Argus
Road Traffic Accidents Detection

Introduction

Over 1.25 million people die in a road traffic crash each year. Unless action is taken, road traffic injuries are predicted to become the fifth leading cause of death by 2030. The main reason for deaths is the long delay in reporting the accidents and the arrival of help. Our project is a system based on computer vision techniques that detect road accidents and report them in nearly real-time and allow monitoring accidents using a client-server architecture and an interactive GUI. It is noted that it will save accidents in the database for later inspection.

Egypt loses about 12 000 lives due to road traffic crashes every year. It has a road traffic fatality rate of 42 deaths per 100 000 population. The majority (48%) of those killed are passengers of four-wheelers though pedestrians also constitute a significant proportion (20%) of these fatalities.

Project Objectives and Problem Definition

Given a video taken from a surveillance/CCTV camera placed in public roads, After proper processing of the camera stream, the stream is fed to a server for car crash detection. A notification is sent to a web client which is supposed to send medical help.

I. The Model Block:

The model is built of three modules. The following is a brief description of every module, its role. The pipeline of the model is quite simple. **First,** we detect vehicles in the scene. **Second,** track the vehicles in order to focus each car, **Third**, detect collision estimation. then the final stage for each car we detect if there is a car crash.

1. Vehicles detection

The You Only Look Once (Yolo) architecture. The YOLOV3 is chosen since it achieves a good trade-off between accuracy and speed, settling in a position between YOLO and YOLOv2. YOLOV3 will be used for detecting. A transfer learning approach is used by exploiting the pre-trained weights of YOLOV3 trained on Large data sets. This is very useful since a lot of capabilities needed for training like storage and GPU power aren’t available.

2. **Vehicles tracking**

    we searched and found three methods to make tracking and implemented two of them, we implemented the Lucas Kanade tracker but we found it takes lots of processing power and time so we had to choose the Mosse tracker because Mosse runs using a correlation filter which depends on the frequency domain and using fast Fourier transform you could get the frequency to the car image fast and because Mosse filter is a correlation filter so it can track complex objects through rotations, occlusions, and other distractions, and Mosse filter produces stable correlation filters when initialized using a single frame and it’s robust to variations in lighting, scale, pose and non-rigid deformations.

3. **Crash detection**

    We need a way to measure the change in a set of frames for a specific car to see the changes in its flow so we want to calculate the optical flow to be able to describe its flow in a fast and good accuracy we have tried before Lucas Kanade optical flow but it is so slow as it depends on feature extraction of corners in the image, and also it’s a local optical flow so we will use Horn & Schunck because it is global optical flow and fast and finally the VIF descriptor is used to detect the variations of flow vectors computed by the tracker because of the very low cost and acceptable accuracy, those vectors are fed to the SVM classifier for crash prediction.

4. **Collision Estimation**

    We here consider a **new approach** by using gaming techniques to solve the crash detection module - Vif descriptor - bad accuracy because of the variety of camera angles, resolutions, lighting and situations by limiting the trackers entering the VIF descriptor.

II. Audience:

This is made mainly for the governments and Insurance Companies that want to collaborate with the governments to use it in making roads safer and to be notified by the accident as fast as possible to send the required help for the location of the accident.

III. Outputs:

- Warning: indicating a vehicle crash has occurred.
- Videos: cutting a short video when an accident occurs and save the video file in the file system at the backend.
- Database: to save records for the accident (place, date, video link) in the database.
- GUI interface: so user could access to record and view accident videos