스마트 주차 시스템

21900764 최준혁, 22100113 김성빈

한동대학교

IoT 시스템 설계

설계과제2

12/2/2023

# 요약

이 보고서는 스마트 주차 시스템 프로젝트를 소개하고 구현한 내용을 상세히 설명한다. 프로젝트의 목적은 주차 공간을 효율적으로 관리하고 사용자에게 편의를 제공하는 것이며, IoT 기술과 센서를 활용하여 주차된 차량의 수를 실시간으로 모니터링한다. 기술적 배경, 문제 정의, 기능 Flowchart, UI부터 최종 구현까지의 과정과 소스 코드를 포함 하고 있고, 발생한 문제와 해결 방법, 그리고 향후 계획도 다룬다. 또한, 팀워크와 실험 파트너와의 협업에 대한 평가도 포함하고 있으며, 프로젝트를 통해 얻은 경험과 능력 강화에 대한 기술이 결론부에 나타나 있다.

# 서론

## 1.1 프로젝트 목적 및 의의

우리 프로젝트는 스마트 주차 시스템을 구현함으로써 주차 공간을 효율적으로 관리하고, 사용자에게 편의를 제공하는 것을 목적으로 하고 있다. 이 시스템은 IoT 기술과 센서를 활용하여 주차된 차량의 수를 실시간으로 모니터링한다.

## 1.2 기술적 배경

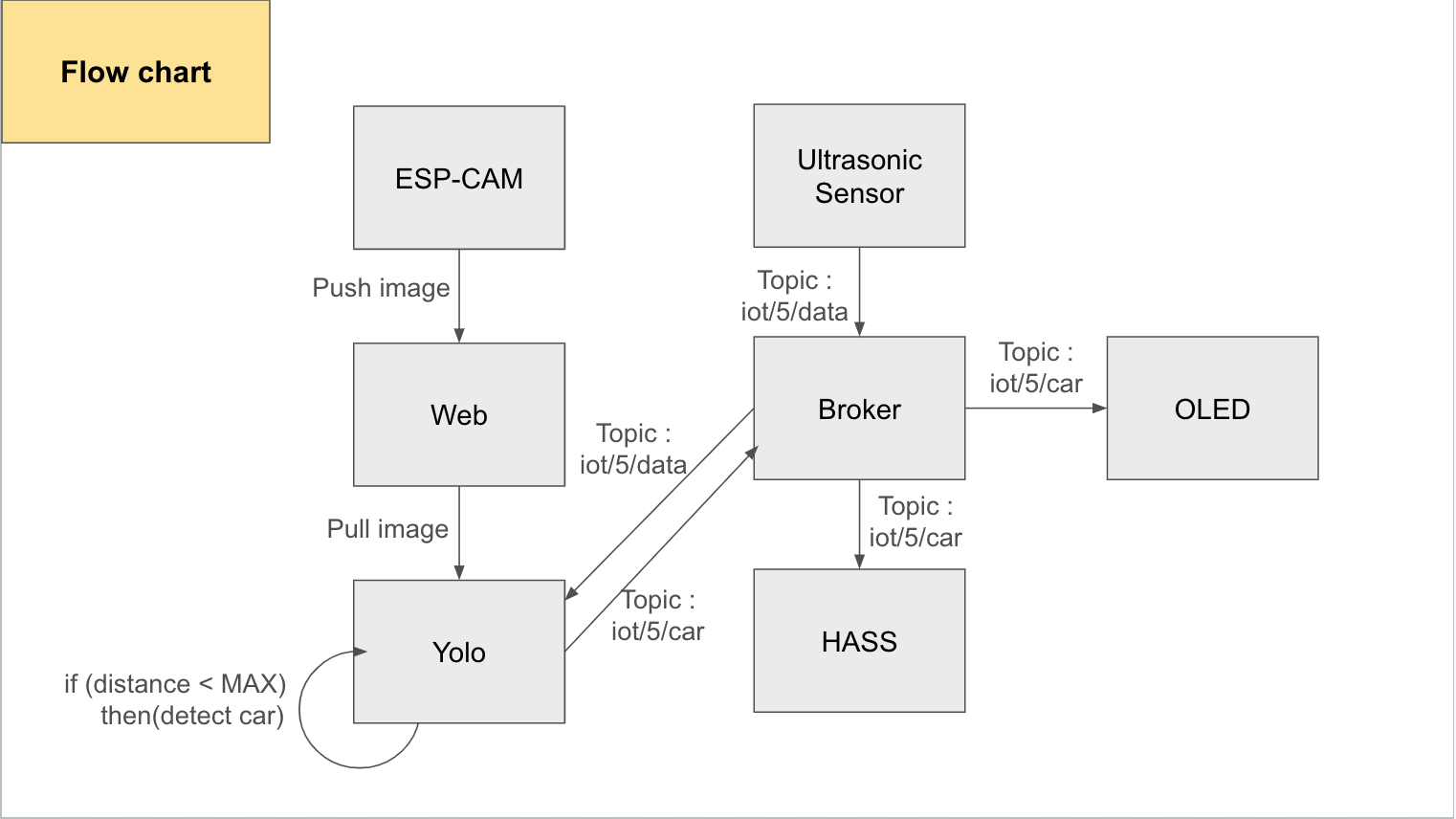
프로젝트를 시작하기 전에 ESP32-CAM 및 초음파 센서와 같은 하드웨어에 대한 기술적 배경을 습득했다. 또한, MQTT 통신과 웹 서버 구축에 대한 이해를 바탕으로 개발을 진행했다.

## 1.3 문제 정의

기존 무인 주차 시스템에 존재하는 문제점들을 파악하고, 해결하는 방법을 선택했다. 크게 정리한 문제로는: 야외 환경 변수, 전력 효율, 주차 공간 정보로 정의했다. 먼저, 초음파로만 관리하는 무인 주차 시스템[1]은 거리 감지가 된 물체가 자동차인지 확인하는 방법이 존재하지 않아서 센서 한개로만 작동하는 방식은 다양한 상황과 환경을 고려하지 못해 안정성이 떨어진다고 판단했다. 다음은 전력 효율이다, 이전 예시와 반대로 각 주차 자리마다 카메라를 한개씩 설치하여 관리하는 시스템도 존재한다. 이는 심한 전력 낭비를 불러올 것으로 예상했다. 간단한 ESP32-CAM을 사용한다고 하더라도, 최대 310 mA[2]를 사용한다. 반면에 HC-SR04 초음파 Sensor[3]는 최대 20 mA로 훨씬 적게 사용한다. 이 점을 통해 정확한 카메라를 사용하여 안정성을 보장하되, 초음파 센서와 협동하여 저전력 효과를 목표로 잡게되었다. 마지막으로, 기존 야외 주차장들은 직접 주차장에 들어가기까지 하거나, 입구에서 확인할 수 있는 디스플레이를 통해 상태가 확인 가능했다. 하지만, 이는 야외 주차장을 이용하는 사용자들의 대부분의 목적과 다르게 작용한다. 한 기사[4]에서 주차장에 가서야 자리가 밀리는 것을 확인하여 수십 미터 줄을 서 대기하는 현상이 빈번하게 일어난다고 보도된다.

# 본론

## 2.1 기능 Flowchart



## 2.2 UI부터 최종 구현까지

## 2.2.1 기본적인 UI 구성

프로젝트 초기에는 사용자에게 직관적인 UI를 제공하기 위해 MQTT 통신과 Serial Monitor를 통한 간단한 인터페이스를 구현했다. 사용자는 실시간 주차 차량 수를 확인할 수 있었다. 후 에는 HASS (Home Assistant)를 사용해서 언제 어디서든 누구나 확인할 수 있는 웹 인터페이스로 업그레이드를 했다.

## 2.2.2 알고리즘 설계

주차 공간 모니터링을 위한 알고리즘은 ESP32-CAM의 이미지 캡처를 포함한다. 또한, 초음파 센서를 활용하여 차량 감지를 수행하고 MQTT를 통해 데이터를 전송한다. 위 기능 Flowchart를 따라 설명을 하면, ESP-CAM을 이용하여 주차장 전체를 찍어서 로컬 IP에 올린다. 이 영상을 Yolo를 활용한 python 프로그램을 이용해서 받는다. 이때 주차장 입출구에 설치된 초음파 센서에 차량이 감지되면 Yolo가 10 프레임간 주차장 내의 차량 수를 감지합니다. 업데이트된 차량 수는 MQTT를 통해 publish 되고 HASS와 OLED Display를 통해 차량 수를 표시합니다.

## 2.2.3 최종 구현 결과

프로젝트의 최종 결과물은 안정적으로 동작하며 사용자에게 정확한 주차 정보를 제공한다.

## 2.3 Source Code

스마트 주차 시스템에서 사용된 source code에는 크게 ESP-CAM을 위한 ESP-CAM.ino, 초음파 센서를 위한 UltraSonic.ino, Yolo를 실행하는 cam\_mqtt\_countCar\_when\_signal.py, 마지막으로 HASS의 configuration.yaml 파일이 있다.

## 2.3.1 ESP-CAM.ino

esp32cam 라이브러리를 활용하여 이미지를 캡쳐한 후 WebServer 라이브러리를 활용하여 해당 이미지를 웹서버에 올린다.

| #include <WebServer.h>  #include <WiFi.h>  #include <esp32cam.h>  const char\* WIFI\_SSID = "wifi\_ssid";  const char\* WIFI\_PASS = "wifi\_password";  WebServer server(80);  static auto loRes = esp32cam::Resolution::find(320, 240);  void serveJpg()  {  auto frame = esp32cam::capture();  if (frame == nullptr) {  Serial.println("CAPTURE FAIL");  server.send(503, "", "");  return;  }  Serial.printf("CAPTURE OK %dx%d %db\n", frame->getWidth(), frame->getHeight(),  static\_cast<int>(frame->size()));  server.setContentLength(frame->size());  server.send(200, "image/jpeg");  WiFiClient client = server.client();  frame->writeTo(client);  }  void handleJpgLo()  {  if (!esp32cam::Camera.changeResolution(loRes)) {  Serial.println("SET-LO-RES FAIL");  }  serveJpg();  }  void setup(){  Serial.begin(115200);  Serial.println();  {  using namespace esp32cam;  Config cfg;  cfg.setPins(pins::AiThinker);  cfg.setResolution(loRes);  cfg.setBufferCount(2);  cfg.setJpeg(80);  bool ok = Camera.begin(cfg);  Serial.println(ok ? "CAMERA OK" : "CAMERA FAIL");  }  WiFi.persistent(false);  WiFi.mode(WIFI\_STA);  WiFi.begin(WIFI\_SSID, WIFI\_PASS);  while (WiFi.status() != WL\_CONNECTED) {  delay(500);  }  Serial.print("http://");  Serial.println(WiFi.localIP());  Serial.println(" /cam-lo.jpg");  server.on("/cam-lo.jpg", handleJpgLo);  server.begin();  }  void loop()  {  server.handleClient();  } |
| --- |

## 2.3.2 UltraSonic.ino

초음파 센서는 초음파를 이용해서 센서와 사물간의 거리를 측정할 수 있다. 이를 활용하여 주차장 출입구에서 차량의 출입을 감지하고 차량과의 거리를 MQTT를 활용하여 Publish하여 거리 정보를 다른 프로그램에서 거리 정보를 활용할 수 있도록 한다. 아래는 새롭게 추가된 초음파 센서 관련 코드이다.

| #include <Adafruit\_GFX.h>  #define D5 14  #define D6 12  // define sound velocity in cm/uS  #define SOUND\_VELOCITY 0.034  #define DIST\_THRESH 70  const int trigPin = D6;  const int echoPin = D5;  String mqttTopic = "iot/5";  String carMqttTopic = "iot/5/car";  String carCount = "Car Count: 0";  // defines variables  long duration;  int distance;  void setup() {  Serial.begin(115200);    bool res;  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output  pinMode(echoPin, INPUT); // Sets the echoPin as an Input  }  void loop() {  // Clears the trigPin  digitalWrite(trigPin, LOW);  delayMicroseconds(2);  // Sets the trigPin on HIGH state for 10 microseconds  digitalWrite(trigPin, HIGH);  delayMicroseconds(10);  digitalWrite(trigPin, LOW);  // Reads the echoPin, returns the sound wave travel time in microseconds  duration = pulseIn(echoPin, HIGH);  // Calculating the distance  distance = duration \* SOUND\_VELOCITY / 2;  // Prints the distance on the Serial Monitor  Serial.print("Distance: ");  Serial.println(distance);  // Check if the distance is less than 30 cm  if (distance < DIST\_THRESH) {  JSONVar myObject;  myObject["dist"] = (int) distance;  String jsonString = JSON.stringify(myObject);  const String pubTopic = mqttTopic + "/data";  // Publish the payload to the MQTT topic  mqttClient.publish(pubTopic, jsonString);  }  delay(2000);  } |
| --- |

## 2.3.3 cam\_mqtt\_countCar\_when\_signal.py

cam\_mqtt\_countCar\_when\_signal.py 파일에서는 ESP-CAM이 만든 Web server에서 주차장 이미지를 불러올 수 있다. 또한 Ultrasonic 센서가 publish한 거리 정보를 Subscribe해서 차량 출입지 Yolo를 활용하여 주차장에 몇 대의 차량이 있는지 세서 iot/5/car topic으로 Publish 한다.

이때 car detection은 10회 반복하여 Yolo의 차량 감지 실패를 최소화 한다.

| import cv2  import cvlib as cv  import urllib.request  import numpy as np  from cvlib.object\_detection import draw\_bbox  import multiprocessing  import paho.mqtt.client as mqtt  import time  import json  def on\_message(client, userdata, message, car\_count):  payload = message.payload.decode("utf-8")  print(f"Received message: {payload}")  car\_count.value = int(payload.split(":")[1].strip())  def on\_data\_message(client, userdata, message, distance, cam\_on\_flag, car\_in\_flag, car\_count, mqtt\_topic\_car):  payload = message.payload.decode("utf-8")  print(f"Received data message: {payload}")  try:  data = json.loads(payload)  dist\_value = data.get('dist')  if dist\_value is not None:  distance.value = float(dist\_value)  if distance.value <= 70 and not car\_in\_flag.value:  cam\_on\_flag.value = 70  car\_in\_flag.value = True  elif distance.value > 70 and car\_in\_flag.value:  cam\_on\_flag.value = 70  car\_in\_flag.value = False    # Publish the updated car count  client.publish(mqtt\_topic\_car, f'Car Count: {car\_count.value}')  except json.JSONDecodeError as e:  print(f"Error decoding JSON: {e}")  def setup\_mqtt(mqtt\_broker, on\_message\_callback, mqtt\_topic):  client = mqtt.Client()  client.on\_message = on\_message\_callback  client.connect(mqtt\_broker)  client.subscribe(mqtt\_topic)  client.loop\_start()  return client  def run2(url, mqtt\_broker, mqtt\_topic\_car, car\_count, cam\_on\_flag, published\_time, car\_in\_flag, distance):  cv2.namedWindow("detection", cv2.WINDOW\_AUTOSIZE)  client = setup\_mqtt(mqtt\_broker, lambda client, userdata, message: on\_message(client, userdata, message, car\_count), mqtt\_topic\_car)  while True:  try:  with urllib.request.urlopen(url) as img\_resp:  imgnp = np.array(bytearray(img\_resp.read()), dtype=np.uint8)  im = cv2.imdecode(imgnp, -1)  bbox, label, conf = cv.detect\_common\_objects(im)  if cam\_on\_flag.value > 0:  car\_count.value = label.count('car') + label.count('bus') + label.count('truck')  im = draw\_bbox(im, bbox, label, conf)  cv2.putText(im, f'Car Count: {car\_count.value}', (10, 30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (0, 255, 0), 2)  print("car count: ", car\_count.value)  cv2.imshow('detection', im)  key = cv2.waitKey(1) & 0xFF  if key == ord('q'):  break  if cam\_on\_flag.value > 0:  cam\_on\_flag.value -= 1  current\_time = time.time()  if current\_time - published\_time.value >= 1:  # Publish the updated car count  client.publish(mqtt\_topic\_car, f'Car Count: {car\_count.value}')  published\_time.value = current\_time  except Exception as e:  print(f"Error in run2: {e}")  client.loop\_stop()  cv2.destroyAllWindows()  def run3(mqtt\_broker, mqtt\_topic\_data, distance, cam\_on\_flag, car\_in\_flag, car\_count, mqtt\_topic\_car):  client = setup\_mqtt(  mqtt\_broker,  lambda client, userdata, message: on\_data\_message(client, userdata, message, distance, cam\_on\_flag, car\_in\_flag, car\_count, mqtt\_topic\_car),  mqtt\_topic\_data  )  while True:  time.sleep(1)  if \_\_name\_\_ == '\_\_main\_\_':  manager = multiprocessing.Manager()  car\_count = manager.Value('i', 0)  published\_time = manager.Value('d', 0.0)  cam\_on\_flag = manager.Value('i', 0)  car\_in\_flag = manager.Value('b', False)  distance = manager.Value('f', 0.0)  # MQTT setup  mqtt\_broker = "sweetdream.iptime.org"  mqtt\_topic\_car = "iot/5/car"  mqtt\_topic\_data = "iot/5/data"  url = 'http://192.168.0.143/cam-lo.jpg'  print("started")  process2 = multiprocessing.Process(target=run2, args=(url, mqtt\_broker, mqtt\_topic\_car, car\_count, cam\_on\_flag, published\_time, car\_in\_flag, distance))  process3 = multiprocessing.Process(target=run3, args=(mqtt\_broker, mqtt\_topic\_data, distance, cam\_on\_flag, car\_in\_flag, car\_count, mqtt\_topic\_car))  process2.start()  process3.start()  process2.join()  process3.join() |
| --- |

## 2.3.4 configuration.yaml

다음은 HASS에서 사용된 configuration.yaml 파일이다. MQTT로 주차장 내 차량 수를 받아서 HASS card를 추가하여 사용자가 간편하게 차량 수를 확인할 수 있게 설계되었다.

| # Loads default set of integrations. Do not remove.  default\_config:  homeassistant:  name: SmartParking  latitude: 36.103295  longitude: 129.387009  elevation: 30    lovelace:  mode: yaml  #mode: storage  # Load frontend themes from the themes folder  frontend:  themes: !include\_dir\_merge\_named themes  # Text to speech  tts:  - platform: google\_translate  #automation: !include automations.yaml  #script: !include scripts.yaml  #scene: !include scenes.yaml  mqtt:  sensor:  - state\_topic: "iot/5/car"  name: car\_count  group:  default\_view:  name: "Parking Lot"  entities:  - sensor.car\_count |
| --- |

# 문제 및 해결과정

## 3.1 발생한 문제점

카메라를 사용해서 감지를 할때 모든 자동차가 바로 감지가 되지않는 경우가 발생한다. 다른 문제로는 초음파 센서를 주차 자리 바닥에 배치할 경우 센서가 감지한 물체가 차량인지 아닌지 알기 어렵다는 문제점이 있다.

## 3.2 문제 해결 방법 기술

Yolo가 주차장 내 모든 차량을 감지하지 못하는 경우를 방지하기 위해 Yolo가 10 프레임의 이미지를 통해 차량을 감지하게 함으로써 해결하였다. 10 프레임 중 몇몇 이미지에서 모든 차량을 감지하는 데 실패하더라도 이 프레임 중 최대 차량의 수를 사용함으로써 차량 감지 실패에 대한 위험을 최소화 하였다.

초음파 센서에 다른 물체가 감지되는 경우를 최소화하기 위해 평균 차량 높이를 이용해서 높이가 다른 물체를 감지 하지 않도록 방지했다. 하지만, 이는 낙엽 같은 환경요인에 인해 문제가 발생할 수 있고 Yolo를 활용한 차량 수 측정으로 충부하다 판단하여 현재와 같이 출입구에 초음파 센서를 설치하게 되었다.

# 향후 계획

## 4.1 관리 중 사고 감지

주차장 전체를 한눈에 보면서 차를 감지한 카메라를 이용해서 추가 기능을 구현할 수 있을 것으로 기대한다. 한 기사[5]에서 야외 주차장에 대한 관리의 부실을 비춘다. AI 감지 기능을 통해 파손을 감지할 수 있는데[6] 이를 이용해서 사고가 발생할때 빠르게 알려 조치를 취할 수 있도록 구현할 수 있다.

## 4.2 만성 주차난 해결방안

꾸준히 문제로 나타나는 만성 주차에 대한 최근 기사이다[7]. Google Maps API[8]를 사용해서 주변 교통상황을 가져올 수 있다. 이를 사용해서 사용자의 위치 및 목적과 주변 교통상황에 따른 주차공간을 추천하는 기능을 구현할 수 있다.

# 팀워크

## 5.1 실험 파트너와의 협업 정량적, 정성적 평가

프로젝트에서의 협업은 공정하게 이루어졌다. 코드 작성과 테스트 과정에서 파트너와의 의사소통이 원활했으며, 일정에 대한 책임감과 협력을 보여주었다. 김성빈은 주로 NodeMCU에 대한 코드 작성을 맡았고, 최준혁은 라즈베리 파이의 코드를 담당했다. 소스 코드 파일로 분배하면, 김성빈: [ESPCAM, UltraSonic, Hass], 최준혁: [cam\_mqtt\_countCar\_when\_signal.py]. 나머지 파일들: 보고서, 발표 자료, README의 경우 같이 공동 작업으로 공평하게 진행하였다.

# 결론

이 보고서를 통해 스마트 주차 시스템 프로젝트의 목표, 구현 방법, 문제 해결과정, 그리고 팀워크에 대한 내용을 상세하게 기술했다. 프로젝트를 통해 다양한 기술을 활용하고 문제를 해결하는 능력을 향상 시켰으며, 효율적인 협업과 소통 능력도 향상시킬 수 있었다.

# 참고문헌

[1] 초음파 방식 유도시스템, 선일일렉트콤, <http://www.ezled.co.kr/page/sub4_3_2_2>

[2] Camera Module Based on ESP32 with ESP32-CAM-MB adapter, GROBOTRONICS, <https://grobotronics.com/camera-module-based-on-esp32-with-esp32-cam-mb-adapter.html?sl=en>

[3] HCSR04 ultrasonic sensor, Pi Supply, <https://uk.pi-supply.com/products/ultrasonic-distance-sensor-hc-sr04>

[4] [현장in] 도심 나들이 명소 부산시민공원 만성주차난 해법은?, 연합뉴스, <https://www.yna.co.kr/view/AKR20230203057300051>

[5] 정부세종청사 주차장, "관리가 엉망", 세종의 소리, <http://www.sjsori.com/news/articleView.html?idxno=23792>

[6] Employ ML to Empower Your Business, ALTOROS, <https://www.altoros.com/solutions/car-damage-recognition>

[7] 시민공원 야외주차장 학교 서는데…만성 주차난 어찌할꼬, 국제신문, <http://www.kookje.co.kr/news2011/asp/newsbody.asp?code=0300&key=20230130.22010008243>

[8] Google Maps Platform, Google, <https://developers.google.com/maps>