

캡스톤 디자인 I 최종 발표

AI를 활용한 열화상 데이터 분석 및 IoT 기반 서버 쿨링 시스템

팀명 : 5인분 같은 4인분

팀원 : 홍수민(팀장), 길기훈, 오민석, 지원근

담당 교수 : 이상금 교수님



INDEX

목차

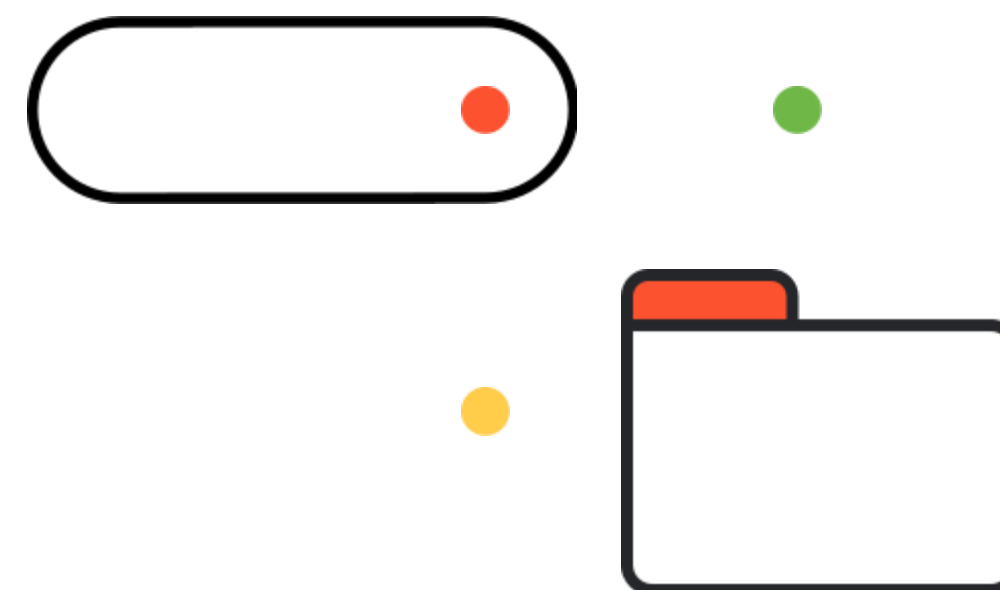
목차1	프로젝트 개요
목차2	진행 상황
목차3	향후 계획



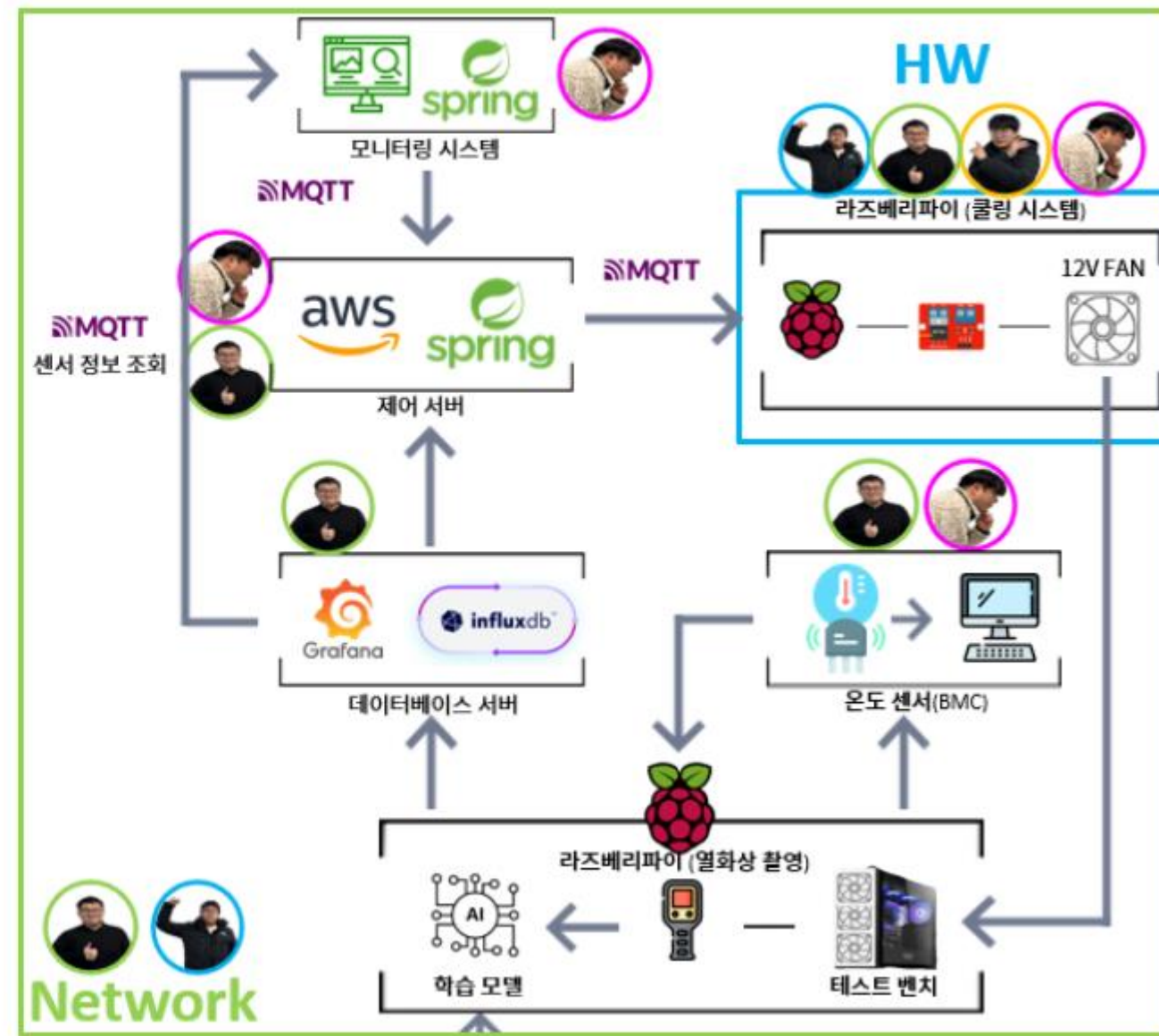
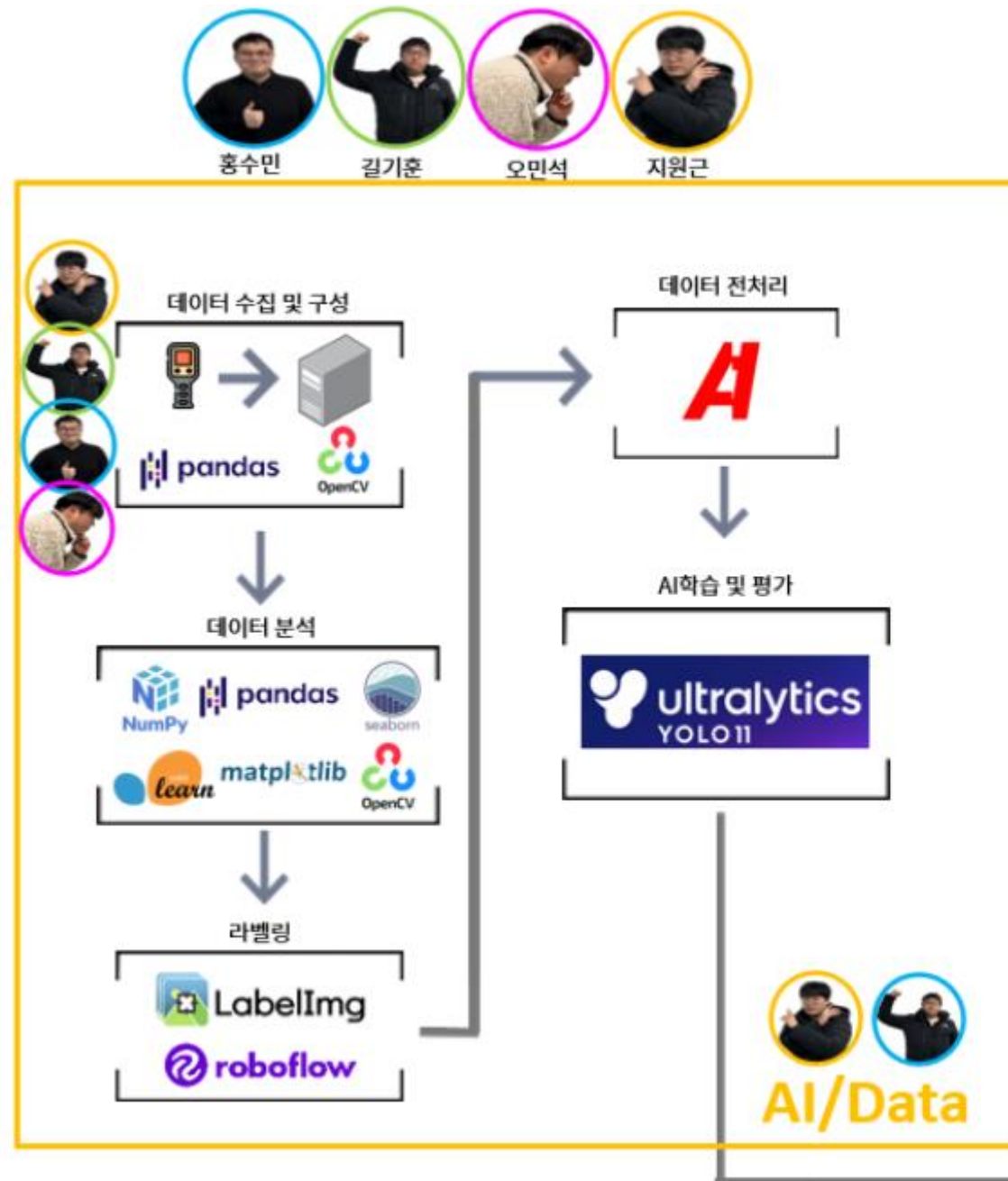


01

프로젝트 개요



전체 구성도



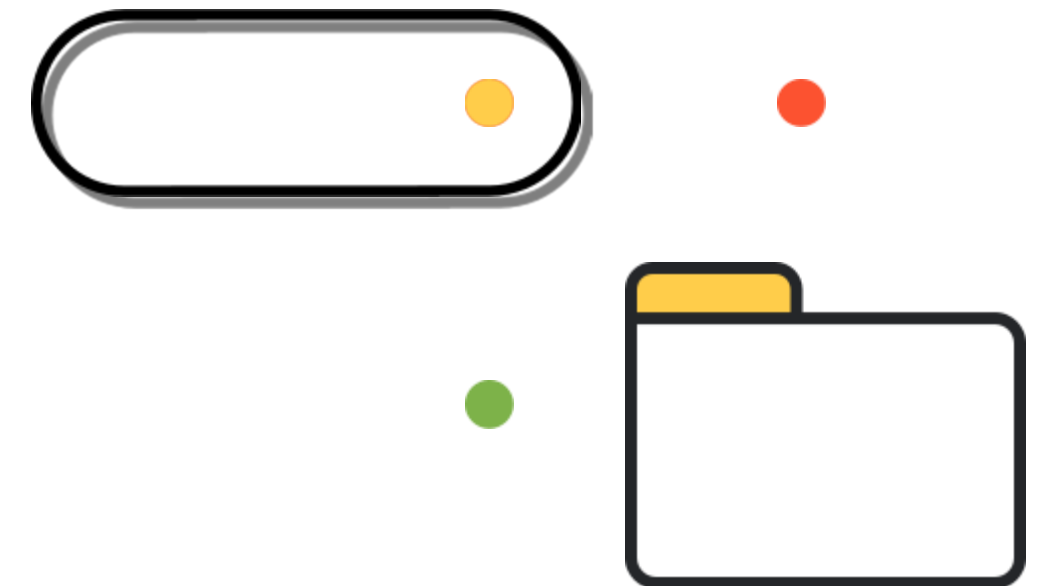
프로젝트 구성 및 목표

- 1 라즈베리파이와 열화상 카메라 연동
- 2 열화상 이미지 데이터 수집
- 3 데이터 분석 및 YOLO 모델 학습
- 4 DB 구축
- 5 서버 구축
- 6 팬 제어
- 7 모니터링 시스템 구현

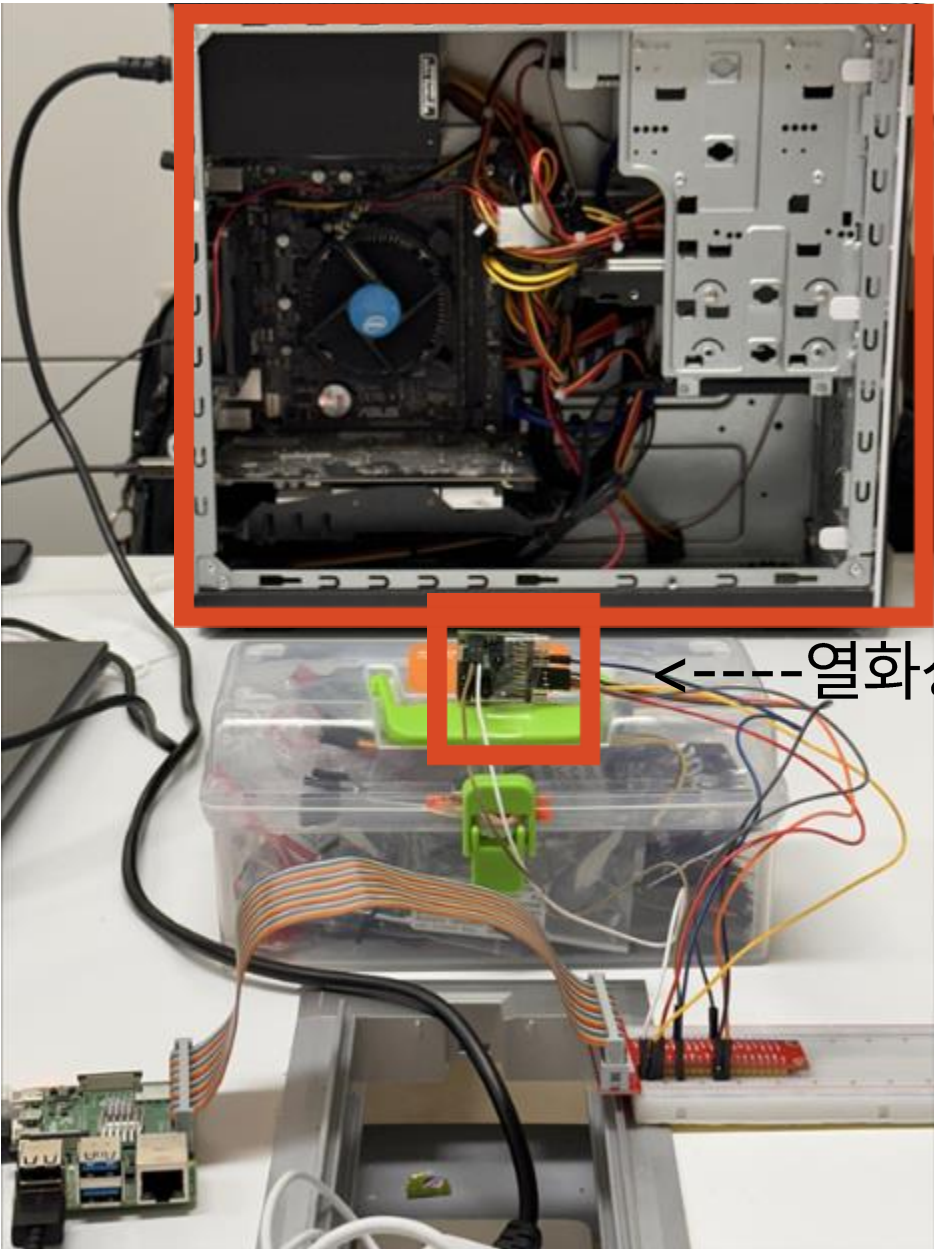


02

진행 상황



환경 설정



촬영범위

<----열화상 카메라

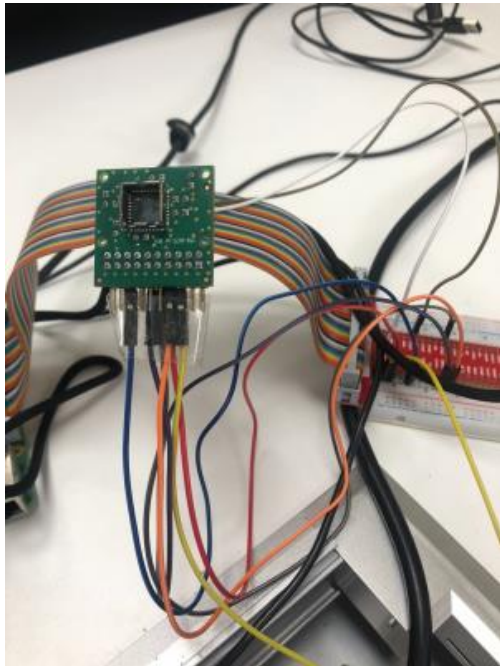
<촬영 환경>

환경 설정 시 고려 사항

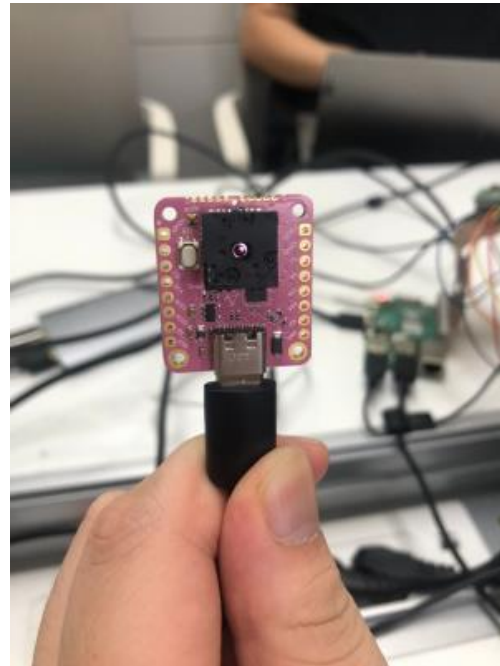
- 컴퓨터와 카메라간 거리 1.5m 이내로 설치[3]
- AGC 비활성화
- 최소 온도 10°C, 최대 온도 60°C로 고정

항목	사양
CPU	Intel Core i7-6700K
GPU	NVIDIA GTX 1050 2GB
RAM	16GB DDR4
OS	Ubuntu 22.04.5 LTS
카메라	FLIR Lepton 3.5

환경 설정 수정 사항



<수정 전>

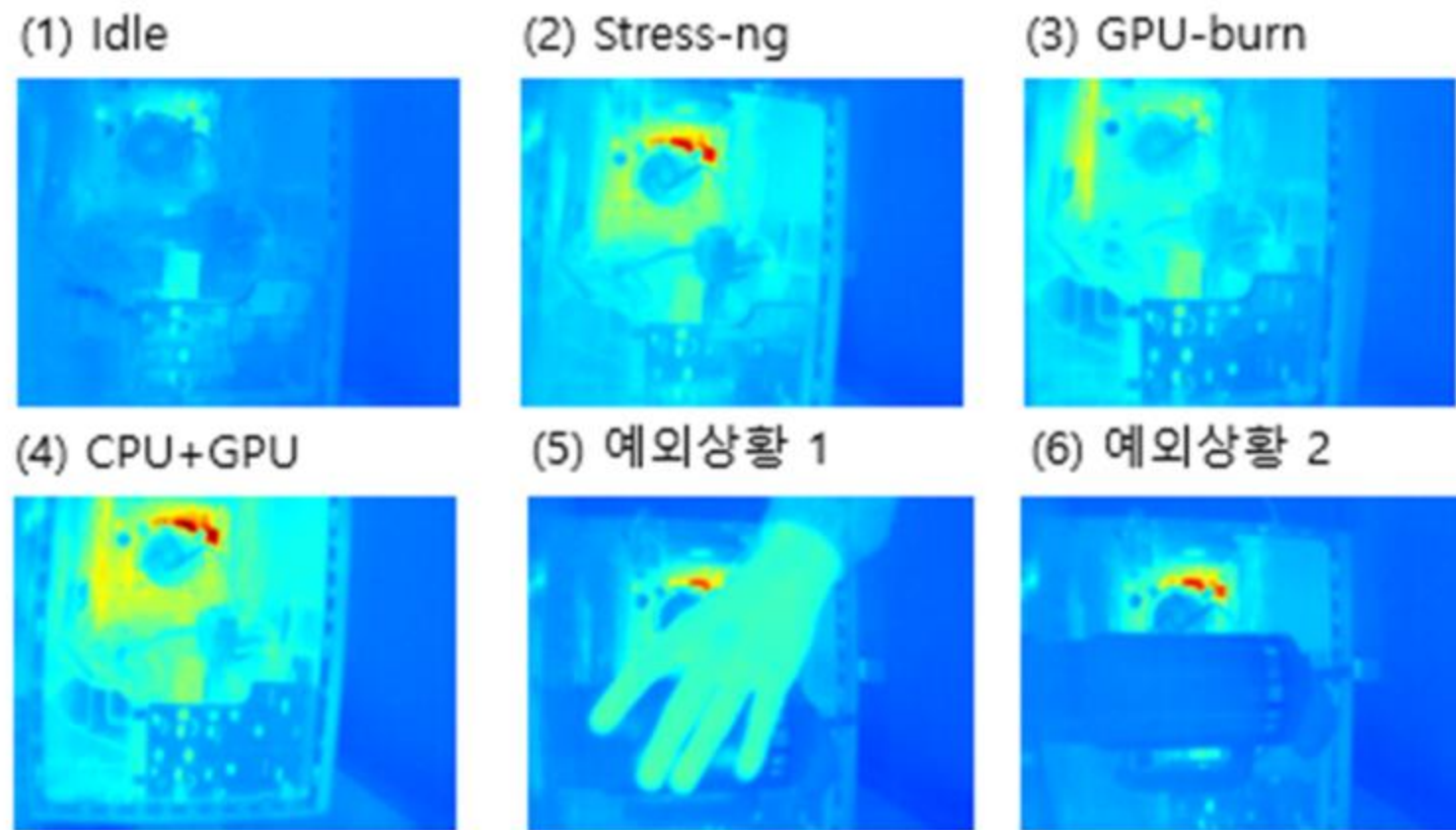


<수정 후>

수정된 내용

- FLIR Lepton 3.5 열화상 카메라 보드 연동 시
- 기존 : Breakout 보드 v2.0, GPIO 연결 방식
- 수정 : PureThermal 3, USB 연결 방식
- 기존 GPIO 연결 방식은 연결이 불안정하기 때문

데이터 수집



<시나리오별 촬영 이미지>

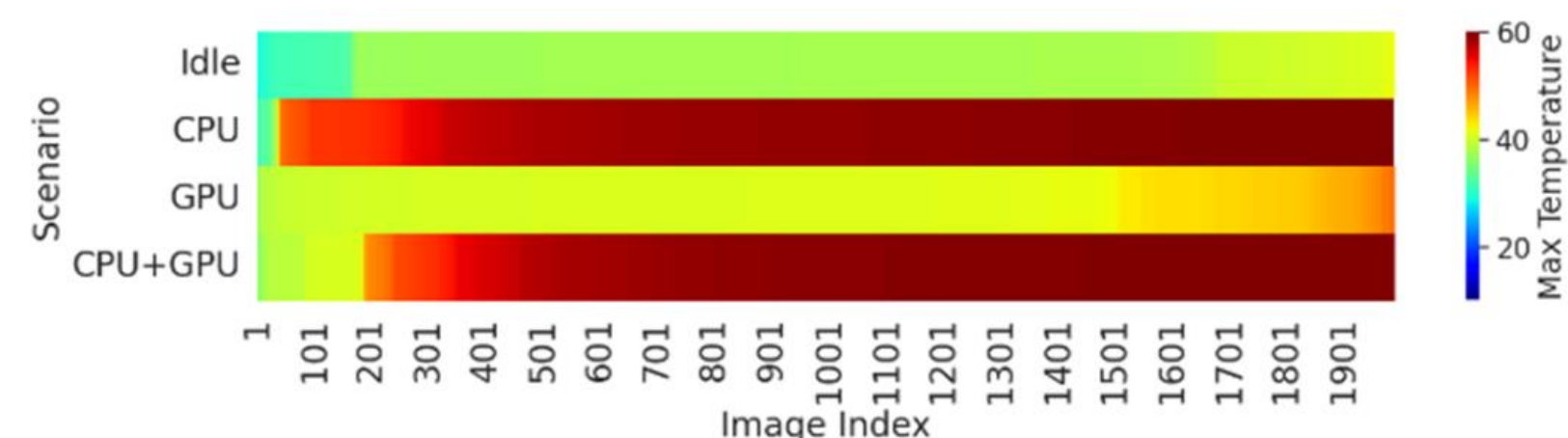
데이터 설명

1. Idle(유휴) 상태 : 1,990장
2. CPU 부하 상태 : 1,990장
3. GPU 부하 상태 : 1,990장
4. CPU + GPU 부하 상태 : 1,990장

총 7,960장

※5, 6(예외상황)은 1, 2, 3, 4에 포함

데이터 분석(히트맵)



<시나리오별 최대 온도값에 대한 히트맵>

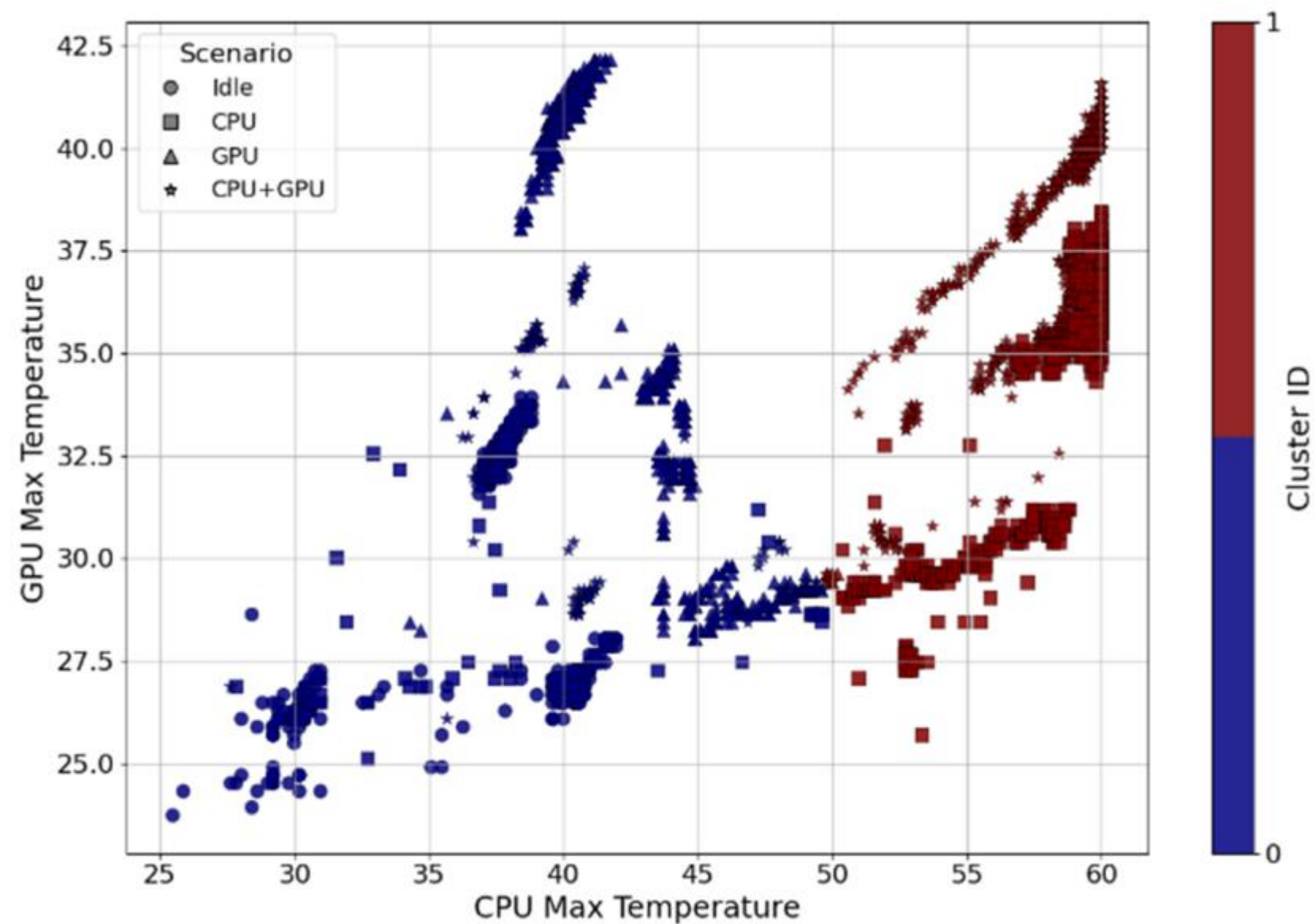
	Mean	Std	Min	Max
Idle 상태	37.0	2.0	28.0	42.0
CPU 부하	58.0	4.0	30.0	60.0
GPU 부하	42.0	2.0	34.0	50.0
CPU+GPU 부하	57.0	6.0	33.0	60.0

<시나리오별 최대 온도값에 대한 통계>

분석 요약

- Idle(유휴) 상태 : 연두색, 낮은 온도
- CPU 부하 상태 : 붉은색, 높은 온도
- GPU 부하 상태 : 노란색, 중간 온도
- CPU + GPU 부하 상태 : 붉은색, 노란색, 높은 온도

데이터 분석(K-Means클러스터링)



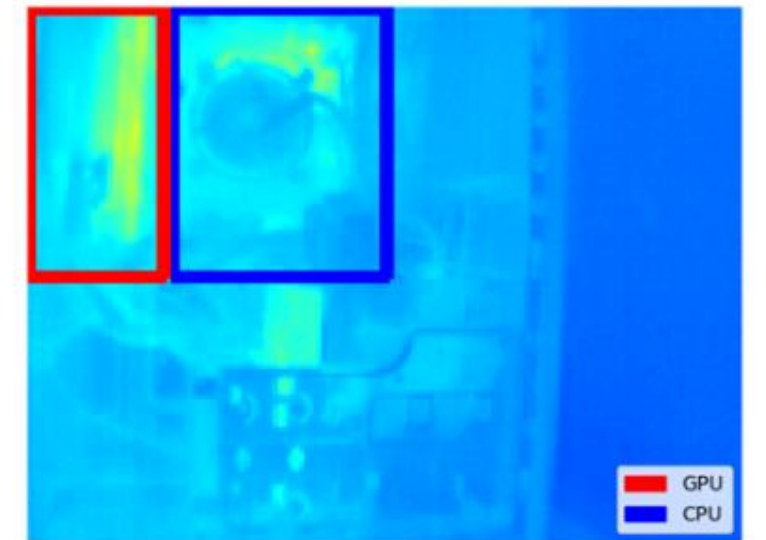
<CPU와 GPU 온도값 기반 K-Means 클러스터링>

분석 요약

Cluster	Scenario	Count
0	Idle 상태	1,990
0	CPU 부하	50
0	GPU 부하	1,984
0	CPU+GPU 부하	228
1	CPU 부하	1,940
1	GPU 부하	6
1	CPU+GPU 부하	1,762

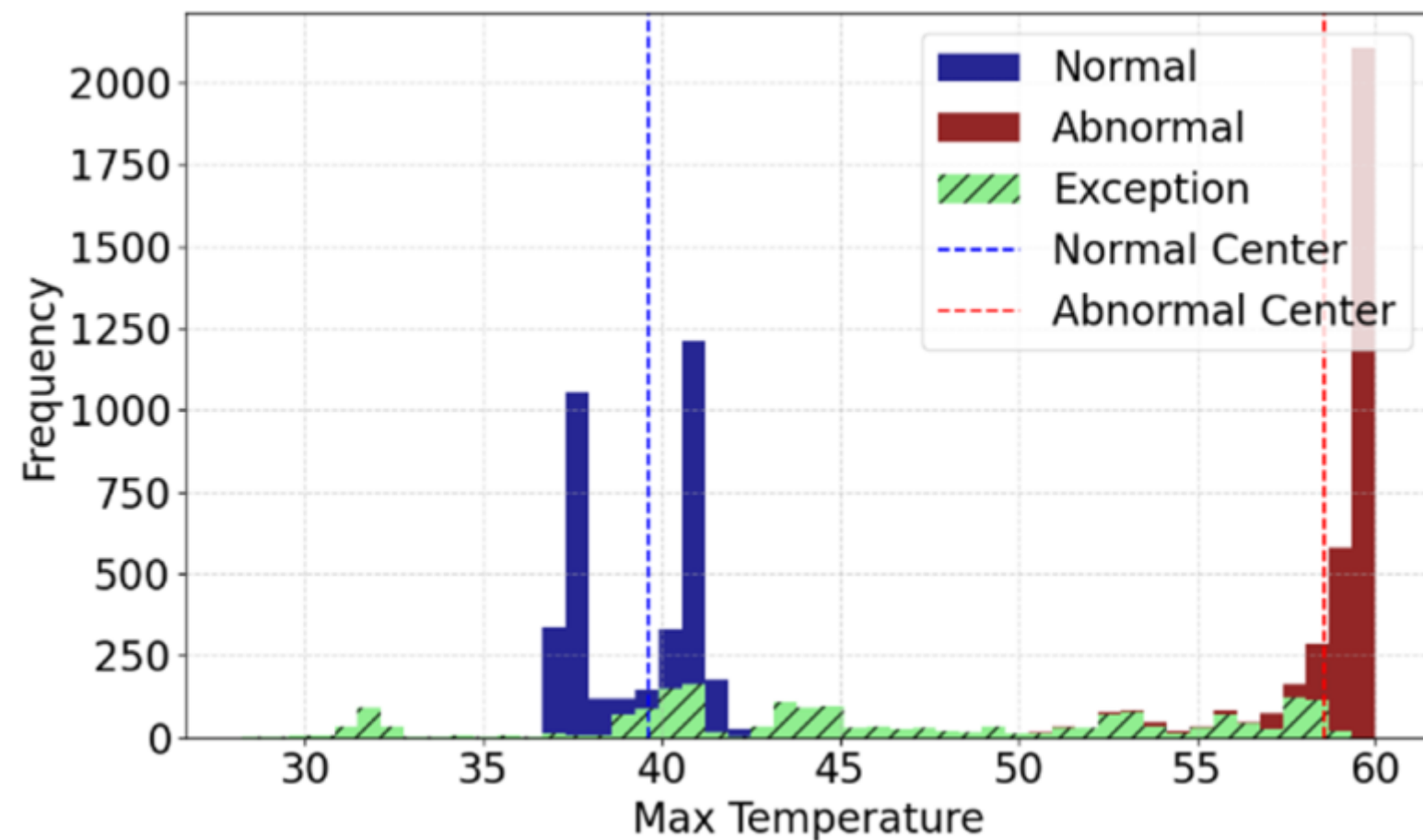
<클러스터링별 정량적 통계>

CPU 온도 50°C를 중심으로 클러스터 경계 형성되는데 이는 후에 AI 모델 학습을 위한 라벨링 기준으로 활용



<CPU와 GPU 영역>

데이터 분석(예외 상황)

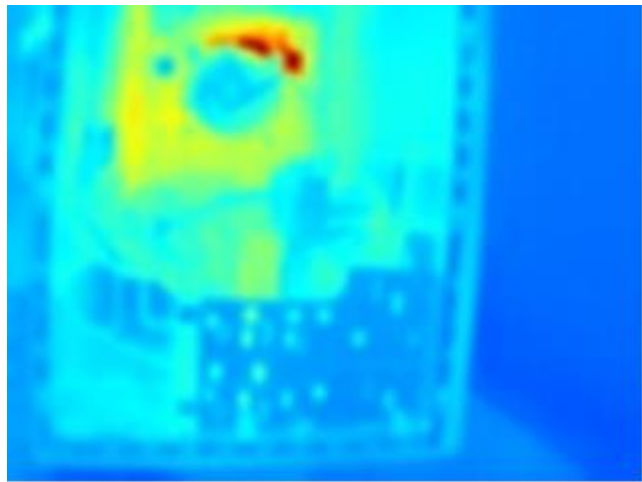


<컴퓨터 상태별 최대 온도값 분포(예외 상황 포함)>

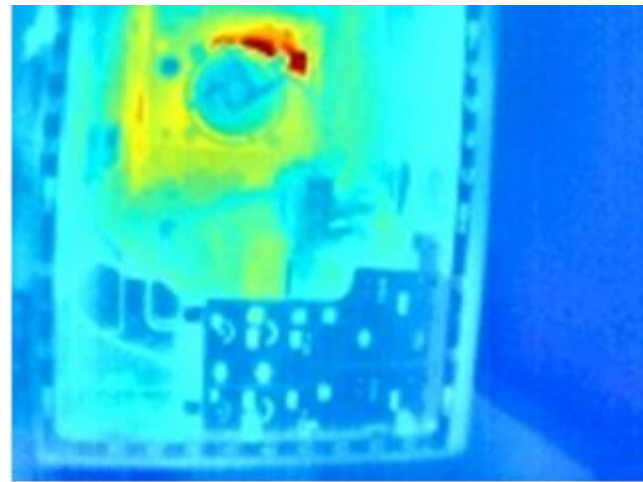
분석 요약

- Exception(예외 상황) : 30°C ~ 58°C에 분포
- 발열 부분의 가려짐 정도에 따라 열화상 이미지에서 추출한 온도 값이 달라짐
- 실제 상태와 클러스터링 결과 간 불일치
- 향후, BMC 센서 데이터와 결합하여 신뢰성 보완

데이터 전처리

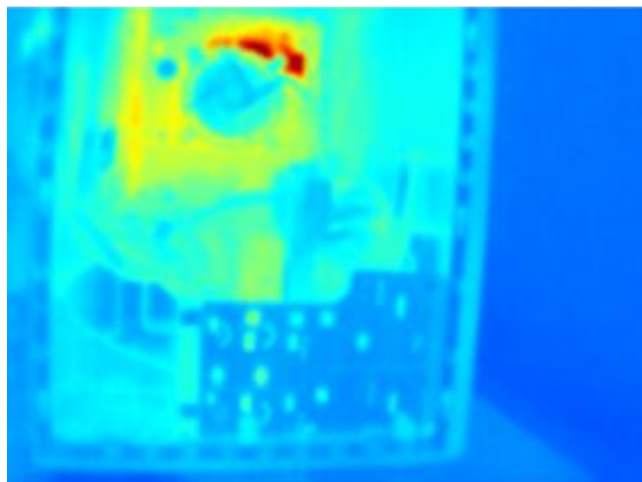


<Blur>

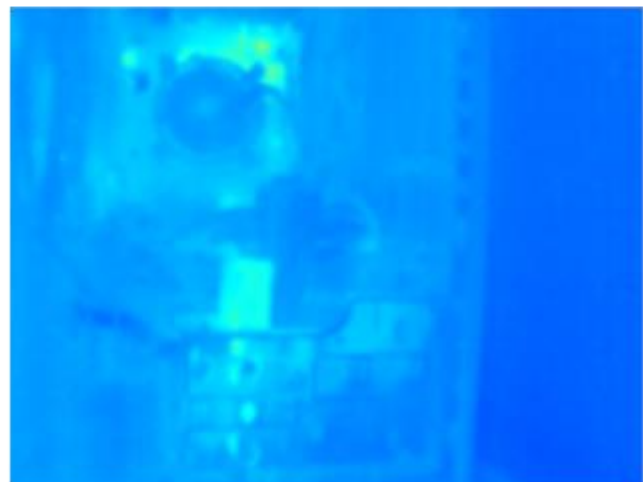


<CLAHE>

<전처리 적용한 이미지>



<비정상>



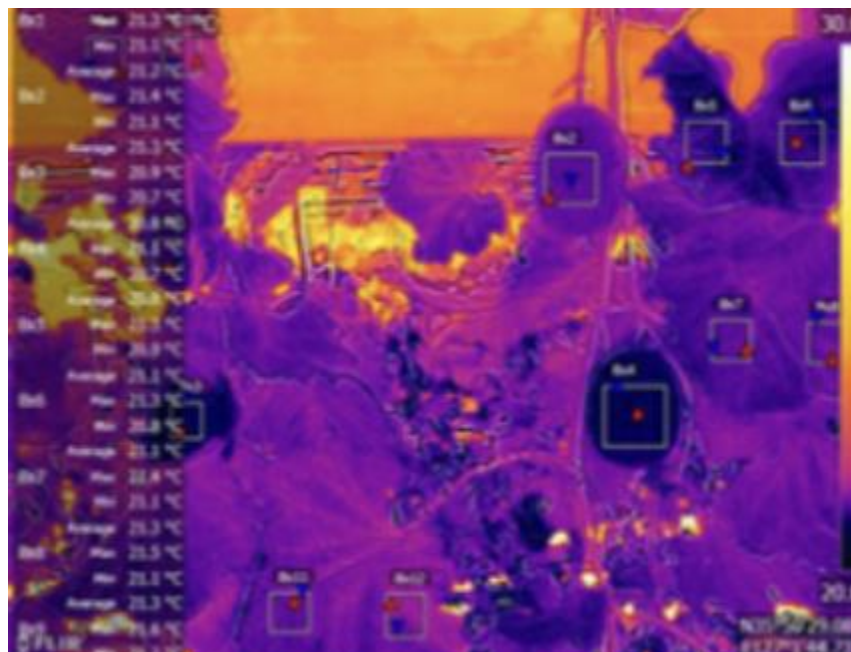
<정상>

<열화상 이미지 (좌 : 비정상, 우 : 정상)>

요약

- Train/Validation 8:2 비율 분할
- 최대 온도 50 °C 기준으로 라벨링(정상/비정상)
- Blur, CLAHE 적용

YOLO 채택



채택 이유

- 최근 심층신경망을 활용한 열화상 기반 이상 영역 분할 연구 진행[4]
 - 강화된 CNN 및 Transformer 기반 모델을 열화상 이상 탐지에 적용 [5][6]
 - 그러나 이러한 모델들은 이미지 분류나 픽셀 단위 세그멘테이션에 치중되어 객체탐지를 통해 이상 부위를 직관적으로 식별하기엔 한계
-
- 반면, YOLO는 산업 현장의 열화상 이미지에서 이상 발열 영역을 실시간으로 탐지하는데 효과적[7]
 - 열화상 카메라의 이점을 YOLO와 결합하여 자동으로 조도가 강한 낮 시간에 화재 탐지[1]
 - 육안으로 확인이 어려운 농작물의 상태를 분석하는 등 다양한 분야에서 융합적 접근 이루어짐[2]



모델 학습

YOLO11	mAP50:95	CPU 속도 (ms)	매개변수 (M)	FLOPs (B)
n	39.5	56.1	2.6	6.5
s	47.0	90.0	9.4	21.5
m	51.5	183.2	20.1	68.0
l	53.4	238.6	25.3	86.9
x	54.7	462.8	56.9	194.9

<YOLO11 모델별 세부 사항>

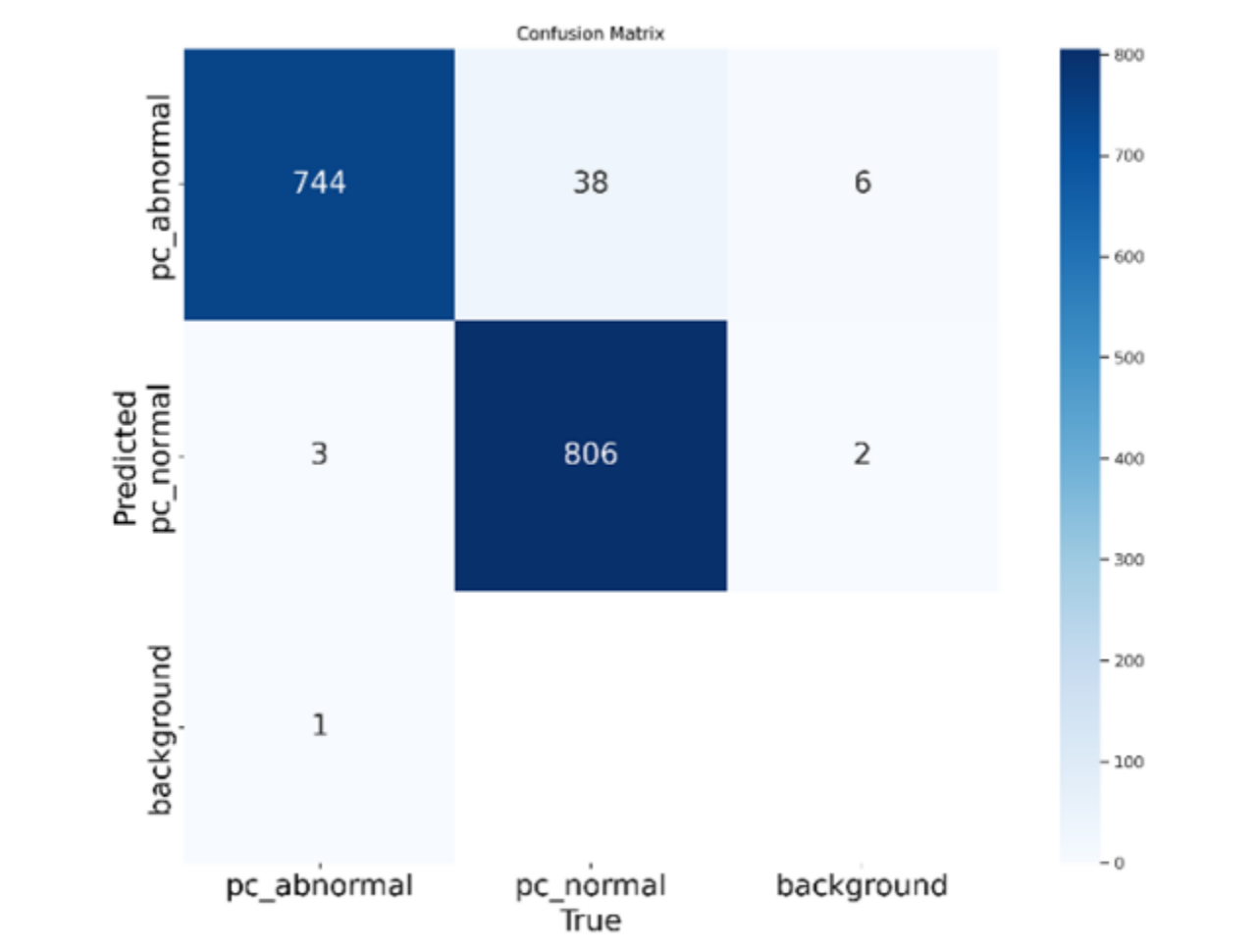


모델 선택 시 고려사항

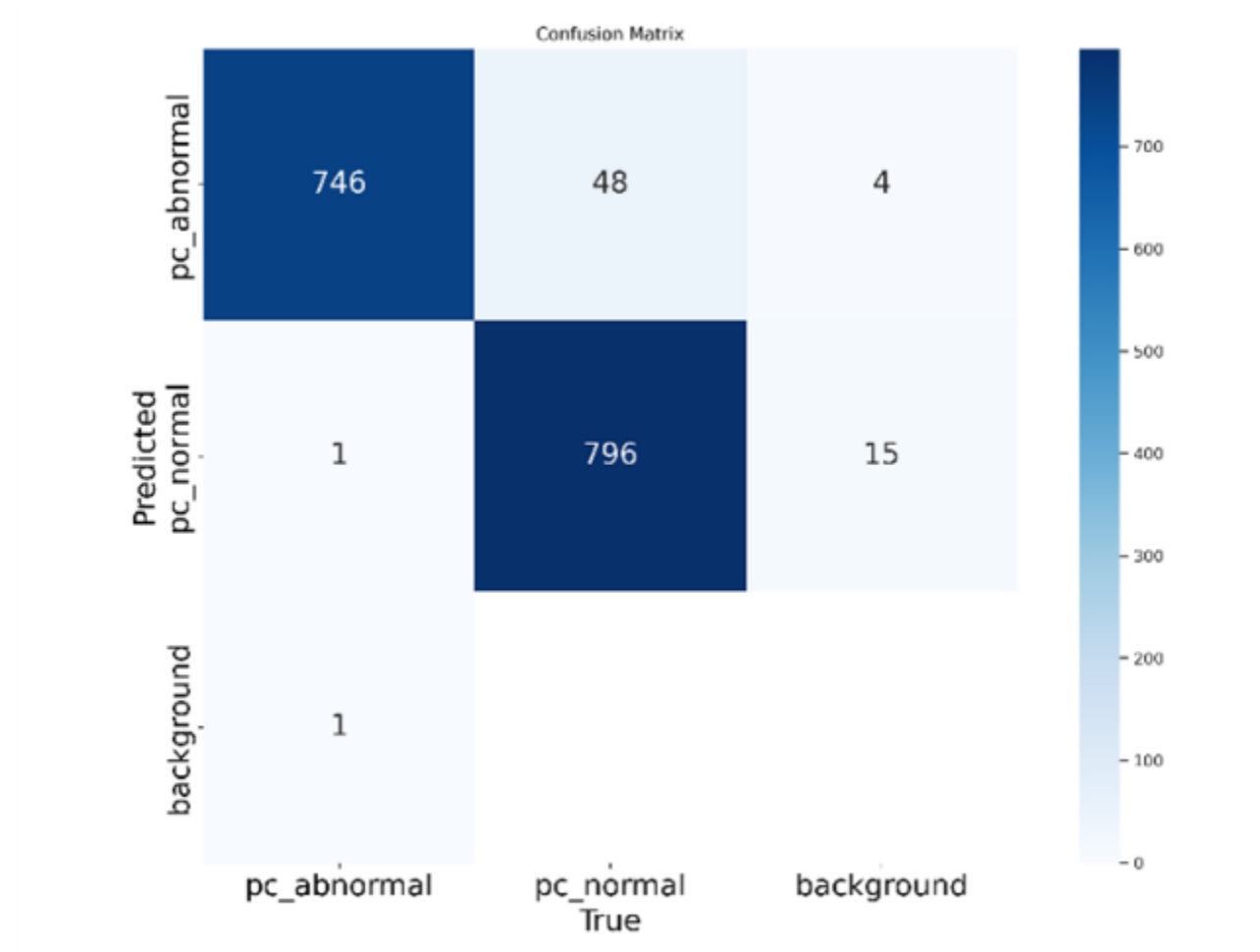
- 대형 모델(m, l, x)의 파라미터 수, 연산량 大[8]
- 소형 시스템(라즈베리파이)와 적합하지 않음
- YOLO11n(2.4M) vs YOLO11s(9.4M)

파라미터	값
Input Size	160 x 160
Epoch	50
Optimizer	AdamW(lr=0.001667, momentum=0.9)
Batch Size	32

모델 평가



<YOLO11n>

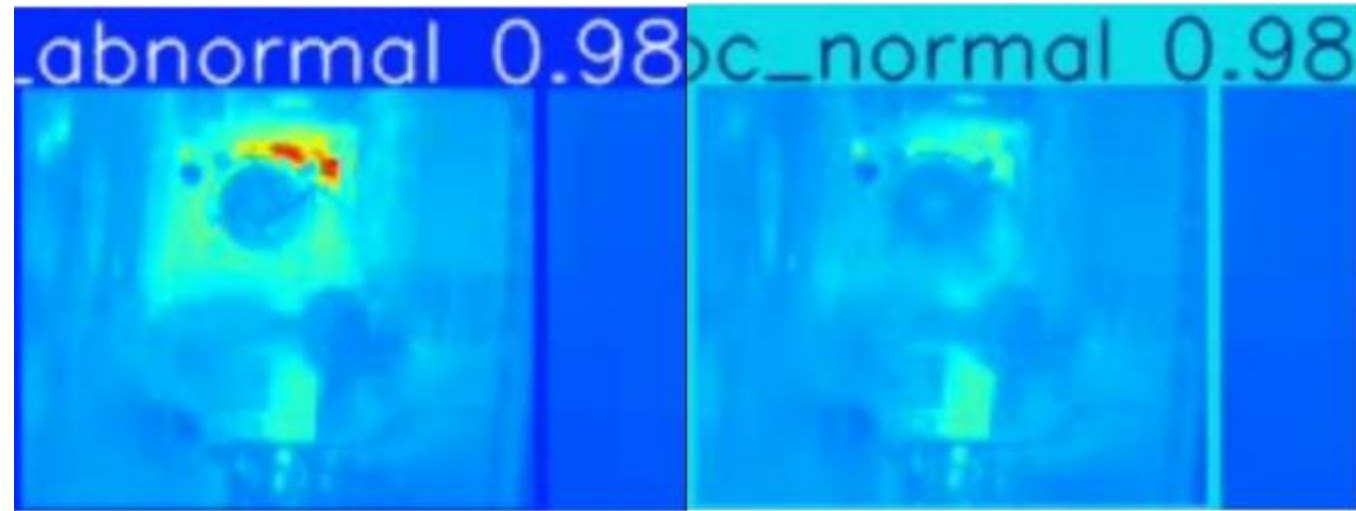


<YOLO11s>

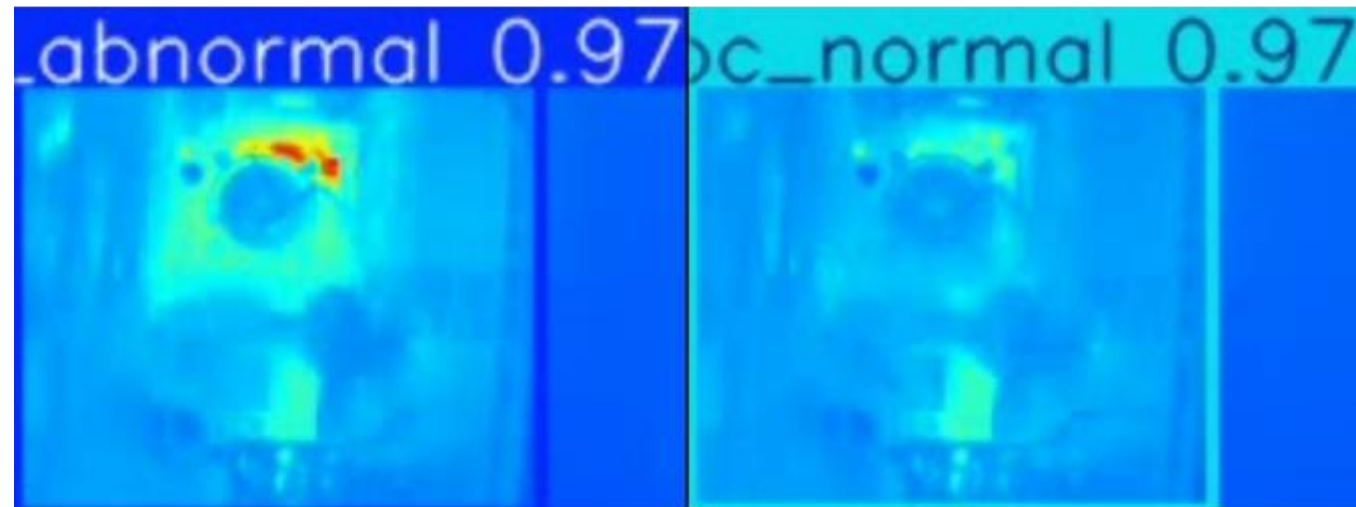
YOLO11	n	s
mAP50:95	0.99329	0.99044
Precision	1.00	1.00
Recall	0.99	1.00
F1	0.97	0.97

<YOLO11 모델별 학습 결과>

모델 평가



<YOLO11n>



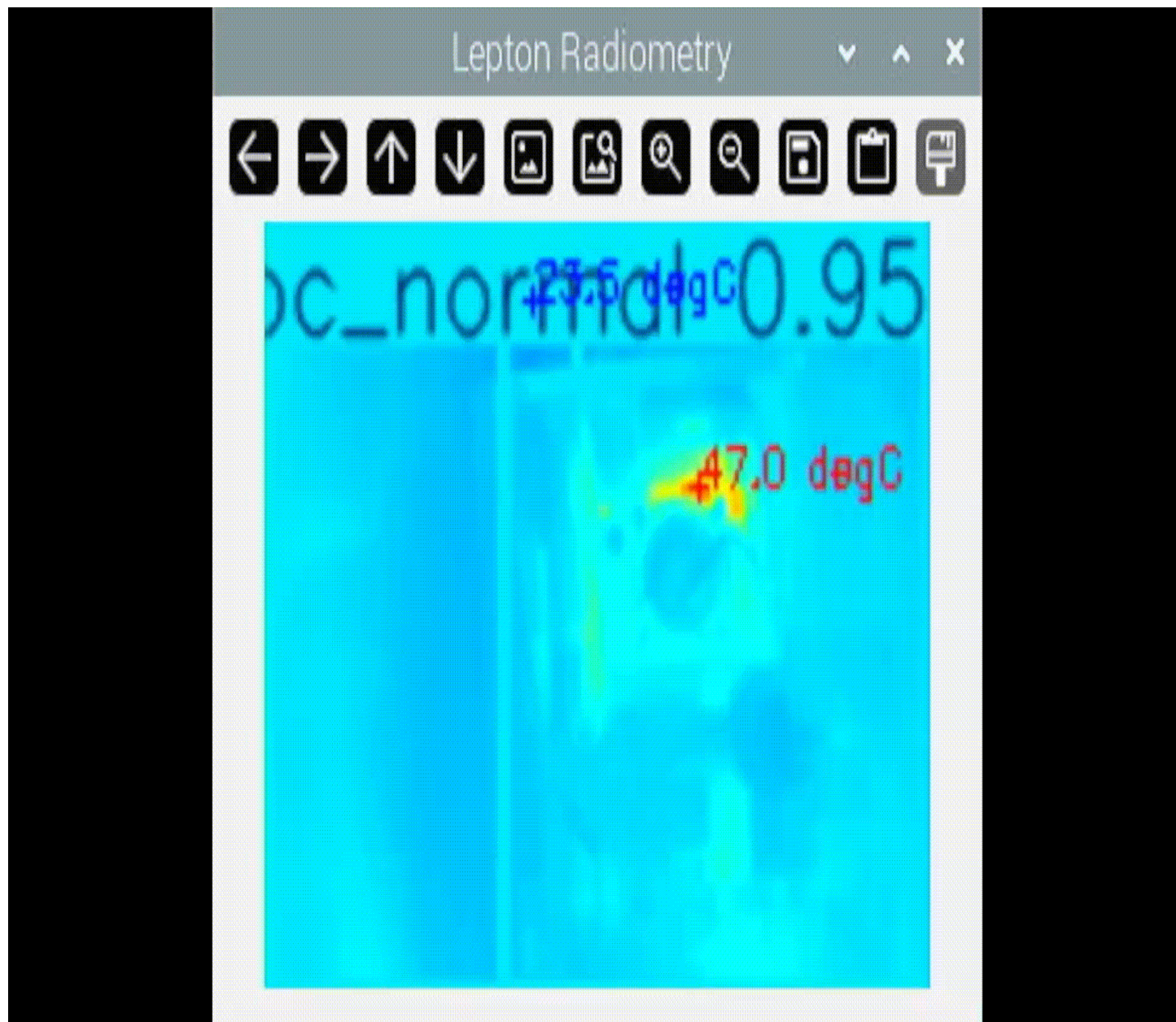
<YOLO11s>

<YOLO11 모델별 탐지 결과 비교>

모델 선택 시 고려사항

- YOLO11n과 YOLO11s는 97~98%의 높은 신뢰도
- 제한된 하드웨어 자원 내에서 실시간 처리 요구
- 상대적으로 경량화된 YOLO11n 모델 채택

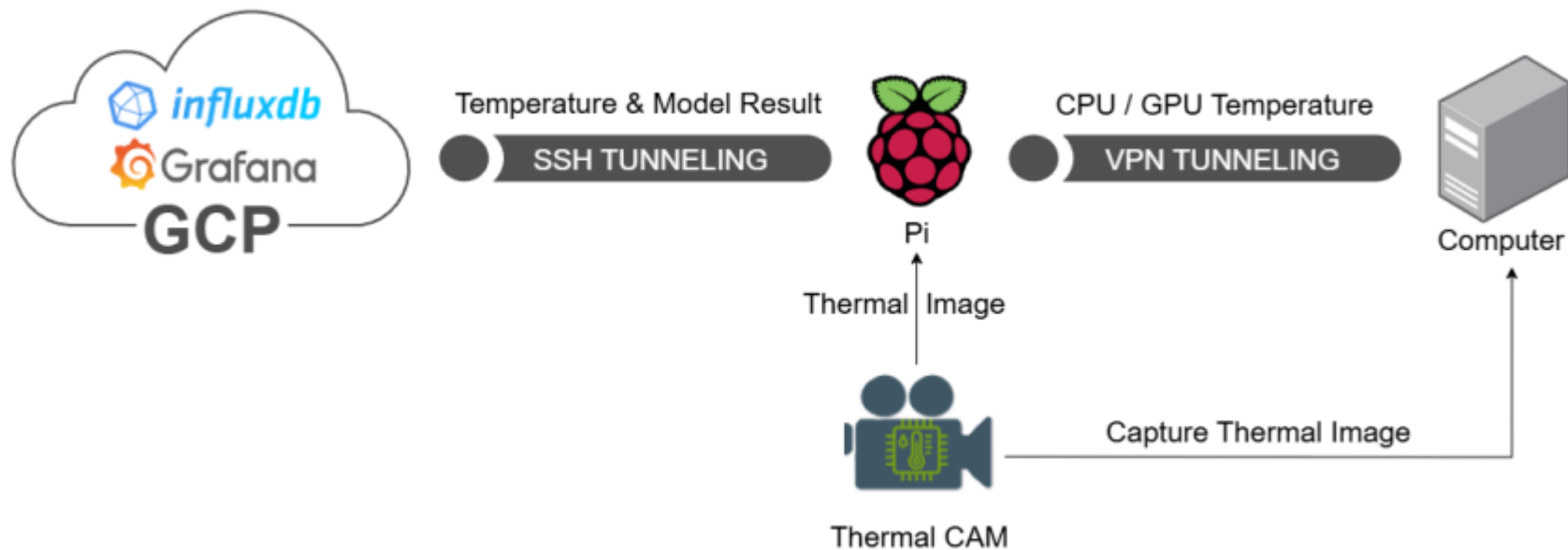
모델(YOLO11n) 시연 영상



모델 선택 시 고려사항

- YOLO11n과 YOLO11s는 97~98%의 높은 신뢰도
- 제한된 하드웨어 자원 내에서 실시간 처리 요구
- 상대적으로 경량화된 YOLO11n 모델 채택

통신



<통신 과정 구성도>

통신 과정

- 컴퓨터는 IPsec 기반 VPN을 통해 라즈베리파이와 연결되어 센서에서 측정된 CPU, GPU 온도를 라즈베리파이에 전송
- 라즈베리파이는 수신한 데이터(CPU, GPU 온도)를 YOLO가 판별한 결과와 통합하여 전송 (데이터의 일관성 & 정확성)
- 통합된 데이터를 SSH 터널링을 이용해 GCP에 위치한 InfluxDB로 전송

통신

```

pi@raspberrypi: ~
파일(F) 편집(E) 탭(T) 도움말(H)
: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc nq state DOWN group
default qlen 1000
  link/ether e4:5f:01:ad:41:47 brd ff:ff:ff:ff:ff:ff
: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP g
roup default qlen 1000
  link/ether e4:5f:01:ad:41:49 brd ff:ff:ff:ff:ff:ff
  inet 192.168.99.123/24 brd 192.168.99.255 scope global dynamic noprefixroute
wlan0
    valid_lft 3282sec preferred_lft 3282sec
  inet 192.168.43.4/32 scope global wlan0
    valid_lft forever preferred_lft forever
  inet6 2001:e60:a020:1e2b:ea9b:f7ba:76be:20ee/64 scope global dynamic noprefi
route
    valid_lft 6887sec preferred_lft 6887sec
  inet6 fe80::7eb9:4695:deba:d2a2/64 scope link noprefixroute
    valid_lft forever preferred_lft forever
: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast sta
e UNKNOWN group default qlen 500
  link/none
  inet 192.168.43.4/32 scope global noprefixroute tun0
    valid_lft forever preferred_lft forever
  inet6 fe80::9618:bf82:7362:6b97/64 scope link stable-privacy
    valid_lft forever preferred_lft forever
pi@raspberrypi:~ $

```

<VPN 클라이언트 IP 조회>

VPN(가상 사설망)

- IPsec 기반 VPN을 구축하여 허용되지 않은 시스템의 접속 차단
- 각 클라이언트에 고정 아이피를 할당하여 시스템 재부팅 시 IP변경으로 인한 접속 문제 해결
- VPN을 사용하여 학교 인터넷과 외부 시스템 간의 접속 문제 완화

통신

```

pi@raspberrypi: ~
파일(F) 편집(E) 탭(T) 도움말(H)
[서버:INFLUX] 전송 완료 → CPU=27.8 | GPU=23.0 | MODEL=pc_normal | HTTP 204
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 2 으로 수신받음
Client 2 lock || model = pc_normal
gogogogogog
cpu_temperature value=27.8
gpu_temperature value=23.0
model_result value="pc_normal"
cpu_temperature value=27.8
gpu_temperature value=23.0
model_result value="pc_normal"
[서버:INFLUX] 전송 완료 → CPU=27.8 | GPU=23.0 | MODEL=pc_normal | HTTP 204
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
Client 1 lock || cpu = 27.8 gpu = 23.0
]

pi@raspberrypi: ~/purethermal1-uv-capture/python
파일(F) 편집(E) 탭(T) 도움말(H)
(1, 3, 128, 160)
Detected class: pc_normal
0: 128x160 1 pc_normal, 112.0ms
Speed: 0.8ms preprocess, 112.0ms inference, 2.3ms postprocess per image at shape
(1, 3, 128, 160)
Detected class: pc_normal
0: 128x160 1 pc_normal, 110.8ms
Speed: 0.8ms preprocess, 110.8ms inference, 2.8ms postprocess per image at shape
(1, 3, 128, 160)
Detected class: pc_normal
0: 128x160 1 pc_normal, 116.6ms
Speed: 0.8ms preprocess, 116.6ms inference, 2.2ms postprocess per image at shape
(1, 3, 128, 160)
Detected class: pc_normal
0: 128x160 1 pc_normal, 120.5ms
Speed: 0.9ms preprocess, 120.5ms inference, 2.2ms postprocess per image at shape
(1, 3, 128, 160)
Detected class: pc_normal

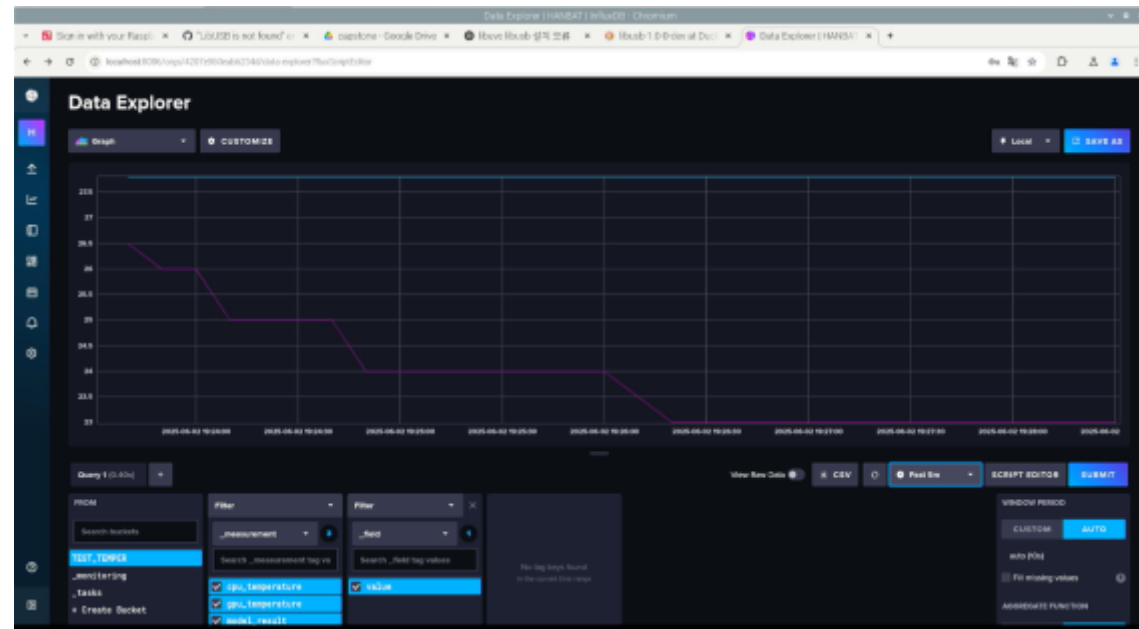
```

< Local Port forwarding으로 DB로 데이터 전송 (8086 포트)>

SSH 터널링

- Local PC 와 Raspberry Pi의 연결
- InfluxDB의 8086포트를 이용하여 SSH Local Port forwarding 이용
- Socket 통신을 이용해 통합된 데이터를 GCP의 8086포트로 전송 (1초의 전송 주기)

데이터베이스



<InfluxDB 구조>



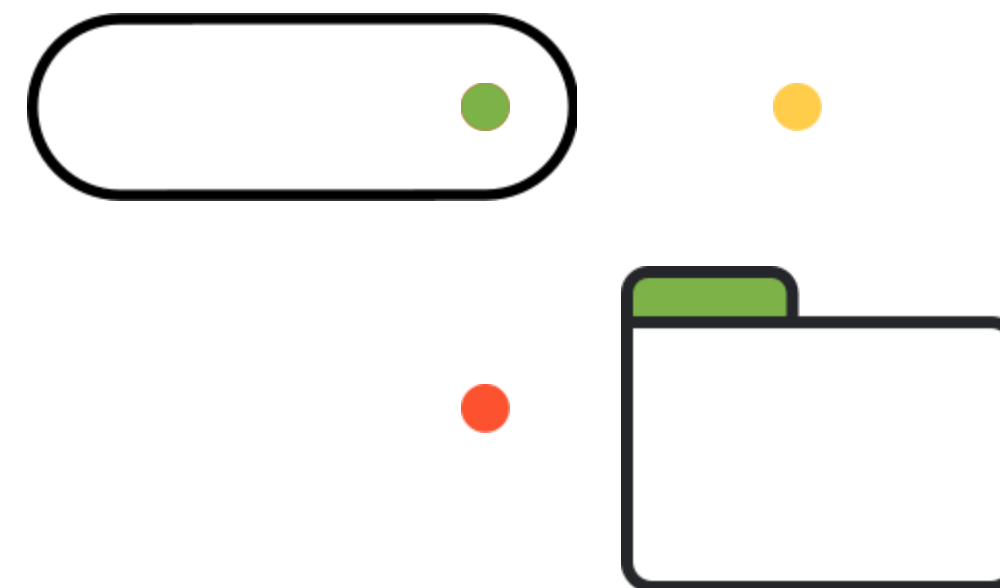
<판별 값 (0/1)과 CPU, GPU 온도 표시 (Grafana)>

Influx DB & Grafana

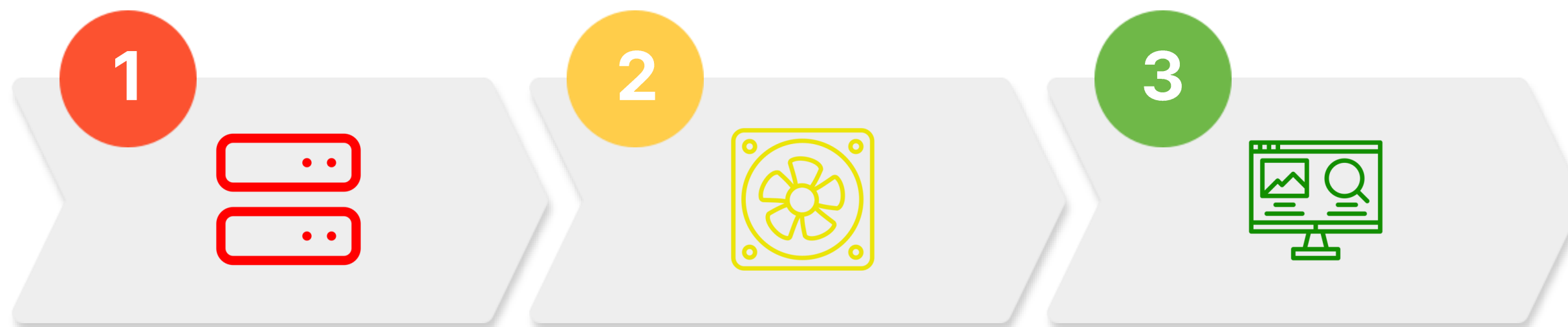
- 관측되는 PC의 CPU, GPU 온도와 판별값을 0과 1로 수신 (0 : 정상, 1 : 비정상)
- Grafana로 연결 -> 임계 값 초과 시 Alert
- CPU 사용량, GPU 사용량 등 항목 추가 예정

03

향후 계획



향후 계획



1단계

서버 구축

2단계

팬 제어

3단계

모니터링 시스템 구축

참고 문헌

- [1] 고아라, 조정원. (2024). 전통시장을 위한 열화상 기반 실시간 화재 감지 시스템 개발. 멀티미디어학회논문지, 27(11), 1398-1405.
10.9717/kmms.2024.27.11.1398.
- [2] 이성민, 김경철, 백자영, 양창주, 김만중, 조병효. (2024). 수경재배 참외 인식을 위한 열화상 및 딥러닝의 적용 가능성 검토. 드라이브·컨트롤, 21(4), 37-45.
- [3] 이지훈, 정임영. (2022-12-20). 열화상 카메라를 통한 온도 측정 데이터 신뢰도 분석. 한국정보과학회 학술발표논문집, 제주.
- [4] J. Kang, S. Park, "Infrared Thermography for Fault Diagnosis in Electronic Equipment," Journal of Electronic Testing, vol.35, no.3, pp.271-283, 2019.
- [5] Y. Wang, X. Liu, et al., "Transformer-Based Abnormal Heat Identification in Thermal Images of Power Transformers," Scientific Reports, vol.14, Art no.81286, 2024.
- [6] C. Bao, J. Cao, et al., "Improved Dense Nested Attention Network Based on Transformer for Infrared Small Target Detection," arXiv:2311.08747, 2023.
- [7] L. Li, H. Xu, et al., "Real-Time Thermal Anomaly Detection in Industrial Scenes Using YOLO," Sensors, vol.20, no.10, 2020.
- [8] Jocher, G., & Qiu, J. (2024). Ultralytics YOLO11 (Version 11.0.0) [Software]. GitHub. <https://github.com/ultralytics/ultralytics>