

Evaluation of Analysing functions

1. Overview of Analysing functions

The Analyser classes include Soccerconnect, Game Analyser, Ball Class, Pitch View Class, Logger and others.

2. Evaluation of Analysing functions

The main goal was to gain information from the server to produce commentary which was based on the events occurred and also predicting events that may occur in the future. Additionally it must be quick enough to detect current and future events so they can be processed and be passed to speech and prioritisation classes to be said in time which is relevant.

2.1 Connection:

Soccerconnect was designed to handle the server connection to the server. Our initial attempts were futile as we realised the server output was not in an easy to read format. The format used was in C struts such as shown below:

```
typedef struct {  
    char pmode ;  
    team_t team[2] ;  
    pos_t pos[MAX_PLAYER * 2 + 1] ;  
    short time ;  
} showinfo_t ;
```

The struts with the data contained were in byte format and there was some trial and error involved to get the correct information from them. Furthermore there was very little protocol information contained in the server manual on the monitor information.

Hence the first few iterations were based on a method of the decoding the information byte by byte until we had all the player details and finally the time. This was frustrating as if there had been detailed information on the protocol this step would have been much quicker and we may have identified extra information contained on the server.

Once we had identified the data we had to make sure our system could handle the data in a quick enough time and process it. Initial ideas were based on temporally saving the data in a log file before processing it however this was determined as being too slow. We managed to get the system to react the last received data packet hence following the principles of a real time system.

Additionally we found that the teams connecting to the server did not always have there goalkeepers as player one (this was a known bug in the simulation

Automated Commentaries for Simulated Soccer

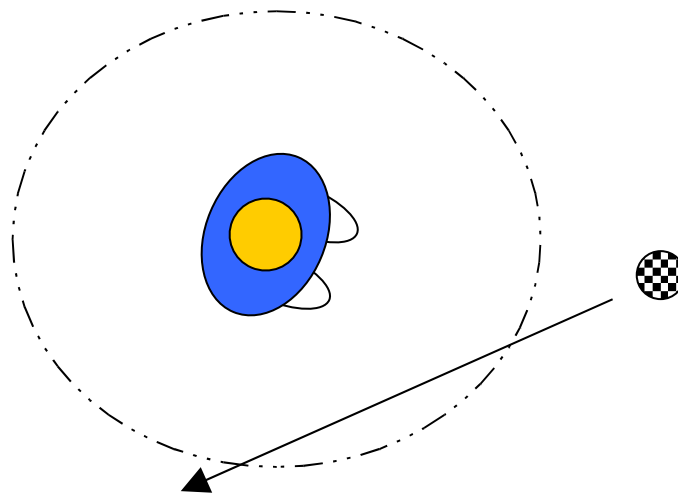
teams and required reconnecting the teams again). This was fundamental to the commentary rules mentioned later on. However in future developments we could identify the individual type of the players (such as defender, keeper etc...) rather than identifying them via player numbers using the server details.

Overall the connection classes were quick enough to handle and decode the data to be passed to the analysing classes.

2.2 Game Analysing:

The Game Analyser class was created to analyse the decoded data from the connection classes.

The main issue occurred in detecting which player had the ball at a current time. The server sends us information when a kick is detected however not all kicks actually touch the ball and move it.



The diagram above shows the player's kickable area in which it will try and kick the ball. The ball passes just through that area and the simulation tries to kick the ball once. The player however does not make any contact with the ball and it continues on its path (not all kicks in the simulation manage to kick the ball).

In our system this became a large problem as the ball may pass a few of players like this and our system would register it as a pass or shot from the player it passed. Workarounds included detecting two kicks from a player, as usually there would be one to control the ball and get into position and finally one to pass or shoot. Again this was not accurate either as the simulation did produce goals and passes which were classified as one touch football (where players pass or shoot with one touch and do not control the ball first)

Automated Commentaries for Simulated Soccer

However we decided to use the initial method of detecting each kick and run our code and detect how many of these anomalies occurred and whether they could be cut down. We realised most of these anomalies occurred when two players went for the same ball in a tackle situation or when goal kicks occurred when the goal keeper missed a save. We realised we could handle the tackle situation as we expected a few comments to be made quickly by our system and the prioritisation classes would remove most and hence get rid of the anomaly. In the other situations we could determine if it was goal kick or a goal that a player of the defending team could not have touched the ball if it's a goal kick or goal (except in own goal cases which are extremely rare in the simulation)

For future developments we could have created our own detection of a kick or touch within our code rather than relying totally on the server. This would have meant using the initial "kick" protocol from the server but then also checking the velocity of the ball to whether it had changed from its previous velocity after passing the player.

Some of the rules in the Game Analyser as mentioned earlier in Design to determine commentary also became a difficult task due to the "kick" protocol. The raw data taken from the connect and player's classes would produce the following into an array list:

```
[Player 7, Player 7, Player 7, Player 8, Player 12]
```

This would need to be broken down and understood by the Game Analyser that player 7 had three touched and before passing to player 8 who shot instantly and it was saved by team 2s goal keeper (Player 12 is keeper for team 2 and Player 22 is the last player on team 2).

The overall implementation of the Game Analyser was done well and it was also very quick. It managed to identify events in real time and process the incoming data from the server in good enough time to be handled.

However the largest problem we faced was that our first few iterations did not use any threads. Once further classes were added including the speech the system needed to wait until the speech was processed before detecting the next event. This caused a huge problem as speech processing took more than 100ms at times and what occurred was that data packets sent by the server in that time were ignored and only the current packet after the processing was used to detect events. Therefore this led to the commentary missing out certain events or detecting different players to what was being shown currently on the screen. Thankfully we managed to identify the issue quickly and we realised we needed to have separate threads in order to receive process the packet information from the server.

Automated Commentaries for Simulated Soccer

Further enhancements to the Game Analyser would be mainly based round the commentary rules. After developing the rules we realised we have around 15 or so variables e.g.

Time, Player on the ball, Play Mode and Score etc...

We could have broken the rules down into an xml file which is loaded when the system start. This would allow maintainability of the rules which do change depending on versions of the server.

Example of XML structure of two rules:

XML Rule for goal:

```
< rule type="goal">
<play_mode>Goal*</play_mode>
<call_method>goal</call_method>
</ rule>
```

XML Rule for pass for Team 1:

```
< rule type="pass1">
<play_mode>Play On</play_mode>
<first touch>1-11*</first touch>
<last touch> 1 -11* </last touch>
<call_method>pass</call_method>
</ rule>
```

Lastly using the Game Analyser we used the stats data to work out intelligent comments to be said at "dead" periods of the game. This worked extremely well and gave an intelligent feel to the match. This could be further added to by expanding these intelligent comments to half time and end of match analysis.

2.3 Neural Networks:

To further enhance the analysing and commentating we can use neural networks as our AI commentator to pick up the events and have them spoken. For instance neural networks show how systems can learn tasks and used to recognise patterns.

"A neural network, also known as a parallel distributed processing network, is a computing solution that is loosely modelled after cortical structures of the brain. It consists of interconnected processing elements called nodes or neurons that work together to produce an output function. The output of a neural network relies on the cooperation of the individual neurons within the network to operate. Processing of information by neural networks is characteristically done in parallel rather than in series (or sequentially) as in earlier binary computers or Von Neumann machines. Since it relies on its member neurons collectively to perform its function, a unique property of a neural network is that it can still perform its overall function even if some of the neurons are not functioning. In other words it is robust to tolerate error

Automated Commentaries for Simulated Soccer

or failure. (see fault tolerant) Additionally, neural networks are more readily adaptable to fuzzy logic computing tasks than are Von Neumann machines.”

http://en.wikipedia.org/wiki/Neural_network

The patterns of certain passes and events can be tracked and identified using this technique. Further certain events can be given more weighting and hence they will be given a larger priority to be said and analysed.

This enhancement will require a lot of work however we believe it's feasible that you can use neural networks to create an AI commentator which can learn from matches by picking up which events are classed as important and therefore a more realistic commentary overall.

2.4 Ball Analysing:

The ball analysing class was a totally new idea for the project which had not been attempted before in regards to other similar commentary systems. For that reason it contained a lot of initial trial and error to whether the idea was possible. Altogether it was a success and produced accurate distances of passes and goals. However there was one issue which caused problems:

Coordinates:

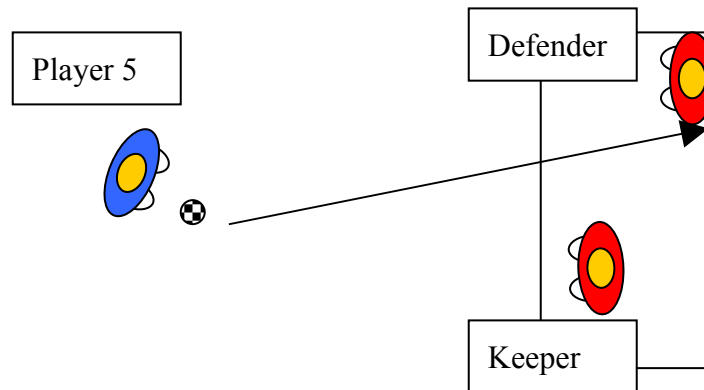
The coordinates in the simulator followed the kick off spot to be the origin. This meant depending where you were on the field there would be negatives components for the distance the ball travelled. This was a very small issue and we had to determine the pythag for the distance by taking the absolute value of the components. However the issue arose in determining the goal distance.

In the case of a goal, you would need the player's position when he last touched the ball and also the position of the centre of the goal. The centre of the goal was not always the same as it depended on which team scored the goal and also whether it was the first or second half the goal was scored in. This wasn't too hard to determine however during the second half we seemed to get very large distances of 80 yards + for goal distances. After spending a while looking through documentation of the server we realised that the teams do not switch sides in the next half. The monitor (2D visualisation of the game) switches the sides in its representation which has confused us. Removing the code we had and giving each a set goal the issue was removed.

Another issue which we faced was that the original data did not give the same coordinates as the monitor and hence it was difficult to follow. After some trial and error and searching through a few old pieces of documentation on monitors where we were able to find that the original data needs to be scaled to give the correct aspect.

Automated Commentaries for Simulated Soccer

For further enhancements we could determine crosses by taking the y value of each pass and determining whether it's considerably larger than the x value. There was also a bug which could have been fixed. This was caused by the kick problem which was mentioned earlier.



As shown above player 5 scores a goal and the ball passes by the keeper and just by the defender who is on the line. The ball passes through the defenders kicking distance and the defender tries to kick it but he misses.

The server recognises the kick and our system would say that the defender scored the goal and the distance would be over 80 yards. This is caused as it determines the defender scored and hence the goal is 80 yards away from where he kicked the ball. This bug can be fixed however due to time it was left and also we realised after fifty matches we have only seen one own goal.

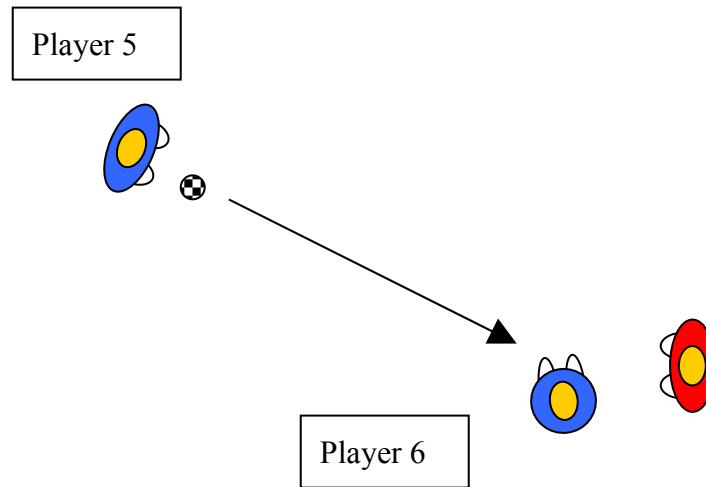
2.5 Pitch View:

The analysing using Voronoi worked very well when linked up to the rest of the analysing classes. It was able to produce time critical analysis in time. While it was based on references from other papers it did not strictly follow there functions. This was due to the time we had for the project so a limited amount of Voronoi events were added.

For future developments more Voronoi could be added so you could ascertain which areas the teams needed to do better in and produce more intelligent commentary. At present the Voronoi was based on attack comments however you could easily ascertain defence comments and have comments using the stats details to identify key areas of the pitch where each team was losing or winning the ball.

Lastly we could have linked up the Voronoi techniques and Game Analyser to determine passing. For instance if player 5 gets the ball below:

Automated Commentaries for Simulated Soccer



Once player 5 kicks the ball you can use the server to work out the balls vector. Hence using the Voronoi diagram you can ascertain which player the ball has been aimed at other whether it's a shot on goal and hence produce predictive commentary. This will however use a lot of maths and just like in the above picture if needs to be accurate otherwise it could be determined that he is giving the ball away.

2.6 Logger:

The logger class was introduced due to the problems in testing that we faced. We realised early on in the project that the pace of the simulation was fast and it was not always easy to match up the analysis to what was occurring. We developed a new testing method which would log the events we were testing with the time and also record the match and hence we could go back and watch it slow motion and determine whether our code was picking up the correct events.

However the code could not be debugged in the replay mode which causes issues in identifying problem code. Secondly a certain event such as a goal from a corner might not occur in a few matches and hence you would need to run until it actually occurs. In future developments we could have created players ourselves and got them to act out certain events so we could then debug our code and not have to wait for certain events. This would save large amount of time into introducing new commentary.

2.7 Past Matches:

The past matches class kept a log of the last games played that the system was commentating on. This gave a realistic feel to the commentary and introduced

Automated Commentaries for Simulated Soccer

some pre game commentary. However the data stored was limited to the score line and in further enhancements we could include details of the individual goal scorers and the player of the match.

Additionally we could have a central store to which could save the results of the games rather than the games run on that computer. The store could be a SQL database on the internet and hence any matches run using our system would be stored. This would be useful feature for Robocup tournaments.

3. Conclusion:

Overall it manages to process information from the server to produce commentary which is based on the events which have occurred and also predicts events that may occur in the future. Additionally it is quick enough to detect current and future events so they could be said in a timely manor for anyone listening.

The functions are able to provide a realistic and intelligent analysis on the game which matches what we wanted to achieve. It matched and went beyond in of some the analysing in similar projects before, this to us was the main indicator of it being a success. We have though identified a large scope to add further analysis which would improve the overall commentary. However this is not due to the fact that we were not able enough to add each of these functions however more towards the benefits to some of the functions mentioned above would be small considering the amount of time spent on the actual function.