#undefine "solsJS"

이제 실제 축구경기 중계방송에서도 선수들의 위치를 알 수 있다!

#undefine 의 soloJS는 그림1에서와 같이 축구게임에서 많이 활용되는 레이더를 세계최초로 실제축구경기에서 활용이 가능하도록 만든 작품이다.

누구나 한번쯤은 생각해봤을 것이다. 실제 축구 중계화면에서도 게임과 모든 선수들의 위치가 표현된다면 얼마나 좋을까. 그러한 의문점에서 출발한 본 작품은 실제 구현단계에 까지 이르게 되었다.

Figure 1. 레이더(화면 아래의 노란 테두리)가 있는 축구 게임 장면

학생신분에서 만들었다는 한계점 때문에 실제 축구경기를 대상으로 테스트를 해보지 못한점이 가장 아쉬우나, 우리가 만든 미니어쳐에서는 훌륭하게 작동을 하였으므로 실제 축구경기에서도 별 무리없이 적용 가능할 것으로 보인다.

축구라는 경기는 모두가 알다시피 세계적은 스포츠 종목이다. 이러한 스포츠 종목을 중계하는데 있어서 본 작품과 같은, 많은 사람들이 한번쯤 생각을 해봤음직하지만 아직 실제로 구현이 된적이 없는, 그러한 아이디어가 지금까지도 적용이 되지 않았다는 것에 약간의 의구심이 드는것도 사실이다. 그러나 그만큼 본 작품이 시장에 나왔을때 기대되는 시장가치는 상상을 초월하는 것이라 생각한다.

정보통신부장관상

SoloJS

1. 작품명 : SoloJS

(실제 축구 경기에서의 레이더 구축 및 방송에의 활용)

2. 제작팀 : #undefine

팀장 : 윤종수
개발자 : 윤종수, 이호진
주소 : 경기도 수원시 팔달구 우만동 499-3번지 301호
전화 : 016-558-2846
전자우편 : rookie@doit.ajou.ac.kr

3. 프로그램 요약설명

3.1 개발 배경 및 목표

누구나 한번쯤 이런 생각을 해 봤을 것 이다. 축구 게임에는 레이더(축구 선수들의 위치가 나타 나있는 조그만 창)가 뜨는데 축구 중계에는 왜 레이더가 안뜨지?

밑의 사진에서도 볼 수 있듯이 축구 중계시 한 부분을 클로즈업 하면 다른 부분을 볼 수 없다. 다른 부분에 우리선수가 있는지, 어느 쪽으로 패스를 해야 하는지, 각각의 팀들이 어떤 포메이션으로 어떤 작전을 쓰는지, 전체 화면으로만 볼 수 있었던 많은 정보를 레이더를 통해서 시청자에게 전할 수가 있다.

Figure 1. 레이더(화면 아래의 노란 테두리)가 있는 축구 게임 장면

Figure 2. 레이더가 없는 실제 축구 경기 화면

3.2 전체 시스템 구성

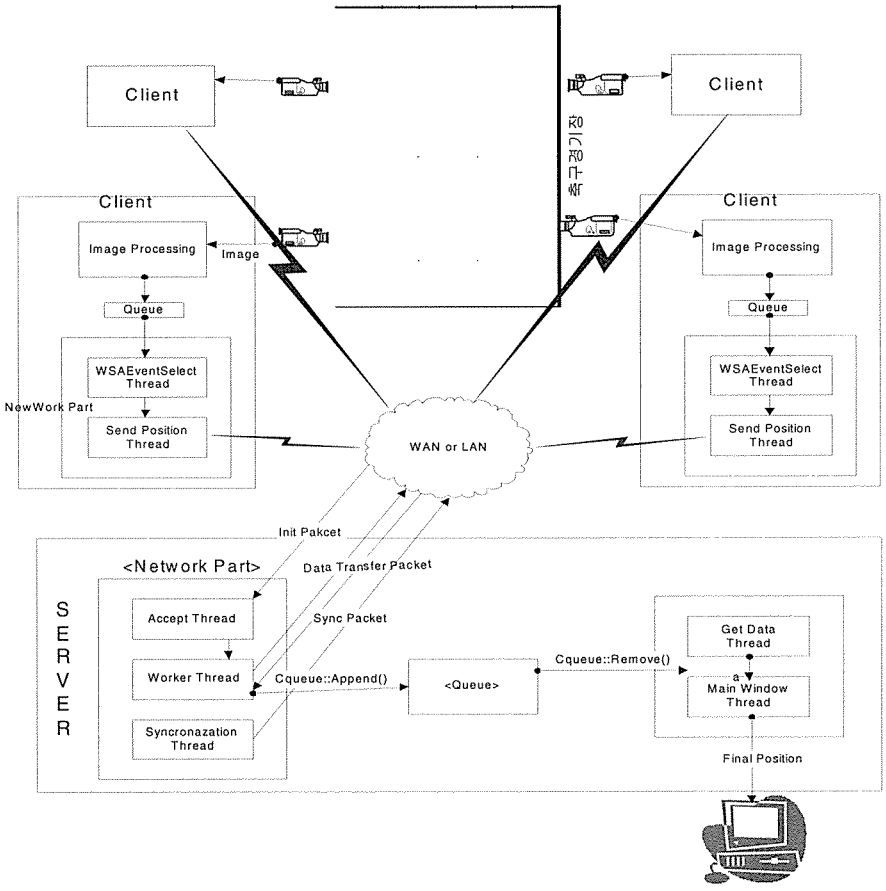

Figure 3. 전체 시스템 구성도

시스템은 그림3에서 보듯이 4대의 비디오 카메라가 다린 Client Part와, Client Part에서부터 넘겨 받은 선수들의 Position Infomation을 받아서 최종 위치 결정을 내린다음 화면(본 프로젝트에서는 모니터화면으로 설정하였으나, 실제 방송에 적용할 경우, 축구경기 화면이 나오는 텔레비전 화면으로의 대체가 가능)에 뿌리는 Server Part로 구성된다. Client가 4대인 이유는 카메라 영상과 실제 축구경기장과의 왜곡현상을 처리하는 과정에서의 오차를 최대한으로 줄이기 위함이다.

3.2.1 Client Computer

- Setting Part
  1. 먼저 경기장을 setting 해야 한다. 각각의 Client Computer가 보는 시점에서 생각하면 경기장은 직사각형이 아니고 마름모다. 마름모의 꼭지점들을 이용하여 가상의 직사각형을 만든다.
  2. 양 팀의 유니폼 색을 설정해준다.
  3. 선수들을 각각 클릭하여 object를 생성한다. 화면 전체를 찾는 것은 processing 하는데 시간이 너무 많이 걸렸다. 클릭한 지점의 근처를 찾아서 object를 찾는 것이 시간이 훨씬 단축 되었다.

- Image Processing Part
  1. 카메라에서 영상을 받아오면 콜백함수(FrameCallbackProc())가 호출이 된다. 이 함수에서 각각의 object들의 이동 여부를 판단하여 위치를 이동한다.

- Network Part
  1. WSAEventSelect Thread
     - 목적 : 일반적인 WSAEventSelect I/O 모델을 사용하는 네트워크 본체
  2. Send Data Thread
     - 목적 : Image Processing Part에서 처리되어진 Position Infomation 들을 Server Computer에게 전달해줌
     - 동작방식 : Multi-Thread 이므로, 언제 어느때에 보낼 Position 정보의 가공이 끝났는지 알 수가 없음. 그래서 생각해낸 것이 Event이고, Image Processing Part 에서 한 프레임당 선수들의 위치에 대한 좌표값 계산이 끝나면 Event를 Signaled 상태로 바꾸어 줌으로써, Send Data Thread에서 이 Event를 기다리고 있다가 데이터를 서버에 보내주게 된다.

3.2.2 Server Computer

- Image Processing Part

: Network Part에서 처리되고, Queue에 넣어준 Position 정보들을 화면에 실제로 뿌려주는 역할을 담당함.

1. Main Thread
   - 목적 : 일반적인 WinMain()함수. 여러가지 윈도우 이벤트를 처리함.
2. Get Data Thread
   - 목적 : Queue에서 데이터를 뽑아와서, 화면에 직접 뿌려줌.
   - 동작방식 : Multi-Thread 환경이므로, Queue에서 데이터를 뽑아오기 위하여 동기화 객체(Event)가 필요하게 됨. 카메라 4대의 Position이 다 들어오면 Worker Thread에서 Event Object를 Signaled 상태로 바꾸어주고, Get Data Thread에서는 이를 기다리고 있다가, Event가 Signaled 상태가 되면 비로소 화면에 데이터를 뿌려줌.

- Network Part

: Client 4대가 보내오는 Position Packet들을 받아서 적절히 가공을 한뒤, Image Processing Part와 함께 공유하는 Queue에 넣어준다.

Socket IO Model은 IOCP(I/O Completion Port)를 사용하였으며, Queue의 디자인의 경우, 쓰레드 안전성을 보장하기 위하여 Mutex등을 사용하였다.

1. Accept Thread
   - 목적 : 클라이언트로부터의 접속을 받는 역할을 하는 쓰레드.
   - 동작 방식 : 클라이언트가 접속을 하면, 올바른 클라이언트인지 검사를 하고, 올바른 클라이언트라면 Worker Thread(작업용 IOCP)와 연결을 한다.
2. Worker Thread
   - 목적 : Sync Packet을 보내거나, Client로부터 받은 Position Packet을 적절히 처리해줌.
   - 동작 방식 : 연결된 소켓들의 여러가지 작업들(send, receive, close)와 같은 것들의 결과를 가지고 여러가지 부수적인 작업들을 해줌.
   - 부수적인 작업들 : Position Packet이 들어오면, 각각의 적절한 Quueu에 넣어줌. Sycronazation Thread에서 보내주는 Sync Packet들의 완료를 처리함.
3. Syncronazation Thread
   - 목적 : 일정한 시간간격으로 SyncPacket을 보내줌으로써 Server와 Client간의 Sync를 맞추어줌.

4. Queue Desjgn Issue

- 목적 : Thread가 여러 개이므로 Thread간의 통신을 위하여 Queue의 사용이 불가피함. 또한 Client에서 Position 정보를 보내오는 것과, Server가 그것을 화면에 나타내는 데에는 어느정도의 Delay가 불가피 함으로 Queue를 사용해야만 함.
- 동작 방식 : Multi-Thread 환경이므로, 필연적으로 동기화 객체의 사용이 필요함. 이를 위하여, Mutex를 사용하였으며, Queue는 Append/Remove 작업전에 Mutex의 소유권을 기다린 후 각각의 작업을 해주어야 함.

3.3 메뉴 구성도

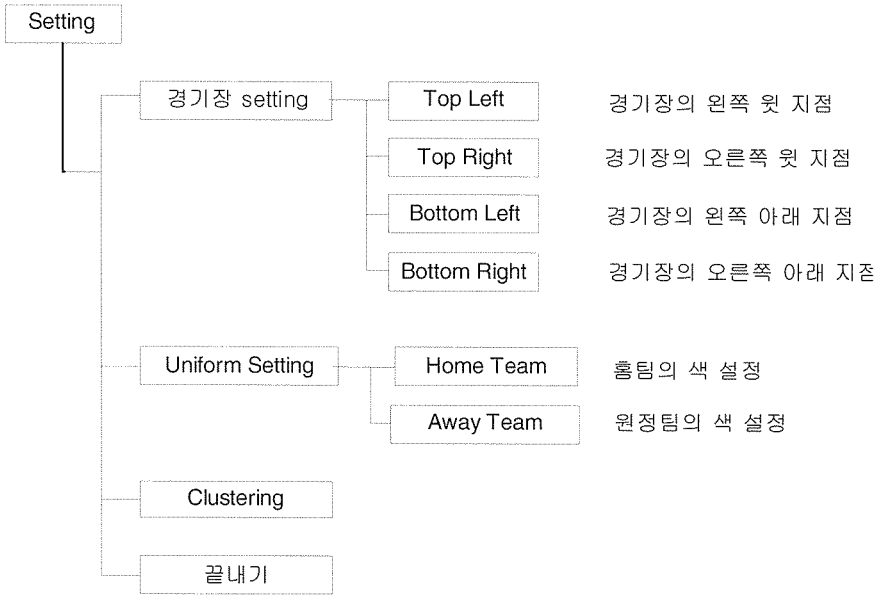

Figure 4. Client Computer 메뉴 구성도

4. 개발단계별 기간 및 투입인원수

개발단계	개발기간	비고
시스템 설계	03. 5. 10 ~ 03. 6. 23	작품 구상 및 자료수집
프로그래밍	03. 6. 23 ~ 03. 9. 3	실제 작품 구현
통합 테스트	03. 9. 3. ~ 03. 10. 6	종합 테스트
매뉴얼제작	03. 10. 6 ~ 03. 10. 9	
계	5개월	

5. 사용 또는 개발언어, TOOL, Library

구분	사용언어 및 Tool	비고
[Server] Main Module	C++, Visual C++ .NET	Windows API
[Server] Network Module	C++, Visual C++ .NET	Winsock2, Multithread
[Client] Main Module	C++, Visual C++ .NET	Windows API, VFW, DirectShow
[Client] Main Module	C++, Visual C++ .NET	Winsock2

6. 사용시스템

구분	이름	비고
CPU	AMD Athlon XP 1800	
RAM	256 RAM	samsung
OS	Windows XP Professional	Microsoft
Tool	Visual C++ .NET	Microsoft
Network Card		
Frame Grabber	Sigma TV 2	교체가능
Camcoder	LG	교체가능