Ontology-Based Approach and Implementation of ADAS System for Mobile Device Use While Driving

Igor Lashkov[1], Alexander Smirnov[1,2], Alexey Kashevnik[1,2(✉)], and Vladimir Parfenov[1]

[1] ITMO University, 49, Kronverkskiy av., 197101 St. Petersburg, Russian Federation
igor-lashkov@ya.ru, {smir,alexey}@iias.spb.su, parfenov@mail.ifmo.ru
[2] SPIIRAS, 39, 14th Line, 199178 St. Petersburg, Russian Federation

Abstract. Many people drive while being tired or drowsy and according to experts, many drivers fail to recognize they are in a fatigued state. The paper describes an ontology-based approach and a real-time prototype of a computer vision system for monitoring driver's dangerous behaviour patterns such as distraction, drowsiness and reduced vigilance. Our approach focuses on widely used mobile devices, such as a smartphone or a tablet, which have become an integral part of our daily life. The main components of the system consist of a front-facing camera, gyroscope sensor installed on the smartphone and various computer vision algorithms for simultaneous, real-time and non-intrusive monitoring of various visual bio-behaviours that typically characterize a driver's level of vigilance. The visual behaviours include eyelid movement (PERCLOS percentage of eye closure, eye blink rate), face orientation (face pose), and gaze movement (pupil movement). The two ontological models presented in the paper are a driver model and a vehicle model. Based on these models and the information available from cameras and sensors, the context is created that allows the system to determine dangerous situations using the driver mobile device mounted on the windshield of the vehicle. The system was tested in a simulated environment with subjects of different ethnic backgrounds, genders, ages, with/without glasses, and under different illumination conditions, and it was found very robust, reliable and accurate.

Keywords: Ontology · Context · Mobile device · ADAS systems

1 Introduction

Advanced Driver Assistance Systems (ADAS) are systems aimed to help the driver in the driving process through the making alerts for preventing dangerous situations. When designed with a safe Human-Machine Interface, they offer increased car and road safety. Safety features are designed to avoid collisions and accidents by offering technologies that provide possibilities to recognize dangerous situations beforehand. This work is an extension of [1] that aims at developing mobile assistant for Segway riders.

© Springer International Publishing Switzerland 2015
P. Klinov and D. Mouromtsev (Eds): KESW 2015, CCIS 518, pp. 117–131, 2015.
DOI: 10.1007/978-3-319-24543-0_9

The EuroFOT consortium [2] published the findings of a four-year study focused on the impact of driver assistance systems in the Europe. The €22-million (US$27.4-million) European Field Operational Test (EuroFOT) project, which began in June 2008 and involved 28 companies and organizations, was led by Aria Etemad from Ford's European Research Centre in Aachen, Germany. Almost 1000 vehicles (cars and trucks) were equipped with ADAS and more than a thousand drivers participated. Data were collected from onboard computer systems, video recordings, driver surveys, and questionnaires submitted to drivers at the start of the FOT, at the end of baseline phase, and at the end of treatment phase. The study [2] shows a decrease of safety risk up to 42% due to timely alert of the driver or an automatic adjustment of speed, and that over 90% of accidents involve driver behavior as a contributing factor.

At the modern market advanced driver assistance systems are divided into two categories: (i) manually installed mobile applications from application stores and (ii) specific hardware and software, integrated into automobiles by manufactures. Despite advanced safety features are equipped on premium and high-end vehicles models, but the majority of drivers still do not have access to these safeguards. But they can manually install such type of application in his/her mobile phone and use it while driving a vehicle.

The paper proposes an approach for mobile application development that assists the driver during the travel by providing visual and tactile feedback if an unsafe situation is predicted. The front camera facing up to the rider allows the application to determine the driver's mental states, whereas the rear camera, GPS, motion and orientation sensors to detect the external environment as well as the dynamical characteristics of the vehicle. The proposed reference model in this paper incorporates different services and algorithms. The reference model is based on the ontological knowledge representation that allows providing for ontology-based information sharing between different services in the developed system. The ontological driver and vehicle models portray them formally in terms of comprehensibility of the information systems and using these descriptions in the assisting processes. For assistive functionalities, the motion related parameters are observed and evaluated simultaneously.

The rest of the paper is structured as follows. Section 2 presents an overview of existing at the moment ADAS solutions. The proposed reference model ADAS system is presented in Section 3. Section 4 and Section 5 present ontological models of the driver and vehicle. Implementation is given in the Section 6. The results are summarized in Conclusion.

2 Related Work

CarSafe [3] is a driver safety application for Android phones that detects and alerts drivers in case of dangerous driving conditions and behavior. It uses computer vision and machine learning algorithms to monitor and detect whether the driver is tired or

distracted using the front-facing camera while at the same time tracking road conditions using the rear-facing camera. CarSafe also tries to solve the problem of processing video streams from both the front and rear cameras simultaneously by using a context-aware algorithm. This application has two main disadvantages. The first one is a lack of emotions, gestures and speech recognition. The second one is that it is not available for downloading in any application store (it has been developing for prototyping an approach).

The iOnRoad [4] is an Android- and iOS-based application that provides a range of personal driving assistance functions including augmented driving, collision warning and "black-box" like video recording. The application uses the GPS sensor, gyroscope, and video camera stream of the native mobile device for monitoring the vehicle position on the road. Application makes alerts for drivers with audio and visual cues in case of dangerous situations. The main weakness of this application is that it does not use front-facing camera for tracking driver behavior.

DriveSafe [5] is a driver safety application for iPhone-based devices that detects inattentive driving behaviors and gives corresponding feedback to drivers, scoring their driving and alerting them in case their behaviors are unsafe. It uses computer vision and pattern recognition techniques on the iPhone to assess whether the driver is drowsy or distracted using the rear-camera, the microphone, the inertial sensors and the GPS.

WalkSafe [6] is an Android-based application that aids people that walk and talk, improving the safety of pedestrian mobile phone users. WalkSafe uses the back camera of the mobile phone to detect vehicles approaching the user. It makes alerts for the user of a potentially unsafe situation. The application uses machine-learning algorithms implemented on the mobile phone to detect the front views and back views of moving vehicles. WalkSafe alerts the user of unsafe conditions using mobile phone sound and vibration.

Augmented Driving [7] is an iPhone-based application that uses the phone's built-in camera to view the road ahead. It detects the road obstacles, warning the driver of any potential hazards with voice notifications. This program provides a rather wide range of driving assistance features including vehicle headway, lane departure warning, speeding avoidance, video recording, and sound & voice output.

Driver Guard application [8] is an Android-based ADAS application that avoids accidents. It uses smartphone's camera to monitor the scene in front of the driver, detect preceding cars, estimates the distance to all preceding cars and fires an alarm when driver approaches any car dangerously. This application also shows the driver's current speed using GPS sensor and distance to the nearest car in front of the driver.

One major disadvantage of the considered applications is that they are taking into account only watching by video camera conditions on the road. The research projects CarSafe, Augmented Driving, and iOnRoad are distinguished as they provide more

significant safety features for drivers than other mobile applications mentioned above. A detailed comparison of ADAS systems is presented by authors in [9].

3 ADAS System Reference Model

The reference model of the smartphone-based ADAS is presented in Fig. 1. It consists of five main modules: mobile application, cameras, sensors, local database, cloud service, and synchronization service. Information from the device's different sensors collected by Sensor fusion component caters for estimating the various useful quantities such as the speed, the acceleration, and the location. Internal components of the mobile application are context-aware camera switching algorithm, multi-core computation planner and image processing unit. Today's smartphones do not have the capability to process video streams from both of the front and the rear cameras simultaneously. In this respect, we use a context-aware algorithm that switches between the two cameras while processing the data real-time with the goal of minimizing missed events inside.

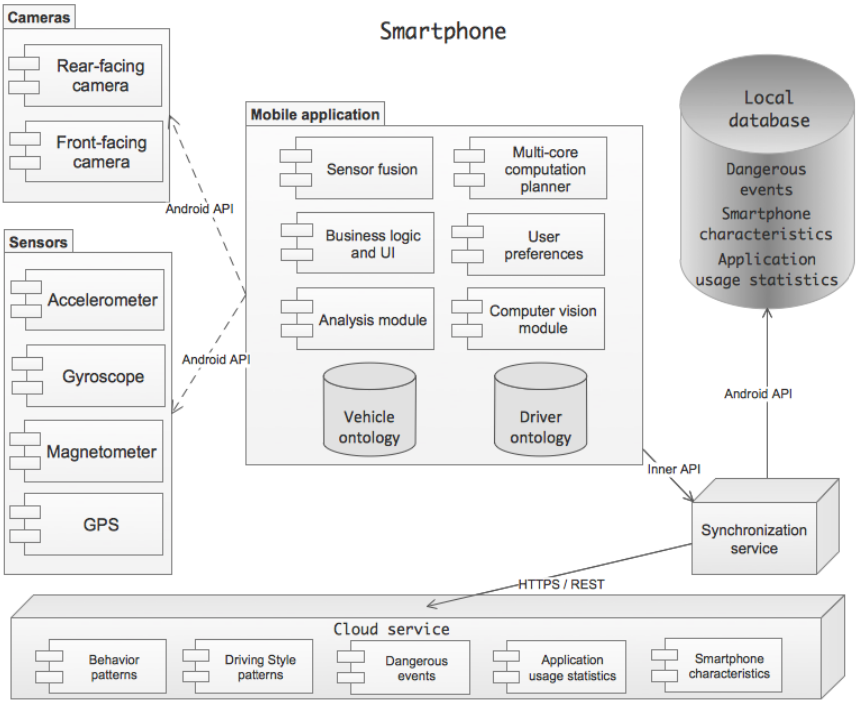

Fig. 1. Reference model of the proposed ADAS system

The image processing unit is responsible for extracting the visual features from the images taken by the rear and front cameras. The computation planner aims to effectively leverage the multi-core architecture of modern smartphones to perform heavy calculations. User preferences module allows saving and retrieving persistent key-value pairs of primitive data types. If the Internet connection is not available, local database is responsible for storing data collected from the smartphone. As soon as the Internet connection becomes available, we are ready to synchronize a local database with the cloud service. Synchronization service is a component of the assistant system responsible for managing the information flows to/from the database located on the smartphone and to/from the cloud.

Such information as smartphone characteristics, application usage statistics, and dangerous events occurred during trip is stored for using in the future. Smartphone characteristics are GPU, sensors (GPS, Accelerometer, Gyroscope, Magnetometer), cameras (front-facing / rear-facing), memory & battery capacity, and version of operation system. In addition, the cloud storage is used for keeping behaviour patterns and driving style patterns. Operations that can be carried out in the cloud storage are:

- Recognition of true and false responses due to occurrence of dangerous events.
- Matching of behavior and driving style patterns.
- Analysis and classification of driver behavior and driving style for further making recommendations for safe driving.

4 Driver Ontology Model

The system is focused on the behavioral and physiological signals acquired from the driver to assess their mental state in real-time. In the presented approach, the driver is considered as a set of mental states. Each of these states has its own particular control behavior and interstate transition probabilities. The canonical example of this type of model would be a bank of standard linear controllers (e.g., Kalman Filters plus a simple control law). Each controller has different dynamics and measurements, sequenced together with a Markov network of probabilistic transitions. The states of the model can be hierarchically organized to describe the short and long-term behaviors.

People in fatigue state exhibit certain types of visual behaviors that can easily be observed from the changes in their facial expressions and features from the eyes, head, and face. The typical visual characteristics observable on a face image of a person are the reduced alertness level that includes slow eyelid movement [10, 11], smaller degree of eye openness (or even closed), frequent nodding [12], yawning, gaze (narrowness in the line of sight), sluggish in facial expression, and sagging posture. The features that can be assessed are given as follows: PERCLOS – PERcentage of CLOSure of eyelid, eye blink time, eye-blinking rate, eye gaze, pupil movement, eyelid movement, postures, and head pose. Visual behaviors observable from the changes in facial features listed above are: eyes are opened or closed, facing to the left, facing to the right, facing forwards, gaze concentration towards the road, gaze concentration not towards the road, dilated pupils, not dilated pupils.

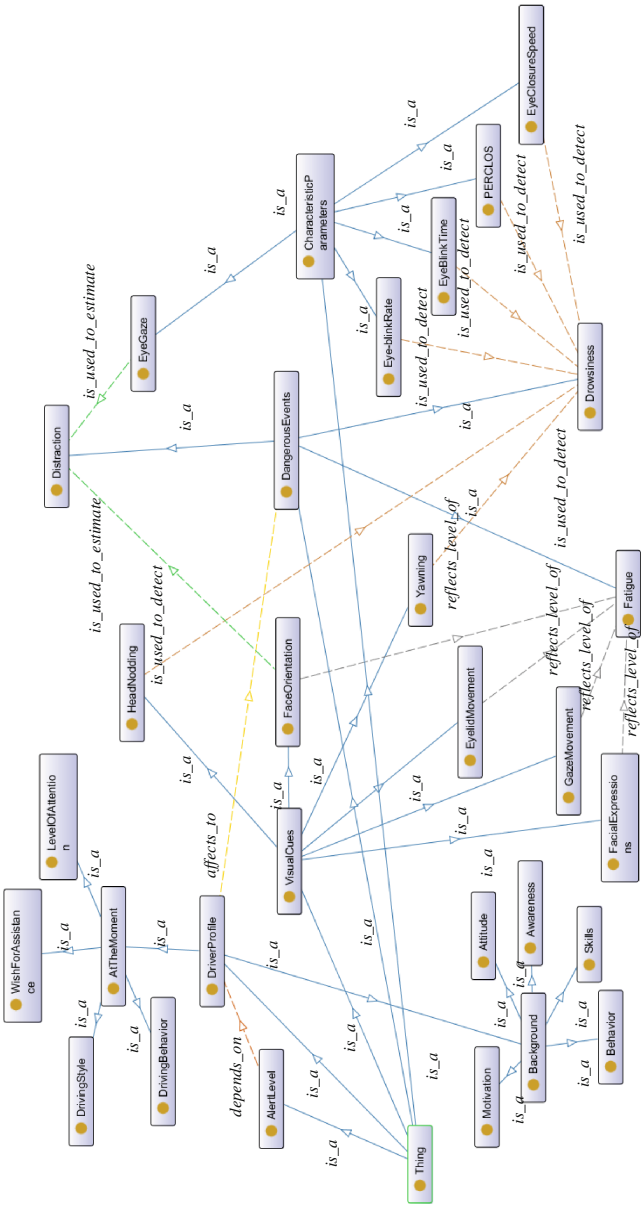

Fig. 2. The Driver ontology model

The developed driver ontology (Fig. 2) includes these visual cues and visual behaviors and determines relationships between them. It consists of the five main top level classes: "AlertLevel" (warning level), "DangerousEvents" (commonly occurring dangerous driving events), "CharacteristicParameters" (visual characteristics observable from the image of a person), "DriverProfile" (a driver profile that reflects the certain personal characteristics) and "VisualCues" (the visual cues on which we focus to detect dangerous events). Each driver has a profile (class "DriverProfile") that reflects personal characteristics. Driver profile consists of background (class "Background") and real-time (class "AtTheMoment") context. Background context in its turn includes five elements that underpin a driver safety culture – behavior (class "Behavior"), attitude (class "Attitude), awareness (class "Awareness"), motivation (class "Motivation") and skills (class "Skills"). These classes are associated with each other with the relationship "is_a". On the other hand we need to consider the real-time data (class "AtTheMoment"). It consists of a wish for assistance (class "WishForAssistance"), driving style (class "DrivingStyle"), level of attention (class "LevelOfAttention") and driving behavior (class "DrivingBehavior"). The relationship between these classes is "is_a". System alert level (class "AlertLevel") depends on driver profile.

Class "AlertLevel" is associated with the class "DriverProfile" with the relationship "depends_on". The basic characteristic parameters (class "CharacteristicParameters") that typically characterize driver's state are PERCLOS (class "PERCLOS"), eye-blink rate (class "Eye-BlinkRate"), eye closure speed (class "EyeClosureSpeed"), eye-blinking time (class "EyBlinkTime"), eye gaze (class "EyeGaze"), pupillary state (class "PupillaryState), yawning (class "Yawning") and nodding level (class "HeadNodding"). The relationship between these classes is "is_a". We infer the dangerous driver behaviours (class "DangerousEvents") such as drowsiness (class "Drowsiness"), distraction (class "Distraction") and fatigue (class "Fatigue"). In the proposed ontology, the corresponding classes ("Drowsiness", "Distraction" and "Fatigue") are associated with the class "DangerousEvents" with the relationship "is_a".At the same time, face orientation and eye gaze are used to detect distraction (classes "FaceOrientation" and "EyeGaze" are associated with the class "Distraction" with the relationship "is_used_to_estimate"). PERCLOS, eye-blink rate, eye closure speed, eye blink time, yawning and nodding level are used to recognize drowsiness. (Classes "PERCLOS", "Eye-blinkRate", "EyeClosureSpeed", "EyeBlinkTime", "Yawning" and "HeadNodding" are associated with the class "Drowsiness" with the relationship "is_used_to_detect"). And finally, eyelid movement, face orientation, gaze movement and facial expressions reflect level of fatigue (classes "EyelidMovement", "FaceOrientation", "GazeMovement" and "FacialExpressions" are associated with the class "Fatigue" with the relationship "reflects_level_of"). Open or closed eyes are a good indicator of fatigue. (Property "EyeState" is associated with the class "Fatigue" with the relationship "is_an_indicator_of").

5 Vehicle Ontology Model

The vehicle drivers are faced with a multitude of road hazards and an increasing number of distractions (e.g. music, phone calls, smartphone texting and browsing, advertising information on the road, and etc.). In the presented approach, the following five of the most commonly occurring dangerous driving events are addressed: drowsy driving, vigilance decrement, inattentive driving, tailgating, ignoring blind spots.

The developed Vehicle ontology model (Fig. 3) consists of five main top level classes: "DangerousEvents" (dangerous events that can occur during driving), "Sensors" (embedded sensors on the phone), "Cameras" (built-in front-facing and rear-facing cameras), "VehicleBehaviorParameters" (vehicle behavior parameters) and "RoadParameters" (parameters that characterize the road the vehicle moves). At the same time, classes "VehicleBehaviorParameters", "Sensors" and "RoadParameters" are used to recognize hazards (class "DangerousEvents"). Classes "Vehicle BehaviorParameters", "Sensors" and "RoadParameters" are associated with the class "DangerousEvents" with the relationship "is_used_to_recognize"). The class "DangerousEvents" is classified as "Tailgating" (driver should maintain a minimum safe distance with the vehicle or moving object ahead) and "IgnoringBlindSpots" (executing lane changes safely also requires a driver to check blind spots before proceeding). In the proposed ontology, the corresponding classes ("Tailgating" and "DangerousEvents") are associated with the class "DangerousEvents" with the relationship "is_a".

Most Android-powered devices have built-in sensors (class "Sensors") such as accelerometer (class "Accelerometer"), gyroscope (class "Gyroscope"), magnetometer (class "Magnetometer") and GPS (class "GPS). Corresponding classes ("Accelerometer", "Magnetometer", "Gyroscope" and "GPS") are associated with the class "Sensors" with the relationship "is_a". But also Android framework includes support of cameras (class "Cameras") and camera features available on device, allowing to capture pictures and videos in applications. We aim to work with front-facing ("FrontFacingCamera") and rear-facing ("RearFacingCamera") cameras. These classes are associated with "Camera" by relationship "is_a". Rear-facing camera pictures help us to recognize behavior parameters such as turn (class "Turn"), vehicle headway (class "VehicleHeadway), trajectory (class "Trajectory") and lane position (class "LanePosition"). These classes are associated with each other with the relationship "is_used_to_recognize". The class "VehicleBehaviorParameters" includes such parameters such as the speed (class "Speed"), acceleration (class "Acceleration"), lane position (class "LanePosition"), trajectory (class "Trajectory), vehicle turns (class "Turn") and vehicle headway (class "VehicleHeadway"). They are associated with the class "VehicleBehaviorParameters" with the relationship "is_a". The GPS sensor (class "GPS") and accelerometer (class "Accelerometer") are used to estimate the position, acceleration (class "Acceleration") and the speed (class "Speed"). The classes "Acceleration" and "Speed" are associated with the classes "GPS" and

"Accelerometer" with the relationship "is_used_to_estimate". Besides, inertial sensors (classes "Acceletometer", "Magnetometer" and "Gyroscope") are used for trajectory detection (class "Trajectory) and these are associated with the relationship "is_used_to_detect". The vehicle turns (class "Turn) are detected by observing the significant changes in the direction from the time-series data of the GPS (class "GPS") positions. The class "GPS" is associated with the class "Turn" with the relationship "is_used_to_detect". The last top level class is "RoadParameters". It contains lane markers (class "LaneMarkers"), road conditions (class "RoadConditions"), obstacles (class "Obstacles") and road signs (class "RoadSigns). Classes "LaneMarkers", "RoadConditions", "Obstacles" and "RoadSigns" are associated with the class "Road" with the relationship "is_a". Finally, the road parameters (class "Road parameters"), sensors (class "Sensors") and the vehicle behaviour parameters ("VehicleBehaviorParameters") are used to recognize dangerous events. These classes are associated with each other with the relationship "are_used_to_recognize".

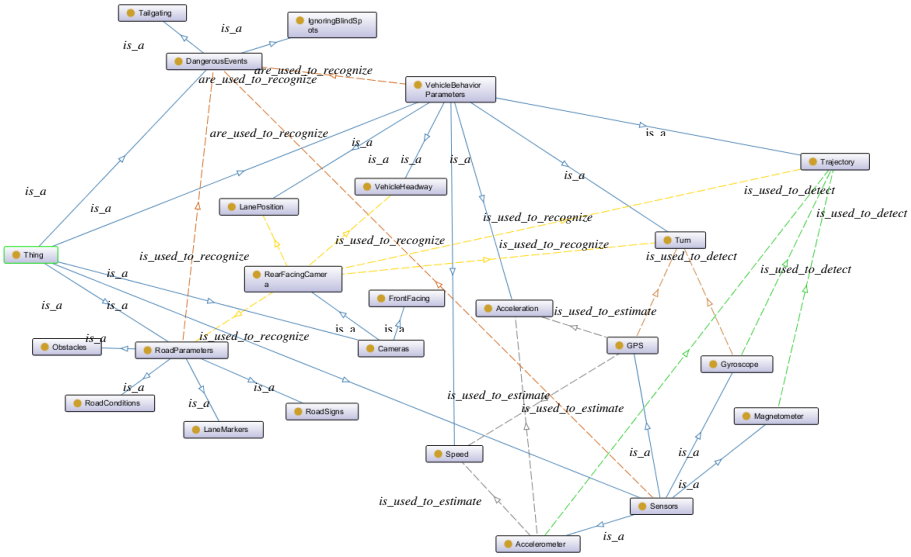

Fig. 3. Vehicle ontology model

The developed ontologies describing the behavior models of driver and vehicle allow the ADAS system to accurately interpret data from multiple sources and verify the mappings between entities as we come to the unsafe factors (driver visual state, road, vehicle parameters) extracted from cameras and embedded sensors on the phone. Dangerous driving events are modeled using vehicle and driver ontologies and we focus on five of the most commonly occurring one:

- Careless lane change;
- Tailgating;

- Drowsy driving;
- Inattentive driving;
- Fatigue.

Each driver has his own skills, motivation, attitude and qualifications that we should consider in monitoring and detecting dangerous events occurring throughout the trip. That is why the driver and vehicle models are rather closely related. For example, executing lane changes safely requires a driver to check blind spots before proceeding. The driver does this by looking in the side and front mirrors of the car to check for unexpected vehicles. The system should be capable of recognising the head position of the driver using the phone's front camera, allowing the app to ensure the appropriate mirror checks are performed before each lane change. Lane changing itself can be detected using the rear camera and inertial sensors, as described above.

6 Implementation

The implementation of proposed ADAS system has been developed for Android-based mobile device. Evaluation has been done for the multi-core Samsung Galaxy S4 Android smartphone. The driver and vehicle classification pipelines, which represent the most computationally demanding modules, are written in C and C++ based on the open source computer vision library (OpenCV) [13] (see Fig. 4) and interfaced with Java using JNI wrappers. Other architectural components (dangerous driving event engine, context-driven camera switching, and multi-core computation planner) are implemented using pure Java. For the image recognition based on OpenCV library for each frame received in video sequence the Haar cascade classifier is called to find faces or eyes (see Fig. 5). Functions return a vector of rectangles where each rectangle contains the detected object. Each rectangle is presented by OpenCV Rect structure (see Table 1).

The image processing module has input/output (I/O) signals that let you transmit image taken with the camera, receive appropriate driver recommendations, and alerts later on to prevent any unsafe situations. Both driver and vehicle ontologies are involved in the work of computer vision module and analysis module. At first, camera image (Bitmap) is scanned to find and identify objects and their position that are relevant to the situation. In the next step, vectors of rectangles that contain the detected objects are passed to the Analysis module for further processing. With the help of predefined types of rules that cover unsafe situations and scenarios, it runs search-match calculation. If the procedure returns true, the system will make an appropriate alert, otherwise this event will be ignored. For example, if the system determines that the driver is likely to enter a dangerous drowsy state, it will make an audible alert for a driver. Quantitative parameters helping to identify dangerous driving events are presented as follows.

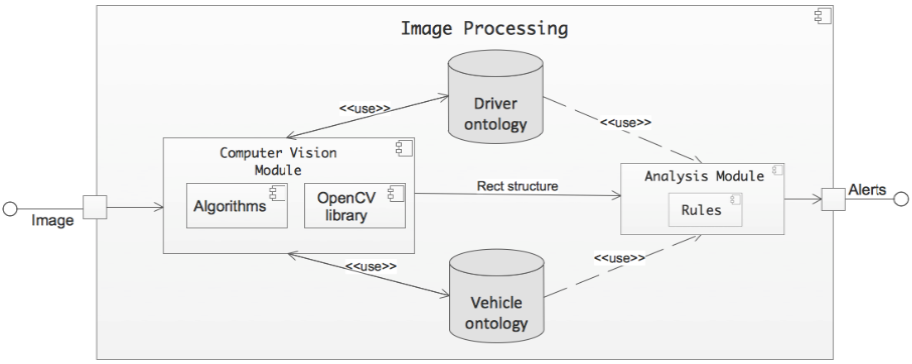

Fig. 4. Image processing module implementation

Fig. 5. Example of finding faces and eyes using OpenCV library

Table 1. OpenCV output data for face and eyes recognition

Face				Left eye				Right eye				Time
X	Y	W	H	X	Y	W	H	X	Y	W	H	
243	108	372	372	429	190	163	124	266	190	163	124	119
243	108	372	372	429	190	163	124	266	190	163	124	60
243	108	372	372	429	190	163	124	266	190	163	124	59

- *PERCLOS.* Regarding PERCLOS, we consider, for each driver, the proportion of time within one minute that eyes are at least 80% closed. The permissible limit is equal to 30%.
- *Eye-blink rate.* We compute for each driver the eye-blink rate as the average number of blinks per minute (the range from 8 to 21 blinks per minute).
- *Face direction.* We classify the direction of the driver's face and recognize four face related categories that include: (1) no face is present; or the driver's face is either; (2) facing forwards, towards the road; (3) facing to the left, if angle greater than 15° rotation relative to facing directly forward; (4) facing to the right, if rotation angle greater that 15° but this time to the right.

These visual parameters provide data to infer drowsiness, distraction and fatigue states. We use a rule-based approach to evaluate our project and to increase the expressiveness of the underlying knowledge. We compose our dangerous events as a collection of simpler events based on IF / THEN rules that provide an outcome. We have defined five rules that provide an output score. They are defined as follows:

- If PECLOS parameter exceeds a threshold of 30%, then the system declares the driver "drowsy".
- If the driver is not facing forward for longer than three seconds while the car is moving forward, then dangerous driving event is inferred.
- If there is no a head turn corresponding to a car turning event, then the driver did not check that the road is clear before turning – as a result, a dangerous event is inferred.
- If appropriate mirror checks are not performed before lane change event, then the driver did not check blinds spots before proceeding – as a result, a dangerous event is inferred.
- If the eye-blink rate of driver misses the above-mentioned range, then dangerous event is signaled.

There exist other rules definition that have not been considered in this paper. Built rules are the result of combining quantitative parameters and ontologies.

Detection Classification. Inferring the direction of the driver's face is divided into two steps: (1) detection classification and (2) direction classification. Images from the front camera are scanned to find the relative position of the driver's face. The overall image is provided to a classifier that determines if a face is present, and if so the face direction is classified.

Facial Features. Facial analysis includes a number of processing steps that attempt to detect or track the face, to locate characteristic of facial regions such as eyes, pupils if speaking more precisely and nose, to extract and follow the movement of facial features, such as characteristic points in these regions or model facial gestures using anatomic information about the face.

Drowsy state. Drowsy driving, a combination of sleepiness and driving, has become a worldwide problem that often leads to tragic accidents and outcomes. The smartphone's front camera should be able to monitor the prolonged and frequent

blinks indicative of micro sleep. Existing research findings have shown that the percentage of closure of eyelid (a.k.a PERCLOS) is an effective indicator to evaluate the driver's drowsiness. A measure of drowsiness, PERCLOS, was generated and associated with degradation in driving performance in a simulated roadway environment.

PERCLOS is the proportion of time that the driver's eyes are 80 to 100% closed over a specified interval. Its validity was tested, along with other new technologies, and it was shown to be an accurate drowsiness detector.

Although PERCLOS is considered the best among other indicators in drowsiness detection, the original PERCLOS is not suitable for smart phones, since smart phones could not analyze every frame accurately or effectively. However, in an analogy to PERCLOS, we can detect fatigue by analyzing a series of states of eyes, classified by Neural Network to either open or closed. The states of eyes are necessary to simulate not only PERCLOS, but also the frequency of wink and the time of continuous closing eyes, which are all using in the driver alert mechanism. The faster we can acquire image of eyes and analysis, the less error between simulation and reality of the indicators.

The algorithm focuses on driver's eyes recognition, implemented in OpenCV computer vision library, classified by Neural Network, and then used to discrete-approximate several indicators including PERCLOS, blink time and blink rate to evaluate the fatigue level of the driver.

Distraction state. Maintaining eye contact with the road is fundamental to safe driving. By using the front camera of the phone, the head position of the driver can be used to determine an inattentive driving behavior when the driver is not looking at the road ahead.

Two types of inattentive driving are monitored by our approach. In the first case, the output of the face direction classifier is tracked. If the driver's face is not facing forward for longer than three seconds [14] while the vehicle is moving forward (i.e., while a positive speed is reported by the available smartphone sensors) and not turning as reported by the turn detector (also reported by car classification pipeline) then a dangerous driving event is inferred. However, you should keep in mind, that it's not a constant value and this parameter depends on various factors (e.g. vehicle speed, acceleration). In the second case, we monitor the turn detector. Each time a turn is detected the historical output of the face direction classifier is checked. If there is no a head turn corresponding to a car turning event then the driver did not check that the road is clear before turning – as a result, a dangerous event is inferred.

Persons's gaze. The direction of a person's gaze is determined by two factors: face orientation (face pose) and eye orientation (eye gaze). Face pose determines the global direction of the gaze, while eye gaze determines the local direction of the gaze. Global gaze and local gaze together determine the final gaze of the person. According to these two aspects of gaze information, video-based gaze estimation approaches can be divided into a head-based approach, an ocular-based approach, and a combined head- and eye-based approach.

7 Conclusion

The paper presents an ontology-based approach and its implementation for ADAS system development. Such systems can be used by drivers while driving vehicles using personal mobile devices. The paper contains an overview of existing current ADAS solutions. The proposed reference model ADAS system is based on ontological knowledge representation and cloud computing technology that allows to provide information exchange between drivers and mobile devices and then take this information into account while discovering dangerous events. Implementation is based on open source computer vision library (OpenCV) that allows to detect objects using smartphone camera. The testing of system shows that it is robust, reliable and accurate and provides the driver alerts in case of dangerous situations beforehand.

Acknowledgements. The presented results are part of the research carried out within the project funded by grants # 13-07-00336, 13-07-12095, 13-01-00286 of the Russian Foundation for Basic Research. This work was partially financially supported by Government of Russian Federation, Grant 074-U01.

References

1. Smirnov, A., Kashevnik, A., Lashkov, I., Hashimoto, N., Boyali, A.: Smartphone-based two-wheeled self-balancing vehicles rider assistant. In: 17th Conference of the Open Innovations Association FRUCT, Yaroslavl, Russia, pp. 201–209 (2015)
2. European Field Operational Test on Activity Safety Systems. http://www.eurofot-ip.eu
3. You, C., Lane, N., Chen, F., Wang, R., Chen, Z., Bao, T., Montes-de-Oca, M., Cheng, Y., Lin, M., Torresani, L., Campbell, A.: CarSafe app: alerting drowsy and distracted drivers using dual cameras on smartphones. In: 11th International Conference on Mobile Systems, Applications, and Services, Taipei, Taiwan, pp. 461–462 (2013)
4. iOnRoad official website. http://www.ionroad.com/
5. Bergasa, L., Almería, D., Almazán, J., Yebes, J., Arroyo, R.: DriveSafe: an app for alerting inattentive drivers and scoring driving behaviors. In: IEEE Intelligent Vehicles Symposium, Dearborn, MI, USA (2014)
6. Wang, T., Cardone, G., Corradi, A., Torresani, L., Campbell, A.: WalkSafe: a pedestrian safety app for mobile phone users who walk and talk while crossing roads. In: 14th International Workshop on Mobile Computing Systems and Applications, No. 5 (2012)
7. Augmented Driving website, iOS application. http://www.imaginyze.com/
8. Driver Guard application, Google Play. https://play.google.com/store/apps/details?id=com.badrit.cv.vehicledetect&hl=ru
9. Smirnov, A., Lashkov, I.: State-of-the-art analysis of available advanced driver assistance systems. In: 17th Conference of the Open Innovations Association FRUCT, Yaroslavl, Russia, pp. 345–349 (2015)
10. Wierville, W.: Overview of research on driver drowsiness definition and driver drowsiness detection, ESV, Munich (1994)

11. Dinges, D., Mallis, M., Maislin, G., Powell, J.W.: Evaluation of techniques for ocular measurement as an index of fatigue and the basis for alertness management. Department of Transportation Highway Safety Publication, 808 762 (1998)
12. Anon. Proximity array sensing system: head position monitor/metric. Advanced Safety Concepts, Inc., Sante Fe (1998)
13. OpenCV library. http://opencv.org
14. Breakthrough Research on Real-World Driver Behavior Released. http://www.nhtsa.gov/Driving+Safety/Distracted+Driving+at+Distraction.gov/Breakthrough+Research+on+Real-World+Driver+Behavior+Released