International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.5 (2014), pp.73-82
http://dx.doi.org/10.14257/ijmue.2014.9.5.07

Provisioning of Context-Aware Augmented Reality Services Using MPEG-4 BIFS

Byoung-Dai Lee

Department of Computer Science, Kyonggi University, Suwon 443-760, South Korea
blee@kgu.ac.kr

Abstract

Since its introduction in the late 90's, Augmented Reality (AR) has emerged as a killer service in various domains. In contrast to conventional monolithic AR services that do not consider individual user contexts, AR services are currently evolving into their next phase. Two important requirements for next generation AR services include a personalized context-aware presentation of AR information and the provisioning of an enhanced user experience in a more realistic manner. Therefore, in this paper, we present methods for supporting such functionalities using the Binary Format for Scenes (BIFS) technology standardized by the Moving Picture Experts Group (MPEG).

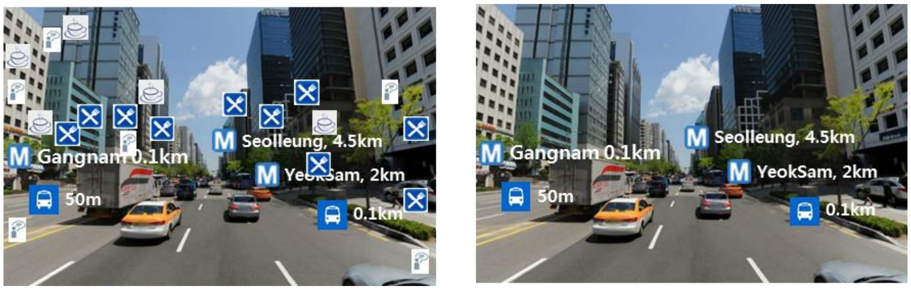
Keywords: *Augmented Reality, BIFS, Context-Awareness, Sensory Effects*

1. Introduction

Augmented Reality (AR) refers to a simple combination of real and virtual worlds and it has been widely used in various domains including education, broadcasting, health, and entertainment. In the developmental stages of AR technology, its main focus was a natural synthesis of real-world and virtual 2D/3D objects to enhance the user's perception of and interaction with the real world through the use of augmented information. Registration, which refers to the accurate alignment of real and virtual objects, is therefore a critical component technology of AR in that the illusion of virtual objects coexisting in the same space within the real world can be severely compromised without an accurate registration. Owing to the widespread use of mobile smart devices equipped with various sensors such as a global positioning system and camera, as well as the availability of diverse high-speed wireless connectivity options, mobile AR services have become increasingly popular in recent years. In particular, personalized context-aware presentation of AR information and the provisioning of an enhanced user experience in a more realistic manner are important requirements for attracting mobile users.

Context-awareness is an important feature of future AR services. In particular, as the amount of AR information increases, presenting all information simultaneously can decrease the readability and usefulness of the information. Therefore, it is strongly recommended that AR information be presented selectively on the basis of user circumstances. An example of a context-aware AR service is shown in Figure 1.

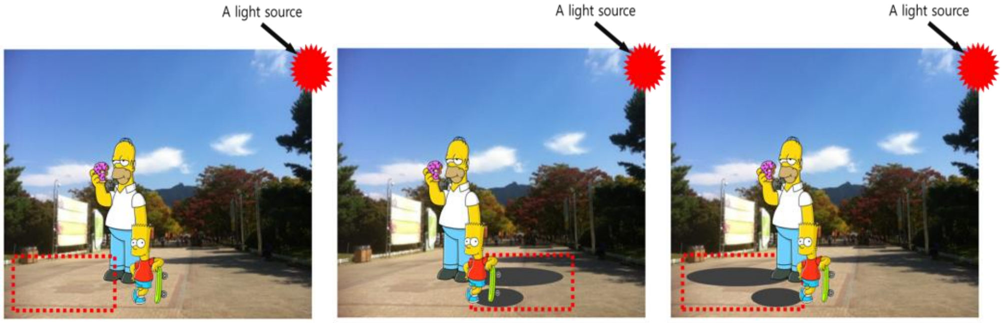
ISSN: 1738-9968 IJHIT
Copyright ⓒ 2014 SERSC



(a) (b)

Figure 1. (a) Conventional and a context-aware (b) AR services.

Suppose that a user runs an AR application during rush hour. In Figure 1(a), four different types of AR information (*i.e.*, subway stations, bus stops, restaurants/cafes, and air tags) are displayed, regardless of the current context of the user. However, suppose that the user seldom visits to restaurants or cafes during rush hour and that the context management system has learned this user habit. It would thus be much more useful to show the user AR information related only to public transportation (see Figure 1(b)). Similarly, the presentation of virtual objects without considering their surrounding environment can lead to an unrealistic user experience. For instance, in Figure 2, the light source is in the upper right corner from the user's perspective. It would therefore be more realistic to render the shadows of the virtual objects (Homer and Bart Simpson) on their left side.



(a) (b) (c)

Figure 2. Virtual Objects with No Shadows (a), Virtual Objects with Shadows on the Incorrect Side (b), and More Realistic Virtual Objects with Shadows on the Correct Side (c)

Binary Format for Scenes (BIFS) [3] is one of the representative standards for representing and delivering rich media services and it is the core technology used in the AR standardization of the Moving Picture Experts Group (MPEG). In this paper, we extended the BIFS to associate context information with a group of virtual objects and allow for the presentation of a subset of AR information that corresponds to given context information. In addition, we propose a mechanism for delivering sensory information to any virtual objects interested in a sensory effect. A sensory effect represents the effect used to augment the perception by stimulating human senses in a particular scene of a multimedia application and sensory information therefore includes odors, wind, light, haptic, and tactile.

Copyright ⓒ 2013 SERSC

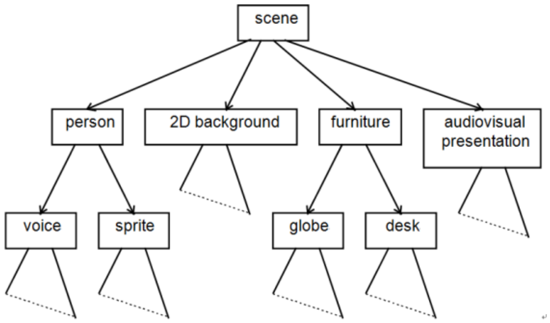


Figure 3. Hierarchical Structure of BIFS Scene Tree

The remainder of this paper is organized as follows: An introduction to a BIFS scene description is presented in Section 2. Sections 3 and 4 describe the proposed methods for supporting context-awareness and delivery of sensory information in detail, respectively. Section 5 provides the implementation results. Finally, Section 6 offers some concluding remarks regarding the proposed methods.

2. Introduction to MPEG-4 BIFS Scene Description

BIFS is an MPEG scene description standard used to represent temporal and spatial relationships among multimedia objects, such as audio, video, image, graphics, and text, as well as user interactions. As shown in Figure 3, a BIFS scene is composed of a collection of nodes arranged in a hierarchical tree. Each node represents, groups, or transforms an object (*e.g.*, audio, video, or graphical objects) in the scene and consists of a set of fields that define and control the node properties, such as the size, color, and location in a 3D space. For example, the sphere node contains a radius field used to define the size of a rendered sphere object. Building on a Virtual Reality Modeling Language (VRML) [2] scene graph, BIFS defines 62 new nodes on top of the 54 nodes defined by VRML. The major extensions include the 2D/3D scene composition, special nodes for facial and body animation, extended sound composition and a query node for terminal resources [1].

The data types of the node fields can be either single or multiple values, as indicated by single-value field (SF) and multiple-value field (MF) immediately before the actual data type. Fields listed as SF data types, such as SFFloat, can contain only one value, whereas MF data types, such as MFFloat, can accept an array of values. The node fields can be classified into one of four possible types: *field*, *exposedField*, *eventIn*, and *eventOut*. The field type is used for values that are set only when instantiating a node. The *eventIn* field is used to receive events, and *eventOut* can be considered the conduit through which the events generated by a node are sent. The *exposedField* allows both sending and receiving events. All fields of the *exposedField* type have an *eventIn* and *eventOut* implicitly associated with them. Some node fields are active and emit events. For example, the BIFS timer, TimeSensor, emits float events, which are values in the interval [0, 1] indicating where the scene is within the current timer cycle. The routes are the means to connect the *eventOut* field of a node to an *eventIn* field of a different node and can be considered the wiring used to connect event generators to event receivers. Sensor nodes sense changes in the user and environment for authoring an interactive scene. They generate events based on user interaction or a change in the scene. For instance, DiscSensor and PlaneSensor2D are drag sensors. They detect the dragging of a pointer-device such as from a mouse or joystick and enable the dragging

Copyright ⓒ 2014 SERSC

of objects on a 2D-rendering plane. Switch nodes allow a subset of nodes to be selected for rendering. Index values are assigned to child nodes contained in a switch node, with the first child having an index 0, and the *whichChoice* field of a switch node specifies the index of the child node to be traversed.

The spectrum of MPEG-4 end-devices ranges from standard computers to mobile devices, and interactive TV sets. For this purpose, BIFS defines several scene description profiles: Basic2D, Simple2D, Core2D, Main2D, Advanced2D, Complete2D, Audio, 3DAudio, and Complete. For instance, the Core2D profile of BIFS has been adopted for interactive data services in Terrestrial Digital Multimedia Broadcasting (T-DMB) and has been deployed on various personal devices with different resource capabilities

3. Support of Context-Awareness in BIFS

To support context-awareness in BIFS, we propose a *ContextGroup* node that associates context information with a group of BIFS nodes that represent AR information, and a *ContextSwitch* node that allows for the presentation of a subset of AR information corresponding to the given context information.

Table 1. Detailed Syntax for the ContextGroup Node

Node	Syntax

ContextGroup	`ContextGroup {`
	`eventIn MFNode addChildren`
	`eventIn MFNode removeChildren`
	`exposedField MFNode children []`
	`exposedField MFString context [“”]`
	`exposedField SFString contextRepType “”`
	`exposedField SFFloat priority 1.0`
	`exposedField MFString relatedURL [“”]`
	`}`

Table 1 shows the detailed syntax and semantics of the *ContextGroup* node. The *children* field specifies a group of nodes for AR information that share the same context. The AR information contained in the children field can be dynamically added or removed by the *addChildren* or *removeChildren* field. The *context* field specifies the context information associated with the AR information represented by the children field. As several techniques are available to represent context information (*e.g.*, a key-attribute approach, an XML schema approach, and an Ontology approach) and because there is no standard for this, the *contextRepType* field indicates a specific technique used for a context representation. Therefore, information contained in the context field must be interpreted according to the method indicated by the *contextRepType* field. The *priority* field specifies the priority of the AR information. This field is used to determine what AR information should be presented when there are multiple matches to the given context and it is impossible to display all of the matched AR information. For instance, when the AR information for subway stations and bus stops are matched with the current context information corresponding to “public transportation,” the device may show only the higher priority AR information if the device display is not large enough to show the AR information for both subway stations and bus stops. The *relatedURL* field specifies the location where additional information about the context such as the context representation method can be acquired.

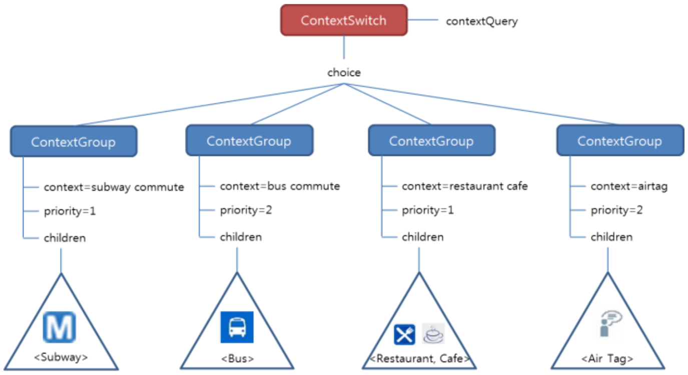


Figure 4. Logical Structure of Scene Description for Context-Awareness

Table 2. Detailed Syntax for the ContextSwitch Node

Node	Syntax

ContextSwitch	`ContextSwitch {`
	`  exposedField MFNode    choice                []`
	`  exposedField SFBool    enabled               FALSE`
	`  exposedField SFString  contextQuery          ""`
	`  exposedField SFString  contextQueryRepType   ""`
	`}`

Table 2 shows the detailed syntax and semantics of the *ContextSwitch* node. The *choice* field specifies AR information against which the *contextQuery* will be executed. The *enabled* field specifies whether context-awareness must be initiated. The *contextQuery* field specifies a query that represents the context of interest. The *contextQueryRepType* field specifies the method for a context query representation.

Figure 4 illustrates the logical structure of the BIFS scene description for the AR service described in Figure 1. Each *ContextGroup* node contains nodes representing AR information in its children field and sets the context field accordingly. Note that in this example, simple string matching is used for a context representation and context comparison. The four *ContextGroup* nodes are listed in the choice field of the *ContextSwitch* node. If it is detected that a user is going to work, then the *contextQuery* field is set to "commute," which results in the presentation of AR information associated with "commute." In addition, if the display size of the user's mobile device is not large enough to accommodate both subway stations and bus schedules, only AR information on the subway stations will be displayed according to the *priory* field.

4. Management of Sensory Information in BIFS

To provide an enhanced user experience in a more realistic manner, two logically separate processes are required: (1) the acquisition of sensory information, such as light, wind, and temperature, from the real world, and (2) the application of sensory effects to virtual objects. Sensory information can be acquired not only by receiving data from various sensor devices, but also by analyzing the media content itself. Although the existing MPEG-4 BIFS provides nodes (*e.g.*, *InputSensor* and *Script*) to deal with the processes mentioned above, it has the following limitations: (1) the *InputSensor* node can only be used to receive sensory information from sensor devices, not from the

Copyright ⓒ 2014 SERSC

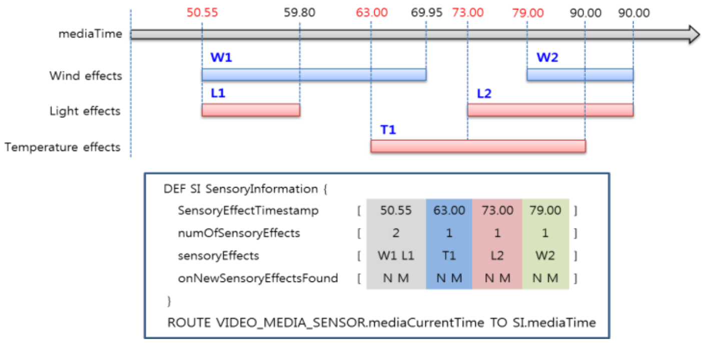
media content, and (2) the acquisition of sensory information and the presentation of virtual objects based on this information are tightly coupled when data are received from the sensor devices, and the *InputSensor* node should invoke BIFS commands that in turn invoke a *Script* node responsible for rendering specific virtual objects using the acquired sensory information. This indicates that if the virtual objects are replaced, not only the corresponding *Script* node but also the BIFS commands contained in the *InputSensor* node may need to be changed. To address these limitations, we propose a new node, *SensoryInformation*, responsible only for delivering sensory information to any virtual objects that are interested in the sensory effects. Table 3 shows the detailed syntax and semantics of the proposed node.

Table 3. Detailed Syntax for the SensoryInformation Node

Node	Syntax

SensoryInformation	`ContextSwitch {`
	`  eventIn       SFTime   mediaTime`
	`  exposedField MFFloat  sensoryEffectTimestamp`
	`  exposedField MFInt   numOfSensoryEffectsPerTS []`
	`  exposedField MFNode sensoryEffects           []`
	`  exposedField MFNode onNewSensoryEffectsFound []`
	`}`

The *mediaTime* field specifies an input event of the node. Each *SensoryInformation* instance must be linked to the media time of the target stream to compare the timestamp values of the *sensoryEffectTimestamp* field with the current media time. The *sensoryEffectTimestamp* field stores a list of time values relative to the *mediaCurrentTime* for the node to know when new sensory effects are activated. The *numOfSensoryEffectsPerTS* field stores a list of integer values, each of which represents the number of new sensory effects activated at the corresponding timestamp. Therefore, the number of timestamps stored in the *sensoryEffectTimestamp* field must be equal to the number of integer values stored in the *numOfSensoryEffectsPerTS* field. The *sensoryEffects* field stores nodes that represent the sensory effects. Nodes are stored in increasing order of the timestamps associated with them. The number of nodes stored in the *sensoryEffects* field must be equal to the sum of the integer values stored in the *numOfSensoryEffectsPerTS* field. The *onNewSensoryEffectsFound* field is used to deliver nodes that represent the sensory effects. When the timestamp value of the *sensoryEffectTimestamp* field matches the current media time, the corresponding nodes stored in the *sensoryEffect* field are sent using the *onNewSensoryEffectsFound*.



```
DEF SI SensoryInformation {
  SensoryEffectTimestamp     [ 50.55  63.00  73.00  79.00 ]
  numOfSensoryEffects        [   2      1      1      1   ]
  sensoryEffects             [ W1 L1   T1     L2     W2   ]
  onNewSensoryEffectsFound   [  N M    N M    N M    N M  ]
}
ROUTE VIDEO_MEDIA_SENSOR.mediaCurrentTime TO SI.mediaTime
```

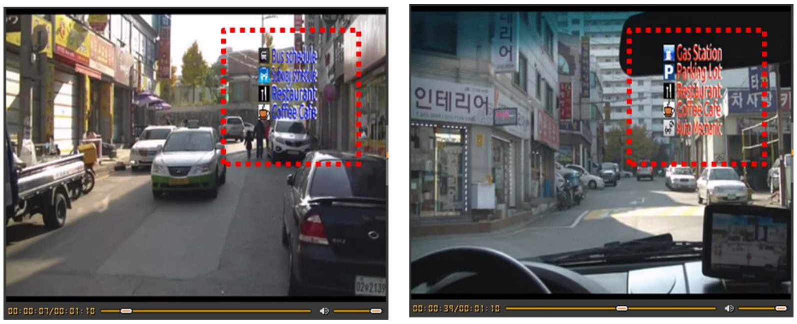
Figure 5. Example of Sensory Information Description

Copyright ⓒ 2013 SERSC

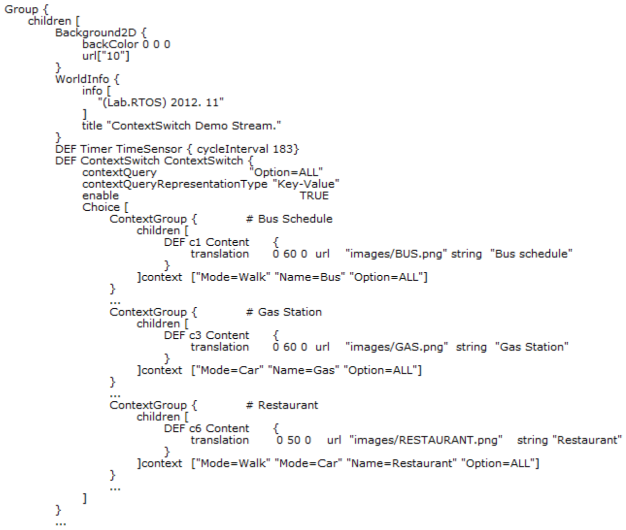
Figure 5 illustrates an example using the *SensoryInformation* node. The timing diagram shows the types of sensory effects detected in the multimedia content and their start and end times. For example, at media time of 50:55 and 79:00, a wind starts blowing and lasts until 69:95 and 90:00, respectively. W1 and W2 are nodes containing detailed information about wind effects such as the intensity, direction, and duration. Note that in this study, we do not specify the syntaxes and semantics for nodes that represent sensory effects (*e.g.*, W1, W2, L1, L2, and T1). However, we envision that the existing MPEG-V specification [4] may be a basis for modeling such nodes. The *SensoryInformation* node contains an array of sensory information in such a way that each element of a field with multiple value types contains information about the sensory effects activated at the same time. For example, the first element of the *SensoryEffectTimestamp*, *numOfSensoryEffects*, and *sensoryEffects* fields contain information on the sensory effects that are activated at a media time of 50:55. Similarly, the second elements of these fields contain sensory information activated at a media time of 63:00. If nodes are interested in receiving sensory information or are responsible for managing scenes based on sensory effects (*e.g.*, N and M), they need to be inserted into the *onNewSensoryEffectsFound* field of the corresponding *SensoryInformation* node.

5. Experimental Results

To evaluate the effectiveness of our approach, we implemented a prototype using the GPAC framework [5]. GPAC is an implementation of the MPEG-4 systems standard written in ANSI C. It provides tools for media playback, vector graphics and 3D rendering, publishing, and content distribution.



(a) (b)

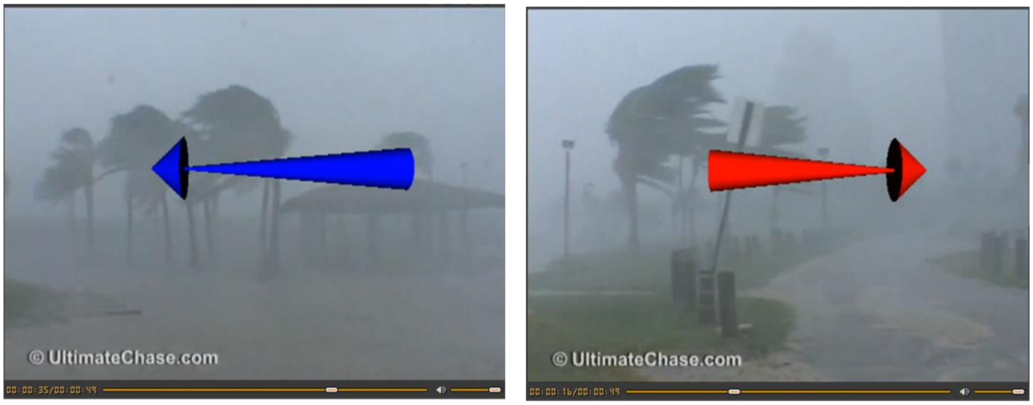


```
Group {
    children [
        Background2D {
            backColor 0 0 0
            url["10"]
        }
        WorldInfo {
            info [
                "(Lab.RTOS) 2012. 11"
            ]
            title "ContextSwitch Demo Stream."
        }
        DEF Timer TimeSensor { cycleInterval 183}
        DEF ContextSwitch ContextSwitch {
            contextQuery                    "Option=ALL"
            contextQueryRepresentationType "Key-Value"
            enable                                 TRUE
            Choice [
                ContextGroup {          # Bus Schedule
                    children [
                        DEF c1 Content     {
                            translation     0 60 0  url    "images/BUS.png" string  "Bus schedule"
                        }
                    ]context  ["Mode=Walk" "Name=Bus" "Option=ALL"]
                }
                ...
                ContextGroup {          # Gas Station
                    children [
                        DEF c3 Content     {
                            translation     0 60 0  url    "images/GAS.png"  string  "Gas Station"
                        }
                    ]context  ["Mode=Car" "Name=Gas" "Option=ALL"]
                }
                ...
                ContextGroup {          # Restaurant
                    children [
                        DEF c6 Content     {
                            translation      0 50 0    url  "images/RESTAURANT.png"    string "Restaurant"
                        }
                    ]context  ["Mode=Walk" "Mode=Car" "Name=Restaurant" "Option=ALL"]
                }
                ...
            ]
        }
        ...
```

(c)

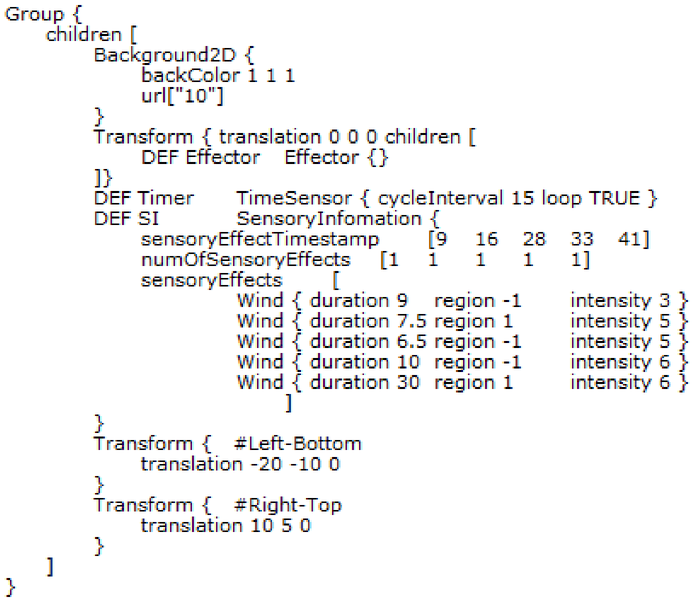
Figure 6. Context-aware AR Tour Guide Service

Figure 6 shows an example of a context-aware AR tour guide service. Icons are displayed for various types of information depending on the user context. For example, while users are walking down the street, they are not interested in information on gas stations or car repair shops. Therefore, the service shows icons for further information on the nearby subway stations, bus schedules, restaurants, and cafes (see Figure 6(a)). However, while driving, users may not be interested in information on subway stations and bus schedules. Therefore, as shown in Figure 6(b), only information that is useful when users are driving is displayed. A BIFS code snippet relevant to the context-awareness is shown in Figure 6(c). In this example, a key-value approach is used for representing the context. When the external context management system detects changes in the user context (*e.g.*, walking or driving), it feeds the corresponding information (*e.g.*, “Mode=Walk” or “Mode=Car”) to the *context* field of the *ContextSwich* node. When the context field changes, the GPAC media player replaces the scene accordingly.



(a)

(b)



```
Group {
    children [
        Background2D {
            backColor 1 1 1
            url["10"]
        }
        Transform { translation 0 0 0 children [
            DEF Effector   Effector {}
        ]}
        DEF Timer      TimeSensor { cycleInterval 15 loop TRUE }
        DEF SI         SensoryInfomation {
            sensoryEffectTimestamp      [9    16   28   33   41]
            numOfSensoryEffects     [1    1    1    1    1]
            sensoryEffects      [
                    Wind { duration 9    region -1      intensity 3 }
                    Wind { duration 7.5 region 1        intensity 5 }
                    Wind { duration 6.5 region -1       intensity 5 }
                    Wind { duration 10  region -1       intensity 6 }
                    Wind { duration 30  region 1        intensity 6 }
                        ]
        }
        Transform {   #Left-Bottom
            translation -20 -10 0
        }
        Transform {   #Right-Top
            translation 10 5 0
        }
    ]
}
```

(c)

Figure 7. Rendering of Virtual Objects based on Sensory Effects

Figure 7 shows an example of virtual objects displayed in consideration of the sensory effects that exist in the multimedia content. As shown in Figures 7(a) and 7(b), the direction of the arrow icon differs depending on the direction of the wind. In so doing, the user experience can be enhanced in a more realistic manner. Figure 7(c) shows the snippet of BIFS code used to implement this feature. In this example, we defined a *Wind* node for delivering detailed information about the wind effect. It consists of three fields representing the duration, intensity and region of the wind.

6. Conclusions

Owing to the widespread use of mobile smart devices equipped with various sensors such as a global positioning system and camera, as well as the availability of diverse high-speed wireless connectivity options, AR services have become increasingly popular in recent years. In particular, a personalized context-aware presentation of AR information and the provisioning of an enhanced user experience in a more realistic manner are important requirements for attracting mobile users. In this paper, we presented methods for providing such functionalities by extending the existing MPEG-4 BIFS technology.

Acknowledgements

This work was supported by Kyonggi University Research Grant 2013.

References

[1] B. D. Lee and J. Y. Song, "Mobile Rich Media Technologies: Current Status and Future Directions", TIIS, (**2011**), vol. 5, no. 32, pp. 359-373.

[2] ISO/IEC 14772-1, Information Technology – Computer graphics and image processing – The Virtual Reality Modeling Language – Part 1: Fundamental specification and UTF-8 encoding, Int'l Organization for Standardization/Int'l Electrotechnical Commission, (**1997**).

[3] ISI/IEC 14496-11:2005(E), Information Technology – Coding of Audio-Visual Objects – Part 11: Scene Description and Application Engine, Int'l Organization for Standardization/Int'l Electrotechnical Commission, (**2005**).

[4] ISI/IEC 23005-5:2013, Information Technology – Media Context and Control – Part 5: Data Formats for Interaction Devices, Int'l Organization for Standardization/Int'l Electrotechnical Commission, (**2013**).

[5] GPAC, http://gpac.wp.mines-telecom.fr/

Authors

Byoung-Dai Lee, he is an assistant professor at the department of Kyonggi University, Korea. He received his B.S. and M.S. degrees in Computer Science from Yonsei University, Korea in 1996 and 1998 respectively. He received his Ph.D. degree in Computer Science and Engineering from University of Minnesota, twin cities, U.S.A. in 2003. Before joining the Kyonggi University, he worked at Samsung Electronics, Co., Ltd as a senior engineer from 2003 to 2010. During the period, he has participated in many commercialization projects related to mobile broadcast systems. His research interests include mobile rich media, augmented reality, and mobile multimedia broadcast.

Copyright ⓒ 2013 SERSC