Audio Commentary System for Real-Time Racing Game Play

Tatsuya Ishigaki[†] **Goran Topić**[†] **Yumi Hamazono**[†] **Ichiro Kobayashi**[†°]
Yusuke Miyao[†‡] **Hiroya Takamura**[†]

[†]National Institute of Advanced Industrial Science and Technology, Japan,
[°]Ochanomizu University, [‡]The University of Tokyo,
{ishigaki.tatsuya, goran.topic, hamazono-yumi, takamura.hiroya}@aist.go.jp,
koba@is.ocha.ac.jp, yusuke@is.s.u-tokyo.ac.jp

Abstract

We introduce a live audio commentator system designed specifically for a racing game, driven by the high demand in the e-sports field. While a player is playing a racing game, our system tracks real-time user play data including speed and steer rotations, and generates commentary to accompany the live stream. The human evaluation suggested that generated commentary enhances enjoyment and understanding of races compared to streams without commentary. Incorporating additional modules to improve diversity and detect irregular events, such as course-outs and collisions, further increases the preference for output commentaries.

1 Introduction

Live commentary enriches the spectator's experience in sports events and e-sports streams, but it is often unavailable for online videos or recordings of amateur sports due to a lack of skilled commentators. Automatic generation techniques offer a potential solution to this problem.

Previous works focus on generating text-based commentaries using pre-stored tracked data (Puduppully and Lapata, 2021). In contrast, we present a real-time automatic system for generating commentaries, specifically targeting race games inspired by the growing e-sports industry. Live commentary generation typically involves tweet extraction (Kubo et al., 2013), rule-based and keyword extraction from videos (Kim and Choi, 2020), and neural network–based data-to-text approaches (Ishigaki et al., 2021; Taniguchi et al., 2019). Our system combines utterance extraction and neural network–based methods.

Our system works with a physical controller for real-time gameplay in a racing game (Assetto Corsa). During gameplay, the system tracks the user's data, such as speed, steering rotation, and lap progress. Then, the system generates candidates by a neural network–based generator and ranks them in terms of diversity. We also address the problem of limited coverage of rare events in the existing commentary generator. To mitigate these problems, our system detects course-outs and collisions and selects appropriate expressions from a predefined list of utterances. The current generation model is trained on an open Japanese dataset (Ishigaki et al., 2021), although it can be replaced with an English one.[1]

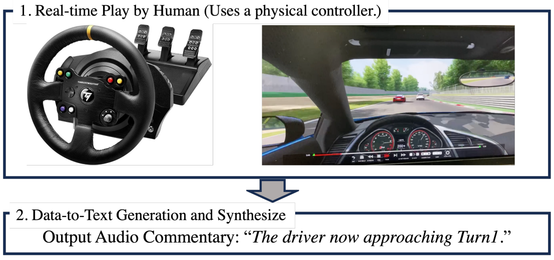
1. Real-time Play by Human (Uses a physical controller.)
2. Data-to-Text Generation and Synthesize
Output Audio Commentary: "*The driver now approaching Turn1.*"

Figure 1: The workflow of our demo. a) user plays a racing game using a physical controller; b) our system generates commentary by analyzing the race situation.

We conducted an evaluation by human judges in terms of enjoyment. The results suggest that: 1) commentary usage enhances user immersion, 2) diversity is important for improving the overall quality of synthesized commentaries, and 3) identifying irregular events further gains the quality.

2 Architecture

Our system consists of five modules in a pipeline. **1: Real-time Data Tracker** Assetto Corsa allows custom functionality to be added to the game via plugins. Thus, we develop a plugin which captures the relevant play data from the game's API. The data is sent to our data processing server. The

[1] An English dataset is also ready. We present our demo in English at the venue.

server samples several features i.e., speed or other metrics, every 0.1 s and keeps the samples in a 10 s moving window. The collected features are sent to the candidate generator as a set of 100-element vectors.

2: Candidate Generator We extend an existing model (Ishigaki et al., 2021), a multi-modal generator for textual and numerical inputs. For textual input, we consider the previous L utterances (we use $L = 5$), while for numerical input, we use vectors produced from the real-time data tracker. Textual input is encoded with BART's encoder (*stockmark/bart-base-japanese-news*), while the original used LSTM. We use an MLP-based encoder for numerical inputs to obtain 728-sized vectors. Another BART encoder processes two types of embeddings: 1) embeddings of the initial token in textual input, and 2) encoded tracked data. BART's decoder generates an utterance to get k candidates by a beam search.

3: Utterance Selection In our preliminary experiments, repetition of the same or similar utterances led to a decline in naturalness. Thus, our system selects the candidate u_{cand} least similar to the previous L utterances u_i. The similarity is calculated as the exponentially weighted sum: $\sum_{i=1}^{L}(1-\alpha)^{L-i}\mathrm{Sim}(u_i, u_{cand})$, where $\alpha = 0.2$. We use BLEU as $\mathrm{Sim}()$.

4: Detection of Irregular Events To generate utterances about irregular events, such as course-outs or collisions, we use an extraction-based approach. The car is assigned a road position value, with 1.0 and -1.0 indicating the center point is at the right and left edge respectively. A course-out is indicated when this value falls beyond a threshold (set here at ± 0.9) Collisions are identified based on the distance to the nearest car, with a distance of less than five meters indicating a collision. When an irregular event is detected, we randomly select an utterance from a predefined list manually created from the training dataset.

5: Text-to-Speech (TTS) The utterance text is sent to VOICEVOX for TTS synthesis[2]. The obtained audio clip is finally played back to the user.

3 Experiments

Training: The candidate generator is trained with 34,897 gold utterance tuples, including previous utterances and tracked numerical data. Validation is performed using 12,295 tuples. The batch size is 5, and AdamW optimizer is used with a learning rate of 10^{-5}. The best model is selected based on cross entropy loss on the validation set.

Evaluation: We assess spectator immersion in synthesized commentary and compare different commentaries. Three models are compared: 1) using the best utterance in beam search, 2) incorporating a diversity module, and 3) combining the diversity and irregular event detection modules. Four human evaluators rank these models in terms of enjoyment as an audience.

4 Results

All the human judges agree that synthesized commentaries enhance immersion. In terms of enjoyment, the commentaries generated by the model with both diversity and irregular detection modules are ten times out of twelve judged better than the model that outputs the best utterance in beam search. This result suggests that these two modules are effective. The model that uses both diversity and irregular event detection modules was eight times out of twelve judged better than the model with only diversity models. Thus, the irregular event detection helps to improve quality.

5 Conclusion

We introduced a commentator for racing games, which generates real-time commentary based on tracked metrics. Future possibilities include utilizing dialogue-styled commentary or enhancing live streams with explanatory graphics. [3]

References

Ishigaki, T., Topic, G., Hamazono, Y., Noji, H., Kobayashi, I., Miyao, Y., and Takamura, H. 2021. Generating racing game commentary from vision, language, and structured data. *INLG*, pp. 103–113.

Kim, B. J. and Choi, Y. 2020. Automatic baseball commentary generation using deep learning. *ACM Sympo. on Applied Computing*, pp. 1056–1065.

Kubo, M., Sasano, R., Takamura, H., and Okumura, M. 2013. Generating live sports updates from twitter by finding good reporters. *WI-IAT*, vol. 1, pp. 527–534.

Puduppully, R. and Lapata, M. 2021. Data-to-text generation with macro planning. *TACL*, 9:510–527.

Taniguchi, Y., Feng, Y., Takamura, H., and Okumura, M. 2019. Generating live soccer-match commentary from play data. *AAAI*, vol. 33, pp. 7096–7103.

[2] https://github.com/VOICEVOX/

[3] This paper is based on results obtained from a project JPNP20006, commissioned by NEDO.