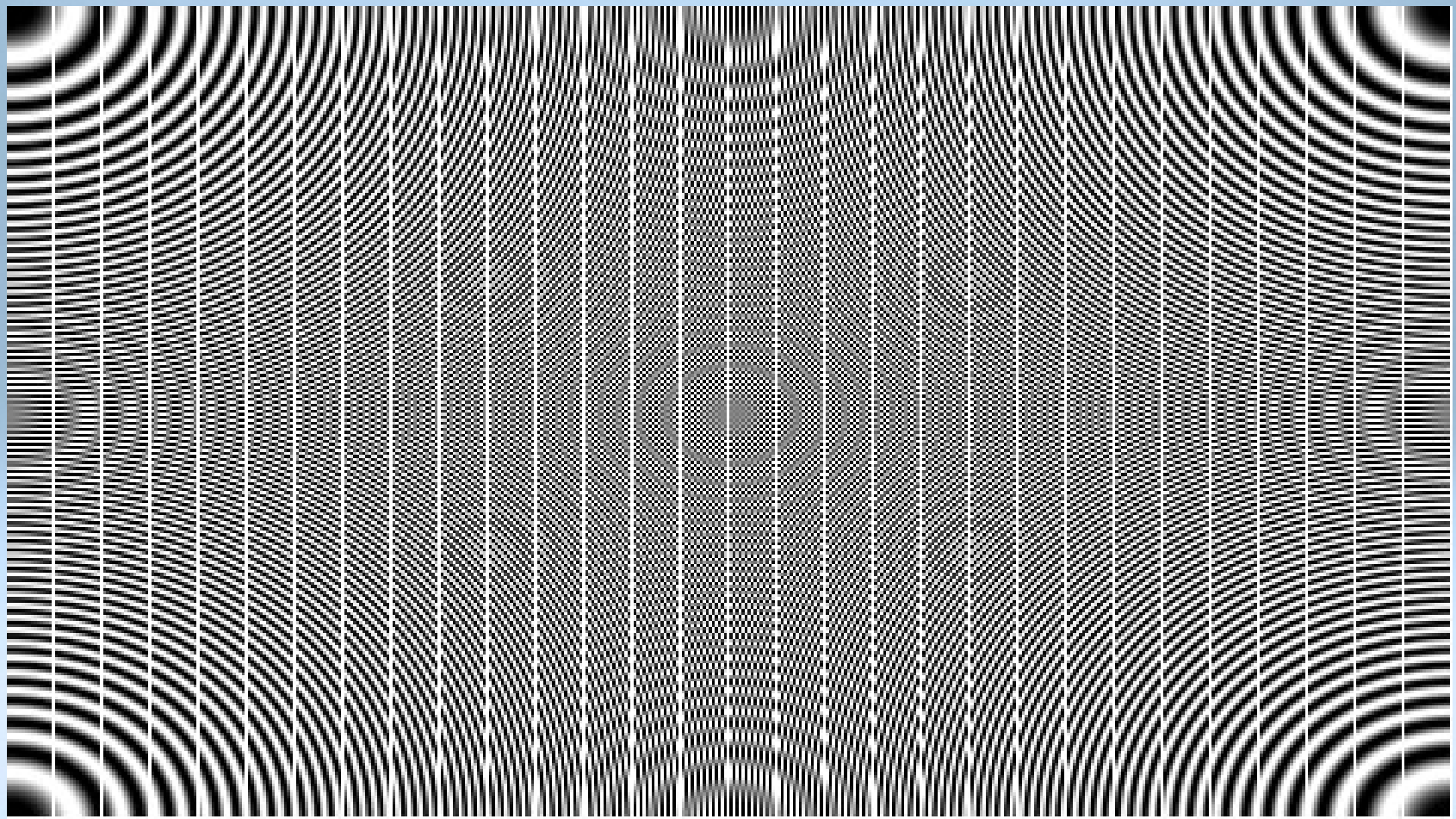
- ティッカー(高さ:画面高の10%)を準備(フォントで実現)
- スクロールさせる際の背景画像：細い等間隔の縦縞模様を有するもの
- CSS transform / transitionを利用しティッカーを水平移動。
  - 移動速度：1フレームあたり文字列が縞1本分進む速度
- 実際に端末上で画像表示させ、フレームレートを測定
  - 例：表示ディスプレイをビデオカメラで録画しその変化量から測定

3. 描画速度のテスト(3)

b) 全画面移動

- 等間隔の縦縞模様の重畳された静止画像(全画面)を準備
- CSS transform / transitionを利用し画像を水平移動。
  - 移動速度：1フレームあたり文字列が縞1本分進む速度
- 実際に端末上で画像表示させ、フレームレートを測定
  - 例：表示ディスプレイをビデオカメラで録画しその変化量から測定

縦縞画像の例



4. 並行処理のテスト

- ある計算処理について、Web Workersを使用するコードを準備
- ワーカースレッドを2個使用した場合と1個使用した場合の処理時間を比較
- 処理時間短縮無し（シングル型）、処理時間短縮（マルチ対応型）

5. ロード容量のテスト

- PNG24ビットの全画面サイズの画像30枚（全て異なる画像）を用意
- JSプレーヤーは、これらプレイリストにセットされた全画像用に、img要素を生成。この時点は非表示の状態(img要素のCSS displayプロパティがnone、またはimg要素のhiddenプロパティがtrue)
- 各img要素のsrc属性に、各画像ファイルのURLを設定
- 全img要素に画像がロードされたら、1枚ずつ画像を全画面表示(当該img要素のCSS displayプロパティにnone以外を設定、またはimg要素のhiddenプロパティにfalseを設定)し、挙動を確認。
- 30枚の画像で動作可能（大容量型）、それ以外（普及型）

6. ビデオ再生のテスト

- 前提:　mp4またはwebmのサポート、自動再生のサポート（→「限定型」以上）
- 全画面のビデオ映像(30fps, フレーム番号を重畳済みのもの)を準備。
  圧縮ビットレート —— 720P: 5Mbps,　1080P: 8Mbps, 4K: 30Mbps
- JSプレーヤーで全画面でビデオ再生し、コマ落ちの有無を確認（無→「普及型」以上）
- CSS Transformsをvideo要素に適用し、縮小表示・拡大表示・回転表示・移動表示を行いながらビデオ再生し、コマ落ちの有無を確認（無→「高機能型」）
  CSS Transformのパラメータ
  —— 縮小:0.5倍、拡大：2倍、回転：時計回りに90度、
  移動：右方向100画素・下方向100画素

7. オーディオ再生のテスト

- 前提: mp3, mp4, webm, oggのうちの少なくとも1種類のサポート、自動再生のサポート（→「普及型」以上）
- オーディオファイルを準備
- 下記の処理をWeb Audio APIを利用して実現、正常動作を確認（→高機能型）
  - 2種類の音源に異なる音量調整と遅延量調整を行って同時に再生
  - スクリプト上で1kHzサイン波音源を生成し再生
  - オーディオファイルをサーバーから取り込み再生

7. 再生遅延時間のテスト(1)

1　JavaScriptからvideo要素（HTMLVideoElementオブジェクト）を生成

2　生成したHTMLVideoElementオブジェクトのpreloadプロパティに"auto"をセット

3　生成したHTMLVideoElementオブジェクトにtimeupdateイベントのリスナーをセット

4　生成したHTMLVideoElementオブジェクトのsrcプロパティにビデオファイルのURLをセット

5　生成したvideo要素（HTMLVideoElementオブジェクト）をドキュメントに挿入

6　プリロードが完了するのに十分な時間(数秒)の経過後に、HTMLVideoElementオブジェクトのplay()メソッドを実行

7　play()メソッドを呼び出してから最初に生成したtimeupdateイベントが発生するまでの時間を測定し、ビデオの再生遅延時間とする。

7. 再生遅延時間のテスト(2)

- オーディオについても同様。
- play()(再生開始コマンド)実行後、実際に再生開始されるまでの遅延が、ビデオ・オーディオどちらかまたは両方とも0.5秒以上　→　普及型
- 同遅延がビデオ・オーディオ共に0.5秒未満　→　高性能型

今後の予定

- 試験詳細設計書の作成
- 試験用コンテンツの作成

Thank you !