M2A: A Framework for Visualizing Information from Mobile Web to Mobile Augmented Reality

Kit Yung LAM*, Lik Hang LEE*, Tristan BRAUD*, and Pan HUI*†

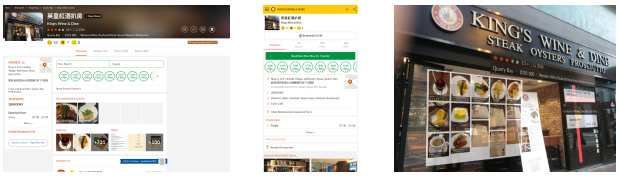
*The Hong Kong University of Science and Technology - Hong Kong
†University of Helsinki - Finland
Email: kylambd@connect.ust.hk,lhleeac@connect.ust.hk, braudt@ust.hk, panhui@ust.hk

***Abstract*—Mobile Augmented Reality (MAR) drastically changes our approach to computing and user interaction. Web browsing, in particular, is impractical on AR devices as current web design principles do not account for three-dimensional display and navigation of virtual content. In this paper, we propose Mobile to AR (M2A), the first framework for designing web pages for AR devices. M2A exploits the visual context to display more content while enabling users to locate relevant data intuitively with minimal modifications to the website. To evaluate the principles behind the framework, we implement a demonstration application in AR and conduct two user-focused experiments. Our experimental study reveals that participants with M2A are 5 times faster to find information on a web page compared to a smartphone, and 2 times faster than a traditional AR web browser. Furthermore, users consider navigation on M2A websites to be significantly more intuitive and easy to use compared to their desktop and mobile counterparts.**

I. Introduction

Mobile Augmented Reality (MAR) augments the physical world by overlaying a virtual layer of information on mobile devices. Recently, the improvement of mobile hardware has enabled the development of advanced MAR applications. In this paper, we focus on web browsing in AR. Efficient website design has been an open problem since the rise of the Web in 1990 [1]. The rapid evolution of technology forced websites to constantly update their design practices. Until recently, terminals established a clear separation between the physical and virtual world, around which most design principles were formulated. In web pages, this separation can either be a constraint (limited display size), or a strength (user interaction relying on physically touching the content). AR presents another shift in user interaction with the web by blurring the line between the physical and the virtual world. This enables for new content display opportunities, among which context-aware web browsing [2] and 3D display of information. In its current state, web browsing is impractical and under-exploits the capabilities of AR devices.

In this paper, we propose Mobile-to-AR (M2A), the first framework to provide a user-centric web browsing experience adapted to AR devices. This framework encompasses both the design principles for enabling AR and the software architecture to provide developers with the core AR functions. M2A expands upon the *responsive design* approach from mobile web development. We formulate guidelines to exploit the capabilities of AR devices with minimal modifications to the website's code. M2A also provides a software architecture to isolate the core AR functions (e.g. object recognition, mapping, tracking, rendering) from the website design. Web developers can then focus on designing context-aware AR websites without considering the software implications.



(a) Computer (b) Mobile (c) AR (concept)

Fig. 1: The website OpenRice as displayed on a personal computer (a), a mobile (b), and in AR (c). The website had to be trimmed down for mobile display. Similarly, some specific information was selected for AR display, and positioned according to the environment.

Throughout this paper, we focus on the website OpenRice[1], a popular restaurant rating website in Asia, as our case study. Indeed, restaurant rating websites are a representative example of context-aware web browsing. Moreover, they present a wide diversity of contents to display in the 3D space. We represent how the OpenRice page is displayed on various devices in Figure 1. The desktop version (Figure 1a) uses a standard 2-column design, with a banner featuring important information. The mobile version (Figure 1b) adopts a single-column view to cope with the small size of the device. The most important information is immediately visible, while secondary content is accessible within tabs. Thanks to responsive design principles, the mobile version contains the exact same elements as the desktop version, displayed in a fashion adapted to the screen size. Finally, Figure 1c represents our vision for web browsing on AR devices. M2A exploits the responsive design blocks and uses the context information to display them. The application displays the elements of the website without obstructing the information coming from the physical world.

The contribution of this paper is threefold:

1) **Web design for AR and Layout CSS (LCSS)**: We formulate design principles for web developers to design

[1]https://www.openrice.com

websites for AR. We establish the foundations of web design in a context-aware 3D environment with the currently impractical interaction methods. In order to accommodate for the specificity of AR, we introduce a CSS extension, Layout CSS, that allows overlaying web content over the physical view of the user.

2) **M2A framework architecture**: We propose a software architecture that enables the development of websites for AR in a similar fashion to developing mobile websites, as an extension of the responsive design principles.
3) **User-centric evaluation of M2A**: We implement a demonstration application that displays context-aware web pages. In this application, we display the web pages according to the M2A design principles through the proposed software architecture. We evaluate the validity of the design principles through a user experiment focused on the Human-Computer Interaction perspective. The participants found the information with M2A 5 times faster than when browsing the web on their smartphone and reported M2A to be significantly less demanding to use than the former method.

The rest of this paper is organized as follows. After a review of the current literature in Section 2, we present the architecture of our framework in Sections 3 and detail its two key features in Sections 4 and 5. Finally, we evaluate our framework in Section 6 and discuss the results for future developments.

II. Related Work

More than a hundred of MAR standalone applications have been proposed in academic literature [3], [4]. However, most of the existing works consider standalone MAR applications for a specific platform and operating system configuration, leading to the current fragmented landscape of AR applications.

A. The Need for AR Web Browsers

In recent years, various MAR application frameworks have been presented, which either leverage the web technologies to implement a platform-specific AR browser [5], [6], or rely solely on web browsers themselves [7], [8]. Macintyre et al. [5] as well as You et al. [6] demonstrate the feasibility of a web-centric approach for AR applications. However, both solutions focus on displaying simple web elements in a contained browser. In this paper, we decompose a traditional webpage to embed its major contents in AR. Regarding web browsers in AR, Klein et al. [7] present a method that encapsulates low-level image-processing to support real-time 3D rendering under HTML5 DOM. Oberhofer et al. [8] propose a natural feature tracking algorithm to achieve an AR pipeline in web browsers. The aforementioned works focus on the AR content generation and rendering, but do not address the issue of visual content presentation and optimized AR layout. Moreover, these works display either limited content or regular websites, which are unadapted to browsing in an AR context. Langlotz et al. [9] suggest that it is important to thoughtfully design a generic AR browser that is capable of converting Web content into the AR environment in a user-friendly and effective manner.

B. AR Web Browsers and User Interaction

Even though existing AR browsers harness the web technologies [10], [11], they do not holistically integrate user interaction within an AR interface. A survey conducted by Grubert et al. [12] shows that AR applications, and thus AR browsers, need to consider the user's interaction effort to appeal to a general audience. A prior work [13] studies optimized information placement. This early work identifies the needs for visual presentation in AR. Our work aligns with their goal to enhance the user's ease of access to the AR information. We also implement optimized layout and improved visual presentation for AR content within the size-limited display on mobile devices. In a more recent work, Dangkham et al. [14] propose a tourism application using an HTML 5 framework to generate AR content. Even though the study specifies the relationship between the sensor data and web content, the web content rendering and its arrangementis not addressed. Tatzgern et al. [15] propose to use adaptive data visualization to handle high amounts of information in AR displays. They organize the data into a hierarchical structure. Currently, AR is shifting from smartphones and tablets to AR headsets [16] which no longer support touch-based interaction [17]. The hierarchical structure of information will, therefore, involve a lot of mid-air interaction. This type of interaction suffers from significant drawbacks such as the Midas problem [18] and the Gorilla Arm Syndrome [19], which decrease users' interest in the application. A hierarchical structure is thus not adapted to AR headsets. In our design guidelines, we thus adopt the concept of a flattened hierarchy. In comparison to these studies, M2A provides context-aware web browsing by directly converting the web content into an optimized AR layout within a generic framework.

C. AR Framework Design for Context-Aware Web Browsing

Regarding AR framework design, Huang et al. [20] propose a client-server architecture to offload computationally intensive tasks to a distant server. Their work contributes to a portion of the foundations of M2A software architecture. Engelke et al. [10] propose a generic framework using web components to render rich AR content when QR codes are detected in the environment. However,they do not emphasize the problems related to web browsing on AR devices. Langlotz et al. propose [9] a similar architecture for context-aware AR browsing. In this study, the geo-localized information is placed at the recognized 'region of interest'. However, their AR browser is limited to the specific subset of route navigation applications and cannot display all web content. Although our work also considers enhancing the context-awareness of web browsing through the usage of various sensors, M2A presents a significant difference in terms of user interaction. In M2A, the idle space is used to show the web content for AR browsing scenarios and we avoid putting the AR content in the 'region of interest' that is intentionally reserved for user's interaction with the physical object. In this way, the information from both the virtual and physical world is appropriately presented to the user.

III. M2A Framework

Our M2A framework focuses on the ease of accessing information from the user's perspective. In other words, our framework aims at reducing the user's effort and time for finding contents in a mobile AR environment. In this section, we explain the motivation behind the design of our M2A framework and its advantages compared to the existing literature.

A. Redefining web design for AR

*1) **Flattening the web for user interaction**:* The ease of accessing information is highly relevant to the user's steps to reach the information as stated in Shannon's information theory [21] and Fitts' Law [22]. According to these theories, the user's displacement of the pointer to reach a target will directly impact the task completion time. The total completion time is multiplied by the number of targets.

Current interface design principles are still massively anchored in the world of desktop computer and handheld mobile devices [23][24]. Most interfaces rely on a hierarchical structure in which layers of information are revealed through direct user interaction. Although these design principles make sense in a 2D environment of limited size such as the screen of a smartphone where the content is directly touchable, MAR features a much wider tridimensional display surface, where the content is intangible without bulky and expensive external hardware. Most interaction methods in AR are cumbersome, error-prone, and can in some cases lead to physical strain (Gorilla arm). Efficient and comfortable user interaction with AR content remains an open problem [17].

We consider that hierarchical interfaces are fundamentally unadapted to AR. Indeed, a significant number of tasks are involved in hierarchical structures to access relevant information. In AR, this methodology leads to a multiplication of cumbersome operations. As such, we consider that prioritizing the display of the content over improving the input interaction methods would provide a major advantage in terms of information access speed. MAR presents two fundamental characteristics: (i) The AR space is 3D and allows for more content to be displayed at the same time. (ii) MAR allows displaying context-relevant information. Through M2A, we propose to flatten the hierarchical structure of applications and websites in the 3D AR environment. As such, we directly eliminate tasks in the sequence to reach information. Combined with the context-awareness of AR, we can show the important information at first glance to avoid further interaction.

By reducing the number of interactions, we significantly simplify the interaction methods. The framework encourages more intuitive interactions between the user and the web elements while avoiding difficult-to-use external devices.

*2) **Responsive design for AR**: Responsive design* allows the elements of a web page to adapt to the screen size of different devices with minimal effort for the developer. Responsive design lets the device perform the display operations, and focus on the content, organized by blocks and marked through specific ids in the HTML code. Through M2A, we propose to extend these principles to the development of AR websites.

Currently, responsive web design mainly adapts the website to the size and resolution of the screen, both in terms of content displayed and interaction methods. Most frameworks use Fluid Grid systems to separate the information into blocks, which are then displayed relatively to the screen size. The AR world introduces several new constraints on top of screen size. First of all, AR brings a third dimension to content display, which allows displaying more information. Another important point is the capability of AR to display context-relevant information and compose with the decor on-screen to guide the user towards relevant information. In the case of a restaurant-review website, our use case, the web designer can opt for a *digital storefront*. Each restaurant storefront is used to display the appropriate web page. The blocks composing the web page show at familiar emplacements over the physical world. For instance, basic information is displayed next to the restaurant sign, contact number and opening hours can show on the door, pictures appear next to the menu etc. This type of design requires the different parts of the website to be clearly annotated in order to be displayed in the correct place.

With M2A, we propose to reuse the fluid grid system to isolate information in blocks and provide additional properties to display them in a 3D AR environment. To this purpose, we propose a transformation engine which can turn the web content into an AR adaptable content. The content in the AR browser is aware of features from the surrounding environment and allows for more relevant web browsing in AR.

B. Software architecture

Our primary goal in developing M2A is providing a web browsing framework for mobile AR devices. The software architecture behind M2A processes the web content and lays it out according to the visual context. Prior works [10], [9] propose generic architectures that facilitate the development of AR application from the perspective of web development. They commonly present the web standards for AR applications, which governs the HTML, CSS (Cascading Style Sheets) Standard, X3D and VRML Standards. Our framework builds on top of the Server-Client model in these studies. In this paper, we emphasize the content transformation and placement within the physical world that these models do not consider.

Figure 2 shows M2A's architecture. M2A relies on a server-client architecture that can be deployed within the same machine, or on a distant machine in a cloud offloading context. The client side involves the various sensors present on the device (*Device Camera*, *GPS Sensor*), a lightweight *Marker Recognition Module* for basic interaction, and the *M2A AR Browser*, that transforms and places the content depending on context cues provided by the server side. The Server side contains the *Object Recognition Module*, that works together with the *Object Recognition Database* to identify objects and build a *Knowledge Graph* that is transmitted back to the AR Browser on the client side. These modules can be classified into three categories: device sensors, Image Analysis, and AR Browser. In the following subsections, we expand on the behavior and interaction of each module.

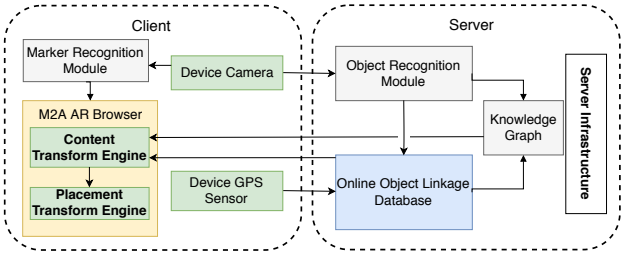


Fig. 2: M2A Server Client module Data Transmission flow and architecture

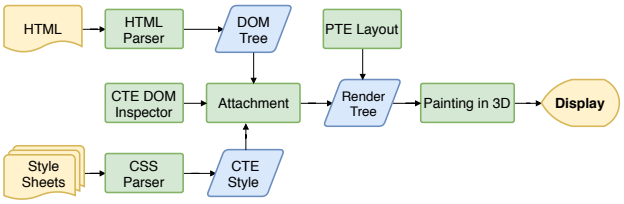


Fig. 3: M2A CTE and PTE workflow, built on the top of WebKit Rendering Engine Workflow [27]. The CTE style, CTE DOM Inspector and PTE Layout are the main components of M2A AR browser.

1) ***Device Sensors****:* The *Device Camera* generates the camera frames, which are the main resources for AR environment recognition. These frames are transmitted to the *Marker Recognition Module* and the *Object Recognition Module* for processing. These modules look for identifiable content within the camera frames to place the web content within the physical world. For simplicity reasons, we assume the AR device to only present a single embedded camera. In the case of devices using multiple cameras, the framework can easily be extended with additional modules on the server side to provide more information to the *Knowledge Graph* module.

The *Device GPS Sensor* is responsible for finding the GPS location of the user, which serve as search filtering parameters for Geo-located content in the database. In this study, we focus on the *Device GPS Sensor* as it is the most obvious sensor for context-awareness. However, data incoming from any other sensors (accelerometer, device clock) on the device can be used as search filtering for context-aware content.

2) ***Image Analysis****:* The *Marker Recognition Module* handles the camera frames from the *Device Camera*. The module analyzes markers such as QR codes and decodes the corresponding web address. It is the most basic image analysis technique that can almost always be performed on the device itself. The web content is then retrieved and sent to the M2A AR Browser for optimized content selection and AR layout design.

The *Object Recognition Module* processes the camera frame sent from the *Device Camera*. The recognizable object in the physical environment is first searched in the *Online Object Linkage Database*. If relevant content is found, it is directly returned by the *Online Object Linkage Database* to the *AR Browser*. Else, the *Object Recognition Module* directly issues a query containing the object to the *Knowledge Graph*.

The *Online Object Linkage Database* returns the web content relevant to the object identified from the *Object Recognition Module* combined with the data from other device sensors. In the case of the OpenRice website, the *Online Object Linkage Database* will only return web pages corresponding to restaurants located in the immediate vicinity of the user. Similar to the *Marker Recognition Module*, the selected website and web content are then processed by the M2A AR Browser.

The *Knowledge Graph* module applies the Google Knowledge Graph Search API [25] to provide missing content from external sources (e.g. Wikipedia).The relevant content will return to the *M2A AR Browser*.

3) ***M2A AR Browser****:* The *M2A AR Browser* receives the web content retrieved from the *Marker Recognition Module* on the client as well as from the image processing performed on the server. The AR Browser extracts the appropriate information from the web page and regulates the visualization style and layout rendering. The AR Browser is the scene manager for the extracted web content. Once the appropriate content is selected, it is displayed according to the visual context. We provide more details about the Content Transform Engine and the Placement Transform Engine in Section IV.

The software architecture we propose for M2A enables the visualization of common HTML web content in AR, associating real-world objects with their corresponding information on the Internet. We design our architecture to handle the extraction and placement of the web content that are critical to the very narrow sight of Spatial AR (SAR) [26] interfaces on head-mounted display or smartglasses. In this paper, we regard the camera as the major source of input due to its omnipresence, from smartphones to AR headsets. Other sensors, peripherals, and accessories can be used as additional input sources, increasing the relevance of information through the *Object Linkage Database*. The *M2A AR Browser* is responsible for determining the search criteria of web content to be displayed or directly load the specified websites. It also determines the placement of the information content.

IV. Transform Engine

The transform engine is at the core of the M2A AR Browser. In conventional web browsing, the layout is rendered within a rectangular area of fixed size. When we apply such principles to AR, the web page blocks the user vision on the whole area used to display. This hinders the user's perception of the real world environment and prevents AR headsets to seamlessly merge digital overlays within the physical environment. Web content should, therefore, undergo certain transformations to fit the physical world and improve readability.

We divide the rendering engine of the AR Browser in two modules: the Content Transform Engine (CTE) and the Placement Transform Engine (PTE). The CTE converts existing web content to dynamic block content for AR smartglasses. Attributes are addressed through CSS style sheets. These blocks

Attribute[2]	Attribute description
Node Name	DOM nodes to be controlled
Node Class	DOM nodes with class identifier to be controlled
Node Placement	Regulate the placement criteria for the node
Effective moving region	The three dimension space for the node to move in the dynamic environment, a fixed space restricting the area for the node by the meters specified.
isSeperate	Boolean show whether the elements is rendered as a web block separately.
Display	Similar to the CSS display property, web block can be hided by none value, and below values are validated in LCSS: inline, block, flex, grid, inline-block, inline-flex,inline-grid

TABLE I: AR Layout CSS (LCSS) style attributes

are then rendered through the WebKit rendering engine. The PTE lays the content out in real time.

Figure 3 shows the workflow of the CTE and PTE from the HTML/CSS code to the AR display. The rendering process follows the following steps:

1) The M2A AR browser receives and parses the HTTP response. If the Content-Type is HTML, the CTE passes it to step 2. Otherwise, the content is ignored.
2) The M2A rendering engine parses the HTML document. The document is converted into a DOM tree.
3) New requests are issued for each additional resource in the HTML source: images, style sheets, and JavaScript files.
4) External scripts such as Javascript are parsed and executed, the DOM elements are updated accordingly.
5) The CSS Parser parses the downloaded style information and aggregates the external CSS files with the markup style elements in the HTML source.
6) The CTE DOM Inspector traverses and inspects the DOM tree to identify the web page structure. To display content in AR, we propose Layout CSS (LCSS), which builds on top of the CSS language to provide AR specific display information. The LCSS instructions are stored as a separated style sheet. We present the basic attributes in Table **??**. Several DOM elements, including <header>, <nav>, and <img>, have default LCSS attributes. Each website can override the default LCSS style and create its own AR layout.
   a) The elements of <header> and <nav> in a web page generally provide the website title and navigation menu. In AR web browsing, the user recognizes a website by its header title and navigates the whole website by the menu on <nav>. Their DOM elements are extracted and rendered in the AR Browser.
   b) The content inside the <iframe>, <object> and <embed> elements, i.e. cross-context content, are associated with their own browsing context [28]. We do not render these elements in our architecture as they are frequently linked to external content, which damages the user experience.
   c) The elements with specified class, name or ID in LCSS, are extracted and rendered as a web block.
7) In the Rendering Tree, the CTE renders the contents as texture blocks. Also, the CTE enables the PTE to decide the position of the blocks in the AR environment.
8) The M2A browser renders each element to a texture block, according to the DOM tree and the style information.

The PTE places the web block surrounding the web content at a triggering 3D coordinate point $p_k \in \mathbb{R}^3$. The triggering point can be a QR code, an AR Marker or an object recognizable in the recognition module. The PTE starts with a keyframe image *key* where p_k coordinate points are discovered. The Web block set $b_0 \cdots b_n \in B$ are generated by M2A rendering engine, where $b_0 \cdots b_n$ are n set of blocks that can be displayed in the Rendering Tree. Below is the algorithm for the PTE:

1) Start at b_0 till b_n. If b_i is attached with a given LCSS Node Placement attribute and Display attribute, either the 2D position or 3D position corresponding to p_k, b_i is transformed and displayed on the AR screen corresponding the 3D position of p_k, and removed from B.
2) For the remaining b_i in B, the PTE inspects their ordering and places them on remaining space one by one. The PTE applies the Law of Proximity to the position of b_i. If b_i has identified other instances of the same class, the PTE will group them together, based on their display properties.

M2A can adapt a webpage to the AR context with minimal modifications. The CTE DOM inspector automatically extracts the main elements of the webpage and assigns them a default value. Most websites following modern design rules can be displayed in AR with no modification. In this work, we establish a basic design centered around a predefined anchor such as a logo or a QR marker. The developer can then optimize the AR display through the LCSS extension. As such, converting a website to AR only consists in tagging the elements within the webpage and writing the associated LCSS property.

M2A greatly benefits from grid-based reactive design frameworks such as Bootstrap. Such frameworks allow defining the element blocks while letting the engine display the content within them. As we apply the AR conversion at each frame in the CTE after executing the eventual javascript code, most common javascript frameworks such as `React` or `Angular` can execute independently from the M2A environment.

Figure 4 summarizes the steps behind web page rendering. After identifying the logo, the web content is recovered and analyzed. The CTE renders the basic blocks of the website separately according to their LCSS attributes. Here, the website

[2]Similar to CSS style, LCSS configuration have default attributes which can be overridden if definitions have been found

Fig. 4: CTE and PTE operations

Fig. 5: Experimental M2A Framework Implementation

renders around the restaurant's logo, in a format more friendly to an AR user than a single square displaying the content.

V. Experimental M2A Framework Implementation

We implement the M2A Framework in two test applications. Through these prototypes, we aim at proving that the M2A design principles and browser improve the user experience.

In the first application, we focus on the OpenRice website, our use case throughout the study. We statically isolate the different components of the webpage and arrange them around the designated marker (see Figure 5). We implement the application on smartphone (Android 8.1), in a standard handheld AR fashion. The webpages are displayed on top of the video feed from the camera. Users can navigate and zoom/unzoom using the touchscreen. This application aims at demonstrating the possibilities provided by the flattened hierarchy paradigm ant the context-aware web browsing model.

With our second application, we generalize the application cases of the M2A framework, while measuring the technology acceptance factor of our solution. This application features the following elements: (i) Full AR-headset support using the `HoloKit`[3], (ii) External controller support for user interaction, (iii) CTE DOM inspector: we extend our application to automatically convert websites to the AR space.

Both applications are implemented using the `Unity` engine. For simplicity reasons, we use the *Marker Recognition Module* in combination with Quick Response (QR) code markers [29], [30]. This design choice allows us to focus on the user experience while enjoying the stability and responsiveness of QR markers. It also considerably simplifies our experiments as all operations can be run on-device. Once the user positions his mobile device camera in front of the QR code, the user can freely interact with the AR content in the M2A Browser.

[3]https://holokit.io/

VI. Experimental evaluation

In this section, we evaluate M2A using our two test applications. First, we implement a basic version of the M2A framework for the OpenRice website for 3 restaurants around our university campus. We are trying to evaluate two key metrics: (1) How efficient is context-aware web browsing compared to mobile web browsing for location-specific information retrieval? (2) How fast can users locate information on an M2A webpage compared to a desktop-oriented webpage shown in AR? We then incorporate hand-free AR function as well as automated website parsing and display and evaluate the technology acceptance for M2A web browsing.

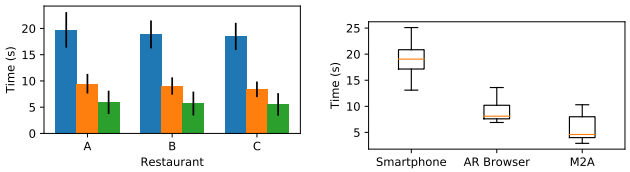
A. Experiment 1: Design principles efficiency

With this experiment, we evaluate the efficiency of M2A's design principles. To do so, we use the first application described in Section V. With this application, we statically implement the display of the OpenRice website around specific markers. The controls are similar to smartphone browsing to remove external parameters that may impact the measure.

*1) **Experimental Methodology**:* We set up our experiment as follows: we invite 16 users – students from the university, aged 20 to 35; 10 men and 6 women– to participate in the experiment. To rate their technological literacy, we ask the following question: "I use a smartphone or a computer for: 1 – once a week, 2 – at least once a week but not every day, 3 – every day, 4 – every day and feel confident with advanced usages, 5 – I work in the field of IT". The average technological confidence is 3.31, with scores ranging from 2 to 5, and 3 participants working in IT. It was the first experience with AR applications for 11 users. We also ask the participants their familiarity with the OpenRice website on a 1–5 scale, 1 being "never" and 5 being "every day. Users evaluate their familiarity on average at 2.125, with scores ranging from 1 to 4.

We select 3 restaurants in our university campus area, which we will refer to as Restaurant A to Restaurant C. Restaurants A, B, and C have an English name, containing respectively 21, 13 and 10 letters. The users are invited to go to these five restaurants, in the following fictional scenario: the restaurant is closed, and the user needs to find contact information on the OpenRice website. The users repeat the experiment three times per restaurant, with different tasks defined as follows:

1) We define a perimeter of 5m around the restaurant. Once the user enters this perimeter, he is allowed to take his phone out of his pocket. The user then opens the browser, types the address for OpenRice, the name of the restaurant and gets the information from the website. The phone is reset



(a) Average time to complete task by restaurant.

(b) Average time to complete task by interaction method.

Fig. 6: Average time to complete tasks by restaurant and interaction method (a), and by interaction method for all restaurants (b). Users got used to the experiment throughout the course of the study and improved their average time to complete for the smartphones and the AR browser. M2A is not impacted. The times to complete tasks are consistent over our sample, with low variance around the same values.

to its home page, between each experiment. For Chinese names, the users write the characters stroke by stroke on the screen, as the time per stroke is relatively similar to the time per letter in the Latin alphabet on a smartphone.

2) The user holds a smartphone (Samsung Galaxy S8), which runs a traditional web browser in AR. The browser opens and automatically displays the desktop version of the restaurant's OpenRice web page when the restaurant is recognized by the system. The user can zoom in and out of the browser, and scroll through the page.
3) The user repeats the same setup with the M2A version of the OpenRice webpage.

For each task, the participants have to find the phone number and opening hours of the restaurant and report this information to an external observer. The observer is in charge of explaining the experiment to the participants, measuring the time spent between the moment they enter the perimeter and the moment they find the required information, and collect the users' feedback. Each participant is assigned the tasks in a random order to avoid bias. In total, each user repeats the 3 tasks for each of the 3 restaurants. In the rest of this paper, we refer to the first task as **browsing on smartphone**, the second task as **AR browser**, and the third task as **M2A browser**.

At the end of the experiment, the participants fill a survey inspired by the NASA Task Load Index (TLX) [31]. Users rate their perceived effort to perform each task. We evaluate this effort over 4 main parameters, ranked from 1 to 10, 1 being the lowest: *Mental Demand* corresponds to the mental activity required; *Physical Demand* represents how strenuous the task is; *Total Effort* is the combination of *Physical*, and *Mental Demand*; *Frustration Level* refers to the amount of irritation caused by the task. This test evaluates the subjective aspects of each method. We aim at showing that websites developed using M2A noticeably enhance the user experience.

We asked the users the following questions, on a scale from 1 to 10, 1 being "very low" and 10 being "very high": (i) How mentally demanding was the task? (ii) How physically demanding was the task? (iii) How hard did you have to work

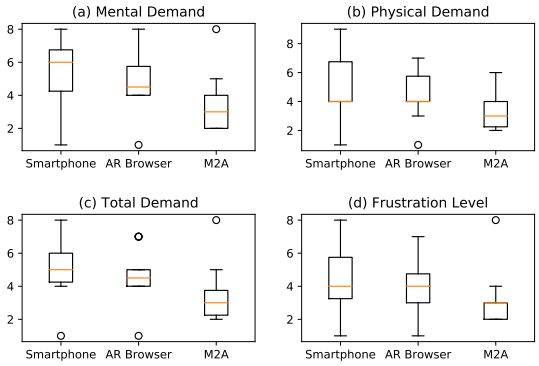


Fig. 7: Boxplots of reported Mental Demand (a), Physical Demand (b), Perceived Total Effort (c), and Frustration (d). Using M2A always results in low Mental and Physical Demand as well as low total effort. The AR browser and smartphone values are strongly overlapping, with higher variance for mental demand using the AR browser and higher variance for physical demand using the smartphones. Overall, the perceived effort and frustration was highest for the smartphones.

to accomplish your level of performance? (iv)How insecure, discouraged, irritated, stressed and annoyed were you?

*2) **Experimental Results**:* Figure 6 presents the average times to complete the task per restaurant and interaction method (6a) and the averages per task (6b). Unsurprisingly, both the AR browser and M2A allow the user to locate the information much faster than traditional browsing, thanks to the context-awareness of AR. Indeed, using the smartphone requires 26 to 50 operations compared to the context-aware AR browser where the user only scrolls. With the flattened structure of M2A websites, users locate information on average twice as fast as the AR Browser, despite the fact that more data is displayed.

As we can see in Figure 6b, for each method, the results exhibit a low variance for the AR Browser and the M2A application. The variance is higher when using the smartphones, with values ranging from 13.1 to 25.1s. This variance appears to be directly correlated to the technology level of participants. The participants with high technological literacy (4 and 5) show completion times under 21s, while those who estimated their technology level as low (2/5) complete the task in more than 23 seconds. Interestingly, the time to complete the task in AR (AR browser or the M2A application) does not show any correlation with the user's technology level. AR web browsing is more intuitive, regardless of technological literacy.

In order to confirm the difference between these metrics, we compute the paired-sample t-test against the null hypothesis (that is the paired data samples have the same mean) for each session. The t-test analysis shows a significant difference between all three methods for each session, whether Smartphone to AR Browsing ($T = 8.46, 10.03, 10.47$, $p << 0.05$), Smartphone to M2A ($T = 10.12, 11.71, 12.06$, $p << 0.05$) or AR Browsing to M2A ($T = 7.57, 12.57, 8.65$, $p << 0.05$).

Figure 7 shows the boxplots of the perceived mental and physical demand, the total effort and the frustration for each

task. The mental demand, physical demand, and the total effort are significantly lower for the M2A application than for the AR browser or the smartphone usage. The first and third quartile of the mental demand, physical demand, and total effort for the AR browser intersect with the first and third quartile for the smartphones. Despite significantly decreasing the completion time, the AR Browser does not provide such a relief to the user. As our experiment relies on handheld smartphones for displaying the AR content, we expect the physical demand to be similar in all three cases. However, users report that the simplification of the operations of M2A significantly reduces the physical demand. The frustration experienced with the smartphones shows extremely high variance: 25% of the users experience low frustration (≤ 3), while 25% display high levels of frustration (≥ 7). The AR Browser presents similar results ($Q_1 = 4$ and $Q_3 = 6$), as the 3D AR space is not efficient in displaying a traditional 2D website. When using M2A, users experience lower frustration, with 75% reporting frustration levels of 3 or less. Exploiting the AR space to display context-aware information significantly improves the user experience.

We compute the paired-sample t-test against the null hypothesis for each parameter and method. Overall, M2A is very different from both Smartphones and AR Browsing, with $T > 2$ and $p < 0.05$ for all mental demand, physical, and total demand pairings. Interestingly, even though M2A proves to be less demanding than the AR browser, we cannot contradict the null hypothesis for the Frustration level between the AR browser and M2A ($T = 1.37$, $p = 0.19$). However, M2A still provides less frustration than smartphones to the users ($T = 2.09$, $p < 0.05$).

M2A significantly improves the performance of users in finding information on websites. The design postulates formulated in Section III-A improve user interaction by reducing the number of tasks needed to retrieve the desired information, resulting in a less demanding and less frustrating experience. We also shed light on the impracticality of traditional web browsers in AR, with the overall effort and frustration halved by using M2A compared to a generic AR browser.

B. Experiment 2: Technology Acceptance

In this section, we evaluate the technology acceptance factor for M2A AR web browsing. To this purpose, we use the second application described Section V.

*1) **Experimental Methodology**:* We set up the experiment as follows: we invite 27 participants. The user panel is composed of university students and staff between 21 and 32 years old, with 18 men and 9 women. We ask the users to rate their technological literacy on a scale from 1 to 5, as well as their prior experience with AR applications. The average technological literacy is high, 3.70, with scores ranging 3 to 5. 18 users had prior experience with AR. We select three websites to browse in AR. These websites include the university online shop, the university center for arts and a coffee shop. We believe that such websites are the most probable to be looked at in an urban/campus AR setting as they are very dependent on the context (e.g. location, time of the day). A small remote allows users to zoom and move around the different elements.

After a brief explanation of the controls, the participants go through a 5 minutes acclimatization phase during which they browse freely the websites. We then ask them to find the following information on the websites: (i) the opening hours of the coffee shop on Tuesday, (ii) the name of the artists featured during an event, (iii) the price of a given item on the online shop. This phase forces the users to look for a piece of information on an unknown website within the M2A environment. Afterward we ask the participants to fill a Technology Acceptance survey. The questions and the results are presented Table II. Through this survey, we aim at measuring the following constructs: Perceived Usefulness (PU), Perceived Ease Of Use (PEOU), and Intention Of Use (IOU). We also calculate the Cronbach Alpha reliability coefficient for the three constructs. For all three measurements, the coefficient is above the minimum of 0.70, and the ideal 0.80. We represent these values in Table II.

*2) **Experimental Results**:* The results from the survey are presented in Table II. Overall, they are particularly positive. In particular, the participants found M2A to be easy to use, solving one of the major concern raised by the participants of the first experiment. Indeed, using the traditional AR browser was reported to be unpractical. Users not only considered M2A to be easy to use (avg=4.26, stdev=0.0.70), but also easy to learn (avg=4.62, stdev=0.88) and to become skillful at (avg=4.44, stdev=0.69). Nobody scored less than 3 out of 5 for any of the items in the PEOU. The users also considered M2A to be a convenient solution to browse the web in AR (avg=3.51, stdev=1.03). Although they considered that using M2A would only slightly improve the way they look for information on the web (avg=3.22, stdev=0.94), they found M2A useful to display content on an AR headset (avg=4.11, stdev=0.81), especially for context-aware information (avg=4, stdev=0.82). Finally, users generally agree with using M2A in the future, either alone, or as a complement to conventional AR web browsers. Users agree that they would use it frequently. Overall, the Perceived Ease Of Use is high with M2A, achieving our goal to make web browsing in AR convenient. Even though the Perceived Usefulness and Intention of Use show more variance, the overall response to M2A was strongly positive.

C. Main findings and discussion

These two experiments present the following findings:

1) Context-aware web browsing allows users to significantly improve the time to access relevant information while lowering their mental, physical and total effort, and their overall frustration (compared to smartphone web browsing).
2) M2A's flattened hierarchy with context-aware web browsing enhances the performance and perceived effort of users. This improvement is much more noticeable with the flattened hierarchy in AR browsing than with context-aware browsing.
3) The completion time for a given task is uncorrelated with the technological literacy when using M2A. However, it is correlated when using a smartphone or an AR browser.
4) Users find M2A useful and easy to use for browsing the web in AR, and intent to use it in the future.

Question	AVG	MED	MIN	MAX	STDEV	95%CONF
Perceived Usefulness (PU), $\alpha = 0.996$						
Using M2A would enable me to browse the web in AR more conveniently.	3.51	3	2	5	1.03	0.41
Using M2A would improve the way I look for information on the web.	3.22	3	1	5	0.94	0.37
I would find M2A useful when using an AR headset.	4.11	4	2	5	0.81	0.32
I find M2A useful for displaying context-aware information from websites.	4	4	3	5	0.82	0.32
Perceived Ease Of Use (PEOU), $\alpha = 0.997$						
Learning to use M2A would be easy.	4.62	5	3	5	0.88	0.35
I would find it easy to get M2A to do what I want to do.	3.70	3	3	5	0.78	0.31
It would be easy for me to become skillful at using M2A.	4.44	4	3	5	0.69	0.27
I find M2A easy to use.	4.26	4	3	5	0.70	0.27
Intention Of Use (IOU), $\alpha = 0.887$						
When M2A becomes available, I intend to use it for browsing the web in AR.	3.96	4	2	5	1.00	0.40
When M2A becomes available, I will use it in parallel to a conventional browser in AR.	3.77	3.5	3	5	0.89	0.35
When M2A becomes available, I predict I would use it frequently.	3.55	3	2	5	1.03	0.41

TABLE II: Technology Acceptance Survey. The PU, PEOU and IU score higher than average, with a low standard deviation. Users found M2A particularly easy to use (4.2) and useful for browsing the web with an AR headset (4.04). This translates to a high intention of use for browsing the web in AR (3.88)

Exploiting the full field of view of AR to display more content does not overload users with information. Users navigate easily on M2A websites, even when confronted with a new paradigm for the first time. User greatly benefit from the flattened structure of M2A. Specifically, users provide much less effort when browsing M2A websites compared to a traditional browser in AR. The technology acceptance is very high. Interacting with AR used to be a major concern. Users found M2A-transformed websites easy to interact with and showed little to no adaptation phase. Most users plan to use it for browsing the web on AR headsets in the future.

From a development point of view, M2A significantly simplifies the deployment of a website in AR. Besides the automatic adaptation of the basic elements, M2A builds on top of existing reactive design frameworks and enables developers to easily specify how to display content in AR In this paper, we introduce the LCSS language, that aims at being "the CSS for AR". This language allows displaying context-aware websites in a tri-dimensional space.

Limitations in the user panels: The sets of participants for both experiments are relatively homogeneous: most are male, with high technological literacy, between 20 and 35. However, this is the typical profile of current AR users. According to *Statistica* [32], 57% of current AR/VR device owner are male. In terms of purchase intention, this number grows to 69%. These users display the typical profile of early adopters, who are the most susceptible to experiment with AR and shape its evolution. Due to the small size of our sample, focusing on this subcategory of users is crucial as a first step towards the development of AR web applications. AR is indeed in its infancy, and web browsing in AR still presents multiple challenges. In this paper, we respond to two of these challenges, by adapting the display of websites to AR, while proposing the first draw at user interaction with 3D web content.

VII. Conclusion

In this paper, we present M2A, the first Mobile-to-AR framework. Similar to responsive design frameworks for mobile websites, this framework integrates the design of a context-aware AR website in the main website development workflow, with minimal modifications to the code. M2A handles the core AR operations such as object recognition, location detection, and interaction methods, and leaves the placement of the information blocks in the AR world to the developer.

By exploiting the third dimension provided by the AR space, M2A enables websites to display more content at the same time, while allowing users to pinpoint and isolate relevant information easily. As such, it significantly simplifies the user interaction flow and accelerates the access to information. Our experiments showed that participants using M2A found information 4 times faster than with a regular smartphone, and 2 times faster than a standard web browser displayed in the AR space. Moreover, users reported that M2A was less frustrating to use, and required less effort, both mental and physical. Regarding the Technology Acceptance Factor, users reported that M2A was easy to use and improved the way they look for information in general and more specifically in AR. Developers benefit from the low effort required to convert a website to AR, thanks to M2A Content Transform Engine and the Layout CSS style sheets.

This study provided a first comprehensive view of the users' expectations when browsing the web in AR. In future works, we plan to expand the framework to provide more accurate object detection and relate it to other sensor data such as the location of the user. We will use strategies such as offloading to improve object recognition, as well as provide additional rendering strategies to improve the layout of information on top of the physical world. We will also focus on the performance of M2A compared to other browsers.

VIII. Acknowledgements

The authors thank the anonymous reviewers for their insightful comments. This research has been supported, in part, by projects 26211515, 16214817, and G-HKUST604/16 from the Research Grants Council of Hong Kong, and the 5GEAR project from the Academy of Finland ICT 2023 programme.

REFERENCES

[1] The Conversation. A nostalgic journey through the evolution of web design, 2018.

[2] Dimitris Chatzopoulos and Pan Hui. Readme: A real-time recommendation system for mobile augmented reality ecosystems. In *Proceedings of the 24th ACM International Conference on Multimedia*, MM '16, pages 312–316, New York, NY, USA, 2016. ACM.

[3] Fengfeng Ke and Yu-Chang Hsu. Mobile augmented-reality artifact creation as a component of mobile computer-supported collaborative learning. 2015.

[4] Jim Hahn. Mobile augmented reality applications for library services. 113, 09 2012.

[5] Blair MacIntyre, Alex Hill, Hafez Rouzati, Maribeth Gandy, and Brian Davidson. The argon ar web browser and standards-based ar application environment. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 65–74, Washington, DC, USA, 2011. IEEE Computer Society.

[6] Yu You and Ville-Veikko Mattila. Visualizing web mash-ups for in-situ vision-based mobile ar applications. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 413–414, New York, NY, USA, 2013. ACM.

[7] Felix Klein, Dmitri Rubinstein, Kristian Sons, Farshad Einabadi, Stephan Herhut, and Philipp Slusallek. Declarative ar and image processing on the web with xflow. In *Proceedings of the 18th International Conference on 3D Web Technology*, Web3D '13, pages 157–165, New York, NY, USA, 2013. ACM.

[8] Christoph Oberhofer, Jens Grubert, and Gerhard Reitmayr. Natural feature tracking in javascript. In *Proceedings of the 2012 IEEE Virtual Reality*, VR '12, pages 113–114, Washington, DC, USA, 2012. IEEE Computer Society.

[9] Tobias Langlotz, Thanh Nguyen, Dieter Schmalstieg, and Raphael Grasset. Next-generation augmented reality browsers: rich, seamless, and adaptive. *Proceedings of the IEEE*, 102(2):155–169, 2014.

[10] Timo Engelke, Mario Becker, Harald Wuest, Jens Keil, and Arjan Kuijper. Mobilear browser–a generic architecture for rapid ar-multi-level development. *Expert Systems with Applications*, 40(7):2704–2714, 2013.

[11] T. LeppÃd'nen, A. Heikkinen, A. Karhu, E. Harjula, J. Riekki, and T. Koskela. Augmented reality web applications with mobile agents in the internet of things. In *2014 Eighth International Conference on Next Generation Mobile Apps, Services and Technologies*, pages 54–59, Sept 2014.

[12] Jens Grubert, Tobias Langlotz, and Raphael Grasset. *Augmented Reality Browser Survey*. ., 2012. Reportnr.: ICG-TR-1101.

[13] Ronald Azuma and Chris Furmanski. Evaluating label placement for augmented reality view management. In *Proceedings of the 2Nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '03, pages 66–, Washington, DC, USA, 2003. IEEE Computer Society.

[14] P. Dangkham. Mobile augmented reality on web-based for the tourism using html5. In *2018 International Conference on Information Networking (ICOIN)*, pages 482–485, Jan 2018.

[15] Markus Tatzgern, Valeria Orso, Denis Kalkofen, Giulio Jacucci, Luciano Gamberini, and Dieter Schmalstieg. *Adaptive Information Density for Augmented Reality Displays*, pages 83–92. IEEE Computer Society, United States, 3 2016.

[16] Statista. Forecast unit shipments of augmented (ar) and virtual reality (vr) headsets from 2016 to 2022 (in millions), 2018.

[17] L. Lee and P. Hui. Interaction methods for smart glasses: A survey. *IEEE Access*, 6:28712–28732, 2018.

[18] Yineng Chen, Xiaojun Su, Feng Tian, Jin Huang, Xiaolong (Luke) Zhang, Guozhong Dai, and Hongan Wang. Pactolus: A method for mid-air gesture segmentation within emg. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '16, pages 1760–1765, New York, NY, USA, 2016. ACM.

[19] Juan David Hincapié-Ramos, Xiang Guo, Paymahn Moghadasian, and Pourang Irani. Consumed endurance: A metric to quantify arm fatigue of mid-air interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1063–1072, New York, NY, USA, 2014. ACM.

[20] Zhanpeng Huang, Weikai Li, Pan Hui, and Christoph Peylo. Cloudridar: A cloud-based architecture for mobile augmented reality. In *Proceedings of the 2014 Workshop on Mobile Augmented Reality and Robotic Technology-based Systems*, MARS '14, pages 29–34, New York, NY, USA, 2014. ACM.

[21] Ricky X. F. Chen. A brief introduction on shannon's information theory. *CoRR*, abs/1612.09316, 2016.

[22] Heiko Drewes. Only one fitts' law formula please! In *CHI '10 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '10, pages 2813–2822, New York, NY, USA, 2010. ACM.

[23] Steve Krug. *Don'T Make Me Think, Revisited: A Common Sense Approach to Web Usability*. New Riders Publishing, Thousand Oaks, CA, USA, 3rd edition, 2014.

[24] Keshab R. Acharya. User value and usability in technical communication: A value-proposition design model. *Commun. Des. Q. Rev*, 4(3):26–34, March 2017.

[25] Google Inc. Google knowledge graph search api, 2018.

[26] Oliver Bimber and Ramesh Raskar. *Spatial Augmented Reality: Merging Real and Virtual Worlds*. A. K. Peters, Ltd., Natick, MA, USA, 2005.

[27] WebKit. The webkit open source project. 2011.

[28] Server-Side Protection Against Third Party. Control what you include! In *Engineering Secure Software and Systems: 9th International Symposium, ESSoS 2017, Bonn, Germany, July 3-5, 2017, Proceedings*, volume 10379, page 115. Springer, 2017.

[29] Mark Graham, Matthew Zook, and Andrew Boulton. Augmented reality in urban places: contested content and the duplicity of code. *Transactions of the Institute of British Geographers*, 38(3):464–479, 2013.

[30] Kong Ruan and Hong Jeong. An augmented reality system using qr code as marker in android smartphone, 05 2012.

[31] Sandra G Hart and Lowell E Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. *Human mental workload*, 1(3):139–183, 1988.

[32] Statistica. Vr and ar ownership by gender 2017, 2017.