

RC카를 활용한 웹 기반 원격 화재 감시 및 제어 시스템

류우상

목차

01

프로젝트 목표

02

개발 환경

03

시스템 아키텍처

04

모듈특징 및 동작분석

05

프로젝트 결과

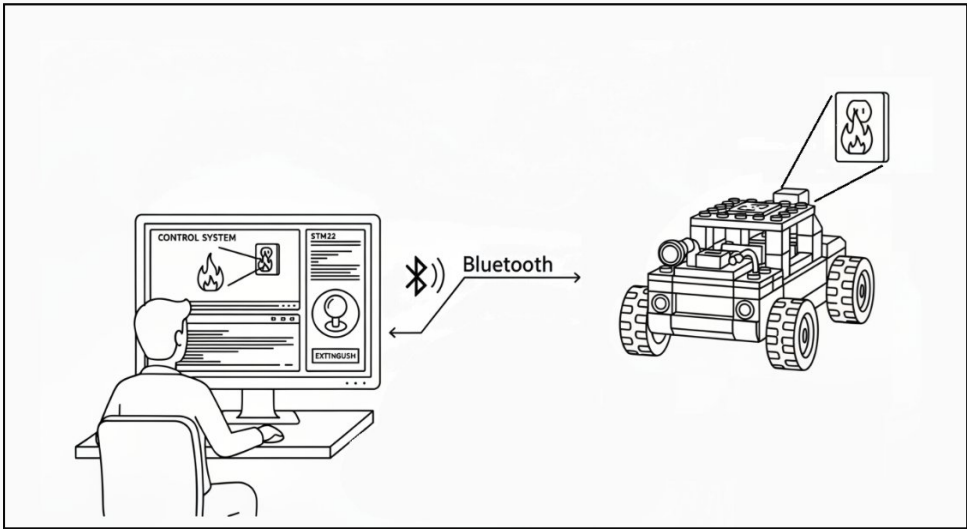
06

프로젝트 고찰

01

프로젝트 목표

1.프로젝트 목표



1. 상시 감시가 어려운 환경에서 화재 발생 시 골든타임 확보를 위한 원격 화재감지 시스템 구축
2. 자율 순찰 및 원격 감시를 통해 초기 대응 기능을 구현
3. 오감지로 인한 오작동을 방지하기 위해 수동 제어 기능을 추가하여 안전성 확보

프로젝트 일정

10.16~10.19

10.20~10.23

10.24~10.27

-하드웨어 구성 및 센서 테스트

-RC카 순찰 기능 및 소화 기능 추가

-시스템 통합 및 테스트

-아두이노 보드를 활용하여 테스트 환경 제작

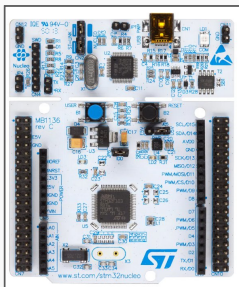
-Node.js 서버 구축 및 웹 UI 구성

-웹 UI 수정 및 개선 및 발표자료 준비

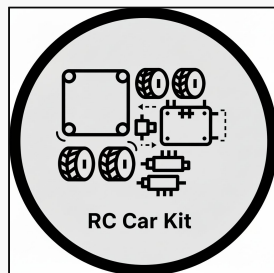
02

개발환경

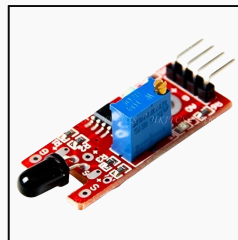
Hardware



1. STM32F103RB (Cortex-M3)



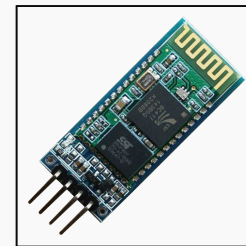
2. RC Car Kit



3. KY026(화재감지센서)



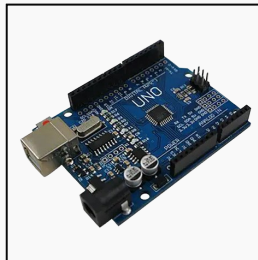
4. 피에조 부저



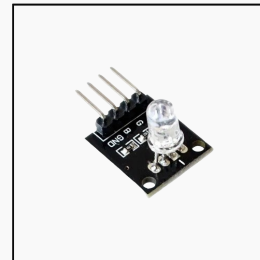
5. HC-06



6. 서보 모터



7. 아두이노 UNO



8. ICF 1224(IR 송신 모듈)

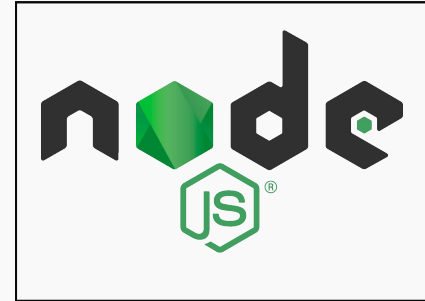
Software



1. STM32 Cube IDE



2. Python

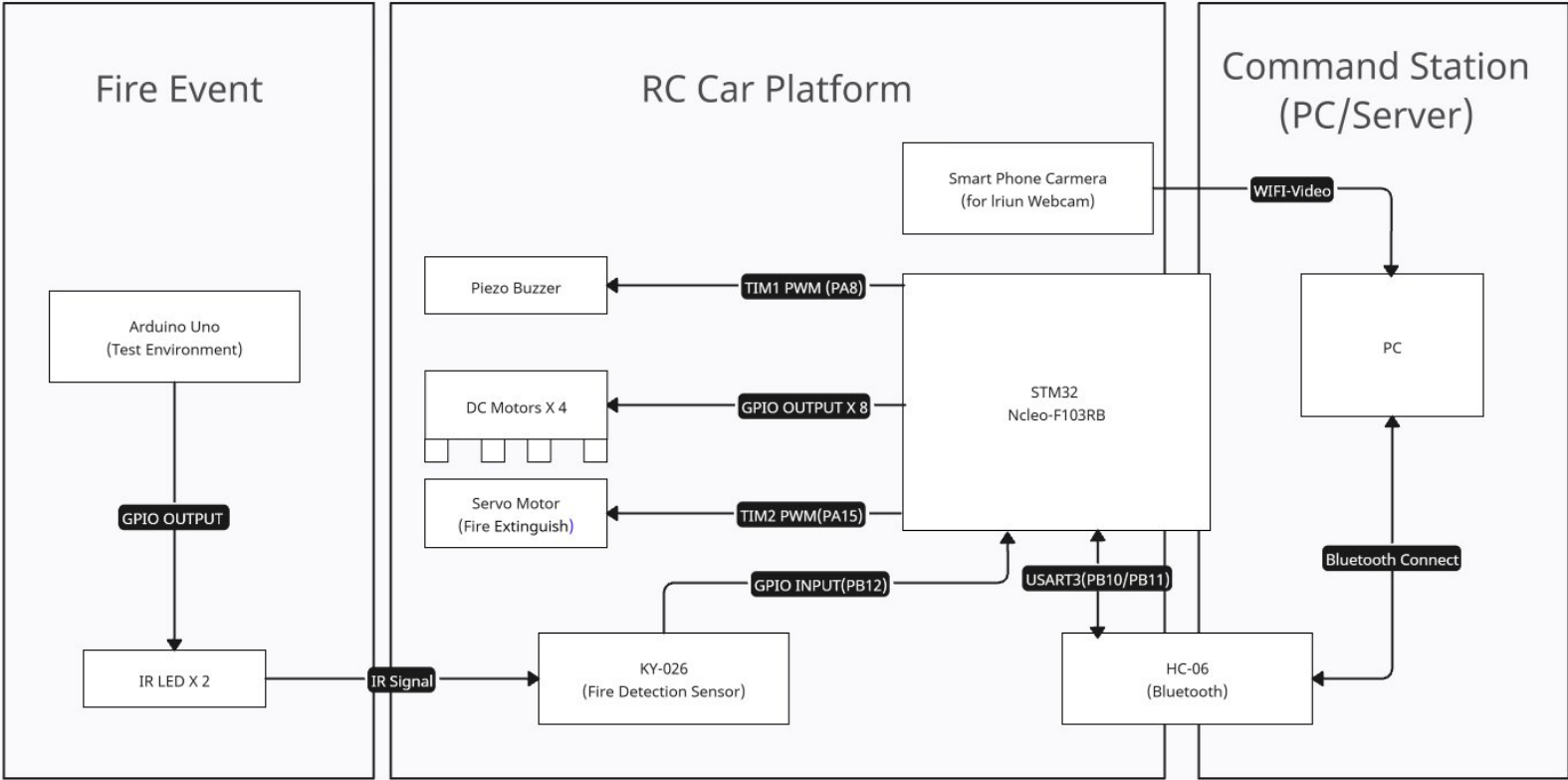


3. Node js

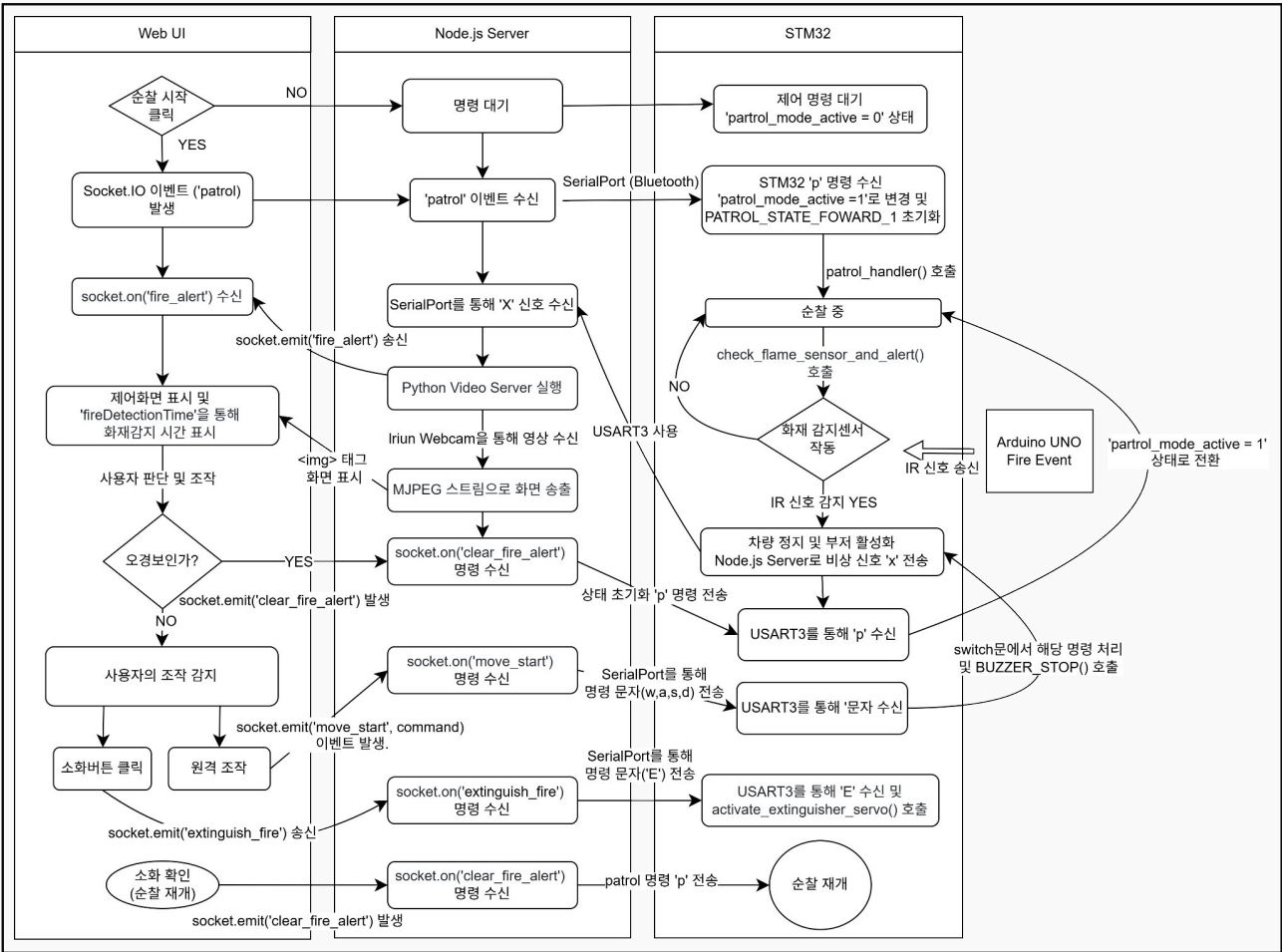
03

시스템 아키텍처

Hardware Architecture



3.시스템 아키텍처



Software Architecture

04

모듈 특징 및 동작분석

주요 모듈 특징

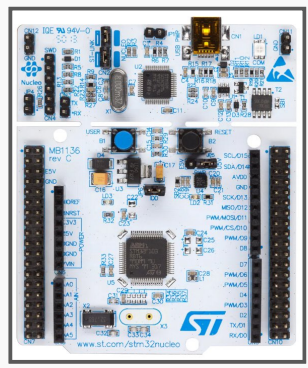
모듈 역할

-시스템의 메인 컨트롤 역할

STM32 성능

- CPU: ARM Cortex-M3
- 메모리:128 KB Flash
- SRAM: 20kB
- 다수의 TIM,USART,ADC, GPIO 핀 51개를 제공

STM32 Nucleo Board



활용 Peripheral

- TIM(TIM1,TIM2): PWM 신호 생성을해 부저 제어 및 서보 모터 제어에 활용
- USART3: Node.js 서버와 통신을 위해 사용
- GPIO (모터, 센서): 모터드라이버 및 센서에 사용

설계 주의사항

- GPIO 제어 핀 중복 할당에 주의하며 핀 설정

주요 모듈 특징

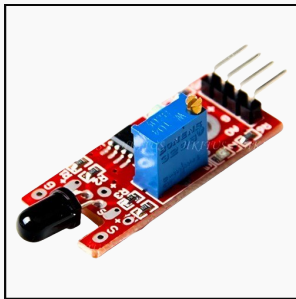
모듈 역할

- RC카 주변의 화재 징후를 1차적으로 감지
- `check_flame_sensor_and_alert()` 함수를 통해 주기적으로 화재 발생 여부를 확인
- 화재 감지 시 즉시 부저를 활성화 시킴

STM32 인터페이스

- 센서의 디지털 출력(D0)핀을 STM32의 PB12(GPIO Input)에 연결
- 입력 신호가 없을시 Low 상태를 유지 및 Floating 입력으로 인한 오작동 방지

KY-026 화재감지센서



감지원리

- 불꽃에서 방출되는 760nm ~ 1100nm 파장 대의 적외선 복사 에너지를 수신 및 작동

설계고려사항

- 동작 전압: 3.3V ~ 5V
- 외부 광원 노이즈에 대한 취약성이 존재
- 이를 극복하기 위해 아두이노 기반의 IR 테스트 환경을 만들어 사용

주요 모듈 특징

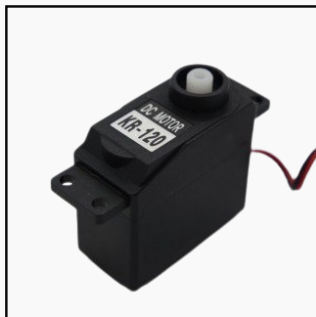
KR 120 모터

모듈 역할

-자율 순찰 및 수동제어를 위한 RC카의 구동

STM32 인터페이스

-모터 드라이버의 각 채널에 연결된 총 8개의 GPIO Output 핀을 사용



제어방식

-HAL_GPIO_WritePin() 함수를 통해 각 핀의 HIGH/LOW 상태를 조절 모터의 회전방향을 결정

-smartcar_forward(), smartcar_right() 등의 함수를 사용하여 좌회전 우회전 구현

설계고려사항

-최대 토크: 5V에서 2.2kgf.cm

-rpm 특성: 5V에서 120rpm

주요 모듈 특징

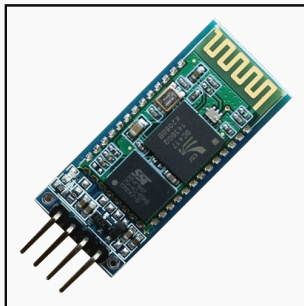
HC-06 블루투스 모듈

모듈 역할

-STM32와 Node.js 서버(PC) 간의 무선
시리얼 통신 채널을 구축

STM32 인터페이스

-HC-06 모듈의 TX/RX 핀을 STM32의
USART3 통신 핀인 PB10, PB11에 연결



제어방식

-HAL_UART_Receive() 함수를 호출하여 수신 버퍼를 지속적으로 확인

-Node.js 서버로부터 전송되는 제어 명령(w,a,s,d)을 처리

-HAL_UART_Transmit() 함수로 화재 경보 신호('X')를 서버로 전송

설계고려사항

-동작전압: 3.3V~5V

-통신속도: 115200 bps

주요 모듈 특징

모듈 역할

- 소화 장치를 구현하기 위해 동작
- 모의소화탄을 발사하는 역할

STM32 인터페이스

- 서보 모터의 신호 핀을 STM32의 PA15 핀에 연결

SG90 서보모터



제어방식

- TIM2의 PWM 모드를 활용
- _HAL_TIM_SET_COMPARE() 함수를 통해 서보의 각도를 정밀 제어
- activate_extinguisher_servo() 함수 내에서 '장전-발사-복귀' 시퀀스를 구현하여 소화 동작을 구현

설계고려사항

- 마이크로 서보 모터: 1.8kg/cm 토크
- 동작전압 : 5V

주요 모듈 특징

모듈 역할

-화재가 감지 됐을 시 경고음을 부저를 통해 구현

STM32 인터페이스

-부저의 신호핀을 STM32의 PA8핀에 연결

피에조 부저



제어방식

- TIM1의 PWM 모드를 활용
- Prescaler와 Period를 계산하여 PWM 파형을 생성

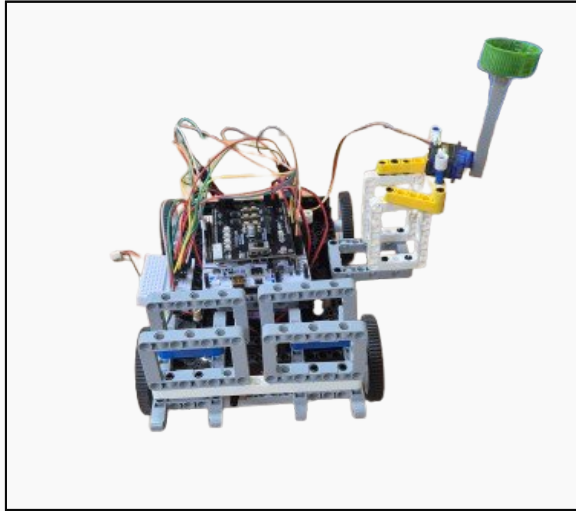
설계고려사항

- 동작전압: 3.3V ~ 5V
- sConfigOC.Pulse값을 조정 적정소리를 출력

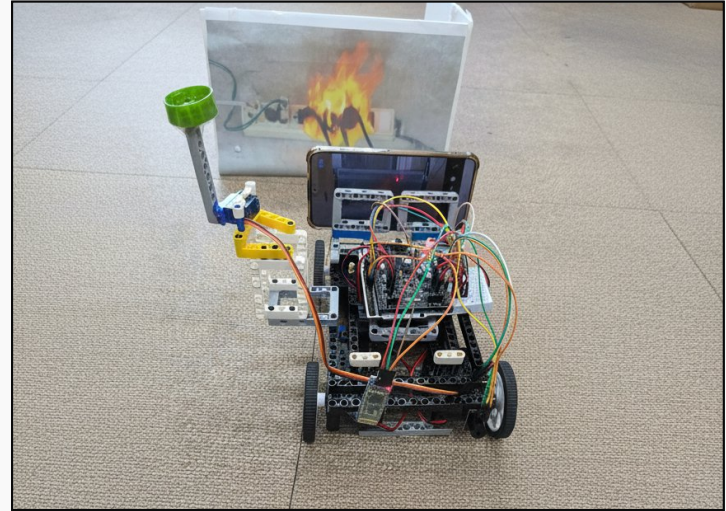
05

프로젝트 결과

RC카 최종 구성

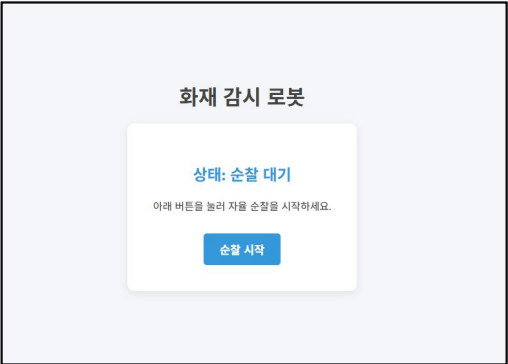


1.전체 구성

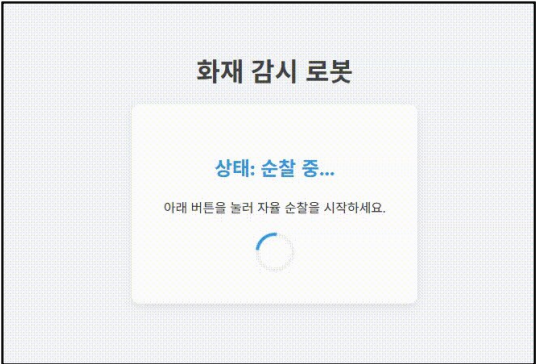


2.테스트 모습

웹 UI



1.순찰 대기



2.자율 순찰



3.화재감지 후 원격제어

동작 영상



1.화재 감지



2.소화 동작



3. 순찰 복귀

06

프로젝트 고찰

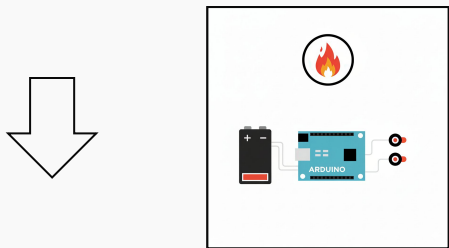
결과 및 달성 목표

프로젝트 목표	달성 목표	달성 여부
화재 감시 시스템 개발	-KY-026 센서를 통해 화재 감지 및 피에조 부저를 통해 화재 경보 기능 구현	V
자율 순찰 및 원격 감시 기능 구현	-patrol_handler() 함수를 통해 자율 순찰 기능 구현 성공 및 Python 영상 서버를 통해 원격 감시 기능 구현	V
수동 제어 및 소화 기능 구현	-블루투스 시리얼 통신과 웹 소켓 기능을 활용하여 원격 제어 구현 및 서보 모터를 통해 소화 기능 구현	V
오작동 감소를 통한 시스템 안전성 향상	-Opencv 기반의 Fire Follower.py를 통해 실제 화재 영역을 구분하여 오작동 방지 -오경보시 오작동버튼을 통해 순찰복귀하는 기능 구현	V

문제발생 및 해결과정

1.화재감지 센서의 문제 발생

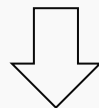
-화재감지 센서의 경우 외부 광원 및 감지 각도
로 인해 오작동 문제가 발생



-아두이노와 IR 모듈을 통해 가상 화재 테스트
환경을 구축하여 테스트 안전성을 향상

2.유선 웹캠 사용으로 인한 문제 발생

-초기 유선 웹캠을 사용 시 RC카의 활동 반경이
제한되는 문제 발생



-유선 웹캠 대신 'Iriun Webcam' 앱을 활용하여 스마트폰 무선 스트리밍으로 대체

-RC카의 활동 반경을 넓히고 테스트 안전성 향상

개선 방안 및 향후 계획

1.RTOS 도입

-현재 'while(1)' 기반의 단일 루프 스케줄링을 FreeRTOS와 같은 실시간 운영체제 기반으로 전환

-각 태스크를 독립적으로 관리하여 시스템의 효율성 및 안전성 향상

-추후 복잡한 기능 추가시에도 안전성 향상 가능

2.화재 감지 기능 향상

-현재 OpenCV 기반의 화재 탐지 파일은 객체의 색상 정보에 의존
이로 인해 오탐지 가능성 존재

-이를 극복하기 위해 경량 AI 모델 YOLO등의 모델을 적용하여 색상이 아닌 불꽃의 형상과 패턴을 학습 및 탐지

감사합니다!