Quality Prediction-Based Dynamic Content Adaptation Framework Applied to Collaborative Mobile Presentations

Habib Louafi, *Student, IEEE,* Stéphane Coulombe, *Senior, IEEE,* and Umesh Chandra

Abstract—Today, professional documents, created in applications such as PowerPoint and Word, can be shared using ubiquitous mobile terminals connected to the Internet. GoogleDocs and EasyMeet are good examples of such collaborative Web applications dedicated to professional documents. The static adaptation of professional documents has been studied extensively. Dynamic adaptation can be very useful and practical for interactive multimedia applications, because it allows the delivery of highly customized content to the end-user without the need to generate and store multiple transcoded versions. In this paper, we propose a dynamic framework that enables us to estimate transcoding parameters on the fly in order to generate near-optimal adapted content for each user. The framework is compared to current dynamic methods as well as to static adaptation solutions. We show that the proposed framework provides a better trade-off between quality and storage compared to other static and dynamic approaches. To quantify the quality of the adapted content, we introduce a measure of the quality of the experience based on its visual quality of the adapted content, as well as on the impact of its total delivery time. The framework has been tested on (but is not limited to) OpenOffice Impress presentations.

Index Terms—Dynamic content adaptation, image transcoding, interactive multimedia applications, mobile device, OpenOffice, presentations, professional documents, SSIM, XHTML.

1 Introduction

TODAY, Web content can be accessed by PCs, as well as by a wide variety of mobile devices (mobile phones, smartphones, etc.) under many brand names, and with varying features. With the frequent introduction of new devices to the market [1], their number and diversity is constantly growing. There is also a great diversity among the communication networks used by these devices. Depending on its location, the same device could use different networks (e.g., Wi-Fi, GPRS, or UMTS). Such diversity has changed the logic behind Web content authoring, with the result that new patterns of Web content presentation have been designed [2]–[5]. At the same time, Web content is becoming richer, and is using various formats (XHTML, JPEG, GIF, SVG, Flash, etc.) and styles.

In a collaborative environment such as hosting a conference meeting comprising PCs and mobile devices, the presentation slides should be shared and presented synchronously to all participants. The obvious solution is to send the PowerPoint presentation to the participants before the meeting. However, many mobile terminals will not support the PowerPoint format (or any other Office document format). Those that support the format still face the problem of downloading a potentially large document (often several MB). This operation is not only time-consuming, but drains the battery of units, and is often costly. Finally, the problem of synchronization with the host remains; participants lose track of which slide is being presented at any given time. This causes serious usability problems. Web technologies represent an attractive alternative because since they only require that the terminal be equipped with a browser (widely supported), they can customize the content to each participant and ensure constant synchronization with the presentation by sending the slide only when it is presented (slides are sent one by one).

In this paper, we focus on the customization or adaptation of slide documents for each participant. However, in a collaborative meeting context, it is very important to keep the original layout and all embedded images, i.e., the same view should be shared between all the meeting participants. A similar problem has been studied regarding the delivery of Web pages to be shared in a collaborative manner. This context is known as co-browsing or escorted browsing [6]. In this context, when it is not possible to keep the original layout, an extra view is added for PCs that reflects what mobile users are seeing. That way, PC users can refer to that extra view to collaborate with mobile users. In our approach, we propose to keep the original layout intentionally, since

- *H. Louafi and S.Coulombe are with the Department of Software and IT Engineering, École de technologie supérieure, Université du Québec, 1100 Notre Dame Street West, Montreal, H3C 1K3, Quebec, Canada. E-mail: habib.louafi.1@ens.etsmtl.ca; stephane.coulombe@etsmtl.ca*
- *U. Chandra is with the Nokia Research Center, 955 Page Mill Road, Palo Alto, CA 94304 USA. E-mail: umesh.1.chandra@nokia.com*

in such meeting contexts, the content can be accessed by mobile devices as well as by PCs. In fact, the proposed solutions which completely change the content (shrinking it, converting it to text or audio, etc.) were dedicated to limited mobile devices. Nowadays, more sophisticated devices, such the Apple iPhone, the Nokia Lumia, etc., have been introduced to the market. With these new devices, it is possible to preserve the original layout of the content and consequently deliver rich content that is visually identical to the original. The current tend is to deliver content that fits the target mobile device's resolution and allows the user to adapt its view by zooming and/or panning.

In recent years, we have witnessed the emergence of context-aware systems [7], which bridge the gap between a diversified and technologically limited clientele and richer Web content. Generally, a context-aware system involves several steps of content analysis and transcoding operations to tailor the content to meet the target mobile device's constraints, user preferences and role (e.g., attendee or host). Two major trends (static and dynamic) in content adaptation have been studied extensively, with a great deal of work done in the area of static content adaptation [8]–[10], where different versions of the original content are created and stored on the server. At runtime, when the content is requested, the best of those versions is selected to be delivered. In dynamic adaptation, a customized version is created on the fly, based on the contextual data gathered mainly at the user's request [5], [11], [12].

There are advantages and disadvantages to using either one of these two strategies. The static approach leads to high processing complexity in generating all the versions, and a great deal of storage space is required to save them. Therefore, to avoid huge response times when content is requested, the processing is often performed offline. In this case, the issue of granularity becomes important, and determines the compromise between the quality of the delivered content and its associated processing complexity and available storage space [13]. The more versions there are available, the better the quality. With the dynamic strategy, the content adaptation is typically performed on the fly, when the terminal's context is known, while the end-user is waiting. In this case, the server could easily be overwhelmed when the number of requests becomes significant [11], [14]. In such a situation, the user himself might lose interest in that content, owing to an unreasonable wait time.

In both content adaptation strategies, selecting the right format is crucial as well. Current solutions, such as the Nokia EasyMeet [15] and GoogleDocs Mobile [16], convert PowerPoint presentations into JPEG images that can be rendered by mobile Web browsers. However, raster formats such as these have major limitations in interactive applications, as they do not allow text editing or keyword searching. The XHTML format could be more suitable in these situations, since instead of converting the whole slide into an image, only embedded images are adapted and the text is resized. In fact, making the dynamic choice between JPEG and XHTML is a challenging task, with the best depending on the rasterized resolution and the amount of text relative to the number of images on a slide.

In this paper, we propose a dynamic framework that enables us to perform an on-the-fly estimation of near-optimal format and transcoding parameters prior to performing transcoding in order to reduce computational complexity, while improving the user experience. In this framework, we predict the visual quality of the adapted content and the amount of time it takes to reach the end-user (delivery time). The framework we propose has been applied to OpenOffice Impress presentations. It is designed to be quite general, but future work can be carried out to validate its applicability to other professional documents types, such as Word (text) and Excel (spreadsheets).

The paper is organized as follows. We begin by stating the transcoding problem in section 2. In section 3, we show how the visual quality and the quality of the experience are evaluated. The experimental setup comparing the static and dynamic approaches is presented in section 4. The experimental results are presented in section 5. Section 6 presents the computational complexity of the proposed dynamic framework. Finally, section 7 concludes the paper.

2 Problem Statement

Let C be a professional document, referred to here as the original document or content, composed of a set of pages c_k made up of various components $c_{k,i}$. We can write this formally as follows:
$C = c_1, c_2, ..., c_n$, where n is the total number of pages in C.
$c_k = c_{k,1}, c_{k,2}, \ldots, c_{k,m(k)}$, where $m(k)$ is the total number of components of the k^{th} page.

For instance, C could be a PowerPoint presentation and c_k the k^{th} slide composed of various components $c_{k,i}$. Theoretically, a component can be any object. For instance, in a slide c_k composed of a text box and a JPEG image, $c_{k,1}$ represents the text box and $c_{k,2}$ represents the JPEG image.

Given a page c_k, let $W(c_k)$ and $H(c_k)$ be its width and height, in pixels, respectively.

For a page c_k, let $h_{k,1}$, $h_{k,2}$,..., $h_{k,m}$ be sets of characteristics that can be adjusted to adapt that page's components $c_{k,1}$, $c_{k,2}$,..., $c_{k,m}$ respectively. For example, for a JPEG image (represented by $c_{k,2}$) embedded in a presentation, we may have the set $h_{k,2}$={resolution, quality factor}.

To be rendered by the target mobile device, the original document must be adapted. To achieve this, various adaptation operations can be used. Conceptually, different transcoding parameter combinations

can be used by the adaptation operations for each page. Let P be the possible transcoding parameters that can be used to adapt the original document's pages and their components, that is:

$$P = \{f, z, QF\}$$

where:

- $f \in \{\text{JPEG}, \text{XHTML}\}$ is the output format into which the original page is transcoded,
- $0 < z \leq 100\%$ is a scaling factor that defines the output resolution of the adapted page, and
- $0 < QF \leq 100$ represents the quality factor of the outputted JPEG images on the adapted page.

These parameters are applied as follows:

- If, for a given page c_k, the selected output format is JPEG, the whole page is rasterized into a JPEG image and wrapped in an XHTML skeleton. As a result, the whole page is converted into a Web page that contains only one JPEG image. In this case, the parameters z and QF are used to create that JPEG image.
- If the selected output format is XHTML, the whole page is transformed into an XHTML file, which may include both text and images. As a result, the output XHTML file will contain the same number of components (text and images) as the original document. In this case, to preserve the initial intentions of the author of the original document, the same z is used for all the components of the original pages and the same QF for all the embedded JPEG images. By preserving the author's intentions, the adapted page will have the same layout (relative sizes and positions of embedded components) as the original one.

Our formulation of the problem is inspired from the work presented in [17]–[19]. Let D be the target mobile device and $W(D)$, $H(D)$, $F(D)$, $S(D)$, $BR(D)$ and $NL(D)$ be its maximum permissible image width, image height, supported formats, file size (in bits), bitrate and latency of the network in use, respectively.

We define $\mathcal{T}(c_k, f, z, QF)$ as the operation that transforms the original page c_k using the transcoding parameters f, z and QF. Let $t_k^{f,z,QF}$ be the adapted content created by $\mathcal{T}(c_k, f, z, QF)$.
We say that the adapted content $t_k^{f,z,QF}$ is renderable by D if the following relations are true:

$$\begin{aligned} S\big(t_k^{f,z,QF}\big) &\leq S(D) \\ W(t_k^{f,z,QF}) = zW(c_k) &\leq W(D) \\ H(t_k^{f,z,QF}) = zH(c_k) &\leq H(D) \\ f &\in F(D) \end{aligned}$$

where $S\big(t_k^{f,z,QF}\big)$, $W\big(t_k^{f,z,QF}\big)$ and $H\big(t_k^{f,z,QF}\big)$ are the file size, the width and the height of the adapted content generated by $\mathcal{T}$ respectively.

The set of transcoding parameter combinations that can be used by $\mathcal{T}$ to transform original content c_k into adapted contents that are renderable by D is given by:

$$R(c_k, D) = \big\{(f, z, QF) | t_k^{f,z,QF} \text{ is renderable by } D\big\}$$

Since there could be multiple transcoding parameter combinations leading to versions renderable by D, we are interested in the combination that maximizes the quality of the user's experience. Let $f^*(c_k, D)$, $z^*(c_k, D)$ and $QF^*(c_k, D)$ be this combination, which is given by:

$$\begin{aligned} \big(f^*(c_k,D), z^*(c_k, D), QF^*(c_k, D)\big) = \\ \underset{(f,z,QF)\in R(c_k,D)}{\arg\max} \mathcal{Q}_{_E}(t_k^{f,z,QF}, D) \end{aligned} \quad (1)$$

where, $\mathcal{Q}_{_E}$ is a function that evaluates the quality of the user's experience. When (1) returns more than one combination, the one that presents the best visual quality is arbitrarily selected. The following section shows how the visual quality and the quality of the experience are evaluated.

3 Quality of Experience Evaluation

The quality of the delivered content, as experienced by the end-user, called the name *quality of experience* ($\mathcal{Q}_{_E}$), is affected by three factors [20]:

1) The quality of the content at the source, that is, the quality of the adapted content before delivery.
2) The quality of service QoS, which is affected by the delivery of the adapted content over the network.
3) The human perception regarding the adapted content.

In other words, $\mathcal{Q}_{_E}$ is affected by the visual quality and the transport quality (quality associated with the total delivery time). The first expresses how the content is appreciated visually, and the second expresses the impact of the total delivery time on the appreciation of the content. Based on these qualities, we propose to evaluate the $\mathcal{Q}_{_E}$ of the adapted content as follows:

For an adapted content $t_k^{f,z,QF}$ and a target mobile device D, we propose to evaluate the quality of experience $\mathcal{Q}_{_E}$ as follows:

$$\mathcal{Q}_{_E}(t_k^{f,z,QF}, D) = \mathcal{Q}_{_V}(t_k^{f,z,QF}, D)\mathcal{Q}_{_T}(t_k^{f,z,QF}, D) \quad (2)$$

where $0 \leq \mathcal{Q}_{_V} \leq 1$ and $0 \leq \mathcal{Q}_{_T} \leq 1$ represent the visual quality and the transport quality respectively. This is not the only way of evaluating the quality of experience, and as explained in [20], $\mathcal{Q}_{_E}$ as well as QoS evaluation represent a completely separate research topic. In our framework, we propose the product of $\mathcal{Q}_{_V}$ and $\mathcal{Q}_{_T}$ rather than the sum to prevent large disparities in $\mathcal{Q}_{_V}$ and $\mathcal{Q}_{_T}$ from being able to produce a high $\mathcal{Q}_{_E}$. In this context, the product is more appropriate than the sum, since $\mathcal{Q}_{_V}$ and $\mathcal{Q}_{_T}$ are not compensatory attributes. When two or more attributes

are combined to produce a single attribute to represent a given problem, the attributes are classified into compensatory and non-compensatory attributes. The former (compensatory) can be summed whereas the others cannot. This is the fruit of research performed particularly in the marketing and decision making fields [21], [22]. In the problem at hand, when a JPEG image is aggressively transcoded, its $\mathcal{Q}_T$ will be close to 1 (very lightweight image) and its $\mathcal{Q}_V$ close to 0 (very distorted image). If $\mathcal{Q}_V$ and $\mathcal{Q}_T$ are summed, the resulting $\mathcal{Q}_E$ will be close to 1, which is misleading. Contrary to the sum, the product will be close to 0, which seems more reasonable.

Note that the $\mathcal{Q}_E$ measure we propose is used to illustrate the benefits of our prediction-based dynamic content adaptation over existing methods when a trade-off between the visual quality of the adapted content and its delivery time is considered. However, the framework applies to other quality measures as well.

3.1 Visual Quality Evaluation

Let $c_k = \{c_{k,1}, c_{k,2}, \ldots, c_{k,m(k)}\}$ be an original page. Let $t_k^{f,z,QF} = \mathcal{T}(c_k, f, z, QF)$, be the adapted version of c_k, composed of the adapted components. We can write $t_k^{f,z,QF}$, for page c_k, as:

$$t_k^{f,z,QF} = \left\{ t_{k,1}^{f,z,QF}, t_{k,2}^{f,z,QF}, \ldots, t_{k,m(k,f)}^{f,z,QF} \right\}$$

where $t_{k,i}^{f,z,QF}$ is the i^{th} transcoded component and $m(k,f)$ is the total number of its components. It is given by (ignoring the XHTML wrapper, which has no impact on quality, for both cases):

$$m(k,f) = \begin{cases} m(k) & \text{if } f = \text{XHTML} \\ 1 & \text{if } f = \text{JPEG} \end{cases}$$

Using the SSIM [23] (or any other reliable visual quality index), it is possible to evaluate the visual quality of the adapted content. The later depends on the visual quality of its components, but also on the area occupied by each component (the larger the area, the larger the weight on quality it should have). Therefore, we propose to compute the visual quality as a weighted sum of its components' visual quality, the weights being the area occupied by each. Thus, we have:

$$\mathcal{Q}_V(t_k^{f,z,QF}, D) = \frac{\sum_{i=1}^{m(k,f)} A(t_{k,i}^{f,z,QF}) \mathcal{Q}_V(t_{k,i}^{f,z,QF}, D)}{\sum_{i=1}^{m(k,f)} A(t_{k,i}^{f,z,QF})} \quad (3)$$

$$\mathcal{Q}_V(t_{k,i}^{f,z,QF}, D) = \begin{cases} \mathcal{Q}_I(t_{k,i}^{f,z,QF}, D) & \text{if } t_{k,i}^{f,z,QF} \text{ is an image} \\ 1 & \text{if } t_{k,i}^{f,z,QF} \text{ is text} \end{cases} \quad (4)$$

where:

- $\mathcal{Q}_I$ is a metric that measures the image quality.
- We assumed that the text is rendered perfectly, and without loss of generality, the visual quality of text boxes is set to 1. However, more sophisticated metrics could be used to take into account the text size, the color and even the font.
- $A(t_{k,i}^{f,z,QF})$ is the visible (not hidden) area occupied by $t_{k,i}^{f,z,QF}$. We always have $A(t_{k,i}^{f,z,QF}) \leq H(t_{k,i}^{f,z,QF}) W(t_{k,i}^{f,z,QF})$ since two components are allowed to partially overlap one another. For instance, a text region can completely or partially overlap an image region. However, if two images overlap, the hidden regions of an image should not be considered neither for computing its region A nor its quality $\mathcal{Q}_I$. This is particularly important when the page contains a background.
- When the adapted content $t_k^{f,z,QF}$ comprises one image (e.g., JPEG), its visual quality is reduced to the visual quality of that image, as is the case when the output format to be used is $f = \text{JPEG}$.

3.2 Visual Quality Estimation

If adapted content were available, it would be straightforward, using (3), to compute its visual quality. The challenge, with the dynamic content adaptation system we propose, is to be able to estimate the visual quality of components subject to transcoding parameters without having to perform any transcoding operation. This estimation process is the key to the proposed system's reduced computational complexity.

The adapted content's areas can be known at runtime (when the content is requested). That is, if $f = \text{XHTML}$, these areas can be computed by scaling the areas of the original content's components using the scaling parameter z and when $f = \text{JPEG}$, the area of the whole of the original content is scaled using z. From (4), the visual quality of the adapted components is set to 1 for text, and for images it is defined as the adapted image's quality using a given quality metric (e.g., SSIM). It is hoped that the image quality can be predicted using a solution proposed in [17], in which it is shown to be possible to estimate the SSIM of JPEG images (characterized by QF_{in}, their actual QF) subject to changing their scaling parameter (z) and quality factor (QF_{out}) and for a viewing conditions (z_v). For original content $c_{k,i}$ the value of z_v controls the resolution, $z_v W(c_{k,i}) \times z_v H(c_{k,i})$, at which the original and the transcoded images should be scaled for comparison, in order to compute their SSIM. For instance, when $z_v = 1$, the two images are compared at the resolution of the original one; when $z_v = z$, the two images are compared at the resolution of the transcoded one; when $z_v = \min(W(D)/W(c_{k,i}), H(D)/H(c_{k,i}), 1)$, the two images are compared at the maximum resolution supported by the terminal or the original size of the

TABLE 1
Sub-array of predicted SSIM values, $\widehat{SSIM}(z_v, QF_{in}, z, QF_{out})$, computed for $QF_{in} = 80$ and $z_v = 40\%$ (from [17].

	Scaling, z,%									
QF_{out}	10	20	30	40	50	60	70	80	90	100
10	0.25	0.43	0.55	0.62	0.69	0.73	0.76	0.79	0.80	0.82
20	0.30	0.52	0.65	0.73	0.79	0.82	0.85	0.87	0.88	0.89
30	0.33	0.56	0.69	0.77	0.83	0.86	0.89	0.90	0.91	0.92
40	0.35	0.58	0.72	0.80	0.85	0.88	0.90	0.92	0.92	0.94
50	0.36	0.61	0.74	0.82	0.87	0.90	0.92	0.93	0.94	0.95
60	0.38	0.63	0.76	0.84	0.89	0.92	0.93	0.94	0.95	0.96
70	0.39	0.65	0.78	0.86	0.90	0.93	0.94	0.95	0.95	0.97
80	0.42	0.68	0.81	0.89	0.93	0.95	0.96	0.96	0.97	1.00
90	0.45	0.72	0.85	0.92	0.95	0.96	0.97	0.97	0.98	0.99
100	0.49	0.78	0.91	0.97	0.98	0.98	0.99	0.99	0.99	1.00

image, whichever is the smaller. In practice, this value can be set by the maximum resolution of the target terminal.

In other words, when a JPEG image is transcoded using a scaling parameter z and quality factor QF, the SSIM of the transcoded image can be estimated using the predicted data that are tabulated in [17], and which are indexed by QF_{in}, z_v, z and QF. For instance, Table 1, which is extracted from [17], shows a sub-array of predicted SSIM values of transcoded JPEG images characterized by their actual $QF_{in} = 80$, transcoded using z and QF and evaluated under viewing conditions of $z_v = 40\%$. As commercial products generally use a QF value between 75 and 85 to encode documents (or re-encode images) into JPEG images to preserve their visual quality, in Table 1, we presented a sub-array for $QF_{in} = 80$. OpenOffice, for instance, proposes a default value of $QF = 75$.

According to this table, we predict that SSIM=0.90 when an image encoded with $QF_{in} = 80$ is transcoded using $z = 50\%$, $QF_{out} = 70$, and viewed at $z_v = 40\%$.

Note that these predicted SSIM values were computed by training and clustering, in which only the QF (QF_{in}) of the original image, and the transcoding parameters z, QF, and z_v were considered. A more sophisticated clustering method, taking into consideration two additional features (the original image's number of bits per pixel and $QF_{out} - QF_{in}$), was proposed in [24] to improve the prediction accuracy. Tables such as those in [17] can be used to estimate the visual quality of the adapted content as follows:

- When the format to be used is $f =$ XHTML, the adapted content will comprise the same number of components as the original one. In this case, using the SSIM index, the visual quality of the adapted content's components (4) becomes:

$$\mathcal{Q}_V(t_{k,i}^{f,z,QF}, D) = \begin{cases} SSIM(t_{k,i}^{XHTML,z,QF}, c_{k,i}, z_v) & \text{if } c_{k,i} \text{ is an image} \\ 1 & \text{if } c_{k,i} \text{ is text} \end{cases}$$

where $t_{k,i}^{XHTML,z,QF}$ is the transcoded version of $c_{k,i}$.
The SSIM of the embedded images of the adapted content can be estimated using the predicted SSIM values [17], and thus, the estimated visual quality of the adapted components becomes:

$$\hat{\mathcal{Q}}_V(t_{k,i}^{f,z,QF}, D) = \begin{cases} \widehat{SSIM}(z_v, QF_{in}(c_{k,i}), z, QF) & \text{if } c_{k,i} \text{ is an image} \\ 1 & \text{if } c_{k,i} \text{ is text} \end{cases}$$

where $QF_{in}(c_{k,i})$ represents the quality factor of $c_{k,i}$ and $\widehat{SSIM}(z_v, QF_{in}(c_{k,i}), z, QF)$ is the estimated SSIM value that can be extracted from the predicted SSIM arrays [17] using z_v, $QF_{in}(c_{k,i})$, z and QF. Thus, the visual quality of the whole of the adapted content can be estimated using (3). This process is illustrated in Fig. 1.

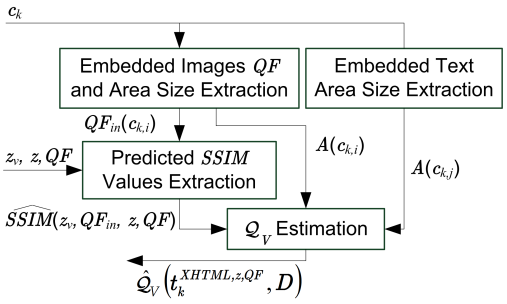


Fig. 1. XHTML $\mathcal{Q}_V$ estimation

- When the format to be used is $f =$ JPEG, the adapted content will comprise only one JPEG image. Let $t_{k,1}^{JPEG,z,QF}$ be this image transcoded at z and QF, and $t_{k,1}^{JPEG,100\%,80}$ be the image created using $z = 100\%$ and $QF = 80$. The latter is used as a reference image. Note that in estimating the visual quality, the image $t_{k,1}^{JPEG,100\%,80}$ is not actually created. It is mentioned here only to illustrate the visual quality computing process. However, from [18], the reference image ($t_{k,1}^{JPEG,100\%,80}$) is needed to estimate the file size of the adapted content, as described in section 3.4. Now, using the SSIM index, the visual quality of the adapted content (3) becomes:

$$\begin{aligned} \mathcal{Q}_V(t_k^{f,z,QF}, D) &= \mathcal{Q}_V(t_{k,1}^{JPEG,z,QF}, D) \\ &= SSIM(t_{k,1}^{JPEG,z,QF}, t_{k,1}^{JPEG,100\%,80}, z_v) \end{aligned}$$

This visual quality can be estimated using the predicted SSIM values computed for various z_v and QF_{in} as illustrated in Fig. 2. For example, Table 1 represents such values for $z_v = 40\%$ and $QF_{in} = 80$. The estimated visual quality is:

$$\hat{\mathcal{Q}}_V(t_k^{f,z,QF}, D) = \widehat{SSIM}(z_v, 80, z, QF)$$

where $\widehat{SSIM}(z_v, 80, z, QF)$ is the estimated SSIM value that can be extracted from the predicted SSIM arrays [17] using z_v, $QF_{in}(t_{k,1}^{JPEG,100\%,80}) = 80$, z and QF. For example, using Table 1, we obtain: $\widehat{SSIM}(40\%, 80, 50\%, 80) = 0.93$.

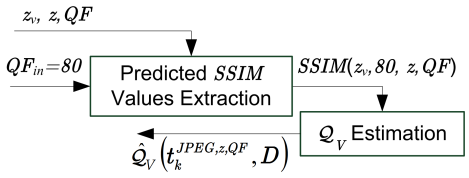


Fig. 2. JPEG $\mathcal{Q}_V$ estimation

3.3 Transport Quality Evaluation

No doubt, the longer it takes to deliver the adapted content, the less it is appreciated by the end-user. As the total delivery time increases, its associated quality is reduced accordingly. Therefore, transport quality is inversely proportional to the total delivery time of the adapted content. For an adapted content $t_k^{f,z,QF}$ and a target mobile device D, the total delivery time is given by:

$$tdt\left(t_k^{f,z,QF}\right)= \frac{S(t_k^{f,z,QF})}{BR(D)}+NL(D)+SL(D)+TL(c_k,f,z,QF) \quad (5)$$

where

- $S(t_k^{f,z,QF})$ is the file size in bits of $t_k^{f,z,QF}$.
- D is the target mobile device and $BR(D)$ and $NL(D)$ its bitrate and the latency of the network to which the mobile device is connected respectively.
- $SL(D)$ is the server latency. For a device D, it represents the time spent by the request in the server (e.g., in the queue) waiting to be processed. This value evaluates the performance of the server.
- $TL(c_k, f, z, QF)$ is the transcoding latency. It represents how long the adaptation operation takes to complete. It depends on the original content c_k and the transcoding parameters f, z and QF in use. It can be estimated based on past transcoding operations. On high-end computers, this value should be small.

We propose to evaluate the transport quality using a normalized *Z-shaped built-in* membership function [25] as illustrated in Fig. 3. This was inspired by the work of [3], [4], in which the authors used *sigmf* and *gaussmf* membership functions to map various parameters, such as the network's bandwidth and latency, to quality values between 0 and 1. By varying the values of α and β, it is possible to create a family of curves that have the same behavior. According to different research works performed to estimate the waiting time that users can tolerate when accessing Web content [26], the values of α and β can be set to model the user's behavior regarding the waiting time. These values can be determined by experience or defined by the end-user. The value of α expresses the period of time in which the end-user is fully satisfied with the response time. The value of $(\alpha+\beta)/2$ expresses the period of time, in which that appreciation is reduced to 50%, and when the total delivery time reaches the value β, the user's appreciation is nil. Thus, the transport quality can be formulated as follows:

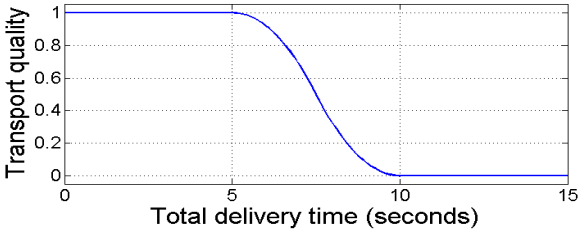


Fig. 3. Transport quality behavior for $\alpha = 5$ and $\beta = 10$

$$\mathcal{Q}_T(t_k^{f,z,QF}, D) = \mathrm{Zmf}\left(tdt(t_k^{f,z,QF}, D), \alpha, \beta\right) \quad (6)$$

with:

$$\mathrm{Zmf}(x,\alpha,\beta)=\begin{cases}1 & \text{if } x \le \alpha \\ 1-2\left(\frac{x-\alpha}{\beta-\alpha}\right)^2 & \text{if } \alpha \le x \le \frac{\alpha+\beta}{2} \\ 2\left(\frac{x-\beta}{\beta-\alpha}\right)^2 & \text{if } \frac{\alpha+\beta}{2} \le x \le \beta \\ 0 & \text{if } x \ge \beta \end{cases}$$

3.4 Transport Quality Estimation

In (5) and (6), all the parameters used in computing the total delivery time and its associated transport quality must, if unknown, be estimated at run-time. Various algorithms have been proposed to estimate the network bitrate at run-time [27]. The network latency is generally estimated by "pinging" the target mobile device at run-time or taking a mean value of previous probings that could have been performed when the user was registered [28].

We propose to estimate the adapted content file size as well, using the method proposed in [18]. That is, when a JPEG image $c_{k,i}$ (characterized by its QF, denoted QF_{in}) is transcoded into another JPEG image ($t_{k,i}^{JPEG,z,QF}$) using a scaling parameter z and quality factor QF (QF_{out}), the relative file size between them, denoted $r(t_{k,i}^{JPEG,z,QF}, c_{k,i}) = S(t_{k,i}^{JPEG,z,QF})/S(c_{k,i})$, can be predicted by the method presented in [18]. As explained in section 3.2, these predictors were computed by training and clustering, where QF_{in}, z, QF were considered as features (in [24], they proposed two new features to increase the prediction accuracy). Table 2 shows predicted relative file sizes for $QF_{in} = 80$ and various values of z and QF (or QF_{out}). The predicted relative file size can be used to compute the total delivery time (tdt) and its associated quality ($\mathcal{Q}_T$) as follows:

- When the format to be used is $f =$ XHTML, the file size of the adapted content can be computed by summing the file sizes of its embedded images and text boxes and adding an additional size related to the XHTML wrapper as follows:

$$S\left(t_k^{f,z,QF}\right) = \sum_{i=1}^{m(k)} r(t_{k,i}^{XHTML,z,QF}, c_{k,i}) S(c_{k,i}) + \psi$$

TABLE 2
Sub-array of predicted relative file sizes $\hat{r}_I(QF_{in}, z, QF_{out})$ computed for $QF_{in} = 80$ (from [18]).

	Scaling, z,%									
QF_{out}	10	20	30	40	50	60	70	80	90	100
10	0.03	0.04	0.05	0.07	0.08	0.10	0.12	0.15	0.17	0.20
20	0.03	0.05	0.07	0.09	0.12	0.15	0.19	0.22	0.26	0.32
30	0.04	0.05	0.08	0.11	0.15	0.19	0.24	0.21	0.34	0.41
40	0.04	0.06	0.09	0.13	0.17	0.22	0.28	0.34	0.40	0.50
50	0.04	0.06	0.10	0.14	0.19	0.25	0.32	0.39	0.46	0.54
60	0.04	0.07	0.11	0.16	0.22	0.28	0.36	0.44	0.53	0.71
70	0.04	0.08	0.13	0.18	0.25	0.33	0.42	0.52	0.63	0.85
80	0.05	0.09	0.15	0.22	0.31	0.41	0.52	0.65	0.78	0.95
90	0.06	0.12	0.21	0.31	0.44	0.59	0.75	0.93	1.12	1.12
100	0.10	0.24	0.47	0.75	1.05	1.46	1.89	2.34	2.86	2.22

where $S(c_{k,i})$ represents the file size of the component $c_{k,i}$ and $r(t_{k,i}^{XHTML,z,QF}, c_{k,i})$ the relative file size between $c_{k,i}$ and its transcoded version $t_{k,i}^{XHTML,z,QF}$.

Using the predicted relative file sizes [18] (e.g., Table 2), the adapted content's file size can be estimated as illustrated in Fig. 4. For instance, Table 2 shows that an image transcoded using $QF_{out} = 80$ and $z = 80\%$ will occupy 65% of its original file size. Formally, the estimated file size is given by:

$$\hat{S}(t_k^{f,z,QF}) = \sum_{i=1}^{m(k)} \hat{r}(t_{k,i}^{XHTML,z,QF}, c_{k,i}) S(c_{k,i}) + \psi$$

$$\hat{r}(t_{k,i}^{XHTML,z,QF}, c_{k,i}) = \begin{cases} \hat{r}_I(QF_{in}(c_{k,i}), z, QF) & \text{if } c_{k,i} \text{ is an image} \\ 1 & \text{if } c_{k,i} \text{ is text} \end{cases}$$

where:

- $QF_{in}(c_{k,i})$ is the QF of original image $c_{k,i}$.
- $\hat{r}_I(QF_{in}(c_{k,i}), z, QF)$ is the estimated relative file size between the image $c_{k,i}$ and its transcoded version $t_{k,i}^{XHTML,z,QF}$, which can be extracted from the predicted relative file sizes arrays tabulated in [18] (Table 2 shows such an array for $QF_{in} = 80$ and various values of z and $QF = QF_{out}$).
- ψ represents the added size of the XHTML wrapper. Typically, the file size of the XHTML wrapper for one slide is equal to 1 KB. Therefore, we set $\psi = 1$KB.

It is interesting to note that although the file size prediction model does not use explicit statistics related to the compressed form of the input image (such as the number of zeroed DCT coefficients), it implicitly takes into account the compressibility of the original image through its file size, $S(c_{k,i})$.

- When the format to be used is f = JPEG, using the reference JPEG image created before ($t_{k,1}^{JPEG,100\%,80}$), the file size of the adapted content becomes:

$$S(t_k^{f,z,QF}) = r(t_{k,1}^{JPEG,z,QF}, t_{k,1}^{JPEG,100\%,80}) S(t_{k,1}^{JPEG,100\%,80}) + \psi$$

The file size of the adapted content can be estimated using the predicted relative file sizes [18] as illustrated in Fig. 5. Formally, the estimated file size of the adapted content is given by:

$$\hat{S}(t_k^{f,z,QF}) = \hat{r}(t_{k,1}^{JPEG,z,QF}, t_{k,1}^{JPEG,100\%,80}) S(t_{k,1}^{JPEG,100\%,80}) + \psi$$

$$\hat{r}(t_{k,1}^{JPEG,z,QF}, t_{k,1}^{JPEG,100\%,80}) = \hat{r}_I(QF_{in}(t_{k,1}^{JPEG,100\%,80}), z, QF) = \hat{r}_I(80, z, QF)$$

where:

- $QF_{in}(t_{k,1}^{JPEG,100\%,80}) = 80$ is the quality factor of $t_{k,1}^{JPEG,100\%,80}$
- $\hat{r}_I(80, z, QF)$ is the estimated relative file size between the two images $t_{k,1}^{JPEG,z,QF}$ and $t_{k,1}^{JPEG,100\%,80}$. It can be extracted from Table 2.
- ψ, as before, is the XHTML wrapper size.

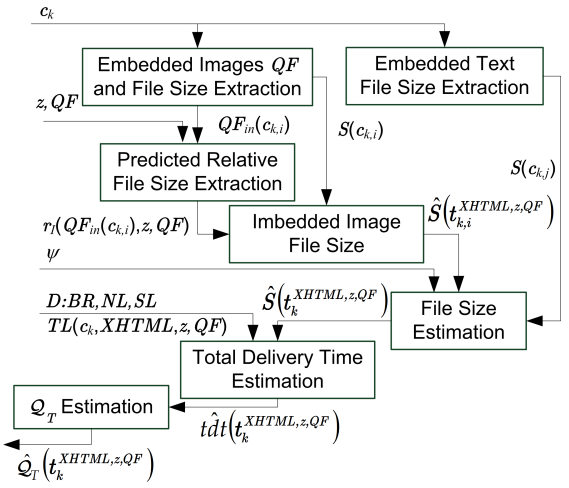


Fig. 4. XHTML $\mathcal{Q}_T$ estimation

Finally, using the proposed dynamic framework, the visual quality, the transport quality and the quality of experience can be estimated on the fly. Using (2), the estimated quality of experience becomes:

$$\hat{\mathcal{Q}}_E(t_k^{f,z,QF}, D) = \hat{\mathcal{Q}}_V(t_k^{f,z,QF}, D) \hat{\mathcal{Q}}_T(t_k^{f,z,QF}, D)$$

where $\hat{\mathcal{Q}}_V(t_k^{f,z,QF}, D)$ and $\hat{\mathcal{Q}}_T(t_k^{f,z,QF}, D)$ are two functions that estimate the visual and transport qualities using the prediction arrays of SSIM and relative file sizes [17], [18] as illustrated in Figs. 1, 2, 4 and 5.

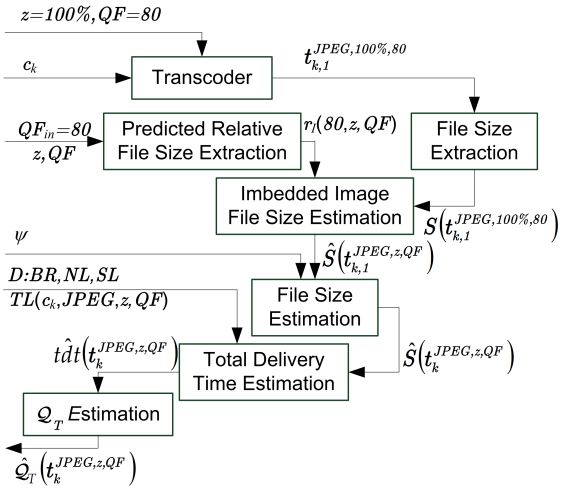


Fig. 5. JPEG $\mathcal{Q}_T$ estimation

3.5 Use of the Estimation Method

To estimate the optimal combination of format and transcoding parameters that should be used to adapt an original content c_k, we compute, using all combinations of f, z and QF, two arrays $\hat{\mathcal{Q}}_V(t_k^{f,z,QF}, D)$ and $\hat{r}(t_k^{f,z,QF})$. Based on these arrays, we compute the estimated quality of experience array $\hat{\mathcal{Q}}_E(t_k^{f,z,QF}, D)$ and solve (1) to determine the best feasible solution (i.e., best transcoding parameters combination). We expect the solution to be near-optimal.

4 Experimental Setup

4.1 Slides Corpus

To test and validate the proposed framework, a large corpus of documents (presentation slides) was required. Such slides could have been searched and collected from the Web. However, to test and analyze the proposed system on a wide range of slide types, we preferred creating slides composed of components with various sizes and positions in the slide. The images, as well the text, however, were collected from the Web in order to be representative of existing content. We developed a Java-based application that uses OpenOffice APIs (UNO) to create a set of Impress slides [29]. Note that a slide is allowed to contain text and images sharing the same area (overlap). The background was set to none (no master style), but an inserted image could represent 100% of the slide, and therefore be considered as a background. The created slides' file sizes varied between 12 KB and 122 KB. The images were collected from Internet Web sites such as [30]. The positions of the text boxes and images on the slides were set randomly by a random number generator that is part of the same application. Since the dimensions of the images and text boxes could be continuous, and to avoid context dilution, quantized values, representing the percentage of areas occupied by images and text boxes, were used as follows:

$$I \in \{0\%, 10\%, 20\%, \ldots, 100\%\}$$
$$T \in \{0\%, 10\%, 20\%, \ldots, 100\%\}$$

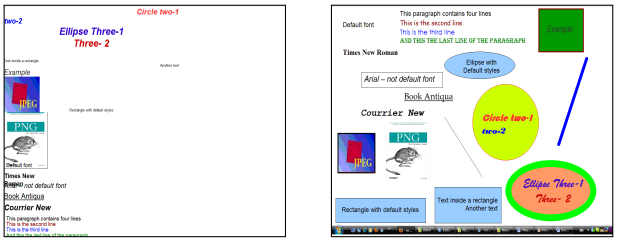
For instance, $I = 40\%$ and $T = 30\%$ mean that the area occupied by the images represents 40% of the slide and that occupied by the text boxes 30%. An example of a slide composed of $I = 40\%$ and $T = 25\%$ is shown in Fig. 7(a).

To facilitate the validation, each document was composed of one slide. This restriction, which can be removed later, does not affect the credibility of the validation, since each slide can be seen as separate content, and so is converted and sent separately. Let $\mathcal{V}$ be this validation set.

4.2 Transcoding Methodology

To compare the quality of the transcoded content, each slide from $\mathcal{V}$ was transcoded using OpenOffice JPEG and XHTML filters, which produce JPEG- and XHTML-based Web pages, respectively. The first filter converts the whole slide into an image and wraps it in a skeleton Web page. In the proposed dynamic framework, to be able to estimate $\hat{\mathcal{Q}}_V$ and $\hat{r}$ of any adapted content when the format used is JPEG, the created JPEG image $t_{k,1}^{JPEG,100\%,80}$ was used as a reference image from which the other images ($t_{k,i}^{JPEG,z,QF}$) were created using ImageMagick command line tools [31]. These images replaced those created by the JPEG-based filter. We could thus use the predicted SSIM and relative file sizes computed by [17], [18] to estimate $\hat{\mathcal{Q}}_V$ and $\hat{r}$ of the adapted contents.

Note that, the native OpenOffice XHTML filter capability was very limited, however, and it was found to have numerous bugs. For instance, the layout was not preserved in the output XHTML Web page, and all the components (text boxes and images) were aligned on the left-hand side. Only one font (the default OS font) was used for the entire presentation. Moreover, it was not possible to embed images in the traditional fashion, which consists in including the image URL in the XHTML text and saving the image in a specific folder. The implemented solution involved converting the embedded images into base-64 encoding, and then include them directly on the Web page (which resulted in larger images). Also, no graphic possibilities were allowed, meaning that all the embedded graphics were simply ignored in the XHTML version. Furthermore, it was not possible to change the embedded images' characteristics, such as the resolution and the quality factor. Ultimately, we improved the filter by fixing these important bugs and limitations, and adding the possibility of manipulating images and their characteristics (with proper URL support), as well as using graphics. After this extension, the modified OpenOffice XHTML filter was able to convert the slide into a standard XHTML



(a) (b)

Fig. 6. An example that summarizes the extensions performed on the native OpenOffice XHTML filter: (a) a slide as exported by the native XHTML filter, (b) as exported by our extended XHTML filter

file, which could include both text and images. Fig. 6 summarizes the majority of the fixed bugs; Fig. 6(a) shows the Web page version of a slide as outputted by the native OpenOffice XHTML filter, while Fig. 6(b) shows the Web page version created by our modified filter (which visually corresponds exactly to the original slide). An example of a slide as exported by the extended OpenOffice XHTML filter is shown in Fig. 7. Fig. 7(a) shows the original slide as rendered by OpenOffice, while, Figs. 7(b), 7(c) and 7(d) show its exported versions with the extended XHTML filter using $z = 30\%$ and $QF = 80$, $z = 80\%$ and $QF = 80$, and $z = 50\%$ and $QF = 60$, respectively.

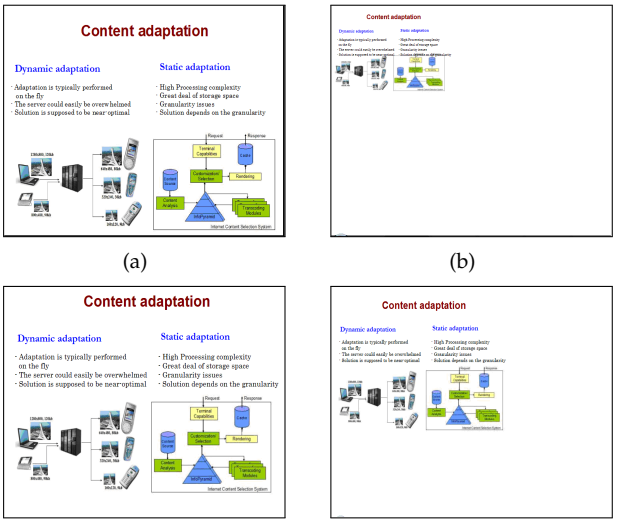
The following are the sets of transcoding parameters used by these filters, as explained in section 2:

$$f \in \{\text{JPEG}, \text{XHTML}\}$$
$$z \in \{10\%, 20\%, 30\%, \ldots, 100\%\}$$
$$QF \in \{10, 20, 30, \ldots, 100\}$$

Let $\mathcal{W}$ be the set of adapted contents created from the original contents of $\mathcal{V}$ using the transcoding parameters f, z and QF.

4.3 Validation Methodology

The OpenOffice (or MS-Office suite) XHTML filters, which produce JPEG-based XHTML pages, offer the possibility of selecting the target resolution and JPEG quality factor. Thought few parameters are offered via their graphical interfaces, high, low and medium quality, more precise parameters can be used programmatically via their APIs, which is what commercial dynamic solutions use [15]. These techniques do not use any quality of experience measure. They typically adjust the JPEG resolution to the maximum resolution supported by the device and use a fixed JPEG quality factor (e.g., 80) to provide good visual quality regardless of the transfer time incurred. This dynamic system will be denoted below as a *fixed-QF system*. On the other hand, with static solutions, since different versions of the content are created, the quality of experience of each can be taken into account.



(a) (b) (c) (d)

Fig. 7. A slide as exported by our extended OpenOffice XHTML filter: (a) the original slide, (b) transcoded using $z = 30\%$ and $QF = 80$, (c) transcoded using $z = 80\%$ and $QF = 80$, (d) transcoded using $z = 50\%$, $QF = 60$

Therefore, we propose to compare our method with a typical dynamic system (fixed-QF system) and various static systems using different granularity levels. Most static transcoding systems create different versions to suit various target mobile devices [3], [4], [8]. When the content is requested by the mobile device, the best adapted version among those created in advance is selected for delivery. The granularity of the created versions should be adequate enough to deliver the best user experience possible. However, in practice, it is not always possible to reach that level of granularity, owing to numerous constraints, such as storage space or CPU processing time limitations, for example. Thus, we propose to compare our solution with the following hypothetical static transcoding systems, which are inspired from realistic needs:

- Exhaustive system: This system creates the maximum possible number of adapted content versions. We can say it uses all combinations of these quantized values: $z = \{10\%, 20\%, \ldots, 100\%\}$ and $QF = \{10, 20, \ldots, 100\}$ for both the JPEG and XHTML formats. As a result, it creates 200 versions for each slide. We assume that this system provides a high enough granularity content, and therefore constitutes a good benchmark for comparing the best adapted content provided by each system.
- Granularity-based systems: In practice, it is not always possible nor desirable to transcode content into 200 versions, and so we may consider using only a limited number of values for z and QF. For instance, for an Impress presentation composed of 30 slides, 6000 versions should be created. For a server dedicated to organizing

TABLE 3
Static transcoding systems used in the validation.

N	Systems	Scaling parameter, z, %									
1	Granularity-based systems (QF = 80)	100									
2		100	50								
3		100	50	70							
4		100	50	70	30						
5		100	50	70	30	80					
6		100	50	70	30	80	60				
7		100	50	70	30	80	60	40			
8		100	50	70	30	80	60	40	90		
9		100	50	70	30	80	60	40	90	20	
10		100	50	70	30	80	60	40	90	20	10
11	Exhaustive system	$z \in \{10\%, 20\%, \ldots, 100\%\}$ and $QF \in \{10, 20, \ldots, 100\}$									

meetings for example, the number of versions that should be created could be very high, and so would be the processing time and storage space needed for that purpose. Since the most widely used quality factor is 80, we propose to compare our solution with ten systems based on that quality factor and different quantized values of z (see Table 3). From the first system to the tenth, the granularity is enriched gradually, always building from the previous system (i.e., system $i+1$ adds one more version on top of system i, whose characteristics are selected to cover the parameters set). For instance, the first system creates only one version (using $z = 100\%$) for each format, while the next system creates two versions (using $z = 100\%$ and 50%) and so on, as shown in Table 3. This schema is not the only one possible, and, depending on the available resources, other sets of systems could be used, such as varying QF around 80 (e.g., 70 or 60).

Note that the proposed static systems are actually more sophisticated than those that are commonly used. Typical static systems select the highest resolution content supported by a device regardless of the delivery time. But the proposed static systems, using the same $\mathcal{Q}_E$ metric, will lead to a fairer comparison between the full potential of static and dynamic approaches.

The comparison focuses on two aspects. First, we compare the performance of the proposed dynamic system, in terms of quality of experience, to the dynamic fixed-QF system and a static system with N versions (i.e., how many versions must a static system generate to match our system). We also compare this quality with that of the exhaustive static system to measure how far we are from optimality. Secondly, we compare the storage space required by each system.

Each adapted content $t_k^{f,z,QF}$ in $\mathcal{W}$, which is in fact a Web page, is parsed, and its actual $\mathcal{Q}_V(t_{c_k}^{f,z,QF}, D)$ and $r(t_{c_k}^{f,z,QF})$ values are computed. We then compute, for the desired mobile device D, $\mathcal{Q}_T(t_{c_k}^{f,z,QF}, D)$ and $\mathcal{Q}_E(t_{c_k}^{f,z,QF}, D)$. On the other hand, for each slide c_k in $\mathcal{V}$, its $\hat{\mathcal{Q}}_V(t_{c_k}^{f,z,QF}, D)$ and $\hat{r}(t_{c_k}^{f,z,QF})$ are computed using the proposed dynamic framework. We then compute, for the same mobile device, $\hat{\mathcal{Q}}_T(t_{c_k}^{f,z,QF}, D)$ and $\hat{\mathcal{Q}}_E(t_{c_k}^{f,z,QF}, D)$. The SSIM index exhibits a highly non-linear relationship with the DMOS (Differential Mean Opinion Score), and therefore cannot be used directly as a measure of the human perception of quality. Consequently, to address the third requirement regarding the $\mathcal{Q}_E$ design [20] (see section 3), the SSIM values are mapped, using a logistic function and regression, to their corresponding subjective MOS (Mean Opinion Score) values [32]. In other words, we compute or estimate the SSIM, but then map it to the corresponding MOS value. As a result two arrays were created:

$$QE: \left[c_k, f, z, QF, \mathcal{Q}_T(t_{c_k}^{f,z,QF}, D), \mathcal{Q}_E(t_{c_k}^{f,z,QF}, D)\right]$$
$$\widehat{QE}: \left[c_k, f, z, QF, \hat{\mathcal{Q}}_T(t_{c_k}^{f,z,QF}, D), \hat{\mathcal{Q}}_E(t_{c_k}^{f,z,QF}, D)\right]$$

The best adapted contents obtained by the previously mentioned transcoding systems are computed as follows:

- Exhaustive static system: the best adapted content is identified by solving (1) on the QE array for each slide c_k.
- Granularity-based systems: first, a sub-array is obtained from QE by selecting the rows corresponding to the values of z and QF that define each system (see Table 3). Then, the best adapted content, for each slide c_k, is identified by solving (1) on that sub-array.
- Fixed-QF system: first, a sub-array is obtained from QE by setting the value of z according to the resolution of the target mobile device and $QF = 80$. Then, the best adapted content, for each slide c_k, is identified by solving (1) on that sub-array.
- Proposed dynamic system: the best transcoding parameters $(\widehat{f}^*(c_k, D), \widehat{z}^*(c_k, D), \widehat{QF}^*(c_k, D)))$ are estimated by solving (1) on $\widehat{QE}$ for each slide c_k. Using these optimal parameter estimates, the actual quality of experience is retrieved from the QE array, which corresponds to $\mathcal{Q}_E(t_{c_k}^{\widehat{f}^*(c_k,D),\widehat{z}^*(c_k,D),\widehat{QF}^*(c_k,D)}, D)$. The latter represents the actual quality of experience obtained by the proposed dynamic system, which is compared to those obtained by the other transcoding systems.

5 Experimental Results

To compare the performance and precision of our method with the transcoding systems mentioned in the previous section, the $\mathcal{Q}_E$ average deviation between the best adapted content obtained by each system and that of the exhaustive system was computed. Since the computed data are too numerous to be presented here, we arbitrarily selected one scenario that uses the mobile device and network communication

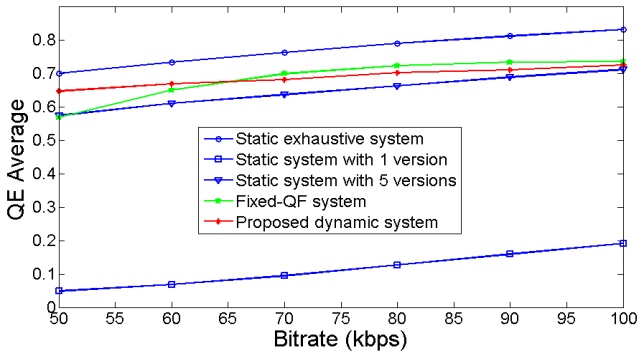


Fig. 8. $\mathcal{Q}_E$ as a function of bitrate for f = JPEG and $NL = 488$ ms

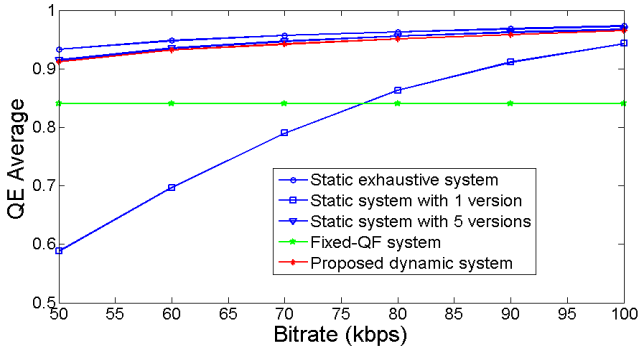


Fig. 9. $\mathcal{Q}_E$ as a function of bitrate for f = XHTML and $NL = 488$ ms

options found in the marketplace. Based on estimated user tolerable waiting times [26], the values of α and β which represent the behavior of the transport quality (see section 3.2) are set as follows: $\alpha = 5$s and $\beta = 10$s. In this scenario, the proposed mobile device D is a Nokia N8, which has a resolution of 640×360, and is connected to a GPRS network that has a bitrate $BR(D) = 50$ kbps and network latency $NL(D) = 488$ ms [28]. Since the default resolution of the slide, as rendered on a PC by the OpenOffice JPEG filter, is 1058×794, the maximum viewing conditions are $\min\left(\frac{640}{1058}, \frac{360}{794}\right) \approx 45\%$, which suggest, from [17], a comparison of images at $z_v = 40\%$. These mobile device and network characteristics are tested using the validation set $\mathcal{V}$. It should be pointed out that similar conclusions are reached with other mobile devices and network conditions when a compromise between visual quality and transfer time must be achieved.

First, the average quality of experience is computed, for various bitrate values for the proposed and the fixed-QF dynamic system, the static exhaustive system as well as the static systems with one and five versions. The results are presented in Fig. 8 for JPEG, and in Fig. 9 for XHTML. As expected, the $\mathcal{Q}_E$ increases with the bitrate up to a point of saturation (quite visible for XHTML and occurring at higher bitrates for JPEG). The average $\mathcal{Q}_E$ values obtained for the proposed dynamic solution are close to those of the exhaustive system for JPEG and very close for XHTML. When f = JPEG, our dynamic solution performs better than the static system with up to five versions and very close to it when f = XHTML.

The $\mathcal{Q}_E$ average deviation (which shows the precision of each system), as computed for this example, is plotted in Fig. 10 for $BR(D) = 50$ kbps and $NL(D) = 488$ ms. This figure shows the difference between our dynamic system, a fixed-QF dynamic system and the static systems (those with N versions and the exhaustive one with 200 versions). Further, it shows the precision of the adapted content achieved for each system by computing the average deviation of the $\mathcal{Q}_E$ for each system from that of the exhaustive one. For this mobile device, when the format used is XHTML, our dynamic system provides better quality than the granularity-based system with $N = 3$ and slightly lower quality compared to the other granularity-based systems (although when N is small, larger fluctuations in quality are observed). All are very close to the optimum. When the format used is JPEG, seven versions are needed by the granularity-based systems to reach the quality obtained by the proposed dynamic system (this number depends on the bitrate, and increases for lower bitrates).
As shown in Figs. 8, 9 and 10, although the $\mathcal{Q}_E$ of our dynamic solution is, on average, significantly better than that of the fixed-QF system for XHTML, the fixed-QF system can be better under some conditions for JPEG. However, these results hide a defect of the fixed-QF system. For a sub-set of documents (slides), we computed the optimal solution provided by our dynamic method, the fixed-QF system and the exhaustive static system. As shown in Figs. 11 and 12, overall, our dynamic method has the same behavior as that of the exhaustive static one. On the other hand, the $\mathcal{Q}_E$ results provided by the fixed-QF system are highly variable. As shown in these figures, the curves follow a certain periodicity due to the nature and order of the documents submitted to the test. Actually, the documents are created by varying the areas taken up by images and text boxes. That is, in the first ten documents, the area of images represents 10% of the slide and the areas of text boxes vary from 10% to 100%. In the second ten documents, the area of images is augmented to 20% and the areas of text boxes are varied form 10% to 100%, and so on.

Of course the dynamic system's performance is highly dependent on the accuracy of the SSIM and file size estimates. With more accurate estimates (and a higher granularity of these estimates, limited here to 10×10 tables from [17], [18]), the proposed system could perform even better. We can also observe that the estimated XHTML data are more precise than the JPEG data (average deviation of about 1% for XHTML compared to 6% for JPEG), and this is explained by the fact that, in the XHTML solution, only the SSIM of the embedded images and the relative file size are estimated (not the textual parts, for which quality and

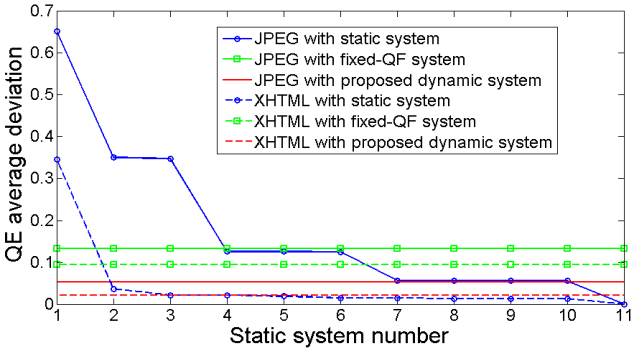


Fig. 10. Average $\mathcal{Q}_E$ deviation from the exhaustive static system computed for $BR = 50$ kbps, $NL = 488$ ms and the entire slide corpus

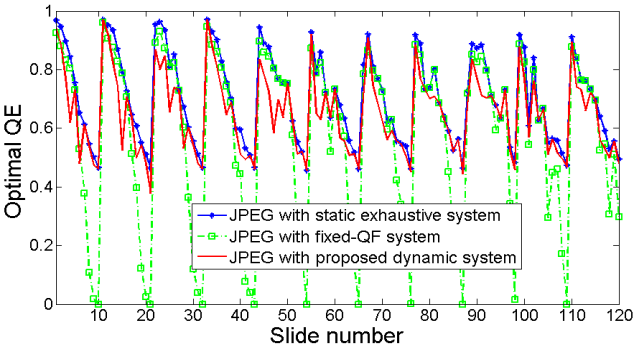


Fig. 11. Optimal $\mathcal{Q}_E$ computed for $BR = 50$ kbps, $NL = 488$ ms and a sub-set of slides. $f =$ JPEG

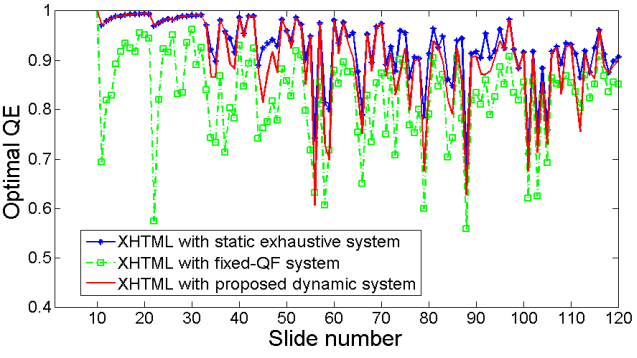


Fig. 12. Optimal $\mathcal{Q}_E$ computed for $BR = 50$ kbps, $NL = 488$ ms and a sub-set of slides. $f =$ XHTML

size are known rather than estimated), whereas in the JPEG solution, the estimated data are computed for the whole slide.

It is important to note that we have been very conservative in the performance evaluation of static systems by assuming that the terminal could render every received image. For example, in this scenario, it was assumed that the static system with $N = 1$ would send a 1058×794 JPEG image which, upon reception, would then be scaled by the terminal to fit its screen resolution. However, in reality, it is possible that none of the versions generated by a static system will be supported by the terminal (especially for low values of N), providing another significant advantage to the

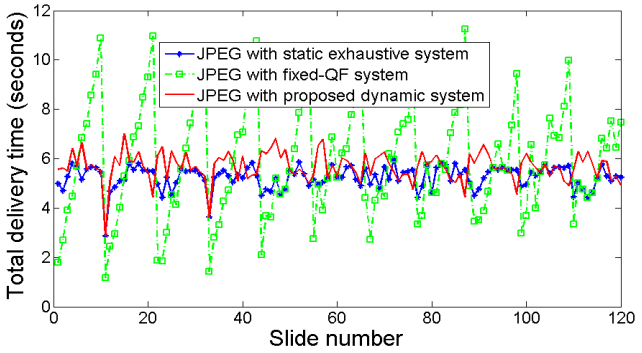


Fig. 13. Total delivery time computed for $BR = 50$ kbps, $NL = 488$ ms and a sub-set of slides. $f =$ JPEG

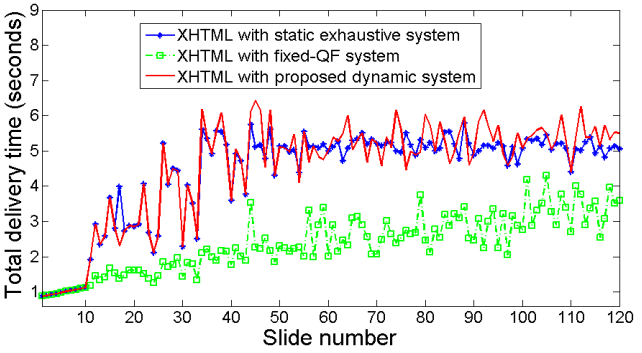


Fig. 14. Total delivery time computed for $BR = 50$ kbps, $NL = 488$ ms and a sub-set of slides. $f =$ XHTML

proposed system, as it sends only content the terminal can support.

Although the fixed-QF dynamic system usually provides good visual quality by setting $QF = 80$, it has no control over the file size, which affects the total delivery time, and therefore the $\mathcal{Q}_E$. This aspect has been tested on the same set of slides by computing, for each system, the total delivery time. Figs. 13 and 14 show the total delivery time required by each slide to be delivered when the format used is JPEG and XHTML, respectively. In these two figures, the proposed dynamic system provides a total delivery time very close to that of the exhaustive static system, whereas the fixed-QF system exhibits a highly variable delivery time (more than 10s in some instances).

Let us now examine the behavior of the storage space needed for the versions created for each transcoding system. To do so, the average file size of the versions created by each system is computed and plotted. Fig. 15 shows the average storage space for the JPEG and XHTML versions. As expected, the XHTML solution is very lightweight, as compared to the JPEG. This is because only the embedded images are rasterized in the XHTML solution, and not the whole slide, as is the case with the JPEG solution. As shown by the curves, our solution becomes increasingly competitive as the granularity of the static

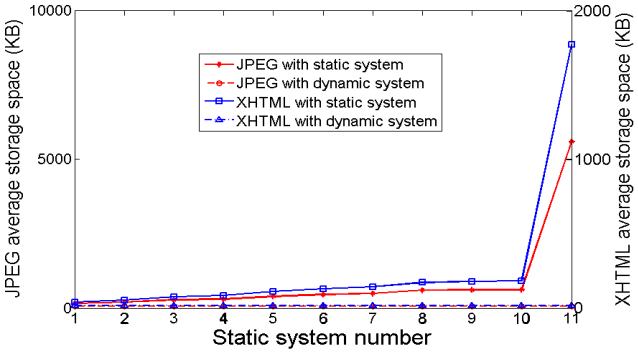


Fig. 15. Average storage space computed for the entire slide corpus

transcoding systems increases. In this example, we stated that when JPEG is used, seven versions are needed to reach our estimated optimal adapted content. If the seventh system is used, the total storage space that should be available is 487 KB on average, whereas in our solution, only one version is created, and only 55.5 KB is needed on average. This means that nearly 9 times more space should be available to accommodate the seventh transcoding system. In the XHTML solution, three versions are needed to achieve our estimated adapted content. In this case, the total space needed is 75.8 KB. The latter is reduced relative to that needed by the JPEG solution, but it is still far more than what is needed by our solution (16.9 KB on average for only one version). Finally, for the same example, ten versions (seven JPEG versions and three XHTML ones) should be created by the static transcoding systems, whereas using our solution, only one version (JPEG or XHTML) is created.

Transcoding many versions is CPU-intensive, and should be performed off-line, which is not always possible. In contrast, our dynamic solution provides near-optimal adapted content on the fly, while the end-user is still on-line. Overall, the proposed solution is very attractive compared to the static transcoding systems previously presented. It achieves a good compromise between performance (little storage space and less processing time) and good quality (close to that of the exhaustive system).

In summary, when the bitrate is very high, there is no significant advantage in terms of quality of experience to select XHTML over JPEG for any of the systems presented. However, XHTML is increasingly attractive as the bitrate becomes smaller since it always ensures crisp and readable text. For static systems, XHTML requires, for a given $\mathcal{Q}_E$ average deviation, significantly fewer versions than JPEG and less storage space. It also offers other advantages, such as being able to edit text. Therefore, it is clear that XHTML is, on average, a better format for sharing presentations than JPEG. Even for the proposed dynamic framework, XHTML leads to more accurate $\mathcal{Q}_E$ prediction and overall system performance. On a final note, the proposed system is not only superior for low bitrate connections, but for any situation where a compromise between transport time and visual quality is required (e.g., high definition content over 3G networks). Yet, it performs competitively in other situations.

6 Complexity of the Proposed Method

In contrast with the exhaustive static system, which requires 200 transcoding operations, the proposed dynamic framework requires a single transcoding operation (performed after estimation of the transcoding parameters). To that, we must add the computation of the $\hat{\mathcal{Q}}_V$ array, which can be computed off-line; $\hat{\mathcal{Q}}_T$ and $\hat{\mathcal{Q}}_E$ arrays, that are performed at runtime, and a look-up search in the $\hat{\mathcal{Q}}_E$ array. These added computations are very light compared to a single transcoding operation, and can be negligible on high-end servers. The fixed-QF system also requires a single transcoding operation, but, as shown, exhibits a highly variable quality. The granularity-based systems require more than one transcoding operation and are thus more complex. The proposed system offers exceptional quality at minimal computational complexity.

7 Conclusion

In this paper, we presented a novel dynamic content adaptation framework applied to professional documents shared using Web technologies. The method estimates, on the fly, near-optimal transcoding parameters based on the original content and its composition (areas occupied by text and images). We have proposed a measure of the quality of experience taking into account the visual aspect of the transcoded content, in addition to the total delivery time. The proposed measure, although useful, is mostly illustrative, as the proposed framework is general, and can be used with other quality measures. The validation results show that dynamic content adaptation based on accurate prediction can provide a good compromise between quality and delivery time, and drastically reduce storage requirements. Even though the proposed framework is quite general, future research can be performed to show and validate its applicability to other professional documents such as Word and Excel.

References

[1] D. Sudhir and W. Tao, *Content Networking in the Mobile Internet, Chapter 7: Content Adaptation for the Mobile Internet*. John Wiley and Sons, 2004.

[2] W. Lum and F. Lau, "User-Centric Adaptation of Structured Web Documents for Small Devices," in *19th Int. Conf. on Advanced Information Networking and Applications (AINA'05)*, vol. 1. IEEE, 2005, pp. 507–512.

[3] ——, "User-Centric Content Negotiation for Effective Adaptation Service in Mobile Computing," *IEEE Transactions on Software Engineering*, vol. 29, no. 12, pp. 1100–1111, Dec 2003.

[4] Y. Zhang, S. Zhang, and S. Han, "A New Methodology of QoS Evaluation and Service Selection for Ubiquitous Computing," in *Wireless Algorithms, Systems, and Applications*, ser. LNCS. Springer Berlin / Heidelberg, 2006, vol. 4138, pp. 69–80.
[5] Y. Hwang, J. Kim, and E. Seo, "Structure-Aware Web Transcoding for Mobile Devices," *IEEE Internet Computing*, vol. 7, no. 5, pp. 14–21, 2003.
[6] H. Chua, S. Scott, Y. Choi, and P. Blanchfield, "Web-page Adaptation Framework for PC Mobile Device Collaboration," in *Advanced Information Networking and Applications, AINA 2005. 19th Int. Conference on*, vol. 2, Mar 2005, pp. 727–732.
[7] J. Hong, E. Suh, and S. Kim, "Context-aware Systems: A Literature Review and Classification," *Expert Systems with Applications*, vol. 36, no. 4, pp. 8509–8522, 2009.
[8] R. Mohan and J. Smith, "Adapting Multimedia Internet Content for Universal Access," *IEEE Transactions on Multimedia*, vol. 1, no. 1, pp. 104–114, Mar 1999.
[9] B. Noble, M. Price, and M. Satyanarayanan, "A Programming Interface for Application-Aware Adaptation in Mobile Computing," *2nd USENIX Symposium on Mobile and Location Independent Computing*, vol. 8, no. 4, pp. 57–66, 1995.
[10] R. Jan, C. Lin, and M. Chern, "An Optimization Model for Web Content Adaptation," *Computer Networks*, vol. 50, no. 7, pp. 953–965, 2006.
[11] S. Chandra and C. S. Ellis, "JPEG compression metric as a quality-aware image transcoding," in *Proc. of the 2nd conference on USENIX Symposium on Internet Technologies and Systems*, 1999, pp. 81–92.
[12] F. Kitayama, S. Hitose, G. Kondoh, and K. Kuse, "Design of a Framework for Dynamic Content Adaptation to Web-enabled Terminals and Enterprise Applications," *Proceedings Sixth Asia Pacific Software Engineering Conference ASPEC99 Cat NoPR00509*, pp. 72–79, 1999.
[13] W. Lum and F. Lau, "On Balancing Between Transcoding Overhead and Spatial Consumption in Content Adaptation," *Proceedings of the 8th annual international conference on Mobile computing and networking MobiCom 02*, p. 239, 2002.
[14] R. Han, P. Bhagwat, R. LaMaire, T. Mummert, V. Perret, and J. Rubas, "Dynamic Adaptation in an Image Transcoding Proxy for Mobile Web Browsing," *IEEE Personal Communications*, vol. 5, no. 6, pp. 8–17, 1998.
[15] D. Li and U. Chandra, "Building Web-based Collaboration Services on Mobile Phones," in *2008 Int. Symp. on Collaborative Technologies and Systems*. IEEE, May 2008, pp. 295–304.
[16] Google, "GoogleDocs Mobile." [Online]. Available: http://www.google.ca/mobile/docs/index.html
[17] S. Coulombe and S. Pigeon, "Quality-aware Selection of Quality Factor and Scaling Parameters in JPEG Image Transcoding," in *2009 IEEE Symp. on Computational Intelligence for Multimedia Signal and Vision Processing*, Mar 2009, pp. 68–74.
[18] S. Pigeon and S. Coulombe, "Computationally Efficient Algorithms for Predicting the File Size of JPEG Images Subject to Changes of Quality Factor and Scaling," in *2008 24th Biennial Symposium on Communications*. IEEE, Jun 2008, pp. 378–382.
[19] S. Coulombe and S. Pigeon, "Low-complexity Transcoding of JPEG Images With Near-optimal Quality Using a Predictive Quality Factor and Scaling Parameters." *Image Processing, IEEE Transactions on*, vol. 19, no. 3, pp. 712–721, Mar 2010.
[20] F. Kuipers, R. Kooij, D. De Vleeschauwer, and K. Brunnström, "Techniques for Measuring Quality of Experience," in *Wired/Wireless Internet Communications*, ser. Lecture Notes in Computer Science. Springer-Verlag Berlin/Heidelberg, 2010, vol. 6074, pp. 216–227.
[21] L. Lee and R. Anderson, "A Comparison of Compensatory and Non-Compensatory Decision Making Strategies in IT Project Portfolio Management," 2009. [Online]. Available: http://aisel.aisnet.org/irwitpm2009/9
[22] A. Dieckmann, K. Dippold, and H. Dietrich, "Compensatory versus Noncompensatory Models for Predicting Consumer Preferences," *Judgment and Decision Making*, vol. 4, no. 3, pp. 200 –213, 2009.
[23] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity." *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
[24] S. Pigeon and S. Coulombe, "Efficient Clustering-based Algorithm for Predicting File Size and Structural Similarity of Transcoded JPEG Images," *Multimedia, Int. Symp. on*, pp. 137–142, Dec 2011.
[25] The MathWorks, "Z-shaped built-in membership function." [Online]. Available: http://www.mathworks.com/help/toolbox/fuzzy/zmf.html
[26] F. Nah, "A study on Tolerable Waiting Time: How Long Are Web Users Willing to Wait?" *Behaviour & Information Technology*, vol. 23, no. 3, pp. 153–163, Jan 2004.
[27] H. Ningning and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 879–894, Aug 2003.
[28] P. Svoboda, F. Ricciato, W. Keim, and M. Rupp, "Measured WEB Performance in GPRS, EDGE, UMTS and HSDPA with and without Caching," in *IEEE Int. Symp. on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Jun 2007, pp. 1–6.
[29] OpenOffice.org, "The OpenOffice.org API Project." [Online]. Available: http://api.openoffice.org
[30] University of Southern California, Signal and Image Processing Institute. Electrical Engineering Department, "The USC-SIPI Image Database." [Online]. Available: http://sipi.usc.edu/database/database.cgi?volume=misc\&image=11
[31] ImageMagick, "ImageMagick Command Line Tools." [Online]. Available: http://www.imagemagick.org/script/index.php
[32] H. Sheikh, Z. Wang, L. Cormack, and A. Bovik, "LIVE Image Quality Assessment Database Release 2." [Online]. Available: http://live.ece.utexas.edu/research/quality/subjective.htm

Habib Louafi (S'11) received an Engineering degree in Computer Science from the University of Oran (Algeria) in 1993, an M.Sc. from the Université du Québec à Montréal (UQÀM) in 2006. He is currently working toward a Ph.D. at the École de technologie supérieure (ÉTS is a part of the Université du Québec network). His fields of interest include mobile computing, context-aware systems, QoE, content adaptation and collaborative mobile Web conferencing.

Stéphane Coulombe (S'90-M'98-SM'01) received a B.Eng. in Electrical Engineering from the École Polytechnique de Montréal, Canada, in 1991, and a Ph.D. from INRS-Telecommunications, Montréal, in 1996. He is a Professor at the Software and IT Engineering Department, École de technologie supérieure (ÉTS is a part of the Université du Québec network). From 1997 to 1999, he was with Nortel Wireless Network Group in, Montreal, and from 1999 to 2004, he worked with the Nokia Research Center, Dallas, TX, as Senior Engineer and as Program Manager in the Audiovisual Systems Laboratory. He joined ÉTS in 2004, where he currently carries out research and development on video processing and systems, media adaptation, and transcoding. Since 2009, he has held the Vantrix Industrial Research Chair in Video Optimization.

Umesh Chandra received an MS in Computer Science (CS) from University of North Texas in 1996 and BS in CS from Osmania University in 1993. He is currently working as research lead at Nokia Research, Bangalore, India. He is currently working on developing mobile services targeting emerging markets in the domain of Social and Location. Prior to this he has worked on Mobile video conferencing and developing standards in the area of video transport. His area of interest is in mobile technologies, IP Multimedia, data management, E2E service creation, Emerging markets.