Multimedia Tools and Applications manuscript No.
(will be inserted by the editor)

Robust QoE-aware Prediction-based Dynamic Content Adaptation Framework Applied to Slides Documents in Mobile Web Conferencing

Habib Louafi · Stéphane Coulombe · Umesh Chandra

Received: date / Accepted: date

Abstract In mobile Web conferencing, enterprise documents are generally adapted into JPEG-based Web pages to be visualized on mobile devices. Dynamically identifying the optimal adapted content is very challenging, as a compromise between high visual quality and small delivery time must be made. In this paper, we propose a robust prediction-based dynamic content adaptation framework for JPEG and XHTML formats that computes near-optimal transcoding parameters dynamically with very few computations. The proposed framework is comprised of five methods making different compromises between computational complexity and accuracy. For JPEG, the average deviation from optimality (exhaustive method) is 6% and 3% respectively for two of the proposed methods. For XHTML, the average deviation from optimality is 3% and 1% respectively using the same two methods. Moreover, the methods reached optimality 30% and 59% of the time on the tested documents for JPEG and XHTML respectively, which makes the proposed framework very appealing.

Keywords Ubiquitous computing · context-awareness · mobile Web conferencing · dynamic content adaptation · QoE · JPEG · XHTML

H. Louafi
Synchromedia Laboratory for Multimedia Communication in Telepresence,
École de technologie supérieure, Université du Québec, Montreal, Canada
E-mail: habib.louafi.1@ens.etsmtl.ca

S. Coulombe
Department of Software and IT Engineering,
École de technologie supérieure, Université du Québec, Montreal, Canada
E-mail: stephane.coulombe@etsmtl.ca

U. Chandra
Nokia Research Center, Palo Alto, CA, USA
E-mail: umesh.1.chandra@nokia.com

1 Introduction

1.1 Motivation

Enterprise documents, such as PowerPoint slides and Word documents are widely used and shared between peers in many Web collaborative applications (e.g., GoogleDocs and Zoho). This has been largely facilitated by the ubiquity of the Web used as a content delivery platform. Besides, the Web has allowed the holding of collaborative meetings with slide decks shared and presented synchronously to all participants, connected via their Web browsers. In this regard, many solutions have been proposed for PCs and laptops [27,1]. However, when mobile devices are involved in such conferencing applications, the content (e.g.: presentation slides) must be adapted to meet the target mobile device constraints (supported formats, maximum resolution and file size) [22,20,11], but more importantly, to provide the end-user with the best experience possible.

Enterprise documents adaptation used in current products is not device independent; that is, documents are optimized only for a few specific mobile devices [7, 35]. The adaptation of Web content has been studied extensively and numerous solutions have been proposed. However, in spite of the ubiquitous use of enterprise documents, adapting them has not attracted the same attention as audiovisual adaptation, and their portability to mobile Web browsers and their interoperability are still a challenge.

In this paper, we concentrate on the dynamic adaptation of slide decks for Web enabled mobile devices. We also focus our work on JPEG and XHTML formats. The former is a required format, since it is widely supported by mobile devices and used in so many Web applications [7,35,16]. However, this is a raster format lacking interactivity and scalability. Consequently, we consider the use of XHTML as well, which is more flexible, in that it allows text editing and keyword searching. Technically, when JPEG is used, an entire document page is converted into a JPEG image and wrapped in a Web page skeleton. In contrast, when XHTML is used, that page's embedded components are converted separately, and individually wrapped in a Web page. Since presentation slides are mostly composed of text and images, the embedded images are converted into JPEG images and the text elements are resized, which leads to the creation of a conventional Web page, comprising both text and images. As a result, in this paper, we consider only slide decks composed of text and image components.

1.2 Static and dynamic adaptation

Globally, two major trends have emerged in content adaptation: static adaptation and dynamic adaptation. In the area of static content adaptation different versions of the original content are created and stored on a server [24,22,20,11,34]. At runtime, when the content is requested, the optimal version, evaluated using a specific quality criterion, is selected for delivery. The static content adaptation approach leads to high processing complexity for generating all the versions and a great deal of disk space to store them, and the adaptation is often performed offline. The issue of granularity becomes important, as it determines the compromise between the quality of the delivered content (the more versions are available, the better the quality) and the associated processing complexity, as well as the storage space available [19]. In dynamic content adaptation, also called just-in-time adaptation, a customized version is created on-the-fly, based on the target mobile device's context (resolution, memory size, etc.) [12, 8, 10]. With the dynamic approach, the adaptation is performed on-the-fly, when the terminal's context is known, while the end-user waits. With this approach, we can eliminate the need to store several versions on disk as well as reducing the number of transcoding operations. But, this is a very challenging task, as the time required to adapt and deliver the content is extremely important. In this case, the server could easily be overwhelmed with a large number of requests, which would negatively affect the end-user's experience [2, 19]. This is another reason why reducing the amount of computations in the dynamic approach is very important.

To adapt a document dynamically (on-the-fly), the desired transcoding parameter values must be determined. These comprise the selected output format (JPEG or XHTML), the resolution scaling parameter, and the JPEG quality factor. Most of the existing dynamic solutions use the mobile device's resolution as the target image resolution to determine the scaling parameter and a quality factor value between 75 and 80. Although such solutions provide good visual quality, they don't control the resulting file size, which has a negative impact on the delivery time and the usability of the adapted content. That is, under certain network conditions, the user might even lose interest in that content, owing to an unreasonable wait time. A better user experience can often be achieved by reducing the visual quality slightly, in order to permit a shorter delivery time. In other words, we need to identify the adapted content that provides the best compromise between the visual quality of the adapted content and the time it takes to reach the recipient. Such a compromise has been considered in context-aware adaptive E-learning applications [13], and in the adaptation of mobile users' interfaces [9].

1.3 Quality of experience

To compute the best adapted content, we need to define an objective measure that quantifies the end-user's experience, and then determine the optimal transcoding parameter values maximizing this measure with as little computational complexity as possible. Therefore, in this paper, we propose to use a quality of experience measure (QoE) that provides a compromise between the visual aspect of the adapted content and its delivery time.

Although performing an actual transcoding operation for each combination of transcoding parameter values and identifying the combination that maximizes the defined measure is theoretically feasible, it is far too complex to implement in a viable communication system. This is because the number of combinations of transcoding parameter values grows exponentially with the number of parameters, and because the adaptation itself is performed online while the end-user waits. Such an exhaustive approach would require far too much computational complexity to be a viable solution. While a certain number of transcoding operations is required for the adaptation, that number should be minimized if possible [22, 21, 11]. Clearly, we can expect that compromises will need to be made between the computational complexity permitted in searching for the optimal parameter values and how far from the optimal value our solution will be.

1.4 Objectives and contributions

Based on the proposed QoE measure, we propose a robust QoE-aware prediction-based framework which has the advantages of the two previously presented approaches (static and dynamic), without their drawbacks. The proposed framework allows the computation of near-optimal (compared to static approach) transcoding parameters at delivery time without performing exhaustive transcoding operations. The proposed dynamic framework is comprised of five methods. The first one, which is presented in [18], produces near-optimal transcoding parameters dynamically with a single transcoding operation. In most cases, the estimated results were very reliable and close to those obtained using an exhaustive static approach, which is considered to be optimal. However, for certain documents, the estimated results were not close to optimality (especially when using the JPEG format), for several reasons. First, the visual quality prediction errors were amplified by the conversion of the estimated visual quality values to human perception scores. Similarly, file size estimation errors were amplified when the Zmf function (*Z-shaped built-in membership function*) [31] was used to model the transport quality as a function of the delivery time. Finally, these errors were further amplified by the multiplication used to combine the visual and transport qualities in the QoE computation.

Therefore, in this paper, we improve the accuracy of the estimated results obtained by the proposed dynamic framework by allowing the system to perform more than one transcoding operation. For JPEG, we showed in [17] that it is possible to improve significantly the estimated results if a small number (1 to 5) of transcoding operations is performed. However, additional experiments are required to show the behaviour of the proposed framework in different network conditions. Besides, we want to improve the estimated results in the case of XHTML as well. Therefore, we modify slighly the estimation method presented in [18] to improve the estimated quality of the documents that were not close to optimality. Then, based on this method, we present four other methods, which are variants of the first one, but with improved accuracy. Each method offers an interesting compromise between performance (how close it is to optimality) and complexity (number of transcoding operations required). In this framework we exploit visual quality and file size predictors of JPEG images subject to changing their resolution and quality factor [26,3,4].

We show that the quality of the content adapted using our framework is near-optimal. For instance, with JPEG, the results obtained were 6% and 3% far from optimality respectively, using methods 2 and 5. In the case of XHTML, they were 3% and 1% far from optimality, using methods 2 and 5 respectively. More importantly, from the set of documents tested, we were able to reach optimality in 30% and 59% of the documents for JPEG and XHTML respectively, which make the proposed framework very attractive.

1.5 Structure of the paper

This paper is structured as follows. In section 2, we present a global overview of the proposed framework architecture as well as the adaptation process. In section 3, we review, in more details, the problem statement, which was presented in [18]. In section 4, we present the proposed quality of experience measure. Section 5 details the proposed prediction-based framework as well as various methods it comprises. In section 6, we present the experimental setup used to validate the performance of our framework. The experimental results are presented and discussed in section 7. Lastly, section 8 concludes the paper.

2 Overview of the proposed adaptation framework

The overall architecture of the proposed content adaptation framework is presented in Fig. 1. It is comprised of four tiers:

1. **The client tier** is represented by the mobile Web browser.
2. **The server tier** is designated by the Web server and comprised mainly of a request handler (e.g. servlet) and a bean application. The request handler plays the role of middleware between the client tier and the business tier, but via a bean, which can be a java class helper. The bean parses the information received from the servlet, extracts the context information(the user's profile, the mobile terminal characteristics, the requested content, etc.), converts it into a processable format (e.g. JSON or XML), and sends it to the business tier.
3. **The business tier** is represented by the transcoding engine and comprises two units: a decision unit (DU), and a content adaptation unit (CAU). The DU computes the best, ideally optimal, transcoding parameters using the context information received from the request handler, and the composition of the requested content. The CAU converts the content into an adapted version, using the transcoding parameters, to be returned to the terminal.
4. **The persistence tier** consists of a repository database used to store content, such as presentations loaded by users and their adapted versions (along with the transcoding parameters used in their creation). If desired, these adapted contents can be delivered to other users, depending on the policy adopted, that is, always transcoding or, when possible, selecting a version that has already been created and stored.

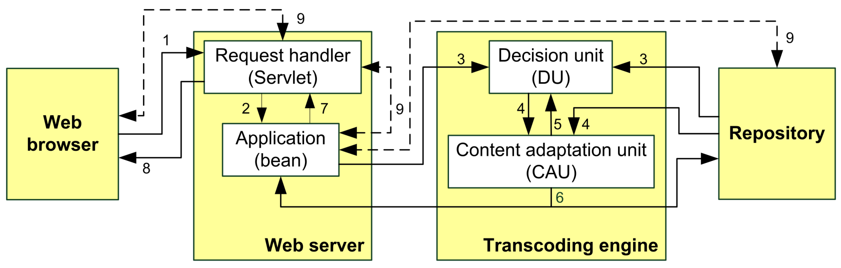


Fig. 1: Architecture of the proposed dynamic content adaptation framework.

To be processed, the user's request goes through several steps, as shown in Figure 1. Typically, these steps are the following:

1. An HTTP request is sent to the Web server to download the main page (XHTML) of the Web application (e.g. meeting application).
2. The servlet receives the request, extracts the important data, such as the user profile and the file name of the document requested, among other things. Then, these data are sent to the bean.
3. The DU receives the data and requests certain meta data of the document to be adapted (its composition), which have already been extracted from the document and stored in the repository. Using these data, it computes the best, ideally optimal, set of transcoding parameters and sends them to the CAU.
4. The CAU requests the enterprise document from the repository, and uses the transcoding parameters received from the DU to generate an adapted content.
5. The characteristics of the created adapted content are extracted and compared to the terminal's constraints to ensure that it satisfies those requirements. If it does not, the DU must calculate new transcoding parameters until the terminal's constraints are met.
6. Once a compliant adapted content has been created, it is first stored in the repository (along with the parameters used in its creation), and then sent back to the bean in the Web server.
7. The bean receives the adapted content and sends it back to the servlet.
8. The servlet forwards the adapted content to the mobile terminal.
9. The dashed line in the figure represents the information being sent back and forth between the mobile terminal and the persistent adapted content. This sequence can occur if the user has received an editable version of the content. Each time the content is updated, a request (XMLHttpRequest) is fired to update the content in the repository. The content sent from the servlet to the Web browser is received first by an AJAX engine, which decides how the data should be presented on the Web browser, and interprets the user's interactions with the content.

2.1 The content adaptation process

From the architecture presented in Fig. 1, we focus on the business tier, which comprises the transcoding engine and its components, as depicted in Fig. 2. To highlight what is novel in this architecture, we present the transcoding engine and the elements with which it interacts, namely the Web server and the preprocessing subsystem.

2.1.1 The preprocessing subsystem

In most mobile Web conferencing services, a PowerPoint presentation is uploaded by the presenter prior to the meeting. To save the precious time of the transcoding engine, the document is processed offline (before making it available to users) to extract its characteristics. These are used later by the transcoding engine when the document is requested. This feature is called the *Preprocessing subsystem.*

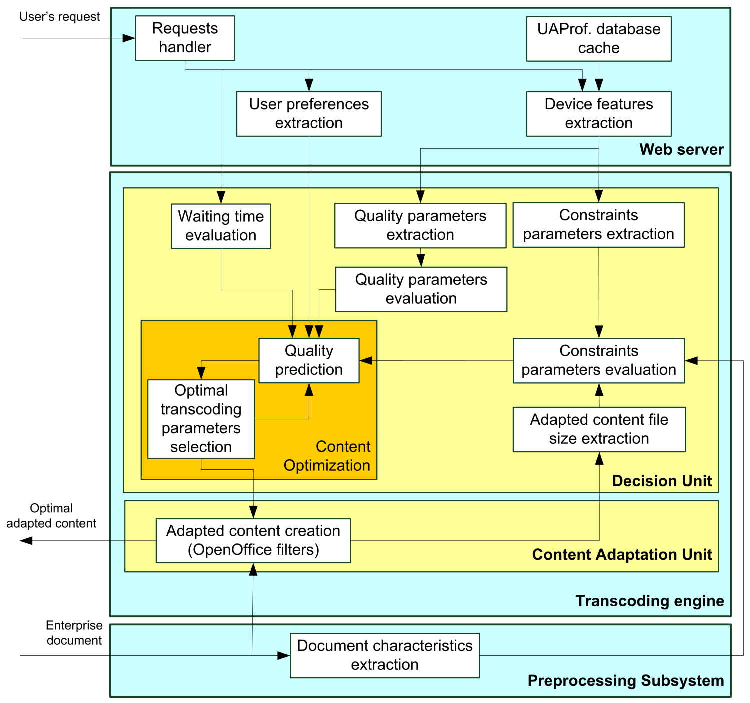

User's request
Requests handler
UAProf. database cache
User preferences extraction
Device features extraction
Web server
Waiting time evaluation
Quality parameters extraction
Constraints parameters extraction
Quality parameters evaluation
Quality prediction
Constraints parameters evaluation
Optimal transcoding parameters selection
Content Optimization
Adapted content file size extraction
Decision Unit
Optimal adapted content
Adapted content creation (OpenOffice filters)
Content Adaptation Unit
Transcoding engine
Enterprise document
Document characteristics extraction
Preprocessing Subsystem

Fig. 2: Proposed prediction-based dynamic content adaptation system.

2.1.2 The Web server

When the enterprise document is requested, the users' requests are captured on the Web server by a *request handler* module, which keeps track of these requests and computes the time they spent on the server before they were processed. The device's capabilities and the user preferences are extracted and resolved on the Web server, with the option of using a cached UAProf database. These data, comprising the device capabilities, the user preferences, and the waiting time are sent to the decision unit in the transcoding engine. Although the user's preferences are presented in the architecture for completeness, we do not focus on this aspect in the content adaptation process proposed in this framework. Such user's preferences can be conveyed using the mechanisms presented in [5], including HTTP's User-Agent header and UAprof.

2.1.3 The transcoding engine

The core of the proposed architecture is the transcoding engine, which consists of the DU and the CAU. In the DU, the constraints and quality parameters are extracted from the device capabilities by the *Constraints parameters extraction* and the *Quality parameters extraction* modules respectively. Then, they are evaluated by the *Constraints parameters evaluation* and *Quality parameters evaluation* modules respectively. To be accepted by the target mobile terminal, the adapted content should satisfy its constraints (memory, resolution, etc.). The quality parameters are used in the computation of the quality of each adapted content, allowing the identification of the optimal one. Besides, the time spent by the user's request in the server before it is processed is evaluated by a *Waiting time evaluation* module. After these data have been evaluated, they are sent to the *Content optimization* module, which comprises a *Quality prediction* and a *Optimal transcoding parameters selection* module. The first module uses the constraints and quality parameters evaluated, the user preferences, and the waiting time to predict the quality of the adapted content as a function of a set of transcoding parameter combinations. The second module selects the optimal transcoding parameter combination that corresponds to the best predicted quality. These parameters are then used by the CAU to adapt the enterprise document into the best adapted content version (it is expected to be near optimal, as it is based on predicted transcoding parameters). Before sending it back to the end-user, the adapted content is sent to the DU, where its file size is extracted and evaluated against the mobile terminal memory. If this evaluation fails, another set of transcoding parameters should be predicted and sent to the CAU. Lastly, when an acceptable version of the content is created, it is sent back to the target mobile device and cached in the repository.

In the remaining sections, we focus on the *Content optimization* module, that is, the computation of the optimal transcoding parameters.

3 Problem statement

Let $\mathcal{C}$ be an enterprise document, referred to here as "the original document" or "the content", composed of a set of pages (or slides) c_k made up of various components $c_{k,i}$ (e.g. text or images). We can write this formally as follows:

$$\begin{aligned} \mathcal{C} &= \{c_k\}_{k=1}^{n} \\ c_k &= \{c_{k,i}\}_{i=1}^{m(k)} \end{aligned} \tag{1}$$

where n is the total number of pages in $\mathcal{C}$, and $m(k)$ is the total number of components of the k^{th} page. For instance, $\mathcal{C}$ 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.

To be rendered by the target mobile device, the original document must often be adapted. Various adaptation operations can be used to achieve this, and, in principle, different transcoding parameter combinations can be used by the adaptation operations for each page. In this paper, for simplicity we concentrate on the adaptation of enterprise documents comprising text and JPEG images, bearing in mind that the concepts can be extended to other media types and formats. For instance, images encoded in other formats (e.g. GIF) can be converted into JPEG using a quality factor value of 80 to preserve their original quality. Regarding the output formats into which enterprise documents can be adapted, two formats are considered in this research, namely, JPEG- and XHTML-based Web pages. Let $\mathcal{P}$ be the possible transcoding parameters that can be used to adapt the original document's pages and their components. Thus, we have:

$$\mathcal{P} = \{f, z, QF\} \tag{2}$$

where:

- $f \in \{\text{JPEG}, \text{XHTML}\}$ is the output format into which the original page is to be transcoded.
- $z \in [0, 1]$ is a scaling factor that defines the output resolution of the adapted page.
- $QF \in [0, 100]$ represents the quality factor of the outputted JPEG images on the adapted page.

Depending on the output format, the transcoding parameters of $\mathcal{P}$ 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 adapted 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 adapted 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.

We define $\mathcal{T}$ as the transcoding operation that adapts the document page (or slide) c_k into a Web page using the transcoding parameters f, z, and QF, as follows:

$$\begin{aligned} \mathcal{T} : \mathcal{C} \times \mathcal{P} &\to \mathcal{C}^{f,z,QF} \\ c_k \times (f, z, QF) &\mapsto c_k^{f,z,QF} \end{aligned} \tag{3}$$

where $\mathcal{C}^{f,z,QF}$ is the set of all the possible adapted content versions that can be created by $\mathcal{T}$ from $\mathcal{C}$, using all the parameters from $\mathcal{P}$. In other words, $c_k^{f,z,QF}$ represents the adapted content version of c_k created by $\mathcal{T}$ using f, z, and QF.

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

Let D be the target mobile device and $\mathcal{W}(D)$, $\mathcal{H}(D)$, $\mathcal{S}(D)$, and $\mathcal{F}(D)$ be its maximum permissible image width, image height, file size (in bits), and supported formats respectively.

From the set of adapted content versions that can be created from c_k using $\mathcal{T}$, only a subset can be rendered by D. Let $\mathcal{R}(c_k, D)$ be the set of transcoding parameter combinations that can be used to create these renderable versions:

$$\begin{aligned} \mathcal{R}(c_k, D) = \Big\{ (f, z, QF) \mid \mathcal{W}(c_k^{f,z,QF}) = z\mathcal{W}(c_k) \leq \mathcal{W}(D) \text{ and} \\ \mathcal{H}(c_k^{f,z,QF}) = z\mathcal{H}(c_k) \leq \mathcal{H}(D) \text{ and} \\ \mathcal{S}(c_k^{f,z,QF}) \leq \mathcal{S}(D) \text{ and} \\ f \in \mathcal{F}(D) \Big\} \end{aligned} \tag{4}$$

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

Since there could be multiple transcoding parameter combinations leading to adapted content versions renderable by D, the objective is to compute the ones that maximize the user's quality of experience, which we denote here by $\mathcal{Q}_E(c_k^{f,z,QF}, D)$, and which will be defined in the next section. Let $\mathcal{R}^*(c_k, D) \subseteq \mathcal{R}(c_k, D)$ be the subset of optimal transcoding parameter combinations that maximize $\mathcal{Q}_E(c_k^{f,z,QF}, D)$. It is given by:

$$\begin{aligned} \mathcal{R}^*(c_k, D) &= \Big\{ \big(f^*(c_k, D), z^*(c_k, D), QF^*(c_k, D)\big) \Big\} \\ &= \underset{(f,z,QF)\in\mathcal{R}(c_k,D)}{\arg\max} \mathcal{Q}_E(c_k^{f,z,QF}, D) \end{aligned} \tag{5}$$

Note that there may be several solutions to (5). In this case, the parameters leading to the best visual quality are arbitrarily selected.

4 Proposed quality of experience measure

The quality of the delivered content, as experienced by the end-user, $\mathcal{Q}_E$, is affected by three factors [14]:

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 of the adapted content.

In other words, $\mathcal{Q}_E$ is affected by the visual quality and 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, for a target mobile device D, we propose to evaluate the $\mathcal{Q}_E$ of the adapted content $c_k^{f,z,QF}$ as follows:

$$\mathcal{Q}_E(c_k^{f,z,QF}, D) = \mathcal{Q}_V(c_k^{f,z,QF}, D)\mathcal{Q}_T(c_k^{f,z,QF}, D) \tag{6}$$

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 [14], the $\mathcal{Q}_E$ and the QoS evaluation are completely separate research topics. In our framework, we propose the product of $\mathcal{Q}_V$ and $\mathcal{Q}_T$ rather than their sum, to prevent large disparities in $\mathcal{Q}_V$ and $\mathcal{Q}_T$ from being able to produce a high $\mathcal{Q}_E$. Indeed, the product is more appropriate than the sum, since $\mathcal{Q}_V$ and $\mathcal{Q}_T$ are not compensatory attributes. In our problem here, when a JPEG image is aggressively transcoded, its $\mathcal{Q}_T$ will be close to 1 (a very lightweight image) and its $\mathcal{Q}_V$ close to 0 (a 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. Unlike the sum, the product will be close to 0, which is more reasonable. In fact, before combining two or more attributes to obtain a single measure that reflects the nature of the problem in context, these attributes should first be classified into compensatory and non compensatory attributes. The former can be summed, whereas the latter cannot.

This is the fruit of research performed elsewhere, particularly in the marketing and decision making fields [15,6].

Although further research and validation are required to establish a metric that accurately matches the user's experience, the proposed metric is adopted here to illustrate the benefits over existing methods of performing prediction-based dynamic content adaptation. Similar benefits are expected with other metrics which consider a compromise between visual quality and delivery time.

4.1 Visual quality evaluation

Let $c_k = \{c_{k,i}\}_{i=1}^{m(k)}$ be a page composed of a set of components, and $c_k^{f,z,QF}$ its adapted version created by $\mathcal{T}$, which comprises a set of adapted components and can be formulated as follows:

$$c_k^{f,z,QF} = \left\{c_{k,1}^{f,z,QF}, c_{k,2}^{f,z,QF}, \ldots, c_{k,m(k,f)}^{f,z,QF}\right\} \tag{7}$$

where $c_{k,i}^{f,z,QF}$ is the i^{th} transcoded component and $m(k,f)$ is the total number of components. Ignoring the XHTML wrapper, which has no impact on quality in either case, $m(k,f)$ is given by:

$$m(k,f) = \begin{cases} m(k) & \text{if } f = \text{XHTML} \\ 1 & \text{if } f = \text{JPEG} \end{cases} \tag{8}$$

Of course, the visual quality of the adapted content depends on the visual quality of its components, but it also depends on the area occupied by each component (the larger the area, the larger the weight it should have in terms of quality). Therefore, we propose to compute the visual quality as a weighted sum of the visual quality of each of its components, each weight being the area that they occupy. So, we have:

$$\mathcal{Q}_V(c_k^{f,z,QF}, D) = \frac{\displaystyle\sum_{i=1}^{m(k,f)} \mathcal{A}(c_{k,i}^{f,z,QF})\mathcal{Q}_V(c_{k,i}^{f,z,QF}, D)}{\displaystyle\sum_{i=1}^{m(k,f)} \mathcal{A}(c_{k,i}^{f,z,QF})} \tag{9}$$

$$\mathcal{Q}_V(c_{k,i}^{f,z,QF}, D) = \begin{cases} \mathcal{Q}_I(c_{k,i}^{f,z,QF}, D) & \text{if } c_{k,i}^{f,z,QF} & \text{is an image} \\ 1 & \text{if } c_{k,i}^{f,z,QF} & \text{is text} \end{cases} \tag{10}$$

where:

- $\mathcal{Q}_I$ measures image quality, such as PSNR or SSIM [33] (or any other reliable full reference objective metric).
- We have assumed that the text is rendered perfectly, and, without loss of generality, the visual quality of the text components is set to 1. However, more sophisticated metrics could be used to take into account the resizing of the text components. The other textual characteristics, such as color, font, and the like, should not be affected, in order to preserve the original intentions of the document's author.

- $\mathcal{A}(c_{k,i}^{f,z,QF})$ is the visible (not hidden) area occupied by $c_{k,i}^{f,z,QF}$. We always have $\mathcal{A}(c_{k,i}^{f,z,QF}) \leq \mathcal{H}(c_{k,i}^{f,z,QF})\mathcal{W}(c_{k,i}^{f,z,QF})$, 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 they do, the hidden regions of an image should not be considered for computing either its region $\mathcal{A}$ or its visual quality $\mathcal{Q}_{I}$. This is particularly important when the page contains a background.
- When the adapted content $c_{k}^{f,z,QF}$ 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 $f =$ JPEG.

4.2 Transport quality evaluation

The second factor that affects the user's quality of experience is transport quality. This factor is itself affected by the total delivery time, which is made up of the time required to perform the adaptation operation, plus the time taken by the adapted content to reach the target mobile device. For adapted content $c_{k}^{f,z,QF}$ and a target mobile device D, the total delivery time T_d can defined as:

$$T_d\big(c_{k}^{f,z,QF}, D\big) = \frac{\mathcal{S}\big(c_{k}^{f,z,QF}\big)}{N_{B}(D)} + N_{L}(D) + S_{L}(D) + T_{L}(c_{k}^{f,z,QF}) \qquad (11)$$

where:

- $\mathcal{S}(c_{k}^{f,z,QF})$ is the file size in bits of $c_{k}^{f,z,QF}$.
- $N_{B}(D)$ and $N_{L}(D)$ are the bitrate and latency of the network to which D is connected respectively.
- $S_{L}(D)$ is the server latency. For a device D, it represents the time spent by the request on the server (i.e. in the queue) waiting to be processed. This period of time depends on the performance of the server, but also on the number of requests waiting for the service. So, for a given server, this value may be different for each device.
- $T_{L}(c_{k}^{f,z,QF})$ is the transcoding latency. It represents how long the adaptation operation takes to complete, and 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.

There is no doubt that 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 perceived quality is reduced accordingly. That is, transport quality is inversely proportional to total delivery time. We therefore propose to evaluate transport quality using a normalization *Z-shaped built-in membership function* (Zmf) [31]. This was inspired by the work of [20,34], in which the authors used the *sigmf* and *gaussmf* membership functions to normalize various parameters, such as the network bandwidth and latency. This function, (Zmf), expresses the end-user's appreciation of (or frustration with) the adapted content, as a function of the wait time, in terms of a behavior. An example of such a behavior is depicted in Fig. 3. In fact, the appreciation or frustration varies from one individual to another, which is why the values of α and β (see Fig. 3) are used. These values can be determined

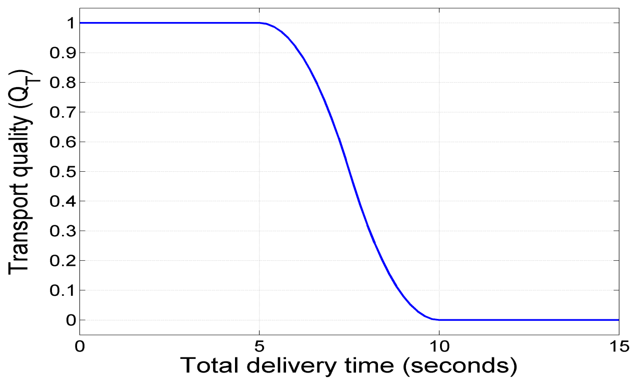


Fig. 3: Transport quality behavior for $\alpha = 5$ and $\beta = 10$

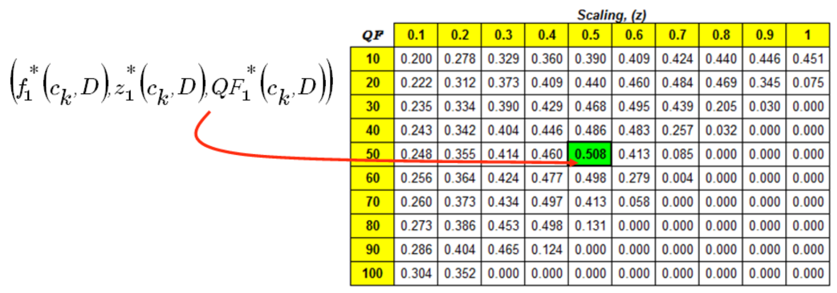
by experience or defined by the end-user. The value α expresses the period of time in which the end-user is fully satisfied with the response time. The value $(\alpha + \beta)/2$ expresses the period of time in which that appreciation is reduced to 50%. When the total delivery time reaches the value β, the user's appreciation falls to 0. According to research performed to estimate the wait time that users will tolerate when accessing Web content [23,28], the values α and β can be set to model the user's actual behavior regarding wait time. Thus, transport quality can be formulated as follows:

$$\begin{aligned} \mathcal{Q}_T(c_k^{f,z,QF}, D) &= \text{Zmf}(x, [\alpha, \beta]) \\ &= \begin{cases} 1 & \text{if} \quad x \leq \alpha \\ 1 - 2\left(\frac{x-\alpha}{\beta-\alpha}\right)^2 & \text{if} \quad \alpha \leq x \leq \frac{\alpha+\beta}{2} \\ 2\left(\frac{x-\beta}{\beta-\alpha}\right)^2 & \text{if} \quad \frac{\alpha+\beta}{2} \leq x \leq \beta \\ 0 & \text{if} \quad x \geq \beta \end{cases} \end{aligned} \tag{12}$$

where $x = T_d\big(c_k^{f,z,QF}, D\big)$.

5 Proposed prediction methods and models

In [18], we proposed a prediction-based dynamic content adaptation framework to solve equation (5), in which near-optimal transcoding parameters are computed. In this paper, we generalize the proposed framework by proposing a set of models and methods designed to improve the accuracy of these near-optimal transcoding parameters. Let us call the solution presented in [18], *Method 1 - Estimation.* There are some inaccuracies in these estimated transcoding parameters, but they nevertheless represent a good starting point from which the other proposed methods improve. These methods are, in fact, variants of method 1. They improve accuracy, but at the expense of increased complexity (number of transcoding operations).


$$\left(f_1^*\left(c_k, D\right), z_1^*\left(c_k, D\right), QF_1^*\left(c_k, D\right)\right)$$

	Scaling, (z)									
QF	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
10	0.200	0.278	0.329	0.360	0.390	0.409	0.424	0.440	0.446	0.451
20	0.222	0.312	0.373	0.409	0.440	0.460	0.484	0.469	0.345	0.075
30	0.235	0.334	0.390	0.429	0.468	0.495	0.439	0.205	0.030	0.000
40	0.243	0.342	0.404	0.446	0.486	0.483	0.257	0.032	0.000	0.000
50	0.248	0.355	0.414	0.460	0.508	0.413	0.085	0.000	0.000	0.000
60	0.256	0.364	0.424	0.477	0.498	0.279	0.004	0.000	0.000	0.000
70	0.260	0.373	0.434	0.497	0.413	0.058	0.000	0.000	0.000	0.000
80	0.273	0.386	0.453	0.498	0.131	0.000	0.000	0.000	0.000	0.000
90	0.286	0.404	0.465	0.124	0.000	0.000	0.000	0.000	0.000	0.000
100	0.304	0.352	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Fig. 4: Example of estimated transcoding parameters computed for a given document using a $\mathcal{Q}_E$ table. The shaded cell contains the estimated optimal $\mathcal{Q}_E$ obtained by method 1

5.1 Method 1 - Estimation

In [3,26,4], and also in [18], quantized values of z and QF are used instead of continuous ones, in order to limit the parameter space; that is, using a granularity of $\Delta z = 0.1$ and $\Delta QF = 10$, the quantized values of z and QF used are as follows:

$$\begin{aligned} \widetilde{z} &\in \{0.1, 0.2, 0.3, \ldots, 1\} \\ \widetilde{QF} &\in \{10, 20, 30, \ldots, 100\} \end{aligned} \tag{13}$$

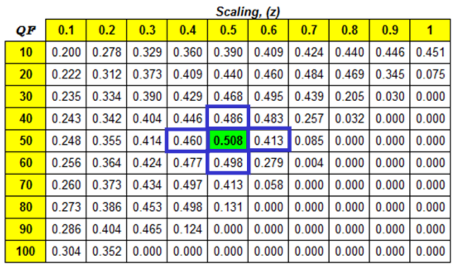
Thus, the solution space consists of 200 distinct combinations of parameters (100 combinations for each format: JPEG and XHTML). With this solution space, an exhaustive static method will perform 200 transcoding operations and select the best one.

With method 1, for a content c_k and a target mobile device D, $\mathcal{Q}_E(c_k^{f,z,QF}, D)$ can be estimated for $f \in \{\text{JPEG}, \text{XHTML}\}$ and the quantized values z and QF. By solving (5) in the estimated solution space, we can identify the near-optimal solution, which consists of the transcoding parameter combinations that maximize $\mathcal{Q}_E$. For instance, Fig. 4 shows the estimated $\mathcal{Q}_E$ values for f = JPEG and the various combinations of $\widetilde{z}$ and $\widetilde{QF}$ computed for a document. The shaded cell represents the estimated near-optimal $\mathcal{Q}_E$ obtained by method 1. This method is based on predicted file size and visual quality of transcoded JPEG images, tabulated in [26,3], and indexed by $\widetilde{z}$ and $\widetilde{QF}$ parameters. A full description of this method can be found in [18].

Let $\mathcal{R}_1^*(c_k, D)$ be the subset of near-optimal transcoding parameter combinations obtained by this method. It is a modified formulation of equation (5). We added the index 1 to clearly indicate that the near-optimal transcoding parameters were obtained using method 1. So, we have:

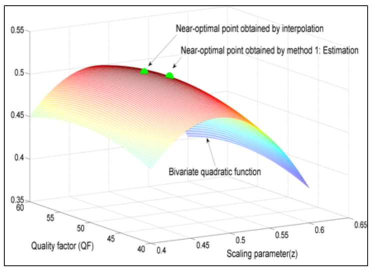
$$\begin{aligned} \mathcal{R}_1^*(c_k, D) &= \left\{ \left(f_1^*(c_k, D), z_1^*(c_k, D), QF_1^*(c_k, D)\right) \right\} \\ &= \underset{(f,z,QF) \in \mathcal{R}(c_k, D)}{\arg\max} \; \mathcal{Q}_E(c_k^{f,z,QF}, D) \end{aligned} \tag{14}$$

In terms of complexity, as presented in [18], method 1 requires a single transcoding operation.



					Scaling, (z)					
QF	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
10	0.200	0.278	0.329	0.360	0.390	0.409	0.424	0.440	0.446	0.451
20	0.222	0.312	0.373	0.409	0.440	0.460	0.484	0.469	0.345	0.075
30	0.235	0.334	0.390	0.429	0.468	0.495	0.439	0.205	0.030	0.000
40	0.243	0.342	0.404	0.446	0.486	0.483	0.257	0.032	0.000	0.000
50	0.248	0.355	0.414	0.460	**0.508**	0.413	0.085	0.000	0.000	0.000
60	0.256	0.364	0.424	0.477	0.498	0.279	0.004	0.000	0.000	0.000
70	0.260	0.373	0.434	0.497	0.413	0.058	0.000	0.000	0.000	0.000
80	0.273	0.386	0.453	0.498	0.131	0.000	0.000	0.000	0.000	0.000
90	0.286	0.404	0.465	0.124	0.000	0.000	0.000	0.000	0.000	0.000
100	0.304	0.352	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(a)



(b)

Fig. 5: (a) Example of the estimated transcoding parameters computed for a given document. The shaded cell contains the estimated optimal $\mathcal{Q}_E$ obtained by method 1. (b) The bivariate quadratic function used to model the region covered by this optimal point and its four nearest neighbors

5.2 Method 2 - Estimation and interpolation

In this method, instead of using quantized values of z and QF, we let the solution space be continuous. Since the estimated solution obtained by method 1 is near-optimal, the optimal solution should be in the same neighborhood. Consequently, using the estimated near-optimal solution and its four nearest neighbors, we suppose that the optimal solution is within the region covered by these five points. We fix the near-optimal format $f_1^*(c_k, D)$ (which is known from method 1) and model $\mathcal{Q}_E$ in this region using a bivariate quadratic function defined as follows:

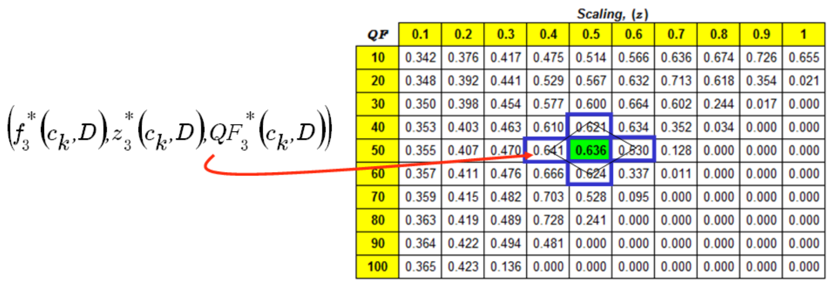
$$f(x, y) = ax^2 + bx + cy^2 + dy + e \tag{15}$$

where x and y represent z and QF in a continuous space respectively.

The optimal point in this region is where the gradient is null:

$$\begin{cases} \dfrac{\partial f}{\partial x} & = 2ax + b = 0 \\ \dfrac{\partial f}{\partial y} & = 2cy + d = 0 \end{cases} \tag{16}$$

Using the estimated near-optimal point and its four estimated nearest neighbors, we compute the coefficients a, b, c, d, and e. Then, using (16), we compute the estimated interpolated near-optimal transcoding parameters $z_I^*(c_k, D)$ and $QF_I^*(c_k, D)$. We expect the interpolated near-optimal point to be close to the actual optimal. Two adapted contents are created using the estimated and interpolated transcoding parameters, and the best of the two is selected as the near-optimal adapted content obtained by method 2. Fig. 5 shows, for a given document, the estimated near-optimal $\mathcal{Q}_E$ obtained by method 1 and its four nearest neighbors, and the bivariate quadratic function obtained by modeling this region. On the surface of this function, we have two points: the one obtained by method 1, and the one obtained by interpolation (where the gradient is null).


$\left(f_3^*(c_k,D), z_3^*(c_k,D), QF_3^*(c_k,D)\right)$

	Scaling, (z)									
QF	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
10	0.342	0.376	0.417	0.475	0.514	0.566	0.636	0.674	0.726	0.655
20	0.348	0.392	0.441	0.529	0.567	0.632	0.713	0.618	0.354	0.021
30	0.350	0.398	0.454	0.577	0.600	0.664	0.602	0.244	0.017	0.000
40	0.353	0.403	0.463	0.610	0.621	0.634	0.352	0.034	0.000	0.000
50	0.355	0.407	0.470	0.641	0.636	0.630	0.128	0.000	0.000	0.000
60	0.357	0.411	0.476	0.666	0.624	0.337	0.011	0.000	0.000	0.000
70	0.359	0.415	0.482	0.703	0.528	0.095	0.000	0.000	0.000	0.000
80	0.363	0.419	0.489	0.728	0.241	0.000	0.000	0.000	0.000	0.000
90	0.364	0.422	0.494	0.481	0.000	0.000	0.000	0.000	0.000	0.000
100	0.365	0.423	0.136	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Fig. 6: Example of estimated transcoding parameters and their computed $\mathcal{Q}_E$ for a given document. The shaded cell contains the near-optimal computed $\mathcal{Q}_E$ obtained by method 1. The diamond shows its four computed nearest neighbors

Formally, the near-optimal transcoding parameters obtained by method 2 are given by:

$$\begin{aligned}\mathcal{R}_2^*(c_k, D) &= \left\{\left(f_2^*(c_k, D), z_2^*(c_k, D), QF_2^*(c_k, D)\right)\right\} \\ &= \underset{(f,z,QF)\in\mathcal{N}_e^2}{\arg\max} \; \mathcal{Q}_E(c_k^{f,z,QF}, D)\end{aligned} \tag{17}$$

where $\mathcal{N}_e^2$ is a set comprising two elements: the estimated and the interpolated (the index of which is denoted I below) near-optimal parameter combinations. Formally, we have:

$$f_2^*(c_k, D) = f_1^*(c_k, D) \tag{18}$$

$$\begin{aligned}\mathcal{N}_e^2 = \Big\{&\left(f_1^*(c_k, D), z_1^*(c_k, D), QF_1^*(c_k, D)\right), \\ &\left(f_1^*(c_k, D), z_I^*(c_k, D), QF_I^*(c_k, D)\right)\Big\}\end{aligned} \tag{19}$$

In terms of complexity, this method requires two transcoding operations, one from method 1 and the other from the interpolation.

5.3 Method 3 - Estimation and one-step Diamond search

In this method, we use the estimated near-optimal adapted content (computed from method 1) and its four nearest neighbors. Unlike method 2, here, these five points are transcoded versions, rather than merely estimated content, and the best of them is selected. For instance, Fig. 6 shows the computed $\mathcal{Q}_E$ array for the same document presented in the previous methods. The shaded cell contains the computed $\mathcal{Q}_E$ obtained using the estimated near-optimal transcoding parameters of method 1, and diamond search points formed by its four nearest neighbors. In this example, the left neighbor represents the near-optimal point obtained by method 3.

Formally, the optimal transcoding parameters obtained by this third method are as follows:

$$\begin{aligned}\mathcal{R}_3^*(c_k, D) &= \Big\{\big(f_3^*(c_k, D), z_3^*(c_k, D), QF_3^*(c_k, D)\big)\Big\}\\ &= \underset{(f,z,QF)\in\mathcal{N}_e^5}{\arg\max}\ \mathcal{Q}_E(c_k^{f,z,QF}, D)\end{aligned} \tag{20}$$

where $\mathcal{N}_e^5$ is a set containing five elements: the estimated near-optimal parameters of method 1 and their four nearest neighbors. So, we have:

$$f_3^*(c_k, D) = f_1^*(c_k, D) \tag{21}$$

$$\begin{aligned}\mathcal{N}_e^5 = \Big\{&\big(f_1^*(c_k, D), z_1^*(c_k, D), QF_1^*(c_k, D)\big),\\ &\big(f_1^*(c_k, D), z_1^*(c_k, D) \pm \Delta z, QF_1^*(c_k, D)\big),\\ &\big(f_1^*(c_k, D), z_1^*(c_k, D), QF_1^*(c_k, D) \pm \Delta QF\big)\Big\}\end{aligned} \tag{22}$$

Regarding complexity, this method requires five transcoding operations. One operation is from method 1 and the others are from its four nearest neighbors.

5.4 Method 4 - Estimation and two-steps diamond search

Like method 3, in this method, we identify and create the estimated optimal adapted content and its four nearest neighbors. From these five points, we identify the best one (equal to that obtained by method 3) and use it as a starting point to explore its four nearest neighbors. This is the origin of the term *two-steps diamond search*. If the best point is equal to that obtained by method 1, there is no need to perform the second step in the search, and so the near-optimal point returned by this method is the one obtained by method 3 $\big(f_3^*(c_k, D), z_3^*(c_k, D), QF_3^*(c_k, D)\big)$. Otherwise, we identify and create its four nearest neighbors, one of which had already been created by method 3. The best of these points becomes the near-optimal point computed by method 4. Fig. 7 shows the same example presented in method 3, in which the two diamond search points are outlined. In this example, the neighbor below the optimal solution obtained by method 3 represents the optimal point of this method ($z = 0.4$ and $QF = 60$).

This can be formulated as follows:

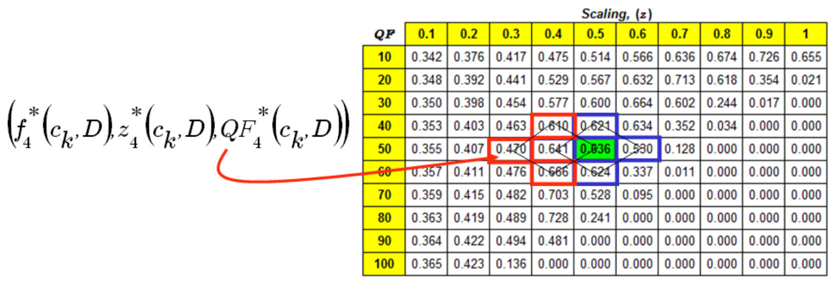
$$\begin{aligned}\mathcal{R}_4^*(c_k, D) &= \Big\{\big(f_4^*(c_k, D), z_4^*(c_k, D), QF_4^*(c_k, D)\big)\Big\}\\ &= \underset{(f,z,QF)\in\mathcal{N}_e^{5,8}}{\arg\max}\ \mathcal{Q}_E(c_k^{f,z,QF}, D)\end{aligned} \tag{23}$$

where $\mathcal{N}_e^{5,8}$ is the set of transcoding parameter combinations used in this method. Thus, we have:

$$f_4^*(c_k, D) = f_1^*(c_k, D) \tag{24}$$

$$\begin{aligned}\mathcal{N}_e^{5,8} = \mathcal{N}_e^5 \cup \Big\{&\big(f_3^*(c_k, D), z_3^*(c_k, D) \pm \Delta z, QF_3^*(c_k, D)\big),\\ &\big(f_3^*(c_k, D), z_3^*(c_k, D), QF_3^*(c_k, D) \pm \Delta QF\big)\Big\}\end{aligned} \tag{25}$$

The complexity of this method is either five or eight transcoding operations:


$\left(f_4^*(c_k, D), z_4^*(c_k, D), QF_4^*(c_k, D)\right)$
Scaling, (z)

QF	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
10	0.342	0.376	0.417	0.475	0.514	0.566	0.636	0.674	0.726	0.655
20	0.348	0.392	0.441	0.529	0.567	0.632	0.713	0.618	0.354	0.021
30	0.350	0.398	0.454	0.577	0.600	0.664	0.602	0.244	0.017	0.000
40	0.353	0.403	0.463	0.610	0.621	0.634	0.352	0.034	0.000	0.000
50	0.355	0.407	0.470	0.641	0.636	0.530	0.128	0.000	0.000	0.000
60	0.357	0.411	0.476	0.666	0.624	0.337	0.011	0.000	0.000	0.000
70	0.359	0.415	0.482	0.703	0.528	0.095	0.000	0.000	0.000	0.000
80	0.363	0.419	0.489	0.728	0.241	0.000	0.000	0.000	0.000	0.000
90	0.364	0.422	0.494	0.481	0.000	0.000	0.000	0.000	0.000	0.000
100	0.365	0.423	0.136	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Fig. 7: Example of estimated transcoding parameters and their computed $\mathcal{Q}_E$ for a given document. The shaded cell contains the near-optimal computed $\mathcal{Q}_E$ obtained by method 1. The two diamonds show its evaluated neighbors

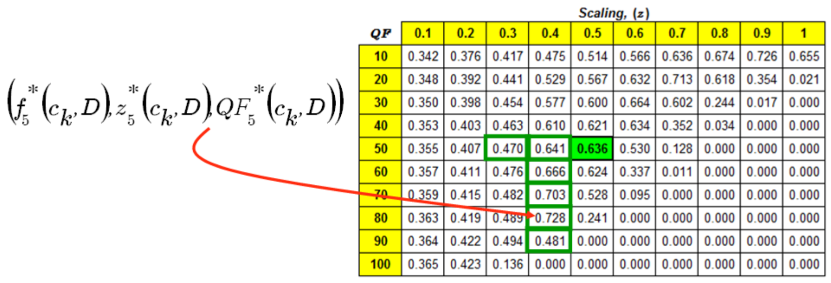
- If $\left(f_3^*(c_k, D), z_3^*(c_k, D), QF_3^*(c_k, D)\right) = \left(f_1^*(c_k, D), z_1^*(c_k, D), QF_1^*(c_k, D)\right)$, then we have $\mathcal{N}_e^{5,8} = \mathcal{N}_e^5$, and so only five transcoding operations are required. One operation from method 1 and the others from its four nearest neighbors.
- Otherwise, we have $\mathcal{N}_e^{5,8} \supset \mathcal{N}_e^5$, and so eight transcoding operations are performed. One operation from method 1, four from its nearest neighbors, and three from the neighbors of the optimal obtained by method 3 if it is different from that obtained by method 1.

5.5 Method 5 - Estimation and greedy search

In this method, we use the estimated near-optimal point (from method 1) as a starting point and explore its neighborhood, seeking to improve the $\mathcal{Q}_E$ obtained until convergence is reached; that is, until this $\mathcal{Q}_E$ cannot be improved any further. We tested various patterns and found that, for the problem at hand, following one of these patterns: LRUD, LRDU, RLUD, RLDU, UDLR, UDRL, DULR, or DURL, improved this $\mathcal{Q}_E$ significantly with the fewest transcodings, compared to other patterns. Note that, using these selected patterns, the performance of this method (optimal $\mathcal{Q}_E$ versus complexity) varies slightly from one pattern to another. But the difference is so small that it can be neglected, and so these patterns can be used interchangeably. Before detailing this method, we explain what the letters L, R, U, and D stand for. For a given point, they constitute its nearest left-hand, right-hand, upward, and downward neighbors respectively. For instance, the four nearest neighbors of the estimated point of method 1 are given by:

$$\begin{aligned}
\text{Left neighbor} &: \left(f_1^*(c_k, D), z_1^*(c_k, D) - \Delta z, QF_1^*(c_k, D)\right) \\
\text{Right neighbor} &: \left(f_1^*(c_k, D), z_1^*(c_k, D) + \Delta z, QF_1^*(c_k, D)\right) \\
\text{Up neighbor} &: \left(f_1^*(c_k, D), z_1^*(c_k, D), QF_1^*(c_k, D) - \Delta QF\right) \\
\text{Down neighbor} &: \left(f_1^*(c_k, D), z_1^*(c_k, D), QF_1^*(c_k, D) + \Delta QF\right)
\end{aligned} \tag{26}$$

Using the LRDU pattern, for example, this method proceeds as follows: We start from the point $\left(f_1^*(c_k, D), z_1^*(c_k, D), QF_1^*(c_k, D)\right)$, then, we verify whether or not the neighbor to the left provides a better solution. If it does, we move towards the left until there is no further improvement. Otherwise, we verify whether or not the


$\left(f_5^*(c_k,D), z_5^*(c_k,D), QF_5^*(c_k,D)\right)$

	Scaling, (z)									
QF	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
10	0.342	0.376	0.417	0.475	0.514	0.566	0.636	0.674	0.726	0.655
20	0.348	0.392	0.441	0.529	0.567	0.632	0.713	0.618	0.354	0.021
30	0.350	0.398	0.454	0.577	0.600	0.664	0.602	0.244	0.017	0.000
40	0.353	0.403	0.463	0.610	0.621	0.634	0.352	0.034	0.000	0.000
50	0.355	0.407	0.470	0.641	0.636	0.530	0.128	0.000	0.000	0.000
60	0.357	0.411	0.476	0.666	0.624	0.337	0.011	0.000	0.000	0.000
70	0.359	0.415	0.482	0.703	0.528	0.095	0.000	0.000	0.000	0.000
80	0.363	0.419	0.489	0.728	0.241	0.000	0.000	0.000	0.000	0.000
90	0.364	0.422	0.494	0.481	0.000	0.000	0.000	0.000	0.000	0.000
100	0.365	0.423	0.136	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Fig. 8: Example of estimated transcoding parameters and their computed $\mathcal{Q}_E$ for a given document. The shaded cell contains the near-optimal computed $\mathcal{Q}_E$ obtained by method 1. The points evaluated by this method are outlined

neighbor to the right provides a better solution, and, if so, we move towards the right until there is no further improvement. The same process is then performed in the downward and upward directions. Each time a new point is evaluated, a new transcoding is performed. For instance, Fig. 8 shows, for the same example, the set of points visited when an LRDU greedy search is performed on the computed $\mathcal{Q}_E$ array. The optimal point in this example corresponds to $z = 0.4$ and $QF = 80$. The pseudo-code of the proposed greedy search algorithm (LRDU pattern) is presented in Algorithm 1. Formally, the near-optimal transcoding parameters combinations obtained by this fifth method are given by:

$$\begin{aligned}\mathcal{R}_5^*(c_k, D) &= \left\{\left(f_5^*(c_k, D), z_5^*(c_k, D), QF_5^*(c_k, D)\right)\right\} \\ &= \underset{(f,z,QF)\in\mathcal{N}_e^{greedy}}{\arg\max} \mathcal{Q}_E(c_k^{f,z,QF}, D)\end{aligned} \tag{27}$$

where $\mathcal{N}_e^{greedy}$ is the set of points evaluated in this method, which can vary greatly, depending on c_k and D. Similarly, $f_5^*(c_k, D) = f_1^*(c_k, D)$.

The number of transcoding operations can be very high if either the starting point is chosen randomly or an exhaustive search is performed. However, in this method, we take advantage of the prediction of the estimated transcoding parameters, which are very reliable. Unlike an exhaustive search, we start here from an estimated point that is relatively close to the optimal. Indeed, experimental results (see section 7) show that the number of transcoding operations is between 4 and 7 - 5.2 on average, which means that we are still in the same range of transcoding operations as with previous methods.

6 Experimental setup

To validate the performance of the proposed set of methods, and particularly the $\mathcal{Q}_E$ improvements made by methods 2 to 5 over method 1, we used the same experimental setup as presented in [17,18]. That is, a corpus that contains 120 OpenOffice Impress presentation documents was created. To this end, we created a Java-based application that uses OpenOffice APIs (known under the name of UNO) to create these Impress documents [25]. For simplicity, each document was

```
function LRDU_Search($c_k, D$)
begin
    $z \leftarrow z_1^*(c_k, D)$, $QF \leftarrow QF_1^*(c_k, D)$
    $\mathcal{Q}_E \leftarrow \mathcal{Q}_E(c_k^{z,QF}, D))$
    $(z, QF, \mathcal{Q}_E) \leftarrow$ LR_Search($c_k, D, z, QF, \mathcal{Q}_E$)
    $(z, QF, \mathcal{Q}_E) \leftarrow$ DU_Search($c_k, D, z, QF, \mathcal{Q}_E$)
    return $(z, QF, \mathcal{Q}_E)$
end

function LR_Search($c_k, D, z, QF, \mathcal{Q}_E$)
begin
    if $\mathcal{Q}_E(c_k^{z-\Delta z,QF}, D)) > \mathcal{Q}_E$ then
        $(z, QF, \mathcal{Q}_E) \leftarrow$ search($c_k, D, z, QF, \mathcal{Q}_E, -1, 0$)
    else
        if $\mathcal{Q}_E(c_k^{z+\Delta z,QF}, D)) > \mathcal{Q}_E$ then
            $(z, QF, \mathcal{Q}_E) \leftarrow$ search($c_k, D, z, QF, \mathcal{Q}_E, +1, 0$)
        end
    end
    return $(z, QF, \mathcal{Q}_E)$
end

function DU_Search($c_k, D, z, QF, \mathcal{Q}_E$)
begin
    if $\mathcal{Q}_E(c_k^{z,QF+\Delta QF}, D)) > \mathcal{Q}_E$ then
        $(z, QF, \mathcal{Q}_E) \leftarrow$ search($c_k, D, z, QF, \mathcal{Q}_E, 0, +1$)
    else
        if $\mathcal{Q}_E(c_k^{z,QF-\Delta QF}, D)) > \mathcal{Q}_E$ then
            $(z, QF, \mathcal{Q}_E) \leftarrow$ search($c_k, D, z, QF, \mathcal{Q}_E, 0, -1$)
        end
    end
    return $(z, QF, \mathcal{Q}_E)$
end

function search($c_k, D, z_o, QF_o, \mathcal{Q}_E, \lambda_z, \lambda_{QF}$)
begin
    $z \leftarrow z_o + \lambda_z \Delta z$, $QF \leftarrow QF_o + \lambda_{QF} \Delta QF$
    while $\mathcal{Q}_E(c_k^{z,QF}, D)) > \mathcal{Q}_E$ do
        $\mathcal{Q}_E \leftarrow \mathcal{Q}_E(c_k^{z,QF}, D)$
        $z \leftarrow z + \lambda_z \Delta z$, $QF \leftarrow QF + \lambda_{QF} \Delta QF$
    end
    return $(z, QF, \mathcal{Q}_E)$
end
```

Algorithm 1: LRDU greedy search pseudo-code

composed of one slide, with these slides themselves composed of one text-box and one image, and their positions set randomly. To span a wide variety of slide characteristics, quantized values, representing the percentage of areas occupied by images (I) and text-boxes (T), were used as follows:

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

Note that it is allowed for a slide to be comprised of text and images sharing the same area or the same part of the area (partial or total overlap). No background was set, but an inserted image covering the whole slide could be considered too. Text and images were collected from different websites such as [32].

Let $\mathcal{V}$ be this set of Impress documents. Let D be a target mobile device that 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 [30]. Note that, similar results were obtained with other mobile device characteristics and network conditions.

We want to compare our framework with a set of optimally adapted contents, using an exhaustive static method, to be used as references (ideal targets to attain). These optimally adapted contents would be used to evaluate the performance of the proposed set of methods. To that end, using the OpenOffice JPEG- and XHTML-based filters, each document c_k from $\mathcal{V}$, was converted into a set of Web pages by varying the transcoding parameters values $\widetilde{z} \in \{0.1, 0.2, 0.3, \ldots 1\}$ and $\widetilde{QF} \in \{10, 20, 30, \ldots 100\}$. As explained in [18], the OpenOffice XHML-based filter was very limited and contains numerous bugs that we fixed.

To compute the $\mathcal{Q}_E$ of these adapted contents, $\mathcal{Q}_V$ and $\mathcal{Q}_T$ were evaluated first. The adapted content visual quality, $\mathcal{Q}_V$, was computed using SSIM [33]. To evaluate the quality of a transcoded image, SSIM requires both the original image and its transcoded version. In our case, we have slides and not images. In the case of adapted content generated by the XHTML-based filter, we still have the original images, which were used in the SSIM evaluation. In the case of the JPEG-based filter, each slide was converted into a JPEG image using $z = 1$ and $QF = 80$, from which a set of JPEG images were created using the various values of $\widetilde{z}$ and $\widetilde{QF}$. Then, these images were wrapped into Web page skeletons. This way, the images created using $z = 1$ and $QF = 80$ were used as original images in the process of visual quality and file size estimation, as detailed in [18]. We used $z = 1$ and $QF = 80$ in the creation of the original image as this combination preserves the image visual quality and ensures lower file size [3]. Moreover, to use SSIM, we needed to provide a resolution at which the two images (original and transcoded) should be scaled for comparison. Since the slides are to be rendered on D and their default resolution, as rendered on PC by OpenOffice, was 1058×794, the resolution to be used by SSIM can be determined, following the methodology of [3], by: $\min\left(\frac{640}{1058}, \frac{360}{794}\right) \approx 45\%$. This suggests a comparison of images at a resolution of 40% of the original image resolution ($z_v = 0.4$).

Note that, the SSIM index exhibits a highly non-linear relationship with the MOS (Mean Opinion Score), which represents a true measure of the human perception of image quality [29]. Therefore, to address the third element regarding the $\mathcal{Q}_E$ design [14] (see section 4), the computed SSIM values were not used directly, but rather, were converted into their corresponding continuous MOS values.

To evaluate the $\mathcal{Q}_T$ and $\mathcal{Q}_E$ of these adapted contents. Based on researches conducted to estimate waiting time that users tolerate when accessing Web content [28, 23], the actual behavior of $\mathcal{Q}_T$ can be determined by $\alpha = 5s$ and $\beta = 10s$ (see Fig. 3). In addition, to facilitate the validation, the values of $S_L(D)$ and $T_L(c_k^{f,z,QF})$ (see equation (11)) were set to zero.

Besides the exhaustive static system, we want to compare our framework with a typical dynamic system, denoted *fixed-QF*, which performs a single transcoding operation using a quality factor $QF = 80$ (which provides good visual quality)

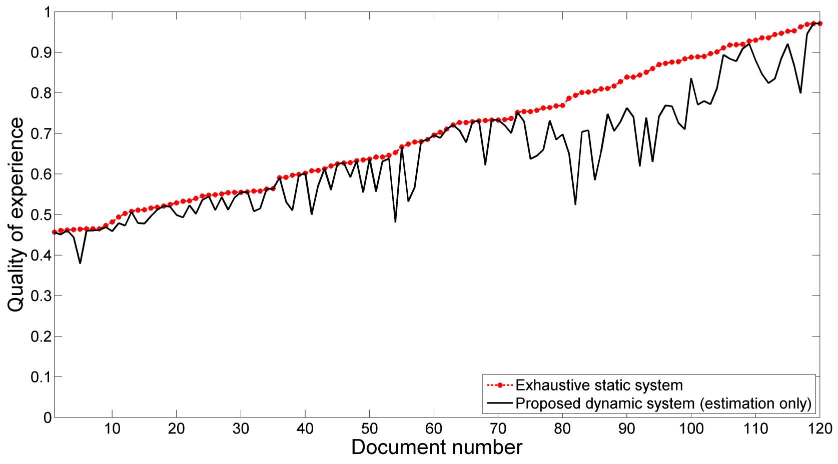


Fig. 9: Optimal $\mathcal{Q}_E$ obtained by the proposed dynamic framework vs. that of the exhaustive static system when f = JPEG. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

and a scaling parameter $z = 0.4$ (which corresponds to the target mobile device resolution).

Regarding our framework, using method 1 [18], the estimated transcoding parameters were very reliable overall. However, in the case of JPEG, there were some outlier combinations. That is, for certain documents, the $\mathcal{Q}_E$ of the estimated near-optimal transcoding parameters was not as close to optimality as the rest of the documents. To visualize this, the $\mathcal{Q}_E$ obtained by the proposed dynamic framework and the exhaustive static system were sorted according to the $\mathcal{Q}_E$ obtained by the latter and plotted (see Fig. 9). We tested the framework with various bitrate values, and reached the conclusion that these outlier points vary with the communication network conditions (e.g. bitrate). After analyzing the curve behavior of $\mathcal{Q}_E$, we found that these outliers were caused by the nature of the Zmf curve (see Fig. 3) used to model the $\mathcal{Q}_T$, which was set using $\alpha = 5$ and $\beta = 10$. Indeed, the Zmf curve for this scenario decreases aggressively between the two values α and β, which makes the $\mathcal{Q}_E$ curve highly sensitive to delivery time (file size and network conditions). Therefore, in method 1, since the file size prediction error in [26] can reach 15%, we have increased the predicted file size by this amount to ensure that the transcoded file size will not lead to a drastically lower $\mathcal{Q}_T$ than predicted. We sacrifice the quality slightly to ensure a good $\mathcal{Q}_T$, as it is much more sensitive to file size. In this scenario, it was not necessary to increase the estimated file size in the case of XHTML, as the estimated XHTML parameters were very precise [18]. However, for lower bitrate values, we observed the same behavior as that observed in the JPEG case, which can be corrected by adjusting the predicted file size (i.e. adding a safety factor). Some experiments with lower bitrate values exhibiting this phenomenon are presented later in section 7.4.

Based on the estimated near-optimal transcoding parameters obtained by method 1, the near-optimal $\mathcal{Q}_E$ obtained by method 2 to method 5 were computed for each slide by solving equations (17), (20), (23), and (27) respectively.

As a result, for each slide c_k, its computed (using the exhaustive static and fixed-QF dynamic systems) and estimated (using methods 1 to 5) best transcoding parameters and their $\mathcal{Q}_{E}$ were stored in arrays, as follows:

$$\begin{aligned} W^*_{E,k} &= \Big[c_k, f^*_{E}(c_k,D), z^*_{E}(c_k,D), QF^*_{E}(c_k,D), \mathcal{Q}_{E}(c_k^{f^*_{E}(c_k,D), z^*_{E}(c_k,D), QF^*_{E}(c_k,D)}, D)\Big] \\ W^*_{FQF,k} &= \Big[c_k, f^*_{FQF}(c_k,D), 0.4, 80, \mathcal{Q}_{E}(c_k^{f^*_{FQF}, 0.4, 80}, D)\Big] \\ W^*_{i,k} &= \Big[c_k, f^*_i(c_k,D), z^*_i(c_k,D), QF^*_i(c_k,D), \mathcal{Q}_{E}(c_k^{f^*_i(c_k,D), z^*_i(c_k,D), QF^*_i(c_k,D)}, D)\Big] \end{aligned} \tag{28}$$

where:

- $W^*_{E,k}$ is an array that contains the optimal transcoding parameters $(f^*_{E}(c_k,D), z^*_{E}(c_k,D), QF^*_{E}(c_k,D))$ that were computed by the exhaustive static system, and their corresponding $\mathcal{Q}_{E}$.
- $W^*_{FQF,k}$ is an array containing the best transcoding parameters $(f^*_{FQF}(c_k,D), 0.4, 80)$, as computed by the fixed-QF dynamic system, and their $\mathcal{Q}_{E}$.
- $W^*_{i,k}$ is an array that contains the near-optimal transcoding parameters $(f^*_i(c_k,D), z^*_i(c_k,D), QF^*_i(c_k,D))$ attained by method $i \in \{1,2,3,4,5\}$, and its $\mathcal{Q}_{E}$.

Lastly, for each of the five proposed methods, the near-optimal transcoding parameters obtained and their corresponding $\mathcal{Q}_{E}$ are compared to that of the exhaustive static system (optimality) and fixed-QF dynamic system (typical dynamic system), the results of which are presented in the next section.

7 Experimental results and discussion

7.1 Optimal $\mathcal{Q}_{E}$ attained by each method

For each slide c_k, the near-optimal $\mathcal{Q}_{E}$ obtained by methods 1 to 5, as well as those computed by the exhaustive static method (from $W^*_{E,k}$) and the fixed-QF dynamic method (from $W^*_{FQF,k}$), were plotted. To visualize this, all the $\mathcal{Q}_{E}$ obtained were sorted according to those of the exhaustive static system, and presented in Figs. 10 and 11 for JPEG and XHTML respectively. Overall, the proposed methods have a $\mathcal{Q}_{E}$ close to that of the exhaustive static system. However, the $\mathcal{Q}_{E}$ obtained by the fixed-QF dynamic system is highly variable, and this is very obvious for lower $\mathcal{Q}_{E}$ values. The fixed-QF dynamic system is especially problematic for large documents and low network bitrate values. Note that the outlier points shown in Fig. 9 were corrected in method 1, and, of course, in methods 2 to 5.

7.2 $\mathcal{Q}_{E}$ improvement made by method 2 to method 5 over method 1

To show the improvements obtained by methods 2 to 5 over method 1, their relative gains in $\mathcal{Q}_{E}$ were computed. For instance, given a slide c_k, the relative

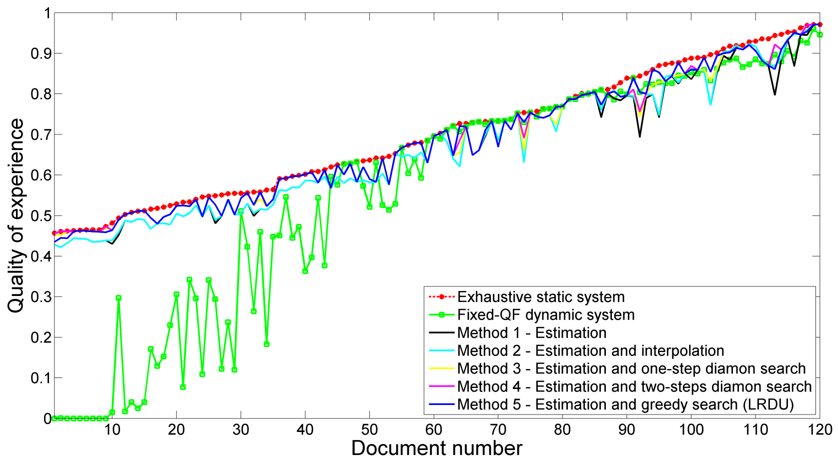


Fig. 10: Optimal $\mathcal{Q}_E$ obtained by methods 1 to 5 vs. that of the exhaustive static and fixed-QF dynamic systems when $f_1^*(c_k, D)$ = JPEG. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

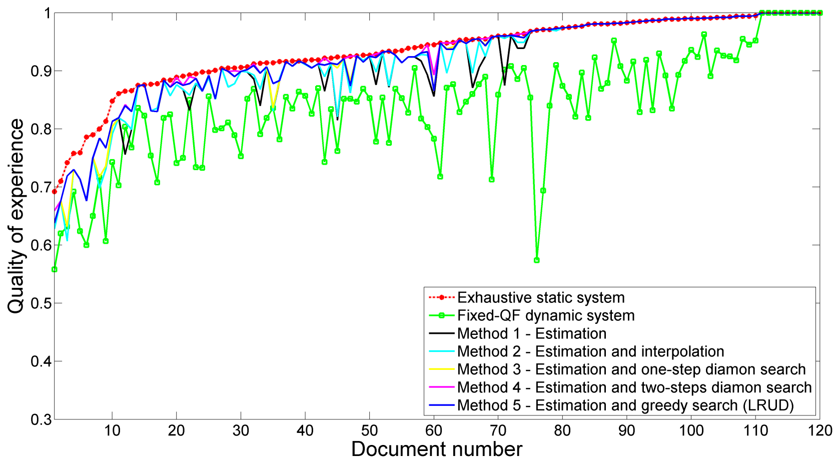


Fig. 11: $\mathcal{Q}_E$ obtained by methods 1 to 5 vs. that of the exhaustive static and fixed-QF dynamic systems when $f_1^*(c_k, D)$ = XHTML. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

gain obtained using method i is computed as follows:

$$\frac{\mathcal{Q}_E(c_k^{f_i^*(c_k,D),z_i^*(c_k,D),QF_i^*(c_k,D)}, D) - \mathcal{Q}_E(c_k^{f_1^*(c_k,D),z_1^*(c_k,D),QF_1^*(c_k,D)}, D)}{\mathcal{Q}_E(c_k^{f_1^*(c_k,D),z_1^*(c_k,D),QF_1^*(c_k,D)}, D)} \times 100\% \tag{29}$$

These computed $\mathcal{Q}_E$ relative gains were plotted as scattered points, as depicted in Figs. 12 and 13 for JPEG and XHTML respectively. For instance, for JPEG, sub-figures 12(a), 12(b), 12(c), and 12(d) show the $\mathcal{Q}_E$ relative gain obtained by

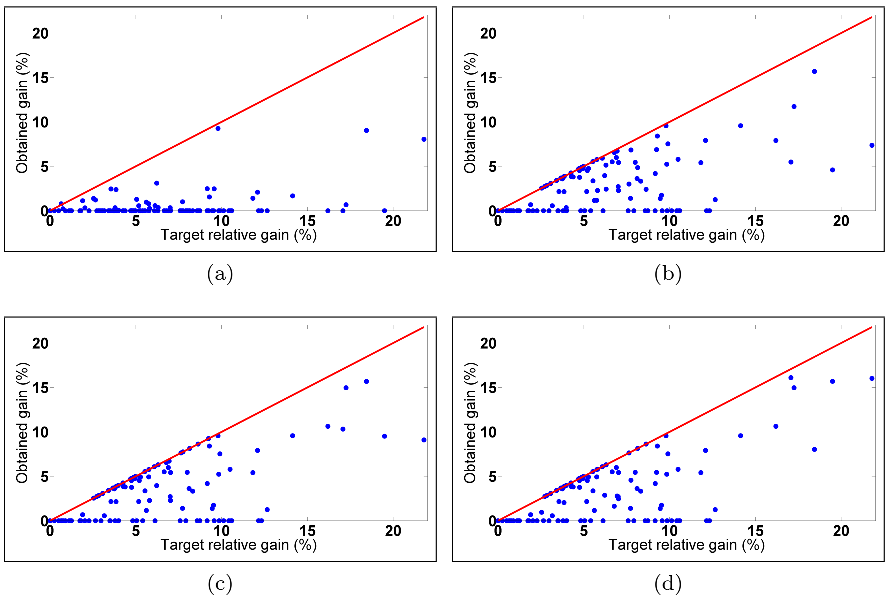


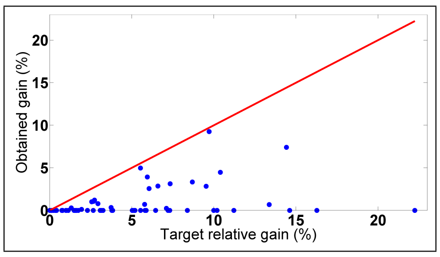
Fig. 12: $\mathcal{Q}_E$ relative gains for methods 2 to 5 with respect to method 1, when $f_1^*(c_k, D)$ = JPEG. (a) Method 2 - Estimation and interpolation, (b) Method 3 - Estimation and one-step diamond search, (c) Method 4 - Estimation and two-steps diamond search, (d) Method 5 - Estimation and greedy search

methods 2, 3, 4, and 5 respectively. In the case of XHTML, the relative gain obtained by methods 2 to 5 are presented in sub-figures 13(a), 13(b), 13(c), and 13(d) respectively. The diagonal line represents the target relative gains, which were computed from $W^*_{E,k}$. The scattered points represent the different slides, and their positions indicate the relative gain obtained versus the target relative gain.

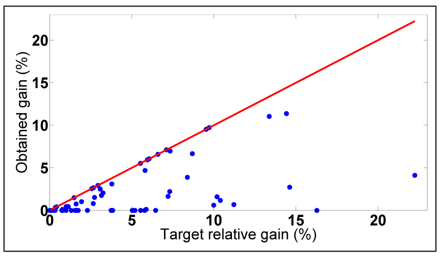
7.3 Number of documents with an improved $\mathcal{Q}_E$

Another view, showing the number of documents with an improved computed $\mathcal{Q}_E$ as a result of applying the proposed methods is depicted in Figs. 14 and 15 for JPEG and XHTML respectively. To show this aspect graphically, the $\mathcal{Q}_E$ range has been split into 10 bins ([0,0.1],]0.1,02],. . . ,]0.9,1]), and the documents that are in the same $\mathcal{Q}_E$ bin were counted and their numbers plotted as a histogram.

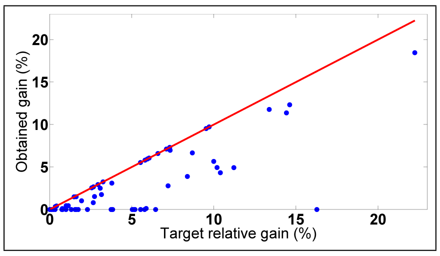
In the case of JPEG, using the fixed-QF dynamic system, the first four bins (for poor quality documents) contain almost 30% of the documents, while these bins are empty for the other methods (methods 1 to 5, in addition to the exhaustive static one). Also, unlike the other methods, for the fixed-QF dynamic system, the last bin (for the best quality documents) contains very few documents. This can also be seen in Fig. 10 for very low or very high $\mathcal{Q}_E$ values.



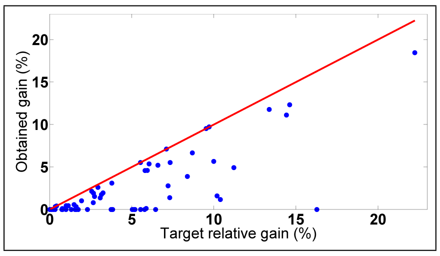
(a)



(b)



(c)



(d)

Fig. 13: $\mathcal{Q}_E$ relative gains for methods 2 to 5 with respect to method 1, when $f_1^*(c_k, D)$ = XHTML. (a) Method 2 - Estimation and interpolation, (b) Method 3 - Estimation and one-step diamond search, (c) Method 4 - Estimation and two-steps diamond search, (d) Method 5 - Estimation and greedy search

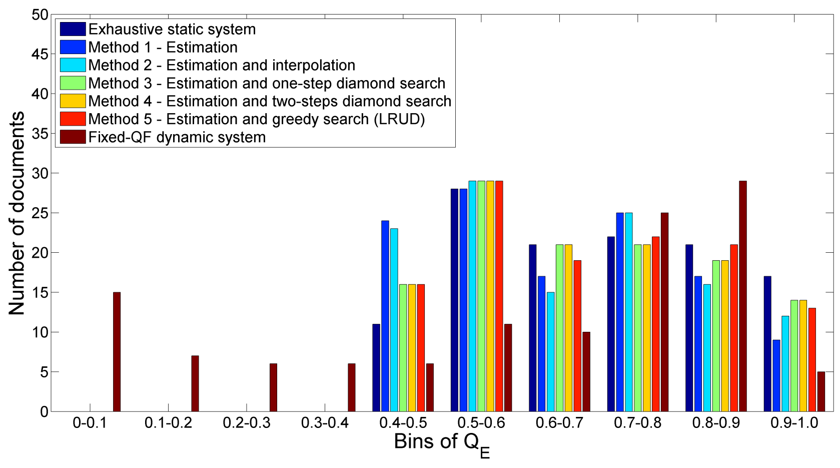


Fig. 14: Performance of the proposed methods by $\mathcal{Q}_E$ slices of 10% when $f_1^*(c_k, D)$ = JPEG

In the case of XHTML, the first 6 bins are empty for all the methods, except for the fixed-QF dynamic system, for which there are some documents in bin]0.5-06]. By contrast, the number of documents corresponding to the fixed-QF

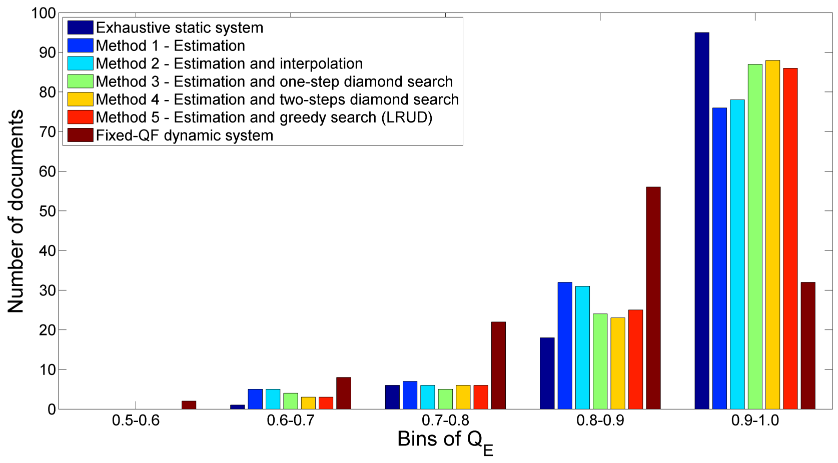


Fig. 15: Performance of the proposed methods by $\mathcal{Q}_E$ slices of 10% when $f_1^*(c_k, D) = \text{XHTML}$

dynamic system in bins]0.6-07] to]0.8-09] is far from that of the exhaustive static system, although this number is closer to that of the exhaustive static system when methods 1 to 5 are used.

These two figures show the improvements achieved by the proposed methods in terms of the number of documents with an increased $\mathcal{Q}_E$. They also serve as a comparison, in terms of accuracy, between the proposed methods. Except for method 4, the greater the complexity of the method used, the greater the accuracy (the number of documents in each bin is closer to that of the exhaustive static system).

7.4 Network conditions

In this section, we show the impact of the network conditions, particularly the variation of the bitrate, on the performance of the proposed dynamic content adaptation framework.

7.4.1 Low network bitrate

In this experiment we keep the same context parameters as in the previous section and suppose that the actual network bitrate has decreased to $N_B(D) = 20$ kbps.

Figs. 16 and 17 show the optimal $\mathcal{Q}_E$ obtained by the proposed dynamic framework versus that obtained by the exhaustive static system for JPEG and XHTML, respectively. These results were computed without adjusting the estimated file size. Globally, the obtained results are close to optimality and as expected they present some outliers points for both JPEG and XHTML. Therefore, we multiplied the estimated file size by a ratio of 1.15, and the results are presented in Figs. 18 and 19. These figures show also the performance of the whole framework (methods 1 to 5) versus the exhaustive static and fixed-QF dynamic systems. As expected, the fixed-QF dynamic system performs very poorly under lower bitrate values whereas the

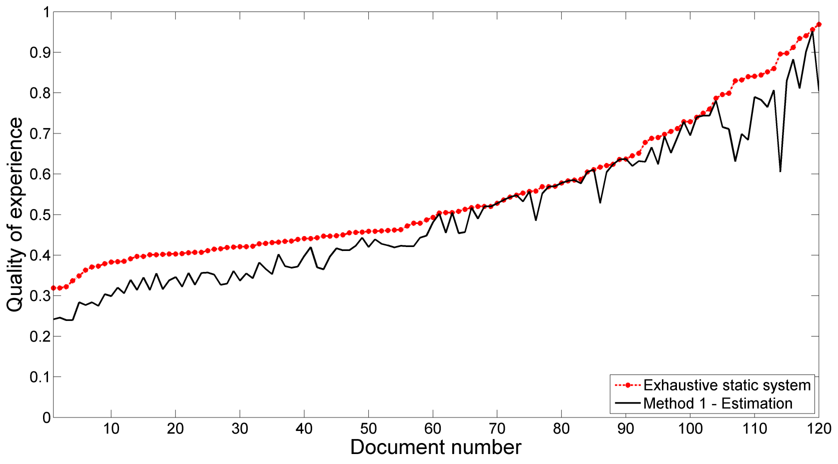


Fig. 16: JPEG optimal $\mathcal{Q}_E$ obtained by our framework (method 1) vs. that of the exhaustive static system computed with a file size ratio of 1, $N_B(D) = 20$ kbps and $N_L(D) = 488$ ms. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

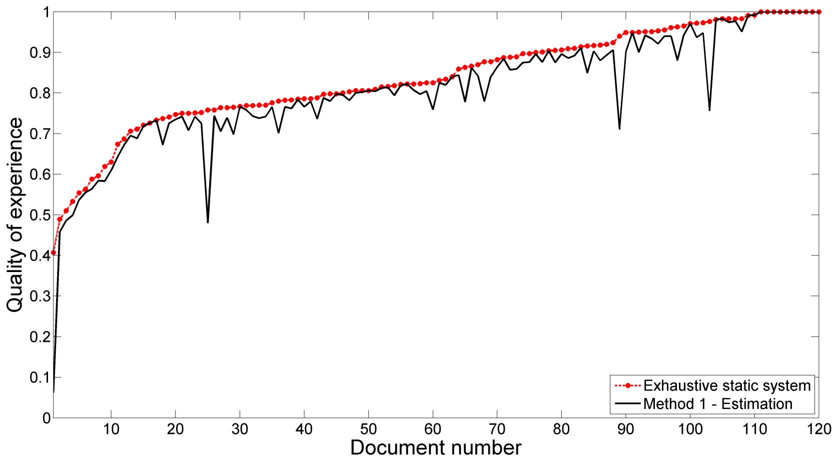


Fig. 17: XHTML optimal $\mathcal{Q}_E$ obtained by our framework (method 1) vs. that of the exhaustive static system computed with a file size ratio of 1, $N_B(D) = 20$ kbps and $N_L(D) = 488$ ms. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

proposed dynamic system is very close to optimality. Statistical details, showing the average deviation from optimality and its variance, are presented in Table 1.

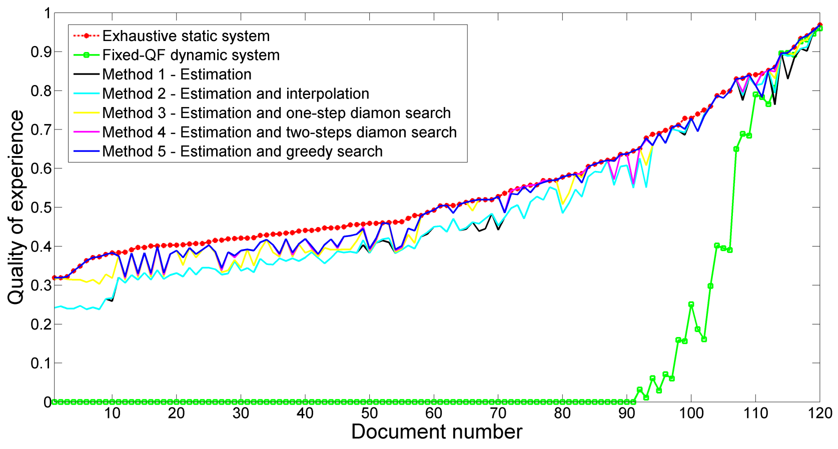


Fig. 18: JPEG optimal $\mathcal{Q}_E$ obtained by our framework vs. that of the exhaustive static and fixed-QF dynamic systems computed with a file size ratio of 1.15, $N_B(D) = 20$ kbps and $N_L(D) = 488$ ms. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

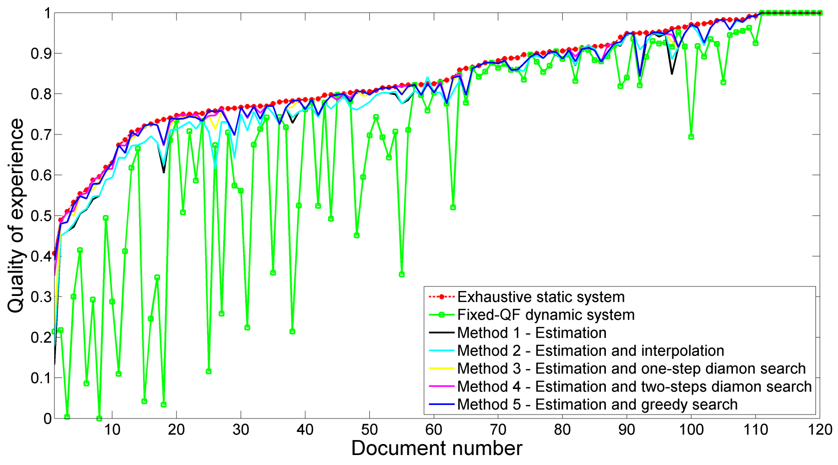


Fig. 19: XHTML optimal $\mathcal{Q}_E$ obtained by our framework vs. that of the exhaustive static and fixed-QF dynamic systems computed with a file size ratio of 1.15, $N_B(D) = 20$ kbps and $N_L(D) = 488$ ms. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

7.4.2 High network bitrate

In this experiment, we suppose that the mobile device D is connected to an EDGE network that has the following characteristics: $N_B(D) = 240$ kbps and $N_L(D) = 504$ ms [30]. The optimal $\mathcal{Q}_E$ obtained by the proposed dynamic system is compared to that obtained by the exhaustive static and fixed-QF dynamic systems, and the results are presented in Figs. 20 and 21 for JPEG and XHTML, respectively.

Table 1: Average deviation form optimality and its variance as computed with a file size ratio of 1.15, $N_B(D) = 20$ kbps and $N_L(D) = 488$ ms

	Average deviation from optimality		Variance ($\times 10^{-3}$)	
Methods	JPEG	XHTML	JPEG	XHTML
Method 1-Estimation	0.058	0.029	0.985	1.248
Method 2-Est. and interpolation	0.055	0.027	0.997	0.959
Method 3-Est. and one step diamond search	0.027	0.014	0.785	0.530
Method 4-Est. and two steps diamond search	0.017	0.011	0.529	0.324
Method 5-Est. and greedy search	0.016	0.012	0.445	0.315

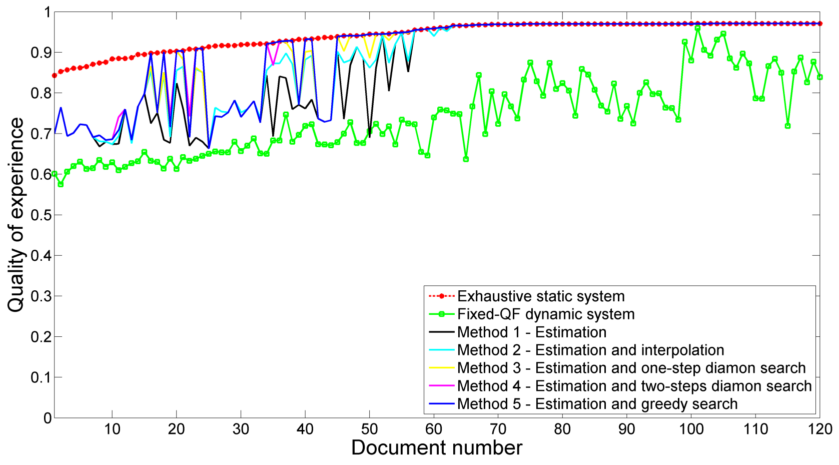


Fig. 20: JPEG optimal $\mathcal{Q}_E$ obtained by our framework vs. that of the exhaustive static and fixed-QF dynamic systems computed with a file size ratio of 1, $N_B(D) = 240$ kbps and $N_L(D) = 504$ ms. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

For JPEG, the optimal $\mathcal{Q}_E$ obtained by the proposed dynamic framework is close to optimality, especially using methods 4 and 5. Besides, almost 50% of the documents reached optimality, as shown in Fig. 20. Note that, these results were computed without adjusting the estimated file size. To improve even more the obtained results accuracy, we tested various file size ratio values (5%, 10%, 15% and 20%) and no noticable improvement was obtained. As discused earlier, we believe, that future research should be conducted to establish the right file size ratio in fucntion of the network conditions. The fixed-QF-dynamic system is still far from optimality compared to the proposed dynamic framework.

For XHTML, the obtained results are exceptional as we reached optimality in 99% of the documents. This confirms again the fact that XHTML is more precise than JPEG, wich makes it a very attractive format to consider in enterprise documents adapation. On the other hand, the fixed-QF dynamic system is very variable and gets far from optimality for a large number of documents.

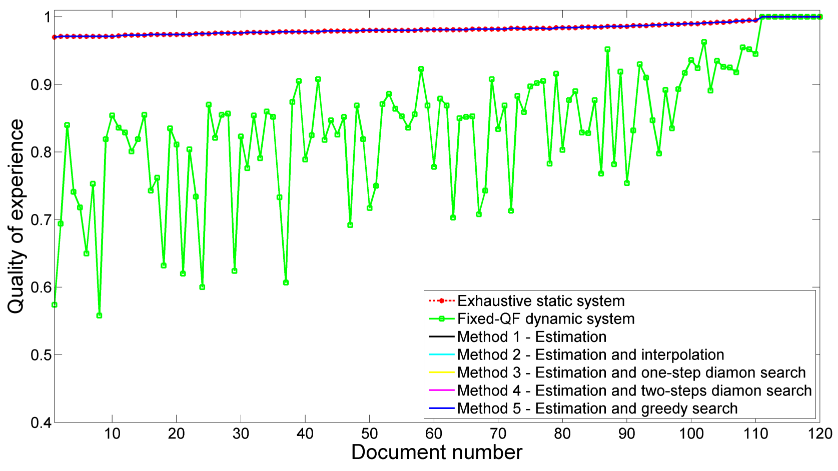


Fig. 21: XHTML optimal $\mathcal{Q}_E$ obtained by methods 1 to 5 vs. that of the exhaustive static and fixed-QF dynamic systems computed with a file size ratio of 1, $N_B(D) = 240$ kbps and $N_L(D) = 504$ ms. The slides are sorted according to the $\mathcal{Q}_E$ of the exhaustive static system

Table 2: Average deviation from optimality and its variance as computed with a file size ratio of 1, $N_B(D) = 240$ kbps and $N_L(D) = 504$ ms

	Average deviation from optimality		Variance ($\times 10^{-3}$)	
Methods	JPEG	XHTML	JPEG	XHTML
Method 1-Estimation	0.070	0	7.443	0
Method 2-Est. and interpolation	0.054	0	5.329	0
Method 3-Est. and one step diamond search	0.047	0	5.526	0
Method 4-Est. and two steps diamond search	0.043	0	5.449	0
Method 5-Est. and greedy search	0.043	0	5.710	0

The accuracy of the obtained results can be read also from Table 2, which shows the average deviation form optimality and its variance for both JPEG and XHTML.

7.5 Complexity of the proposed framework

The percentage of average $\mathcal{Q}_E$ obtained by methods 1 to 5 compared to that obtained by the exhaustive static system, versus the average complexity of these methods, is plotted in Fig. 22 for JPEG and in Fig. 23 for XHTML. These results are computed with the context information used in the experimental setup section, that is, $N_B(D) = 50$ kbps and $N_L(D) = 488$ ms.

In the case of JPEG, the $\mathcal{Q}_E$ obtained by method 1 (estimation only) is, on average, close to that obtained by the exhaustive static system, that is, 94% for JPEG and 97% for XHTML. They are even closer when the other methods are

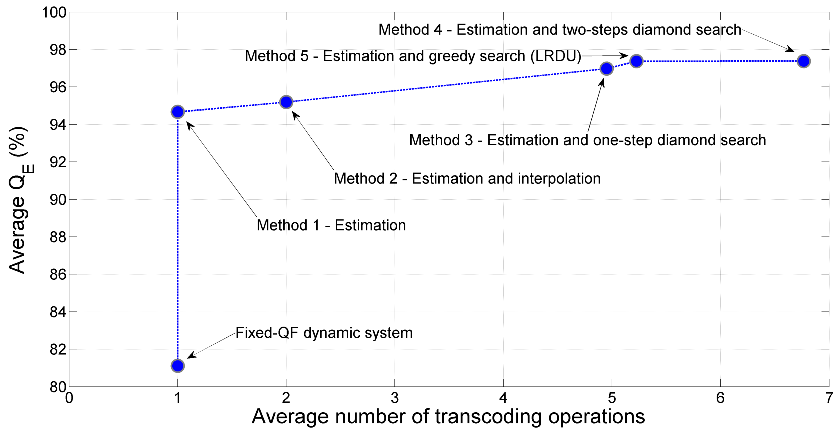


Fig. 22: Average $\mathcal{Q}_E$ vs. average complexity $f_1^*(c_k, D) =$ JPEG

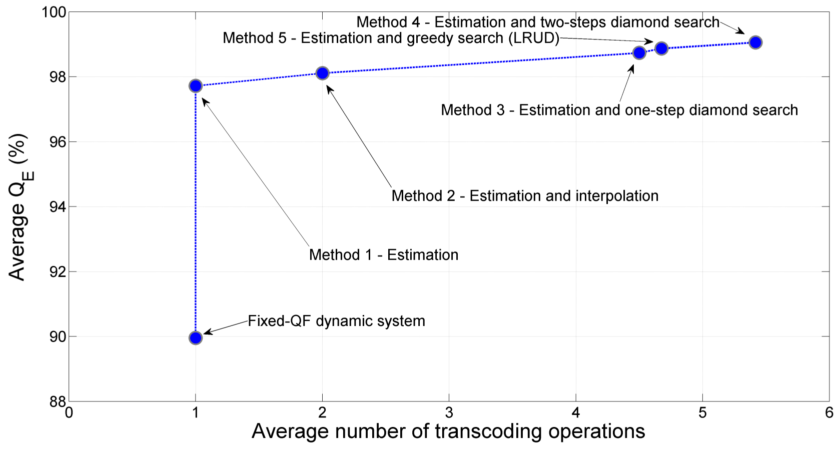


Fig. 23: Average $\mathcal{Q}_E$ vs. average complexity when $f_1^*(c_k, D) =$ XHTML

used: from 94% to 97% for JPEG, and from 97% to 99% for XHTML. For instance, for JPEG, method 5 reached 97%, which is only 3% away from optimality with a complexity of close to 5 operations, and for XHTML, it is less than 1% from optimality with a complexity near 5 operations.

The average improvement in $\mathcal{Q}_E$ obtained by methods 2 to 5 is relatively small. However, these figures hide the fact that the improvement in $\mathcal{Q}_E$ obtained follows that needed to reach optimality. This is clearly visible in Figs. 12 and 13, where we see that, when the target's relative gain is larger, the improvement obtained is also larger; conversely, the average relative gain is small because, for some slides, the target gain is small as well. This conclusion is also justified by the fact that the $\mathcal{Q}_E$ obtained by method 1 are around 94% for JPEG and 97% for XHTML, which is already an improvement.

Figs. 12 and 13 are very interesting, as they show that we can reach the optimal $\mathcal{Q}_E$ for a large number of slides using the proposed methods, especially using

Table 3: Average deviation from optimality, and its variance as computed with a file size ratio of (1.15 for JPEG and 1 for XHTML), $N_B(D) = 50$ kbps and $N_L(D) = 488$ ms

	Average deviation from optimality		Variance ($\times 10^{-3}$)	
Methods	JPEG	XHTML	JPEG	XHTML
Method 1-Estimation	0.037	0.021	0.872	0.951
Method 2-Est. and interpolation	0.034	0.018	0.676	0.729
Method 3-Est. and one-step diamond search	0.021	0.012	0.553	0.508
Method 4-Est. and two-steps diamond search	0.018	0.009	0.485	0.276
Method 5-Est. and greedy search	0.018	0.011	0.453	0.320

method 5 for JPEG and method 4 for XHTML (the scattered points that are on the diagonal line). Statistically speaking, for JPEG, 10% and 30% of the documents of $\mathcal{V}$ reached optimality using methods 1 and 5 respectively. For XHTML, 45% and 59% of the documents of $\mathcal{V}$ reached optimality using methods 1 and 4 respectively. Furthermore, overall, the results obtained were very reliable and close to optimality, as illustrated in Table 3, where the average deviation from optimality is computed, as well as the variance of these deviations, for each method.

Unlike JPEG, in the case of XHTML, method 4 is, on average, better than method 5 in terms of performance (average deviation versus complexity). As explained in [18], XHTML is very precise, and therefore, we don't need to search far from the second diamond area (as detailed in method 4). Indeed, we reached 1% of average deviation using method 4, which is very close to optimality.

On a final note, from Fig. 10, we can see that the delivery time is problematic for only about 45% of the slides, where we see the fixed-QF dynamic system performing very poorly. Obviously, transport is an issue, since the fixed-QF dynamic system always yields good visual quality using $QF = 80$. With a lower bitrate, the number of problematic slides would increase for the fixed-QF dynamic system, and could easily reach 100% with a low enough bitrate. This would make the fixed-QF dynamic system totally unusable. Of course, the opposite is also true, if the bitrate is high enough, the fixed-QF dynamic system would provide an excellent $\mathcal{Q}_E$. One advantage of the proposed methods is that they perform as well as possible under any circumstances. In fact, we could even exceed a QF of 80 if the bitrate were very high (while the fixed-QF dynamic system has constant quality). This is very important, as the bitrate can vary significantly during a Web conferencing session.

7.6 Examples of adapted versions of selected slides

In this section, we show adapted versions of two slides arbitrarily selected from the slide corpus. For the first selected slide, sub-figures 24a, 24b, 24c, and 24d show its optimal adapted version (computed with the static exhaustive method), two adapted versions, and the predicted near-optimal one respectively. They were created using $f =$ JPEG and the following combinations of z and QF:

- ($z = 0.4, QF = 70$): optimal combination.

Table 4: Computed $\mathcal{Q}_V$, T_d and $\mathcal{Q}_E$ for the example of Fig. 24, when f = JPEG.

Figures	(z, QF)	$\mathcal{Q}_V$	T_d	$\mathcal{Q}_E$
24a	$(0.4, 70)$	0.621	5.163	0.620
24b	$(0.2, 80)$	0.399	2.191	0.399
24c	$(0.7, 10)$	0.540	3.763	0.540
24d	$(0.5, 40)$	0.569	5.134	0.568

Table 5: Computed $\mathcal{Q}_V$, T_d and $\mathcal{Q}_E$ for the example of Fig. 25, when f = XHTML.

Figures	(z, QF)	$\mathcal{Q}_V$	T_d	$\mathcal{Q}_E$
25a	(0.9,40)	0.767	5.358	0.759
25b	(0.1,70)	0.466	1.213	0.466
25c	(0.9,10)	0.583	2.470	0.583
25d	(0.6,80)	0.733	5.586	0.713

- $(z = 0.2, QF = 80)$: low resolution and high quality factor.
- $(z = 0.7, QF = 10)$: high resolution and low quality factor.
- $(z = 0.5, QF = 40)$: predicted near-optimal combination.

Yet visually, sub-figure 24d is the closest one to the optimal adapted content 24a, whereas sub-figure 24b is small ($z = 0.2$) and becomes very blocky and blurred when zoomed. Sub-figure 24c contains visible blocks on the right and lower sides of the image. In addition, as shown in Table 4, the computed near-optimal adapted content (sub-figure 24d) is the closest one to the optimal adapted content in terms of $\mathcal{Q}_V$, T_d and $\mathcal{Q}_E$.

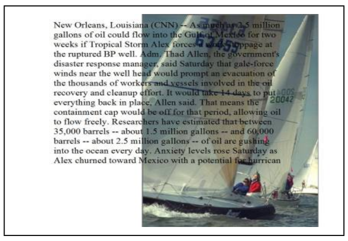
Similarly, for the second selected slide, sub-figures 25a, 25b, 25c, and 25d show its optimal adapted version (computed with the static exhaustive method), two adapted versions, and the predicted near-optimal one respectively. They were created using f = XHTML and the following combinations of z and QF:

- $(z = 0.9, QF = 40)$: optimal combination.
- $(z = 0.1, QF = 70)$: low resolution and high quality factor.
- $(z = 0.9, QF = 10)$: high resolution and low quality factor.
- $(z = 0.6, QF = 80)$: predicted near-optimal combination.

Visually, sub-figure 25d is the most similar to the optimal adapted version 25a. Sub-figure 25b is very small ($z = 0.1$) and became very blurred and blocky when zoomed. In sub-figure 25c, the text quality has been preserved since it is an XHTML output. However, the image contains visible blocks on the right and lower sides of the figure. This is also justified by the computed $\mathcal{Q}_V$, T_d and $\mathcal{Q}_E$ presented in Table 5.

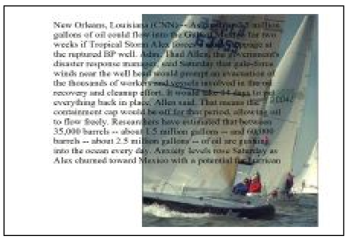
8 Conclusion

In this paper, we studied the adaptation of enterprise documents, considering two target formats: JPEG-based and XHTML-based Web pages. In the latter, the components of each enterprise document page are converted separately and wrapped in a Web page, which can comprise both text and images. Unlike the



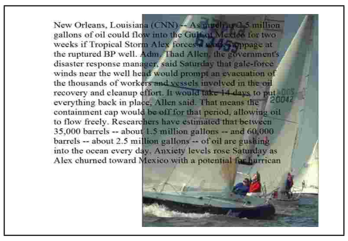
New Orleans, Louisiana (CNN) -- As much as 2.5 million
gallons of oil could flow into the Gulf of Mexico for two
weeks if Tropical Storm Alex forces a work stoppage at
the ruptured BP well. Adm. Thad Allen, the government's
disaster response manager, said Saturday that gale-force
winds near the well head would prompt an evacuation of
the thousands of workers and vessels involved in the oil
recovery and cleanup effort. It would take 14 days to put
everything back in place, Allen said. That means the
containment cap would be off for that period, allowing oil
to flow freely. Researchers have estimated that between
35,000 barrels -- about 1.5 million gallons -- and 60,000
barrels -- about 2.5 million gallons -- of oil are gushing
into the ocean every day. Anxiety levels rose Saturday as
Alex churned toward Mexico with a potential for hurrican

(a)



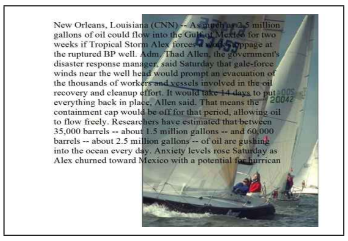
New Orleans, Louisiana (CNN) -- As much as 2.5 million
gallons of oil could flow into the Gulf of Mexico for two
weeks if Tropical Storm Alex forces a work stoppage at
the ruptured BP well. Adm. Thad Allen, the government's
disaster response manager, said Saturday that gale-force
winds near the well head would prompt an evacuation of
the thousands of workers and vessels involved in the oil
recovery and cleanup effort. It would take 14 days to put
everything back in place, Allen said. That means the
containment cap would be off for that period, allowing oil
to flow freely. Researchers have estimated that between
35,000 barrels -- about 1.5 million gallons -- and 60,000
barrels -- about 2.5 million gallons -- of oil are gushing
into the ocean every day. Anxiety levels rose Saturday as
Alex churned toward Mexico with a potential for hurrican

(b)



New Orleans, Louisiana (CNN) -- As much as 2.5 million
gallons of oil could flow into the Gulf of Mexico for two
weeks if Tropical Storm Alex forces a work stoppage at
the ruptured BP well. Adm. Thad Allen, the government's
disaster response manager, said Saturday that gale-force
winds near the well head would prompt an evacuation of
the thousands of workers and vessels involved in the oil
recovery and cleanup effort. It would take 14 days to put
everything back in place, Allen said. That means the
containment cap would be off for that period, allowing oil
to flow freely. Researchers have estimated that between
35,000 barrels -- about 1.5 million gallons -- and 60,000
barrels -- about 2.5 million gallons -- of oil are gushing
into the ocean every day. Anxiety levels rose Saturday as
Alex churned toward Mexico with a potential for hurrican

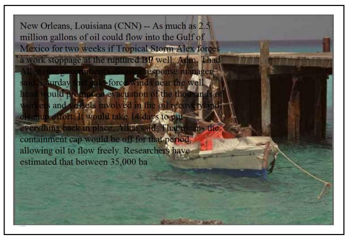
(c)



New Orleans, Louisiana (CNN) -- As much as 2.5 million
gallons of oil could flow into the Gulf of Mexico for two
weeks if Tropical Storm Alex forces a work stoppage at
the ruptured BP well. Adm. Thad Allen, the government's
disaster response manager, said Saturday that gale-force
winds near the well head would prompt an evacuation of
the thousands of workers and vessels involved in the oil
recovery and cleanup effort. It would take 14 days to put
everything back in place, Allen said. That means the
containment cap would be off for that period, allowing oil
to flow freely. Researchers have estimated that between
35,000 barrels -- about 1.5 million gallons -- and 60,000
barrels -- about 2.5 million gallons -- of oil are gushing
into the ocean every day. Anxiety levels rose Saturday as
Alex churned toward Mexico with a potential for hurrican

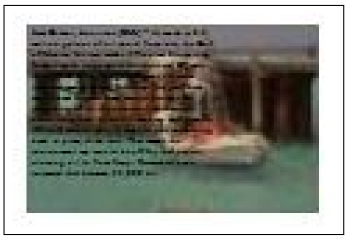
(d)

Fig. 24: Adapted versions of the the first selected slide, when $f_1^*(c_k, D)$ = JPEG. (a) Transcoded with optimal parameters ($z = 0.4$ and $QF = 70$), (b) Transcoded with $z = 0.2$ and $QF = 80$, (c) Transcoded with $z = 0.7$ and $QF = 10$, (d) Transcoded with near-optimal parameters computed with method 5 ($z = 0.5$ and $QF = 40$)



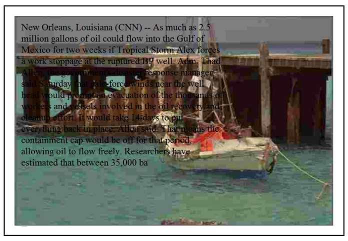
New Orleans, Louisiana (CNN) -- As much as 2.5
million gallons of oil could flow into the Gulf of
Mexico for two weeks if Tropical Storm Alex forces
a work stoppage at the ruptured BP well. Adm. Thad
Allen, the government's disaster response manager,
said Saturday that gale-force winds near the well
head would prompt an evacuation of the thousands of
workers and vessels involved in the oil recovery and
cleanup effort. It would take 14 days to put
everything back in place, Allen said. That means the
containment cap would be off for that period,
allowing oil to flow freely. Researchers have
estimated that between 35,000 ba

(a)



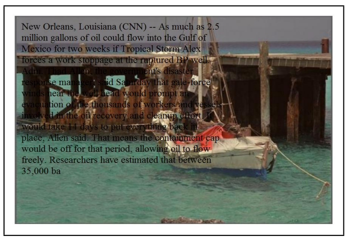
[illegible]

(b)



New Orleans, Louisiana (CNN) -- As much as 2.5
million gallons of oil could flow into the Gulf of
Mexico for two weeks if Tropical Storm Alex forces
a work stoppage at the ruptured BP well. Adm. Thad
Allen, the government's disaster response manager,
said Saturday that gale-force winds near the well
head would prompt an evacuation of the thousands of
workers and vessels involved in the oil recovery and
cleanup effort. It would take 14 days to put
everything back in place, Allen said. That means the
containment cap would be off for that period,
allowing oil to flow freely. Researchers have
estimated that between 35,000 ba

(c)



New Orleans, Louisiana (CNN) -- As much as 2.5
million gallons of oil could flow into the Gulf of
Mexico for two weeks if Tropical Storm Alex
forces a work stoppage at the ruptured BP well.
Adm. Thad Allen, the government's disaster
response manager, said Saturday that gale-force
winds near the well head would prompt an
evacuation of the thousands of workers and vessels
involved in the oil recovery and cleanup effort. It
would take 14 days to put everything back in
place, Allen said. That means the containment cap
would be off for that period, allowing oil to flow
freely. Researchers have estimated that between
35,000 ba

(d)

Fig. 25: Adapted versions of the the second selected slide, when $f_1^*(c_k, D)$ = XHTML. (a) Transcoded with optimal parameters ($z = 0.9$ and $QF = 40$), (b) Transcoded with $z = 0.1$ and $QF = 70$, (c) Transcoded with $z = 0.9$ and $QF = 10$, (d) Transcoded with near-optimal parameters computed with method 5 ($z = 0.6$ and $QF = 80$)

JPEG-based format, the XHTML-based format provides more flexibility, allowing text editing and keyword searching.

To dynamically identify the optimal transcoding parameter combinations that provide the end-user with the best user experience possible while satisfying the target mobile device's constraints, we presented a prediction-based dynamic content adaptation framework and applied it to JPEG and XHTML formats. First, we defined an objective quality of experience measure that takes into account the visual quality of the adapted content and the time it takes that content to reach the recipient (transport quality), and so addresses the three requirements needed in any quality of experience design [14].

Using the proposed quality of experience measure as a quality criterion, the proposed framework estimates, on-the-fly, near-optimal transcoding parameters (format, scaling, and quality factor), that maximize this measure with less computational complexity. It exploits the predicted SSIM and the relative file sizes of transcoded JPEG images, subject to changing their scaling parameter and quality factor [26,3,4].

In [18], we presented a content adaptation method that estimates near-optimal transcoding parameters dynamically and requires only one transcoding operation. Overall the obtained results were very close to optimality (exhaustive static system), but the presence of some outlier points was noted. In the case of JPEG, we showed in [17] that the results can be improved significantly if more transcoding operations are tolerated. In this paper, we improved the estimation method of [18], by adjusting the estimated file size, to correct these outlier points, and based on this method, we presented four other methods, which are variants of the first one, but with improved accuracy. For instance, for JPEG, the obtained results were 6% and 3% far from optimality respectively, using methods 2 and 5. In the case of XHTML, they were 3% and 1% far from optimality, using methods 2 and 5 respectively. Furthermore, we reached optimality in 30% and 59% of the documents tested for JPEG and XHTML respectively, which make the proposed framework very appealing.

In the future, it will be interesting to investigate the relationship between the proposed $\mathcal{Q}_E$ measure and human perception. Alternatively, a mapping function could be found to map the $\mathcal{Q}_E$ values to human perception. Finally, the proposed framework has been applied to OpenOffice Impress presentations, which are mostly used in Web conferencing applications. Though our framework is designed to be general, future research could be conducted to validate its applicability to other enterprise document types, such as MS Word and MS Excel.

References

1. Adobe Systems Incorporated. Adobe Connect: Web Conferencing for the Enterprise. `http://www.adobe.com/products/adobeconnect.html`, 1012. Accessed on 13 February 2014.
2. 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*, pages 81–92, 1999.
3. 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*, pages 68–74, Mar 2009.
4. 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*, 19(3):712–721, Mar 2010.

5. Stephane Coulombe and Guido Grassel. Multimedia adaptation for the multimedia messaging service. *IEEE Communications Magazine*, 42(7):120–126, July 2004.
6. A. Dieckmann, K. Dippold, and H. Dietrich. Compensatory versus Noncompensatory Models for Predicting Consumer Preferences. *Judgment and Decision Making*, 4(3):200–213, 2009.
7. Google Inc. Access Google Docs on a mobile browser. `https://support.google.com/drive/answer/1066285?hl=en&ref_topic=1361435`, 2011. Accessed on 13 February 2014.
8. 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*, 5(6):8–17, 1998.
9. Andreas Holzinger, Michael Geier, and Panagiotis Germanakos. On the Development of Smart Adaptive User Interfaces for Mobile e-Business Applications - Towards Enhancing User Experience - Some Lessons Learned. In *DCNET/ICE-B/OPTICS*, pages 205–214, 2012.
10. Y. Hwang, J. Kim, and E. Seo. Structure-Aware Web Transcoding for Mobile Devices. *IEEE Internet Computing*, 7(5):14–21, 2003.
11. R. Jan, C. Lin, and M. Chern. An Optimization Model for Web Content Adaptation. *Computer Networks*, 50(7):953–965, 2006.
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*, pages 72–79, 1999.
13. Thomas Kleinberger, Andreas Holzinger, and Paul Mller. Adaptive Multimedia Presentations Enabling Universal Access In Technology Enhanced Situational Learning. *Universal Access in the Information Society*, 7(4):223–245, 2008.
14. F. Kuipers, R. Kooij, D. De Vleeschauwer, and K. Brunnström. Techniques for Measuring Quality of Experience. In *Wired/Wireless Internet Communications*, volume 6074 of *Lecture Notes in Computer Science*, pages 216–227. Springer-Verlag Berlin/Heidelberg, 2010.
15. L. Lee and R. Anderson. A Comparison of Compensatory and Non-Compensatory Decision Making Strategies in IT Project Portfolio Management. `http://aisel.aisnet.org/irwitpm2009/9`, 2009. Accessed on 13 February 2014.
16. D. Li and U. Chandra. Building Web-based Collaboration Services on Mobile Phones. In *Int. Symp. on Collaborative Technologies and Systems*, pages 295–304. IEEE, May 2008.
17. H. Louafi, S. Coulombe, and U. Chandra. Efficient Near-Optimal Dynamic Content Adaptation Applied to JPEG Slides Presentations in Mobile Web Conferencing. In *The 27th IEEE International Conference on Advanced Information Networking and Applications (AINA-2013)*, pages 724–731, Mar 2013.
18. H. Louafi, S. Coulombe, and U. Chandra. Quality Prediction-Based Dynamic Content Adaptation Framework Applied to Collaborative Mobile Presentations. *IEEE Transactions on Mobile Computing*, 12(10):2024–2036, Oct 2013.
19. W. Y. Lum and F.C.M. 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*, page 239, 2002.
20. W. Y. Lum and F.C.M. Lau. User-Centric Content Negotiation for Effective Adaptation Service in Mobile Computing. *IEEE Transactions on Software Engineering*, 29(12):1100–1111, Dec 2003.
21. W. Y. Lum and F.C.M. Lau. User-Centric Adaptation of Structured Web Documents for Small Devices. In *19th Int. Conf. on Advanced Information Networking and Applications (AINA'05)*, volume 1, pages 507–512. IEEE, 2005.
22. R. Mohan and J.R. Smith. Adapting Multimedia Internet Content for Universal Access. *IEEE Transactions on Multimedia*, 1(1):104–114, Mar 1999.
23. F. F. Nah. A study on Tolerable Waiting Time: How Long Are Web Users Willing to Wait? *Behaviour & Information Technology*, 23(3):153–163, Jan 2004.
24. B. D. 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*, 8(4):57–66, 1995.
25. OpenOffice. The OpenOffice.org API Project. `http://api.openoffice.org`, 2010. Accessed on 13 February 2014.

26. 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*, pages 378–382. IEEE, Jun 2008.
27. PresenterNet. Interactive Web Presentations. `http://www.presenternet.com/`, 2009. Accessed on 13 February 2014.
28. G. Ryan and M. Valverde. Waiting in line for Online Services: A Qualitative Study of The User's Perspective. *Information Systems Journal*, 16(2):181–211, Apr 2006.
29. H. R. Sheikh, M. F. Sabir, and A. C. Bovik. A Statistical Evaluation of Recent Full Reference Image Quality Assessment Algorithms. *IEEE Transactions on Image Processing*, 15(11):3440–3451, Nov 2006.
30. 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)*, pages 1–6, Jun 2007.
31. The MathWorks. Z-shaped built-in membership function. `http://www.mathworks.com/help/toolbox/fuzzy/zmf.html`,, 2012. Accessed on 13 February 2014.
32. The USC-SIPI. The USC-SIPI Image Database. `http://sipi.usc.edu/database/`, 2012. Accessed on 13 February 2014.
33. 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*, 13(4):600–612, 2004.
34. 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*, volume 4138 of *LNCS*, pages 69–80. Springer Berlin / Heidelberg, 2006.
35. Zoho. Zoho Mobile. `http://www.zoho.com/mobile/`, 2010. Accessed on 13 February 2014.

Habib Louafi 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, and Ph.D. from the École de technologie supérieure (ÉTS is a part of the Université du Québec network) in 2013. He is currently working as a postdoctoral fellow at Synchromedia Laboratory for multimedia communication in telepresence (ÉTS). His fields of interest include mobile computing, context-aware systems, QoE, content adaptation and collaborative mobile Web conferencing.

Stéphane Coulombe 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. 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.