

Automated Commentaries for Simulated Soccer

Evaluation Output

1. Introduction

The fundamental part of this project was to convey information to the audience in a clear, timely and appropriate manner whilst attempting to express a degree of atmosphere between the two.

I carried out research into a number of alternative ways to bridge the gap between the game analysis and the user. I was at the time capable of producing extensive text commentary thus I looked at ways of converting this text into speech by using a text-to-speech tool that could be easily merged with our current code. I had ruled out that I was not to record our own voices to be used for speech therefore I reduced the pressure of time in choosing an existing package.

The Functions

Java defines a number of interfaces for text-to-speech but does not implement any functionality associated to computer generated voice. Fortunately Java works hand in hand with third party companies that give implementations for TTS engines for the Java platform¹. To pick the best one out of the many implementations I defined a number of criteria:

1. We were able to call the voices using pure Java code
2. The lag between the call and speech is kept to a minimum
3. The parameters of the voice can be modified to alternate the excitement level.
4. It is available free of charge
5. It would be relatively straight forward to integrate with the current code
6. A number of voices are available for future development

When checking the number of different systems I quickly came to the conclusion that FreeTTS² was our best option. It was straightforward to setup and widely used in the community so support was at hand. In addition, the integration was seamless.

The availability of examples within the downloaded package enabled us to understand how to call the engine. The challenge from this point was to appropriately implement functionality to change the voice aspects on the fly and to introduce a further voice for statistics.

The default available voice was kevin16, a high quality male voice. To change the parameters of the voice at run time, every sentence to be spoken would be

¹ <http://java.sun.com/products/java-media/speech/forDevelopers/jsapifaq.html>

² <http://freetts.sourceforge.net/docs/index.php>

Automated Commentaries for Simulated Soccer

parsed with an integer defining the level of excitement. The first problem encountered was that meticulous changes to the voice would result in random modifications to the pitch of the voice. For example, attempting to change the tempo of the voice resulted in the pitch increasing or decreasing accordingly. These pitch changes resulted in the voice sounding different therefore was not appropriate for this task. To overcome this problem we limited changes to the voice to very small increments and only modified the words per minute rather than the tempo. This limitation meant that the voice could not demonstrate as much excitement as originally intended but we were still able to provide some enhancements with some background crowd excitation, which seemed to give an effect to the voice. I did come to realise though that the kevin16 voice was generally quite dull therefore the introduction of an alternative voice would be needed to convey the information with more atmosphere.

Evaluation

The introduction of the new voice did not prove as straight forward as I thought. FreeTTS unfortunately had only one available voice and the only Java add on is MBrola³. MBrola provides a number of voices, with support for foreign languages and integrates directly with FreeTTS. I did encounter problems when attempting to integrate the new voices however. MBrola failed to work on Windows platforms and was failing to install on the Unix platforms. The instructions were unclear and support was not available. I spent a few weeks attempting to get the extra voice. After scouring through a number of forums, the new voice, mbrola1 was finally installed ready for use with our commentary. The problem was due to the filename been downloaded as a .exe and the vm argument needing to be set within the project space.

We now had 3 voice, 2 male, 1 female available. The male voice was more realistic and clearer and changes to it were not as obvious. The female voice was very different from the male voice and it seemed appropriate to use that voice for any statistic commenting carried out during the match.

Another challenge that I faced was the possibility of lag. The initial implementation meant that every sentence parsed to the speech engines would eventually be spoken thus a queue eventually started to form resulting in lag of about 3 or 4 seconds. To overcome this problem I implemented a blocking mechanism to the engine meaning that every call to output a sentence would block if the engine is currently running and every sentence attempting to call the engine during the blocking will simply overwrite any older sentence. Once the thread unblocks, the latest added sentence is spoken. The commentaryProducer will itself now make sure that important sentences are parsed at the appropriate time with the use of priotisation.

³ <http://tcts.fpms.ac.be/synthesis/mbrola.html>

Automated Commentaries for Simulated Soccer

Performance was another issue that concerned the system as a whole. It seemed like the speech engine was hogging the CPU for large amounts of time. To overcome this problem I threaded any speech object separately and reduced the time the speech engine was holding the monitor by suspending the thread when there was no current sentence to be spoken by the engine. This was controlled by synchronized calls to wait and notify within the Output class. Every time a new sentence came in, the notify call is made to the speech engine and the engine is reactivated for further utterance.

One final problem I encountered occurred during the initial system testing that I carried out. At a random time during a match in play, the engine would fail throwing an unidentifiable error but seemed to crash after the fourth call to a specific commentary sentence. I never understood why this problem occurred but the removal of that particular sentence from the commentary set fixed the error.

6. Overall Evaluation

To conclude, the speech engine performs well and reduces lag to a minimum meaning that spoken sentences are uttered within an appropriate time frame. All information is conveyed clearly and with some level of atmosphere. All sentences are spoken with no problems and after testing there has not been a failure in the implementation apart from the error mentioned above.

The speech is an area for further work. Because of the time pressures for the release of this project, we were not able to implement a truly realistic text to speech system that could utter with the same technical aspect as a human voice. Text to speech is still an area of worldwide research and implementing a realistic voice commentary system would potentially be a completely separate project. I have identified further areas of development that I feel could be bought to this existing release. These are listed below:

1. Modify the parameters of the voice at speech-time
 - The ability to alternate the speech speed during an utterance. When carrying out research into the way real human commentators speak, we analysed that the voice increases in pitch and tempo. This is not currently possible with FreeTTS or MBrola.
2. The ability to speak words outside the English language
 - Human commentators tend to accentuate certain words or names to create tension and atmosphere, for example saying 'ROONEY!!!' or 'GOOAAAALLLLLLL'. FreeTTS or MBrola did not allow us to do this because it attempted to separate the unrecognisable words into separate smaller words. One area of development would be to add these words within the engine.
3. Use of recorded voices for speech
 - We were told not to use pre-recorded voices for the speech. Many current football games available on the market use real sentences

Automated Commentaries for Simulated Soccer

usually studio recorded using television commentators. This means that the problems above are solved and there are no problems with pronunciation. It also makes the system very flexible for addition of further comments and support for new languages.

4. Use of 2 simultaneous voices

- The ability to have some interactivity between 2 separate voices would result in a more interesting game to view. For example one voice saying...what do you think about this John, and the other voice giving an appropriate answer. This does mean increase complexity in terms of scheduling and accuracy and some further work in the commentary generation algorithms.

Crowd Atmosphere Evaluation

1. Introduction

To enhance the levels of atmosphere in during the matches we decided to introduce some background sound to complement the commentary. Recorded audio was retrieved from a number of stadiums representing different phases of play during a live match such as crowd cheering because of a goal, crowds chanting during play, crowds booing because of an offside and crowds getting excited because of a shot. Also simulated is the referees whistle for kick-offs, out-of-play situations such as offsides, throw inn and goal kicks or resume plays.

The functionality is divided into two classes. One class deals with the scheduling of the audio files and the other deals with the opening and execution of each crowd file including threading and buffering.

The Functions

There is really only one function linked with the crowd atmosphere functionality. It already builds on existing functionality within this development.

1. Overview of Crowd Atmosphere

As discussed above the crowd atmosphere function plays a relevant crowd reaction audio file in time with an event happening on the pitch. I will evaluate the functionality as a whole discussing the initial aims, the actual implementation and finishing with any problems and possible future developments.

2. Evaluation

This functionality was introduced in the latter stages of the project when we came to realise that the speech excitement levels were not sufficient to express energy.

Automated Commentaries for Simulated Soccer

We therefore had no initial aim or designs; it was purely an extension to the current build.

This was implemented in stages, keeping inline with our Extreme Programming approach. The first stage had some simple crowd chants playing in a loop with a separated by a slight pause between each load of the crowd file. This did introduce some of the energy we were looking for but the pause between each file did not make it sound realistic and the chants were too repetitive.

I later developed a way of randomly selecting a chant file from a pool of other files so that there would be some variation in what was heard. The second step was to solve the gaps in between each file. I realised that it was not possible to have a continuous chant, I therefore introduced multi threaded chanting. Because of the way the code was structured it was relatively simple to introduce the threading mechanism. The CrowdAudioPlayer class, which holds the functionality for loading a audio file from a pool of audio file and play them either in a loop or as a one off broadcast. By threading a number of these objects, we are able to overlay a number of different chants. From this point there would be one thread running a background crowd sound, looping continuously for the entirety of the match. These chants would not express any specific behaviour; these can be regarded as an inactive crowd.

We were now in a position to introduce further sounds suited to events happening on the pitch such as goals, offsides, missed shots, referees whistle etc. Each sound is despatched within the CrowdManager object that holds all the references to each file. Each time that event happens, a sound is despatch to be played by the CrowdAudioPlayer. Because each event is despatched threaded then each file just overlaps each other as chants and noises would in the real world.

Each file is despatched when the GameAnalyser identifies a certain event occurs. This is relayed to the CrowdManager which despatches the file.

One final problem that occurred was that the audio file would just cut off rather than fade out as in the real world. I basically ran through all the audio files and introduced in-fading and out-fading into the file to make it sound more natural.

3. Future Development

These are the areas of further development that I have identified:

1. Make this functionality more efficient
 - a. Lengthen crowd audio files to limit loading of a number of smaller files
 - b. Generate a more efficient audio player algorithm
2. Make atmosphere more responsive

Automated Commentaries for Simulated Soccer

- a. Crowd reacts to bad play
- b. Noises for certain events such as posts, shots sounds etc.
- c. Crowd excitement increases as ball approaches goal

4 Overall Evaluation

This development, although not anticipated initially, enabled some excitement relating to the match in play to be expressed. Furthermore, the excitement is synchronized with the events analysed by the GameAnalyser which couples well with the speech to give some accent on the utterance from the commentators.

The build is robust and response is good in line with the information passed from the Game Analyser. Although we faced a number of challenges, we were able to solve these and build on our development by introducing further enhancements such as randomness, blending and multi-threading.