|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 구분 | 제출  자료명 | 제출  여부 | 제출형태 | 비고 |
| 1 | 최종보고서  첨부 자료  1) 프로젝트 제안서  2) 요구사항분석서  3) 설계서  4) 주간보고서  5) 창의설계경진대회 발표 자료 | **O** | 동영상 및  PPT, 팀 화일 | 팀장 명의로 제출 |
| 2 | SW 프로그램 등록증 | **O** | 팀 pdf 화일 | 프로그램등록절차와 등록증 샘플 첨부 참조(수수료 영수증 및 통장 사본) |
| 3 | Git hub 등록실적 | **O** | 팀 화일 | 프로그램등록절차 첨부 |
| 4 | Git hub 등록 source code | **O** | 팀 화일 | source code를 하나의 압축 파일로 생성 |
| 5 | 논문실적 | **X** | 팀, 혹은 개별 | 최종 보고서에 첨부 |
| 6 | 특허실적 | **X** | 팀, 혹은 개별 | 최종 보고서에 첨부 |
| 7 | 외부경진대회 | **X** | 팀, 혹은 개별 | 최종 보고서에 첨부 |
| 8 | 기타 결과물 | **X** | 팀, 혹은 개별 | 최종 보고서에 첨부 |
| 9 |  |  |  |  |
|  |  |  |  |  |
| 1 | 동료평가 | **O** | 개별 화일 | 블랙보드에 개별 제출 |
| 2 | TOPCIT 응시 | **O** | 개별 화일 | 재출치 않아도 됨 |

캡스톤 디자인 최종결과물 제출 리스트

2020. 06. 25.

캡스톤 2반 2조 팀장: 조현기(서명)

팀원: 구본학 (서명), 조남규 (서명), 이세영 (서명)

**2020 학년도 1학기**

**캡스톤 디자인 결과 보고서**

|  |  |  |
| --- | --- | --- |
| **소 속 (전공)** | 컴퓨터공학과 | |
| **팀명(주제명)** | 2조 시한폭탄 | |
| **지 도 교 수** | 문현준교수 (인)  권기학교수 (인) | |
| **팀 장** | 학번: 14011074 | 이름: 조현기 |
| **팀 원** | 학번: 15010980 | 이름: 구본학 |
| 학번: 15010993 | 이름: 조남규 |
| 학번: 15011059 | 이름: 이세영 |
| 학번: | 이름: |
| 학번: | 이름: |
| **제 출 일 자** | 2020. 06. 25 | |

**세종대학교 컴퓨터공학과**

**최종 보고서**

1. **개발 목표**

2013~2018년 동안의 도로 유형별 사고 통계 (자료 출처: taas 교통사고분석시스템)

교차로 사고란, 교차로 내 또는 교차로 측 단 후방 30m 이내의 부분에서 발생한 사고를 말한다. 교차로는 여러 방향에서 진행하는 차들로 인하여 상충의 위험성이 높아 주의 운전이 힐요 하다. 하지만 교차로에서의 사고 현황은 줄어들지 않고 있다. 따라서, 본 시스템은 교차로에 설치되는 각 임베디드 기기를 통하여 교차로상에 위치한 횡단보도 보행자 및 도로의 차량을 검출하여 교차로에서의 보행자 안전을 보장하며 차량의 검지, 특히 좌회전 차선의 차량을 검출해 좌회전 대기중인 차량이 있을 경우에만 좌회전 신호를 주는 방식 등의 방법으로 교차로의 신호체계의 효율성을 높일 것이다.

1. **설계 사양서**
   1. **제안 프로젝트 계획서**

* **프로젝트 방향**

본 시스템은 임베디드 기기 내부의 영상처리 및 인공지능, 서버, 웹으로 이루어진다. 임베디드 기기는 내부의 카메라를 이용하여 영상을 촬영하며 그 영상을 활용해 교차로의 보행자와 차량을 감지한다. 인공지능은 보행자의수 뿐만 아니라 좌회전 차량의 대기 정보와 직진차량의 대기 수를 서버로 전달한다. 서버는 이러한 정보를 받아 기기별로 관리하며 이 정보를 웹에게 제공한다. 웹에서는 각 임베디드 기기 들에서 들어온 정보를 이용해 신호들 제어한다.

* 1. **개발 환경**

해당 프로젝트의 개발환경은 다음과 같다.

|  |  |  |
| --- | --- | --- |
| **Language** | **Library** | **Tool** |
| Python  C# | OpenCV | Unity  VIM |

* 1. **요구사항 정의서**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **사용자 요구사항 정의서** | | | | | | | | | | | | |
| 보행자 및 차량 검지시스템 | | | 서브시스템명 | | 교차로시스템 | | | | | | | |
| 분석 | | | 작성일자 | | 2020. 04. | | | | 버전 | 1.0 | | |
| 요구사항 명 | 구분 | 요구사항 설명 | | 요구사항 출처 | | 제약  사항 | 중요도 | 해결방안 | | | 검수기준 | 비고 |
| 영상처리 | 기능 | -카메라가 검지하고 있는 영역의 보행자를 감지하여 보행자의 위치와 수를 알 수 있어야 한다.  -카메라가 검지하고 있는 교차로 영역의 차량을 감지해야 한다 | | 시한폭탄\_요구사항명세서docx.3p | | 파이카메라의 영역 범위 확인 필요 | **상** | 교차로의 전 범위가 감지되도록 카메라 범위 설정 Test 실시 | | | -보행자의 위치, 수와 차량의 인식을 90% 이상 정확도를 목표로 한다. |  |
| 성능 | -보행자와 차량의 정보에 대해 실시간으로 입력 받아 처리해야 한다.  -교차로 시스템 구현에 있어 필수 데이터이므로 정확한 인식을 해야 한다.. | | 효율적인 인식 알고리즘 설계 필요 | 오픈소스 및 패키지 활용하여, 기능 구현 | | |  |
| 인공지능 | 기능 | -보행자의 위치로 보행자의 이동방향을 구별해야 한다.  -좌회전 대기차량, 대기차량의 수 등의 정보를 추출해야 한다. | | 시한폭탄\_요구사항명세서docx.3p | | 영상처리 거친 데이터 필요 | **상** | 인식 처리가 되지 않은 데이터에 대하여 return | | | -보행자의 이동방향의 정확도를 95% 이상을 목표로 한다.  -좌회전 대기차량의 구분 정확도를 99%이상을 목표로 한다 |  |
| 성능 | -영상처리를 거친 데이터에 대해 실시간으로 입력 받아 처리해야 한다.  -교차로 신호 제어에 요구되는 데이터이므로 이동방향 및 대기상태를 정확히 파악해야 한다. | | 정확한AI알고리즘 설계 필요 | AI알고리즘과 데이터셋 학습을 통한 기능 구현 | | |  |
| 서버 | 기능 | -Raspberry Pi-Server와 Server-Client 간의 연결 및 파일 송신/수신 | | 시한폭탄\_요구사항 명세서docx.4p | | 연구실에서만 서버구축 가능 (공간적 제약) | **상** | 여분의 파이 서버를 활용하여 지속적인 Test 시도 및 오류 검출 | | | -신호체계를 위한 클라이언트와의 응답시간을 50ms 미만을 목표로 한다. |  |
| 성능 | -각 임베디드 기기와 서버 간의 와이파이를 이용한 무선통신이 가능해야 한다.  -각 임베디드 기기에서 들어오는 정보를 실시간으로 업데이트 해야 한다. | | 데이터 송수신 제한이 없을 정도의 컴퓨터 성능 필요 | 연구실 810호 서버 사용 | | |  |
| 웹 | 기능 | -서버에서 교차로의 보행자, 차량의 정보를 가져와야 한다.  -서버에서 얻은 정보로 신호를 적절히 조작해야 한다. | | 시한폭탄\_요구사항명세서 docx.4p | | 구체적인 시뮬레이션 구현 | **상** | 유니티 프로그램을 활용하여 구현 | | | -50ms 안에 서버에서 얻은 정보 표현  -신호제어는 99%의 정확도를 목표로 한다. |  |
| 성능 | -서버로부터 실시간으로 데이터를 받아야 한다.  -가공된 정보를 활용하여 즉각적이고 정확한 신호를 제어해야 한다. | | 웹 상에서 유니티 Player 구동시 Delay 발생 | 유니티 제공 소스 활용 및 지속적인 서버, 네트워크 연결 확인 | | |  |

* 1. **유스케이스 다이어그램**

|  |  |  |  |
| --- | --- | --- | --- |
| UCD ID | SS\_UCD\_001 | UCD 명 | 교차로시스템 |
| 관련 서브시스템 ID | SS\_IP\_001 | 관련 서브시스템명 | 영상처리 |
| SS\_AP\_002 | 인공지능 |
| SS\_MS\_003 | 메인 서버 |
| SS\_WH\_004 | 웹 |
|  | | | |

* + 1. **유스케이스 목록**

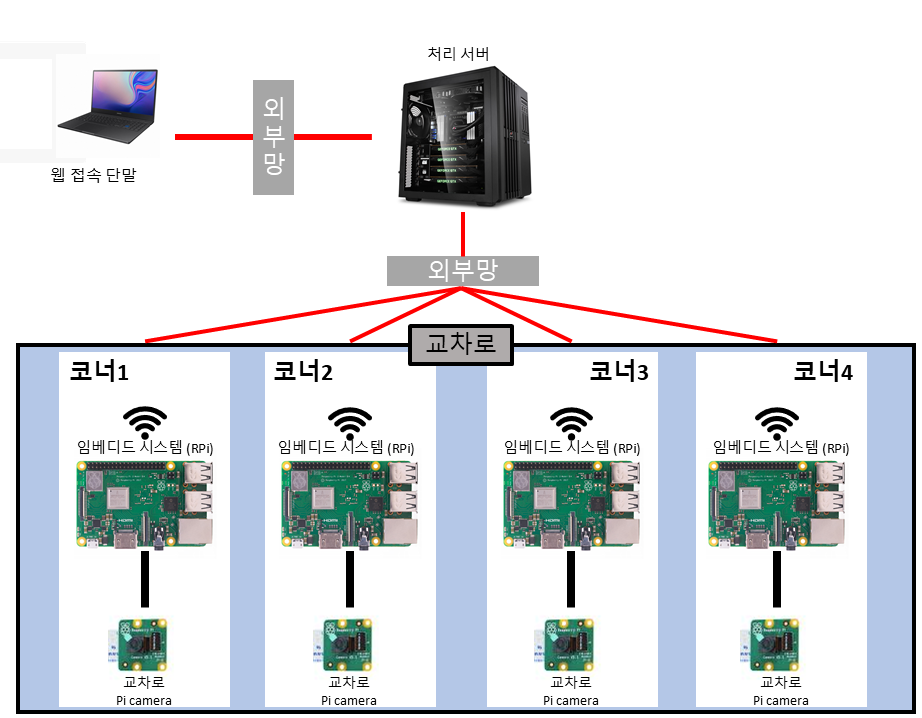
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 유스케이스  ID | 유스케이스명 | 유스케이스 설명 | 관련액터 ID | 관련 UCD ID | 관련 요구사항 ID |
| SS\_UCD\_001\_01 | 횡단보도 대기, 건너기 | 보행자가 횡단보도에서 대기하고 건너는 모습이 카메라에 촬영되어 임베디드 기기에 저장된다. | SS\_AC\_001  SS\_AC\_003 | SS\_UCD\_001 | SS\_ IP 001 |
| SS\_UCD\_002\_01 | 교차로 정지, 이동 | 차량이 교차로에서 정지하고 이동하는 모습이 카메라에 촬영되어 임베디드 기기에 저장된다. | SS\_AC\_002 SS\_AC\_003 | SS\_ IP 001 |
| SS\_UCD\_003\_001 | 영상처리 | 입력으로 들어오는 영상에서 보행자 및 차량을 인식한다. | SS\_AC\_001  SS\_AC\_002  SS\_AC\_003 | SS\_ IP 001 |
| SS\_UCD\_004\_001 | 인공지능 | 영상처리에서 들어오는 보행자 및 차량의 정보를 통해 더욱 자세한 정보를 얻는다. | SS\_AC\_003 | SS\_AP 002 |
| SS\_UCD\_005\_001 | 서버 | 영상처리와 인공지능을 걸친 데이터를 교통신호제어를 하기 위한 웹으로 전송한다. | SS\_AC\_003  SS\_AC\_004 | SS\_MS 003 |
| SS\_UCD\_006\_001 | 교통신호 제어 (웹) | 서버에서 전송된 정보로 교통신호를 제어한다.. | SS\_AC\_003  SS\_AC\_004 | SS \_WH004 |

* 1. **시퀀스도 및 클래스 다이어그램**
     1. **시퀀스도**

|  |  |  |  |
| --- | --- | --- | --- |
| 시퀀스도 ID | SS\_SD\_001-01 | 시퀀스도명 | 교차로시스템 |
| 관련 유스케이스 ID | SS\_UC\_001-01 | | |
| 주요 액터 | 보행자, 운전자(차량) | 주요 객체 (또는 클래스) | 영상처리, 인공지능, 메인 서버, 웹 |
|  | | | |

* + 1. **설계 클래스도 및 클래스 정의**
       1. **전체 클래스도**

|  |  |  |  |
| --- | --- | --- | --- |
| 설계 클래스도 ID | SS\_DCD\_001\_01 | 설계 클래스도명 | 전체 클래스도 |
| 관련 유스케이스 ID | SS\_UC\_001\_01 | | |
| 텍스트, 지도이(가) 표시된 사진  자동 생성된 설명 | | | |

* 1. **시스템 아키텍쳐**

웹(시뮬레이션)

* + 1. 아키텍처 요구사항 및 구현방안

|  |  |  |
| --- | --- | --- |
| 요구사항 ID | | SS\_IP\_REQ\_010 |
| 요구사항 내용 | | |
| 카메라가 검지하고 있는 영역의 보행자 및 차량을 감지하여 차량 및 보행자의 수와 개별 물체의 위치를 식별할 수 있어야 한다. | | |
| 구현방안 | | |
| 영상처리 알고리즘을 이용하여 차량 및 보행자의 수와 위치를 90%이상의 정확도로 구현하는 것을 목표로 한다. | | |
| 요구사항 ID | | SS\_AI\_REQ\_020 |
| 요구사항 내용 | | |
| 보행자의 위치와 이동방향을 식별해야 하며, 교차로상에 위치한 차량 중 좌회전 대기차량, 대기 차량의 수와 같은 정보를 얻어야 한다. | | |
| 구현방안 | | |
| 인공지능을 이용하여 보행자 정보의 식별은 95%이상의 정확도로, 좌회전 대기차량의 식별은 신호체계 연동에 있어 중요한 요소이므로 99% 이상의 정확도로 구현하는 것을 목표로 한다. | | |
| 요구사항 ID | SS\_MS\_REQ\_030 | |
| 요구사항 내용 | | |
| 서버는 교차로상에 설치된 임베디드 기기에서 처리된 후 보내진 데이터를 실시간으로 수신할 수 있어야 한다. | | |
| 구현방안 | | |
| -신호체계를 위한 클라이언트와의 응답시간을 50ms 미만을 목표로 한다. | | |

* 1. **화면 상세설계**

|  |  |  |  |
| --- | --- | --- | --- |
| 화면 ID | VS\_MV | 화면 명 | 메인View |
| 관련 유스케이스 | SS\_UCD\_006\_001(웹) | | |
| 관련 시퀀스도 ID | SS\_SD\_001-01 (9번) | | |
| 화면 유형 | 서버에서 전송된 정보 표시 | | |
| 화면 개요 | 메인 화면을 보여준다.  검지중인 구역의 횡단보도를 보행하고 있는 보행자가 있을 때, 경고를 보여준다.  현재 검지중인 한 도로의 차량의 수를 보여준다.  떠, 도로의 좌회전 대기차량을 보여준다. | | |
| 비행기, 표지판이(가) 표시된 사진  자동 생성된 설명 | | | |
| 기술적 고려사항 | | | |
| 마우스를 이용한 시점 변경, 실시간으로 업데이트되는 정보 | | | |

1. **테스트 명세서**
   1. **시험 수행 절차**

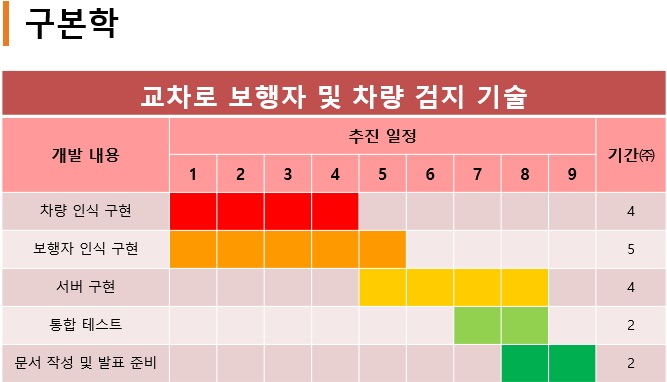
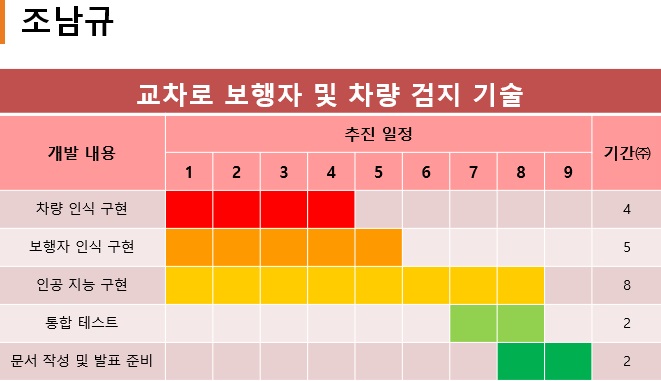
|  |  |
| --- | --- |
| 구분 | 시한폭탄 |
| 총괄 시험 계획 | * 시험 전략 및 수행절차 수립 * 시험 전체 일정 수립 * 총괄시험 계획서 작성 |
| 단위 시험 계획 | * 부분 시스템별 단위시험 케이스 도출 * 수행 * 오류 검토 및 수정 |
| 통합시험 계획 | * 통합시험 시나리오 작성 * 수행 * 오류 검토 및 수정 |

* 1. **시험 수행 일정**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 순번 | 주요활동 | 시험일자 | 완료일자 | 비고 |
| 1 | 총괄시험 계획 | 2020.04.24 | 2020.04.30 | 계획 수립 |
| 2 | 단위시험 수행 | 2020.05.25 | 2020.05.29 | - 서버 클라이언트 동작확인  - 차량 인식 확인  - 보행자 인식 확인  - 동영상 속도저하 확인  🡪 해상도 조정  - 교차로 영상 데이터 확보 시급 |
| 3 | 단위시험 결함수정 및 결과 확인 | 2020.05.29 | 2020.06.04 | - 영역 지정  -> 동영상 속도개선  - 교차로 영상 데이터 1차 확보  -> 테스트 결과 부적합 / 재촬영 필요  - 차량 인식 정확도 개선 필요 |
| 4 | 통합시험 수행 | 2020.06.04 | 2020.06.11 | - 교차로 영상 데이터 2차 확보  -> 테스트 결과 적합 / 이상 없음  - haar training을 통한 차량 인식 정확도 개선 완료 |
| 5 | 통합시험 결함수정 및 결과 확인 | 2020.06.11 | 2020.06.18 | - 실제 교차로 영상 및 모형판 테스트 결과 양호  - 서버 클라이언트 데이터 전송 양호  - Unity 테스트 양호 |

1. **요구사항 대비 시스템 구현 내용 + 정도**

|  |  |  |  |
| --- | --- | --- | --- |
| **요구사항** | **품질요소** | **완성도** | **설명** |
| 영상처리  보행자 및 차량의 인식률이 90% 이상이어야 한다.  보행자 이동방향 구별, 좌회전 대기차량, 대기차량의 수 정보 추출 | 성능 | 90% | 좌회전 차량의 인식률 96%  보행자 및 직진차량 인식률 95.7%  보행자 이동방향에서  횡단보도 보행 유무로 변경  인식률 95.7% |
| 서버  Raspberri Pi와 server – client 간의 송수신  임베디드 기기와 무선통신이 가능  임베디드 기기에서 들어오는 정보 실시간 업데이트 | 기능 | 100% | Raspberri Pi, server 그리고 유니티와 송수신 가능  서버에서 각 임베디드 기기에서 들어오는 정보 실시간 업데이트 |
| 웹  서버로부터 실시간으로 데이터 업데이트  가공된 정보를 활용 즉각적으로 신호 제어 | 기능 | 100% | 신호제어 이상 없음 |

1. **개발 추진 내역**
   1. **개인별 목표 개별일정**
   2. **캐비닛이(가) 표시된 사진

      자동 생성된 설명전체 개발 추진 내역**
   3. **업무 분담 표**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **이름** | **1** | **2** | **3** | **4** | **5** | **6** |
| 조현기 | 시뮬레이션 유니티 구현 | 유니티 신호체계 구현  (Traffic Info 클래스 구현) | 클라이언트 개발, 신호제어 스크립트와 연결 | 설계서 기반 서버와 클라이언트 개발 | 전체 테스트 및  프로젝트 통합  창의설계 경진대회 준비  PPT제작  모형제작 | 창의설계 경진대회 준비  PPT제작  시연준비  및  시연영상 촬영 |
| 구본학 | 차량 및 보행자 알고리즘 조사, 자료수집 | 라즈베리파이 부품 수령,  차량 및 보행자 인식 스크립트 개발 및 테스트 | 라즈베리파이 서버 초기설정 및 환경 구축  차량 및 보행자 인식 스크립트 개발 | 실제 교차로 영상 수집  Haar training을 통한 알고리즘 개선 |
| 조남규 | 라즈베리파이  초기개발환경  세팅 | CCTV영상을 이용한 차량 및 보행자 인식 스크립트 개발 | 영상 인코딩 개선 | 차량 및 보행자 인식 스크립트 개발 |
| 이세영 | 라즈베리파이 개발환경 구축 | 보행자 및 차량 인식 스크립트 구현 | 보행자 및 차량 인식 스크립트 정확도 개선 | 영상 인식 결과를 프로그램 변수로 출력  실제 교차로 영상 수집 |

1. **구현된 개발 프로그램**

**▶ 개발 프로그램 개요**

본 교차로 보행자 및 차량 검지 기술 과제는 관심 영역 지정을 통해 직진 차량 영역, 좌회전 차량 영역, 보행자 영역 3가지로 영역이 구분되어 있으며, 각각의 영역에 적합한 알고리즘을 적용한다. 직진 차량 영역과 보행자 영역은 Image Contour를 이용하여 직진 차량 영역의 차량 수와 보행자 영역의 횡단보도 위 보행자 유무를 파악한다. 좌회전 차량 영역은 Haar training을 통해 만들어진 분류기를 이용하여 Haar cascade 알고리즘으로 좌회전 차량 영역 내 차량 유무를 파악한다. 얻어진 정보는 서버로 전송되고, 이를 유니티 클라이언트가 실시간으로 각 영역에서 제공되는 정보들을 수신하여 해당 정보를 화면에 표시한다.

**▶ 주요 기능**

본 프로그램은 2가지 주요 기능을 가진다. 첫째, 보행자 및 차량 인식 기능이다. 각 구역에 하나씩의 임베디드 기기로, 한 방향의 도로를 맡아 인식하며, 도로의 차량과 보행자를 검지한다. 둘째, 와이파이를 이용한 무선통신으로 기기에서 인식한 정보들을 전송하며, 총 4개의 검지 기기가 각각 맡은 범위를 인식하여 보행자와 차량의 정보를 얻는다.

**▶ 구현 성능**

본 프로그램의 인식과 정확도의 성능 부분이다. 교차로에 위치한 보행자 및 차량의 인식에서 보행자가 어느 쪽의 횡단보도에서 횡단하는 지와 직진차량 영역의 차량 수, 좌회전 차량과 보행자 영역에서의 유무 정보를 얻을 수 있다. 보행자 및 차량 인식의 경우 90% 이상의 인식률을 목표로 하였으며, 좌회전 차량 Haar의 정확도는 96%, 컨투어의 정확도는 95.7%로 구현하였다.

**▶ 보행자 및 차량 인식 작동 원리**

**1. 관심 영역 (Region Of Interest, ROI) 지정**

**- 차량 영역 -**

****

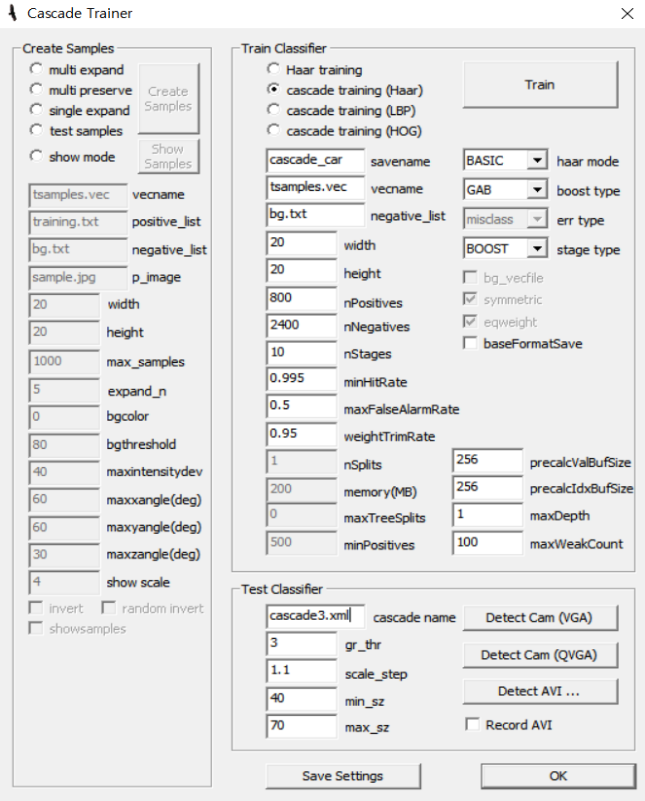
* **보행자 영역 -**

****

- 입력된 영상에서 차량과 보행자의 인식 과정을 진행할 관심 영역을 설정하고, 직진 차량과 보행자 영역은 Image Contour, 좌회전 차량 영역은 Haar-cascade 알고리즘을 적용하여 해당 영역의 정보들을 검출한다.

- 프레임 전체의 모든 차량과 보행자를 인식하지 않고, 지정된 관심 영역 내에서의 차량과 보행자를 인식함으로써 불필요한 리소스 낭비를 줄이고 영상의 처리 성능을 개선한다.

**2. Haar-Cascade Detection**

Haar-Cascade 알고리즘은 매 프레임을 복잡한 처리과정을 거치지 않고 분류기만 거쳐도 차량의 식별이 가능하기 때문에 저 성능의 라즈베리파이에 적합하다. 본 프로그램에서는 이를 더 높은 정확도로 구현하고자 다음과 같이 Haar Training 과정을 거친 후에 해당 학습 과정에서 나온 분류기를 사용하여 구현하였다.

**- Cascade Trainer - -haarcascade\_car.xml -**

분류기를 만들기 위한 Haar training 과정은 이하와 같이 진행되었다.

**1) Sample 생성**

- 약 900장의 Positive sample과 약 2500장의 Negative sample을 사용하여 분류하였다.

- 일반적으로 Positive sample과 Negative sample이 1 : 3 의 비율을 유지한다면 좋은 성능을 기대할 수 있다.

- Positive sample이란 추출하고 싶은 정보가 담긴 데이터를 말하며, Negative sample은 추출하고 싶은 정보와는 관련이 하나도 없는 데이터를 말한다.

**2) Vec 파일 생성**

**-** Createsamples를 이용하여 vec 파일을 생성한다.

- vec 파일은 Positive sample 영역들을 그대로 크기만 넓이와 높이로 바꿔서 저장한 파일이다.

**3) Classifier 생성**

**-** Haar training을 이용하여 10 stage의 classifier를 생성하였다.

- Cascade 검출기는 일련의 기본 검출기들로 구성되는데, 검출 방식은 먼저 1단계 분류기에서 false들을 걸러내고, 나머지에 대해서는 2단계 분류기로 false들을 걸러내는 방식으로 최종 단계까지 도달하였을 때 남아있는 물체를 검출 성공으로 간주한다.

**4) 최종적인 xml 생성**

- Haarconv를 이용하여 haarcascade\_car.xml 생성

- Haar training을 거치게 되면 각 단계별로 classifier가 생성되는데, 이를 통합하여 최종적인 xml을 생성한다.

**5) 테스트**

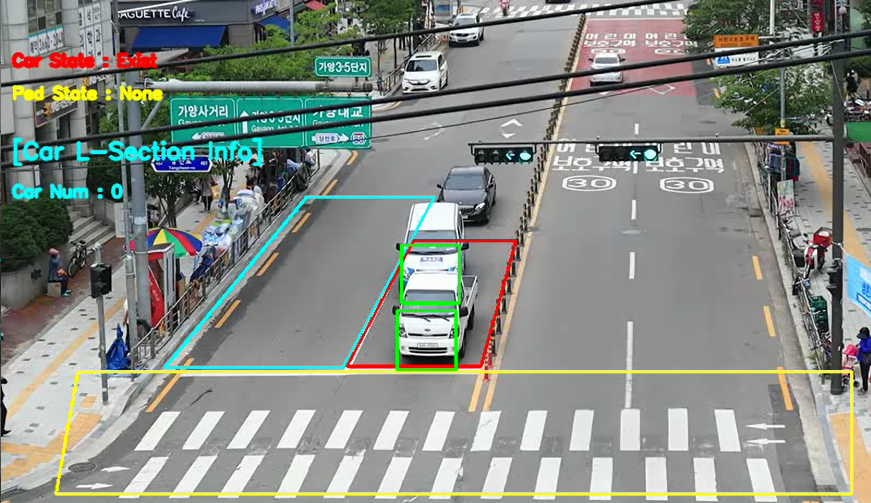
**-** 테스트하고자 하는 avi 동영상 파일을 이용하여 완성된 Haar 분류기를 확인할 수 있다.

**6) 좌회전 차량 인식 과정에 적용**

**- 기존 OpenCV Haar 분류기 사용 결과**

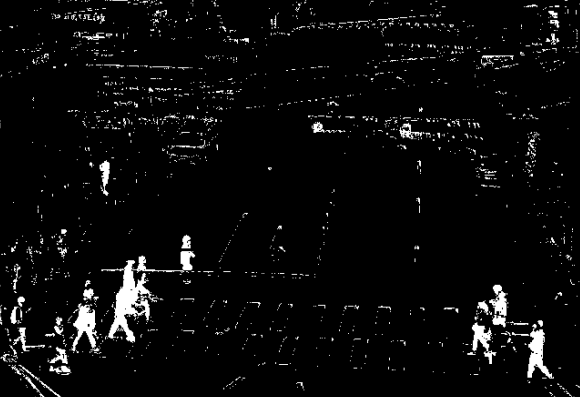


**- 별도의 Haar training을 통해 생성된 분류기 사용 결과**



- 완성된 분류기를 사용하여 좌회전 차량 영역의 차량 유무를 검출한다.

- 기존 OpenCV에서 지원하는 Haar 분류기보다 차량 검출에 있어 좋은 성능을 얻었다.

**3. Image Contour**

**1) 원리**

**-** Background Subtractor (배경 제거) 알고리즘을 이용하여 영상의 배경을 검은색으로 만든다. 이후 이진화 처리를 통해 검출하려는 물체를 하얀색의 성질을 띄도록 변환하여 윤곽선을 추출한다.

**-** 이후 인식된 물체의 무게중심 좌표를 구하고, 실시간으로 움직이는 물체의 좌표를 최신화 및 저장하여 (Tracking), 물체의 위치를 파악할 수 있다.

**2) 개선 과정**

**-** 1차적으로 보행자가 화면에서 잡히는 영역의 크기를 계산 후 파라미터를 조정하여 차량의   
오인식 빈도를 줄였다.

- 보행자 영역에서 좌 / 우 진입 방향 부근과 직진 차량 영역에 상 / 하 진입 방향 부근에 Line을 추가하여, 횡단보도 위의 보행자의 유무와 차량의 수를 파악한다.

**3) 보행자 인식 과정에 적용**

**- 기존 OpenCV에서 제공하는 Haar 분류기 사용 시 -**

**- Image Contour 사용 시 -**



**-** 기존 OpenCV에서 지원하는 Haar-cascade 알고리즘보다 높은 검출율을 보여준다.

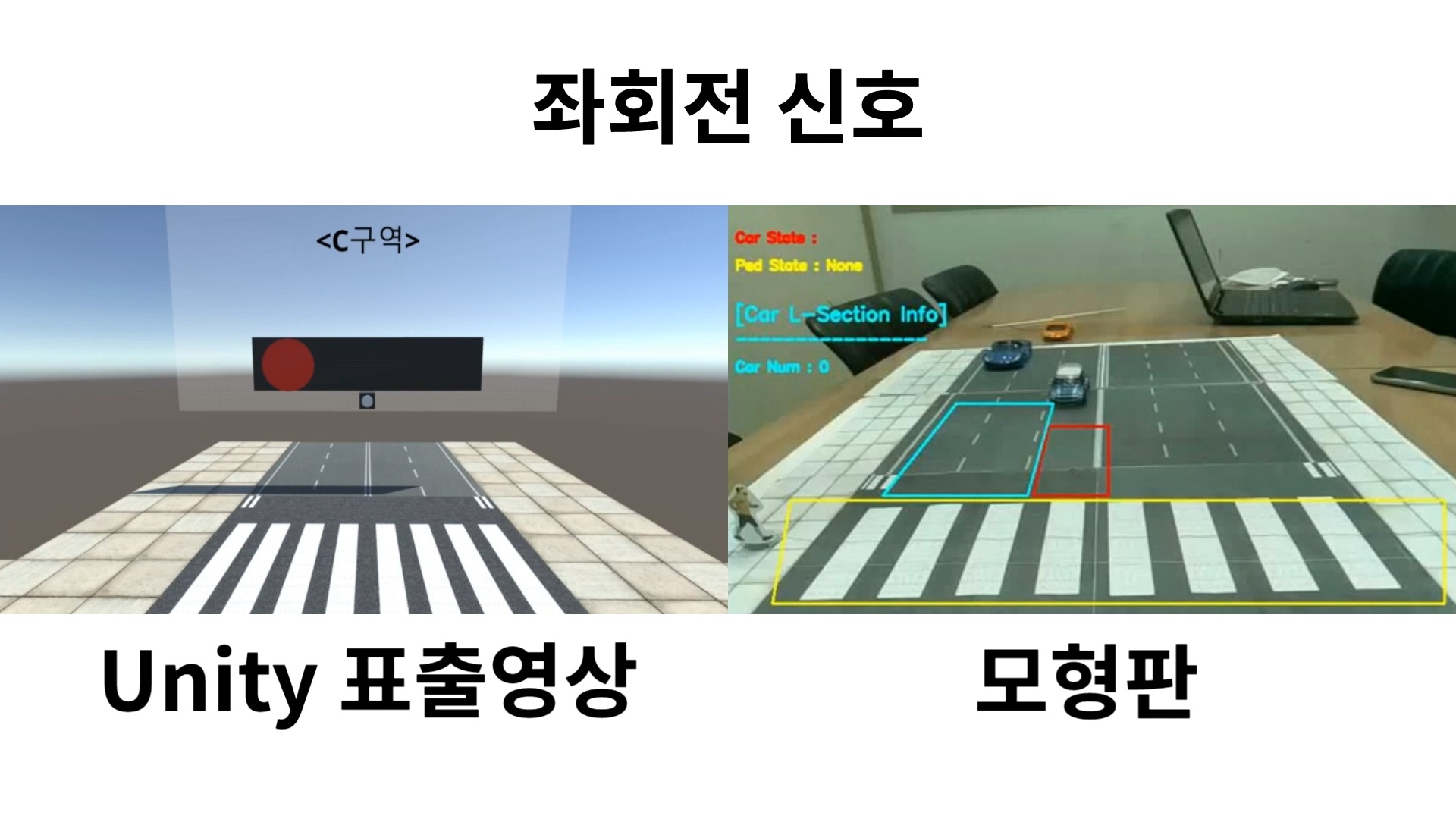
- HOG 알고리즘을 이용하여 보행자 검출을 할 경우 저성능의 라즈베리파이 환경에서 CPU 점유율을 100%를 초과하며 기기가 꺼지는 현상이 나타났다. 하지만 Image Contour를 이용하면 저성능의 라즈베리파이 환경에서도 원활하게 보행자가 검출되는 것을 확인할 수 있었다.

**4. Unity 프로그램**

본 과제에서는 라즈베리파이 기기를 통해 얻어진 정보를 Unity 프로그램에서 수집하여 이를 적절하게 표출해주게 된다. 실제 교차로에서는 테스트할 수 없는 제약사항이 있어 실제 교차로를 모사한 모형판을 제작하여 실시간으로 구동해보았고, 아래는 그 결과이다. 프로그램의 기능은  
다음과 같다.

**▶ 주요 기능**

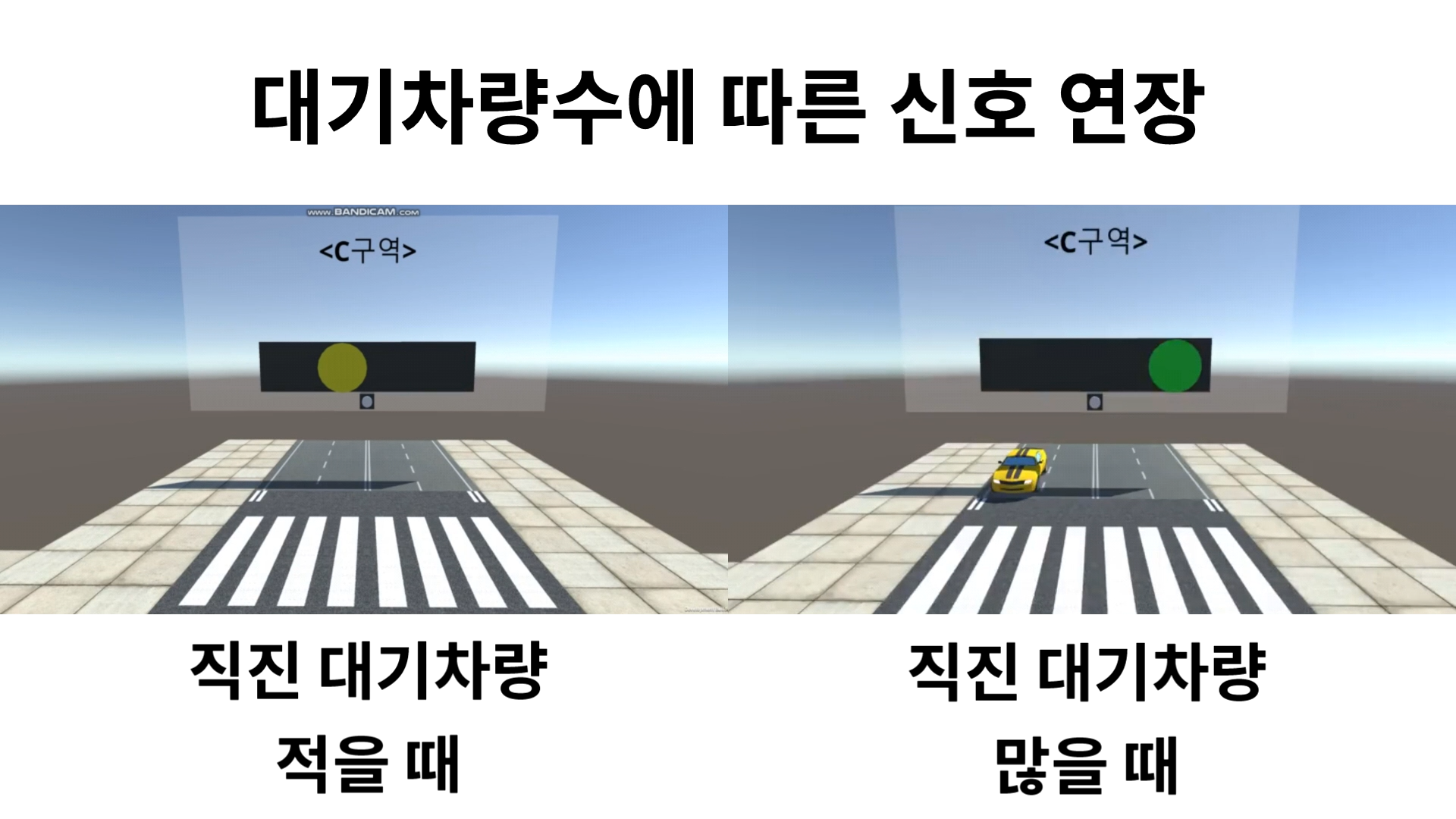
1. **좌회전 신호 연장**





좌회전 차량이 기기에서 검지 될 경우 이를 신호에 반영하여 좌회전 신호를 주게 되고, 좌회전 차량이 없을 경우 좌회전 신호는 생략하여 차량의 대기시간을 감소시킨다.

1. **대기 차량 수에 따른 신호 연장**

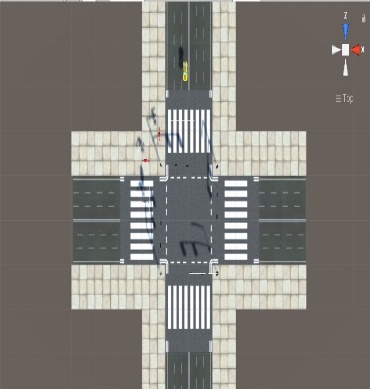


직진 신호의대기 차량 수의 변화에 따라 직진 신호의 길이를 조정한다. 직진 신호를   
대기하는 차량이 많을 경우 이를 반영하여 직진 신호를 더 길게 주게 된다.

1. **보행자 주의 신호**

보행자가 횡단보도에 진입한 것이 확인되면 보행자 주의 신호를 표출하게 된다. 이 신호로 보행자 주의 신호등을 사용하거나 보행자 신호 연장과 같은 부분에 사용이 가능하다.

**\*구현과정**

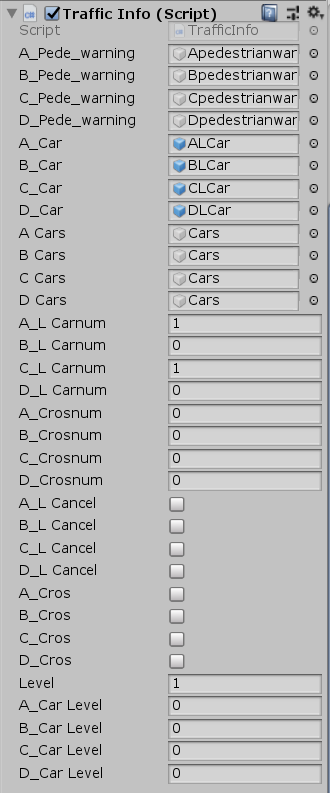
**1) 유니티 교차로 구현**

ARCADE – FREE Racing Car, KajamansRoads, Tarbo-CITY 에셋들을 사용해 4거리 교차로 맵을 구현하였고, UI를 이용해 정보를 표시한다.

**2) 신호체계 구현**

******<신호체계 구현>**

* 위 이미지들은 각 상황의 신호등이 켜진 것이다. 모두 한 UI캠버스 안에 구현된 것이며 이러한 UI캠버스는 각 구역별로 총 4개가 있다. 위 보행자 표시는 보행자의 수를 표시할 예정이었지만, 구현중의 변경으로 이후에 빠지게 되었다.
* 신호등은 정해진 시간동안 딜레이 후 변경되는 방법으로 진행되며 자체적인 딜레이가 있어야 하기 때문에 IEnumerator함수로 독립적으로 실행된다.
* TrafficInfo 클래스에서 보내는 직진 대기차량의 수에 대한 정보인 level을 받아 신호가 유지되는 딜레이에 추가시간을 준다.

**<Traffic Info 클래스 구현>**

* 서버에서 보낸 정보를 client에서 받고, 받은 정보를 Traffic Info에서 정리하게 된다.

각 구역의 Pede\_Warning(횡단보도 내의 보행자 주의 신호를 보여줄 오브젝트)

Car(좌회전 대기차량 오브젝트),

Cars(직진 대기차량 오브젝트),

LCarnum(좌회전 차량 대기정보),

Crosnum(횡단보도 내의 보행자정보),

L Cancel(좌회전 신호 캔슬 여부),

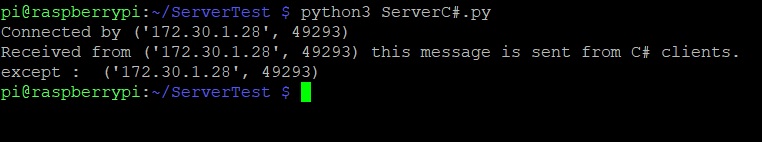
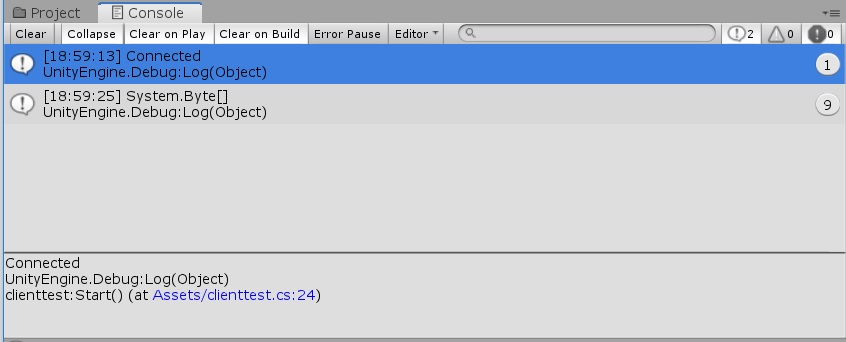
Cros(횡단보도 내의 보행자 유무)

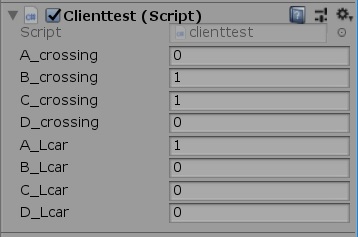
Level(차량의 대수를 레벨로 나눠 신호

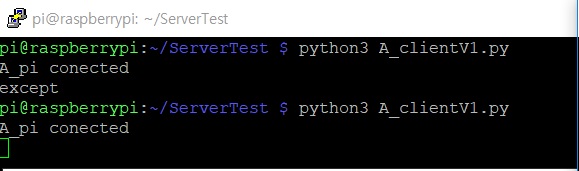
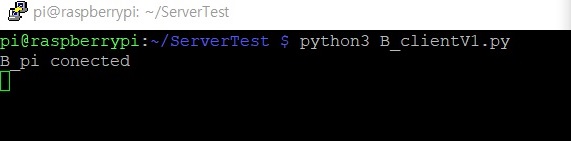
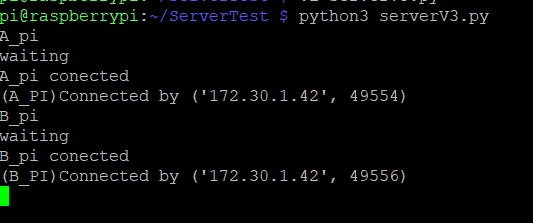
연장의 기준으로 사용할 변수)

Car Level(각 구역의 레벨)

3) **서버 클라이언트 구현**

**<클라이언트 개발>**

* 서버에서 전송된 각 구역의 정보들을 Client 클래스에서 전송받고 업데이트 한다. 그 업데이트된 정보들은 Traffic Info클래스에서 신호 제어를 위한 정보들로 가공되고, Traffic Control클래스에서 신호 제어가 일어나게 된다.

**<서버 구현>**

* 서버와 A클라이언트 B클라이언트가 연결된 이미지이다. 최대 A~D까지 4개의 클라이언트가 연결 가능하고, 신호 제어를 위한 클라이언트 하나까지 총 5개의 클라이언트와 통신한다.

**▶ 결론**

본 프로그램을 이용함으로써 교차로상에서 차량과 보행자의 상태를 실시간으로 파악하여 이를 신호와 연계함으로써 지능형 교통체계 및 자율주행 차량에 적용할 수 있을 것이다. 지능형 교통체계의 적용을 통해 교차로에서의 도로 흐름을 보다 빠르게 개선할 수 있을 것이며, 자율주행 차량에 교차로 안정성 기술에 접목할 경우 교차로상에서의 자율주행 안정성을 향상시킬 수 있을 것이다. 이와 더불어 보행자의 교통사고 예방이라는 효과 또한 기대할 수 있을 것이다.

1. **설계 구성 요소**
   1. **설계 구성 요소**

|  |  |  |
| --- | --- | --- |
| **설계**  **구성**  **요소** | **목표**  **설정** | 현재 교차로에서 많은 사고 발생. 따라서 교차로에 임베디드 기기를 설치하여 횡단보도의 보행자 및 도로의 차량을 검출하여 신호체계의 효율성을 높이도록 개발 |
| **합성** | OpenCV에서 Image Contour와 Haar cascade 알고리즘을 사용하여 보행자 및 차량을 인식하도록 하였다. Unity를 이용해서 서버에서 전송된 정보를 표시하도록 하였다. |
| **분석** | 요구사항 명세서, 유스케이스 명세서등을 작성하였고, 그에 맞게 시스템을 개발하였고, 보행자 및 차량에 대한 인식률을 개선하였다. |
| **제작** | 유스케이스 및 클래스 다이어그램을 만든 다음 제작하였다. |
| **시험** | 녹화 테스트의 경우, 교차로 영상을 녹화해서 해당 영상에 인식 테스트하였다. 실시간 테스트의 경우, 실제 교차로와 유사한 모형판을 제작하여 실시간으로 구동 테스트하였다. |
| **평가** | 보행자 및 차랑 인식 90% 인식률,  좌회전 차량 Haar의 정확도 96%, 컨투어의 정확도 95.7% |
| **제한**  **조건** | **산업**  **표준** | 해당사항 없음 |
| **경제성** | 해당사항 없음 |
| **안정성** | 서버는 Linux 환경에서 구동하도록 제작 |
| **미학** | 라즈베리파이 기기를 통해 얻어진 정보를 Unity를 사용해서 해당 인식 정보와 정보를 이용한 시뮬레이션을 제작 |
| **사회**  **영향** | 교차로상에서 보행자 및 차량의 상태를 실시간으로 파악하여 이를 신호체계와 연계함으로써 지능형 교통체계 및 자율주행 차량에 적용할 수 있을 것이다. 또한 보행자의 교통사고를 예방할 수 있을 것으로 기대하고 있다. |

* 1. **. 팀 목표 대비 달성 정도**

|  |  |
| --- | --- |
| **팀 목표** | 보행자 및 차량 인식, 영상처리, 인공지능, 서버, 웹으로 이루어진 시스템을 구현해야 하므로 시스템의 전체적인 구성을 이해하고 제작하도록 한다. 그리고 인식 부분에서는 90%이상의 정확도를 구현하는 것이다. |
| **달성 정도** | 보행자 및 차량 인식을 구현하면서 90%이상의 정확도를 보여주었다. 그리고 다만 인공지능 부분에서 신호 제어에 중요하지 않고 시간이 걸리는 몇가지 요소를 구현에서 제외하고 변경하였다. 하지만 그 외의 부분에서 구현하고 결합하면서 최종적으로 안정성 있는 시스템을 구축하였고 팀원들의 전체적인 프로그래밍 경험과 능력이 향상되었다. 따라서 달성정도는 90%이다. |

1. **향후 개선 계획**

* **구현중 끝내지 못하거나, 변경한 요소 구현**

현재 인식 요소는 직진 차량 영역에서의 차량의 수, 좌회전차량 유무. 횡단보도 영역에서의 보행자 유무 파악 기능만 구현이 되었다. 원래의 설계 요소 중 보행자의 수, 보행자의 이동방향은 구현하지 못했다. 이 요소를 향후 더 구현할 예정이고, 더 발전하여 Image Contour에서 인식되는 물체에 고유 ID를 부여함으로써 보행자 영역의 보행자 수는 물론이며, 정교한 차량의 면적 계산을 통하여 오토바이, 트럭, 버스 등 차량의 종류까지 분류하도록 개선할 것이다. 추가적으로 영역 지정 범위를 넓힘으로써 보행자 대기장소에서의 보행자 유무와 무단횡단, 신호위반 등의 예기치 못한 경우들을 보완할 것이다.

* **한국의 변화무쌍한 날씨의 경우 (창의설계 경진대회 질문)**

현재 positive sample 900장과 negative sample 2500장을 분류하여 haar training을 통해 얻은 haar classifier는 96%의 인식률을 보여준다. 비와 눈 등과 같은 기후 변화에 대한 다양한 학습 데이터의 수집과 영상 테스트를 통하여 정확도를 높이고, 24시간 가동되는 CCTV의 특성에 따라 구현할 것이다.

* **실제 신호 조작기기에 적용**

2017년 경찰철에서 배포한 교통신호제어기 표준규격서를 보면 교통신호 제어기는 매우 복잡한 기능을 가지고 있는 것을 볼 수 있다. 특히 본 프로젝트의 신호제어를 위한 부분을 보면, 제어의 기능만 해도 중앙제어모드, 지역제어모드, 안전제어 모드가 있다. 또, 장치 간 통신규약은 매우 다양한 프로토콜이 사용되는 것을 볼 수 있다. 이러한 교통신호 제어기에도 사용할 수 있도록 구현하여 실제로 사용될 수 있도록 개선할 수 있을 것 이다.

* **WIFI뿐만 아니라 4G, 5G에 적용**

현재 ITS 시스템을 사용하고 있는 도시들은 통신회사의 지원을 받아 4G 무선인터넷을 통해 ITS시스템을 설계하고 사용하고 있다. 이는, 인터넷과 모뎀을 따로 설치해야 하고, 보안에도 문제가 있을 수 있는 WIFI보다 효율적일 것이다. 따라서 시한폭탄조의 교차로 보행자 및 차량 검지 시스템도 서버와 통신할 때에 4G와 5G의 사용도 고려해야 할 것이다.

1. **첨부**
2. **창의설계 경진대회 발표 자료**

**Q. 테스트 환경이 이상적인 환경에서 이루어 졌는데 혹시 비나 눈이 오는 환경에 대해서는 생각해보았는가?**

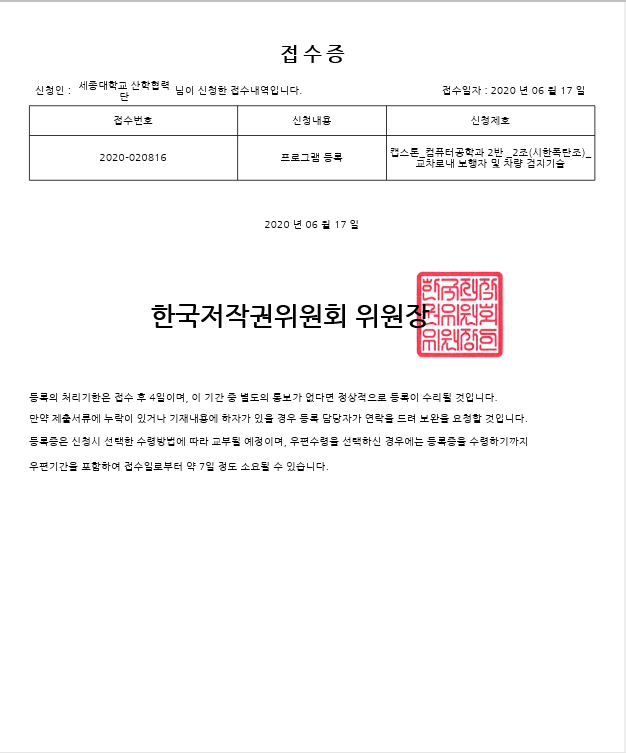
**A.** 그것에 대해서는 생각을 하지 못했다. 테스트 진행을 위해 교차로를 직접 촬영하는 과정에서 우리 조원들이 직접 교차로 영상을 촬영하여 테스트를 진행했는데, 그 부분에서 한계가 있었던 것 같고, 구현에 집중하여 날씨는 생각하지 못한 것 같다. (이는 추후에 더 개발할 사항으로 생각하여 향후 개선사항 항목에 적어두었다.)

**Q. 활용 가능성에서 차량의 통행 속도 약30% 가량 증가와 지체시간 약 20~30초가량 감소는 어떠한 근거를 바탕으로 나온 것 인가?**

**A.** 화성에서 먼저 실시된 ITS시스템에서 우리 기능과 비슷한 ITS 시스템 (좌회전 차량이 있을 때만 좌회전 신호) 이 있어 그 결과를 가져왔다.

**Q. 그렇다면 그 기존의 CCTV기반 ITS시스템말고 시한폭탄조의 시스템을 사용해야하는 이유**

**A.** 좌회전 신호뿐만이 아닌, 차량 대기수에 따른 신호 연장으로 교차로에서의 효율을 도모할 수 있고, 보행자 주의 신호로 보행자에 안전을 도모 할 수 있으며 이 정보를 활용해 자율주행차량의 교차로 안정성을 증가시킬 수 있다.

1. **SW 프로그램 등록**

SW등록 추가사항은 첨부파일에 별도로 첨부

1. **스크린샷, 실내, 노트북, 컴퓨터이(가) 표시된 사진

   자동 생성된 설명GitHub 등록 내용 요약 및 등록 (GitHub에 등록한 source는 별도의 압축파일 제출)**

https://github.com/jo1132/Capstone\_2\_timebomb\_Sejong

용량의 제한 때문에 유니티 시뮬레이션 파일과, 시연용 mp4데이터 올리지 못함

제출시에는 전부 제출할 예정

1. **TOPCIT 응시**

**13-1 TOPCIT응시 후 느낀 점**

|  |  |
| --- | --- |
| **조현기** | 시험을 통해 현재 기업 등의 회사에서 인재에게 무엇을 원하는지 볼 수 있었다. 프로그래밍 언어에 대한 기본적인 지식부터 데이터 베이스, 보안등의 능력과 더 나아가 IT회사에서 근무할 역량을 체크할 수 있었던 것 같다.  반대로 시험에 출제된 문제를 보며 기업에서 어떤 능력을 원하는지 엿볼 수 있었던 것 같다. |
| **구본학** | TOPCIT을 통해 개인의 역량이 부족하다는 점을 느꼈다. 생소한 파트도 있었으나, 이를 바탕으로 부족한 부분에 대하여 추가적으로 공부해야겠다는 생각이 들었다. 좋은 경험이 된 것 같다. |
| **조남규** | Topcit 시험을 접해본 것은 이번이 처음이었으나 이번 응시를 통해 본인의 ICT분야 관련 문제풀이 능력 및 지식을 점검해보는 계기가 되었습니다. |
| **이세영** | 실제 TOPCIT 시험을 접하면서 보안 등 생소한 분야도 있었으나, IT 업무에 대해서 전체적으로 필요한 지식이 어떤 것이 있는지 이번에 많이 알 수 있었다. |

1. **논문**
2. **특허**
3. **외부 경시대회**
4. **그 외 결과물**