1페이지

안녕하세요. 3조 antifragile.
COMPUTER VISION 기술을 이용한 유해 야생돌물 탐지, 퇴치 및 경로추적 솔루션 발표를 시작하겠습니다.
저는 발표를 맡은 박종혁입니다.

2페이지

발표 순서는 프로젝트 개요부터, 사용한 데이터와 모델에 대한 소개, 서비스되는 웹페이지 소개, 서비스 시연, 그리고 프로젝트 결론과 함께
프로젝트를 하면서 진행했던 연구와 고찰, 그리고 참고했던 문헌 순서로 발표를 하겠습니다.

3페이지

저희 팀은 조장 박종혁, 기획과 프론트엔드를 맡은 윤율, 모델부 탐지 및 경로추적 기술 구현을 맡은 심재훈, 모델의 학습을 맡은 손원준 총 네 명으로 이루어져 있습니다.

저희 팀 명 안티 프래질,
작은 위기를 통해 배운 경험으로 더욱 단단해져, 큰 위기가 왔을 때 견딜 수 있는 태도,

즉 다시 말해, 리스크를 다루는 팀원들 각자가 견지할 태도를 의미합니다.

4페이지

저희 프로젝트에서 사용한 자원은 보시는 바와 같습니다.

학습 과정에서 제공받은 노트북과 함께, AWS에선 RDS, EC2, S3 를 이용했습니다.

모델 학습은 코랩을 통해 진행되었구요, 코랩 프로 3개, 코랩 프로 플러스 1개였습니다.

5페이지

프로젝트 개요 먼저 소개해드리겠습니다.

6페이지

저희 프로젝트는 '유해동물로 인해 겪는 문제점'에서부터 시작했습니다.

올해 1월에 아프리카 돼지열병이 발병하면서 1만여두의 가축이 도살되었습니다.

아프리카 돼지열병은 백신이 없는 치사율 100%의 바이러스로서 발병 객체가 발견되면 반경 500M 의 가축은 전부 도살처분됩니다.

더불어, 고라니와 멧돼지 등은 농가의 농작물 피해를 입혀 유해동물로서 관리가 필요합니다.

이에 저희는 유해동물을 탐지하고 퇴치하며 관련 이벤트 결과를 사용자에게 보여주는

7페이지
FARM-GUARDIAN 서비스를 기획하였습니다.

8페이지
저희 서비스의 기능에 대해 소개해드리겠습니다.
유해동물이 탐지 영역 내 진입을 하게 됩니다.

9페이지

탐지장치를 통해 객체가 탐지되면 모델 백엔드에서 객체를 인식하고 분류합니다.
그리고 여기에 더해, 객체의 이동 경로도 추적하여 표시합니다.

10페이지

탐지된 객체가 위험범위 내 진입하게 되면,

11페이지

퇴치부가 작동합니다. 현재 퇴치부는 호랑이 울음소리 사운드로 구현되어있습니다.

12페이지

탐지된 객체가 탐지 영역에서 사라지면 그동안 발생한 이벤트 내용이 로그값, 영상, 동영상 자료로 AWS 로 전송됩니다.

13페이지

전송된 자료는 구축된 웹 사이트를 통해 서비스 사용자, 예를 들면 농장주 들에게 제공됩니다.

14페이지
저희는 이러한 서비스를 4단계의 버전으로 기획하였습니다.

버전 1에서는 객체를 탐지하고, 이를 웹 서버로 보내는 기능 구현을

버전 2에서는 탐지된 객체의 경로를 추적하고 , 위험 구역에 객체가 진입시 경보를 재생하는 기능을

버전 3에서는 서비스의 주, 야간 모드를 위해 이에 맞는 가중치 파일을 제작하였습니다.

이후에는 객체 거리 탐지, 미니맵 구현, 구동 최적화등이 계획되어 있습니다.

매일 오전 오후에 각각 회의를 하였고 이러한 기록은 노션으로 수행했습니다.

15페이지
이렇게 기획한 서비스의 대상자, 즉 고객은

B2C 부문에서는 유해동물 피해가 우려되는 농가,

B2B 부문에서는 스마트팜 서비스를 제공하는 개발사의 기존제품과 결합 서비스를 염두하였고,

그리고 B2G로는

동물의 질병을 담당하는 농림축산검역본부,

로드킬 방지 대책이 필요한 도로교통공사,

야생동물 추적이 필요한 국립공원공단을

서비스 사용 대상자로 삼았습니다.

16페이지
물론 이러한 유해동물 퇴치 서비스가 저희가 처음은 아닙니다. 이전에 비슷한 시도를 한 여러 업체들이 있습니다.

초기에는 움직임 감지 센서를 기반으로 한 멧돼지 퇴치기 등이 있었습니다.

그리고 최근에는 LG CNS에서 AI를 활용한 객체 검출과 센서를 통한 디지털 허수아비라는 서비스를 개발하였구요,

여기에 저희는 사용자 중심의 서비스를 더하였습니다.

발생한 이벤트의 내역을 사용자에게 보여줌으로써,
서비스에 대한 신뢰를 향상하고

관리자는 퇴치 성공률 등을 모니터링 하면서 성능을 업그레이드 할 수 있는 장점이 있습니다.

<<<>>>
이러한 정밀한 제어는
센서기반 제품들보다 농가에 스마트한 도움을 줄 수 있습니다.

17페이지
이어서 사용한 데이터셋에 대해 소개해드리겠습니다.

18페이지
모델 훈련을 위해 총 세 가지의 데이터셋을 준비했습니다.

첫 번째는 프로젝트의 핵심이 되는 야생동물 활동 영상 데이터

두 번째는 다양한 객체들이 포함된 CoCo 데이터

세 번째는 KAIST의 야간 보행자 데이터셋입니다.

19페이지
가장 핵심적인 데이터셋은 야생동물 활동 영상 데이터로, 총 11종의 주야간 동물 영상 데이터로 구성되어있는데요,

20페이지
이번 프로젝트에서는 서비스 대상을 농가로 좁혀 고라니, 멧돼지, 너구리,
반달가슴곰 4종의 데이터를 사용했습니다.

21페이지
이렇게 야생동물 데이터셋을 이용해 모델을 훈련시켰는데요..

화면에 보시다시피, 야생동물만을 학습했을 경우 사람의 머리 부위 등 어둡고
각지지 않은 물체들을 동물로 잡는 경우가 있었습니다.

이는, 모델이 알고있는 객체가 야생동물밖에 없기 때문에, 야생동물과 비슷하게
보이는 객체들을 야생동물로 분류했기 때문입니다.

그래서 저희는 이 문제의 해결을 위해 서비스 대상 지역에 등장할 수 있는 객체들을 모델에 학습시키기기로 했습니다.

22페이지

사람 등 다양한 객체가 포함된 CoCo 데이터셋과 야간 보행자(사람) 데이터셋 이렇게 두 가지를 사용했고, 결과적으로는 오분류율이 줄어들었습니다.

오분류율 감소에 대한 자세한 내용은 뒤의 성능평가에서 더 말씀드리겠습니다.

그런데, CoCo 데이터셋은 주간용, KAIST 데이터셋은 야간용이라고 되어있죠?

맞습니다. 저희는 주간, 야간을 나누어
학습을 진행했습니다.

23페이지

앞에 보이는 두 멧돼지 사진들, 보이는 특징이 매우 다릅니다.

24페이지

낮에는 물체와 장면을 비추는 충분한 자연광이 있어 색상 균형과 대비가 좋은 고품질 이미지가 생성됩니다. 반대로 야간 이미지는 일반적으로 인공 조명이나 낮은 수준의 주변 조명에서 캡처되므로 이미지가 더 어둡고 노이즈가 많습니다.

주간과 야간 훈련을 나눔으로써 보다 정확하게 예측할 수 있기 때문에, 주간과 야간을 나누어 학습을 진행했고,

실제 모델에도 주간과 야간 모드를 나눠 적용시킬 예정입니다.

25페이지

야생동물과 함께 학습 시킬 CoCo 데이터셋을 활용 때,
사람을 제외한 모든 객체들을 'others' 클래스로 분류하였습니다.

이렇게 분류한 이유는,

모델이 탐지 할 수 있는 객체가 너무 많아지면 모델이 복잡해지고, 학습하기 어려워져서 우리의 주요 탐지 대상인 야생동물의 탐지가 힘들어지기 때문입니다.

26페이지

데이터에 대한 문제 중, 데이터 불균형의 문제도 있었는데요,
각 클래스 간 데이터 수의 불균형이 있었고, 이는 언더샘플링 처리하였습니다.
언더샘플링 후, 클래스당 15,000장 ~ 18,000장씩의 데이터가 남게 되었습니다.

27페이지

그 다음으로는, 이미지 리사이징이 있습니다.

YOLO에 최적화된 학습 이미지 사이즈는 416 * 416 사이즈였고, 이에 맞춰 이미지들을 리사이징 해줬습니다.

이에 맞춰 바운딩 박스 범위값을 절대 좌표 값으로 수정해 이미지의 사이즈가 달라져도 바운딩 박스 범위가 엇나가지 않도록 전처리를 했습니다.

28페이지

모델의 성능 평가는 AP 점수를 사용했습니다.

컴퓨터 비전 분야에서 객체인식 모델의 성능을 평가할 때 자주 사용되는 지표인데요,

재현율과 정밀도가 각각 x축, y축에 표시된 PR 곡선 그래프에서, 이 그래프의 면적값이 바로 AP입니다.

29페이지

ㅇ이 AP는 면적이 클수록, 즉 면적이 1에 가까울수록 모델의 성능이 우수하다고 판단됩니다.

그리고, 분류하는 객체 즉, 클래스가 여러 가지인 경우엔 모든 클래스의 AP의 평균인 mAP를 성능 평가 지표로 삼게 됩니다.

30페이지
이러한 AP 수치의 토대가 되는 재현율과 정밀도는 임계값에 따라 어느정도 반비례, 즉 Trade-off 관계가 있습니다.

잘 아시겠지만, 재현율이 낮을 경우에는 객체가 검출되지 않을, 즉 미검출 확률이 높아집니다.

반대로 정밀도가 낮을 때에는 엉뚱한 객체를 대상 객체로 인식하는 경우가 많아집니다.

이 둘 간의 균형이 맞으면서 최대값을 나타내는 게 가장 중요하죠.

31페이지 : 보완 필요
저희 학습 성능의 결과는 다음과 같습니다.

여러 번의 학습 과정을 거치면서, 데이터 불균형 조정, 주야간 데이터 구분 등을 진행했습니다.

최종적으로는 주간 학습 성능 mAP가 0.824, 야간 학습 성능 mAP가 0.926을 보였습니다.

32페이지
이어서 저희가 사용한 모델과, 구현 기능들을 소개하겠습니다.

33페이지
저희는 버전1에서 YOLOv5를 이용했고, 버전2에서부터 YOLOv7을 이용하였습니다.

모델을 변경한 이유는
첫째, 더 정확한 모델을 사용하기 위함이었습니다.
yolov7은 yolov5보다 더 많은 레이어와 더 큰 모델을 사용하기 때문에 더 높은 정확도를 기대할 수 있습니다.

두 번째 이유로는 객체의 경로 추적 알고리즘을 좀 더 잘 조합할 수 있는 모델이었기 때문입니다.

34페이지
YOLO v7은 22년 7월기준, 속도와 정확도 측면에서 다른 모든 Object Detector의 성능을 능가합니다.

그래프에서 보시는 바와 같이, YOLO 중에서 객체 인식 추론 속도와, 그 성능 AP 점수에 있어 가장 좋은 모습을 보여주고 있습니다.

35페이지
추적 알고리즘은 3개를 테스트해보았고,

최종적으로는 sort알고리즘을 선택하였습니다.

DeepSoRT와 Bytetrack은 SORT에 비해 뛰어난 추적 알고리즘이지만,
DeepSort는 리얼타임 트래킹이 안된다는 점이 있었고, 바이트 트랙은 자체 학습한 데이터를 적용할 수 없었기 때문에 최종적으로 SORT 알고리즘을 선택하였습니다.

36페이지
다음은 기능 소개입니다.

앞서 설명드린 객체 탐지 YOLO와 경로 추적 SORT 알고리즘을 이용하여,
서비스의 기능들을 구현했는데요,

첫 번째는 로그, 영상, 동영상 저장 기능입니다.

탐지된 이벤트의 결과물은 로그, 영상, 그리고 동영상으로 저장됩니다.

저장된 자료는, 앞선 서비스 개요에서 설명드렸듯, 서비스 사용자와 관리자에게 제공됩니다.

로그에는 탐지 날짜와 시각, 탐지된 객체의 종류와 수가 기록됩니다.

37페이지
이미지, 즉 영상은 최초 등장시, 그리고 최종 퇴장시 총 두 번 저장됩니다.

최초 등장시에는 객체 등장 전의 감시 지역의 모습을 알 수 있으며, 최종 퇴장시 영상에는 탐지된 객체의 이동 경로 및 객체가 지나간 뒤의 감시 지역의 모습을 볼 수 있습니다.

동영상은 객체가 탐지되는 모든 순간에 기록되며, 객체의 이동경로가 화살표로 표시됩니다.

38페이지
이렇게 저장된 로그, 영상, 동영상 자료는 이벤트 종료 후 AWS로 자동 전송됩니다. 데이터 전송을 위해 pymysql, boto3 라이브러리를 사용했습니다.

39페이지
로그는 정형 데이터베이스 관리 시스템인 RDS에 저장됩니다.

DB는 사용자 정보를 저장하는 user 테이블, 카메라 정보를 담고 있는 cam 테이블 ,로그 정보를 저장하는 로그 테이블로 구성되어 있습니다.

40페이지
이미지나 동영상과 같은 비정형 데이터는 S3에 저장됩니다.

41페이지
다음 기능은, 객체 퇴치 기능입니다.

유해동물 객체가 사용자가 설정한 위험 범위 내에 진입하면, 미리 설정된 퇴치 솔루션이 실행됩니다.

현재는, 호랑이 울음소리를 내는 것으로 설정되어있고, 퇴치 방법은 각 객체마다 다르게 구현할 수 있습니다.

이 기능은, 탐지 객체의 위치값을 바운딩박스 중앙 하단으로 잡아, 이 위치값이 위험 범위의 y값을 넘어서는 순간 솔루션이 작동되도록 구현했습니다.

그리고 여기서 보이는 위험 범위는, 사용자 및 설치 환경에 따라 맞춤 설정을 할 수 있습니다.

첫 번째는 탐지 객체의 위치값 판단입니다. 탐지 모델을 통해 탐지된 객체는 영상에서 바운딩 박스로 표시됩니다. 우리는 바운딩 박스의 중앙 하단, 즉 객체가 땅을 딛고 서 있는 부분을 객체의 위치로 설정했습니다.

두 번째는 위험 구역의 설정입니다. 영상은 최소 단위인 픽셀로 이루어져 있으며, 각 픽셀에는 좌표값이 부여됩니다. 이 중 높이(y값)을 통해 위험 구역을 설정했습니다.

객체의 위치값이 위험구역 y값보다 높아질 경우(=더 가까이 올 경우) 경보 알람이 울리도록 구현하였습니다.

42페이지
이렇게 구현한 기능들은, YOLO의 detect 파일의 옵션으로 추가하여, 프로그램 실행 시 설정할 수 있게끔 하였습니다.

43페이지
다음으로, 웹 페이지를 통해 서비스되는 모습을 소개해드리겠습니다.

44페이지
저희 사이트는

메인화면, 회원가입과 로그인 화면, 그리고 이벤트 내역을 볼 수 있는 이벤트 게시판, 이벤트의 상세 내용을 볼 수 있는 이벤트 상세 페이지로 구성되어 있습니다.

자세한 내용은 뒷쪽 시연에서 보여드리겠습니다.

45페이지
그럼 바로 시연, 진행하겠습니다.

46페이지

47페이지
앞서 보신 시연에서 구현한 개념 몇 가지를 추가로 말씀 드리겠습니다.

저희가 고민했던 부분인데, 객체가 등항하고, 존재하고, 퇴장하는 내역을 어떻게 사용자가 보기 쉽게 보여줄 것인가 였습니다.

우리의 서비스에서는 이벤트를 묶어주는 알고리즘을 구축해 객체 탐지의 결과값이 무질서하게 나열되는 것이 아니라, 탐지 시점에 따라 체계적으로 정리되고, 이벤트별로 나뉘게 설계하였습니다.

이렇게 이벤트를 묶음으로서 AWS 와의 통신횟수를 줄여, 서버 부하를 줄일 수 있다는 장점이 있었습니다.

48페이지
그 다음으로는, 멀티 스레딩입니다.

본 서비스의 객체 탐지부에서는 객체탐지, 경보, AWS 서버로의 자료 전송이 동시에 이루어져야 합니다.

이때, 기능들 중 하나의 기능이 작동될 동안 다른 기능이 중지되지 않도록 Multi Threading을 적용 하였습니다.

프로세스 내에서 작업의 실행 흐름 단위를 말하는 스레드,

만약 이 스레드가 하나일 경우에는 경보가 울릴 동안 객체 탐지 기능은 중지됩니다. 그러면 서비스에 치명적인 단점이 되겠죠.

이에 따라 각 기능들을 각각 다른 스레드에 올려 실시간으로 동시에 실행될 수 있도록 구현하였습니다.

49페이지
프로젝트를 끝마치는 결론 말씀드리겠습니다.

50페이지
저희의 프로젝트를 정리하겠습니다

(1) 환경에 적응하는 모델(서비스)
요즘 CCTV들은 많이 좋아져서, 실시간 모니터링도 가능하고, 객체 검출도 가능합니다.
여기에서 더 나아가 우리의 서비스는, 어떤 환경에서, 어떤 객체들을 검출할지, 그 대상을 서비스 사용자에 맞게 학습시킬 수 있습니다. 더불어, 주간과 야간 같은 환경에 대한 변수도 함께요.

(2) 경로 추적
저희는 경로 추적 기능을 구현했습니다. 이로써 객체의 행동특성을 파악할 수 있고, 이를 서비스 개선에 이용할 수 있습니다. 예를 들어 멧돼지가 어디에서 등장해서 어디로 퇴장했다 라는 것을 서비스 사용자와 관리자가 모두 모니터링 할 수 있고, 서비스 사용자는 이에 대한 대응책을 세울 수 있고, 관리자는 더 좋은 퇴치 서비스를 고민할 수 있습니다.

(3) 커스텀 가능
앞서 말한 첫 번째 이유와 비슷한데요, 우리 서비스의 주요 기능 중 하나인 퇴치 기능, 이 퇴치 기능에서는 위험범위을 사용자가 직접 설정할 수 있습니다.

또한 센서기반 제품들보다 이미지 딥러닝 기반으로, 지속적인 모니터링이 가능하며, 이를 이용해 서비스 개선을 할 수 있습니다.

다음은 저희가 보완해야 할 사항입니다.

학습단계에서 배치사이즈를 늘려서 프레임속도를 높인다면 더 빠르게 탐지할 것으로 기대합니다.

또한 서비스면에서
실제 사용자들의 경험을 수집하여 스마트폰 알림등 유저 친화적으로 서비스를 개선하려고 합니다.

51페이지
향후 발전방향 다음과 같이 3가지로 설정했습니다.

먼저 단안카메라를 이용한 객처 거리 탐지기술 연구 입니다.

저희는

거리 탐지 기능을 통해 탐지부 최적 세팅값(높이, 각도) 구현
- 거리에 따른 퇴치율 측정 등으로 서비스 정량평가 가능
객체와 카메라의 물리적으로 떨어진 거리를

이미지상에서 객체의 수치와 의 관계를 알아보려는 실험을 진행중입니다.

이미 선행 연구자료를 통해 단안카메라의 차간거리측정 등 연구는 테슬라를 비롯한 자율주행 자동차 분야의 메인 과제였었습니다.

가설과 검증을 통해 어느정도 수치의 유의미함을 알 수 있었습니다

장비와 실험여건의 부족등이 개선하여 충분한 데이터를 확보한다면 서비스개선에 도움이 될 것이라 기대합니다.

52페이지
나아가 저희는 게임의 좌측이나 우측하단에서 유저의 게임진행을 돕는 미니맵의 개념을 구상하였습니다.

이에 카메라의 화각을 넓이는 방법으로 다수의 카메라의 영상을 겹치게 하여 탐지영역을 넓힌다면 감시영역을 확장하고

다수의 카메라 구동으로 비용감소를 기대합니다

세번째로 소프트웨어를 개선하는 방법으로 더블 버퍼링 원리를 적용하는 방법 입니다.

이는 시스템 부담을 저감하고 빠르게 구동하여 사용자 UI 만족을 확대할 것으로 기대합니다.

53페이지
저희가 프로젝트에서 참고한 참고문헌입니다.

이후 : 발표자료