Ensuring Safety in Augmented Reality from Trade-off Between Immersion and Situation Awareness

Conference Paper · October 2018

CITATIONS
0

READS
92

7 authors, including:

Jinki Jung
Korea Research Institute of Ships and Ocean Engineering
19 PUBLICATIONS **69** CITATIONS
SEE PROFILE

Hyeopwoo Lee
Korea Advanced Institute of Science and Technology
6 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Jeehye Choi
Korea Advanced Institute of Science and Technology
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Abhilasha Nanda
Korea Advanced Institute of Science and Technology
4 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Ensuring Safety in Augmented Reality from Trade-off Between Immersion and Situation Awareness

Jinki Jung*
Korea Research Institute of Ships and Ocean Engineering
Hyeopwoo Lee†, Jeehye Choi‡, Abhilasha Nanda§
KAIST
Uwe Gruenefeld¶, Tim Claudius Stratmann‖
University of Oldenburg
OFFIS - Institute for IT
Wilko Heuten**
OFFIS - Institute for IT

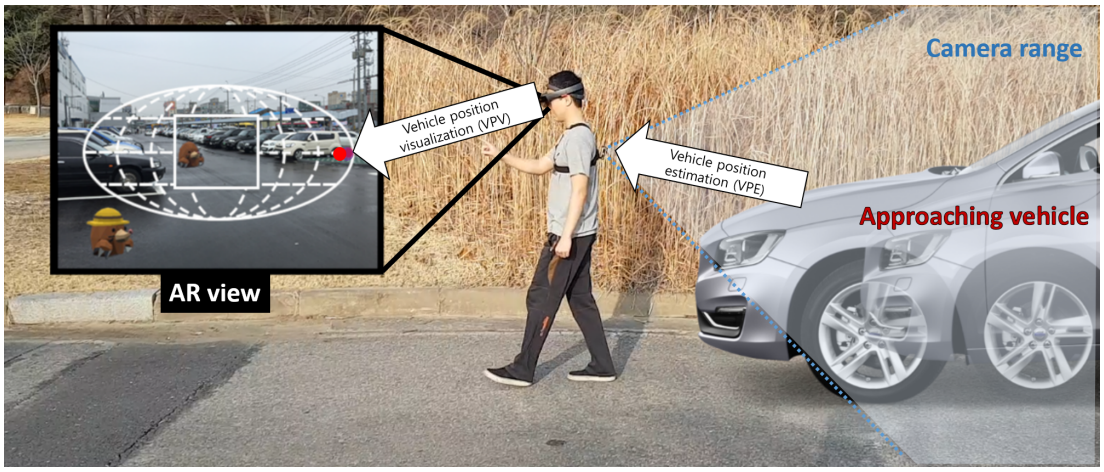


Fig. 1: Overview of proposed third-party component for AR safety with two functions: vehicle position estimation (VPE) and vehicle position visualization (VPV). VPE estimates the 3D relative positions of approaching vehicles using a RGB camera. The estimated positions are then transferred to an out-of-view visualization of VPV (here with EyeSee360) for collision warning.

Abstract

Although the mobility and emerging technology of augmented reality (AR) have brought significant entertainment and convenience in everyday life, the use of AR is becoming a social problem as the accidents caused by a shortage of situation awareness due to an immersion of AR are increasing. In this paper, we address the trade-off between immersion and situation awareness as the fundamental factor of the AR-related accidents. As a solution against the trade-off, we propose a third-party component that prevents pedestrian-vehicle accidents in a traffic environment based on vehicle position estimation (VPE) and vehicle position visualization (VPV). From a RGB image sequence, VPE efficiently estimates the relative 3D position between a user and a car using generated convolutional neural network (CNN) model with a region-of-interest based scheme. VPV shows the estimated car position as a dot using an out-of-view object visualization method to alert the user from possible collisions. The VPE experiment with 16 combinations of parameters showed that the InceptionV3 model, fine-tuned on activated images yields the best performance with a root mean squared error of 0.34 m in 2.1 ms. The user study of VPV showed the inversely proportional relationship between the immersion controlled by the difficulty of the AR game and the frequency of situation awareness in both quantitatively and qualitatively. Additional VPV experiment assessing two out-of-view object visualization methods (EyeSee360 and Radar) showed no significant effect on the participants' activity, while EyeSee360 yielded faster responses and Radar engendered participants' preference on average. Our field study demonstrated an integration of VPE and VPV which has potentials for safety-ensured immersion when the proposed component is used for AR in daily uses. We expect that when the proposed component is developed enough to be used in real world, it will contribute to the safety-ensured AR, as well as to the population of AR.

Index Terms: Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Mixed / augmented reality; Human-centered computing—Visualization—Visualization design and evaluation methods

*e-mail: your.jinki.jung@gmail.com
†e-mail: leehyeopwoo@kaist.ac.kr
‡e-mail: jeehye0324@kaist.ac.kr
§e-mail: ananda@kaist.ac.kr
¶e-mail: uwe.gruenefeld@uol.de
‖e-mail: tim.claudius.stratmann@uol.de
**e-mail: wilko.heuten@offis.de

1 Introduction

Augmented Reality(AR) has a property of connecting virtual and real worlds by superimposing a virtual existence as if it is in reality [4]. According to the reality-virtuality continuum suggested by Milgram et al. [27], an AR user has mixed presences from both sides and should be able to interact with the virtual and real worlds simultaneously. Unlike VR, which requires full immersion only in the virtual world [5], an AR user has to juggle two tasks at the same time: immersion in virtual content (represented on an AR display) and situation awareness of the real environment. The use of AR causes distraction from real world because AR requires highly intensive concentration [2, 8]. When such distraction occurs in some context, e.g., a traffic environment, a user is exposed to risks with a high probability [20, 37, 38]. Accident reports of Pokémon Go

showed that most pedestrian accidents are preposterous and would rarely occur to people with normal cognitive ability: bumping into signs, poles and other people [8]; collisions between a car and a utility pole [21]; pedestrian-vehicle collisions due to users crossing a street despite a red traffic light [3]; and walking in front of a car without engagement of the car [2]. As a fundamental factor of the accidents due to the use of AR, here we define *a trade-off between immersion and situation awareness* which is caused by the limitation of cognitive loads of human. Under this trade-off, the more the user's immersion is strengthened, the lower the cognitive loads for situation awareness which leads to higher probabilities of accidents. Unfortunately as the fidelity with which AR gets better in the future, the accidents due to the trade-off will increase more. However assume that a component that notifies the risks by performing the situation awareness on behalf of the user exists, then the user will be able to allocate more cognitive loads to the AR experiences. This is defined as *safety-ensured immersion of AR*.

We propose a third-party component for safety-ensured AR to prevent pedestrian-vehicle accidents by estimating 3D position of vehicles nearby and visualizing the vehicles for collision warning. The third-party component is defined as a complete system including both hardware and software apart from AR. Proposed component aims to detect approaching vehicles accurately and promptly and transfer the risk, i.e., distance to the vehicle, to user effectively. More precisely, the proposed component aims to estimate the three-dimensional positions of vehicles behind the user within very short time (vehicle position estimation, VPE for short) and visualize the positions in a 2D screen coordinate for out-of-view object visualization (vehicle position visualization, VPV for short) as shown in Fig. 1. Although VPE requires high accuracy, the time performance is also crucial because the situation must be transferred before the accident occurs. VPV should be efficient to transfer the positions of multiple potential risks, where we used visual feedback representing two-dimensional data instead of auditory [38, 44] or haptic [12,38,44]. For the sake of simplicity, our study visualizes the relative positions between cars and the user as a potential risk instead of classifying the context [26] or providing instant notification of the risk [20,47]. The major risk in our paper is the one specific case in pedestrian-vehicle accidents [2, 3], which a pedestrian (and AR user) crosses street without noticing a car that go around the street corner. Three assumptions are given to put the scenario concretely: a vehicle comes from behind the user; a RGB sensor of the proposed component is attached to the back of the user; the RGB sensor looks toward the backside of the user. In this scenario, if the component is able to detect vehicles approaching in the backward direction, the component will be able to inform the user in advance of the risk.

Through our paper we investigated the validity of VPE and VPV through two independent experiments:

- VPE experiment aims to assess configurations of the deep learning model that estimated the three-dimensional relative position from given 2D ROI (region of interest) of a vehicle based on the ground truth data.
- VPV experiment aims to analyze the correlation between the degree of immersion in the AR game and the situation awareness of the real world and evaluates two out-of-view object visualization methods for collision warning: EyeSee360 [14] and a radar-like graph (Radar) [10, 30].

Ideally, our experiments should be performed in real traffic environment, but we used the virtual environment in both VPE and VPV environments in consideration of the safety of participants [24, 28, 39]. The use of virtual environments has additional advantages such as use of ground truth data for VPE evaluation [7,24], ease of repetition of specific situations [34], and independent evaluation of VPE and VPV with an identical situation. We performed empirical and solid evaluations for the two functions of the proposed component based on ground truth data without dependency. In summary, the contributions of this paper are as follows: 1) a novel solution of employing a third-party component with VPE and VPV to ensure the safety of a pedestrian AR user; 2) real-time ROI-based 3D position estimation of vehicle using deep learning; 3) demonstration of the trade-off between immersion and situation awareness; and 4) quantitative and qualitative evaluations of out-of-view visualization methods for head-mounted AR. The paper is organized as follows. Section 2 introduces backgrounds of our study. Section 3 and 4 describes method and experiment of VPE and VPV, respectively. Section 5 presents a field study including VPE and VPV integration. Section 6 report and discuss the obtained results of experiments in Section 3, 4, and 5. Finally, Section 7 concludes remarks.

2 Background

In this section, we review related literature on collision warning, vehicle position estimation, and vehicle position visualization.

Collision warning Risks due to the distraction addressed in the earlier literature have been addressed in the use of smartphones [29, 32, 37]. Several studies have presented accident prevention methods for these risks [38, 44, 46]. Uchida et al. proposed a system that alerts users by predicting potential collisions based on a Markov chain process using the wireless signal and sensor data of mobile phones [44]. Wang et al. presented an application integrating vehicle detection and pedestrian alert components to ensure the safety of mobile phone users [46]. Their application employs two RGB cameras (front and back) on a phone to capture nearby traffic situations and produces a vibration alert when a vehicle is detected within the predefined region by the Haar-like detector. Unlike the previous two studies, which mainly addressed the technical elements of risk detection, Straughn et al. conducted a user study to compare the effectiveness of sensory formats for warnings (i.e., tactile and auditory) in arbitrary simulated risky situations [38]. They also visualized suggestions (e.g., steering direction) to avoid risk and indicate the occurrence of the potential risk. In our evaluation of the collision warning, we applied the distance and time of user response used by Straughn et al [38]. For pedestrian safety with AR, Jain et al. proposed using the GPS and inertial sensor of the mobile phone to determine whether the user is in a position safe from vehicle collision (in-street detection) [19]. Although they were motivated to ensure the safety of pedestrians using AR, in-street detection was more suited for assuring the safety of pedestrians using mobile phones rather than an AR-specific problem [20]. There is also a difference in exposure length: Jain et al. employed very short exposure for the collision warning (i.e., imminent collision), while ours required longer (i.e., predictable collision through tracking).

Vehicle position estimation (VPE) Vehicle position estimation with monocular RGB images have been extensively researched [1, 22, 31]. Kampelmuhler et al. [22] recently studied RGB-based vehicle location estimation using real data containing the position and velocity of a vehicle relative to a moving car obtained from a LIDAR sensor, simple neural networks, and spatiotemporal information. Ali and Hussein [1] described two important methods: distance estimation for detecting the distance of preceding vehicles with a monocular camera and vehicle position detection for finding the relative vehicle position to the road. Park et al. [31] proposed a range estimation method that uses monocular RGB images and serves as a collision warning system. This range estimation method was evaluated with video clips recorded in both highway and urban traffic environments. Meanwhile, Chabot [6] proposed using deep convolutional neural networks (CNNs) for joint 2D and 3D vehicle analyses of monocular image. They proposed a multi-task CNN (Deep MANTA) for accurately estimating the 2D vehicle bounding boxes, vehicle part coordinates, part visibility and 3D template for each detection. Depth estimation can be considered as a way to

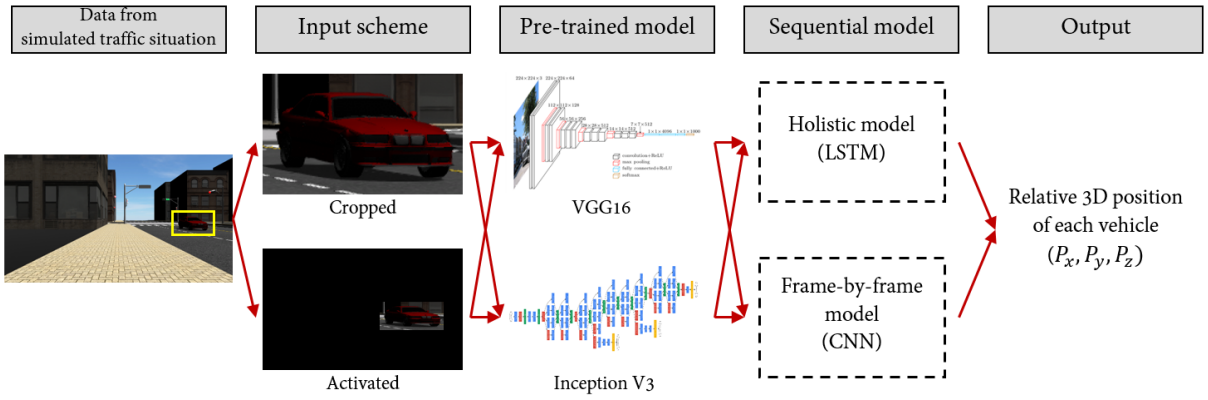


Fig. 2: Network architecture and parameters for VPE

estimate the distance for given ROIs. Laina et al. [23] showed the state of the art for vision-based depth estimation considering fully connected layer as a 1 by 1 convolution layer to predict pixel-wise depth estimation. Their result showed very high accuracy but it is not suitable for AR application due to its time performance; 1 sec for the feed forward time, which does not meet to the real-time performance.

Vehicle position visualization (VPV) As a preliminary study on out-of-view visualization for AR, Gruenefeld et al. [13] adapted Arrow, Halo, and Wedge for head-mounted AR. Their results showed that all of these techniques are applicable to head-mounted devices, but their approach was limited to 90° in front of the user. Therefore, Gruenefeld et al. developed a new visualization technique called EyeSee360 [14] that was inspired by EdgeRadar [15]. EyeSee360 utilizes the users periphery by showing radar-like visualization. It allows multiple out-of-view objects located 360° around the user to be visualized. Their choice of using color as an attribute, which can be somewhat controversial, was based on the comparison of color vs. size vs. color+size, which results the use of color as the most effective attribute [14].

3 Vehicle Position Estimation (VPE)

Goal of VPE is to estimate a vehicles 3D position (P_x, P_y, P_z) relative to the AR users position from a vehicle's 2D location in RGB image sequence using a deep network. We propose the use of the rectangular ROI of a vehicle as an input rather than the use of a whole RGB image [9], [25] that hardly satisfies time performance (more than one second), required in our problem (around 1 second over 1280x720 resolution). Here an RGB image is able to contain multiple vehicles. We apply the use of 3D position as output to cope with complex road environment such as uphill roads where elevation exists. We assumed that the vehicle ROIs were given by vehicle detection method like [33] while we used ground truth areas given by simulation in the VPE experiment. In this section we examine the possible input scheme of the network and its accuracy and time performance using simulated traffic image sequences. From the foundation of Thorpe et al. [42] which indicates that with 30 km/s of vehicle speed and general human reaction time of 150 ms, the system should propose risk information under error range of 1.2 m, we hypothesize the accuracy of VPE should be under error range of 1.2 m.

3.1 Method

Because our input consists of RGB images and ROIs, we based our learning tasks on the CNN. A sequential model was used to train consecutive frames. We used transfer learning, which is the process of getting features from a pre-trained model on a very large dataset and applying it to improve the task performance of another model. We used two of the most well-known deep network models for image recognition in frame-by-frame 3D VPE: VGG16 [36] and InceptionV3 [41]. A fully connected layer was added to the last layer for each of VGG16 and InceptionV3, which were optimized according to the mean squared error of the 3D relative vehicle position. We used two kinds of training schemes: training models from scratch and fine-tuning them. For training the models from scratch, all layers were trained from the random initialization. With fine-tuning, layers were trained according to weights trained from ImageNet challenges. The sequential model had both input and output sequences with spatiotemporal information. The model was trained on a sequence of RGB images, and the result was in form of output sequences. Because our goal was to estimate the vehicle position, we set both the input and output sequences to have the same dimensions and trained the models with long short-term memory (LSTM), which is a kind of recurrent neural network where the neurons are replaced by memory cells. For training, we added a single LSTM layer on the final fully connected layer of the CNN model.

3.1.1 Network input scheme

Our input dataset included bounding boxes of existing cars captured by the RGB camera. Some schemes exist for feeding this to the network as input. Two input schemes were used to train the 3D vehicle positions. All input images for the pre-trained network without data augmentation were normalized to 224 × 224 pixels. The bounding box information of the vehicles were used to crop the vehicles from each image and train the dataset containing local information of the vehicles. In the activated scheme, the bounding box information of the vehicles was used to obtain the region of interest (vehicle). Rather than cropping the image, however, we activated the vehicle region (remaining RGB value for the vehicle) and set the background to zero to pass on local and global information of the vehicle.

3.2 Experiment

Data generated from a virtual simulation was used to train and test deep network models. We split the data generated from the virtual simulation into training and testing sets to measure the estimation. The relative position of the vehicle was estimated from consecutive frames of the situation, which was assumed to be captured from the rear view RGB camera of the AR user. The experiment was based on data generated from the movement of cars on the surrounding roads while the AR user was moving forward in a virtual urban environment. In the ground truth data, the relative position was represented as a differential 3D vector of the users camera and the front bumper of a vehicle. The relative positions of the AR user and vehicles were estimated from the 2D RGB view based on deep learning.

pre-trained model	VGG16								InceptionV3							
model candidate	cr-cnn-s	cr-cnn-f	cr-L-s	cr-L-f	a-cnn-s	a-cnn-f	a-L-s	a-L-f	cr-cnn-s	cr-cnn-f	cr-L-s	cr-L-f	a-cnn-s	a-cnn-f	a-L-s	a-L-f
RMSE (m)	0.69	0.75	5.50	1.55	0.47	0.48	5.89	5.44	0.92	0.42	1.22	1.17	1.34	0.34	0.64	0.45
Test time (ms/image)	3.2	3.3	3.5	3.7	3.2	5.2	3.6	5.6	2.1	2.1	2.2	2.1	2.1	2.1	2.2	2.1

Table 1: Vehicle position estimation result table
cr: cropped, a: activated; cnn: frame-by-frame model, L: sequential lstm model; s: train from scratch, f: fine tuning from trained weights

3.2.1 Simulation environment

Under the assumption that realistic traffic situations can be simulated as virtual environments, we implemented a 3D virtual traffic simulation as the input of VPE, which is available online [1]. The simulation was implemented by using Unity 3D Engine based on Volkhins implementation [45] of the Intelligent Driver Model (IDM) [43]. The virtual environment consisted of functional entities of roads, buildings, vehicles, and traffic signals on a flat ground. The entire virtual environment is generated intersection-wise consisting of a total of 5×5 intersections, and any two intersections were connected by two-lane roads in each direction (orthogonal to each other) for a total of four lanes. The distance between all the intersections was 112 m and each road had a 4 meter wide sidewalk in the direction of the building. Cube-shaped buildings are randomly placed within a 104 m $\times$ 104 m rectangular area surrounded by four intersections. Each intersection has a crosswalk of 4 meters wide and 16 meters long in each of the four directions. Virtual cars with a length of 4.66 meters, a width of 1.96 meters and a height of 1.44 meters were spawned from one of the 20 end of roads. A series of roads chosen at random from the starting point was given to the car as a route. Each car had a speed of up to 30 km/h and adjusted its speed by interacting with the distance to the front car, distance to the intersection, and traffic lights as defined in the IDM [43]. Because our simulation was based on right-hand traffic, a car turned to the right direction at an intersection regardless of the traffic light signal. Our simulator had the functions of storing the 3D position of the vehicle in every frame as a scenario file and reproducing the stored scenario. The appearance of the vehicles in our simulator only varied in color.

3.2.2 Data generation

Data were generated based on vehicles that could be seen in the rear view of an AR user in a virtual urban environment. We simulated a total of N scenarios and recorded M consecutive frames per simulation environment instead of random vehicle respawning due to the sequential model. In the situation, the AR user walked along the sidewalk at a constant speed. Vehicles respawned on roads next to the sidewalk, and the front-most vehicle was collected as a sample. We generated the 2D bounding box and 3D relative position to the AR user of each vehicle for the application of various training schemes to the VPE model. The recording camera was placed at a height similar to that of a person (1.7 m). The FOV of the camera in the virtual simulation was 40°. For the parameters in the clipping planes, we set the near plane to 0.3 and far plane to 1000. Data were captured at a resolution of 1280×720 pixels. To verify VPE, 1932 training data and 483 test data were generated (8:2). It took 5 hours to generate all the data samples. In this experiment, we set M = 40 and the capture rate to 50 fps. Only the vehicle that was continuously captured for 40 frames without being occluded was selected for the training data.

3.2.3 Experimental environment

We used stochastic gradient descent to optimize the mean squared loss for the 3D relative position and set the learning rate to 0.0001. In the sequential model, M consecutive frames were learned in each time step. For the frame-by-frame model, learning was performed with a batch size of M. Based on the pre-trained network, input scheme, sequential model, and training scheme, we experimented with 16 deep networks. We trained 16 network models with 20 epochs for fine-tuning and 30 epochs for scratch. VPE was on the Ubuntu 16.04 system with CPU i7 with 32GB RAM on GPU GTX 1080 Ti. It took 10 h to train models in average. These were implemented on Tensorflow with the backend of Keras.

[1] https://github.com/VirtualityForSafety/RoadTrafficSimulation3D

3.2.4 Measurement

The model input was a sequence of RGB images and the output was a relative position vector(P_x, P_y, P_z). The measurement was based on the magnitude (root mean squared error distance) of the differential position vector between the ground truth data and output of the VPE model.

3.3 Result

Table 1 presents the root mean squared error of the 3D position relative to the AR user for 16 model combinations. For the pre-trained models, InceptionV3 performed better than VGG16 (2.60 m vs 0.81 m). The sequential model performed worse than the frame-by-frame model (2.73 m vs 0.68 m). Fine-tuning showed a lower error rate than training from scratch (1.33 m vs 2.08 m). The activated scheme produced improvement in most of the models except VGG16 with the sequential model. If we exclude VGG16 with the sequential model, the activated scheme better estimated the vehicle position than the cropped scheme (0.62 m vs 0.86 m). VGG16 with the sequential model performed the worst with an error of around 5 m. InceptionV3 with the frame-by-frame model, fine-tuning, and the activated scheme showed the lowest error rate for the test set at 0.3379 m. On average, it took 3 ms to process a single sample.

4 Vehicle position visualization (VPV)

VPV aims to transfer the detected vehicles and their distances from the real world to a two-dimensional screen coordinate in a head-mounted display. We used an out-of-view visualization method to allow the user immerse in AR content but to be aware of the surrounding situation simultaneously. Our hypothesis of VPV experiment is that the relationship between immersion and situation awareness is an inverse proportion. An AR game "Shoot-a-mole" and the virtual traffic environment of VPE were employed to simulate immersion and situation awareness, respectively. Additionally, we conducted the comparison of two out-of-view visualization methods: EyeSee360 [14] and Radar [10, 30]. To our best knowledge, EyeSee360 is the one and only out-of-view object visualization for head-mounted AR. Radar is a radar-like display based on a 2D circular coordinate and was chosen by its uses in computer games, which is the most common technique when it comes to the visualization of out-of-view objects in first-person perspective. The comparison was designed to evaluate the efficiency of information transfer according to the visualization method.

4.1 Method

In VPV, we have mapped the detected car location to a colored dot. The color of the dot presents the Euclidean distance of the object motivated by the heat map [16], e.g., blue represents far distance, while red represents proximity. The color was determined according

to the following equation: [14]:

$$D_{scale} = D_{relative}/D_{max}$$
$$Color_{dot} = Color_{Red} \times (1 - D_{scale}) + Color_{Blue} \times D_{scale} \quad (1)$$
$$where\ D_{relative} : relative\ distance\ of\ the\ object,$$
$$D_{max} : maximum\ distance\ of\ the\ visualization\ can\ represent$$

EyeSee360 projects the 3D position information onto a 2D ellipsoidal coordinate. Fig. 3(a) shows the inner rectangle of EyeSee360 representing the FOV of the display, and the area outside of the rectangle representing out-of-view positions in 360 degrees. Each dotted line represents a 45° section of the users view. The horizontal line expresses the altitude of the object. The vertical curved lines represent the horizontal direction of the object. In Fig. 3(a), for example, the blue dot represents an object almost directly behind the user, while the purple circle on the second-most left line of the y-axis represents an object almost 90° to the left of the users front view. Radar PPI(Plan position indicator)-scope represents 2D positions in a circular coordinate by projecting from top view to bottom direction. The center of the coordinate is the origin representing the user's location, and the upper direction represents the user's looking direction. The center of Radar is the origin of the actual screen $P_{(0,0)}$. Let L be the actual radius of the outer circle on the screen. Plotting the detected object eliminates the y-axis value from the given vector $V_{(x,y,z)}$ and then rotates it with the y-axis angle of the look-at vector as follows:

$$V_{modifi} = V_{(x,z)}R \quad (2)$$

$$where\ R = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \theta : y-angle\ of\ look-at\ vector$$

Then, the modified vector has to be fitted to the coordinates of Radar. Let D be the maximum relative distance set up for Radar and $P_{(x,y)}$ be the position display on Radar.

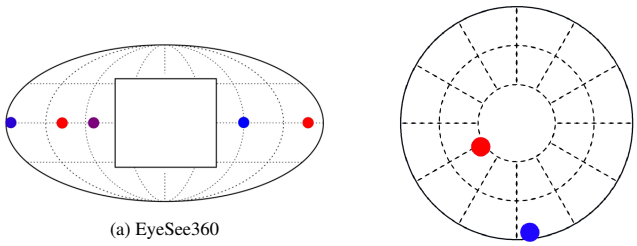
$$P_{(x,y)} = \begin{cases} V_{modifi} \times \frac{L}{D}, & if \left\| V_{modified} \right\| < D \\ \hat{V_{modifi}} \times L, & otherwise \end{cases} \quad (3)$$

$$where\ \hat{V_{modifi}} = \frac{V_{modifi}}{\left\| V_{modifi} \right\|}$$

If the object is located within D, then it should be inside the outer circle. Therefore, we divide D by the vector and multiply it by L to fit it into the coordinate. Conversely, if the magnitude of the vector is bigger than D, we plot the vector on the boundary of the outer circle. In Fig. 3(b), for example, the blue dot represents an object located behind the pedestrian at a distance of about D while the red dot is a nearby object located in the 8 o'clock direction from the pedestrian.

4.2 Experiment design

We used a counterbalanced measure design; EyeSee360/Radar and Radar/EyeSee360 groups had 21 and 22 participants respectively. We applied two mutually exclusive modes in the game: game and evasion. In game mode, a user was able to play the game but was exposed to vehicle collision. To motivate the immersion, the participants were asked to shoot as many moles as possible to get a high score. In evasion mode, a user could evade the collision but could not play the game. The user switched the mode by pressing the evasion mode button. We designed that the moment pressing the evasion mode button (i.e., evasion mode entrance) is the moment



(a) EyeSee360

(b) Radar

Fig. 3: Two visualization methods in VPV

that the user starts to evacuate. To make the participants get used to VPV setup, we prepared a four-stage tutorial before the real experiment: 1) a virtual traffic situation assumed to be the real world and virtual accident experience; 2) EyeSee360 interface and cognitive mapping of vehicles; 3) Radar interface and cognitive mapping of vehicles; and 4) test play of the AR game. We did not consider any artificially-generated accidents in this experiment due to the extremely small area of the crossing area, which was designed for collisions, compared to all the path along sidewalks. We assumed that it was not realistic that an accident would occur more than twice during 9 min of movement, which could be a significant negative factor to AR immersion during repetitive sessions. Instead, we made participants experience an accident in the tutorial and asked them to pay attention when playing the AR game. VPV utilized a HTC Vive and controllers as the hardware interface. The FOV of the HMD was 100° horizontally and 110° vertically. The desktop environment used in the experiments had an Intel I7 dual core 3.40 GHz with 8 GB of memory and an NVIDIA GeForce GTX 970 graphics card. The desktop environment used Windows 10 (64 bit) as the operating system.

4.3 Simulation environment

We used the identical traffic environment used in VPE. Two virtual cameras were used for a user: an observation camera for VPE that was placed and facing the user's backside and a viewpoint camera for visualization. The observation camera was calibrated with the RGB camera used in VPE and the field of view of the camera (60°) used to capture ground truth vehicle positions that were supposed to be given by VPE. We used one fixed scenario using the reproducing function of our simulator for all the conditions including the tutorial. The duration of a single session was 9 min (walking along the sidewalk with a constant velocity of 5 km/h), and the total number of frames was 47,563. The users action (pressing button) and vehicle information (relative position and speed) captured by the simulated VPE camera were automatically logged. The log showed that the VPE camera captured more than one vehicle in 5400, 4932, and 5176 frames for levels 1, 2, and 3, respectively. For Radar, we used 10 m for D_{max}.

4.4 AR game

A simple AR game Shoot-a-mole was used to control the degree of immersion. A trigger and a touch pad of a Vive controller were used for the shooting and the evasion mode respectively. A 3D text was given to state the evasion mode, as shown in Fig. 7. As the player moved around, two moles were simultaneously generated at random positions, within a semicircle having a radius of 10 m around the front of the player. The spawn rate of the moles increased with the level. We made variations of moles' appearance and their rewards for entertainment. Each mole stayed on the screen for 5 sec; one for ascending, three for staying, and one for descending. The player was able to aim a mole by gazing slightly upwards to it in order to

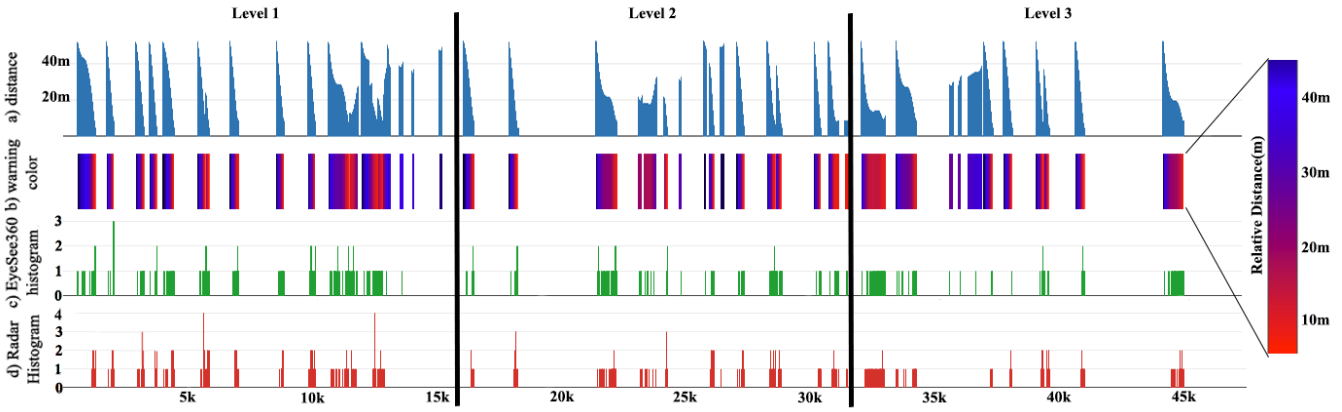


Fig. 4: Sequential visualization of approaching vehicles and entrance into evasion mode: (a) bar graph of the distance from the user to the closest vehicle; (b) solid color graph that is visualized to the user; (c) histogram of evasion mode entrance with EyeSee360; (d) histogram of evasion mode entrance with Radar

hit the mole with the gravity applied ball. If the ball hit the mole, it disappeared with particle effects, and the player gained a point. We designed three levels of game difficulty to enforce immersion. Levels 1, 2, and 3 spawned 88, 134, and 280 moles, respectively.

4.5 Participant

We recruited 43 participants (M = 23.95, SD = 3.52) aged between 19 and 33. All participants were undergraduate and graduate students (32 male and 11 female) and paid about $18 for their participation. Among these participants, only two had no prior experience with AR. For the statistics on the use of AR platforms, 40 participants had experience with mobile phones, 13 participants with a desktop/laptop, nine participants with a tablet, and five participants with a head-mounted AR display. A total of 31 participants had experienced a mobile AR content such as Pokémon Go. Three participants had experiences of near-miss or actual accidents while playing the game.

4.6 Procedure

Participants volunteered through the university intranet. We asked participants to choose their available time and fill in a prequestionnaire by using a Google form. The survey asked participants to give their demographic data before being assigned to the experiment. At the beginning, participants listened to a brief introductory presentation for about 10 min. They learned the objective of the experiment, how to interpret Eyesee360 and Radar, and how to play the game. We encouraged the participants to engage in the AR game by posting the rank of the people who received the highest scores. Next, they equipped the VR equipment and took part in a 10 min tutorial, where they could practice with the controller to play the game and evade car collisions. They also had some time to get used to the visualizations of EyeSee360 and Radar. After the tutorial, the real experiment started. For the first 9 min, the participants played the game with either EyeSee360 or Radar. After a 3 min break, they played the game again but with the other visualization method. After the experiment, the participants filled a user survey.

4.7 Measurement

In our VPV experiment, we recorded participants AR game activity logs for both the degree of immersion and situation awareness. Specifically, the number of caught moles for each stage was recorded because it was the major factor for measuring the game immersion. For situation awareness, we logged the statistics on evasion mode entrance (i.e., the moment when the user pressed the evasion mode button). We only considered valid data for the statistical analysis on evasion mode: the moment the button was pressed when more than one car was visible with VPV. For the quantitative evaluation of the VPV methods, the relative distance of a car and its approximated time of collision at the evasion mode entrance were used. We measured the approximated time to collision by dividing the distance with the vehicle speed when evasion mode was triggered. We used NASA Raw-TLX to compare the workloads of AR immersion and traffic situation awareness [17]. We also asked participants to answer a subjective evaluation questionnaire with 12 questions in total (described in the Appendix) to assess the usability. Participants answered according to a five-point Likert scale. In our questionnaire, the first six questions were used to compare the subject workloads of the immersion versus situation awareness. The latter six questions were used for the subjective evaluation of EyeSee360 and Radar as visualization methods. We also performed the analyses of top and bottom two boxes that combine the percentage of positive/negative respondents of the Likert scale.

4.8 Result

As a comparative evaluation of the simulated immersion under all conditions (n= 43), the users' activity logs were submitted to a 2 $\times$ 3 mixed-factorial analysis of variance (ANOVA). The visualization methods served as the between-subjects variable, and the level of the game (from 1 to 3) served as the within-subjects variable. In the statistical analysis of the number of caught moles, ANOVA detected that the level had a significant main effect, $F(2,40) = 521.21$, $p < 0.001$, but there was no effect of the visualization method($F(2,40) = 0.69$, $p < 0.405$) and visualization method by level ($F(2,40) = 0.01$, $p < 0.987$). Fig. 5(a) shows a bar chart of the number of caught moles. Statistics on the number of caught moles for EyeSee360 were M = 66, SD = 10.22(level 1), M = 112.55, SD = 12.39(level 2), and M = 152.32, SD = 26(level 3). Statistics on the number of caught moles for Radar were M = 67.60, SD = 9.42(level 1), M = 114.11, SD = 13.12(level 2), and M = 154.65, SD = 25.83(level 3). For the situation awareness, the total sum of evasion mode entrances per level were 332, 207, 156 (EyeSee360) and 434, 264, 202 (Radar) as shown in Fig. 5 (b).

When the distances for evasion mode entrance with EyeSee360 and Radar were compared, both the visualization method ($F(2,40) = 13.83$, $p < 0.001$) and level ($F(2,40) = 15.54$, $p < 0.001$) were detected to have significant main effects. No significant main effect was detected for the visualization method by level ($F(2,40) = 4.34$, $p < 0.013$).

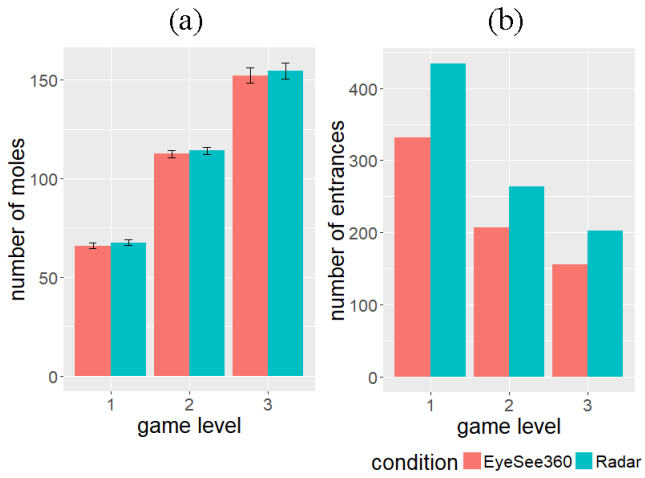


Fig. 5: Tradeoff between immersion and situation awareness (per level): a) number of caught moles in the AR game; b) total sum of evasion mode entrance events

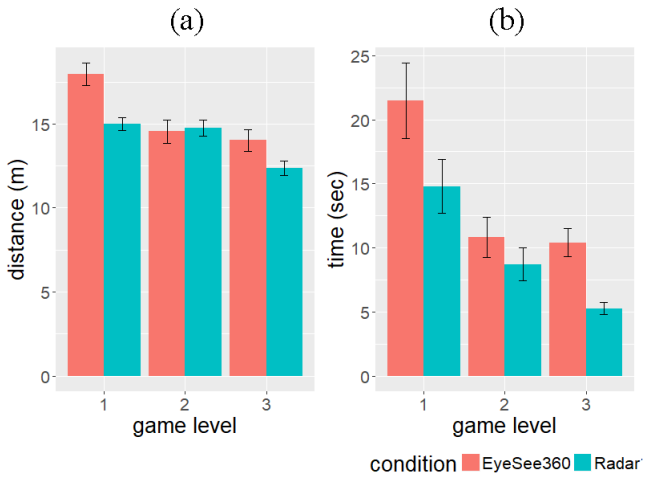


Fig. 6: Results for the EyeSee360 and Radar comparison (per level): a) distance at evasion mode entrance; b) time to collision at evasion mode entrance

As shown in Fig. 6(a), the statistics on the distance(in meters) at evasion mode entrance for EyeSee360 were M = 17.95, SD = 12.10(level 1), M = 14.54, SD = 9.84(level 2), and M = 14.02, SD = 8.14(level 3). Statistics on the distance at Evasion mode entrance for Radar were M = 14.99, SD = 8.38(level 1), M = 14.77, SD = 7.86(level 2), and M = 12.35, SD = 6.26(level 3). Overall, the average evasion mode entrance distances of EyeSee360 and Radar were 16.05 (SD = 10.80) and 14.33 (SD = 7.86), respectively. For the time-to-collision at the evasion mode entrances of EyeSee360 and Radar, ANOVA revealed that the visualization method($F(2,40) = 16.81$, $p < 0.001$) and level($F(2,40) = 26.18$, $p < 0.001$) had significant main effects, while the visualization method by level did not ($F(2,40) = 1.17$, $p < 0.309$). As shown in Fig. 6(b), the statistics on the time-to-collision (in seconds) at the evasion mode entrance for EyeSee360 were M = 21.45, SD = 36.88(level 1), M = 10.82, SD = 22.93(level 2), and M = 10.39, SD = 19.80(level 3). The statistics on the time-to-collision at the evasion mode entrance for Radar were M = 14.78, SD = 29.85(level 1), M = 8.71, SD = 20.47(level 2), and M = 5.26, SD = 9.94(level 3). Fig. 4 presents the sequential graphs of the users' action and vehicle information. Overall, the average time-to-collision values of EyeSee360 and Radar were 13 (SD = 25.84) and 8.41 (SD = 19.58), respectively. For the NASA Raw-TLX [17] average scores, AR immersion scored 51.47 and traffic situation awareness scored 51.32. Both values indicate an acceptable workload. A t-test revealed no significant difference between the AR immersion (M = 51.47, SD = 12.79) and traffic situation awareness (M = 51.32, SD = 14.13) conditions; $t(42) = 0.05$, $p = 0.957$. Table 2 presents the results of the usability study as a qualitative evaluation of VPV. Five participants reported VR sickness after the VPV experiment.

5 Field Study

To validate the actual applicability of the proposed component, we conducted a VPE and VPV integration and a VPE test with the real traffic data as a field study. The integrated VPE and VPV was operated in the order as follows: the camera captures backward scene sequentially. For every frame the object detector ([33]) detects vehicles resulting 2D ROIs. VPE then estimates the 3D position of the vehicle from the recent 40 consecutive frames and the corresponding ROIs. The estimated 3D positions are then directly transformed to the coordinate by the equation of EyeSee360 or Radar. VPV finally visualizes the positions as colored dots using Equation 1. We had tested VPE with the real dataset while the model was trained by the virtual dataset in the same approach as in [35]. A Zed stereo camera was used to capture two RGB images of the scene with the resolution of 1280×720 at 30 frames-per-second capture rate. The depth image with the identical resolution was then calculated from the stereo image using the Zed's depth estimation software. We applied the vehicle detection [33] to get the ROI of a vehicle which was also used to the depth image. Distance from the predicted 3D position of a vehicle was considered as a depth value on the stereo camera. In Fig. 8, RMSE on real dataset was given. Unlike in virtual dataset, the sequential model with the activated scheme on InceptionV3 showed the best performance (3.6 m).

6 Discussion

Vehicle position estimation Our hypothesis of VPE experiment was fully approved as the deep learning model with InceptionV3, frame-by-frame, fine-tuned showed the best performance with a mean squared error of 0.3379 m with 2.1 ms, which resulted much lesser than 1.2 m criteria. In our sequential models, we used a simple architecture with only one LSTM layer added at the end of the network. VGG16 with the LSTM layer performed worse than our other deep CNN models because of its lack of complexity. However, InceptionV3 with the LSTM layer performed significantly better. The InceptionV3 model performed better than VGG16 due to its model complexity. Experiments on both the cropped and activated input schemes suggested that almost all the models trained with activated input data has a lower root mean squared error than models trained with the cropped input data. CNN models seemed to learn better features for relative position estimation because the activated input data had both local and global information of the 2D image.

Vehicle position visualization Interestingly, both the quantitative and qualitative evaluations showed that the participants' situation awareness (number of PRESS actions for evasion mode) and the simulated immersion (number of spawned moles per level) tended to be inversely proportional even though both tasks had similar workloads, which complied with the VPV hypothesis. As the level of the game increased, the number of evasion mode entrances and situation awareness performance decreased. Moreover, the results of the distance and time-to-collision were inversely proportional to the level. Because the frequency of situation awareness and response time(distance) are considered to be critical factors for the

Table 2: Statistics on VPV usability study responses (n=43)

Category	immersion vs. situation awareness						EyeSee360 vs. Radar					
Number	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
Average point	4.21	4.33	4.33	3.84	4.30	2.88	2.60	4.23	1.93	4.35	2.05	4.21
Std. of point	0.91	0.92	0.64	0.75	0.94	1.03	0.76	0.57	0.91	0.81	1.02	0.86
Top two boxes (%)	86.05	88.37	90.70	72.09	86.05	27.91	11.63	93.02	4.65	93.02	9.30	81.40
Bottom two boxes (%)	9.30	9.30	0.00	4.65	9.30	41.86	53.49	0.00	79.07	6.98	72.09	4.65

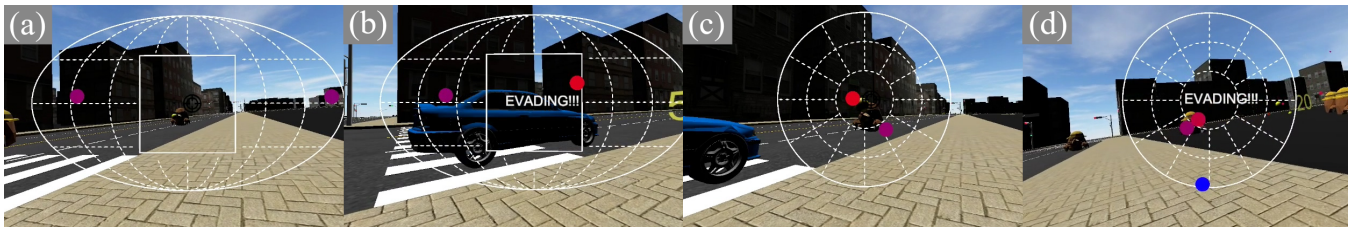


Fig. 7: Screenshots for comparison of the two visualization methods: EyeSee360 (a and b) and Radar (c and d). Approaching two vehicles are visualized as two colored points in (a) and (c). The "EVADING!!!" text is displayed when the user presses the button for evasion mode ((b) and (d)).

possibility of an accident [40], the experiment demonstrated that strengthening immersion by increasing the difficulty of the AR game inhibits the user's situation awareness, and thus increases the collision probability. In the comparisons of out-of-view visualization methods, even though there was a significant main effect from the use of EyeSee360 and Radar, no significant difference was found between them for the distance and time-to-collision. This tendency was caused by the equality of the collision representation at the end: the color. In other words the difference in coordinate systems for vehicle visualization was not a significant factor in the perception of the user in our experimental setup. In the usability study, participants tended to prefer Radar over EyeSee360 (see Q7 - Q12 in Table 2). Because EyeSee360 uses angle-wise visualization and Radar uses distance-wise visualization, Radar appears to be more effective for perceiving potential collisions. However, note that the slight difference in the distance of the two visualization methods (avg. 1.72 m) yielded a fairly significant time difference (avg. 4.59 s) in the time-to-collision. Even though we just used approximation of time-to-collision without considering the acceleration, an earlier response to the warning had a greater impact on the time for evacuation. In conclusion although Radar is a user-friendly interface, EyeSee360 is better in terms of safety.

Field study Field study confirmed the possibility of real AR scenario through the integration of VPE and VPV with VPE test on real data. In the real case, unlike the virtual environment, InceptionV3 with sequential model and activated scheme showed the best RMSE value (3.6 m). The reason for lower accuracy with the actual data was the learning model that only learned with the virtual data generated by the limited viewpoints. In addition, the sequential model with activated scheme showed better than frame by frame and cropped scheme. The cropped scheme seemed to capture mainly the texture information as a feature in estimating the position because it gives only the local information of the car. Consequently, it showed lower accuracy due to the texture difference between virtual and real. In the case of the activated scheme, however, since the global information of the car was also given including 2D position of a vehicle, it showed better result on the real dataset compared to the cropped scheme. For sequential model, the transfer learning with spatial information was not working on virtual to real so that temporal information was useful to estimate the position of a vehicle. The average time performance of VPE (10.53 ms) satisfied the real-time requirement by itself and also with the full integration with the vehicle detection/tracking [33] which takes less than 10 ms. However, this time performance was based on the use of GPUs so that the proposed approach has the limitation on the hardware performance, which is difficult to satisfy with current mobile technology. If hardware such as GPUs becomes smaller and lighter in the future, the proposed third-party component can be applied as a standalone system to ensure safety.

Linking the field study to our accident scenario, under the context where the user is about to cross the crosswalk, the VPV visualization will be able to warn the car that it can corner towards the user. Currently, VPE predicts the relative position only, but potential collision prediction is also possible if the virtual world generates an accident and uses it as a training label. The complete immersion into safety-ensured AR will be possible if VPE can accurately calculate the probability of a potential collision from the various car-approaching situations and VPV enables filtered visualization w.r.t this probability.

Limitations Our research has limitations on situation visualization, sensor, accident range, and the use of virtual data. It is obvious that, in the aspect of situation awareness, visualizing all the traffic around the user will interfere with the immersion. The relative position per se is not directly related to the one of the causes of the accident, depending on the context (e.g., a parked car and a pedestrian). If the VPE can predict the probability of accidents rather than just positions, it will be able to alert real dangerous situations by filtering with some criteria. In the case of the RGB image sensor we used, it was limited to its FOV, so it could only recognize the risk of the user's backward direction. Using multiple instances of the proposed component or a 360-degree camera [18] or performing context awareness in conjunction with other sensors around the user [44] will be able to perceive the situation in a wider area. In spite of the various reported cases of pedestrian-vehicle accidents, the situation and data generated by the simulations were very limited because our assumptions about the accident were very specific. In contrast to the fact that the problems we have raised occur in real environments, the VPE and VPV experiments that we performed employed only in virtual environments. As the requirements of the VPE have been verified through this experiment, the following experiments should be performed on VPE learning and testing with actual traffic data. VPV also needs to measure the degree of interference to immersion through the experiment using actual AR display.

pre-trained model	VGG16								InceptionV3							
model candidate	cr-cnn-s	cr-cnn-f	cr-L-s	cr-L-f	a-cnn-s	a-cnn-f	a-L-s	a-L-f	cr-cnn-s	cr-cnn-f	cr-L-s	cr-L-f	a-cnn-s	a-cnn-f	a-L-s	a-L-f
RMSE (m)	12.35	27.57	11.23	11.853	7.13	8.13	7.65	6.87	14.08	15.22	9.83	9.88	10.58	8.81	3.64	3.62
Test time (ms/image)	3.4	3.7	3.1	3.2	3.6	3.5	3.7	3.5	2.2	2.5	2.9	2.0	2.2	2.2	2.3	2.5

Table 3: Vehicle position estimation result table for real dataset in our field study
cr: cropped, a: activated; cnn: frame-by-frame model, L: sequential lstm model; s: train from scratch, f: fine tuning from trained weights

Fig. 8: RGB (a) and depth (b) data captured from Zed stereo camera. Each red box is the ROI of vehicle given by YOLO detector [33].

7 Conclusion

We propose a third-party component with VPE and VPV functionalities for potential collision estimation and potential collision warning to ensure AR safety. Our experiments based on synthetic data demonstrated that the best configuration of VPE estimates the 3D position of a car in real time with an accuracy of 0.3379 m and VPV transfers the position throughout an out-of-view visualization method while the user is playing the AR game. The following field study based on the integration of VPE and VPV and the test with real traffic environment showed practical applicability of the proposed component. We believe that, when the VPE is able to prioritize visualization of the potential risks, which is our primary future work, the proposed component will enable full immersion into AR by ensuring safety. In addition, we plan to conduct the VPE experiment with a dataset collected in real traffic environment (e.g., KITTI dataset [11]). After that, a feasibility study in the real world with moving vehicles should be performed with an optical see-through head-mounted device that focuses on measuring the visual clutter and interference of AR. We expect that if the functions of the proposed component mature and become a compact and solid solution, it will significantly contribute to the safety of users, and thus popularization of AR. Moreover, if the risk warning, which is limited by visualization in our study, utilizes other senses such as auditory and tactile senses, it will contribute to a wider range of pedestrian safety (e.g., a blind or a hearing-impaired person).

Acknowledgements

The contents of this paper are the results of the research project of the Ministry of Oceans and Fisheries of Korea(A fundamental research on maritime accident prevention - phase 2, PMS3840.

Appendix A Usability Questionnaire

Q1. I was able to be immersed in the game when the level was 1 (1=strongly disagree, 5=strongly agree).
Q2. I was able to recognize the surrounding traffic situation when the level was 1.
Q3. I was able to be immersed in the game when the level was 2.
Q4. I was able to recognize the surrounding traffic situation when the level was 2.
Q5. I was able to be immersed in the game when the level was 3.
Q6. I was able to recognize the surrounding traffic situation when the level was 3.
Q7. I was able to easily detect a potential collision with EyeSee360.
Q8. I was able to easily detect a potential collision with Radar.
Q9. I was able to precisely recognize position of an approaching vehicle with EyeSee360.
Q10. I was able to precisely recognize position of an approaching vehicle with Radar.
Q11. I was able to plan an actual evacuation plan from the potential collision with EyeSee360.
Q12. I was able to plan an actual evacuation plan from the potential collision with Radar.

References

[1] A. A. Ali and H. A. Hussein. Distance estimation and vehicle position detection based on monocular camera. In *Multidisciplinary in IT and Communication Science and Applications (AIC-MITCSA), Al-Sadeq International Conference on*, pp. 1–4. IEEE, 2016.
[2] J. W. Ayers, E. C. Leas, M. Dredze, J.-P. Allem, J. G. Grabowski, and L. Hill. Pokémon goa new distraction for drivers and pedestrians. *JAMA internal medicine*, 176(12):1865–1866, 2016.
[3] S. Barbieri, G. Vettore, V. Pietrantonio, R. Snenghi, A. Tredese, M. Bergamini, S. Previato, A. Stefanati, R. M. Gaudio, and P. Feltracco. Pedestrian inattention blindness while playing pokémon go as an emerging health-risk behavior: a case report. *Journal of medical internet research*, 19(4), 2017.
[4] O. Bimber and R. Raskar. *Spatial augmented reality: merging real and virtual worlds*. CRC press, 2005.
[5] D. A. Bowman and R. P. McMahan. Virtual reality: how much immersion is enough? *Computer*, 40(7), 2007.
[6] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière, and T. Chateau. Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.(CVPR)*, pp. 2040–2049, 2017.
[7] S. T. Chrysler, O. Ahmad, and C. W. Schwarz. Creating pedestrian crash scenarios in a driving simulator environment. *Traffic injury prevention*, 16(sup1):S12–S17, 2015.
[8] A. Colley, J. Thebault-Spieker, A. Y. Lin, D. Degraen, B. Fischman, J. Häkkilä, K. Kuehl, V. Nisi, N. J. Nunes, N. Wenig, et al. The geography of pokémon go: beneficial and problematic effects on places

and movement. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 1179–1192. ACM, 2017.

[9] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pp. 2366–2374, 2014.

[10] F. Eisenkeil, J. Ernst, R. Stadelhofer, U. Kühne, and O. Deussen. Traffic visualization in helmet-mounted displays in synchronization with navigation displays. In *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*, pp. 3A1–1. IEEE, 2015.

[11] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. doi: 10.1177/0278364913491297

[12] V. Girbes, L. Armesto, J. Dols, and J. Tornero. Haptic feedback to assist bus drivers for pedestrian safety at low speed. *IEEE transactions on haptics*, 9(3):345–357, 2016.

[13] U. Gruenefeld, A. E. Ali, W. Heuten, and S. Boll. Visualizing out-of-view objects in head-mounted augmented reality. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '17, pp. 81:1–81:7. ACM, New York, NY, USA, 2017. doi: 10.1145/3098279.3122124

[14] U. Gruenefeld, D. Ennenga, A. E. Ali, W. Heuten, and S. Boll. Eyesee360: Designing a visualization technique for out-of-view objects in head-mounted augmented reality. In *Proceedings of the 5th Symposium on Spatial User Interaction*, SUI '17, pp. 109–118. ACM, New York, NY, USA, 2017. doi: 10.1145/3131277.3132175

[15] S. G. Gustafson and P. P. Irani. Comparing visualizations for tracking off-screen moving targets. In *CHI '07 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '07, pp. 2399–2404. ACM, New York, NY, USA, 2007. doi: 10.1145/1240866.1241014

[16] C. L. Hardin. Red and yellow, green and blue, warm and cool: explaining colour appearance. *Journal of Consciousness Studies*, 7(8-9):113–122, 2000.

[17] S. G. Hart and L. E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, vol. 52, pp. 139–183. Elsevier, 1988.

[18] H.-N. Hu, Y.-C. Lin, M.-Y. Liu, H.-T. Cheng, Y.-J. Chang, and M. Sun. Deep 360 pilot: Learning a deep agent for piloting through 360 sports video. In *CVPR*, vol. 1, p. 3, 2017.

[19] S. Jain, C. Borgiattino, Y. Ren, M. Gruteser, and Y. Chen. On the limits of positioning-based pedestrian risk awareness. In *Proceedings of the 2014 workshop on Mobile augmented reality and robotic technology-based systems*, pp. 23–28. ACM, 2014.

[20] S. Jain, C. Borgiattino, Y. Ren, M. Gruteser, Y. Chen, and C. F. Chiasserini. Lookup: Enabling pedestrian safety services via shoe sensing. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 257–271. ACM, 2015.

[21] B. Joseph and D. G. Armstrong. Potential perils of peri-pokémon perambulation: the dark reality of augmented reality? *Oxford medical case reports*, 2016(10), 2016.

[22] M. Kampelmühler, M. G. Müller, and C. Feichtenhofer. Camera-based vehicle velocity estimation from monocular video. *arXiv preprint arXiv:1802.07094*, 2018.

[23] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pp. 239–248. IEEE, 2016.

[24] K. Lee, H. Kim, and C. Suh. Crash to not crash: Playing video games to predict vehicle collisions. In *ICML Workshop on Machine Learning for Autonomous Vehicles*, 2017.

[25] F. Liu, C. Shen, and G. Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5162–5170, 2015.

[26] D. Martín, F. García, B. Musleh, D. Olmeda, G. Peláez, P. Marín, A. Ponz, C. Rodríguez, A. Al-Kaff, A. de la Escalera, et al. Ivvi 2.0: An intelligent vehicle based on computational perception. *Expert Systems with Applications*, 41(17):7927–7944, 2014.

[27] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino. Augmented reality: A class of displays on the reality-virtuality continuum. In *Telemanipulator and telepresence technologies*, vol. 2351, pp. 282–293. International Society for Optics and Photonics, 1995.

[28] M. Mueller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem. Ue4sim: A photo-realistic simulator for computer vision applications. *arXiv preprint arXiv:1708.05869*, 2017.

[29] J. Nasar, P. Hecht, and R. Wener. Mobile telephones, distracted attention, and pedestrian safety. *Accident analysis & prevention*, 40(1):69–75, 2008.

[30] M. Nusinov, S. J. Yang, J. Holsopple, and M. Sudit. Visaw: Visualizing threat and impact assessment for enhanced situation awareness. In *Military Communications Conference, 2009. MILCOM 2009. IEEE*, pp. 1–7. IEEE, 2009.

[31] K.-Y. Park and S.-Y. Hwang. Robust range estimation with a monocular camera for vision-based forward collision warning system. *The Scientific World Journal*, 2014, 2014.

[32] D. Pešić, B. Antić, D. Glavić, and M. Milenković. The effects of mobile phone use on pedestrian crossing behaviour at unsignalized intersections–models for predicting unsafe pedestrians behaviour. *Safety science*, 82:1–8, 2016.

[33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[34] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *European Conference on Computer Vision*, pp. 102–118. Springer, 2016.

[35] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3234–3243, 2016.

[36] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[37] D. Stavrinos, K. W. Byington, and D. C. Schwebel. Distracted walking: cell phones increase injury risk for college pedestrians. *Journal of safety research*, 42(2):101–107, 2011.

[38] S. M. Straughn, R. Gray, and H. Z. Tan. To go or not to go: Stimulus-response compatibility for tactile and auditory pedestrian collision warnings. *IEEE Transactions on Haptics*, 2(2):111–117, 2009.

[39] M. Strickland, G. Fainekos, and H. B. Amor. Deep predictive models for collision risk assessment in autonomous driving. *arXiv preprint arXiv:1711.10453*, 2017.

[40] Å. Svensson and C. Hydén. Estimating the severity of safety related behaviour. *Accident Analysis & Prevention*, 38(2):379–385, 2006.

[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, et al. Going deeper with convolutions. Cvpr, 2015.

[42] S. Thorpe, D. Fize, and C. Marlot. Speed of processing in the human visual system. *nature*, 381(6582):520, 1996.

[43] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.

[44] N. Uchida, S. Takeuchi, T. Ishida, and Y. Shibata. Mobile traffic accident prevention system based on chronological changes of wireless signals and sensors. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 8(3):57–66, 2017.

[45] A. Volkhin. Road traffic simulator, 2018. [Online; accessed 10-March-2018].

[46] T. Wang, G. Cardone, A. Corradi, L. Torresani, and A. T. Campbell. Walksafe: a pedestrian safety app for mobile phone users who walk and talk while crossing roads. In *Proceedings of the twelfth workshop on mobile computing systems & applications*, p. 5. ACM, 2012.

[47] J. White, C. Thompson, H. Turner, B. Dougherty, and D. C. Schmidt. Wreckwatch: Automatic traffic accident detection and notification with smartphones. *Mobile Networks and Applications*, 16(3):285, 2011.