Abstract

Object detection is an important feature of computer science. The benefits of object detection is however not limited to someone with a doctorate of informatics.
객체 탐지는 컴퓨터 과학의 중요한 특징입니다. 그러나 객체 검출의 이점은 정보학 박사 학위를 가진 사람으로 제한되지 않습니다.

Instead, object detection is growing deeper and deeper into the common parts of the information society, lending a helping hand wherever needed. This paper will address one such possibility, namely the help of a Haar-cascade classifier.
대신, 개체 탐지는 정보 사회의 공통 부분을 더욱 깊고 깊게 성장시켜 필요한 곳 어디에서나 도움의 손길을 빌려주고 있습니다. 이 논문은 그러한 가능성 중 하나, 즉 Haar-cascade 분류자의 도움을 다룰 것입니다.

The main focus will be on the case study of a vehicle detection and counting system and the possibilities it will provide in a semi-enclosed area - both the statistical kind and also for the common man. The goal of the system to be developed is to further ease and augment the everyday part of our lives.
주된 초점은 차량 탐지 및 계산 시스템의 사례 연구와 그것이 반 밀폐 지역에서 제공 할 수있는 가능성에 대한 것입니다 - 통계 종류 및 일반인 모두. 개발 될 시스템의 목표는 일상 생활의 일부를 더욱 쉽게하고 증대시키는 것입니다.

1. Introduction and theory

1.1 Computer vision

Computer vision is a field of informatics, which teaches computers to see. It is a way computers gather and interpret visual information from the surrounding environment [1].
컴퓨터 비전은 컴퓨터에 알려주는 정보 과학 분야입니다. 컴퓨터가 주변 환경의 시각 정보를 수집하고 해석하는 방법입니다 [1].

Usually the image is first processed on a lower level to enhance picture quality, for example remove noise. Then the picture is processed on a higher level, for example detecting patterns and shapes, and thereby trying to determine, what is in the picture [2].
일반적으로 이미지는 노이즈 제거와 같이 화질을 높이기 위해 먼저 낮은 레벨에서 처리됩니다. 그런 다음 그림은 패턴과 모양을 감지하는 등 더 높은 수준에서 처리되어 그림에있는 내용을 확인하려고 시도합니다 [2].

1.2 Object detection

Object detection is commonly referred to as a method that is responsible for discovering and identifying the existence of objects of a certain class. An extension of this can be considered as a method of image processing to identify objects from digital images.

객체 검출은 일반적으로 특정 클래스의 객체의 존재를 발견하고 식별하는 책임을지는 메소드라고합니다. 이 확장은 디지털 이미지에서 객체를 식별하는 이미지 처리 방법으로 간주 될 수 있습니다.

1.3 Simple detection by colour

One way to do so, it to simply classify objects in images according to colour. This is the main variant used in, for example, robotic soccer, where different teams have assembled their robots and go head to head with other teams.
이를 수행하는 한 가지 방법은 이미지를 색상별로 분류하는 것입니다. 이것은 다른 팀이 로봇을 조립하고 다른 팀과 머리를 맞대는 로봇 축구에서 사용되는 주요 변형입니다.

However, this color-coded approach has its downsides. Experiments in the international RoboCup competition have shown that the lighting conditions are extremely detrimental to the outcome of the game and even the slightest ambient light change can prove fatal to the success of one or the other team.
그러나이 색으로 구분 된 접근 방식에는 단점이 있습니다. 국제 RoboCup 대회의 실험에 따르면 조명 조건은 게임 결과에 매우 해롭고 약간의 주변 조명 변경조차도 한 팀 또는 다른 팀의 성공에 치명적일 수 있음을 보여줍니다.

Participants need to recalibrate their systems multiple times even on the same field, because of the minor ambient light change that occurs with the time of day. [3] Of course, this type of detection is not suitable for most real world applications, just because of the constant need for recalibration and maintenance.
참가자는 하루 중 발생하는 주변 조명의 변화가 적기 때문에 동일한 필드에서도 여러 번 시스템을 재조정해야합니다. [3] 물론 이러한 유형의 탐지는 재 보정 및 유지 관리가 끊임없이 필요하기 때문에 대부분의 실제 응용 프로그램에는 적합하지 않습니다.

1.4 Introduction of Haar-like features

A more sophisticated method is therefore required. One such method would be the detection of objects from images using features or specific structures of the object in question.
따라서 더 정교한 방법이 필요합니다. 이러한 방법 중 하나는 문제의 대상물의 특징 또는 특정 구조를 사용하여 이미지에서 대상물을 검출하는 것입니다.

However, there was a problem. Working with only image intensities, meaning the RGB pixel values in every single pixel in the image, made feature calculation rather computationally expensive and therefore slow on most platforms.
그러나 문제가있었습니다. 이미지 강도만으로 작업하면 이미지의 모든 단일 픽셀에서 RGB 픽셀 값을 사용하므로 대부분의 플랫폼에서 연산이 비싸고 속도가 느려집니다.

This problem was addressed by the socalled Haar-like features, developed by Viola and Jones on the basis of the proposal by Papageorgiou et. al in 1998.
이 문제는 Papageorgiou 등의 제안을 토대로 Viola와 Jones가 개발 한 Haar와 유사한 기능으로 해결되었습니다. 1998 년 알.

A Haar-like feature considers neighbouring rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums. This difference is then used to categorize subsections of an image.
Haar와 같은 특징은 탐지 창의 특정 위치에서 이웃 직사각형 영역을 고려하여 각 영역의 픽셀 강도를 합산하고 이러한 합계 간의 차이를 계산합니다. 이 차이는 이미지의 하위 섹션을 분류하는 데 사용됩니다.

An example of this would be the detection of human faces. Commonly, the areas around the eyes are darker than the areas on the cheeks. One example of a Haar-like feature for face detection is therefore a set of two neighbouring rectangular areas above the eye and cheek regions. [4]
이것의 한 예가 인간의 얼굴을 탐지하는 것입니다. 일반적으로 눈 주위의 영역은 볼의 영역보다 더 어둡습니다. 얼굴 검출을위한 Haar-like 특징의 한 예는 눈과 뺨 영역 위의 두 인접한 직사각형 영역의 집합입니다. [4]

1.5 Cascade classifier

The cascade classifier consists of a list of stages, where each stage consists of a list of weak learners.
계단식 분류기는 단계 목록으로 구성되며, 각 단계는 약한 학습자 목록으로 구성됩니다.

The system detects objects in question by moving a window over the image. Each stage of the classifier labels the specific region defined by the current location of the window as either positive or negative – positive meaning that an object was found or negative means that the specified object was not found in the image.
시스템은 창을 이미지 위로 이동시켜 문제의 물체를 감지합니다. 분류기의 각 단계는 창의 현재 위치로 정의 된 특정 영역을 양수 또는 음수로 레이블링합니다. 양수인은 객체가 발견되었거나 음수는 지정된 객체가 이미지에서 발견되지 않았 음을 의미합니다.

If the labelling yields a negative result, then the classification of this specific region is hereby complete and the location of the window is moved to the next location.
라벨링이 부정적인 결과를 산출하면,이 특정 영역의 분류가 완료되고 윈도우의 위치가 다음 위치로 이동된다.

If the labelling gives a positive result, then the region moves of to the next stage of classification.
라벨링이 긍정적 인 결과를 제공하면 지역 분류의 다음 단계로 이동합니다.
.
The classifier yields a final verdict of positive, when all the stages, including the last one, yield a result, saying that the object is found in the image.
분류기는 최종 단계를 포함하여 모든 단계가 결과를 산출하고 객체가 이미지에서 발견되면 양의 최종 평결을 산출합니다.

A true positive means that the object in question is indeed in the image and the classifier labels it as such – a positive result. A false positive means that the labelling process falsely determines, that the object is located in the image, although it is not. A false negative occurs

when the classifier is unable to detect the actual object from the image and a true negative means that a nonobject was correctly classifier as not being the object in question.
진정한 긍정적 인 것은 문제의 대상이 실제로 이미지에 포함되어 있고 분류자가 해당 대상으로 표시한다는 것을 의미합니다. 긍정적 인 결과입니다. 거짓 긍정 (false positive)은 레이블링 프로세스가 이미지가 아닌 경우에도 이미지가 해당 이미지에 있음을 잘못 판단하는 것을 의미합니다. 분류기가 이미지의 실제 객체를 감지 할 수 없으며 참된 부정적 결과는 객체가 아닌 객체가 문제의 객체가 아니라는 것을 의미합니다.

In order to work well, each stage of the cascade must have a low false negative rate, because if the actual object is classified as a non-object, then the classification of that branch stops, with no way to correct the mistake made. However, each stage can have a relatively high false positive rate, because even if the n-th stage classifies the non-object as actually being the object, then this mistake can be fixed in n+1-th and subsequent stages of the classifier. [5]
실제로 잘 작동하기 위해서는 캐스케이드의 각 단계에서 낮은 오탐률을 가져야합니다. 실제 객체가 비 객체로 분류 된 경우 해당 분기의 분류가 중단되고 실수를 바로 잡을 방법이 없기 때문입니다. 그러나, 비록 n 번째 스테이지가 실제로 비 객체를 객체로 분류하더라도, 이러한 실수는 분류기의 n + 1 번째 및 후속 스테이지에서 고정 될 수 있기 때문에, 각 스테이지는 비교적 높은 오 탐지율을 가질 수있다 . [5]

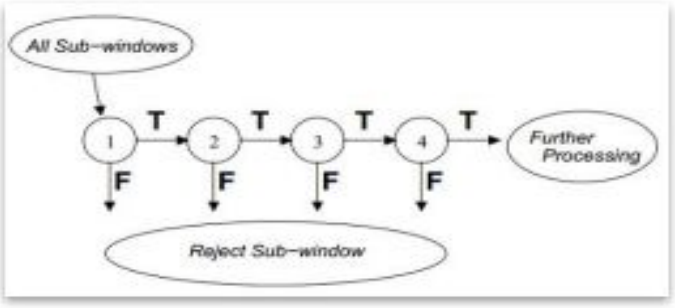

***Image 1:** Stages of the cascade classifier*

2. Related work

2.1 PureTech Systems Car Counting

The system uses images received from IP and analog video cameras it to detect and count vehicles. Advanced background algorithms then filter any unnecessary and probable interference, such as shadows or lighting changes. When an object in detected, special filters make sure to minimize the chance to count nonvehicle items, for example humans and luggage. The finalized count is then outputted, based on the initial configuration - by floor or special zone.
이 시스템은 IP 및 아날로그 비디오 카메라에서 수신 한 이미지를 사용하여 차량을 감지하고 계산합니다. 고급 배경 알고리즘은 그림자 또는 조명 변경과 같은 불필요하고 가능한 간섭을 필터링합니다. 감지 된 특수 필터의 물체가 비히클 항목 (예 : 사람과 수하물)을 계산할 기회를 최소화해야합니다. 완성 된 카운트는 바닥 또는 특수 구역별로 초기 구성에 따라 출력됩니다.

The video is processed at a central monitoring location, which means there is no need to make cuts into pavement or similarly preinfluence the environment in such a way, which is commonly needed for inductive loop traffic detectors, where the detectors need to be placed inside the pavement, which is a fairly common method of detecting vehicles behind traffic lights.
비디오는 중앙 모니터링 위치에서 처리됩니다. 즉, 포장 도로를 자르거나 이와 유사한 방법으로 환경을 사전에 유출 할 필요가 없습니다. 이는 유도 루프 트래픽 탐지기에 일반적으로 필요합니다. 탐지기는 내부에 배치해야합니다. 포장 도로는 교통 신호등 뒤에있는 차량을 감지하는 일반적인 방법입니다.

One problem with this system is, however, that it only detects cars as they enter or leave the parking lot. This in turn means that there needs to be such a camera or counter in front of every entrance and exit point, in order to ensure that none of the vehicles are missed, which means that all these cameras need to be bought, installed and serviced, which means that the specific client will face a possibly substantial expenditure in order to keep all the devices up and running. [6]
그러나이 시스템의 한 가지 문제는 주차장에 들어가거나 주차 할 때만 차량을 감지한다는 것입니다. 이것은 차량의 어느 것도 놓치지 않도록 보장하기 위해 모든 입구와 출구 지점 앞에 카메라 또는 카운터가 있어야한다는 것을 의미합니다. 즉, 모든 카메라를 구입하고 설치하고 수리해야한다는 것을 의미합니다. 즉, 특정 클라이언트가 모든 장치를 계속 실행하기 위해 상당한 지출에 직면하게됩니다. [6]

2.2 Autonomous Real-time Vehicle Detection from a Medium-Level UAV

The primary means for vehicle detection make use of multiple cascaded Haar classifiers. For the application, four separate Haarcascade classifiers were trained based on sample vehicles categorized into four different positional orientation to the horizontal line, resulting in the clockwise offsets of 0, 45, 90, 135 degrees. The training set was thereby divided into disjoint subsets based on the nearest perceived angle of the vehicle wheelbase. A separate cascaded Haar-classifier was trained for each of the subsets.
차량 탐지의 주요 수단은 다중 계단식 Haar 분류기를 사용합니다. 응용 프로그램의 경우 4 개의 개별 Haarascade 분류기가 수평선에 대해 4 가지 위치 방향으로 분류 된 샘플 차량을 기반으로 훈련되어 0, 45, 90, 135도 시계 방향으로 오프셋되었습니다. 트레이닝 세트는 차량 휠베이스의 가장 가까운 지각에 기초하여 분리 된 서브 세트로 분할되었다. 각 하위 집합에 대해 별도의 계단식 Haar 분류기를 교육했습니다.

The results are subsequently passed through a secondary disjunctive verification process, which means that a vehicle may exist in one or more of the input images, if one or more of the different orientation specific classifiers yields a positive result. The returned set of regions are then merged to resolve overlaps and output a singular set of detections from the inputs. Each region is also tested by various logical tests of width and height – is it actually probable for the image of this size to be a vehicle. [7]
그 결과는 2 차 분리 검증 프로세스를 거쳐 전달되며, 이는 하나 이상의 상이한 오리엔테이션 특정 분류기가 긍정적 결과를 산출하는 경우 차량이 하나 이상의 입력 이미지에 존재할 수 있음을 의미합니다. 그런 다음 반환 된 영역 집합을 병합하여 겹침을 해결하고 입력에서 단일

검출 집합을 출력합니다. 각 영역은 폭과 높이의 다양한 논리적 테스트에 의해 테스트됩니다. 실제로이 크기의 이미지가 차량이 될 가능성이 큽니다. [7]

2.3 Monocular Vehicle Detection and Tracking

The system developed involves three main steps.
개발 된 시스템은 세 가지 주요 단계로 이루어집니다.

Firstly, the preliminary car targets are generated with Haar-cascade classifier.
첫째, 예비 차량 표적은 Haar-cascade 분류기로 생성됩니다.

Only the candidates that pass through all the stages are classified as positive and the ones that are rejected at any stage are classified as negative. The advantage is that the majority of the initial candidates are actually negative images, which usually cannot pass the first few stages. This early-stage rejection thus greatly reduces the overall computing time.
모든 단계를 거친 후보자 만 긍정적으로 분류되고 어떤 단계에서 거부 된 후보는 부정적으로 분류됩니다. 장점은 초기 후보자의 대부분이 실제로는 처음 몇 단계를 통과 할 수없는 음화 이미지라는 점입니다. 이렇게 초기 단계의 거부는 전체 컴퓨팅 시간을 크게 줄입니다.

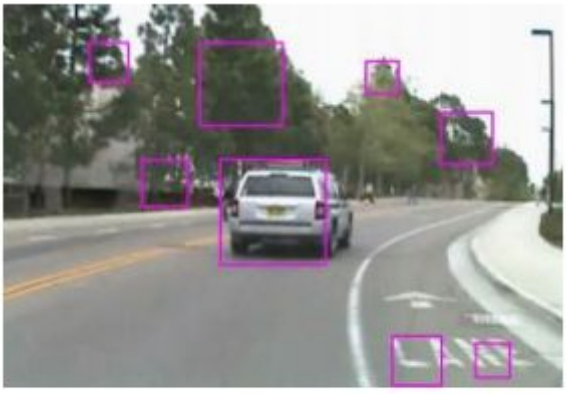

Image 2: Car candidates detected by Haar-cascade

이미지 2 : Haarascascade가 탐지 한 자동차 후보

After the first stage is complete, the target validation is used, which is based on car-light feature. Since there can be several false positives within the images output by the first stage of the system, which comes from the limitation of the training set. To reduce the false alarm rate, the algorithm uses the before mentioned car-light feature. Regardless of the shape, texture or colour of the vehicle, they all share a common feature – they all have red lights in the rear. Thereby assuming that most of the false positives detected by the first stage do not possess such a feature, the result can thus be refined.
첫 번째 단계가 완료된 후 자동차 조명 기능을 기반으로하는 타겟 검증이 사용됩니다. 시스템의 첫 번째 단계에서 출력되는 이미지에는 몇 가지 오 탐지 (false positives)가있을 수 있으므로 이는 교육 세트의 제한 사항에서 비롯됩니다. 잘못된 경보 속도를 줄이기 위해 알고리즘은 앞서 언급 한 자동차 조명 기능을 사용합니다. 차량의 모양, 질감 또는 색상에

관계없이 모두 공통점이 있습니다. 후면에는 빨간색 등이 있습니다. 따라서 첫 번째 단계에서 탐지 된 대부분의 오 탐지 (false positive)가 그러한 특징을 갖고 있지 않다고 가정하면 결과를 정제 할 수 있습니다.

Lastly, the results are further refined by Kalman tracking of the objects. The main idea of this step is based on a three-stage hypothesis tracking. Firstly, if a newly detected area appears and lasts for more than a certain number of steps, a hypothesis of the object is generated. Then it is predicted where the next location of the object should be.
마지막으로, 결과는 Kalman에 의해 개체의 추적에 의해 더욱 정교 해집니다. 이 단계의 주요 아이디어는 3 단계 가설 추적을 기반으로합니다. 첫째로, 새롭게 검출 된 영역이 나타나서 일정 단계 이상 지속될 경우, 객체의 가설이 생성됩니다. 그런 다음 개체의 다음 위치를 예측해야합니다.

3. Object detection using Haarcascade classifier

This section will highlight on the work conducted on the author's research in the field of object detection using Haarcascade classifier. The experiments were conducted mainly on the parking lot located on Kivi street side parking lot of Raatuse Shopping center. The location was chosen mainly for the ease of access and security for the hardware required to gather information.

3.1Hardware

Initial testing was conducted with Qoltec WebCam15QT NightVision 30Mpx USB camera. The device was chosen due to its alleged high capabilities, especially the 30MP camera.
초기 테스트는 Qoltec WebCam15QT NightVision 30Mpx USB 카메라를 사용하여 수행되었습니다. 이 장치는 높은 성능, 특히 30MP 카메라로 인해 선정되었습니다.

As it later turned out, the information provided was faulty, and the quality of the camera was in fact much poorer. For this reason, the device was replaced with Logitech HD Webcam C525 which offered much clearer images than its predecessor.
나중에 밝혀진 바와 같이, 제공된 정보는 불완전하며 카메라의 품질은 실제보다 훨씬 뒤떨어져있었습니다. 이러한 이유 때문에이 장치는 이전 제품보다 훨씬 선명한 이미지를 제공하는 Logitech HD Webcam C525로 대체되었습니다.

The camera was programmed to take pictures every five minutes, to minimize the impact on the storage capacity and duplicate images, since the changes during five minutes in the parking lot were observed to be minimal.
카메라는 5 분마다 사진을 찍어서 저장 용량에 미치는 영향을 최소화하고 이미지를 복제하도록 프로그램되었습니다. 왜냐하면 주차장에서의 5 분간의 변화가 최소한으로 관찰되었기 때문입니다.

If the object is not detected for a certain number of frames, the hypothesis is discarded. This method can thusly eliminate false positives that do not last long enough and still keep track of objects that are missing for only a short period in a detection step. [8]
특정 수의 프레임에 대해 물체가 감지되지 않으면 가설이 삭제됩니다. 이 방법은 오랫동안 지속되지 않는 오 탐지 (false positives)를 제거하고 감지 단계에서 단시간 동안 누락 된 객체를 계속 추적 할 수 있습니다. [8]

3.2 Software

Several programs were developed in the course of this paper, ranging from a simple convert to grayscale and get size of picture to recorder, detector and PosCreator.
이 문서의 과정에서 단순 변환에서부터 그레이 스케일에 이르기까지 다양한 프로그램이 개발되었으며 사진의 크기가 레코더, 감지기 및 PosCreator로 확대되었습니다.

3.2.1 Recorder

Recorder application was a simple application which after every 5 minutes tries to take a picture. If it can, then a picture is saved to a folder of the corresponding date with the filename of the corresponding time. If it cannot, then it simply cuts the connection within 30 seconds and will simply wait for the next 5 minutes. This ensures that if there is a problem with taking a picture, which would cause the program to “freeze”, then it is simply stops the program and tries again later, instead of potentially waiting until the power runs out someone manually stops the program. This is a must-have feature is such an application, due to the fact that several hours’ worth of image gathering would be wasted due to any simple problem that halts the execution of the recorder thread.
레코더 응용 프로그램은 5 분마다 사진을 찍으려고하는 간단한 응용 프로그램이었습니다. 가능한 경우 해당 시간의 파일 이름으로 해당 날짜의 폴더에 그림이 저장됩니다. 연결할 수 없으면 30 초 이내에 연결을 끊고 다음 5 분 동안 기다립니다. 이렇게하면 사진을 찍는 데 문제가 생기면 프로그램이 "정지"될 수 있습니다. 그러면 누군가 프로그램이 수동으로 중지 될 때까지 기다리지 않고 프로그램을 중지하고 나중에 다시 시도 할 수 있습니다. 레코더 스레드의 실행을 중단시키는 간단한 문제로 인해 수 시간의 이미지 수집이 낭비된다는 사실 때문에 이러한 응용 프로그램이 반드시 있어야하는 기능입니다.

3.2.2 Detector

The initial design of the detector application was quite simplistic. Firstly, the detector would load the classifier and determine it is not empty. If it is, then it simply exits with an error message. Then the image in question is loaded and same procedure is followed. Then classifier is applied to the image, which outputs an array of rectangles, which correspond to the detected positions of the objects, in this case automobiles. The program would then draw bright red rectangles in the locations of the detections and also add a text to the image, which could for example identify the classifier used, since one classifier would usually detect one thing.
검출기 응용 프로그램의 초기 디자인은 매우 단순했습니다. 첫째, 탐지기가 분류기를로드하여 비어 있지 않다고 결정합니다. 그럴 경우 오류 메시지와 함께 종료됩니다. 그런 다음 해당 이미지가로드되고 동일한 절차가 수행됩니다. 그런 다음 분류기가 이미지에

적용되어 개체의 감지 된 위치 (이 경우 자동차)에 해당하는 사각형 배열을 출력합니다. 그런 다음 프로그램은 탐지 위치에 밝은 빨간색 직사각형을 그리고 이미지에 텍스트를 추가합니다.이 텍스트는 예를 들어 하나의 분류기가 일반적으로 한 가지를 감지하므로 사용 된 분류기를 식별 할 수 있습니다.

3.2.2.1 Background subtraction

However, as shown by the testing process and the literature, the classifiers trained can produce errors – either false positives or false negatives, as described above.
그러나 테스트 프로세스 및 문헌에서 볼 수 있듯이, 훈련 된 분류기는 위에 설명 된 것처럼 오탐 (false positive) 또는 거짓 네거티브 (false negative)를 생성 할 수 있습니다.

In order to minimise the false positive rate originating from the imperfections of the classifier, an additional layer was added to the algorithm, before the classifier is applied to the image. This layer has additional knowledge of the complete background. In this case it would be an image of only the parking lot and everything that would normally be in the parking lot, except for the cars themselves. This knowledge can be applied to attempt the filtering of the background from the image from which we would like to detect vehicles. The background subtraction type used was MOG.
분류 자의 불완전성에 기인 한 오 탐지율을 최소화하기 위해 분류자를 이미지에 적용하기 전에 추가 계층이 알고리즘에 추가되었습니다. 이 레이어는 완전한 배경 지식을 가지고 있습니다. 이 경우 자동차 자체를 제외하고는 주차장에만있는 이미지와 일반적으로 주차장에있는 모든 이미지가됩니다. 이 지식은 우리가 차량을 감지하고자하는 이미지로부터 배경 필터링을 시도하는데 적용될 수 있습니다. 사용 된 배경 빼기 유형은 MOG입니다.

MOG (abbr. from Mixture of Gaussians) is a technique used to model the background pixels of an image like a mixture of Gaussians of different weight that represent the pixel intensity. If a match is found for the pixel of the new image with one of the Gaussians then the point is classified as a background pixel. If no match is found, then the pixel is classified as the foreground pixel. [9] Other algorithms, such as MOG2 were considered, but MOG was finally chosen due to the simple fact that clearer results were obtained by using MOG.
MOG (혼합 가우시안 (Mixture of Gaussians)의 약자)는 픽셀의 강도를 나타내는 서로 다른 가중치의 가우시안이 혼합되어있는 것처럼 이미지의 배경 픽셀을 모델링하는 데 사용되는 기술입니다. 가우시안 중 하나를 갖는 새로운 이미지의 픽셀에 대한 일치가 발견되면, 그 포인트는 배경 픽셀로 분류된다. 일치하는 것이 없으면 픽셀은 전경 픽셀로 분류됩니다. [9] MOG2와 같은 다른 알고리즘이 고려되었지만, MOG를 사용하여 더 명확한 결과를 얻을 수 있다는 단순한 사실 때문에 MOG가 최종적으로 선택되었습니다.

MOG gives us the background mask, so in order to apply it to the original picture, one would simply need to compute the bitwise and between the original image and the mask provided.
MOG는 배경 마스크를 제공하므로 원본 그림에 적용하려면 원본 이미지와 제공된 마스크 사이의 비트를 계산하면됩니다.

MOG is, however, not perfect. If we were to just take the mask provided by the default MOG background extractor, then the output for one image of the parking lot would be rather low quality, as illustrated on image 3. Although a person may differentiate the regions of cars in

the image, a cascade classifier proved unable to properly comprehend the regions of cars on a similar image.

그러나 MOG는 완벽하지 않습니다. 기본 MOG 배경 추출기가 제공 한 마스크를 그냥 가져 가면 이미지 3에 나와있는 것처럼 주차장의 한 이미지에 대한 출력이 오히려 낮은 품질이됩니다. 이미지에서 자동차 영역을 차별화 할 수는 있지만 계단식 분류기는 유사한 이미지에서 자동차 영역을 제대로 이해할 수 없음을 입증했습니다.

Image 3: Output using MOG with default parameters

이미지 3 : 기본 파라미터로 MOG를 사용하여 출력

3.2.2.2 Background subtraction augmentation

In order to amend this issue, different augmenting features had to be used. The ones chosen were eroding and dilating.
이 문제를 수정하려면 다양한 기능을 사용해야했습니다. 선택한 사람들은 부식하고 팽창했습니다.

Dilation is a way to make the bright regions of an image to “grow”. As the kernel (small matrix used for image processing) is scanned over the image, the maximal pixel value overlapped by the kernel is calculated and the image pixel in the anchor point of the kernel (usually at the centre of the image) is replaced by the maximal value.
확대는 이미지의 밝은 영역을 "grow"시키는 방법입니다.
커널 (이미지 처리에 사용되는 작은 행렬)이 이미지 위에 스캔 될 때 커널에 의해 중첩 된 최대 픽셀 값이 계산되고 커널의 앵커 포인트 (일반적으로 이미지의 중심)에있는 이미지 픽셀이 최대 값으로 대체됩니다.

Erosion works similarly to dilation, but instead of the maximal, it computes the local minimum over the area of the kernel, thus making the dark areas of the image larger.
침식은 팽창과 유사하게 작동하지만 최대 값 대신에 커널 영역에 대한 로컬 최소값을 계산하므로 이미지의 어두운 영역을 더 크게 만듭니다.

If one were to apply dilation to the mask provided by MOG, then the areas of the mask which are not zeros would get larger, thus improving the overall quality of the image.

MOG에서 제공하는 마스크에 확장 기능을 적용하면 0이 아닌 마스크 영역이 커져 이미지의 전반적인 품질이 향상됩니다.

This can however raise a new issue, namely the fact that the small noisy areas present in the original mask could grow larger and have a negative effect on the provided mask.
그러나 이것은 새로운 문제, 즉 원래의 마스크에 존재하는 작은 잡음이 많은 영역이 커져서 제공된 마스크에 부정적인 영향을 줄 수 있다는 사실을 제기 할 수 있습니다.

For this reason, the once dilated mask is eroded with a kernel with a smaller size, so that it would not nullify the result provided by the dilating but still reducing the amount of noise produced by the dilation process, thus providing a symbiotic relation between the two operations

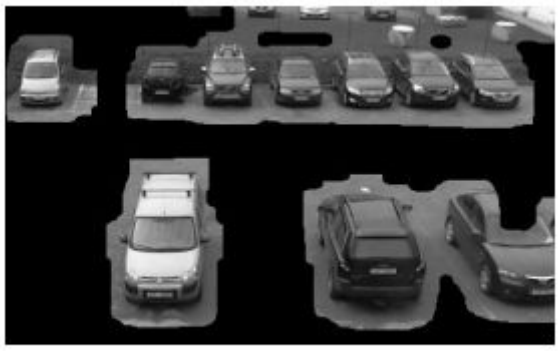

Image 4: MOG with dilation and erosion

이미지 4 : 팽창과 침식이있는 MOG

The results provided by this sort of background filtering were improved. Since a lot of the false positives provided by the original detections were in fact on the background part, such as the trees, pavement etc., which is always there, then the algorithm discarded these areas before the Haarcascade classifier would be applied. However, the regions created by the background removal created additional problems, such as the classifier mistaking the grey to black regions as the positive image.
이러한 종류의 백그라운드 필터링이 제공하는 결과가 개선되었습니다. 원래 감지에 의해 제공되는 오 탐지 (false positive)의 많은 부분이 실제로 나무, 포장 도로 등과 같이 배경 부분에 있었기 때문에 하카 스케이드 분류자가 적용되기 전에 알고리즘이이 영역을 폐기했습니다. 그러나 배경 제거로 생성 된 영역은 분류기가 회색에서 검정색 영역을 긍정적인 이미지로 잘못 인식하는 것과 같은 추가 문제를 야기했습니다.

3.2.3 Regional merging

In addition, yet another layer of filtering was added, inspired by the “Autonomous Real-time Vehicle Detection from a Medium-Level UAV” article. This layer was added after the Haar-cascade has done its work. Since it is possible for the classifier to detect many parts of a car as several distinct cars, the results of the classifier regions must be merged. The merge cannot however just detect whether the two regions have an overlap of any size, because the overlap can simply be the result of two adjacent cars, which would thereby be merged into one result. To avoid this unwanted behaviour, the two regions are merged only

if the overlapped area is at least a certain percentage of one of the regions. This constant was observed to yield the best results if the value was approximately 40-60%, meaning that if the overlap of the two regions made up at least 40- 60% of the overall area of one of the regions, then the two regions were merged and considered as one region from there on. An example of this procedure is shown on image 5, where green lines denote the original detection and the red line the final output of the detector.

또한 "중간 수준의 UAV로부터의 자율 실시간 차량 탐지"기사에서 영감을 얻은 또 다른 필터링 계층이 추가되었습니다. 이 층은 Haar-cascade가 작업을 마친 후에 추가되었습니다. 분류기가 차의 여러 부분을 여러 차종의 자동차로 감지 할 수 있으므로 분류 자 지역의 결과를 병합해야합니다. 그러나 병합은 두 영역의 크기가 겹치지 않았는지 여부를 감지 할 수 없습니다. 왜냐하면 오버랩은 인접한 두 자동차의 결과 일 뿐이므로 하나의 결과로 병합됩니다. 이러한 원하지 않는 동작을 피하기 위해 두 영역은 겹친 영역이 영역 중 하나의 특정 비율 이상인 경우에만 병합됩니다. 이 상수는 값이 약 40-60 % 인 경우 최상의 결과를 나타내는 것으로 관찰되었는데, 이는 두 영역의 중첩이 영역 중 하나의 전체 영역의 최소 40-60 %를 구성하면 두 영역 병합되어 거기에서 한 지역으로 간주됩니다. 이 절차의 예는 이미지 5에 표시되어 있습니다. 녹색 선은 원래 탐지를 나타내고 빨간색 선은 탐지기의 최종 출력을 나타냅니다.

Image 5: Regional merge in action

이미지 5 : 지역 통합 병동

Due to the probabilistic nature of the Haar-cascade, the false positives may also include detection, which logically cannot contain an automobile – namely when the rectangle identified as a positive result is either too large or too small. Such a result would seriously compromise the result set, because the algorithm described for the merging of areas would simply merge all these areas into one large area. In order to avoid this, the results of the Haarcascade are first prefiltered, so that areas that are too large to contain just a single car are removed from the results. This size constant was found by observing the largest truck that was closest to the camera and adding 20% to the size.
Haar-cascade의 확률 적 특성으로 인해 false positive는 또한 논리적으로 자동차를 포함 할 수없는 탐지를 포함 할 수 있습니다. 즉 긍정적 인 결과로 식별 된 사각형이 너무 크거나 작을 때입니다. 이러한 결과는 결과 집합을 심각하게 손상시킬 수 있습니다. 왜냐하면 영역 병합에 대해 설명 된 알고리즘은이 모든 영역을 하나의 큰 영역으로 병합하기 때문입니다. 이것을 피하기 위해 Haarascascade의 결과가 미리 사전 필터링되어 하나의 자동차를 담을 수 없을만큼 큰 영역을 결과에서 제거합니다. 이 크기 상수는 카메라에 가장 가깝고 크기에 20 %를 더한 가장 큰 트럭을 관찰함으로써 발견되었습니다.

3.2.4 Training cascade

The training of the cascade proved to be no easy task. The first necessary bit was to gather the images, then create samples based on them and finally starting the training process. The opencv traincascade utility is an improvement over its predecessor in several aspects, one of them being that traincascade allows the training process to be multithreaded, which reduces the time it takes to finish the training of the classifier. This multithreaded approach is only applied during the precalculation step however, so the overall time to train is still quite significant, resulting in hours,days and weeks of training time. [10, 11, 12]
캐스케이드 훈련은 쉬운 일이 아니었다. 첫 번째 필요한 비트는 이미지를 수집 한 다음 이미지를 기반으로 샘플을 만들고 마침내 교육 프로세스를 시작하는 것이 었습니다. opencv traincascade 유틸리티는 여러 측면에서 이전 버전보다 향상된 기능입니다. 그 중 하나는 traincascade를 사용하면 트레이닝 프로세스를 다중 스레드 할 수 있으므로 분류 프로그램을 완료하는 데 걸리는 시간이 단축됩니다. 그러나이 다중 스레드 접근법은 사전 계산 단계에서만 적용되므로 교육에 소요되는 전체 시간은 여전히 상당히 중요하므로 수 시간, 수일 및 수주의 교육 시간을 필요로합니다. [10, 11, 12]

Since the training process needs a lot of positive and negative input images, which may not always be present, then a way to circumvent this is to use a tool for the creation of such positive images.
교육 과정에는 항상 긍정적이거나 부정적인 입력 이미지가 많이 필요하기 때문에 이러한 이미지를 만들려면 도구를 사용하는 것이 좋습니다.

OpenCV built in mode allows to create more positive images with distorting the original positive image and applying a background image. However, it does not allow to do this for multiple images.
OpenCV 내장 모드는 원래의 포지티브 이미지를 왜곡하고 배경 이미지를 적용하여보다 긍정적 인 이미지를 만들 수 있습니다. 그러나 여러 이미지에 대해서는이 작업을 수행 할 수 없습니다.

By using the Perl script createsamples to apply distortions in batch and the mergevec tool [11], it is possible to create such necessary files for each positive input file and then merging the outputted files together into one input file that OpenCV can understand.
Perl 스크립트를 사용하여 일괄 처리 및 mergevec 도구 [11]에서 왜곡을 적용하면 긍정적 인 입력 파일마다 필요한 파일을 만든 다음 출력 된 파일을 OpenCV가 이해할 수있는 하나의 입력 파일로 병합 할 수 있습니다.

Another important aspect to consider is the number of positives and negatives. When executing the command to start training, it is required to enter the number of positive and negative images that will be used. Special care should be taken with these variables, since the number of positive images here denotes the number of positive images to be used on each step of the classifier training, which means that if one were to specify to use all images on every step, then at one point the training process would end in an error. This is due to the way the training process is set up, as described in section 1. The process needs to use many different images on every stage of the classification and if one were to give all to the

first stage, then there would be no images left over for the second stage, thus resulting in an error message. [10, 11, 12]
고려해야 할 또 다른 중요한 측면은 긍정과 부정의 수입니다. 훈련을 시작하기 위해 명령을 실행할 때, 사용될 긍정적이고 부정적인 이미지의 수를 입력해야합니다. 여기서 긍정적 인 이미지의 수는 분류 자 훈련의 각 단계에서 사용할 긍정적 인 이미지의 수를 나타내므로 각 변수마다 특별한주의를 기울여야합니다. 즉, 모든 단계에서 모든 이미지를 사용하도록 지정하면 어느 시점에서 교육 과정은 오류로 끝날 것입니다. 이는 섹션 1에서 설명한대로 트레이닝 프로세스가 설정되는 방식 때문입니다. 프로세스는 분류의 모든 단계에서 여러 이미지를 사용해야하며 한 단계가 모두 첫 번째 단계에 제공되는 경우 아무 것도 없을 것입니다 이미지는 두 번째 단계에서 남겨져 오류 메시지가 나타납니다. [10, 11, 12]

The training can result in many types of unwanted behaviour. Most common of these is either overtraining or undertraining of the classifier. An undertrained classifier will most likely output too many false positives, since the training process has not had time to properly determine which actually is positive and which is not. An output may look similar to image 6.
교육은 많은 유형의 원치 않는 행동을 초래할 수 있습니다. 이들 중 가장 일반적인 것은 분류 자의 과부하 또는 과부화입니다. 트레이닝 프로세스가 실제로 실제로 긍정적인지 아닌지를 적절하게 결정할 시간이 없기 때문에, 과소 평가 된 분류 기준은 오탐 (false positive)을 너무 많이 출력 할 가능성이 큽니다. 출력은 이미지 6과 비슷하게 보일 수 있습니다.

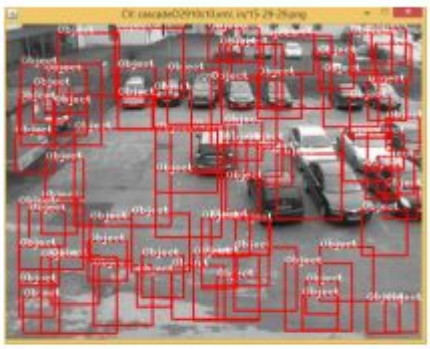

Image 6: Depiction of the results of undertrained classifier

이미지 6 : undertrained 분류기의 결과 묘사

The opposite effect may be observed if too many stages are trained, which could mean that the classification process may determine that even the positive objects in the picture are actually negative ones, resulting in an empty result set.
너무 많은 단계가 훈련되면 분류 과정에서 그림의 양의 물체도 실제로 음의 물체라고 판단하여 빈 결과 집합이 나올 수 있음을 의미 할 수 있습니다.

Fairly undefined behaviour can occur if the number of input images are too low, since the training program cannot get enough information on the actual object to be able to classify it correctly.
훈련 프로그램이 실제 객체에 대한 정보를 올바르게 분류 할 수 없기 때문에 입력 이미지의 수가 너무 적 으면 너무 정의되지 않은 동작이 발생할 수 있습니다.

One of the best results obtained in the course of this work is depicted on image 7. As one can observe, the classifier does detect some vehicles without any problems, but unfortunately also some areas of the pavement and some parts of grass are also classified as a car. Also some cars are not detected as standalone cars.
이 작업 과정에서 얻은 최상의 결과 중 하나가 이미지 7에 묘사되어 있습니다. 관찰 할 수 있듯이 분류기는 문제없이 일부 차량을 감지하지만, 불행히도 포장 도로의 일부 영역과 잔디 일부도차 로 분류됩니다 . 또한 일부 차량은 독립형 차량으로 감지되지 않습니다.

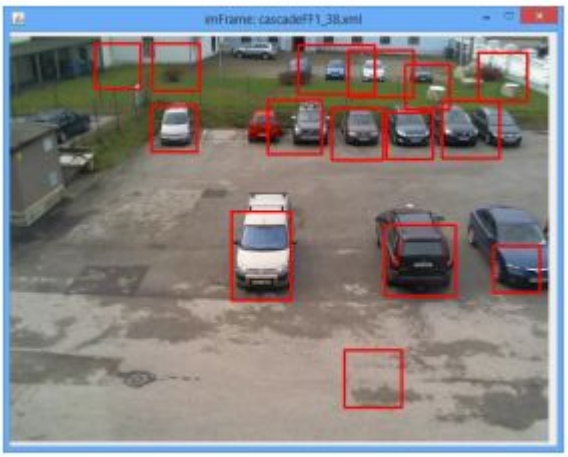

Image 7: Best solution obtained by the author

Image 7: Best solution obtained by the author
이미지 7 : 저자가 얻은 최상의 솔루션

The time taken to train the classifier to detect at this level can be measured in days and weeks, rather than hours. Since the training process is fairly probabilistic, then a lot of work did also go into testing the various parameters used in this work, from the number of input images to the subtle changes in the structuring element on the background removal, and verifying whether the output improved, decreased or remained unchanged.
이 레벨에서 분류기를 분류하는 데 걸리는 시간은 시간이 아닌 일 및 주 단위로 측정 할 수 있습니다. 훈련 과정이 매우 확률 적이기 때문에 많은 작업이 입력 이미지의 수에서 배경 제거에 대한 구조 요소의 미묘한 변화에이 작업에 사용 된 다양한 매개 변수를 테스트하고 출력 개선되었거나, 감소했거나, 변경되지 않았습니다.

For the same reason, unfortunately the author of this work was unable to produce a proper classifier, which would give minimal false positives and maximal true positives.
같은 이유로, 불행히도이 저작물의 저자는 적절한 분류 기준을 생성하지 못했기 때문에 가양성 (false positive)과 최대한의 진정한 긍정 (false positive)을 제공 할 수있었습니다.