平 成 29 年 度　修 士 研 究 論 文

題目　Recursive Model Localization
and Voice Pattern Prioritization
for Automatic Baseball Video Tagging

指導教員　　服部 峻

提出者　　室蘭工業大学大学院 工学研究科
情報電子工学系専攻

氏　名　　荒澤　孔明

学籍番号　　16043005

提出年月日　　平成 30 年 1 月 31 日

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Highlights videos are frequently used in sports news programs. Because most of these highlights videos are produced by the side of sports news programs, the same sports game has different highlights videos depending on sports news programs. Because highlight scenes are high spots in a sports video, one aim of producing a highlights sub-video of the sports video is to enable those who could not satisfyingly watch the sports video to enjoy it in a short time. However, scenes which a viewer wants to watch in a sports video are dependent on her/his personal preferences, and a highlights sub-video of the sports video cannot have all the scenes that s/he wants to watch. Therefore, highlights videos based on estimating the general needs of many viewers and produced by the side of sports news programs cannot satisfy each individual's needs completely.

The above-mentioned limitation of a highlights video produced by a sports news program could be solved by enabling viewers to produce a highlights video by themselves. One method of enabling a viewer to produce a highlights video by her/himself is "s/he records a sports relay program previously and then edits its video by selecting only the specific scenes that s/he wants to watch." However, in general, such work requires a great deal of time because the viewer has to watch the entire video while editing the scenes that s/he wants to watch and fast-forwarding through the other scenes.

Let us imagine that a sports video has already been divided into multiple chapters per scene. Chaptering is a function to enable a viewer to easily move to the point of a sports video that s/he wants to watch by dividing the sports video into multiple sub-videos per scene and appending a caption to each scene of the sports video. Because the sports video has already been divided into chapters per scene with their caption, the viewer does not have to watch the entire sports video and all s/he needs to do is to collect only the specific chapters that s/he wants to watch by using their captions as a reference. Therefore, the viewer does not need to spend a large amount of time.

Providing a sports video that is previously divided into chapters per scene would enable viewers to watch their wanted scenes easily. There are several existing researches [1–9]

that tackle how to divide a video into sub-videos per scene. Mukunoki et al. [7] discussed a division method of sports videos such as baseball videos into play units (scenes) using regularity of cut composition. Kumano et al. [8] discussed a high-speed extraction method of PC (Pitcher and Catcher) scenes from a live baseball video broadcast. However, these methods cannot recognize what scene it is and whose scene it is. Therefore, I focus on the "Tagging" [10–28] that divides a video into sub-videos per scene with not only their caption but also their Tag information, which is their detailed information showing what event happened in each scene, and are developing an automatic tagging system [29–32] of baseball videos using ball-by-ball textual reports on the Web [33] and voice recognition. My system utilizes ball-by-ball textual reports on the Web which are produced by such a specified organization as Yahoo! JAPAN Sportsnavi, while Nakazawa et al. [9] discussed a labeling method of significant scenes from TV programs by analyzing Tweets that are produced by many and unspecified users.

My proposed system of the basic research utilized only voice-recognized play-by-play comments which represent the batter-name of an at-bat scene, while this paper proposes a novel Tagging method that utilizes multiple kinds of play-by-play comment patterns for voice recognition which represent the situation (e.g., not only the batter-name but also the start/end, his batting order, his batting result, the count of outs, etc.) of an at-bat scene and take their "Priority" into account. In addition, to search for a voice-recognized play-by-play comment on the start/end of at-bat scenes, this paper proposes a novel modelling method called as "Local Modelling," as well as Global Modelling used by the basic research.

The remainder of this paper is organized as follows. Chapter 2 explains what the tagging for baseball videos is. Chapter 3 proposes a novel method for automatic baseball video tagging, that is equipped with Voice Pattern Prioritization and Recursive Model Localization. And Chapter 4 shows and discusses experimental results to verify the proposed method. Finally, Chapter 5 concludes this paper.

Chapter 2

Tagging for Baseball Videos

The proposed Tagging in this paper has the key words: "event," "event time," and "event tag." Fig. 2.1 shows the functions of the proposed Tagging system. When a targeted baseball video is input to the system, which is connected to the Internet, the system divides it into multiple sub-videos per at-bat scene, e.g., the first at-bat scene E_1 is from 00:30 to 02:10, and the second at-bat scene E_2 is from 03:00 to 05:45. A divided at-bat scene is defined as an **event**. That is to say, events are created as many as there are at-bat scenes in the targeted baseball video. Moreover, each event has **event tags** and **event time**. Event tags shows "what events happened in the baseball video."

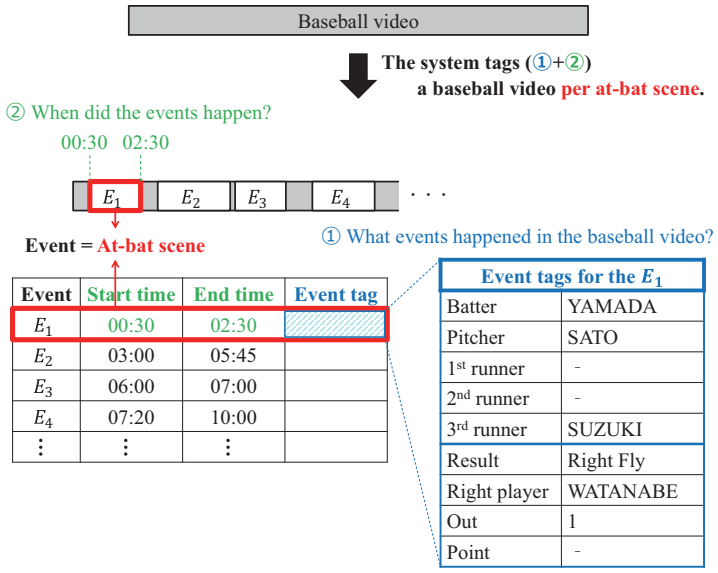

Fig. 2.1 What is the Tagging system for baseball videos?

In Fig. 2.1, "the batter of a divided event E_1 is YAMADA," "the pitcher of the event is SATO," and "the result of the event is Right-fly" are mainly appended to the event E_1 as its event tags. Meanwhile, event time shows "when the events happened," and consists of the start time and the end time of each divided event. A baseball video has not only at-bat scenes, but also the other kinds of scenes, e.g., replay scenes and scenes of cheerleaders' performances. Dividing the baseball video into sub-videos per event (at-bat scene) with distinguishing them from the other kinds of scenes and appending event tags to each event enable a user to create a personalized highlights video which consists of only the specific at-bat scenes that s/he wants to watch.

Chapter 3

Proposed Method

This chapter proposes a novel method for automatic baseball video tagging, that is newly equipped with Voice Pattern Prioritization and Recursive Model Localization.

Firstly, it gives an overview of the proposed method. Secondly, it describes an extraction method for event tags about any at-bat scene of a targeted baseball game, and an appending method for the start time and end time of an at-bat scene. Subsequently, it describes a searching method for voice-recognized play-by-play comments which represent the situation of the at-bat scene and have higher priority for the start/end of the at-bat scene, and a locally-modelling (also globally-modelling) method to estimate the start time and end time of the at-bat scene.

3.1 An Overview

Fig 3.1 shows an overview of the proposed system. Firstly, the system extracts event tags of every event (at-bat scenes) in a targeted baseball game. This process is shown as Step 1 of Fig. 3.1. Secondly, the system appends the start time and end time of every event to divide a baseball video into multiple sub-videos per at-bat scene automatically. This process is shown as Steps 2-6 of Fig. 3.1. To divide a baseball video into multiple sub-videos per at-bat scene (i.e., to populate an event of a batter's at-bat scene with its event's start time and end time), this paper utilizes voice-recognized play-by-play comments which represent the situation of the at-bat scene, which include multiple kinds of play-by-play comments which represent the start of the at-bat scene or the end of the at-bat scene. Meanwhile, my basic system [29–32] utilized only play-by-play comments which represent the name of the batter of the at-bat scene.

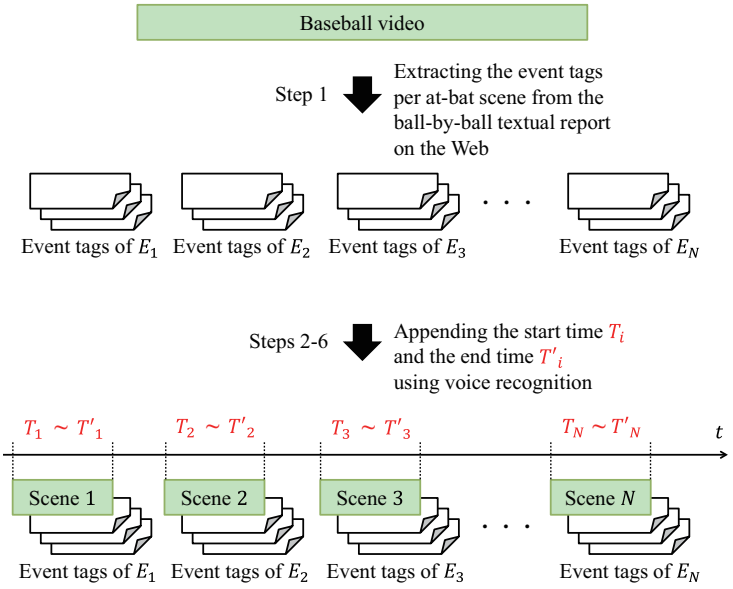

Fig. 3.1 An overview of the proposed system.

In other words, this paper proposes a system that voice-recognizes a play-by-play comment which represents the start of an at-bat scene and then employs its commented time as the start time of the at-bat scene, and also voice-recognizes a play-by-play comment which represents the end of an at-bat scene and then employs its commented time as the end time of the at-bat scene.

3.2 Event Tag Extraction per At-Bat Scene

To divide a baseball video into multiple sub-videos per at-bat scene, the system requires event tags per at-bat scene (event) E_i $(i = 1, 2, \ldots, N)$ of a baseball game. When a targeted baseball video is input, the system requires event tag per at-bat scene (event) E_i $(i = 1, 2, \ldots, N)$ of the baseball game, to know about what events happened in the baseball game and enable a user to collect only the specific scenes that s/he wants to watch in the targeted baseball video. This process is shown as Step 1 of Fig. 3.1 and Step 1 of Fig. 3.4. The event tags of at-bat scenes (events) of a targeted baseball video are automatically extracted from its ball-by-ball textual report on the Web [33]. The web page [33] is the top of ball-by-ball textual reports per game, which enable those who cannot watch a baseball game live or on TV in real time to capture the information for the baseball game quickly. And a ball-by-ball textual report per game has all pitching's results per pitch in the game. Here, Fig. 3.2 shows the structure of the web site [33] of ball-by-ball textual reports per game, and an image of pitching's result per pitch in a game (e.g., the pitching's result for the 144th pitch in the Game I). A pitching's result in the ball-by-ball textual report about a game, which is updated per pitch in the game, has the score board of the game up to the pitch (e.g., Team A has gotten 4 runs and Team B has gotten 3 runs), the batter's information (e.g., the batter is YAMADA and his batting average is 0.275), the pitcher's information (e.g., the pitcher is SATO and his ERA is 3.15), the on-base runner's information (e.g., 3rd runner is SUZUKI), the pitching's information (e.g., the 144th pitching is Straight at 145 km/h and the result is Right-fly) and so forth.

If a viewer wants to watch all scenes of a player A in the baseball video, the viewer collects the only scenes whose tag information contains "player A." In this case, the system appends tag information relevant to an at-bat scene, which indicates how the player A participates in the scene, even if the scene is not an at-bat scene of the player A and the player A participates little in the scene. Therefore, as event tags for an event E_i, the system requires not only the batter-name of the event, the situation when the batter stepped to the plate, and the result of the event, but also the names of the other players who participated even if a little in the at-bat scene, and how they participated.

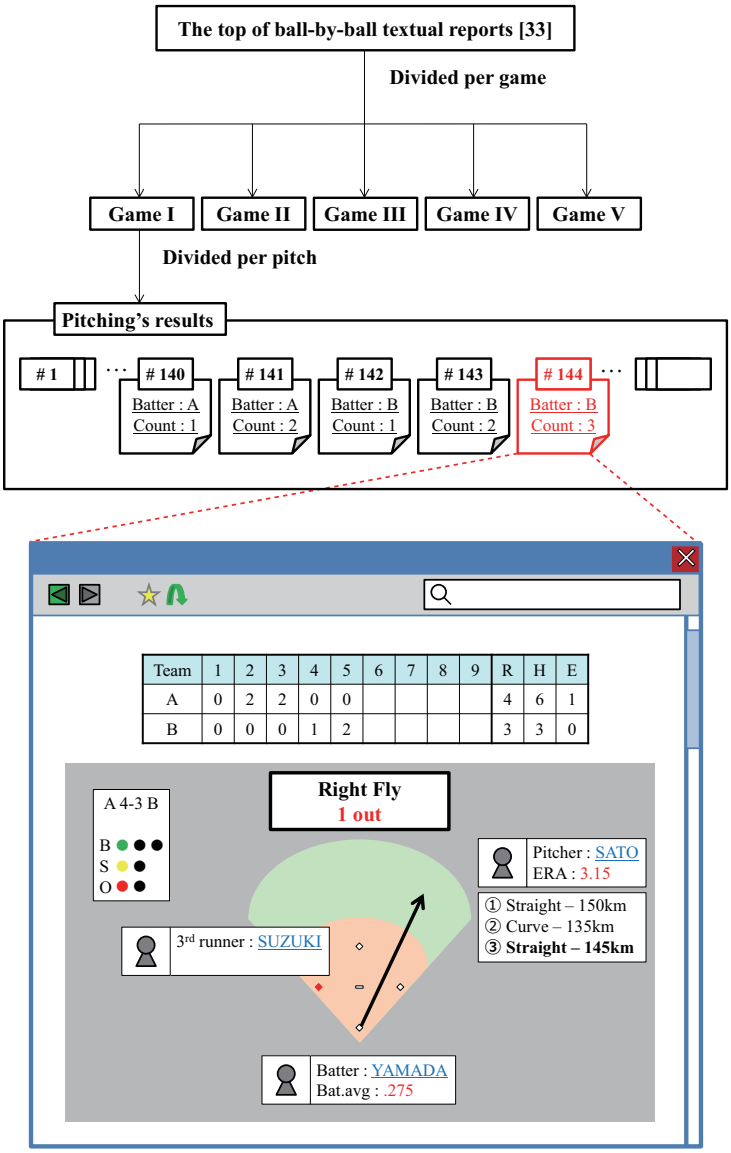

Fig. 3.2 Ball-by-ball textual reports on the Web.

3.3 Event Time Appending

This section describes in detail an appending method for the event time (i.e., the event's start time and the event's end time) of every event (at-bat scenes) for a targeted baseball video.

First, the event's start time T_1 of the first event E_1 and the event's end time T'_N of the last event E_N are exceptionally calculated. Because I suppose that the system is loaded to the television recorder, the system can judge where the game starts in the video time based on the relationship between the clock time when the television program starts and the clock time when the game starts.

This process is shown as Fig. 3.3. In an instance of Fig. 3.3, by extracting the information that the game starts at 18:00 from its ball-by-ball textual report on the Web, and the information that its television program starts at 17:50 from the television recorder, the system can judge that the event's start time T_1 of the first event E_1 of the game is 10 minutes after its baseball video starts.

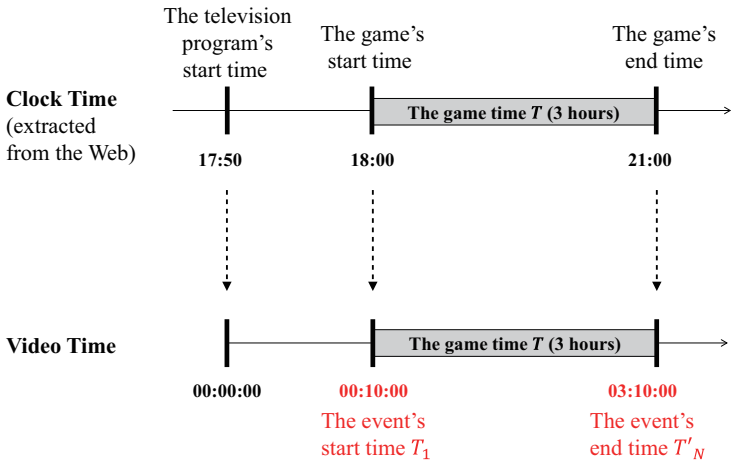

Fig. 3.3 Calculation of the event's start time T_1 (the game's start time) and the event's end time T'_N (the game's end time).

In addition, by extracting the information that the duration of the game equals to T from its ball-by-ball textual report on the Web, the system can also judge that the event's end time T'_N of the last event E_N of the game equals to $T_1 + T$. The event's start time T_i $(i = 2, 3, \ldots, N)$ and the event's end time T'_i $(i = 1, 2, \ldots, N-1)$ of the other events are appended by using the results of play-by-play voice recognition for a targeted baseball video. This process is shown as Steps 2-6 of Fig. 3.4.

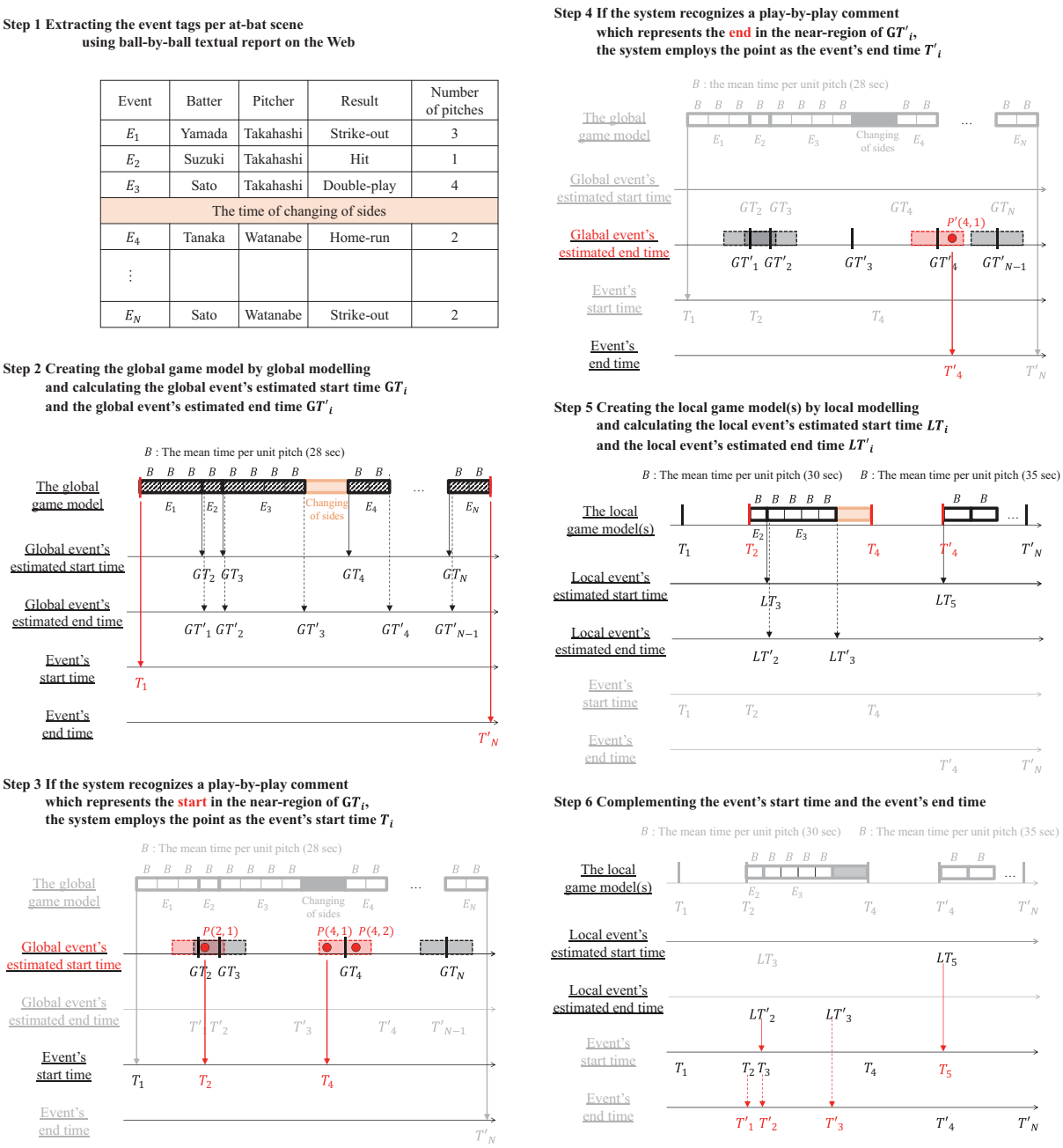

Fig. 3.4 A flow of appending event's start/end time.

In Step 2 of Fig. 3.4, the system creates the global game model for the whole of a targeted baseball game by Global Modelling based on all the event tags that are extracted in Step 1 of Fig. 3.4. A game model shows the structure of a set of at-bat scenes of a baseball game, e.g., what scenes the game has, and how much time each scene has. In this paper, game models for a baseball game include the global game model and the local game model(s). The global game model shows the structure of the universal set of all at-bat scenes of a baseball game, while the local game model shows the structure of a subset of two or more at-bat scenes of a baseball game. In an instance of Step 2 of Fig. 3.4, the event E_1 uses $B \times 3$ [sec], the event E_2 uses $B \times 4$ [sec], and the event E_3 uses $B \times 1$ [sec] (where B

denotes the mean time per unit pitch in the whole of a targeted baseball game), and there is a time for the changing of batting and fielding sides after the event E_3. By creating the global game model, the system calculates the global event's estimated start time $G\hat{T}_i$ $(i = 2, 3, \ldots, N)$, which is globally estimated about the event's start time T_i of an event E_i, and the global event's estimated end time $G\hat{T'}_i$ $(i = 1, 2, \ldots, N-1)$, which is globally estimated about the event's end time T'_i of an event E_i.

In Step 3 of Fig. 3.4, to populate as many events as possible with their event's start time, the system searches for one voice-recognized play-by-play comment which represents the start of an event E_i (at-bat scene) in the near-region of its global event's estimated start time $G\hat{T}_i$. If there is only one voice-recognized play-by-play comment $P(i, j)$ which represents the start of an event E_i, the system employs the time when the play-by-play comment is voice-recognized as the event's start time T_i. Here, "j" means the order of the voice-recognized play-by-play comment that represents the start of an event E_i (at-bat scene), i.e., $P(i, j)$ is the j-th voice-recognized play-by-play comment that represents the start of an event E_i. Else if there are two or more voice-recognized play-by-play comments which represent the start of an event E_i, one is selected from among them based on their "Priority." In this paper, the priority of a voice-recognized play-by-play comment which represents the start of an event E_i is calculated based on whether or not it is only the batter-name of the event (in general, the priority should indicate how appropriately it represents the start of an event). A voice-recognized play-by-play comment which is only the batter's name has a lower priority than a voice-recognized play-by-play comment which is not only the batter's name and contains the other description (e.g., his order and/or position, "Welcome," like in Table 3.1).

In Step 4 of Fig. 3.4, to populate as many events as possible with their event's end time, the system searches for one voice-recognized play-by-play comment which represents the end of an event E_i in the near-region of the global event's estimated end time $G\hat{T'}_i$. If there is only one voice-recognized play-by-play comment $P'(i, j)$ which represents the end of an event E_i, the system employs the time when the play-by-play comment is voice-recognized as the event's end time T'_i. Here, "j" means the order of the voice-recognized play-by-play comment that represents the end of an event E_i (at-bat scene), i.e., $P'(i, j)$ is the j-th voice-recognized play-by-play comment that represents the end of an event E_i. Else if there are two or more voice-recognized play-by-play comments which represent the end of an event E_i, one is selected from among them based on their "Priority." In this paper, the priority of a voice-recognized play-by-play comment which represents the end of an event E_i is calculated based on whether or not it is only the name of player who caught the batted ball in the event (in general, the priority should indicate how appropriately it represents the end of an event). A voice-recognized play-by-play comment which is only the player's name has lower priority than a voice-recognized play-by-play comment which

is not only the player's name and/or contains the other description (e.g., the direction of the batted ball, and the count of outs, like in Table 3.1).

In Step 3 and Step 4, the system can append the event's start time T_i and the event's end time T'_i of only the events E_i that have one or more voice-recognized play-by-play comment(s) which represent the start/end of an event E_i in the near-region of the global event's estimated start/end time $G\hat{T}_i$ or $G\hat{T}'_i$.

In Step 5 of Fig. 3.4, the system creates the local game model(s) for a part of a targeted baseball game by Local Modelling based on two edge events to which have already been appended the event's start time and/or end time (in Steps 3/4) and the event(s) between them if any. In global modelling, the system calculates the mean time B per unit pitch that is applied to the entire game, and creates the game model globally, because the system has only the event's start time T_1 of the first event E_1 and the event's end time T'_N of the last event E_N. On the other hand, in local modelling, the system can calculate the mean time B per unit pitch that is applied partially to each of some parts of a game, and creates the game model(s) locally, because the system has the event's start/end time of the two edge events whose event time has already been appended by Step(s) 3/4. By creating the game model(s) locally, the system calculates the local event's estimated start time $L\hat{T}_i$ $(i = 2, 3, \ldots, N)$, which is locally estimated about the start time T_i of an event E_i, and the local event's estimated end time $L\hat{T}'_i$ $(i = 1, 2, \ldots, N-1)$, which is locally estimated about the end time T'_i of an event E_i.

Finally, in Step 6 of Fig. 3.4, the system ends up populating all the events with their event's start/end time by complementing based on locally-estimated event's start/end time for only the events whose event's start time and/or end time has not yet been appended.

3.4 Play-by-Play Point Prioritization and Searching for At-Bat Start/End Play-by-Play Point

In this paper, the system has been given in advance multiple kinds of play-by-play comment patterns (in Japanese) which represent the start/end of at-bat scenes to search for the voice-recognized play-by-play comment(s) that represent the start/end of an event E_i, and recognizes the play-by-play voice of a targeted baseball game using AmiVoice [34] as a voice recognition software.

Table 3.1 shows examples of the play-by-play comment patterns (in Japanese and translated into English). The play-by-play comment patterns that represent the start of an event (at-bat scene) contain a combination of three kinds of event tags, "batter's order," "batter's position," and "batter's name," among ones that are automatically extracted from ball-by-ball textual reports on the Web [33]. Meanwhile, the play-by-play comment patterns that represent the end of an event (at-bat scene) contain a combination of several kinds of event tags, "out-count" (if the batter of the event is out), and "the name of player who caught the batted ball" (if there is a defense chance in the at-bat scene), and the other "pitching's result" (e.g., the direction of batted ball), among ones that are automatically extracted from ball-by-ball textual reports on the Web.

The voice-recognized play-by-play comments that represent the start/end of an at-bat scene include two kinds of play-by-play points (comments): "Superior Play-by-Play Point" and "Inferior Play-by-Play Point." These points in detail are shown as follows, and Table 3.2 shows instances of applying these points to an event E_i.

Table 3.1 Examples of play-by-play comment patterns which represent the start/end of at-bat scenes.

Comment patterns for the at-bat start
[打順][ポジション][打者名] です [*1]
[打者名] を迎えます [*2]
[ポジション][打者名] [*3]

[*1]: It is [batter's order], [his position], [his name].
[*2]: Welcome, [batter's name].
[*3]: [his position][batter's name].

Comment patterns for the at-bat end
[打球方向][処理した野手] [*4]
[1-3] アウトです [*5]

[*4]: [ball's direction][name of player who caught].
[*5]: [1-3] out(s).

Table 3.2 Instances of superior/inferior play-by-play points which represent the start/end of at-bat scenes.

Instances for the at-bat start

Superior point	Inferior point
9 番センター山田です [*6] 山田を迎えます [*7] 9 番山田 [*8] etc.	山田 [*9]

[*6]: It is 9th, Center, Yamada.
[*7]: Welcome, Yamada.
[*8]: 9th, Yamada.
[*9]: Yamada.

Instances for the at-bat end

Superior point	Inferior point
サード鈴木 [*10] 2 アウトです [*11] etc.	鈴木 [*12]

[*10]: Third, Suzuki.
[*11]: 2 outs.
[*12]: Suzuki

- Superior Play-by-Play Point: is a voice-recognized play-by-play point (comment) which **completely matches one of the play-by-play comment patterns** for voice recognition which represent the start/end of an at-bat scene. It has the highest priority, i.e., a higher priority than Inferior Play-by-Play Points have.

- Inferior Play-by-Play Point: is a voice-recognized play-by-play point which **represents only the player name** on an at-bat scene, which is **only a part of play-by-play comment patterns** for voice recognition. It has a lower priority than Superior Play-by-Play Points have, because its condition to fulfill is looser, and thus, its player name was not always commented on the at-bat scene. However, it has a higher priority than the other play-by-play points (noises) have. Here, the Inferior Play-by-Play Point for the start of an at-bat scene E_i is the batter name of the at-bat scene, while the Inferior Play-by-Play Point for the end of an at-bat scene E_i is the name of the player who caught the batted ball.

The remainder of this section shows a searching method for voice-recognized play-by-play comment(s) which represent the start/end of an at-bat scene. The play-by-play points of an event E_i that are superior or inferior play-by-play points of the at-bat start are defined as **start play-by-play points** $P(i,j)$ $(j = 1, 2, \ldots)$ in order of their appearing. Meanwhile, the play-by-play points of an event E_i which are superior or inferior play-by-play points of the at-bat end are defined as **end play-by-play points** $P'(i,j)$ $(j = 1, 2, \ldots)$ in order of their appearing.

A play-by-play point P has its **recognized time** P.time, and its **priority** P.priority as its properties. The recognized time P.time $\in [T_1, T'_N]$ shows when the play-by-play point P was commented in a targeted baseball video, while the priority P.priority $\in [0.0, 1.0]$ shows how superior the play-by-play point P is as a play-by-play comment on the at-bat start or the at-bat end and is defined as follows in this paper.

$$P.\text{priority} = \begin{cases} 1.0 & \text{(Superior point)} \\ 0.5 & \text{(Inferior point)} \end{cases}$$

To append the event's start time T_i to an event E_i, the system searches for the start play-by-play point $P(i,j)$ in the near-region of the global event's estimated start time $G\hat{T}_i$, which is calculated by Step 2 of Fig. 3.4, and employs the recognized time $P(i,j)$.time as the event's start time T_i. Meanwhile, to append the event's end time T'_i to an event E_i, the system also searches for the end play-by-play point $P'(i,j)$ in the near-region of the global event's estimated end time $G\hat{T}'_i$, which is calculated by Step 2 of Fig. 3.4, and employs the recognized time $P'(i,j)$.time as the event's end time T'_i.

Fig 3.5 shows a flow of selecting the start play-by-play point $P(i,j)$ of an event E_i and finally appending the event's start time T_i to the event E_i. Here, the system establishes the searching region for the start play-by-play point $P(i,j)$ as $G\hat{T}_i - \Delta t_1 \sim G\hat{T}_i + \Delta t_2$ (where $G\hat{T}_i + \Delta t_2 \leq$ the game time T) using the global event's estimated start time $G\hat{T}_i$ and parameters Δt_1 and Δt_2. The finally selected play-by-play point $P \in \{P(i,j)\}$ as the event's start time T_i in the searching region meets two requirements as follows, and it is the start play-by-play point that has the least time in the play-by-play points that have the highest priority in the searching region.

1. P.priority $\geq \forall P(i,j)$.priority
2. P.time $\leq \forall P(i,j)$.time where $P(i,j)$.priority $= P$.priority

The system also establishes the searching region of the end play-by-play point $P'(i,j)$ as $G\hat{T'}_i - \Delta t_1 \sim G\hat{T'}_i + \Delta t_2$ (where $G\hat{T'}_i + \Delta t_2 \leq$ the game time T) using the global event's estimated end time $G\hat{T}'_i$ and the same parameters Δt_1 and Δt_2. The finally selected point $P' \in \{P'(i,j)\}$ as the event's end time T'_i in the searching region meets three requirements as follows, and it is the end play-by-play point that has the least time in the play-by-play points that have the highest priority. Here, the recognized time P'.time of the selected point P' is greater than the recognized time P.time of the start play-by-play point P.

1. P'.priority $\geq \forall P'(i,j)$.priority
2. P'.time $\leq \forall P'(i,j)$.time where $P'(i,j)$.priority $= P'$.priority
3. P'.time $> P$.time

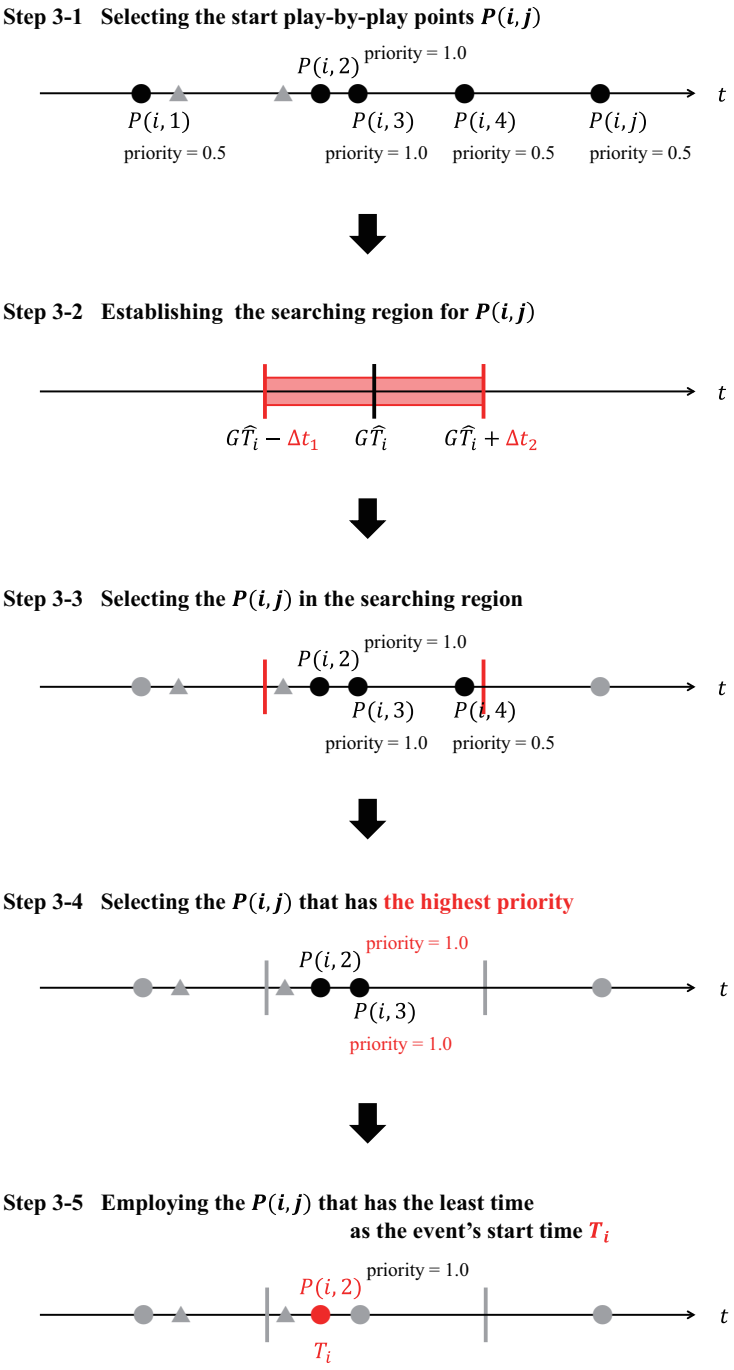

Fig. 3.5 A flow of selecting the start play-by-play point $P(i,j)$.

3.5 Recursive Model Localization and Event Time Estimation Based on Global/Local Model

In Step 2 and Step 5 of Fig. 3.4, the system creates the global game model by Global Modelling and the local game model(s) by Local Modelling, to calculate the global/local event's estimated start/end time for all the events in a target baseball video, which is globally/locally estimated about the event's start/end time, respectively. Fig. 3.6 shows examples for each type of modelling.

- In global modelling, the system creates the game model globally **for the whole of a targeted baseball game**, i.e., the system calculates the unified mean time B_G per unit pitch that is applied to the entire game by using only the game's start time (the event's start time T_1 of the first event E_1) and the game's end time (the event's end time T'_N of the last event E_N).
- In local modelling, which is newly proposed in this paper, the system creates the game model(s) locally **for a part of a targeted baseball game**, i.e., the system can calculate the distinct mean time B_L per unit pitch that is applied partially to each of some parts of a game by using the event's start/end time of the two edge events whose event time has been already appended by Step 3 and/or Step 4 of Fig. 3.4.

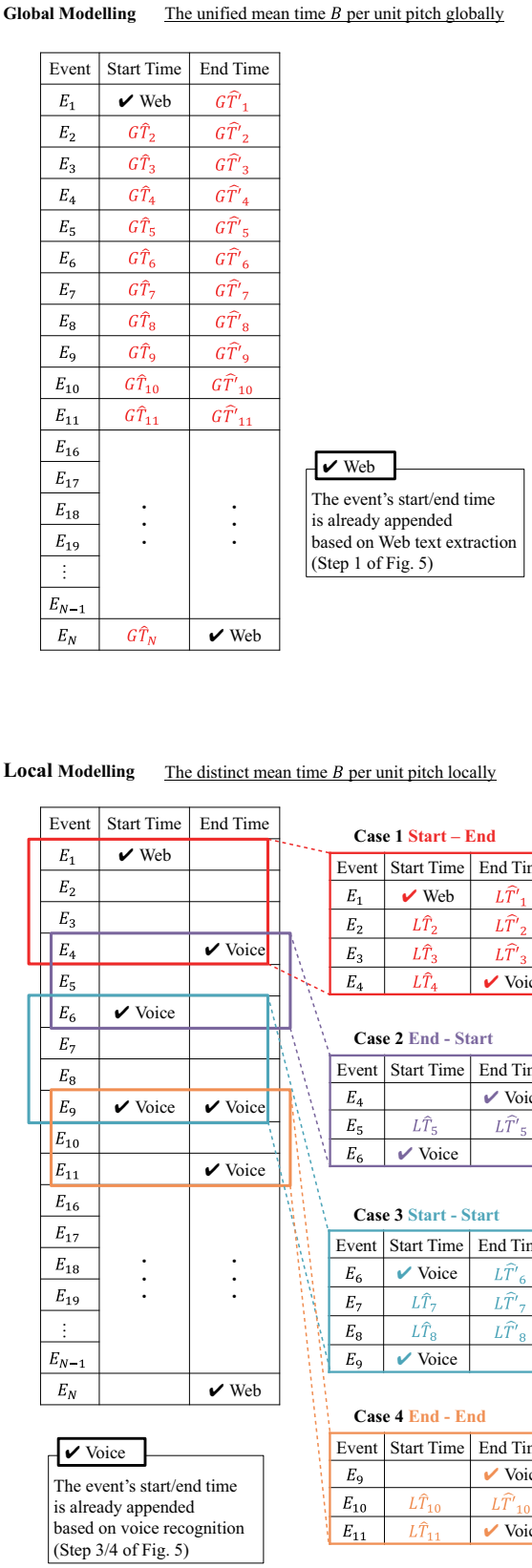

Fig. 3.6 Global modelling and local modelling (four cases).

The local event's estimated start/end time $L\hat{T}_i$ or $L\hat{T'}_i$ by Step 5 could be expected to be more precise for an event E_i than the global event's estimated start/end time $G\hat{T}_i$ or $G\hat{T'}_i$ by Step 2, and finally the next Step 6 could be expected to more precisely complement the event's start/end time only for the events whose event's start time and/or end time has not yet been appended. The local modelling for a part of a baseball game has four cases depending on two edge events E_x and E_{x+n} of the part to which have already been appended the event's start time and/or end time: Case 1) Start time - End time; Case 2) End time - Start time; Case 3) Start time - Start time; Case 4) End time - End time.

The next section shows one case of the globally-modelling method and four cases of the locally-modelling method to create the global game model and the local game model(s), i.e., to calculate the mean time B_G and B_L per unit pitch in the entire game and in a part of the game, and then the global/local event's estimated start/end time, respectively.

First, four kinds of functions that are used commonly in any kind of modelling are defined sequentially. The 1st function $\mathrm{cs}(E_i)$ shows whether or not an event E_i of a targeted baseball video is the preceded event by a change of batting and fielding sides, and is extracted from its ball-by-ball textual report on the Web.

$$\mathrm{cs}(E_i) = \begin{cases} 1.0 & (E_i \text{ is preceded by a change of sides}) \\ 0.0 & (\text{otherwise}) \end{cases}$$

The 2nd function $\mathrm{cp}(E_i)$ shows whether or not an event E_i occurs right after changing a pitcher, and is also extracted from its ball-by-ball textual report on the Web.

$$\mathrm{cp}(E_i) = \begin{cases} 1.0 & (E_i \text{ follows after changing a pitcher}) \\ 0.0 & (\text{otherwise}) \end{cases}$$

The 1st function $\mathrm{cs}(E_i)$ and the 2nd function $\mathrm{cp}(E_i)$ are introduced to refine the mean time per unit pitch, because the changing time of batting and fielding sides and changing time of a pitcher are not a part of at-bat scenes and must be removed.

The 3rd function $\mathrm{W_l}(E_i)$ shows whether or not the inning of an event occurs during the late innings (in this paper, after the seventh inning), and assigns the pitches of events (at-bat scenes) after the seventh inning with a higher weight w_l. The inning in which each event of a baseball game happened is also extracted from its ball-by-ball textual report on the Web, and the weight based on the parameter w_l (≥ 1.0) is added to the pitching-time per unit pitch by using the following function:

$$\mathrm{W_l}(E_i) = \begin{cases} w_l & (E_i \text{ is after the seventh inning}) \\ 1.0 & (\text{otherwise}) \end{cases}$$

The last function $\mathrm{W_r}(E_i)$ shows whether or not there is a runner on a base during an event, and assigns the pitches of events (at-bat scenes) in which there is a runner on a

base with a higher weight w_r. The existence of runner(s) when the batter of an event of a baseball game stepped to the plate is also extracted from its ball-by-ball textual report on the Web, and the weight based on the parameter w_r (≥ 1.0) is added to the pitching-time per unit pitch by using the following function:

$$\mathrm{W_r}(E_i) = \begin{cases} w_r & \text{(there is a runner)} \\ 1.0 & \text{(otherwise)} \end{cases}$$

The 3rd function $\mathrm{W_l}(E_i)$ and the last function $\mathrm{W_r}(E_i)$ are introduced to discriminate between the mean time per unit pitch for events that fulfill the condition of $\mathrm{W_l}(E_i)$ (when the inning of an event E_i occurs during the late innings) or $\mathrm{W_r}(E_i)$ (when there is a runner on a base during an event E_i) and the mean time per unit pitch for the other events that do not fulfill the condition of $\mathrm{W_l}(E_i)$ or $\mathrm{W_r}(E_i)$.

3.6 Calculation Method for Models

This section explains the calculation method of estimating (modelling) each event (at-bat scene) in a targeted baseball video. Fig. 3.7 shows an instance of the calculation method of estimating events. The modelling, i.e., estimating the event time (event's start time and end time) for each event, has three steps. Step 1 determines the range in which the system creates a model. In the instance of Fig. 3.7, the system creates a model in the range from the event E_1 to the event E_3. Step 2 calculates the mean time per unit pitch by using the information about only the events in the range that is determined by Step 1, i.e., their event tags extracted from the ball-by-ball textual report on the Web. Step 2 calculates the mean time per unit pitch by using only the events in the range that is determined by Step 1. The system utilizes the number of pitches that are used in an event (at-bat scene) and the four kinds of functions defined in the previous section, to calculate the mean time per unit pitch. In the instance of Fig. 3.7, the event E_2 is right after changing a pitcher, and there is a runner on a base during the events E_2 and E_3. In this case, the system gets 22.06 [sec] as the calculated mean time B per unit pitch only in the range like Fig. 3.7. Step 3 calculates the estimated event time for each event based on the mean time B per unit pitch calculated by Step 2 and the event tags in the range. The instance of Fig. 3.7 shows the process of calculating the event's estimated start time $\hat{T}_3$ and event's estimated end time $\hat{T'}_3$ of the event E_3, i.e., from 227.65 [sec] to 360.00 [sec].

The rest of this section explains the calculation method based on each model (Global Modelling or Local Modelling in four kinds of cases).

Creating the model in the range
whose necessary time of a range is already calculated

Event	Event Tags				Necessary Time
	# of Pitches	Result	Aft 7	Run	
E_1	4	Hit	-	-	
Changing a pitcher					
E_2	3	Strike-out	-	✓	
E_3	5	Home-run	-	✓	
Total	12	-	-	-	360 [sec]

Calculating the mean time per unit pitch
by using only the events in the range

Event	Event Tags				Necessary Time
	# of Pitches	Result	Aft 7	Run	
E_1	4	Hit	-	-	
Changing a pitcher					
E_2	3	Strike-out	-	✓	
E_3	5	Home-run	-	✓	
Total	12	-	-	-	360 [sec]

When the system defines the necessary time of changing a pitcher as 60 [sec]
and the weight that is added to the pitching-time per unit pitch as 1.2,
the mean time B per unit pitch is calculated as the following formula:

$$B = \frac{\text{The total of necessary time of each event}}{\text{The total of weighted number of pitches of each event}}$$
$$= \frac{360 - 60}{4 + 3 \times 1.2 + 5 \times 1.2} \approx 22.06$$

Calculating the necessary time of each event
based on the mean time per unit pitch

Event	Event Tags				Necessary Time
	# of Pitches	Result	Aft 7	Run	
E_1	4	Hit	-	-	22.06×4
Changing a pitcher					60
E_2	3	Strike-out	-	✓	$(22.06 \times 1.2) \times 3$
E_3	5	Home-run	-	✓	$(22.06 \times 1.2) \times 5$
Total	12	-	-	-	360 [sec]

For instance, the estimated start time $\hat{T}_3$ of the event E_3
and the estimated end time $\hat{T}'_3$ of the event E_3
are calculated as the following formulas:

$$\hat{T}_3 \approx 22.06 \times 4 + 60 + (22.06 \times 1.2) \times 3$$
$$\hat{T}'_3 \approx 22.06 \times 4 + 60 + (22.06 \times 1.2) \times 3 + (22.06 \times 1.2) \times 5$$

Fig. 3.7 An instance of the calculation method of estimating events' start/end time by modelling.

0) Global Modelling

The temporal interval T for calculating the global weighted mean time B_G per unit pitch as the global game model between the event E_1 whose start time T_1 has been appended exceptionally (extracted from the ball-by-ball textual report) and the event E_N whose end time T'_N has been also appended exceptionally is defined as follows:

$$T = T'_N - T_1$$

The global weighted mean time B_G per unit pitch uses a parameter Δt_s, which is a uniform necessary time of a change of batting and fielding sides and a parameter Δt_p, which is a uniform necessary time of changing a pitcher, and is calculated by the following formula:

$$B_G = \frac{T - \Delta t_s \times \sum_{i=1}^{N} \mathrm{cs}(E_i) - \Delta t_p \times \sum_{i=1}^{N} \mathrm{cp}(E_i)}{\sum_{i=1}^{N} \beta_i \times \mathrm{W_l}(E_i) \times \mathrm{W_r}(E_i)}$$

Finally, the globally-estimated start time $G\hat{T}_i$ of an event E_i $(i = 2, \ldots, N)$ whose event's start time has not yet been appended is calculated by the following formula with the global weighted mean time B_G per unit pitch:

$$G\hat{T}_i = T_1 + \Delta t_s \times \sum_{j=1}^{i} \mathrm{cs}(E_j) + \Delta t_p \times \sum_{j=1}^{i} \mathrm{cp}(E_j) + B_G \times \sum_{j=1}^{N} \beta_j \times \mathrm{W_l}(E_j) \times \mathrm{W_r}(E_j)$$

Then, the globally-estimated end time $G\hat{T}'_i$ of an event E_i $(i = 1, \ldots, N-1)$ whose event's end time has not yet been appended is calculated by the following formula:

$$G\hat{T}'_i = G\hat{T}_{i+1} - \Delta t_s \times \mathrm{cs}(E_{i+1}) - \Delta t_p \times \mathrm{cp}(E_{i+1})$$

1) Local Modelling (Case 1: Start time – End time)

In the Case 1 of local modelling, the event's start time of the head E_x of two edge events of a part of a baseball game has already been appended by Step 3 of Fig. 3.4, and the event's end time of the tail E_{x+n} of two edge events has already been appended by Step 4. Because the event's end time of the head E_x, the event's start time of the tail E_{x+n}, and both the event's start time and end time of the event(s) between them if any have not yet been appended, they are finally assigned the local event's estimated start/end time $L\hat{T}_i$ or $L\hat{T}'_i$ respectively. Here, if and only if the start time of the head E_x is less than the end time of the tail E_{x+n}, the system creates the local game model in the Case 1.

The temporal interval T for calculating the local weighted mean time B_L per unit pitch as a local game model between the event E_x whose start time T_x has been appended by Step 3 and the event E_{x+n} whose end time T'_{x+n} has been appended by Step 4 is defined as follows:

$$T = T'_{x+n} - T_x \quad (\text{where } T'_{x+n} > T_x)$$

The local weighted mean time B_L per unit pitch is calculated by the following formula:

$$B_L = \frac{T - \Delta t_s \times \sum_{i=x+1}^{x+n} \mathrm{cs}(E_i) - \Delta t_p \times \sum_{i=x+1}^{x+n} \mathrm{cp}(E_i)}{\sum_{i=x}^{x+n} \beta_i \times \mathrm{W_l}(E_i) \times \mathrm{W_r}(E_i)}$$

Finally, the locally-estimated start time $\hat{LT_i}$ of an event E_i $(i = x+1, \ldots, x+n)$ whose start time has not yet been appended is calculated by the following formula with the local weighted mean time B_L per unit pitch:

$$\begin{aligned} \hat{LT_i} = T_x &+ \Delta t_s \times \sum_{j=x+1}^{i} \mathrm{cs}(E_j) + \Delta t_p \times \sum_{j=x+1}^{i} \mathrm{cp}(E_j) \\ &+ B_L \times \sum_{j=x}^{i-1} \beta_j \times \mathrm{W_l}(E_j) \times \mathrm{W_r}(E_j) \end{aligned}$$

Then, the locally-estimated end time $\hat{LT'_i}$ of an event E_i $(i = x, \ldots, x+n-1)$ whose end time has not yet been appended is calculated by the following formula:

$$\hat{LT'_i} = \hat{LT_{i+1}} - \Delta t_s \times \mathrm{cs}(E_{i+1}) - \Delta t_p \times \mathrm{cp}(E_{i+1})$$

However, if the temporal interval T has the following inequality with the values of the parameter Δt_s and Δt_p, the local weighted mean time B_L is less than 0. Therefore, in the case that the local weighted mean time B_L is less than 0, that is to say, the temporal interval T meets the following formula, the local weighted mean time B_L per unit pitch has to be exceptionally calculated.

$$T - \Delta t_s \times \sum_{i=x+1}^{x+n} \mathrm{cs}(E_i) - \Delta t_p \times \sum_{i=x+1}^{x+n} \mathrm{cp}(E_i) < 0$$

To be precise, the local weighted mean time B_L per unit pitch is calculated by the following formula:

$$B_L = \frac{T}{\sum_{i=x}^{x+n} \beta_i \times \mathrm{W_l}(E_i) \times \mathrm{W_r}(E_i)}$$

Subsequently, the locally-estimated start time $\hat{LT}_i$ of an event E_i $(i = x+1, \ldots, x+n)$ whose start time has not yet been appended is calculated by the following formula with the local weighted mean time B_L per unit pitch:

$$\hat{LT}_i = T_x + \ B_L \times \sum_{j=x}^{i-1} \beta_j \times \mathrm{W_l}(E_j) \times \mathrm{W_r}(E_j)$$

Then, the locally-estimated end time $\hat{LT}'_i$ of an event E_i $(i = x, \ldots, x+n-1)$ whose end time has not yet been appended is calculated by the following formula:

$$\hat{LT}'_i = \hat{LT}_{i+1}$$

2) Local Modelling (Case 2: End time – Start time)

In the Case 2 of local modelling, the event's end time of the head E_x of two edge events of a part of a baseball game has already been appended by Step 4 of Fig. 3.4, and the event's start time of the tail E_{x+n} of two edge events has already been appended by Step 3. Because both the event's start time and end time of the event(s) between them if any have not yet been appended, they are finally assigned the local event's estimated start/end time $\hat{LT}_i$ or $\hat{LT}'_i$ respectively. Here, if and only if the end time of the head E_x is less than the start time of the tail E_{x+n}, the system creates the local game model in the Case 2.

The temporal interval T for calculating the local weighted mean time B_L per unit pitch as a local game model between the event E_x whose end time T'_x has been appended by Step 4 and the event E_{x+n} whose start time T_{x+n} has been appended by Step 3 is defined as follows:

$$T = T_{x+n} - T'_x \quad (\text{where } T_{x+n} > T'_x)$$

The local weighted mean time B_L per unit pitch is calculated by the following formula:

$$B_L = \frac{T - \Delta t_s \times \sum_{i=x+1}^{x+n} \mathrm{cs}(E_i) - \Delta t_p \times \sum_{i=x+1}^{x+n} \mathrm{cp}(E_i)}{\sum_{i=x+1}^{x+n-1} \beta_i \times \mathrm{W_l}(E_i) \times \mathrm{W_r}(E_i)}$$

Finally, the locally-estimated start time $\hat{LT}_i$ of an event E_i $(i = x+1, \ldots, x+n-1)$ whose start time has not yet been appended is calculated by the following formula with the local weighted mean time B_L per unit pitch:

$$\hat{LT}_i = T'_x + \Delta t_s \times \sum_{j=x+1}^{i} \mathrm{cs}(E_j) + \Delta t_p \times \sum_{j=x+1}^{i} \mathrm{cp}(E_j)$$
$$+ \ B_L \times \sum_{j=x+1}^{i-1} \beta_j \times \mathrm{W_l}(E_j) \times \mathrm{W_r}(E_j)$$

Then, the locally-estimated end time $L\hat{T}'_i$ of an event E_i $(i = x+1, \ldots, x+n-1)$ whose end time has not yet been appended is calculated by the following formula:

$$L\hat{T}'_i = L\hat{T}_{i+1} - \Delta t_s \times \mathrm{cs}(E_{i+1}) - \Delta t_p \times \mathrm{cp}(E_{i+1})$$

However, if the temporal interval T has the same inequality with the values of the parameter Δt_s and the parameter Δt_p, that is to say, if the local weighted mean time B_L is less than 0, the system processes exceptionally like described in the last paragraph of the Case 1.

3) Local Modelling (Case 3: Start time – Start time)

In the Case 3 of local modelling, the event's start time of the head E_x of two edge events of a part of a baseball game has already been appended by Step 3 of Fig. 3.4, and the event's start time of the tail E_{x+n} of two edge events has already been appended by Step 3. Because the event's end time of the head E_x, and both the event's start time and end time of the event(s) between them if any have not yet been appended, they are finally assigned the local event's estimated start/end time $L\hat{T}_i$ or $L\hat{T}'_i$ respectively. Here, if and only if the start time of the head E_x is less than the start time of the tail E_{x+n}, the system creates the local game model in the Case 3.

The temporal interval T for calculating the local weighted mean time B_L per unit pitch as a local model between the event E_x whose start time T_x has been appended by Step 3 and the event E_{x+n} whose start time T_{x+n} has been appended by Step 3 is defined as follows:

$$T = T_{x+n} - T_x \quad (\text{where } T_{x+n} > T_x)$$

The local weighted mean time B_L per unit pitch is calculated by the following formula:

$$B_L = \frac{T - \Delta t_s \times \sum_{i=x+1}^{x+n} \mathrm{cs}(E_i) - \Delta t_p \times \sum_{i=x+1}^{x+n} \mathrm{cp}(E_i)}{\sum_{i=x}^{x+n-1} \beta_i \times \mathrm{W_l}(E_i) \times \mathrm{W_r}(E_i)}$$

Finally, the locally-estimated start time $L\hat{T}_i$ of an event E_i $(i = x+1, \ldots, x+n-1)$ whose start time has not yet been appended is calculated by the following formula with the local weighted mean time B_L per unit pitch:

$$L\hat{T}_i = T_x + \Delta t_s \times \sum_{j=x+1}^{i} \mathrm{cs}(E_j) + \Delta t_p \times \sum_{j=x+1}^{i} \mathrm{cp}(E_j) + B_L \times \sum_{j=x}^{i-1} \beta_j \times \mathrm{W_l}(E_j) \times \mathrm{W_r}(E_j)$$

Then, the locally-estimated end time $L\hat{T}'_i$ of an event E_i $(i = x, \ldots, x+n-1)$ whose end time has not yet been appended is calculated by the following formula:

$$L\hat{T}'_i = L\hat{T}_{i+1} - \Delta t_s \times \mathrm{cs}(E_{i+1}) - \Delta t_p \times \mathrm{cp}(E_{i+1})$$

However, if the temporal interval T has the same inequality with the values of the parameter Δt_s and the parameter Δt_p, that is to say, if the local weighted mean time B_L is less than 0, the system processes exceptionally like described in the last paragraph of the Case 1.

4) Local Modelling (Case 4: End time – End time)

In the Case 4 of local modelling, the event's end time of the head E_x of two edge events of a part of a baseball game has already been appended by Step 4 of Fig. 3.4, and the event's end time of the tail E_{x+n} of two edge events has already been appended by Step 4. Because the event's start time of the tail E_{x+n}, and both the event's start time and end time of the event(s) between them if any have not yet been appended, they are finally assigned the local event's estimated start/end time $L\hat{T}_i$ or $L\hat{T}'_i$ respectively. Here, if and only if the end time of the head E_x is less than the end time of the tail E_{x+n}, the system creates the local game model in the Case 4.

The temporal interval T for calculating the local weighted mean time B_L per unit pitch as a local model between the event E_x whose end time T'_x has been appended by Step 4 and the event E_{x+n} whose end time T'_{x+n} has been appended by Step 4 is defined as follows:

$$T = T'_{x+n} - T'_x \quad (\text{where } T'_{x+n} > T'_x)$$

The local weighted mean time B_L per unit pitch is calculated by the following formula:

$$B_L = \frac{T - \Delta t_s \times \sum_{i=x+1}^{x+n} \mathrm{cs}(E_i) - \Delta t_p \times \sum_{i=x+1}^{x+n} \mathrm{cp}(E_i)}{\sum_{i=x+1}^{x+n} \beta_i \times \mathrm{W_l}(E_i) \times \mathrm{W_r}(E_i)}$$

Finally, the locally-estimated start time $L\hat{T}_i$ of an event E_i $(i = x+1, \ldots, x+n)$ whose start time has not yet been appended is calculated by the following formula with the local weighted mean time B_L per unit pitch:

$$L\hat{T}_i = T'_x + \Delta t_s \times \sum_{j=x+1}^{i} \mathrm{cs}(E_j) + \Delta t_p \times \sum_{j=x+1}^{i} \mathrm{cp}(E_j) + B_L \times \sum_{j=x+1}^{i-1} \beta_j \times \mathrm{W_l}(E_j) \times \mathrm{W_r}(E_j)$$

Then, the locally-estimated end time $L\hat{T}'_i$ of an event E_i $(i = x+1, \ldots, x+n-1)$ whose end time has not yet been appended is calculated by the following formula:

$$L\hat{T}'_i = L\hat{T}_{i+1} - \Delta t_s \times \mathrm{cs}(E_{i+1}) - \Delta t_p \times \mathrm{cp}(E_{i+1})$$

However, if the temporal interval T has the same inequality with the values of the parameter Δt_s and the parameter Δt_p, that is to say, if the local weighted mean time B_L is less than 0, the system processes exceptionally like described in the last paragraph of the Case 1.

3.7 Event Time Complementing

In the final Step 6 of Fig. 3.4, the system ends up populating all events with their event's start/end time by complementing based on the globally-estimated event's start/end time for only the events whose event's start time and/or end time have not yet been appended (based on exceptional event time appending using Web-extracted text in Step 1, play-by-play voice recognition in Step 3 and Step 4, or event time estimation by local modelling in Step 5).

1) Event's Start Time Complementing

The complementing method of the event's start time of an event E_i (at-bat scene) whose event's start time has not yet been appended has two kinds of cases. For an event E_i whose event's end time T'_i has been already appended, the system calculates its event's start time T_i by the following formula:

$$T_i = T'_i - B_G \times \mathrm{W_l}(E_i) \times \mathrm{W_r}(E_i) \times \beta_i$$

For an event E_i whose event's end time has NOT yet been appended, the system employs its global event's estimated start time $G\hat{T}_i$ as its event's start time T_i.

$$T_i = G\hat{T}_i$$

2) Event's End Time Complementing

For an event E_i whose event's start time has been already appended, the system calculates its event's end time T'_i by the following formula:

$$T'_i = T_i + B_G \times \mathrm{W_l}(E_i) \times \mathrm{W_r}(E_i) \times \beta_i$$

Chapter 4

Experiment

This chapter evaluates my proposed Tagging algorithm using 4 recorded television videos of rebroadcasted baseball games. To reveal the interlap between an at-bat scene which is computed by the tagging system, and the correct at-bat scene that is defined by the author, the following criteria are considered: recall, precision, F-measure, square error of the event's start time, and square error of the event's end time. The above-defined parameters, Δt_1, Δt_2, Δt_s, Δt_p, w_l, and w_r, are varied in the range, whose upper limit is set to be enough large and increment is set to be enough small based on my heuristics and an analysis of the 4 baseball videos used for the experiment. The experiment simulates my proposed tagging system to discover the optimum combination of parameters from among all combinations of each parameter varying in the range.

- $0 \leq \Delta t_1 \leq 10$ (increments of 1 [min])
- $0 \leq \Delta t_2 \leq 10$ (increments of 1 [min])
- $0 \leq \Delta t_s \leq 360$ (increments of 30 [sec])
- $0 \leq \Delta t_p \leq 360$ (increments of 30 [sec])
- $1.0 \leq w_l \leq 2.0$ (increments of 0.1)
- $1.0 \leq w_r \leq 4.0$ (increments of 0.1)

In addition, the dictionary of AmiVoice [34] used as a voice recognition software is set to be localized per game, and has play-by-play comment patterns which represent the start/end of at-bat scenes not in a general game, but especially in each game. This paper assumes that the accuracy of voice recognition only for these play-by-play comment patterns is sufficient for my tagging system.

Table 4.1 shows 5 kinds of methods based on combinations of the processes of Steps 2-6 in the tagging system. The method (i) has the 2 processes of Step 2 and Step 6 of Fig. 3.4, that is to say, it is the method that the system simply employs the global event's estimated start/end time $G\hat{T}_i$ and $G\hat{T'}_i$, which are calculated by the global modelling in Step 2 of Fig. 3.4 as the event's start/end time T_i and T'_i. The method (ii) is the basic

method that was proposed by the author, that is to say, it is the method that the system employs the batter name's recognized time of an event E_i as the event's start time T_i of the event E_i. The method (iii) has the 3 processes of Step 2, Step 3, and Step 6 of Fig. 3.4, that is to say, it is the method that the system searches for only the voice-recognized play-by-play comment(s) that represent the start of an event E_i (not only the batter name), and employs this comment's recognized time as the event's start time T_i. The method (iv) has the 4 processes of Step 2, Step 3, Step 4, and Step 6 of Fig. 3.4, that is to say, it is the method that the system searches for the voice-recognized play-by-play comment(s) that represent the start/end of an event E_i, and employs these comment's recognized time as the event's start/end time T_i and T_i'. The method (v) has all the proposed steps, that is to say, the system utilizes the voice-recognized play-by-play comment(s) that represent the start/end of an event E_i, and is newly equipped with Local Modelling.

Table 4.1 Tagging methods based on the mandatory Step 1 and different combination of processes (Steps 2-6).

	Step 2	Step 3	Step 4	Step 5	Step 6
(i)	✓	–	–	–	✓
(ii)	✓	–*	–	–	✓
(iii)	✓	✓	–	–	✓
(iv)	✓	✓	✓	–	✓
(v)	✓	✓	✓	✓	✓

Step 1: Event Tag Extraction.
Step 2: Global Modelling.
Step 3: Comments on the start of at-bat scenes.
Step 4: Comments on the end of at-bat scenes.
Step 5: Local Modelling.
Step 6: Event Time Complementing.
*: Voice-recognizing only batter names.

Table 4.2 Tagging accuracy depending on respective methods. (Data 1: 77 at-bat scenes)

Method	Δt_1	Δt_2	Δt_s	Δt_p	w_l	w_r	R	P	F	S-Err	E-Err
(i)	–	–	150	180	1.0	3.2	0.528	0.578	0.552	101.37	112.02
(ii)	1	4	120	180	1.0	2.6	0.595	0.627	0.610	99.14	107.00
(iii)	1	5	120	180	1.0	2.6	0.616	0.650	0.633	93.58	102.58
(iv)	1	5	120	180	1.0	2.6	0.603	0.645	0.623	94.19	101.82
(v)	1	5	90	240	1.0	3.6	0.696	0.624	**0.658**	105.00	100.79

R : Recall / P : Precision / F : F-measure / S-Err : Start Error / E-Err : End Error

Table 4.3 Tagging accuracy depending on respective methods. (Data 2: 65 at-bat scenes)

Method	Δt_1	Δt_2	Δt_s	Δt_p	w_l	w_r	R	P	F	S-Err	E-Err
(i)	–	–	90	180	1.0	2.6	0.429	0.411	0.420	190.88	201.03
(ii)	1	10	30	210	1.1	1.2	0.572	0.514	0.542	171.12	176.72
(iii)	1	9	60	240	1.0	1.2	0.544	0.531	0.537	152.01	160.06
(iv)	2	3	150	180	1.0	3.1	0.488	0.528	0.508	190.13	193.16
(v)	0	10	330	270	1.1	2.5	0.619	0.523	**0.567**	121.85	138.34

R : Recall / P : Precision / F : F-measure / S-Err : Start Error / E-Err : End Error

Table 4.4 Tagging accuracy depending on respective methods. (Data 3: 64 at-bat scenes)

Method	Δt_1	Δt_2	Δt_s	Δt_p	w_l	w_r	R	P	F	S-Err	E-Err
(i)	–	–	90	0	1.0	2.3	0.596	0.507	0.548	72.01	63.36
(ii)	1	3	120	0	1.0	2.5	0.669	0.625	0.646	50.89	55.56
(iii)	0	3	120	0	1.0	2.6	0.729	0.682	0.705	49.24	44.37
(iv)	0	3	60	0	1.0	2.3	0.743	0.662	0.700	52.90	46.12
(v)	0	3	150	30	1.0	2.7	0.727	0.731	**0.729**	42.94	42.27

R : Recall / P : Precision / F : F-measure / S-Err : Start Error / E-Err : End Error

Table 4.5 Tagging accuracy depending on respective methods. (Data 4: 67 at-bat scenes)

Method	Δt_1	Δt_2	Δt_s	Δt_p	w_l	w_r	R	P	F	S-Err	E-Err
(i)	–	–	150	60	1.0	1.6	0.562	0.527	0.544	167.95	158.88
(ii)	1	5	180	60	1.0	1.8	0.602	0.608	0.605	123.28	119.39
(iii)	1	6	180	60	1.0	1.7	0.609	0.615	0.612	121.83	119.59
(iv)	2	5	180	0	1.0	1.8	0.674	0.645	**0.659**	79.37	85.97
(v)	3	2	180	0	1.0	2.4	0.694	0.619	0.654	149.19	132.41

R : Recall / P : Precision / F : F-measure / S-Err : Start Error / E-Err : End Error

Table 4.6 The mean tagging accuracy depending on respective methods.

Method	Δt_1	Δt_2	Δt_s	Δt_p	w_l	w_r	R	P	F	S-Err	E-Err
(i)	–	–	180	90	1.0	2.0	0.380	0.403	0.391	197.57	198.96
(ii)	1	5	150	120	1.0	1.8	0.451	0.467	0.459	189.76	192.49
(iii)	1	8	150	150	1.0	2.1	0.469	0.497	0.482	178.43	180.65
(iv)	1	7	150	150	1.0	2.1	0.526	0.494	0.508	178.12	183.94
(v)	1	7	180	150	1.0	2.4	0.575	0.475	**0.520**	170.67	171.99

R : Recall / P : Precision / F : F-measure / S-Err : Start Error / E-Err : End Error

Tables 4.2–4.5 show the tagging accuracy for the 4 baseball games depending on the respective methods. Furthermore, Table 4.6 shows the mean of tagging accuracies for the 4 baseball games depending on the respective methods. I expect that the square error of the start time becomes smaller along with the change from the method (ii) to the method (iii), because the system utilizes not only the player name of an at-bat scene, but also the voice-recognized play-by-play comment(s) that represent the start of the at-bat scene in the method (iii). Tables 4.2–4.5 reveal that the square errors of the start time for all data become expectedly smaller along with the change from the method (ii) to the method (iii).

And I expect that the square error of the end time becomes smaller along with the change from the method (iii) to the method (iv), because the system also utilizes the voice-recognized play-by-play comment(s) that represent the end of an at-bat scene in the method (iv). Tables 4.2–4.5 reveal that the square errors of the end time for Data 1 and Data 4 become expectedly smaller along with the change from the method (iii) to the method (iv), while the square errors of the end time for Data 2 and Data 3 become unexpectedly larger along with the change from the method (iii) to the method (iv).

In addition, I evaluate the Local Modelling by the square error of the start time and the square error of the end time. Tables 4.2–4.5 reveal that the square errors of the start time for Data 2 and Data 3 become expectedly smaller along with the change from the method (iv) to the method (v), while the square errors of the start time for Data 1 and Data 4 become unexpectedly larger along with the change from the method (iv) to the method (v). Tables 4.2–4.5 also reveal that the square errors of the end time for Data 1, Data 2 and Data 3 become expectedly smaller along with the change from the method (iv) to the method (v), while the square error of the end time for only Data 4 becomes unexpectedly larger along with the change from the method (iv) to the method (v).

Unfortunately, Table 4.6 reveals that the square error of the start/end time becomes only a little smaller along with the change from the method (i) to the method (v). In summary, the experimental results on the square error are different from my expectation, and my future work needs to survey them in more detail, e.g., on their standard deviation,

arithmetical mean, and the error per event.

Subsequently, this chapter discusses their F-measure. Tables 4.2–4.5 reveal that the F-measure rises along with the change from the method (i) to the method (v), except from (iii) to (iv) for Data 1, from (ii) to (iii) for Data 2, from (iii) and (iv) for Data 2, from (iii) to (iv) for Data 3, and from (iv) to (v) for Data 4. Table 4.6 also reveals that the mean F-measure rises along with the change from the method (i) to the method (v).

From the above discussion, to summarize the experimental results on the F-measure, searching for the voice-recognized play-by-play comment(s) that represent the start/end of an at-bat scene as well as its batter name and being newly equipped with the Local Modelling as well as global modelling are effective to improve the F-measure on appending the event's start/end time.

However, to compare the individual F-measures with the mean F-measure of the 4 baseball games, the mean F-measure of the method (v) is lower than the individual F-measure of the method (v) for Data 1 by 0.138, the mean F-measure of the method (v) is lower than the individual F-measure of the method (v) for Data 2 by 0.047, the mean F-measure of the method (v) is lower than the individual F-measure of the method (v) for Data 3 by 0.209, and the mean F-measure of the method (v) is lower than the individual F-measure of the method (v) or method (iv) for Data 4 by 0.134 or 0.139. That is to say, the mean F-measure tends to be lower than the individual F-measure for each baseball game unfortunately.

This disappointing impact would be affected by the value-setting of each parameter which is used to search for the voice-recognized play-by-play comment(s) and to created global/local game models. The individual F-measure for each baseball game when setting the value that maximizes the mean tagging accuracy to each parameter is lower than the individual F-measure for each baseball game when optimizing the parameters to each baseball game. To be explained in detail, the former individual F-measure of the method (v) is lower than the latter individual F-measure of the method (v) optimized to Data 1 by 0.062, the former individual F-measure of the method (v) is lower than the latter individual F-measure of the method (v) optimized to Data 2 by 0.228, the former individual F-measure of the method (v) is lower than the latter individual F-measure of the method (v) optimized to Data 3 by 0.157, the former individual F-measure of the method (v) is lower than the latter individual F-measure of the method (v) optimized to Data 4 by 0.083.

4.1 About the Dependency of Parameters

This section discusses each parameter in detail to solve the above-mentioned problem. Firstly, I discuss the parameters Δt_1 and Δt_2, which determine the searching region of the voice-recognized play-by-play comment(s) that represent the start/end of an at-bat scene. Fig. 4.1 shows the F-measure for each baseball game (Data 1∼4), and Fig. 4.2 shows the square error of the start/end time depending on Δt_1. Here, the other parameters are set as values that maximize the individual F-measures. Fig. 4.3 shows the F-measure for each baseball game (Data 1∼4), and Fig. 4.4 shows the square error of the start/end time depending on Δt_2. Figs. 4.1 and 4.3 reveal that the parameters Δt_1 and Δt_2 have an effect on the F-measure because each curve has a peak, and the values of the parameter Δt_2 that maximize the individual F-measure have a variety. My future work needs to discuss a setting method for the optimum value of the parameters Δt_1 and Δt_2 again, because this paper could not get a clue for the optimization of the parameters Δt_1 and Δt_2 in the proposed method.

In this paper, the value of the parameter Δt_1 when searching for the voice-recognized play-by-play comment that represents the start, and the value of the parameter Δt_1 when searching for the voice-recognized play-by-play comment that represents the end are the same. The value of the parameter Δt_2 when searching for the voice-recognized play-by-play comment that represents the start, and the value of the parameter Δt_2 when searching for the voice-recognized play-by-play comment that represents the end are also the same. To optimize the searching region of the play-by-play comment that represents the start/end, we need to discriminate the parameter for determining the head of the searching region of the play-by-play comment that represents the start from the parameter for determining the head of the searching region of the play-by-play comment which represents the end, i.e., the former Δt_1^{start} and the latter Δt_1^{end} can be set to different values. Moreover, we also need to discriminate the parameter for determining the tail of the searching region of the play-by-play comment that represents the start from the parameter for determining the tail of the searching region of the play-by-play comment that represents the end, i.e., the former Δt_2^{start} and the latter Δt_2^{end} can be set to different values.

Secondly, I discuss the parameters Δt_s and Δt_p, which are used when the system creates the global/local models for baseball games. Fig. 4.5 shows the F-measure for each baseball game (Data 1∼4), and Fig. 4.6 shows the square error of the start/end time depending on Δt_s. Here, the other parameters are set as values maximize the individual F-measures. Fig. 4.7 shows the F- measure for each baseball game (Data 1∼4), and Fig. 4.8 shows the square error of the start/end time depending on Δt_p. Figs. 4.5 and 4.7 reveal that the

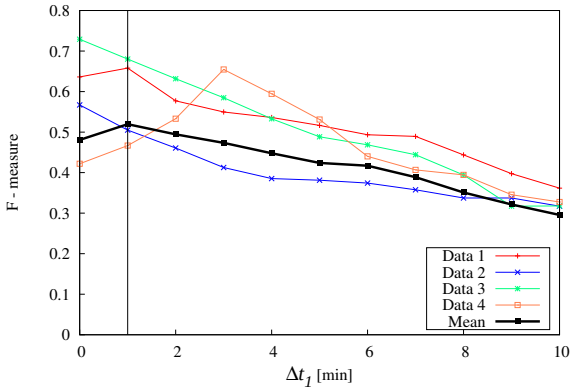

Fig. 4.1 F-measure for each data based on Δt_1 [min].

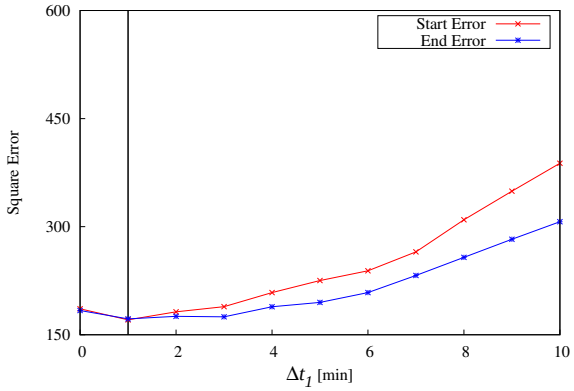

Fig. 4.2 Square Error in the mean based on Δt_1 [min].

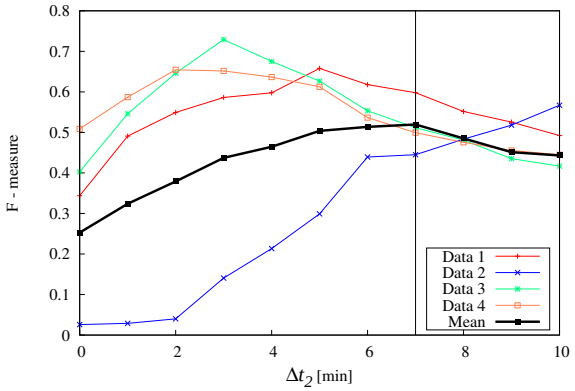

Fig. 4.3 F-measure for each data based on Δt_2 [min].

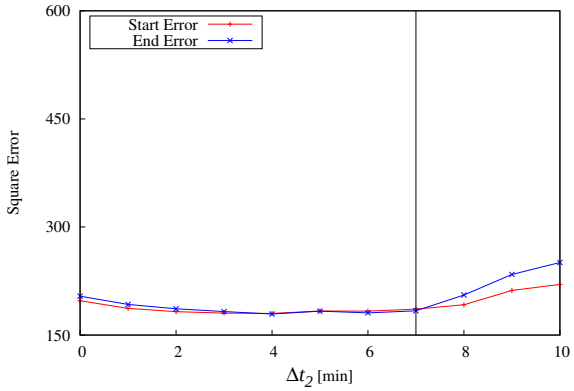

Fig. 4.4 Square Error in the mean based on Δt_2 [min].

parameters Δt_s and Δt_p have a great effect on the F-measure, and that the parameter Δt_p has an especially-great effect, because the curve on Δt_p has a sharper peak than on Δt_s. Figs. 4.6 and 4.8 also reveal that if the system does not set the optimum value to the parameter Δt_p or the parameter Δt_s, the square error becomes considerably large. Furthermore, there is a large gap between the value of the parameter $\Delta t_s/\Delta t_p$, which maximizes the individual F-measure for each baseball game, and the value of the parameter $\Delta t_s/\Delta t_p$, which maximizes the mean F-measure. From the above discussion, to create a refined global/local model to improve the tagging accuracy, my future work needs to conduct a study on a method that allows the values of the parameter Δt_s and Δt_p to vary depending on each occasion of changing sides or changing a pitcher in a baseball game, unlike the proposed method, in which the values of the parameters Δt_s and Δt_p cannot vary during the whole of the baseball game.

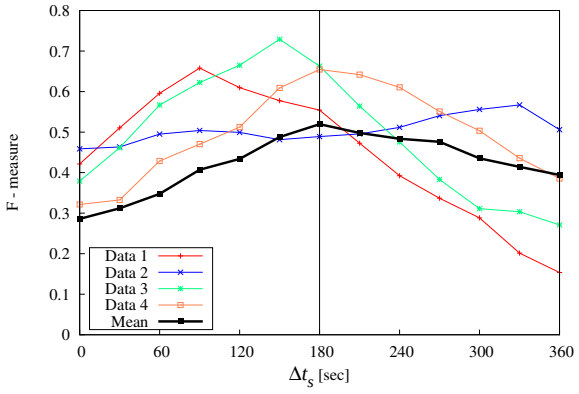

Fig. 4.5 F-measure for each data based on Δt_s [sec].

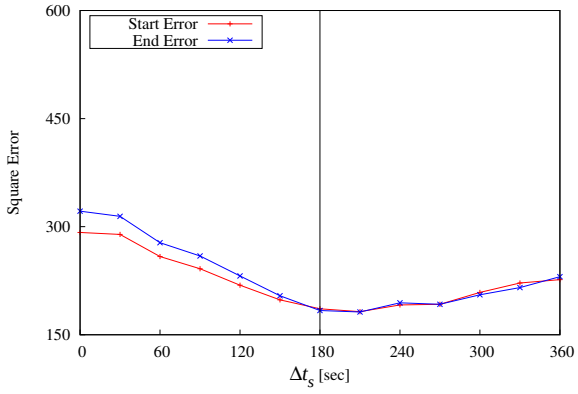

Fig. 4.6 Square Error in the mean based on Δt_s [sec].

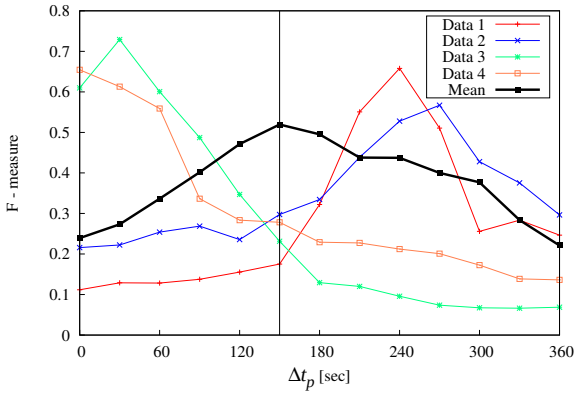

Fig. 4.7 F-measure for each data based on Δt_p [sec].

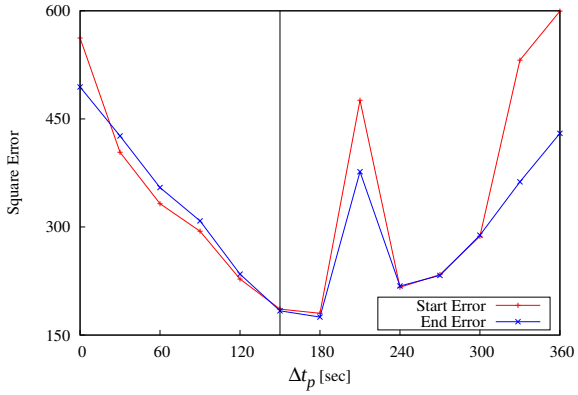

Fig. 4.8 Square Error in the mean based on Δt_p [sec].

Finally, I discuss the parameters w_l and the parameter w_r. Fig. 4.9 shows the F-measure for each baseball game (Data 1∼4), and Fig. 4.10 shows the square error of the start/end time depending on w_l. Here, the other parameters are set as values that maximize the individual F-measures. Fig. 4.11 shows the F-measure for each baseball game (Data 1∼4), and Fig. 4.12 shows the square error of the start/end time depending on w_r. Fig. 4.9 reveals that the parameter w_l has a tendency that the greater the parameter w_l is, the lower the individual F-measures are. Therefore, the system needs to set a rather small value to the parameter w_l. Meanwhile, Fig. 4.11 reveals that the value of the parameter w_r that maximizes the individual F-measure for each baseball game is around 2.4. To optimize the value of the parameter w_r more accurately, my future work plans to perform experiments using many baseball games.

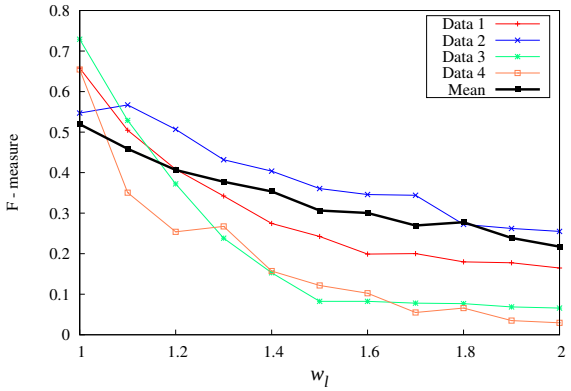

Fig. 4.9 F-measure for each data based on w_l.

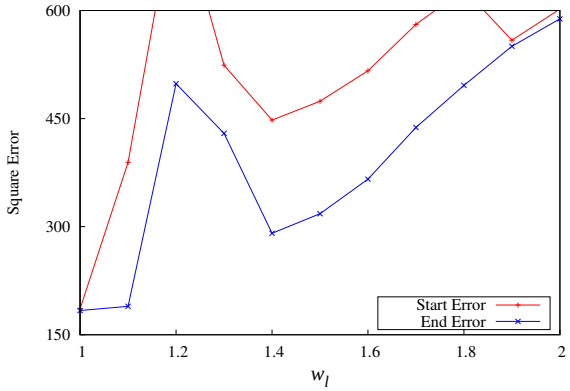

Fig. 4.10 Square Error in the mean based on w_l.

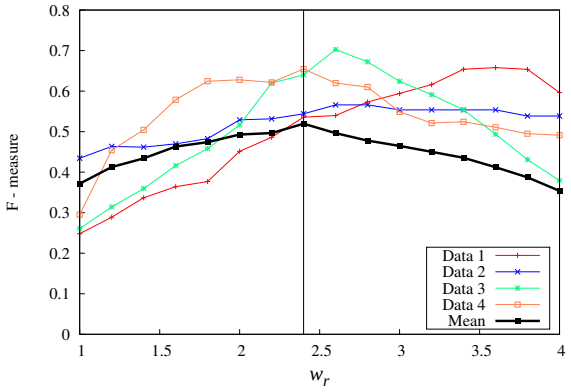

Fig. 4.11 F-measure for each data based on w_r.

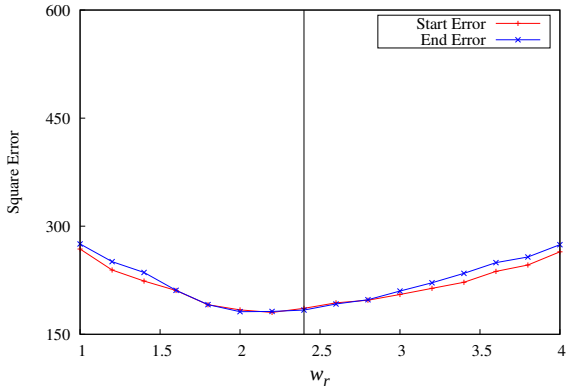

Fig. 4.12 Square Error in the mean based on w_r.

4.2 About Voice Pattern Prioritization

In addition, this paper also finds a problem about the prioritization of the patterns of play-by-play comments that represent the start/end of an event (at-bat scene). This paper has defined the priority of a play-by-play comment that represents the start/end of an event as 0.5 (inferior) or 1.0 (superior). The percentage of the at-bat scenes whose event's start time is appended by using the Superior Play-by-Play Point is 54.2%, while the percentage of at-bat scenes whose event's start time is appended by using the Inferior Play-by-Play Point is 33.7%. And the percentage of at-bat scenes whose event's end time is appended by using the Superior Play-by-Play Point is 13.2%, while the percentage of at-bat scenes whose event's end time is appended by using the Inferior Play-by-Play Point is 11.7%, when the system sets each parameter as the value that maximizes the mean F-measure. The Local Modelling uses the temporal interval T based on the events whose start/end time is appended by using a voice-recognized play-by-play comment which represents the start/end of an event. Therefore, to enable the system to perform the Local

Modelling more exactly and improve the F-measure, my future work plans to revise the patterns of play-by-play comments that represent the start/end of an event (in Table 3.1), and more finely subdivide their priorities, unlike the three-level prioritization (0.0, 0.5, or 1.0) utilized in this paper.

Chapter 5

Conclusion

Firstly, I explain the social contributes of this research. My proposal for Automatic Baseball Video Tagging systems has enabled viewers to select the scenes that they want to watch easily by referring the scene tags (information) that are appended by the system, and to create the personalized sports digest video. Moreover, we can manage a video not per game but per scene when saving and editing the video, and make use of scene retrieval and scene recommendation not only from a sports video but also from multiple sports videos.

Secondly, I explain the technical contributes of this research. To develop an Automatic Baseball Video Tagging system, this paper has proposed a novel Tagging method that utilizes the play-by-play comment patterns for voice recognition that represent the situation of at-bat scenes and take their "Priority" into account. In addition, to search for a voice-recognized play-by-play comment on the start/end of at-bat scenes, this paper has proposed a novel modelling method called as "Local Modelling," as well as Global Modelling used in the basic research. The evaluation experiments have verified the effectiveness of my proposed method, which is equipped with Voice Pattern Prioritization and Recursive Model Localization.

However, this paper gets a clue of the optimization of only the parameter w_l that the system needs to set a rather small value to the parameter w_l. My future work plans to perform experiments by using many baseball games to inspect the optimization for the other parameters in more detail, and aims to improve the tagging accuracy by inventing a novel tagging algorithm that reflexively searches for the voice-recognized play-by-play comment that represents the situation in the near-field region of the local event's estimated start/end time.

Acknowledgements

This paper has used the discussions in the Joint 8th International Conference on Soft Computing and Intelligent Systems and 17th International Symposium on advanced Intelligent Systems (SCIS&ISIS'16). And Assistant Prof. Shun Hattori and the members of Web Intelligence Time-Space (WITS) Laboratory gave me insightful comments and suggestions. I am grateful to everyone who gave us useful advice and comments.

References

[1] C. Xu, J. Wang, K. Wan, Y. Li, and L. Duan, "Live Sports Event Detection based on Broadcast Video and Web-casting Text," Proc. of the 14th ACM Int. Conf. on Multimedia (MM'06), pp. 221-230, October 2006.

[2] N. Babaguchi, Y. Kawai, and T. Kitahashi, "Event Based Indexing of Broadcasted Sports Video by Intermodal Collaboration," IEEE Trans. on Multimedia, Vol.4, Issue 1, pp. 68-75, March 2002.

[3] C. Xu, Y.-F. Zhang, G. Zhu, Y. Rui, H. Lu, and Q. Huang, "Using Webcast Text for Semantic Event Detection in Broadcast Sports Video," IEEE Trans. on Multimedia, Vol.10, Issue 7, pp. 1342-1355, November 2008.

[4] A. Ekin, A. M. Tekalp, and R. Mehrotra, "Automatic Soccer Video Analysis and Summarization," IEEE Trans. on Image Processing, Vol.12, Issue 7, pp. 796-807, July 2003.

[5] D. A. Sadlier and N. E. O'Connor, "Event Detection in Field Sports Video Using Audio-visual Features and a Support Vector Machine," IEEE Trans. on Circuits and Systems for Video Technology, Vol.15, Issue 10, pp. 1225-1233, October 2005.

[6] L.-Y. Duan, M. Xu, and Q. Tian, "A Unified Framework for Semantic Shot Classification in Sports Video," IEEE Trans. on Multimedia, Vol.7, Issue 6, pp. 1066-1083, December 2005.

[7] M. Mukunoki, M. Terao, and K. Ikeda, "Division of Sports Video into Play Units Using Regularity of Cut Composition," IEICE Trans., Vol.J85-D-II, No.6, pp. 1016-1024, June 2002.

[8] M. Kumano, N. Kanzaki, M. Fujimoto, Y. Ariki, K. Tsukada, S. Hamaguhci, and H. Kiyose, "Automatic Extraction of PC Scenes For a Real Time Delivery System of Baseball Highlight Scenes," IEICE SIG-MVE, IEICE Technical Report, Vol.103, No.209, MVE2003-30, pp. 27-34, July 2003.

[9] M. Nakazawa, K. Hoashi, and C. Ono, "Detection and Labeling of Significant Scenes from TV Program based on Twitter Analysis," DEIM Forum 2011, F5-6, February 2011.

[10] A. Ulges, C. Schulze, D. Keysers, and T. M. Breuel, "Content-based Video Tagging for Online Video Portals," Proc. of the 3rd MUSCLE/ImageCLEF Workshop on

Image and Video Retrieval Evaluation, pp. 40-49, October 2007.

[11] S. Siersdorfer, J. S. Pedro, and M. Sanderson, "Automatic Video Tagging Using Content Redundancy," Proc. of the 32nd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 395-402, July 2009.

[12] S. Koelstra, C. M'uhl, and I. Patras, "EEG Analysis for Implicit Tagging of Video Data," Proc. of the 3rd Int. Conf. on Affective Computing and Intelligent Interaction and Workshops (ACII'09), pp. 27-32, September 2009.

[13] J. S. Pedro, S. Siersdorfer, and M. Sanderson, "Content Redundancy in YouTube and Its Application to Video Tagging," ACM Trans. on Information Systems (TOIS), Vol.29, Issue 3, pp. 13:1-31, July 2011.

[14] C.-Y. Chiu, P.-C. Lin, S.-Y. Li, T.-H. Tsai, and Y.-L. Tsai, "Tagging Webcast Text in Baseball Videos by Video Segmentation and Text Alignment," IEEE Trans. on Circuits and Systems for Video Technology, Vol.22, Issue 7, pp. 999-1013, July 2012.

[15] T. Yao, T. Mei, C.-W. Ngo, and S. Li, "Annotation for Free: Video Tagging by Mining User Search Behavior," Proc. of the 21st ACM Int. Conf. on Multimedia (MM'13), pp. 977-986, October 2013.

[16] L. Ballan, M. Bertini, T. Uricchio, and A. Del Bimbo, "Data-driven Approaches for Social Image and Video Tagging," Multimedia Tools and Applications, Vol.74, Issue 4, pp. 1443-1468, February 2015.

[17] M. Larson, M. Soleymani, P. Serdyukov, S. Rudinac, C. Wartena, V. Murdock, G. Friedland, R. Ordelman, and G. J. F. Jones, "Automatic Tagging and Geotagging in Video Collections and Communities," Proc. of the 1st ACM Int. Conf. on Multimedia Retrieval (ICMR'11), No.51, April 2011.

[18] M. Wang, R. Hong, G. Li, Z.-J. Zha, S. Yan, and T.-S. Chua, "Event Driven Web Video Summarization by Tag Localization and Key-Shot Identification," IEEE Trans. on Multimedia, Vol.14, Issue 4, pp. 975-985, August 2012.

[19] R. Ando, K. Shinoda, S. Furui, and T. Mochizuki, "A Robust Scene Recognition System for Baseball Broadcast Using Data-driven Approach," Proc. of the 6th ACM Int. Conf. on Image and Video Retrieval (CIVR'07), pp. 186-193, July 2007.

[20] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei, and H.-J. Zhang, "Correlative Multi-label Video Annotation," Proc. of the 15th ACM Int. Conf. on Multimedia (MM'07), pp. 17-26, September 2007.

[21] M. Wang, X.-S. Hua, R. Hong, J. Tang, G.-J. Qi, and Y. Song, "Unified Video Annotation via Multigraph Learning," IEEE Trans. on Circuits and Systems for Video Technology, Vol.19, Issue 5, pp. 733-746, May 2009.

[22] M. Wang, X.-S. Hua, J. Tang, and R. Hong, "Beyond Distance Measurement: Constructing Neighborhood Similarity for Video Annotation," IEEE Trans. on Multimedia, Vol.11, Issue 3, pp. 465-476, February 2009.

[23] Y.-P. Tan, D. D. Saur, S. R. Kulkami, and P. J. Ramadge, "Rapid Estimation of Camera Motion from Compressed Video with Application to Video Annotation," IEEE Trans. on Circuits and Systems for Video Technology, Vol.10, Issue 1, pp. 133-146, February 2000.

[24] T. Volkmer, J. R. Smith, and A. (Paul) Natsev, "A Web-based System for Collaborative Annotation of Large Image and Video Collections: An Evaluation and User Study," Proc. of the 13th Annual ACM Int. Conf. on Multimedia (MULTIMEDIA'05), pp. 892-901, November 2005.

[25] J. Hagedorn, J. Hailpern, and Karrie G. Karahalios, "VCode and VData: Illustrating A New Framework for Supporting the Video Annotation Workflow," Proc. of the Working Conf. on Advanced Visual Interfaces (AVI'08), pp. 317-321, May 2008.

[26] M. Bertini, A. Del Bimbo, C. Torniai, R. Cucchiara, and C. Grana, "MOM: Multimedia Ontology Manager. A Framework for Automatic Annotation and Semantic Retrieval of Video Sequences," Proc. of the 14th ACM Int. Conf. on Multimedia (MM'06), pp. 787-788, October 2006.

[27] J. Tang, X.-S. Hua, T. Mei, G.-J. Qi, and X. Wu, "Video Annotation based on Temporally Consistent Gaussian Random Field," Electronics Letters, Vol.43, Issue 8, pp. 448-449, April 2007.

[28] J. Yang, R. Yan, and A. G. Hauptmann, "Multiple Instance Learning for Labeling Faces in Broadcasting News Video," Proc. of the 13th Annual ACM Int. Conf. on Multimedia (MULTIMEDIA'05), pp. 31-40, November 2005.

[29] K. Arasawa and S. Hattori, "Automatic Baseball Video Tagging Using Ball-by-Ball Textual Report and Voice Recognition," IEICE SIG-IN, IEICE Technical Report, Vol.115, No.405, IN2015-95, pp. 1-6, January 2016.

[30] K. Arasawa and S. Hattori, "Modeling Refinement for Automatic Baseball Video Tagging," Proc. of the 43rd SICE Symp. on Intelligent Systems (SICE-IS43), March 2016.

[31] K. Arasawa and S. Hattori, "Comparative Experiments on Models for Automatic Baseball Video Tagging," Proc. of the Joint 8th Int. Conf. on Soft Computing and Intelligent Systems and 17th Int. Symp. on Advanced Intelligent Systems (SCIS&ISIS'16), Sa3-3-4, pp. 678-685, August 2016.

[32] K. Arasawa and S. Hattori, "Error-Corrected Update Time of Web Flash Report for Automatic Baseball Video Tagging," IEICE SIG-IN, IEICE Technical Report, Vol.116, No.304, IN2016-65, pp. 31-36, November 2016.

[33] Yahoo! JAPAN Sportsnavi – Ball-by-ball textual report –, `http://baseball.yahoo.co.jp/npb/` (2016).

[34] Advanced Media, Voice Recognition Software AmiVoice SP2, `http://sp.advanced-media.co.jp/`.