A New Multimedia Synchronization Specification Method for Temporal and Spatial Events

Jongho Nang
Dept. of Computer Science
Sogang University
1 Shinsoo-Dong, Mapo-Ku
Seoul 121-742, Korea
e-mail : jhnang@ccs.sogang.ac.kr

Sujin Kang
Telecomm. Network Research Lab.
Korea Telecom
62-1, Whaam-dong, Yusung-gu
Taejon, Korea
e-mail : sjkang@npx12.kotel.co.kr

Abstract

This paper proposes a new multimedia synchronization specification method, called SSTS (Synchronization Specification method for Temporal and Spatial events) that can specify spatial events as well as temporal events synchronizations between multiple media objects in a single framework. In this proposed method, a temporal and spatial events of a media object are specified as nodes, and their temporal relations are specified as edges, in a synchronization specification graph. This approach helps to easily specify the temporal relation that one temporal or spatial event triggers other temporal or spatial events without breaking the media object into several independent media objects. This paper also presents a prototype authoring system based on SSTS with ScriptX under Windows95 to show the usefulness of proposed specification method.

1 Introduction

Multimedia applications are required to present several kinds of media objects such as voices, graphics, animation, image, audio and full-motion video which are captured or created independently, while keeping the synchronization constraints specified by the programmers [4]. In order to build a sensory-rich multimedia application, the programmer (or author) of the multimedia applications should be able to specify not only the temporal event synchronizations but also various spatial event synchronizations between several media objects. The former synchronization refers to the temporal dependencies between the playing of media objects, whereas the latter synchronization refers to it the spatial layout of screen at a specific time instance or the spatial moving of the media object. An example of the latter spatial event is that when an audio object finishes its playback, it triggers the event that a video object starts to move to a specific position along a specified path. However, the research on the multimedia synchronization have been focused mainly on the temporal synchronization of media objects, so that it has been a main obstacle to build a sensory-rich multimedia application with various spatial events.

There have been a lot of specification methods for temporal event synchronization between multimedia objects such as OCPN [7] which an extension of Petri-net with duration and resources, a specification method based on time-axes [5, 10] in which the presentation events like the start and end of a presentation are mapped to axes that are shared by the objects of the presentation, and a script language with timing operations [14]. On the spatial synchronization, there have been also a few research efforts [14, 6] to develop a way to specify the spatial events synchronization between media objects, but they are too primitive to specify the various playing scenarios because they have focused only on the static spatial information of media objects. In the script-based spatial specification method proposed in [14, 13], for example, only the static spatial information of each media object when it begins its playback can be specified. Although the spatial information specification method used in Action! [8] can express the moving path of media object itself while playing back, it still cannot express the synchronization of spatial moving events between several media objects. In order to specify this kind of synchronization in Action!, the media object should be split into several independent media objects and specify the synchronizations separately because the media object itself is an atomic (*i.e. indivisible*) synchronization unit.

0-8186-7819-4/97 $10.00

There was also a research [3] considering synchronization of spatial events, however, it also cannot specify temporal and spatial synchronization in a single framework, since it has separate editors for temporal synchronization, spatial synchronization and user interaction. Furthermore, all specification methods discussed above cannot express a synchronization that temporal event triggers other spatial event or vice versa.

This paper proposes a new multimedia synchronization specification method, called SSTS (Synchronization Specification method for Temporal and Spatial events), that can specify temporal events, spatial events and their temporal relations in a single framework. In SSTS, a temporal and spatial playing event of media objects (not a media object itself), are specified as the nodes, and their temporal relations are specified as the edges, in a synchronization specification graph. The completion of one temporal or spatial event triggers other media object events connected by several temporal edges in the synchronization specification graph. This specification method allows to specify the synchronization relationship that the completion of an media object (which implies the activation of the end node of the media object) triggers the spatial moving events of other media object (which implies the activation of spatial node attached to the media object), or vice versa. A prototype multimedia authoring system based on SSTS is also presented to show the usefulness of the proposed synchronization specification scheme. It consists of an editor with graphical user interface to graphically specify the synchronization relationship between the media objects, a translator that automatically translates the synchronization specification specified with SSTS into an equivalent ScriptX [12] language program, and a player that shows the presentation scenario according to the synchronization constraints specified with SSTS under Windows95.

The proposed specification method is adequate for the multimedia applications such as advertisements or educations, in which a variety of synchronization relations should be specified in order to show many media objects in a timely- and spatially-synchronized fashion.

2 Design of a New Synchronization Specification Method

The basic synchronization specification method proposed in this paper is a graph-based method since it is more simple than script-based one. Actually, the Firefly's graph notations [2] are used in the new specification method, and the transition rules of OCPN [7] are augmented to express the temporal relations between spatial and temporal events. The required static spatial information are extracted from the document formatting languages, whereas the essential spatial events specifiable in SSTS such as media object movings are defined by analyzing the several multimedia applications.

2.1 The Basic Model

The basic idea of synchronization specification method proposed in this paper is all synchronization events whether temporal or spatial are represented as a set of nodes and their temporal relations are represented as a set of edges in a synchronization specification graph. This specification method allows us to represent spatial and temporal events, such as ends of audio playing and image moving, in the exactly same way within a single framework. Using this method, an event synchronization such as starting an audio object when an image object arrives at a specific position can be easily specified.

The basic synchronization specification model, called SSTS (Synchronization Specification method for Temporal and Spatial events), is defined as a direct graph $G_{SSTS} = \{BN, EN, SN, PL, CE\}$, where

$$
\begin{aligned}
&BN = \{B_i\}, \quad B_i = B_i^{OR} \bigcup B_i^{AND} \\
&EN = \{E_i\}, \quad E_i = E_i^{OR} \bigcup E_i^{AND} \\
&SN : \{S_{ij}\}, \quad S_{ij} = S_{ij}^{OR} \bigcup S_{ij}^{AND} \\
&PL : \{B_i \times S_{ij}^{*} \times E_i\} \\
&CE = \text{1-}CE \vee \text{2-}CE \\
&\quad = \{B_i \times B_j\} \bigcup \{B_i \times E_j\} \bigcup \{B_i \times S_{ke}\} \bigcup \\
&\quad\quad \{E_i \times B_j\} \bigcup \{E_i \times E_j\} \bigcup \{E_i \times S_{je}\} \bigcup \\
&\quad\quad \{S_{ie} \times B_j\} \bigcup \{S_{ie} \times E_k\} \bigcup \{S_{ie} \times S_{kf}\}
\end{aligned}
$$

BN, EN, and SN represent a set of begin-nodes, a set of end-nodes, and a set of spatial-nodes of media objects, respectively. The begin-node represents the temporal event that a media object starts its playing, whereas the end-node represents its completion. The spatial-node represents a spatial event such as the start of image moving along a specified path while it is being displayed. The OR-version of these nodes such as B_i^{OR}, E_i^{OR} and S_{ij}^{OR} could be activated when any incoming edge is activated, whereas the AND-versions such as B_i^{AND}, E_i^{AND} and S_{ij}^{AND} could be activated only when all incoming edges are activated. PL is a set of playing-edges each of which represents media object is being played. A playing-edge connects the begin-node and the corresponding end-node of a media object, and there can be zero or more spatial-nodes between them. One begin-node, end-node and playing-edge are minimum elements for playing one media object. 1-CE is a set of one way control-edges each of which activates its

target node after the labeled time is expired, whereas 2-CE is a set of two way control-edges each of which activates its source and target nodes at the same time. Both edges represent the temporal relations of spatial and temporal events of media object being played. The number of begin-nodes and corresponding end-nodes is equal to that of media objects involved in a multimedia application.

The activations of temporal and spatial events, which are represented as begin-nodes, end-nodes and spatial nodes, are controlled by the one way and two way control-edges. The event firing rules used in SSTS are summarized as follows;

- A node starts its action if it is completely activated.
- A begin-node without incoming control-edge is completely activated at time zero.
- An AND-node is completely activated only when all incoming edges are activated.
- An OR-node is completely activated when any incoming edge is activated
- A playing-edge cannot activate corresponding end-node until the media object finishes its playback.
- A node with outgoing one way control-edge cannot activate corresponding target node until the time specified on the control-edge has elapsed. If there is no time specified, it activates the target node immediately
- Two nodes connected by a two way control-edge could be activated only when all incoming edges of both nodes are activated. If this condition is satisfied, both nodes are activated simultaneously.

With this synchronization specification graph and events firing rules, we can specify various temporal and spatial events synchronizations as explained in the next section, in which the graphical symbols shown in <Figure-1> are used for explanations.

2.2 Synchronization Specification of Temporal Events

In order to specify the start and end of media object presentation, there should be a begin-node, end-node, and a playing-edge connecting them. The begin-node has an information on the media object it stands for

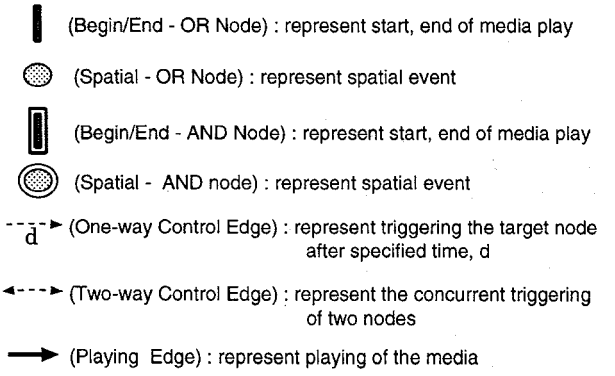


Figure 1. Graphical Notations used in SSTS

and its initial static spatial position at screen if it is an image or a video clip. The end-node is activated only when the playback of the media object is finished by playing-edge.

Let us explain how the temporal event synchronization could be specified with SSTS via example. <Figure 2> shows an example of synchronization specification between temporal events.

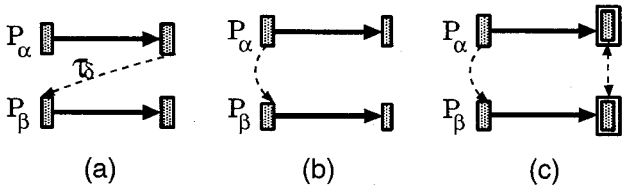


Figure 2. An Example of Synchronization Specification of Temporal Events

In <Figure 2>-(a), the begin-node of object P_β is completely activated after playback of object P_α is finished and time τ_δ has elapsed. In <Figure 2>-(b), object P_α and P_β start their playback simultaneously since P_α's begin-node has completely activated P_β's begin-node. In <Figure 2>-(c), playbacks of P_α and P_β start simultaneously like <Figure 2>-(b), however, their end-nodes could be completely activated at the same time when the object that has a longer playback duration finishes its playback, since both end-nodes are AND-nodes and connected by a two way control-edge. It implies that any media object events synchronized with the end of P_α and P_β will not be activated until both P_α and P_β end their playbacks.

OCPN[7] is a widely used temporal synchronization specification method based on graph like SSTS. The temporal interval relations between two media objects targeted in OCPN are *before*, *meets*, *overlaps*, *during*, *starts*, *finishes*, *equals*, and their inverse relations ex-

cluding *equals*. These temporal interval relations could also be specified easily with SSTS as shown in <Figure 3>, in which only the 7 temporal relations are shown for the sake of simplicity. The omitted 6 inverse relations could be also specified easily if we change the corresponding control edges.

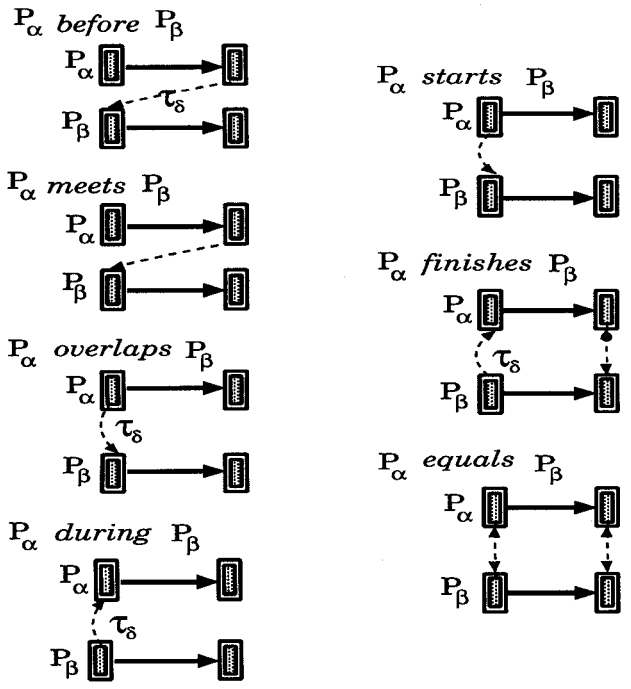


Figure 3. Seven Temporal Relations Specified with SSTS

2.3 Synchronization Specification of Spatial Events

Besides the static spatial information such as the initial position and visibility of media object when it starts its playback on the output device, more spatial events should be specifiable in order to make a sensory-rich multimedia applications. In order to fulfill this requirement, we first define the static spatial information of media object specifiable in SSTS by referencing the ones used in the document formatting languages such as *pic* [9] and *Postscript* [11] and multimedia standard MHEG [1, 6], whereas the essential spatial events are defined by analyzing the several multimedia applications such as advertisements and CAI (Computer Aided Instruction). They are summarized as follows ;

- `AbsolutePath(time, pos)` : move the object to x, y coordinates of `pos` during `time`.
- `RelativePath(time, dest_obj, dest_pt, src_pt, dist, angle)` : move the object to the position relative to `dest_obj`.
- `CurveMotion(time,` pos_1, pos_2, $\cdots$, pos_n`)` : move the object during `time` along a curve which passes pos_1, pos_2, $\cdots$, pos_n.
- `ShapeMotion(time, shape)` : move the object along `shape` during `time`.
- `Visibility(mode)` : make the object shown or hidden.
- `Basis(obj)` : put the object on the object `obj`.
- `Scaling(time, x_pt, y_pt)` : scale the object up or down by `x_pt%`, `y_pt%` during `time`
- `Rotating(time, angle)` : rotate the object as much as `angle`.

`AbsolutePath()` can be compared with absolute coordinate system in *PostScript*, *pic* and parallel relation in MHEG, while `RelativePath()` can be compared with relative coordinate system and serial relation in MHEG. `CurveMotion()` and `ShapeMotion()` are events for a media object movement following a curve and a shape, respectively. Although the set of spatial events currently supported in SSTS is not complete to express all spatial information and actions, the spatial events commonly used in conventional multimedia applications could be expressed with the combinations of the above spatial events. Furthermore, they could be easily expanded as required if a translation rule is added as discussed later.

The spatial events are represented as spatial nodes and located between begin- and end- node (*i.e. on the playing-edge*) of a media object in a SSTS specification graph, if they are required. The order (or position) of the spatial nodes attached to the same playing-edge does not convey any meaning, because these nodes are controlled only by their incoming control edges. Its activation rules are the same as the rules for the begin- and end- node of media object, *i.e.*, it is activated if incoming control edges are all activated (in the case of AND spatial node) or any control edge is activated (in the case of OR spatial node). The activation of spatial node means that the media object starts its spatial action specified in the corresponding spatial node. <Figure 4> shows an example of specification of spatial event synchronization, in which a media object, `image1`, moves along the path defined by `AbsolutePath(3,(100,200))` which means to move to the absolute position `(100,200)` during 3 time units

smoothly, if the spatial node is completely activated by the control edge, a.

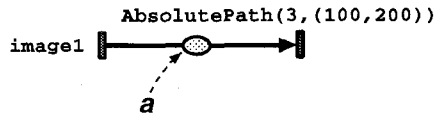


Figure 4. An Example of Synchronization Specification of Spatial Event

2.4 Synchronization Specification of Temporal-Spatial Events

The main advantage of the SSTS is that synchronization of the temporal event and spatial event could be specified as the same way, so that it is possible to make a presentation scenario that a temporal event can trigger the movement of other object or vice versa. Let us explain the meaning of this kind of event synchronization with an example shown in <Figure 5>.

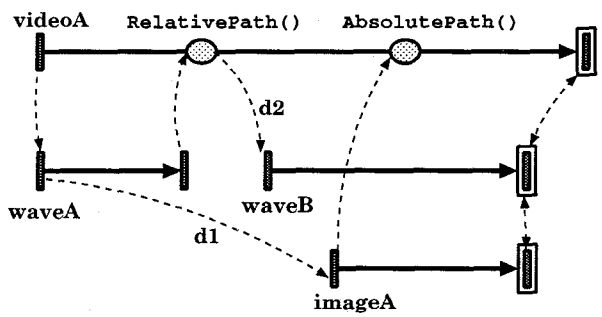


Figure 5. An Example of Synchronization Specification of Temporal-Spatial Events

In <Figure 5>, `videoA` and `waveA` objects start their playback simultaneously, and the completion of `waveA` temporal event triggers the spatial event that `videoA` moves along the path specified in `RelativePath()` spatial node. After `videoA` completes its movement and `d2` time has elapsed, it triggers the playback of `waveB` object. In the meantime, `imageA` appears on the screen, just `d1` time after the start of `waveA`. This temporal events triggers the the event that `videoA` moves along the path specified in `AbsolutePath()` spatial node. The media object `imageA` will remain on the screen until `waveB` and `videoA` finish their playbacks.

Note that there can be an inconsistency in the specification shown in <Figure 5> if `AbsolutePath()` spatial node is activated while the movement of `RelativePath()` is not finished yet. This inconsistency could occur when the playing time of `waveA` specified in its start node plus the duration time specified in `RelativePath()` is longer than the time specified in the control edge connecting `waveA` and `imageA`, `d1`. This inconsistency can be detected as an error by the translator since the playing times of all media objects involved in the presentation are known in advance in a SSTS specification graph.

3 Implementation of SSTS Execution System

We have implemented a SSTS execution system (a prototype multimedia authoring system) with ScriptX [12] on Windows95 whose system architecture is shown in <Figure 6>[1]. It consists of a specification editor with a graphic user interface, a translator, ScriptX compiler, and KMP (Kaleida Media Player) interpreter. The specification editor equipped with several graphic icons for SSTS specification primitives helps the multimedia programmer to specify the temporal and spatial event synchronizations in a graphical form easily. The event synchronizations specified with the editor is translated into an equivalent ScriptX program by the translator with the helps of predefined event function libraries. The translated scenario expressed in ScriptX is then compiled and executed by KMP interpreter. Let us explain each component in more detail, focusing on how the synchronization specification with SSTS is translated into an equivalent ScriptX program.

3.1 Specification Editor

The graphical user interface of the specification editor is shown in <Figure 7>. At the left side of the window, icons for specification of nodes and edges are located, and command buttons are located in the lower part. Users can drag and drop any icons for nodes and edges to the main window, and input the temporal and spatial information of media objects by double-clicking the icons.

3.2 Translator

The translator interprets the temporal and spatial events of the media objects, and computes the time at which each temporal and spatial event start their actions. The ScriptX functions used in implementing the temporal events are `importAVI()`, `importBMP()`,

[1] what we have actually implemented is the shaded part of <Figure 6>.

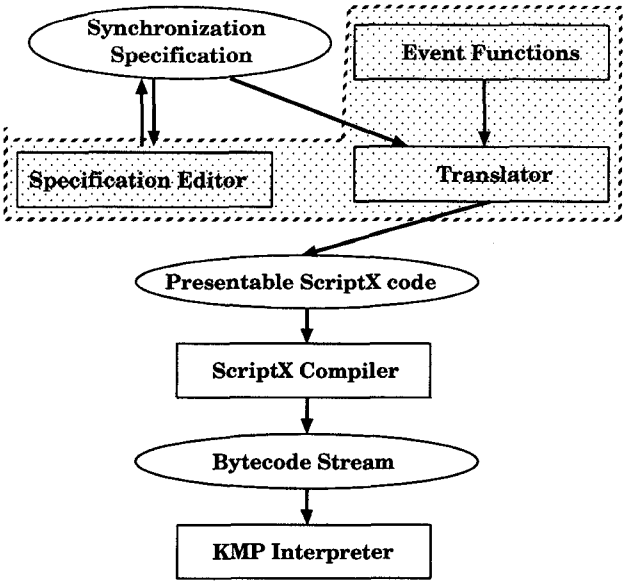


Figure 6. A SSTS Execution System

`importWAVE()`, `BeginVideo()`, `BeginImage()`, `BeginAnnounce()`, `play()` and `stop()` functions. Before temporal events are executed, all media files used in scenario have to be imported, and then actions of temporal events are executed by using playback/stop functions at the time that is computed by SSTS scheduler.

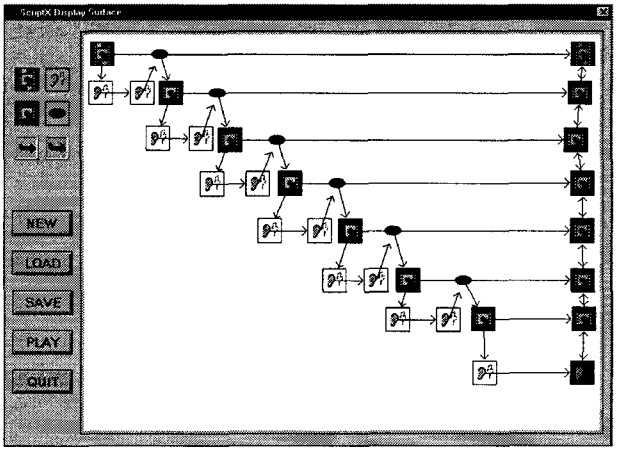


Figure 7. A Specification of CAI Example with Editor

3.3 Execution of Example Scenario

Let us show an example of multimedia application, especially CAI (Computer Aided Instruction), which requires various kinds of the spatial as well as temporal event synchronizations. This presentation scenario explains six plants with their images together with a full-motion video for instructor and seven voice media objects each of which is captured separately. They should be integrated and presented in a spatially and temporally synchronized fashion in order to effectively explain the main characteristics of six plants to students. The SSTS synchronization specification of this scenario is shown in <Figure 8>, and its editing snapshot on the graphic SSTS editor is shown in <Figure 7>. The snapshots of actual screen outputs when above scenario is presented are also shown in <Figure 9>.

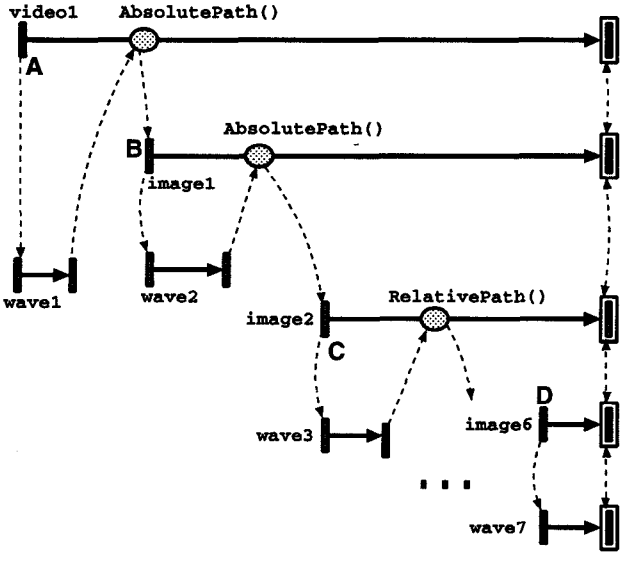


Figure 8. A CAI Example Specified with SSTS

First, `video1` and `wave1` start their playbacks at the time `A` in <Figure 8>, and its corresponding execution scene is shown in <Figure 9>-(a). When `wave1` finishes its playback, it completely activates the spatial event `AbsolutePath()` of `video1`. On the finishing of `AbsolutePath()` spatial event, `image1` and `wave2` start their playbacks. This is a point of the time `B` in <Figure 8>, and corresponding execution scene is also shown in <Figure 9>-(b). After location of `image1` is changed, `image2` and `wave3` start their playbacks. It is the time `C` in <Figure 8>, and <Figure 9>-(c) shows the execution scene at that time. As soon as `wave3` finishes its playback, `RelativePath()` which moves `image2` to relative location of `image1` is executed. This process is repeated until all images are shown with their corresponding explanation voices. At the time `D` in <Figure 8>, `wave7` and `image6` start their playbacks after the locations of `image2`, `image3`, `image4`, and `image5` are changed by `RelativePath()` events. This is shown in <Figure 9>-(d).

Through this CAI example, we show that tempo-

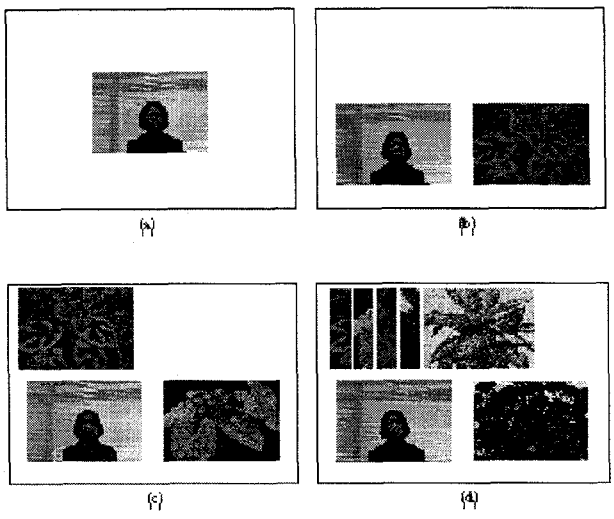
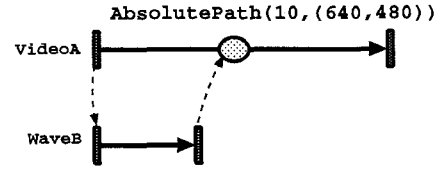
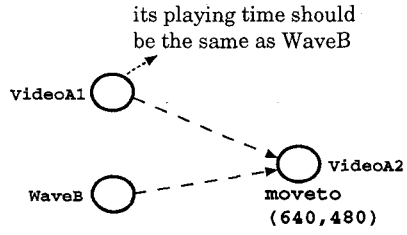


Figure 9. Execution processes of CAI Example



(a) A Synchronization Specification with STSS (Event as a Node Specification)



(b) A Synchronization Specification with Action! (Object as a Node Specification)

Figure 10. A Comparison of *Event as Node* and *Object as Node* Specification

ral/spatial events and their temporal relations are naturally specified with SSTS and the specification can be actually executed by our execution system.

4 Comparison with Related Works

The main characteristics of SSTS is that an event (not object itself) is represented as a node, and their temporal relations are represented as edges in the synchronization specification graph. The node can be a spatial event as well as a temporal event. Let us show the expressiveness power of this specification method, while comparing with related works.

Assume that `VideoA` should move to the screen position of `(640,480)` when `WaveB` finishes its playback in a presentation scenario. The temporal and spatial events synchronization required in this simple scenario can be specified with SSTS easily as shown in <Figure 10>-(a). In principle, this presentation scenario could not be specified with the specification methods that have only the temporal event specification mechanism such as OCPN [7] and the specification methods based on time-axes [5, 10]. Furthermore, it cannot also be specified with the specification methods that only support the static spatial information (such as position, size, and overlapping) of media object when it starts its playback. Examples belonging to this category are the specification methods proposed in [14] and [6].

The presentation scenario shown in <Figure 10>-(a) can be represented with the specification methods that can support the spatial movement of the media object while playing back such as the one used in Action! [8]. In this specification as shown in <Figure 10>-(b), however, the `VideoA` should be split into two independent video clips (`VideoA1` and `VideoA2`) and the temporal relation with the `WaveB` is specified separately. This is caused by the fact that in Action! the media object itself is an atomic presentable unit so that the spatial (or temporal) events required during its playback could not be specified. Furthermore, in this method, if the `WaveB` is changed to `WaveB'`, the `VideoA1` and `VideoA2` should be split again according to the playing length of `WaveB'`.

The specification method proposed by Eun *et. al.* [3] has separate editors for temporal synchronizations, spatial synchronizations, and user interactions for specifying various event synchronizations. This specification method allows the author of a multimedia applications to specify the spatial as well as the temporal event synchronizations. However, since it has separate editors for temporal and spatial event synchronizations and their relations are specified indirectly via *synchronization point*, it may be very hard to specify the temporal and spatial event synchronization in a single framework. On the other hand, in SSTS, both of the temporal and spatial event are represented as nodes and their temporal relations are represented as edges in *a single synchronization specification graph*. It allows the author of a multimedia application to specify various temporal and spatial event relations between media objects naturally and easily in a single framework. That is an unique feature that other synchronization specification methods (or authoring tools) did not have.

Furthermore, the presentation scenario specified with SSTS can be automatically converted in ScriptX and presented under Windows95, and it can be used any platform where ScriptX system is available.

5 Summary

As multimedia technologies are being used in the wide range of applications, there has been increased demands for specifying the synchronization of various playing events between media objects. Since researches on the multimedia synchronizations so far have been focused mainly on the temporal synchronization between multiple objects, it has been difficult to specify the multimedia presentation scenarios such as the spatial movement of a media object triggers the playing of other media objects or vice versa. This paper aims at developing a multimedia synchronization method that can naturally specify the spatial as well as the temporal event synchronizations in a single framework.

This paper presents a graph-based synchronization specification method, called SSTS, in which the spatial as well as temporal events are represented as nodes and their temporal relations are represented as edges. The innovative features of the proposed specification method compared to existing specification models and authoring tools are, firstly, the spatial events synchronization required in the course of a media object playback can be expressed without breaking the media object, and secondly temporal events and spatial events synchronizations are specified as the same way in a single synchronization specification graph. These features help to easily specify the temporal relation that one temporal or spatial event triggers other temporal or spatial events without breaking the media object into several independent media objects. This paper also presented a prototype multimedia authoring system based on SSTS which has been implemented with ScriptX under Windows95. The proposed specification method may help the multimedia programmers to build a sensory-rich multimedia applications that integrate a lot of media objects each of which is synchronized in a variety way.

We are currently investigating a way to incorporate some user interaction specifications in SSTS, and developing a translator that converts the SSTS specification to an equivalent MHEG documentation for the portability of the multimedia presentations.

References

[1] T.M. Boudnik and W. Effelsberg, "MHEG Explained," *IEEE Multimedia*, Vol.2, No.1, Spring, 1995, pp.26-38.

[2] M.C. Buchanan and P.T. Zellweger, "Automatic Temporal Layout Mechanisms," *Proc. of ACM Multimedia 93*, June 1993, pp.341-350.

[3] S.B. Eun, E.S. No, H.C. Kim, H.S. Yoon and S.R. Maeng, "Eventor: An Authoring System for Interactive Multimedia Applications," *Multimedia Systems*, Vol.2, No.3, September 1994, pp.129-139.

[4] B. Fuhrt, "Multimedia Systems : An Overview," *IEEE Multimedia*, Vol.1, No.1, Spring 1994, pp.47-59.

[5] M.E. Hodges, R.M. Sasnett, and M.S. Ackerman, "Athena Muse : A Construction Set for Multimedia Applications," *IEEE Software*, Jan. 1989, pp.37-43.

[6] F. Kretz and F. Colaitis, "Standardizing Hypermedia Information Objects," *IEEE Communication Magazine*, Vol.30, May 1992, pp.60-70.

[7] T.D.C. Little and A. Ghafoor, "Synchronization and Storage Models for Multimedia Objects," *IEEE Journal On Selected Areas in Communications*, Vol.8, No.3, April 1990, pp.413-427.

[8] T. Vaugham, *Action!3.0 User's Guide*, MacroMedia, 1994.

[9] N. Gehani, *Document Formatting and Typesetting on the UNIX System*, Silicon Press, 1986.

[10] S.R. Newcomb, N.A. Kipp, and V.T. Newcomb, "HyTime," *Communication of the ACM*, Nov. 1991, pp.67-83.

[11] G.C. Reid, *POSTSCRIPT Language Tutorial and Cookbook*, Addison Wesley, 1985.

[12] Kaleida Labs, *ScriptX Core Classes Reference*, Inc. Mountain View, CA 94043, 1994.

[13] Y. Theodoridis, M. Vazirgiannis, and T. Sellis, "Spatio-Temporal Indexing for Large Multimedia Applications," *Proc. of Int'l Conference on Multimedia Computing and Systems*, 1996, pp.441-448.

[14] M. Vazirgiannis and C. Mourlas, "An Object-Oriented Model for Interactive Multimedia Presentations," *The Computer Journal*, Vol.36, No.1, 1993, pp.78-86.