S^3G: A Semantic Sequence State Graph for Indexing Spatio-Temporal Data - a Tennis Video Database Application

Mitesh Naik, Vani Jain, and Ramazan S. Aygun
Computer Science Department
University of Alabama in Huntsville
{mnaik,vjain,raygun}@cs.uah.edu

Abstract

The indexing of spatio-temporal is important for retrieval of data demanded by spatio-temporal queries. The previous techniques on spatio-temporal indexing miss the semantics of the application since they are usually based on traditional indexing structures that has little to no semantic information incorporated. In those systems, the semantic queries were executed by using the low-level index structures. In this paper, we introduce a novel indexing method for spatio-temporal data: semantic sequence state graph (S^3G). S^3G maintains the properties of events-objects-locations for efficient spatio-temporal queries. In S^3G, the spatial information is maintained in states whereas semantic events that result in temporal ordering link the states. S^3G supports our SMART (semantic modeling and retrieval) system.

1. Introduction

With the growing interest in spatio-temporal data more than a decade, the indexing and retrieval of spatio-temporal has been a challenging research area. Spatio-temporal data may appear in biological databases, atmospheric systems, geographic information systems, and multimedia database systems. In this paper, we focus on spatio-temporal data in multimedia database systems since multimedia data plays a key role in today's world including but not limited to education, advertisement, entertainment, communication, and information retrieval. Text, image, video, and graphics are some of the forms of multimedia. Especially, videos have been the most intrigue media since videos have multimodal features along with spatio-temporal properties. Growth in internet technology, processing power, and computer architecture has resulted in further proliferation of videos. This has led to the challenge of storing, retrieving and indexing of video data. The complexity of video data along with the growing demand has made efficient storage and retrieval of video a vital area of interest for researchers. Different strategies have focused on modeling of different aspects of videos such as modeling fuzzy information [3] and spatiotemporal features of the objects in a video [1, 2]. The goal of our research is to model, store, query, and index the semantic contents of the videos.

Significant research has been performed on indexing temporal and spatial databases [5, 7]. A good survey on indexing temporal data appears in [6]. Indexing spatio-temporal data is mostly based on traditional database indexing techniques such as R-trees [8] and B+-trees [9]. One of the major problems of indexing based on these indexing methods is the lack of necessary semantics to build queries that require semantic information. Therefore, semantic retrieval on top of these indexing methods can only be implemented by an upper layer of semantic operations. For example, B+ tree provides actually an ordering of the data using a comparison measure. This comparison measure is usually based on a key field that has to be different for each object in the database. The semantic retrieval system processes this ordering and tries to match with a semantic concept. This puts additional burden on the retrieval. If the indexing method could capture semantic properties, the retrieval efficiency could be improved.

In order to provide an efficient storage and retrieval of video data, we proposed a semantic modeling and retrieval system, *SMART* [4]. SMART provides an efficient method of modeling the semantic contents of video. The semantic content of a video represents the information or data in the video that interests the viewer. For example, the player

names in sports video, the shots given by the players, and their score. This semantic information is represented as a grammar-based string that enables spatio-temporal queries in SQL query language.

In this paper, we improve our SMART system by providing a novel indexing method, named as S^3G: a semantic sequence state graph. The major difference between this indexing method and the traditional ones is that in S^3G the links between states have semantics where states maintain the discrete information about the spatial properties of objects. The events correspond to the transitions in S^3G graph. Since transitions correspond to semantic events, it is possible to perform queries based on semantic concepts following the transitions in S^3G. We should note that we are not interested in the shapes of the objects. We are rather interested where and when they appear. We are interested in spatio-temporal events that can be denoted at discrete times. We assume that a semantic event causes the difference between two states. The spatial queries are performed with respect to the object-location pairs. Temporal queries are performed by following the transitions in S^3G graph. Spatio-temporal queries combine the both.

This paper is organized as follows. The following section provides the background of semantic modeling and retrieval system, *SMART*. Section 3 defines S^3G. Section 4 explains mapping from SMART string to S^3G. The interface and querying is briefly discussed in Section 5. The last section concludes the paper.

2. Background on SMART

The major semantic contents of video are considered as objects, events, locations, and cameras in SMART.

Objects are the main entities of the video such as players in a game. An object is represented as a symbol from the domain $\Sigma_O = \{O_1, O_2, O_3, \ldots, O_n\}$.

Events are the main actions of objects such as serving in a tennis game. An event is represented as a symbol from the domain $\Sigma_E = \{E_1, E_2, E_3 \ldots E_m\}$.

Location is the spatial information of the object such as near the net or outside the court. Location L is represented from the domain $\Sigma_L = \{L_1, L_2, L_3, \ldots, L_p\}$.

Camera is the different footage recording provided by the various cameras located. For example, a camera may be used for the close-view of players and another for the action replay. Camera C is represented from the camera domain $\Sigma_C = \{C_1, C_2, C_3, \ldots, C_q\}$.These semantic contents are described over the time τ.

SMART proposes a grammar that defines the rules for representing the above information. Following is the generic grammar that can be customized in accordance to the game:

```
<video> :: = <sequence of clips>
<sequence of clips> :: =<clip>
                      | <sequence of clips>
<clip> ::= <camera>
                      "["<sequence of spt>"]"
<sequence of spt> ::= <spt>
                      |<sequence of spt>
<spt>:: =  [<event>] <obj> <loc>
```

where *video* is a sequence of clips; *sequence of clips* has many clips; *clip* has a camera view and a sequence of spatiotemporal instances; *sequence of spatiotemporal instances* has spatiotemporal instance; and a spatiotemporal instance (*spt*) is represented with an object (*obj*), location (*loc*), and an optional *event*.

2.1 SMART for Tennis Videos

The above approach and grammar can be applied and used on various sports videos. We have conducted experiments on the tennis game.

Objects. Three main objects, as $\Sigma_O = \{U, V, b\}$, were identified. Here, U represents the player who serves, V represents the other player and b is the ball.

Events. Two events, as $\Sigma_E = \{F, B\}$,were identified. Here, F represents the forehand shot and B represents the backhand shot.

Locations. The tennis court was divided into various regions as shown in Fig 1: $\Sigma_L = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, N\}$.

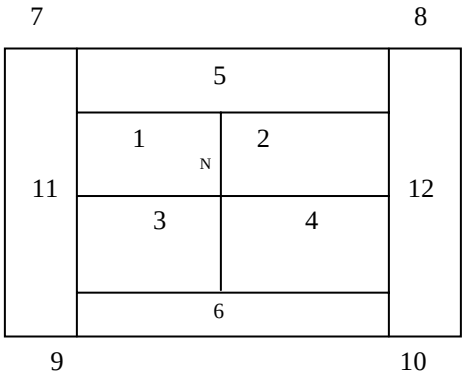

Figure 1: Tennis Court Segmentation

2.1.1. Camera view. Six different camera views are considered for tennis videos:

A – Gives a close view of the player at locations 7 & 8 in Figure 1
B - Gives a close view of the player at locations 9 &10 in Figure 1
C - Court view
D - Action Replay
R - Rest time
Com – Commentators

2.1.2. Grammar for Tennis Game. We have extended the generic grammar to incorporate the semantics of a tennis game. The following is the grammar for the tennis videos:

```
<obj> ::= U|V|b
<event>::= F|B
<location> ::= 1|2|3|4|5|6|7|8|9|10
             |11|12|N
<camera> ::= <close-view camera>|C|D
<close-view camera>::= A|B
<spt>:: =  [<event>] <obj> <loc>
<sequence of spt> ::= <spt>
                    |<sequence of spt>
<clip> ::= < close-view camera >
      "["<obj>"]" | C
      "["<  sequence  of  spt  >  "]"
      |D"["""]"
<sequence of clips> :: =<clip>
             | <sequence of clips>
<video> :: = <sequence of clips>
```

2.1.3. Example. Using the above grammar we can represent the video as a string. For example, consider the video sequence in Figure 2. This video sequence can be written as:

T= {B[U]C[U10b10V9b1BV7b6FU9b5FV8b6FU9b12] }

The above string represents the close view of the player who serves and then the court view. Player serving is at location 10 and other player is at location 9. The ball is initially at location 10. After the serve, the ball's location is 1. Then the other player gives a backhand shot from location 7 and the ball goes at 6. Then the first player gives a forehand shot from location 9 and ball goes at location 5 and so on.

(a)

(b)

(c)

(d)

Figure 2: Sequence from a Tennis Video

2.2 Spatio-Temporal Queries in SMART

The above representation supports spatio-temporal queries. Different types of queries supported by this representation are:

1) Event-Object-Location
2) Object-Location
3) Event- Location
4) Event-Object

An event-object-location type of queries finds out the clip where an event is associated with an object at the given location. Consider the query:

List all the clips where the player who serves gives a forehand shot from area outside the court.

This can be written as

```
SELECT clip
FROM MainTable
WHERE sequence like '%FU7%' or sequence like
'%FU8%';
```

Here, FU7 and FU8 represent the forehand shot by the player who serves from the location 7 and 8. This finds out the pattern of 'FU7' and 'FU8' anywhere in the string.

Similarly, Object-Location, Event-Location and Event-Object type of queries can be defined [4].

3. S^3G: Semantic Sequence State Graph

The semantic sequence graph is a graph where the events, objects, and locations are maintained as states and transitions. S^3G resembles to a finite state machine. An S^3G can be defined as $M=[S,\Sigma,\delta,s_0,F]$ where S is a set of internal states, Σ is a finite input alphabet, s_0 is the initial state, F is a set of external states, and δ is a transition function mapping $S \times \Sigma$ to $S \cup F$. An internal state s ($s \subseteq S$) is a set of object-location pairs. An internal state is a subset of cross product of objects spatial locations ($S \subseteq \Sigma_O \times \Sigma_L$). An external state $f \subseteq F$ corresponds to a decision in a sequence of events. The alphabet (Σ) is a subset of the cross product of the object-event pairs ($\Sigma \subseteq \Sigma_O \times \Sigma_E$). It should be noted that not all objects are associated with an event.

3.1 S^3G Components

The components of SMART grammar have events, objects, locations and camera view as the alphabet. The most important part in building an S^3G is the identification of transition function and the states.

States. A state is identified by objects and their corresponding locations. In addition to object-location pairs, states also maintain pointers for quick retrieval of objects. For each state, there is a list of clip pointers to retrieve the corresponding clips having that state.

The maximum number of states for a video is determined by

$$\prod_{i=1}^{|\Sigma_O|}(|\Sigma_L|) = |\Sigma_L|^{|\Sigma_O|}.$$

Transitions. The transitions are determined by the semantic events. In our applications, the semantic events result in the displacement of objects. In most cases, an action is continuous. Dividing an action into discrete steps is critical in building an S^3G. The determination of discrete steps is decided by semantic events.

3.2 An Example: Tennis Video

A video can be considered as a collection of a series of events and a series of states due to these events. Some videos like a sports video can have finite types of states and events.

We have chosen tennis videos as our application since we can express the details of S3G in a simple way. For example, in a tennis game the most important object is the ball. Each event causes the ball move from one location to another location. Whenever a player hits the ball, the ball changes its position. To reduce the number of states, we are interested only in specific situations. For example, we are interested in whenever a player hits the ball or ball hits the court or net.

States of Tennis. Consider a tennis video for instance. In a tennis game, there are three objects: two players and the tennis ball. At a given instance, the objects with their locations define a state in the video. Since there are 3 objects and 13 locations identified for a tennis video, the total number of states 13^3. However, it is unlikely to have these many states for a tennis game. Therefore, we create a state as long as that state exists in the database.

Transitions of Tennis (Semantic Events). Now principally we can say that there can be two types of

shots that the tennis players can make, i.e. the forehand shot and the backhand shot. So, we can define four types of events for a tennis video, namely a forehand shot by player1, a backhand shot by player1, a forehand shot by player2 and a backhand shot by player2. We can say that there are four types of transitions in a tennis video.

We can thus define a tennis video in terms of a series of states where each state consists of a value for the locations of player1, player2 and the tennis ball. Corresponding to the four types of events possible there can be four types of transitions possible for changing a state with a set of locations values for the players and the ball before the transition to another state with a new set of locations values after the transition (Figure 3).

Example. Consider an initial state which has the values for spatial locations as 7, 10 and 7 for the player1, player2 and the ball, respectively. This means that the player1 and the ball were in location 7, while the player2 was in location 10. Now, if player1 hits a forehand shot and the ball goes to location 4 and so does the player2 to hit the ball back, we will have a new state with values 7, 4, and 4 corresponding to the new location values of the player1, player2, and the ball, respectively. In this case the transition for the initial state to the new state should be defined as player1 forehand shot (Figure 4).

Now there can be many video clips for a given set of locations values for the player1, player2 and the ball. So each state will have many clips corresponding to it. So an array of these clip ids can be maintained for the corresponding clips (Figure 5).

4. Converting String Representation to S^3G

We can use the string representations of SMART [4] to get the state representations of the video. Each string can be read; and the location values obtained can be converted to a state; and the actions/events can be converted to state transitions. For example, we have a string as **U7V10b4FV10b8** that states that initially player1 & ball are in location 7 and player2 is in location 10. The player1 serves the ball that goes to location 4. The player2 in location 10 hits the ball that goes to location 8. Here we can define two states the initial state with location values 7, 10, and 7 for the player1, player2, and ball locations, respectively and the next state with location values as 7, 10, and 8 for the player1, player2 and ball locations, respectively.

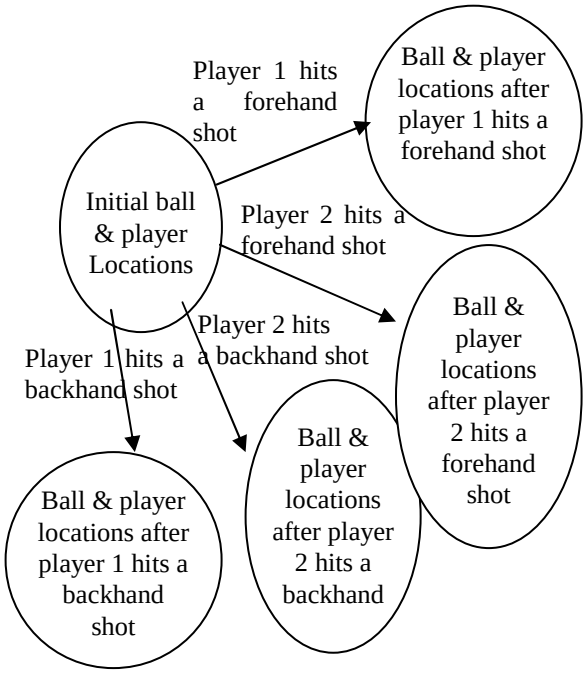

Figure 3: States and Transitions for Tennis Game

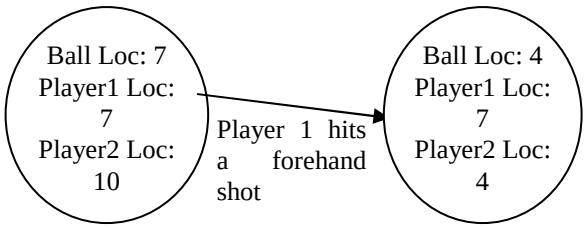

Figure 4: A transition that causes a new state

We will have a transition defined from the initial state to the next state as the player1 forehand shot transitions. The assumption used in string representations should be taken care of during converting them to state representation. For example, the ball location will be the same as the player location who is serving the ball. The players change their location 7 & 10 to 8 & 9 in alternating fashion, which will not be expressed explicitly in the string representations. For example, if the beginning of the video clip has player locations as 7 & 10, the next play will have the player locations as 8 & 9 and vice versa.

Reuse existing state. When we get a set of location values for clip, a check should be made whether a state already exists with the given values; if so the clip id can be added to the array corresponding to the state else a new state is to be created with this clip id as the first entry in its corresponding clip id array.

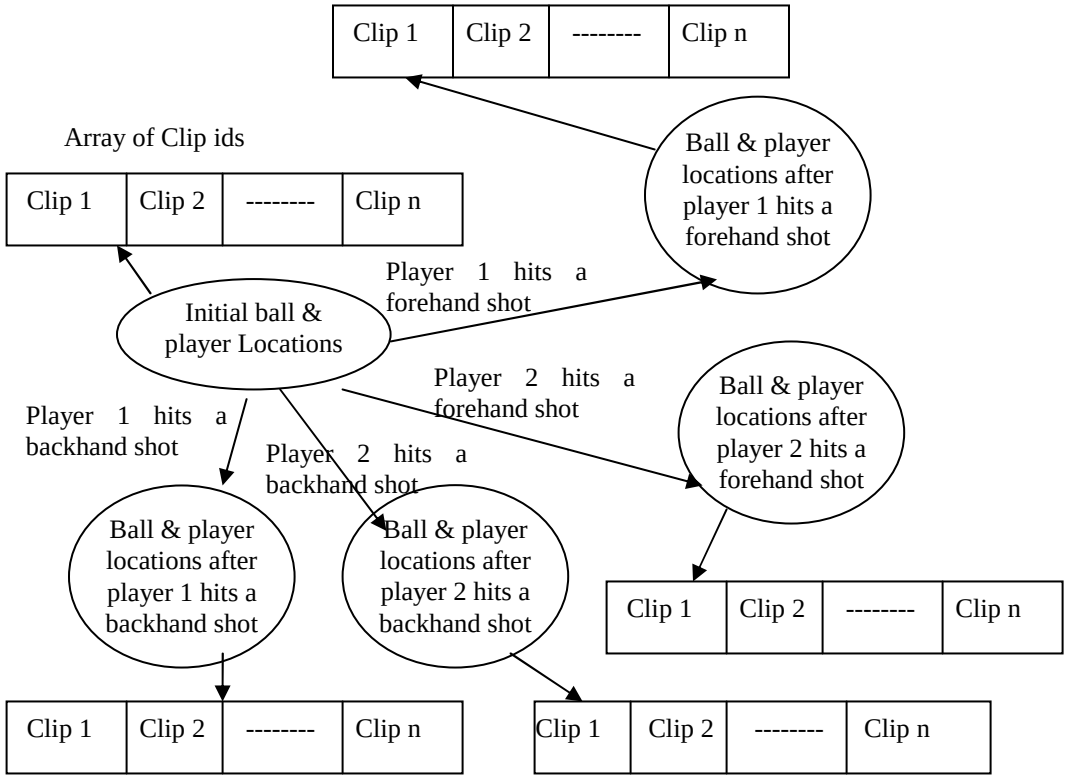

Figure 5: S3G with clips

Assume that we already built an S^3G as in Figure 6. For example, we need to process a new substring as "U7b10FV10b7" for a clip with id 2134**,** which means player2 is in location 10 with the tennis ball while player1 is in location 7. This defines an initial state with location values as 10, 7 and 10 for the tennis ball, player1 & player2, respectively. Since a corresponding state in Figure 6 does not exist, we will have to create a new state (S4 in Figure 7). Next we have an event in which the player 2 hits a forehand shot and the ball goes to location 7. So this corresponds to another state (S1 in Figure 6) with location values 7, 10 and 7 for the tennis ball, player1 & player2 respectively. The clip id 2134 is added to the clip array of state S1 in Figure 7. In this case, we did not need to create a new state. Instead make our initial state of this substring to point to the existing state S1, and add an entry in its array with the given clip id 2134.

The following algorithm provides the pseudo-code of finding an existing state in S^3G.

```
bool SearchState(OP,S,N)
IN: Objects with Positions OP={(o1,p1),
(o2,p2), …, (on,pn)}
IN: An initial state S
OUT: a boolean value found
OUT: N as the found state
begin
  found = false;
  for each object (oi pi) pair in OP
    if pi == (pi in S) for all oi
      state is found
      assign it to N
    endif
  endfor
  if state is not found
   for each unvisited transition t of S
    // transition: e.g. forehand shot
      SearchState(OP,S.t)
   endfor
  endif
  return found
end
```

The following algorithm provides the pseudo-code for converting from string to state.

```
String2State(StrClip)
IN: string StrClip
OUT: S3G graph
begin
 while not end of string
```

```
if (alphabet α ∈ Σ_O)
 α is an object
 Parse the location of the α
 if location of α is implied
  Compute it from another object β
 OP ← object-location pairs
else if (alphabet α ∈ Σ_E)
 if no state is created for OP
  Create a new StartState(OP)
 endif
 α is an event (transition)
 if !SearchState(OP,StartState,
                 NextState)
   create new NextState if a state
     for OP does not exist
   link previous state through α
     to NextState
 endif
endif
read new alphabet α
end while
end
```

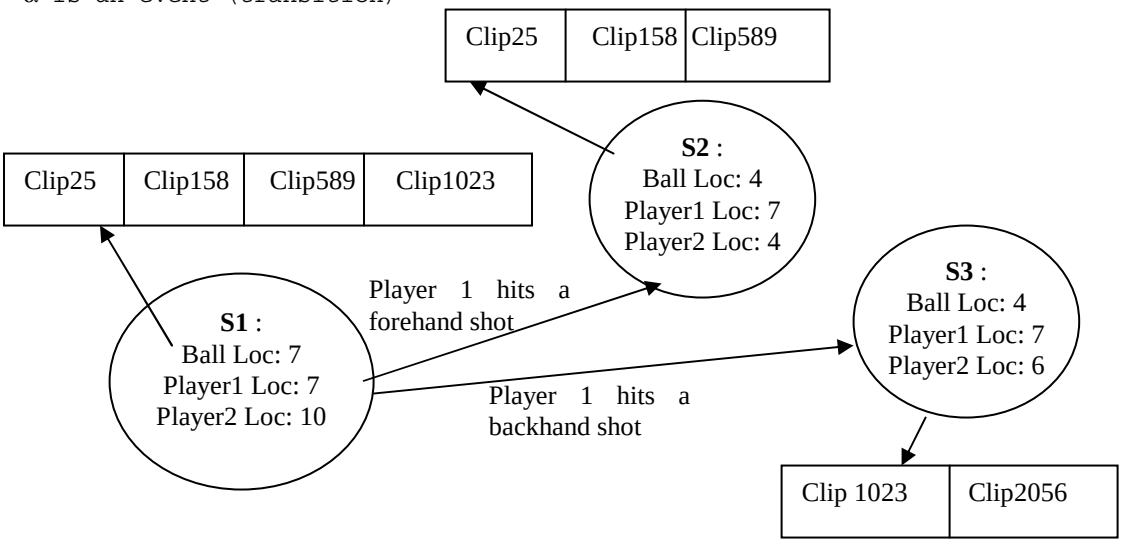

Figure 6: An S3G graph before inserting clip 2134

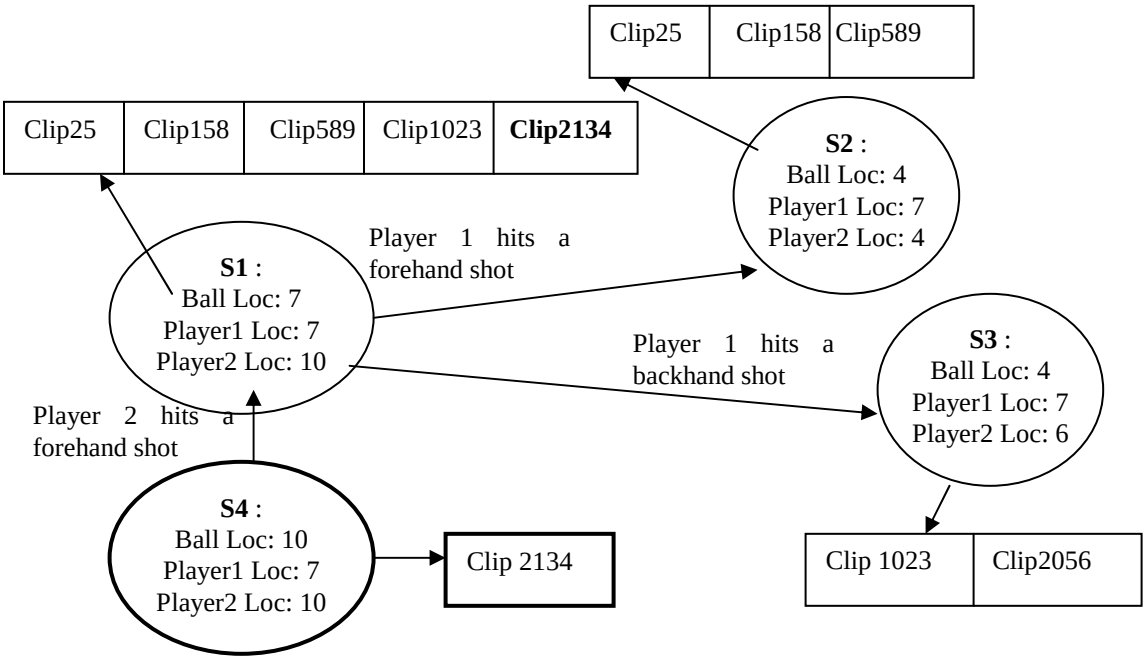

Figure 7: S3G graph after inserting clip 2134 with a new state and an existing state.

5. Interface and Querying

Figure 8 provides the interface for selecting the spatial information. For each object, the location is determined by clicking on the corresponding location on the tennis image. When locations are selected for all objects, the state information is ready. SearchState function given in the previous section finds the state in the S^3G graph. We can display all the clips having the states with these locations values. For example, if the user clicks location 8 for player1 & the tennis ball and location 9 for player2, then all clips in the array stored for the state having location values as 8, 9, and 8 corresponding to player1, player2 and ball, respectively is displayed.

To retrieve states based on an event, we just need to follow the links to retrieve the corresponding links. For example, if we want to retrieve the clips that result after an event such as forehand shot by player1, the transition for this event followed and the clips from the relevant states are retrieved.

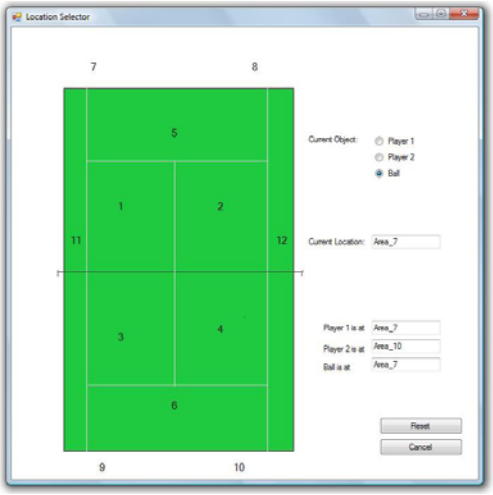

Figure 8: Interface for the Spatial Locations

5. Conclusion and Future Work

In this paper, we provided a new indexing method for spatio-temporal data using the semantic contents of the video. We called this new semantic sequence state graph as S^3G. S^3G is very effective in spatio-temporal queries since it utilizes semantic events as transitions in the events. We plan to develop an efficient way of finding and inserting states from a given string. One of the advantages of S^3G is that queries based on Allen's temporal intervals [11] can be implemented efficiently as future work. Our motivation is to retrieve queries based on temporal ordering using temporal logic. Especially, we are interested in linear temporal logic (LTL).

References

[1] Niki Pissinou, Ivan Radev, Kia Makki and William J. Campbell, "Spatio-Temporal Composition of Video Objects: Representation and Querying in Video Database Systems", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 13, No 16, P.P. 1033-1040, 2001.

[2] John Z.Li, M.Tamer Ozsu and Duane Szafron, "Modeling of moving objects in a video database", *1997 International Conference on Multimedia Computing and Systems*, P.P. 336**,** 1997

[3] Ramazan Savas Aygun and Adnan Yazici, "Modeling and management of fuzzy Information in Multimedia Database Application", *Multimedia Tools and Applications*, Vol. 24,No 1, P.P. 29-56 ,2004

[4] Vani Jain, Ramazan Savas Aygun, "SMART: A grammar -based semantic video modeling and representation" , *IEEE SouthEast Con 2008*

[5] Jensen, C. S. and Snodgrass, R. T. 1999. Temporal Data Management. *IEEE Trans. on Knowl. and Data Eng.* 11, 1 (Jan. 1999), 36-44.

[6] Salzberg, B. and Tsotras, V. J. 1999. Comparison of access methods for time-evolving data. *ACM Comput. Surv.* 31, 2 (Jun. 1999), 158-221.

[7] Jin Suk Min; Dong Ho Kim; Keun Ho Ryu, "A spatiotemporal data and indexing," *Electrical and Electronic Technology, 2001. TENCON. Proceedings of IEEE Region 10 International Conference on* , vol.1, no., pp.110-113 vol.1, 2001

[8] Simonas Saltenis and Christian S. Jensen, "Indexing of Moving Objects for Location-Based Services", ICDE, 2002

[9] Lin, D., Jensen, C. S., Ooi, B. C., and Šaltenis, S. 2005. Efficient indexing of the historical, present, and future positions of moving objects. In *Proceedings of the 6th international Conference on Mobile Data Management* (Ayia Napa, Cyprus, May 09 - 13, 2005). MDM '05. ACM, New York, NY, 59-66.

[10] Hopcroft, J. E., Motwani, R., and Ullman, J. D. 2006 *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc.

[11] James F. Allen: Maintaining knowledge about temporal intervals. In: Communications of the ACM. 26/11/1983. ACM Press