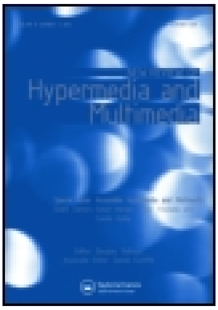
This article was downloaded by: [Sangchul Ahn]
On: 10 February 2014, At: 12:48
Publisher: Taylor & Francis
Informa Ltd Registered in England and Wales Registered Number: 1072954 Registered office: Mortimer House, 37-41 Mortimer Street, London W1T 3JH, UK



New Review of Hypermedia and Multimedia
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tham20

Webizing mobile augmented reality content

Sangchul Ahn[a][b], Heedong Ko[b] & Byounghyun Yoo[b]

[a] HCI & Robotics, University of Science and Technology, Daejeon, Korea

[b] Imaging Media Research Center, Korea Institute of Science and Technology, Seoul, Korea

Published online: 05 Feb 2014.

To cite this article: Sangchul Ahn, Heedong Ko & Byounghyun Yoo , New Review of Hypermedia and Multimedia (2014): Webizing mobile augmented reality content, New Review of Hypermedia and Multimedia, DOI: 10.1080/13614568.2013.857727

To link to this article: http://dx.doi.org/10.1080/13614568.2013.857727

PLEASE SCROLL DOWN FOR ARTICLE

Taylor & Francis makes every effort to ensure the accuracy of all the information (the “Content”) contained in the publications on our platform. However, Taylor & Francis, our agents, and our licensors make no representations or warranties whatsoever as to the accuracy, completeness, or suitability for any purpose of the Content. Any opinions and views expressed in this publication are the opinions and views of the authors, and are not the views of or endorsed by Taylor & Francis. The accuracy of the Content should not be relied upon and should be independently verified with primary sources of information. Taylor and Francis shall not be liable for any losses, actions, claims, proceedings, demands, costs, expenses, damages, and other liabilities whatsoever or howsoever caused arising directly or indirectly in connection with, in relation to or arising out of the use of the Content.

This article may be used for research, teaching, and private study purposes. Any substantial or systematic reproduction, redistribution, reselling, loan, sub-licensing, systematic supply, or distribution in any form to anyone is expressly forbidden. Terms & Conditions of access and use can be found at http://www.tandfonline.com/page/terms-and-conditions

New Review of Hypermedia and Multimedia, 2014
http://dx.doi.org/10.1080/13614568.2013.857727

Taylor & Francis
Taylor & Francis Group

Webizing mobile augmented reality content

SANGCHUL AHN[†‡], HEEDONG KO[‡] and BYOUNGHYUN YOO[‡*]
[†]HCI & Robotics, University of Science and Technology, Daejeon, Korea
[‡]Imaging Media Research Center, Korea Institute of Science and Technology, Seoul, Korea

(Received 17 March 2013; final version received 17 October 2013)

This paper presents a content structure for building mobile augmented reality (AR) applications in HTML5 to achieve a clean separation of the mobile AR content and the application logic for scaling as on the Web. We propose that the content structure contains the physical world as well as virtual assets for mobile AR applications as document object model (DOM) elements and that their behaviour and user interactions are controlled through DOM events by representing objects and places with a uniform resource identifier. Our content structure enables mobile AR applications to be seamlessly developed as normal HTML documents under the current Web eco-system.

Keywords: Augmented reality; Cyber physical system; Hypermedia; Mobile computing; Web of things

1. Introduction

Recently, the widespread use of mobile computing has given prominence to the situated interaction between people, places and things (Kindberg *et al.* 2002). Augmented reality (AR), which enhances users' perceptions of their surroundings by combining the physical world with complementary virtual worlds, is an effective way of presenting hyperlinked real and virtual objects (Guven and Feiner 2003). Most mobile AR applications that have been developed over the last two decades can be classified into two categories: (1) location-based AR, which is primarily designed for large-scale outdoor applications mostly using location and orientation information without or with only limited object recognition and image tracking technologies and (2) nonlocation-based AR (e.g. recognition-based AR and tracking-based AR), which is widely used for small-scale indoor applications on a table top or in a room environment with a handful of objects. In location-based AR applications, it is relatively easy to be aware of the situation because the geospatial condition is determined by positioning sensors such as a global positioning system (GPS). Moreover, the information model of points of interest (POIs) can be represented in a standardised form. Therefore, it is relatively easy to

---

*Corresponding author. Email: yoo@byoo.net develop location-based AR applications since existing POI data (e.g. constructed for GIS) can be utilised. On the other hand, nonlocation-based AR applications use rather ad hoc methods to be aware of their situation. Furthermore, an AR application and certain AR content have been strongly coupled with their interaction/application logic. Consequently, it is difficult to separate the AR content, which is customised for a specific AR application, from the recognition methodology when using these applications. These characteristics of common mobile AR approaches cause difficulties in terms of reuse of AR content as well as interoperability of related applications. In addition, the scalability of a mobile AR eco-system has been limited.

To overcome the aforementioned issues and enable seamless mobile AR across location-based and nonlocation-based AR approaches, the AR content should be represented in a standard way regardless of the type of AR application. This necessitates a mobile AR framework as a content platform to guarantee the scalability of authoring AR content and applications.

This work is inspired by the Web architecture, the principles of which include separation of concerns, a generic interface, a self-descriptive syntax and the network effect (Metcalfe's Law) (Jacobs and Walsh 2004), thus enabling the World Wide Web to be highly scalable. Our principal idea is to employ the Web architecture to embrace our surrounding physical environment with the concept of resources in the Web architecture so as to be accessible on the Web. By mapping the elements of AR content to the Web information model, use of HTML can satisfy the requirements of the AR content specifications. Consequently, a physical object can be included using a hyperlink in HTML syntax like as if an image is externally referenced in an HTML document. Therefore, virtual objects, physical objects and users are uniformly situated in the physical space within the Web architecture.

This paper presents the details of the content structure proposed to build mobile AR content and applications in HTML5 by extending the concept of resources. Section 2 describes related work. Section 3 discusses the principle of "webizing" AR content while Section 4 gives a detailed explanation of the designed content structure. Section 5 presents the architecture and implementation of our prototype system. To validate the feasibility of our approach in facilitating cyber-physical augmentation, Section 6 presents a case study for webizing AR content using a physical sensor dataset.

2. Related work

Most mobile AR systems are location-based (Höllerer *et al*. 1999, Dahne and Karigiannis 2002, MacIntyre *et al*. 2011). They are aware of the user's location from positioning sensors and provide information related to nearby POIs. Layar (2013), which is one of the most popular location-based AR browsers (Wolde 2010), recently included support for vision-based tracking. The supply of POI data is not based on a standardised representation format, but on a server-centric API (Madden 2011). ARML (Lechner and Tripp 2010) and KARML (Hill *et al*. 2010) are location-based AR specifications, both of which extend KML (Wilson

2008), a location-based content standard created by the Open Geospatial Consortium (OGC). ARML is based on a subset of KML, excluding tags that are not directly related to AR. The extension is more modest and focused on adding markup extensions to support specific browser features, such as “wikitude: thumbnail” and “ar:provider”. On the other hand, KARML adds simple high-level user interface (UI) components to the KML specifications, for example, “karml: Balloon” and “karml:Label.” Although KML and other geolocation-based approaches have an advantage in that they enable direct reuse of POI data from existing GIS systems, the dilution of precision from actual positioning sensors (e.g. GPS) is not acceptable for human-scale use. Moreover, using geocentric coordinates (e.g. latitude and longitude) in an augmentation model is not intuitive in situation-aware AR. Our approach is to make use of the existing HTML standard by extending the use of uniform resource identifiers (URIs) to identify and reference physical objects and places. Any HTML document object model (DOM) element can be linked to these physical URIs, and the DOM event mechanism can be handled by a JavaScript layer considering behaviour and user interaction. ARML 2.0 is the latest AR specification published by the OGC as a draft standard. It defines a GML-based new XML schema that differs from the previous version that was a subset of KML. It introduces Feature, VisualAssets and Anchor representing a POI, virtual objects and registration, respectively. ARML 2.0 supports both location-based and vision-based AR by composing a Trackable or a Geometry, inherited from ARAnchor, to KML feature. However, this method merely integrates selecting the recognition modules separately, and does not provide the fundamental unification of spatial representation.

There are several commercial solutions that rely on HTML5-based AR content platforms [e.g. Junaio (Metaio Inc. 2013b) and ARchitect (Wikitude Gmbh 2013)]. Junaio provides augmented reality experience language (AREL) (Metaio Inc. 2013a) in XML for static content definitions and a JavaScript API for application logic. An HTML document is used to bridge the XML content definitions and the JavaScript application logic. It also provides latitude/longitude/altitude markers (LLA Markers) for marker-based positioning and Junaio GLUE for vision-based tracking. ARchitect is another implementation of a JavaScript AR SDK by Mobilizy, providing support for both location-based and vision-based AR. The content structure of ARchitect does not rely on HTML. For this reason, the JavaScript application logic of ARchitect does not use the DOM API. This procedural approach makes AR content difficult to intermix and interoperate with standard HTML content and complicates their interaction through the DOM event-handling mechanism. Whereas these solutions use the HTTP protocol only to transmit their own content specifications or provide a JavaScript API without considering the DOM, our approach reconstructs the AR content structure based on HTML and the selection/binding mechanism of cascading style sheets (CSS) with the associated JavaScript event handling of DOM events.

As mentioned above, there are many AR content specifications such as AREL, ARML 1.0, KARML, ARML 2.0, Layar JSON and ARchitect. The content structure of each AR markup language is compared in Table 1. Our work differs from existing approaches in terms of the base document structure, registration

Table 1. Comparison of AR markup languages.

AR markup language	Base document structure	Registration model	Target model	Available virtual object	Available rendering of HTML parts
AREL	XML	Location-based	location	2D image and 3D model	HTML overlay on a screen
		Vision-based[a] (with GLUE)	trackingXML		
ARML 1.0	KML	Location-based	Placemark	2D text and icon	Unavailable
ARML 2.0	XML (GML)	Location-based	Geometry Ancnor	2D object and 3D model	Embed HTML page in 2D objects
		Vision-based[a]	Trackable Anchor		
KARML	KML	Location-based	Placemark	2D object and 3D model	Embed HTML page in 2D objects
		Vision-based[a]	Tracker		
Layar JSON	JSON	Location-based	geolocation	2D object and 3D model	Unavailable
		Vision-based[a]	referenceimage		
Architect	JavaScript API	Location-based	GeoLocation	2D object and 3D model	Embed HTML page in 2D objects
		Vision-based[a]	Trackable		
Webized AR	HTML	Unified[b]	-ar-target[c]	Any HTML elements	Blended in the DOM of a base document

[a] Location-based and vision-based registration models are separately represented and are not concurrently available.
[b] A unified representation is available by abstraction of spatial representations of location-based and vision-based registration models.
[c] In the webized AR, the target model is unified using the place-based location model with -ar-target, for both place and object instead of providing separated models for different tracking methods.

model, target description, available virtual object and available rendering of HTML documents. In most existing specifications, a document contains physical world models by themselves. Our content structure regards physical entities as a Web resource (an entity that is separated from the document and identified by a URI). Some AR markup languages can organise physical descriptors using an HTTP URI for use in the vision-based registration model. However, this rather fragments the content structures depending on the registration model. On the other hand, we propose a content structure that enables a unified representation by abstraction of spatial representations of location-based and vision-based registration models.

Although X3DOM (Behr *et al.* 2011) is not a mobile AR model, it also attempts to integrate X3D scene elements seamlessly as HTML DOM elements and events, as in our approach. However, it does not have the same notion of physical URIs as objects and POIs for in situ AR or the notion of a location context. In our approach, location-based AR content can be intermixed within regular HTML documents by augmenting any HTML DOM element with its location context through the extension of CSS3.

3. Webizing methodology for AR content

3.1. AR content requirements

AR content should deal with the physical world as well as the virtual world. To achieve a unified representation for these heterogeneous environments, we describe the requirements for content structure for AR in terms of the webizing methodology (Berners-Lee 1998). Table 2 summarises the comparison of information models for AR and Web applications based on the characteristics of their targets, enriched with add-on content, and the characteristics of the content, which is added to the target. Whereas a location, which is the unit entity of a target resource in location-based AR, is distinguished by an absolute location model based on geographical coordinates, an object or space, which is the unit entity of a target resource in nonlocation-based AR, is distinguished by a relative location model based on the local coordinates defining the independent space. For intermixed as well as unified representations of the aforementioned heterogeneous components, AR content should satisfy the following requirements:

- An infrastructure for the supply and management of objects and places in a physical world should be provided.
- It should be easy to create augmenting content (known as virtual objects) using various types of rich multimedia.

Table 2. Extended information model conforming to the current Web architecture to embrace AR content specifications.

	AR		Web	
	Location-based AR	Non-location-based AR	Common Web	Webized AR
Target	Location	Object/Space	Resource	Resource
Target type	Physical	Physical	Virtual	Physical/Virtual
Target identifier	Coordinate	Feature (e.g. marker)	URI	URI
Target information	Positioning sensors (e.g. GPS)	Object-recognition/ Image-tracking	CSS	Target physical entity + CSS
Content	Location model (e.g. POI)	Object model	HTML element	HTML element
Content type	Virtual	Virtual	Hypermedia	Hypermedia
Coordination model	Absolute location model	Relative location model	Relative location model	Relative location model
Coverage	Large space (i.e. outdoor)	Small space (e.g. room, table top)	Cyber space	Situated space (cyber-physical space)

- Sharing and reusing AR content, including a physical world model (e.g. POI) as well as virtual objects should be supported.
- A unified information model should be consistently applicable to both location-based and nonlocation-based AR.
- The information model should accurately coordinate dynamically moving objects as well as static objects during the augmentation process.
- Playback of AR content should be possible wherever the user is located to support mobility.

In particular, the unified information model should represent and embrace the characteristics of both location-based and nonlocation-based AR. For example, the absolute location model used in the coordination model of location-based AR in Table 2 can be represented using a relative location model in a generalised manner. In addition to these requirements, for the best use of the Web architecture, a webized AR information model should guarantee that it does not violate the current Web content eco-system.

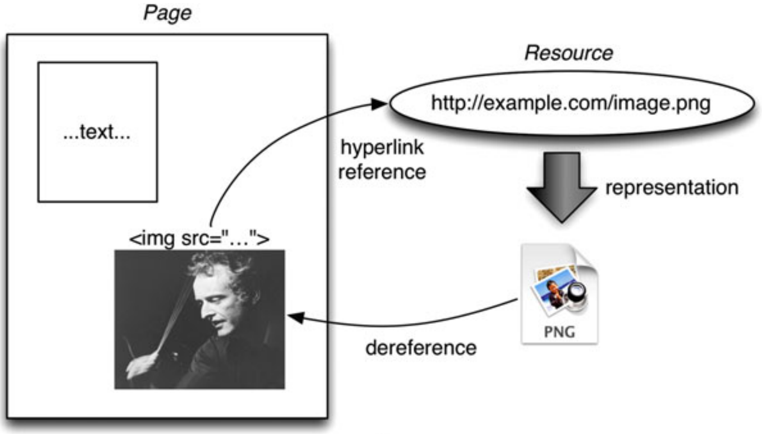
3.2. Webized AR information model

HTML is widely used to represent information in the modern Internet system. The main reasons for this are the simplicity of its content structure, the modularity of the system and decentralisation of the information source (Berners-Lee 1996). In the architectural view, Web has the potential to relate the virtual world to the physical world. In the Web information model, a resource is a primitive where a hyperlink can reach, and the hyperlink is targeted by a URI. A URI does not contain any actual data when it is referenced but it provides representations, as a response to a Web browser. As the concept of resource evolved, a resource can be anything in the real world, including the physical entities such as objects and places. The rightmost column in Table 2 shows a webized AR information model that embraces the content structure of AR and the common Web to satisfy the requirements discussed in Section 3.1.

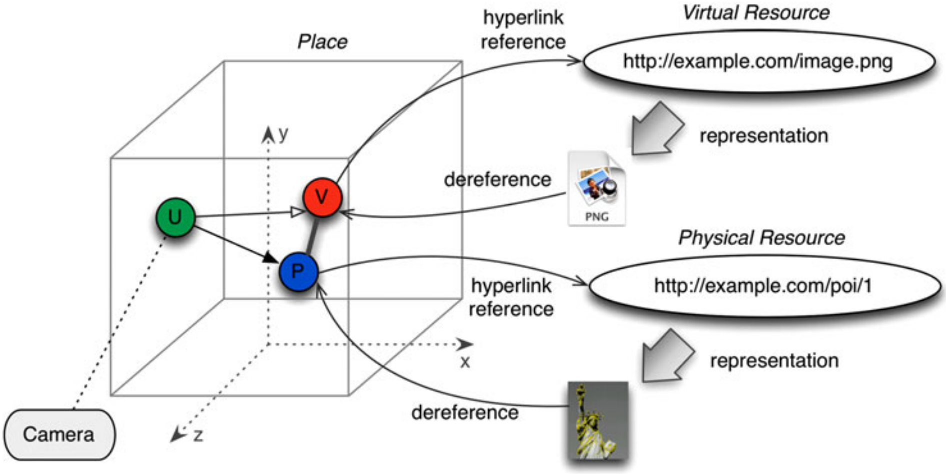
3.3. AR content in HTML

HTML also has the potential to be a container for AR content with physical resources. A hyperlink enables HTML document to embrace the physical resources. There are two types of hyperlinks in HTML according to the referencing method in terms of using its attribute (Berjon *et al.* 2013): (1) *src* attribute (as used in IMG and INPUT elements) accompanies dereferencing of linked resource and (2) *href* attribute (as used in A and LINK elements) leads a user agent to move to the linked destination. Physical resources can be directly embedded into an HTML document using the former use case; the linked resources are immediately acquired when the document is rendered. We use this approach, whereas a user agent has to navigate to reach linked resources in the latter use case, which has been used in existing approaches shown in Table 1.

Figure 1 demonstrates how the webized AR content is represented while the extended use of the resource model for physical environment conforms to the current Web architecture. An instance of common HTML document dereferencing a hyperlinked information resource is shown in Figure 1(a). Dereferencing physical resource in an HTML document is commonly available (Figure 1b). The referencing method of physical resources differs from a generic HTML document because a physical resource does not belong to an HTML element. To avoid conflict over the current web standard, we do not add a new HTML element or attribute to deal with physical resources. Instead, we utilise CSS, the separated presentation logic of HTML, for referencing physical resources. It conveys the context from the physical environment to the application by adding a style sheet



(a)



(b)

Figure 1. Conceptual information model of webized AR content from a data-flow perspective: (a) dereferencing hyperlinked information resources of common Web content; and (b) situating three different resources [virtual (V), physical (P) and user (U)] by dereferencing hyperlinked virtual and physical resources of webized AR content.

without modifying the document content. Consequently it enables reuse of numerous HTML contents in the Web content eco-system. We have created a CSS property, -ar-target, which stores the URIs of objects and places. The detailed mechanism for referencing and dereferencing physical resources using the CSS rule is described in Section 4.3.

3.4. Situated HTML rendering

In AR systems, sharing common spatiotemporal context among user, physical objects and virtual objects is important for the registration of them. While the user is moving, the physical environment surrounding the user is being continuously changed. Because the living environment, especially the indoor environment, is partitioned into smaller subspaces (e.g. a room), the partitioning of space has been often regarded as the obstacle that restricts the service area of positioning systems in AR.

Each subspace can be viewed as a unit medium to manage the commonly shared context among user, physical objects and virtual objects. If the model of physical environment can be dynamically updated across subspaces, and the contents can be located and rendered in the local space for every subspace where the user is situated, the spatial coverage of an AR application can be enlarged; then eventually the mobility of system is secured. To address this, we propose the concept of place as a unit of partitioned space and introduce use of place as a rendering media. Whenever a mobile user enters an adjacent place from a place, it is detected by continuous containment checking between the boundary of places and the user's location. Then, a mobile AR browser gets the environment model of the new place, and all rendering information is transitioned to the place and integrated into the new place model.

HTML has a potential to support the concept of place. The space model of typical Web content is a continuous page model. When a document is rendered, HTML elements are laid on a 2D plain page. The page is displayed as a plane on a window of a GUI desktop in a display. However, this planar space on a page is not appropriate to situated environments for mobile AR. Rendering a HTML document as AR content should be rendered in 3D space, which is physically situated to a place, rather than on a 2D page. CSS 2.1 introduced the "media type", an extensible method to define diverse rendering modes such as aural, braille, embossed, screen and print. It enables Web browsers to support various rendering types. We have defined the place media, as new CSS media type, to support rendering of a HTML document as a 3D volumetric medium in each place. The detailed explanation of the place media is given in Section 4.2.

4. Content structure

This section describes what is extended for AR from the common Web to represent physical entities in HTML. Before the components are explained in detail, we show the overall content structure in Figure 2.

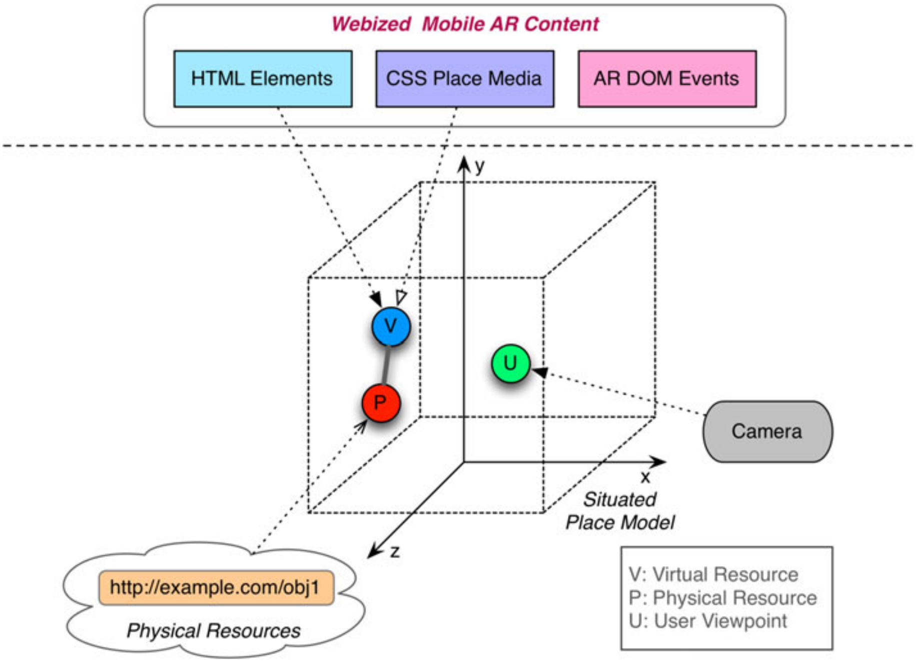
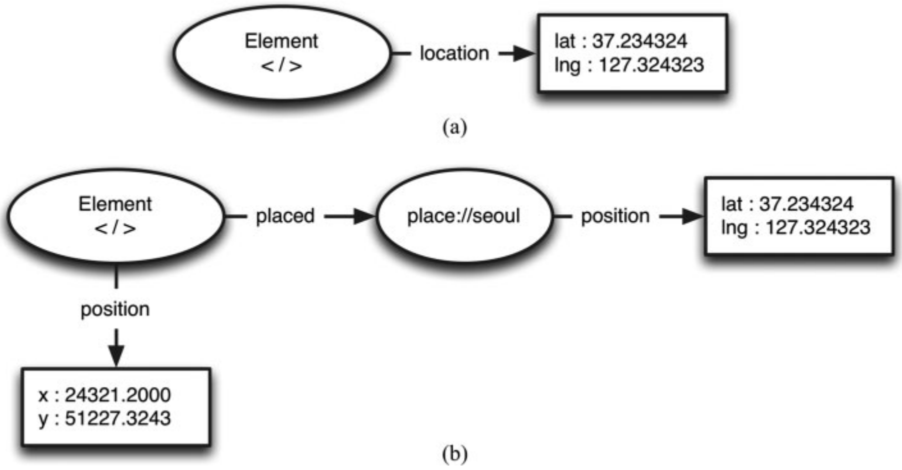


Figure 2. Content structure of the webized AR. The heterogeneous elements, namely, the virtual resource, the physical resource and the user, are situated in a “place”.

4.1. Representation of the real world

Many existing AR specifications, such as ARML (Lechner and Tripp 2010) and KARML (Hill *et al*. 2010), use geocentric coordinates as their coordination model to augment a physical space with virtual objects. Such a location-based AR representation uses explicit information to coordinate virtual objects at any specific location in the world, as shown in Figure 3(a). However, this model is not applicable to relatively dynamic augmentation processes that involve attachment to an object (e.g. a car or table top). We define the concept of “place” and propose our own relative location model based on the relationship between an object and a place.

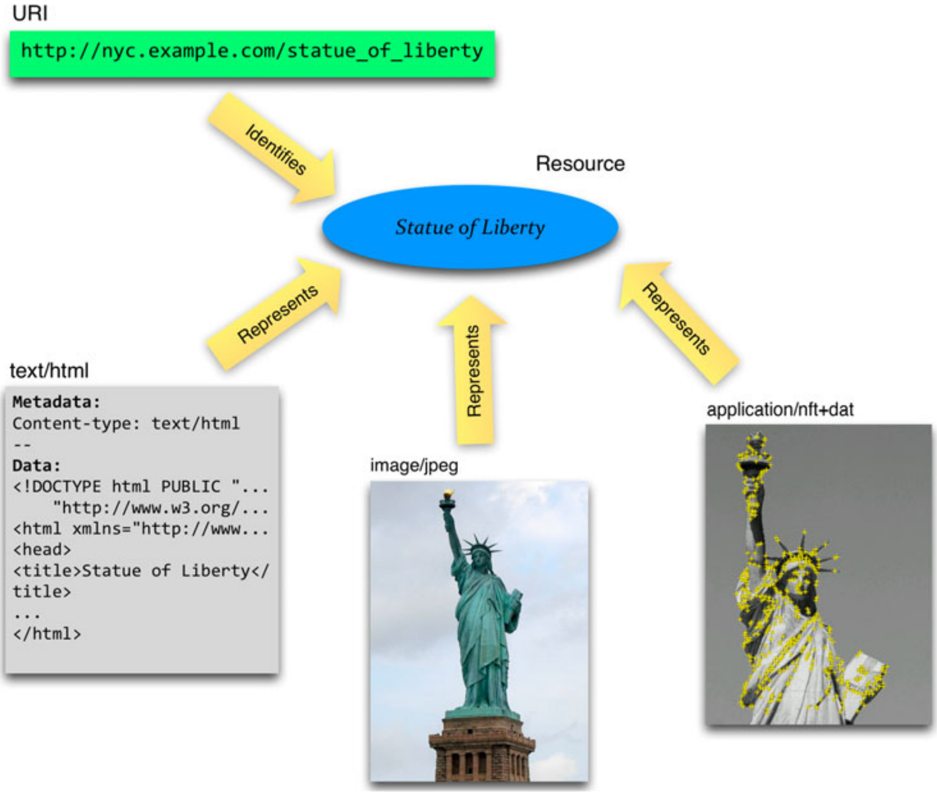
4.1.1. Physical resource. In our model, a physical resource refers to a physical entity as an extension of a Web resource. Therefore, each physical entity has its own HTTP URI and can be included in an HTML document using a hyperlink, as shown in Figure 4. As a physical resource, the representation of physical entities contains information with which they can recognise themselves. An object description is composed of its *location and descriptors*. A descriptor represents the physical recognition method and its feature descriptor of the object. It is not used when rendering the document, but instead to detect objects with image recognition processes by browsers. It requires descriptors suited to the recognition method. For example, the recognition method can be an image-tracking tool such as Vuforia (Qualcomm 2013) or a sensor system such as radio-frequency



(a)

(b)

Figure 3. Location model for webizing AR content: (a) location-based AR representation; and (b) place-based AR representation.



```
Metadata:
Content-type: text/html
--
Data:
<!DOCTYPE html PUBLIC "...
    "http://www.w3.org/...
<html xmlns="http://www...
<head>
<title>Statue of Liberty</
title>
...
</html>
```

Figure 4. Physical entity as a Web resource.

identification (RFID). A location consists of a combination of a place identifier and the position of the place. The object description is represented in the resource description framework and is encoded to JSON-LD.

The acquisition of an object description and a feature descriptor follows the content negotiation of HTTP 1.1 in accordance with representational state transfer (REST). If one (or more) tag attribute or CSS property, with a target object as its value, is found in the document loaded, the browser obtains metadata pertaining to the object from its URI by sending an HTTP request message containing the "application/ld+json" MIME type in the Accept header. The browser obtains a feature descriptor for natural feature tracking in a similar way. The feature descriptor is obtained by sending an HTTP request message with a new MIME type,"application/nft+dat", in the Accept header; we defined this MIME type, since no MIME type for a feature descriptor existed.

4.1.2. Place. A "place" is an extended representation of an object that can contain other object locations. The place is regarded as an object when the user is outside. However, it forms a spatial environment when the user enters it (e.g. a building, room, and the earth).

In addition to the default properties of a physical entity, we define several environmental properties, including *boundary*, the local coordinate reference system (*LCRS*), and *local map*. *The boundary* is the geometry separating the inside/outside of the place. The subspace of a place is defined using an *LCRS* description, which is represented by a simplified schema of the coordinate reference system's package model in geography markup language. Each place has a *local map* indicating the accurate location of internal physical entities in detail. The map information consists of the URIs of the physical entities, as well as their positions and orientation. The position coordinate is set to the LCRS of the place, and its orientation is represented in a clockwise rotation from magnetic north. With this extension, a space-partitioning tree of the physical world can be constructed recursively. The scope is dynamically determined by the user location.

4.1.3. Place-based location model. To locate the contents over the partitioned spaces on a human scale, we propose a place-based location model. This is basically the relative location model using coordinates based on a local datum for each place, instead of a geocentric datum. The representation of place-based location consists of a place identifier and a local coordinate, as shown in Figure 3(b).

According to the mutual inclusion relationships between multiple places, the place-based location model can recursively represent relative locations at multiple levels. The location of a physical resource can be specified recursively as a unified representation regardless of the granularity of the resource (i.e. object or place) by including a coordination model of a DOM element (i.e. augmenting content) and the relationship between the higher level place and the current place.

4.2. CSS place media

This section describes a novel media type for presenting an HTML document in a physical space. Figure 5 shows the webized AR content from the perspective of

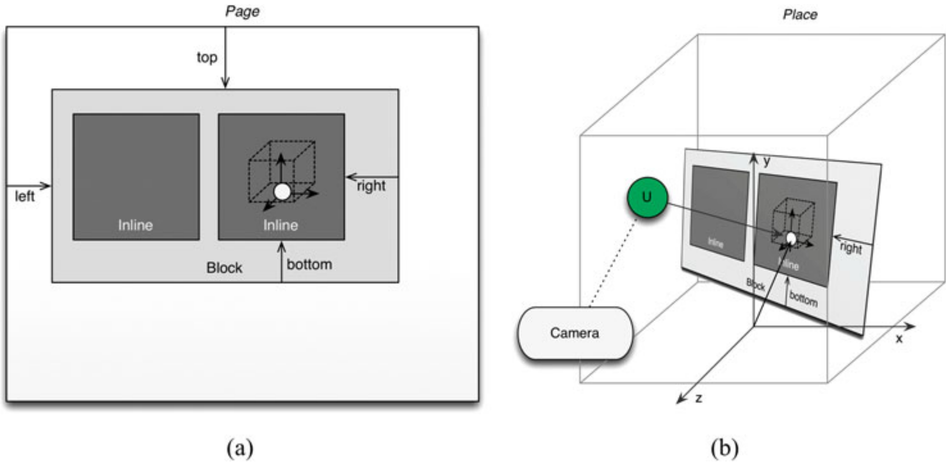


Figure 5. Representation of HTML elements from the perspective of rendering webized AR content: (a) common Web; and (b) webized AR.

rendering. The presentation of the webized AR content in terms of the rendering of an HTML document is quite different from that of typical Web content. Typical Web browsers create a virtual plane (i.e. a page) and lay out DOM elements by coordinating a block at a specific location using relative coordinates (i.e. top, bottom, left and right) as shown in Figure 5(a). However, webized AR content is rendered in a different manner. First, an HTML element representing webized AR content can be coordinated by means of an existing method precisely in the same manner as typical Web content. Second, an HTML element can be coordinated at a specific location in a physical space. Third, an HTML element can also be coordinated through a relative location from a physical object that is coordinated at a specific location. To embrace all three cases, we extended the CSS media type and added "place media" instead of using the CSS 2D box model, which is only suitable for screen space, such that the coordination is applicable to 3D content. As shown in Figure 5(b), webized AR content can be located in a physical space using the CSS place media.

A mobile user moves dynamically through multiple levels of places; thus, we define a local coordinate system according to each space and systematically manage the relationship between places and the transformation hierarchy.

4.3. CSS augmentation model

In our approach, CSS is used to augment HTML elements onto the physical environment. The current CSS specification provides a plain box model to lay out HTML elements on a page. The CSS box model is composed of the positions of elements (i.e. top, bottom, left and right). We propose a CSS augmentation model that extends CSS to define our location model within the standard specification of CSS. The CSS augmentation model defines a bounding box of a DOM element and specifies the coordinates of the space that fills the box using the relative location from the boundary of the box.

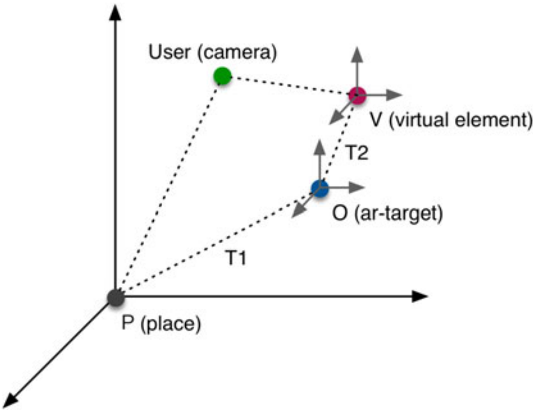


Figure 6. Augmentation model using the properties defined in the CSS place media.

The augmentation process is as follows. First, we define a target object to situate. Then, the scope of the space is constrained by the object (i.e. the place). Thereafter, we compute the specific location of an element using the coordinate system described in the place model and the relative position coordinate described in the DOM element.

To provide information about this augmentation process, we add a new CSS property, *-ar-target*. The new property is designed to harmonise with existing CSS properties. The augmentation model using the properties defined in the CSS place media is depicted in Figure 6. The *-ar-target* property is used to specify the target object. The property allows the positioning of an element to move responsively according to the movement of the target object. For example, in Figure 6, an element V (virtual element) is coordinated by the combination of the relative location T2 from the target object (O [ar-target]) and the relative location T1 of the target object from the datum of a place P. The property has a functional notation that distinguishes an object from a place. If the target is considered to be a normal object, the object is located at the same position as the target, allowing them to coexist. If the target is considered to be a place, the element is located according to the separate coordinate system described in the place. A situated element can also be transformed by means of CSS3D transformation.

4.4. Events for situated interaction

The DOM level-2 event specification (Miller *et al.* 2010) defines a primitive interface to generate, handle and register event listeners. By default, five UI events have been defined: *FocusEvent*, *MouseEvent*, *KeyboardEvent*, *WheelEvent* and *CompositionEvent*. *TouchEvent* has recently been added to support mobile devices. To address the requirements for situated interactions, we created three extended UI events: *PlaceEvent*, *ObjectEvent* and *TrackerEvent*. Figure 7 shows the extension of DOM level-2 events for situated interactions.

PlaceEvent occurs when the movement of the user changes the place. During a situated interaction, the environment context changes when the user moves from

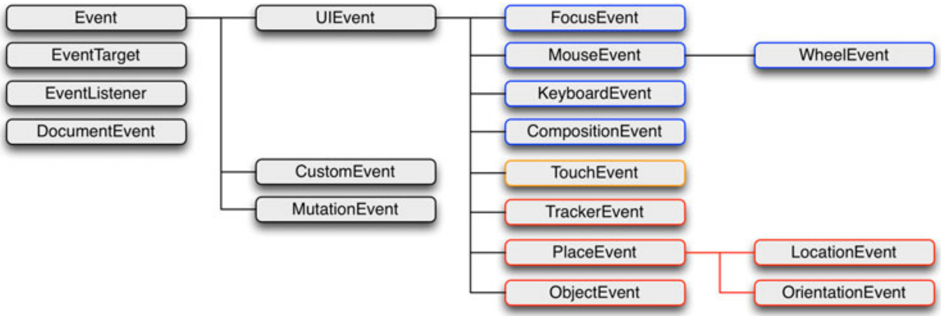


Figure 7. Extension of DOM level-2 events for situated interactions.

one place to another. PlaceEvent contains a place identifier, a timestamp, an LCRS and a local map URI. It encompasses three types of events: *placein*, *placeout* and *placechanged*.

ObjectEvent is generated when the state of the object is changed in the place. It has three event types: *objectin*, *objectout* and *objectmove*. Typically *ObjectEvents* are triggered in the following cases. First, *objectin* events are triggered when the scene is initialised with a local map. If a *placein* event is triggered by the entrance of a user, a local map is automatically obtained from the URI in the place description. When the local map is parsed, *objectin* events are generated. Second, *objectin* events are triggered if the new object is registered and starts to be managed by a scene. When the object moves, *objectmove* events occur with the updated position coordinates. Finally, an *objectout* event is triggered simultaneously for each object in a place when the user leaves the place. An ObjectEvent includes an object identifier, timestamp, position and orientation.

TrackerEvent is created when the target POIs enter or leave the view frustum. It is motivated by *MouseEvent*, one of the most frequently used UI events. Like *mousein*, *mouseout* and *mouseover* events, which are fired according to the spatial relationship between the pointer and the visible HTML elements, TrackerEvent is created when virtual and physical objects enter or leave the view frustum. A *TrackerEvent* instance contains a URI, its position and a timestamp. It also has three event types: *trackerin*, *trackerout* and *trackermove*.

5. Implementation

To demonstrate the feasibility of the proposed content structure, we developed a mobile AR Web browser, called Insight, as a reference implementation. We first explain how the mobile AR Web browser works, and then describe simple application cases based on the content structure.

5.1. Insight: a mobile AR Web browser

To validate the proposed content structure, we developed a prototype mobile AR Web browser for iOS and Android. Here we explain the different components thereof in contrast to typical page-based Web browsers. The prototype system

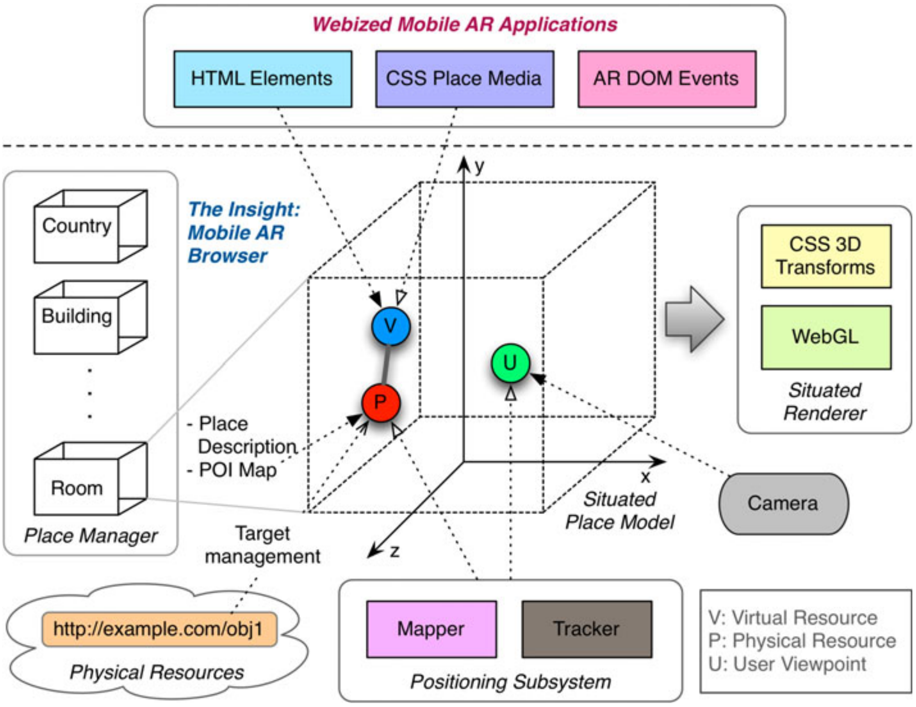


Figure 8. Architecture of the webized mobile AR application platform.

consists of four main components: a target manager, positioning subsystems, a place manager and a situated renderer. Figure 8 shows the system architecture of the webized mobile AR application platform and how these core components support webized AR content. The components provide the tightly related context information for performing the situated interactions. Figure 9 shows the system architecture of the AR Web browser implementation. The target manager selects the physical entities from style sheets and obtains the feature descriptors through REST access to their URIs. The types of feature samples are extensible to support various tracking methods. Currently, RFID and image recognition features are supported. The positioning subsystem determines the position and orientation of the user and each movable target object. Its main components are a *mapper* and a *tracker*. *Mappers* correlate the different location representations from diverse local positioning systems into the LCRS of the situated place. A *tracker* provides the pose estimation of target objects from the place's origin. *Trackers* include *WifiTracker*, *MotionTracker* and *ImageTracker*. *WifiTracker* uses the WiFi fingerprint-based positioning engine (LaMarca *et al*. 2005, Youssef and Agrawala 2008), *MotionTracker* uses the values of the gyroscope and compass of a mobile device and *ImageTracker* is an image-recognition–based tracker. We have two separate *ImageTracker* implementations, *KistNftImageTracker* and *VuforiaImageTracker* based on Vuforia (Qualcomm 2013).

The place manager provides the information about the place environment and the local map. The information is obtained from an external web service by

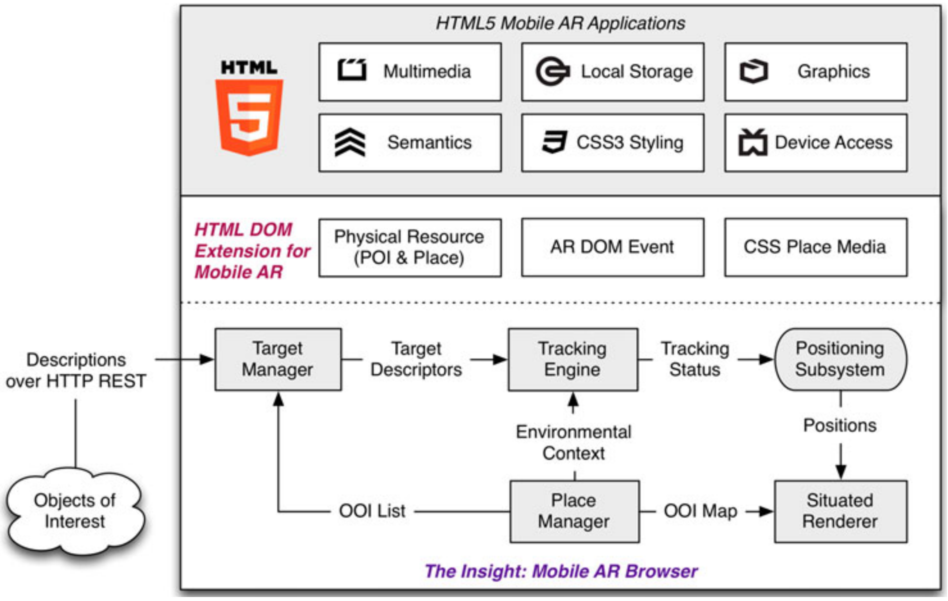


Figure 9. System architecture of the Insight Web browser.

resolving the place ID. The situated renderer manages the unified scene graph to combine physical entities, content and the user viewpoint into the situated place.

In a mobile AR Web browser, the runtime environment must be aware of the user's context (e.g. position, orientation and field of view) as well as the physical environment (e.g. object and place). This context is not directly accessible by generic Web browsers. To support these mobile-specific requirements, methods must be provided for the aggregation of sensor data (e.g. GPS, compass, gyroscope and camera), which are usually available by native functions, and to provide the information as a DOM event to the mobile Web application. In our implementation, Adobe PhoneGap (Adobe Systems Inc. 2013) is used as a hybrid mobile application platform that provides JavaScript APIs access to the device capabilities and the plug-in architecture to offer additional native functions.

5.2. Simple AR Web application cases

We tested various cases to develop a simple AR Web application to validate the feasibility of the proposed specifications. The sample AR application is explained using example code for the various parts. An update of the place model is a common operation for all applications. When the Insight browser is launched, it continuously detects the location of the user and the places surrounding the user. If the user enters a place, the browser detects the place ID and acquires a description of the place. We assume that a certain indoor location positioning system is available in a given area.

To begin with, the simplest "Hello, world!" example is shown in Figure 10. In this code, the *-ar-target* CSS property is applied to a <*p*> tag containing "Hello

```
<html>
  <head>
    <style type=“text/css” media=“place”>
      #hello {
        -ar-target: object(“http://example.org/poi/12”);
      }
    </style>
  </head>
  <body>
    <p id=“hello”>Hello World!</p>
  </body>
</html>
```

Figure 10. Hello world! AR example augmenting simple text on a given target.

world!” text by the CSS ID selector. The browser obtains the object description in JSON-LD from the URI of the object using the REST protocol. The acquired metadata contains the URI of physical descriptors that can physically recognise the target object, and the browser injects this into the internal tracking engine.

Figure 11, as a second example, depicts the augmentation of various media files. The three different types of elements, <img>, <video> and <svg>, are augmented onto the same target. As hypermedia, HTML has a modular design for handling diverse types of media.

As a third example, Figure 12 presents a combined use of CSS 3D transform and augmentation. CSS3 provides the transform module in both 2D and 3D. Because the specification is in a draft phase, the property name includes the browser-specific prefix *-webkit*. In this code, the application shows a rotating image with augmentation to a target.

The fourth example (see Figure 13) shows how to handle AR DOM events. This code uses the *addEventListener* methodology as introduced in the DOM Level 2. When a target appears in the capturing boundary and tracking starts, an anonymous function is called with a “*trackstart*” event as a callback parameter.

```
<html>
  <head>
    <style type=“text/css” media=“place”>
      .media {
        -ar-target: object(“http://example.org/poi/21”);
      }
    </style>
  </head>
  <body>
    <img class=“media” src=“gif_animation.gif” />
    <video class=“media” src=“html5_video.ogg” controls />
    <svg class=“media” xmlns=“http://www.w3.org/2000/svg” version=“1.1”>
      <g>
        <polygon points=“0,0 100,0 100,100 0,100” style=“fill:lime” />
      </g>
    </svg>
  </body>
</html>
```

Figure 11. Augmenting diverse types of media onto a given target: <img>, <video> and <svg> elements are augmented onto the same object target using the “media” class.

```
<html>
  <head>
    <style type=“text/css” media=“place”>
      #container {
        -ar-target: object(“http://example.org/poi/22”);
      }
      #image {
        -webkit-transform: translate3d(200px, 100px, 100px)
                           rotate3d(1,2,0,30deg)
                           scale(1.3);
      }
    </style>
  </head>
  <body>
    <div id=“container”>
      <img id=“image” src=“image.png” />
    </div>
  </body>
</html>
```

Figure 12. Augmentation of an animated image.

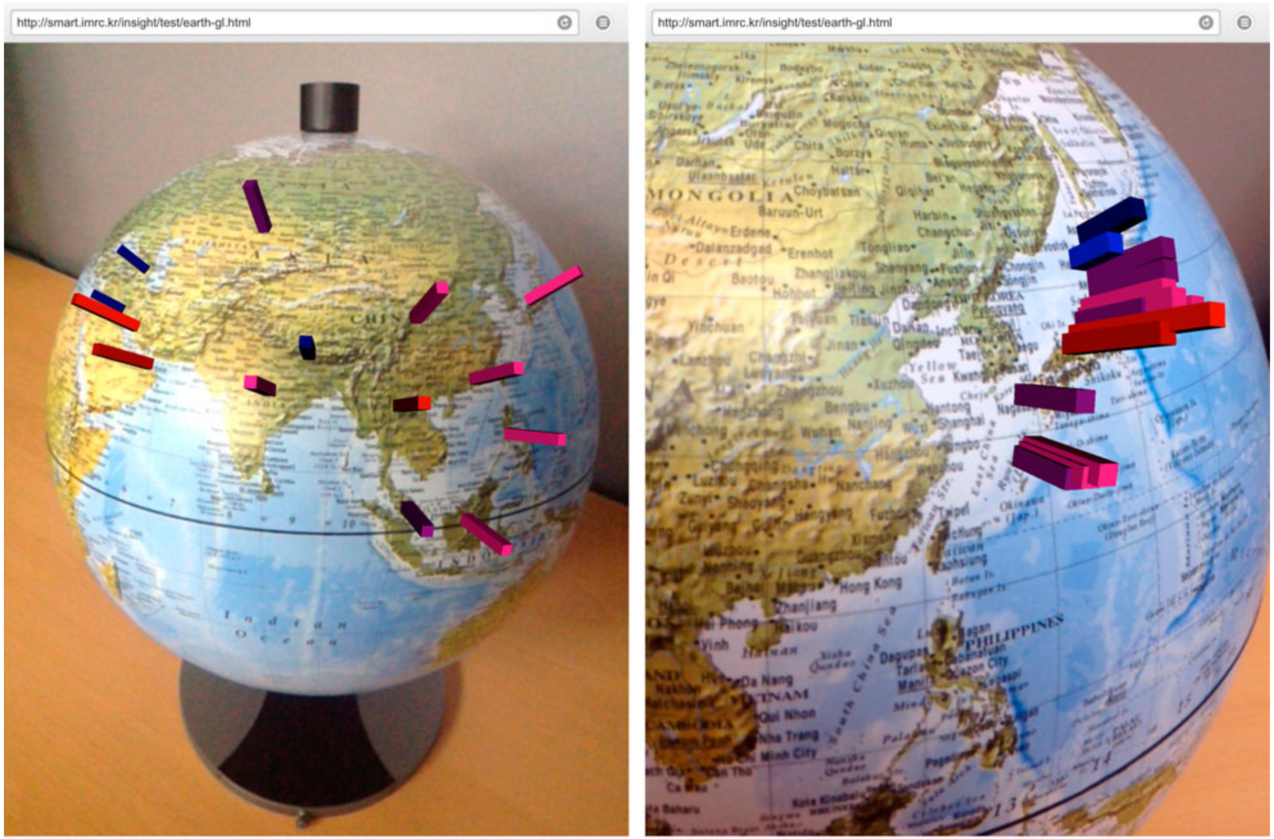
```
<script type=“text/javascript”>
  function initialize() {
    insight.addEventListener(
      “trackstart” // specify event type
      “http://example.org/poi/25”, // specify target uri
      function(event) {
        alert(“Track Start: ” + event.target);
        event.element.src = “image2.png”;
      }
    );
  };
</script>
```

Figure 13. Simple JavaScript code handling the “*TrackStart*” event.

6. Cyber-physical augmentation for a Sensor Web

As part of our validation process, to investigate how webized AR content facilitates cyber-physical augmentation, we applied our approach to a data exploration interface featuring AR for a sensor network and used it visually to explore weather stations. A total of 34,700 personal weather stations were collected from the Personal Weather Station project (Weather Underground 2012) and registered in our online database. A large number of weather stations are located in North America and Europe, but in general the stations are broadly scattered across the globe. The typical UI for an exploration system for global sensor data is a mashup of dynamic sensor data on a Web-based 3D globe. Whereas this type of interface augments a virtual globe in a cyber space (i.e. on the Web) using cyber data fed from the physical world, we augmented a physical globe with sensor data fed from physical sensor stations in cyber-physical space.

We used the prototype implementation explained in Section 5 for the cyber-physical augmentation. When a user selects sensor data (e.g. the temperature) in the Insight AR Web browser, our prototype system displays the average values of temperature recorded by sensors in the weather stations within each country's borders (Figure 14). A geospatial query that computes the average values of the



(a) (b)

Figure 14. Example of webized AR content using a physical sensor dataset: (a) a summarised visualisation of sensor data showing statistical temperature values from weather stations in each country; and (b) a local view of individual temperatures from weather stations in Japan.

temperature by country from registered weather stations is implicitly formulated within our system, and statistical dynamic 3D bar charts are generated and registered onto the physical globe. The 3D bar charts are dynamically added as AR DOM elements in the webized AR application platform (Figure 8). The physical globe is a place in our augmentation model using the CSS place media, and each country is a target object as shown in Figure 6. Then the 3D bar charts are added as virtual elements onto the physical globe using the augmentation model depicted in Figure 6. The Insight AR browser augments the physical globe in the real world by overlaying the dynamic 3D bar charts on the physical globe. The colour and the height scale of the bar chart in Figure 14(a) vary in proportion to the temperature. The 3D bar chart shows actual temperature values scaled to fit the physical globe, and the user can easily assess differences in the average temperatures between stations. If a user is interested in a specific region, for example, Japan, as shown in Figure 14(b), he/she simply moves his/her mobile device to focus on Japan on the physical globe. Accordingly, the dynamic 3D bar chart is automatically updated by a geospatial query, and our system augments Japan on the globe with the bar chart, which shows the temperature values generated by the weather stations located in Japan, as shown in Figure 14(b). This interface facilitates not only the augmentation of cyber-physical information in a unified space but also provides a more intuitive experience for users navigating the physical sensor network. This is possible using the proposed content structure, which does not violate the current Web architecture.

The physical globe is modelled as a place, which is visualised by a 3D sphere in WebGL using Three.js. For vision-based tracking of countries as targets, we extract feature descriptors by cropping segments of a picture of the physical globe as a planar image target. The centroid of the descriptor is mapped to the geospatial coordinate of the centroid of the correlated target object. The overall workflow of this system is as follows. If the physical globe appears in the field of view of the camera, a *placein* event containing the URI of the physical globe is triggered. Moreover, *objectin* events of Asia and Japan, which are included in the physical globe, are triggered. *ObjectEvents* at different granularity levels are triggered by the following rules. Three levels, namely, earth, continental region and country, have been defined as granularity levels for place and object. When multiple targets in different granularity levels are recognised, only the *ObjectEvents* at the topmost level are triggered and *ObjectEvents* of lower levels are ignored by determining the proximity from triggered *ObjectEvents* according to the scale of the geometric boundary for each object.

7. Conclusion

Webizing mobile AR content cannot be achieved partially by embedding HTML and XML tag sets but rather by fundamentally adapting an AR content model according to the Web principle and the separation of concerns among content (HTML), presentation style (CSS) and behaviour (JavaScript). We presented an HTML-based mobile AR content model without introducing any new tag sets and by preserving the current Web content eco-system. The structure and behaviour can be captured within the CSS and JavaScript framework of HTML5 such that current page-based Web content can be fully leveraged for mobile AR content using the Web eco-system. We validated the feasibility of our content structure by presenting successful application cases. We expect that the proposed AR content structure conforming to the Web architecture will facilitate the development of scalable AR applications and the sharing of AR content.

Acknowledgement

This research is supported in part by Ministry of Culture, Sports and Tourism (MCST) and Korea Creative Content Agency (KOCCA) in the Culture Technology (CT) Research & Development Program 2013 and by the Korea Institute of Science and Technology (KIST) Institutional Program (Project No. 2E24100).

References

Adobe Systems Inc., 2013. PhoneGap. Available online at: http://phonegap.com (accessed 13 March 2013).

J. Behr, Y. Jung, T. Drevensek and A. Aderhold, “Dynamic and interactive aspects of X3DOM”, in *International Conference on 3D Web Technology*, New York, NY: ACM, pp. 81–87, 2011.

R. Berjon, S. Faulkner, T. Leithead, E.D. Navara, E. O’connor, S. Pfeiffer and I. Hickson, 2013. HTML5, W3C candidate recommendation 6 august 2013. Available online at: http://www.w3.org/TR/2013/CR-html5–2013 0806/ (accessed 13 March 2013).

T. Berners-Lee, 1996. Universal resource identifiers – axioms of web architecture. Technical report, W3C, December 1996. Available online at: http://www.w3.org/DesignIssues/Axioms.html (accessed 13 March 2013).

T. Berners-Lee, 1998. Webizing existing systems. Available online at: http://www.w3.org/DesignIssues/Webize.html (accessed 13 March 2013).

P. Dahne and J.N. Karigiannis, “Archeoguide: System architecture of a mobile outdoor augmented reality system”, in *Mixed and Augmented Reality, 2002. ISMAR 2002. Proceedings. International Symposium on*. Darmstadt, Germany: IEEE, pp. 263–264, 2002.

S. Guven and S. Feiner, “A hypermedia authoring tool for augmented and virtual reality”, *New Review of Hypermedia and Multimedia*, 9(1), pp. 89–116, 2003.

A. Hill, B. Macintyre, M. Gandy, B. Davidson and H. Rouzati, “KHARMA: An open KML/HTML architecture for mobile augmented reality applications”, in *Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on*, Seoul, Korea: IEEE, pp. 233–234, 2010.

T. Höllerer, S. Feiner, T. Terauchi, G. Rashid and D. Hallaway, “Exploring mars: Developing indoor and outdoor user interfaces to a mobile augmented reality system”, *Computers & Graphics*, 23(6), pp. 779–785, 1999.

I. Jacobs and N. Walsh, 2004. Architecture of the world wide web, W3C recommendation 15 december 2004. Available online at: http://www.w3.org/TR/webarch (accessed 13 March 2013).

T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan and H. Morris, “People, places, things: Web presence for the real world”, *Mobile Networks and Applications*, 7(5), pp. 365–376, 2002.

A. Lamarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes and F. Potter, “Place lab: Device positioning using radio beacons in the wild”, in *Proceedings. International Conference on Pervasive Computing*, pp. 116–133, 2005.

Layar, 2013. Layar. Available online at: http://www.layar.com (accessed 13 March 2013).

M. Lechner and M. Tripp, 2010. “ARML – An augmented reality standard”, in *Mobile Augmented Reality Summit, Mobile World Congress*. Barcelona.

B. Macintyre, A. Hill, H. Rouzati, M. Gandy and B. Davidson, 2011. “The Argon AR web browser and standards-based AR application environment”, in *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, Basel, Switzerland: IEEE, pp. 65–74.

L. Madden, *Professional Augmented Reality Browsers for Smartphones: Programming for Junaio*, layar and wikitude: Wiley, 2011.

Metaio Inc., 2013a. AREL (augmented reality experience language). Available online at: http://dev.metaio.com/arel/overview (accessed 13 March 2013).

Metaio Inc., 2013b. Junaio. Available online at: http://www.junaio.com (accessed 13 March 2013).

T. Pixley, 2000. Document Object Model (DOM) Level 2 Events Specification Version 1.0, W3C Recommendation 13 November 2000. Available online at: http://www.w3.org/TR/DOM-Level-2-Events (accessed 13 March 2013).

Qualcomm, 2013. Vuforia. Available online at: http://www.qualcomm.com/solutions/augmented-reality (accessed 13 March 2013).

Weather Underground, 2012. The personal weather station project. Available online at: http://www.wunderground.com/weatherstation/index.asp (accessed 13 March 2013).

Wikitude Gmbh, 2013. Wikitude Architect. Available online at: http://www.wikitude.org/architect (accessed 13 March 2013).

T. Wilson, 2008. OGC KML, Open Geospatial Consortium Inc., document # 07–147r2. Available online at: http://www.opengeospatial.org/standards/kml (accessed 13 March 2013).

H.T. Wolde, 2010. Dutch layar signs global augmented reality deals. *Reuters* [online]. Amsterdam, Reuters. Available online at: http://www.reuters.com/article/2010/06/18/us-augmented-idUSTRE65H45B20100618 (accessed 13 March 2013).

M. Youssef and A. Agrawala, “The Horus location determination system”, *Wireless Networks*, 14(3), pp. 357–374, 2008.