Determining the Relative Position of Vehicles Considering Bidirectional Traffic Scenarios in VANETS

Subhadeep Patra
subpat@doctor.upv.es

Carlos T. Calafate
calafate@disca.upv.es

Juan-Carlos Cano
jucano@disca.upv.es

Department of Computer Engineering, Universidad Politecnica de Valencia,
Camino de Vera S/N 46022, Valencia, Spain.

Peter Veelaert
Peter.Veelaert@UGent.be

Wilfried Philips
Wilfried.Philips@UGent.be

Department of Telecommunications and Information Processing, Ghent University,
St. Pietersnieuwstraat 33, 9000 Gent, Belgium.

ABSTRACT

Researchers pertaining to both academia and industry have shown strong interest in developing and improving the existing critical ITS solutions. In some of the existing solutions, specially the ones that aim at providing context aware services, the knowledge of relative positioning of one node by other nodes becomes crucial. In this paper we explore, apart from the conventional use of GPS data, the applicability of image processing to aid in determining the relative positions of nodes in a vehicular network. Experiments conducted show that both the use of location information and image processing works well and can be deployed depending on the requirement of the application. Our experiments show that the results that used location information were affected by GPS errors, while the use of image processing, although producing more accurate results, require significantly more processing power.

Keywords

Relative position; GPS data; vehicular network; ITS; license plate recognition; image processing.

1. INTRODUCTION

The aim of ITS applications is to contribute towards solutions that provide better safety and improvements in terms of smart coordination making use of vehicular or infrastructured networks. While designing such type of applications, sometimes situations arise where information is to be diffused to nodes or vehicles moving in a particular direction. Another example of such a situation is when it is necessary for a vehicle to communicate to another vehicle that is directly ahead or behind it. In these cases, the knowledge of the relative node location becomes necessary. In this paper we try to address this kind of challenges.

Calculation of relative location has been a topic of interest for a long time [1]. In the context of vehicular networks we can find methods like Dead Reckoning, Cellular Localisation, and Image/Video Localisation apart from the traditional GPS based algorithms. Image processing has already been proved to be useful in cases like preventing cars from crashing into one another [2], vision-based control of platooning [3], behaviour planning [4] of vehicles like lane detection, object classification, tracking, and representation of task-relevant objects. Here we consider very specific cases where ITS applications could be improved with the knowledge of the relative location that allows identifying vehicles moving in the same direction, or travelling just ahead of another. Examples of applications that can be enhanced based on the availability of such information are solutions like the ones proposed by Ji et al. [5], Johnson et al. [6] and Dai et al. [7]. In their work, Ji et al. presented a driver-fatigue monitoring system, while Johnson et al. proposed an application for smartphones that detects aggressive driving styles. Regarding the work of Dai et al., it is aimed at identifying drunk drivers. In this context, the knowledge of the relative location of nearby vehicles travelling ahead or behind could be useful to warn them of abnormal driver behaviour nearby. Another application as an example where the knowledge of the relative position becomes necessary is EYES [8], an application that aims at providing a visual overtaking aid from the vehicle ahead to the driver of the car behind. Here we want to perform a comparative study of the utility of a GPS based technique and an image processing algorithm to deliver relative location information.

The rest of this paper is organised as follows: In section II, we will present two possible methodologies that can be used to solve the problem stated in this paper. Later, we will discuss the results achieved for the proposed solutions, in section III. Finally, section IV concludes this paper by summarising our main contributions.

2. METHODOLOGY

As pointed out previously, we want to solve the problem of relative positioning. In particular, we propose to evaluate the impact of two different methods, one using location

SMARTOBJECTS'16, October 03-07, 2016, New York City, NY, USA
© 2016 ACM. ISBN 978-1-4503-4254-4/16/10…$15.00
DOI: http://dx.doi.org/10.1145/2980147.2980148

information provided by the GPS, and another one by employing an image processing technique.

Thus, the general requirements for our experiments are the availability of GPS, camera and a vehicular network for the exchange of data. Since all of the mentioned facilities are not available on all the cars used on a daily basis, we decided to use smartphones for this purpose. Smartphones are equipped with GPS, as well as with a camera that can be used for taking images for later processing.

For the exchange of information over the vehicular network we used the GRCBox [9], that is based on a Raspberry Pi [10] to be placed within each car. The GRCBoxes are used to create the vehicular network, and smartphones that actually process exchanged messages are connected to the network created. Hence, the GRCBox acts as a router for information exchange between vehicles.

2.1 Relative Positioning Based On GPS

The first solution involves the use of location information for resolving the relative positioning problem. The location information is collected using the smartphones which are equipped with GPS, and is exchanged between the cars using the advertisement messages or periodic beacons with the help of the vehicular network created using GRCBoxes.

Once the location information of the neighbours is received, each node can process the data as necessary to find out which vehicles are travelling in the same direction, and which node is directly ahead or behind it. For finding out which neighbouring vehicles are travelling in the same direction, we propose the *same direction test.*

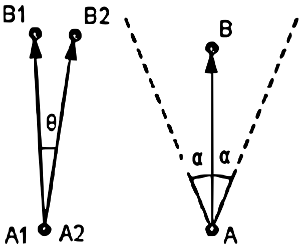

Figure 1: Same direction test.

Figure 1 explains how the *same direction test* works. For understanding the *same direction test*, let us assume we have two cars, one travelling from the point A1 to B1, and the other from A2 to B2. We measure the angle θ between these two vectors and compare it to a predefined threshold α. If θ is less than α, we can safely assume that the two vehicles are travelling in the same direction.

For more specific information like when it is necessary to know if a vehicle is travelling ahead of the other on the same lane, we have designed the *same lane test.* For this purpose we draw an imaginary line joining the current locations of the two vehicles, as shown in figure 2, where B1 and B2 are the current locations. Then we measure the angle of intersection of this line joining the points B1 and B2 with the displacement vectors of the vehicles. When the measured angle of intersection θ is less than a predefined angle β, then the vehicles are considered to be travelling on the same lane. Being on different lanes will result in a higher value of the measured angle θ, and the *same lane test* will fail.

Algorithm 1, provides a more detailed insight as to how the relative position of a node is to be identified based on the available location information. First, each node listens for periodic beacons or advertisement messages from its neighbours; the message contains the sender id and location history. This information is extracted from the advertisements received, and compared with the current node displacement vector to evaluate the *same direction test* or *same lane test.* Among all the neighbours that are found to be valid after evaluating the *same direction test* or the *same lane test*, the nearest neighbour is chosen based on distance.

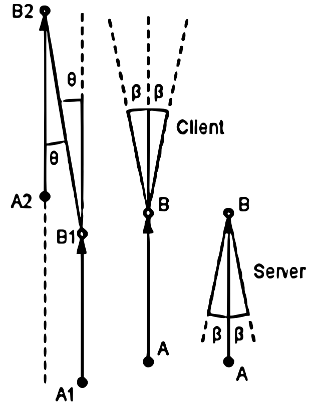

Figure 2: Same lane test.

Algorithm 1 Selecting the next hop neighbour for exchange of information, using GPS data

```
procedure NextHopUsingGPS
    neighbrId[] ← list of neighbours
    neighbrLoc[][] ← location history of neighbours
    devLoc[] ← location history of current node
    counter ← 0
    for each neighbour n in neighbrId[] do
        temp ← evaluate validation test
        if temp < threshold then
            validNeighbr[counter] ← neighbour n
            counter ← counter + 1
        end if
    end for
    nextHop ← validNeighbr[0]
    nextHopDist ← distance between nextHop
                  and current node
    for each neighbour n in validNeighbr[] do
        temp ← distance between neighbour n and
               current node
        if temp < nextHopDist then
            nextHop ← neighbour n
            nextHopDist ← temp
        end if
    end for
    return nextHop
end procedure
```

2.2 Using licence plate recognition

The next technique to address the issue of relative location, involves the use of image processing techniques to identify the license plate. This method is more suitable to detect obstacles, lanes or other vehicles ahead or behind the current node. Hence this method is to be used as a replacement

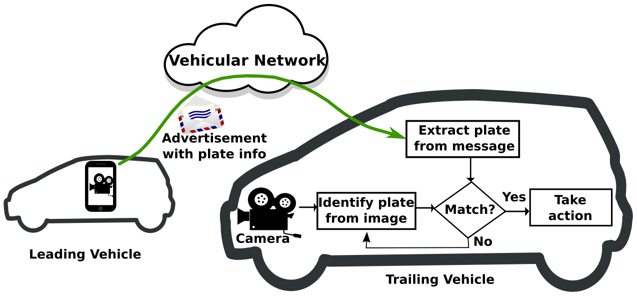
Vehicular Network
Advertisement with plate info
Extract plate from message
Identify plate from image
Match?
Yes
No
Take action
Camera
Leading Vehicle
Trailing Vehicle

Figure 3: Use of image processing in relative positioning.

for the *same lane test*.

The idea is explained in Figure 3, all vehicles would be broadcasting their plate information embedded in the advertisement message. The vehicle following it, would try to identify the license plate of the one ahead using the camera of the smartphone and compare the identified plate number with the information received in the advertisement. If there is a similarity between the identified plate information and the received plate number, then further action is taken.

It is to be noted that since identifying the license plate from an image is a CPU intensive task, this step could take a considerable amount of time. In our experiments, we used the processing capabilities of the smartphones to identify the plates.

Algorithm 2 Selecting the next hop neighbour for exchange of information, using image processing

```
procedure NextHopUsingImageProcessing
    neighbrId[] ← list of neighbours
    neighbrPlate[] ← plate info of the neighbours
                        from broadcast messages
    image ← image from camera mounted within
                        the vehicle
    rcgnzdPlate ← identify plate from the image
    if rcgnzdPlate != null then
        for each entry number n in neighbrPlate[] do
            accuracy[n] ← compare entry n with
                        rcgnzdPlate
        end for
    end if
    bestNeighbr ← neighbrPlate[0]
    bestAccuracy ← accuracy[0]
    for each entry number n in neighbrPlate[] do
        if bestAccuracy < accuracy[n] then
            bestNeighbr ← entry n
            bestAccuracy ← accuracy[n]
        end if
    end for
    if highestAccuracy > threshold then
        return bestNeighour
    end if
    return null
end procedure
```

Algorithm 2, provides an elaborate explanation as to how the relative position of a node is to be identified based on image processing. First, each node listens for periodic beacons or advertisement messages from its neighbours, the message contains the sender id and the actual license plate number of the vehicle, instead of its location history. At the receiver end, a camera is used to click a photo and identify the license place of the vehicle in the picture. The plate information from the advertisement message is compared with the identified license plate from the image. If there is a match then that vehicle is considered to be the next hop neighbour travelling directly ahead. This algorithm can also be used to locate vehicles following another vehicle, but in that case the camera has be be placed at the back of the vehicle instead.

3. EXPERIMENTAL RESULTS

In this section, we will compare the two methods of determining the relative position of vehicles. Experiments done using the algorithm that uses GPS data were performed in a real scenario involving real cars driven around the Universidad Politecnica de Valencia. Concerning the algorithm using image processing, tests were also performed outdoor, but they do not involve mobility.

Figure 4: The experimental setup.

Figure 4 shows the setup used for our experiments with GPS data. For our experiments that used GPS information, we used two cars driven around the university, each of them mounted with a tablet for storing the location of the cars. Later the information was used to evaluate the *same direction test* and *same lane test*, as explained in Section 2.1. A similar setup was used for tests with image processing, but in that case the images were taken of different cars to find out how effective the solution based on image processing was in identifying different license plates.

3.1 Using GPS data

The first case we are going to consider is a scenario where we want to send messages to all vehicles travelling in the same direction as the sender. In that case, we are going to use the *same direction test* described in Section 2.1. Since the *same direction test* is dependent on a threshold value, we tried to measure the output of the test without setting this threshold to find what a good value for it could be.

Figure 5 shows the density plot of the angles measured by the *same direction test*. As it can be seen from the graph, most observations for the *same direction test* lie within 20 degrees, which is satisfactory. It is also noticeable that many peaks occur due to GPS errors, also because the route followed had a lot of turns and curves, and so the two cars were

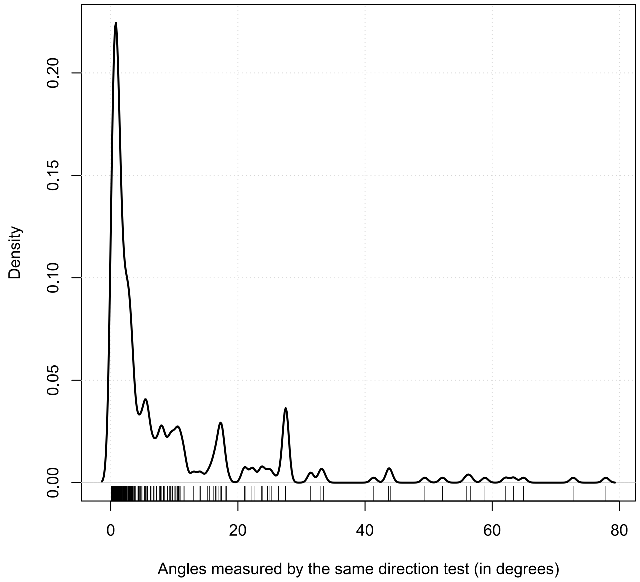

Figure 5: Results of the *same direction test*.

not always on a straight path.

Figure 6 shows the density graph of the *same lane test*, which is the case when the applications need to communicate with a vehicle directly ahead or behind. From this particular plot, we can see that most observations for the *same lane test* also lie within 20 degrees. Notice that this value is too high considering that this test is very sensitive. In fact, we were expecting a much lower threshold value, and so we find that for this particular condition the accuracy of current GPS technology is not good enough.

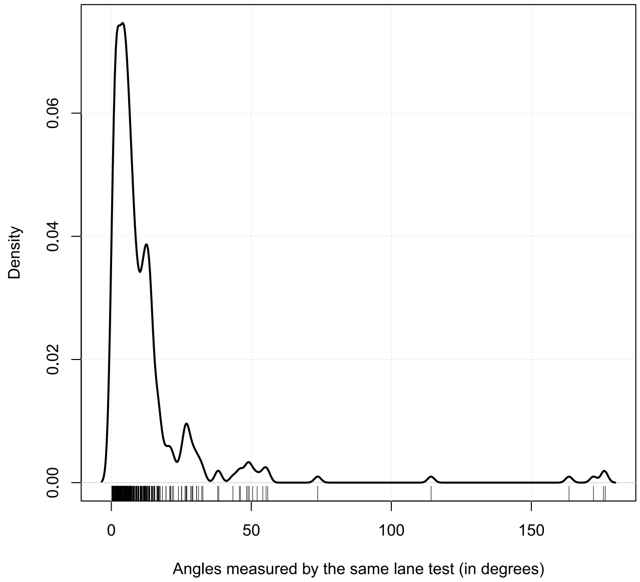

Figure 6: Results of the *same lane test*.

3.2 Relying on Image Processing

In this section, we will discuss the results obtained for the algorithm involving image processing. It is to be noted here that, in our first experiments to identify the license plate of cars, all targets were static or in slow motion.

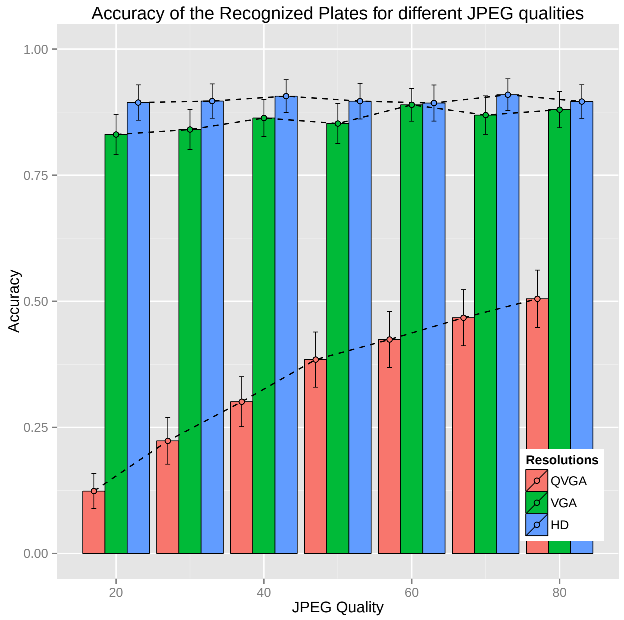

Figure 7: Variation of accuracy of the identified plate with images of different quality and resolution.

Perhaps the most important parameter that would help us decide whether the algorithm is worthwhile, is the accuracy of plate recognition. The value of the accuracy in our case ranges from 0 to 1. The value of 0 implies that the plate was not recognised, while 1 means that the recognised plate exactly matched the actual one. Next, the quality of the JPEG images can also affect accuracy of the recognised plate, and so we plot an Accuracy vs. JPEG Quality graph, as shown in Figure. 7. From this graph, we observe that there is a huge improvement in the accuracy with the increase in the JPEG quality for QVGA. However, the increase in accuracy is not so evident for higher resolutions of VGA and HD. For resolutions of HD and VGA, the average accuracy for all JPEG qualities were over 80 percent.

The next big concern regarding this technique is the processing time. Thus, we want to check if the idea of license plate recognition is actually feasible for smartphones. So, for the subsequent test in the static environment, we used photos of 200 static cars and converted them into HD, VGA and QVGA resolutions, finding out the processing time associated to each of these images using five different smartphones with different processing capabilities.

Figure. 8 shows the time taken to process images of different resolutions by Nexus 7 tablet, Moto G 3rd generation, Nexus 5X, Nexus 6 and Samsung Galaxy Note 10.1. It can be seen that devices with higher single core speeds perform better that the others. In this case, Nexus 6 and Samsung Galaxy Note 10.1 outperforms the rest, even beating the more recent Nexus 5X. The Nexus 6 processed HD images with average time slightly above 2 seconds, and VGA and QVGA images were processed successfully in 1.5 and 1.2 seconds, respectively. For Galaxy Note 10.1, the average processing time for HD images was 1.8 seconds, 1.4 for VGA

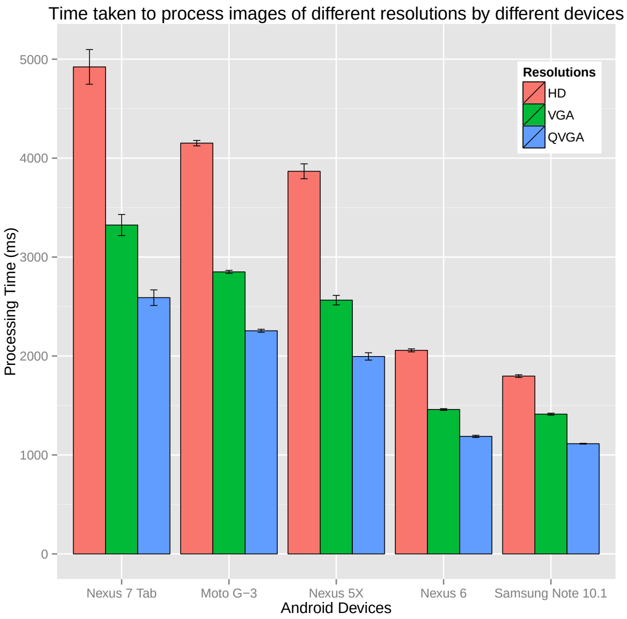

Figure 8: Processing Time for images of different resolutions by devices with different processor speeds.

and 1.1 seconds for QVGA images. Thus, using a powerful device can really reduce the processing time to an acceptable limit.

In the end, the experiments with image processing were only performed in a static environment, trying to recognise license plates of cars in a open parking area. Thus, in the future, we want to perform more experiments with our image processing algorithm, this time involving real cars in a high mobility scenario.

4. CONCLUSIONS

In this paper we analysed two methods to determine the relative position of vehicles using GPS data and image processing. For the algorithm using GPS data, the availability of GPS hardware is assumed. Two different scenarios were tested, both real scenarios involving real cars. In the first case involving GPS data, message diffusion takes place between all cars travelling in the same direction. In such a scenario, the results obtained were satisfactory. In the second case involving GPS, we considered a scenario where the message has to be send from one car to another directly ahead of it. In this particular case, the inaccuracy of the GPS data came into play, and the obtained results were affected by it. The experimental results obtained using image processing, which assumes the presence of optical sensors within the cars, produced more accurate results compared to the use of location information, but there was much more data processing involved. Thus, the selection of which technique to be used depends on the application requirements. If the importance of speed out-weights the accuracy factor, then the GPS based algorithm should be used. However, if more accurate tests are required to find the next hop that is directly ahead or behind a vehicle, then the use of image processing is the best option.

Acknowledgments

This work was partially supported by the *European Commission* under *Svāgata.eu*, the Erasmus Mundus Programme, Action 2 (EMA2) and the *Ministerio de Economía y Competitividad, Programa Estatal de Investigación, Desarrollo e Innovación Orientada a los Retos de la Sociedad, Proyectos I+D+I 2014*, Spain, under Grant TEC2014-52690-R.

5. REFERENCES

[1] A. Boukerche, H. A. Oliveira, E. F. Nakamura, and A. A. Loureiro, “Vehicular ad hoc networks: A new challenge for localization-based systems,” *Computer communications*, vol. 31, no. 12, pp. 2838–2849, 2008.

[2] W. D. Jones, “Keeping cars from crashing,” *Spectrum, IEEE*, vol. 38, no. 9, pp. 40–45, 2001.

[3] S. Benhimane, E. Malis, P. Rives, and J. R. Azinheira, “Vision-based control for car platooning using homography decomposition,” in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on.* IEEE, 2005, pp. 2161–2166.

[4] T. Bücher, C. Curio, J. Edelbrunner, C. Igel, D. Kastrup, I. Leefken, G. Lorenz, A. Steinhage, and W. Von Seelen, “Image processing and behavior planning for intelligent vehicles,” *Industrial Electronics, IEEE Transactions on*, vol. 50, no. 1, pp. 62–75, 2003.

[5] Q. Ji, Z. Zhu, and P. Lan, “Real-time nonintrusive monitoring and prediction of driver fatigue,” *Vehicular Technology, IEEE Transactions on*, vol. 53, no. 4, pp. 1052–1068, 2004.

[6] D. A. Johnson and M. M. Trivedi, “Driving style recognition using a smartphone as a sensor platform,” in *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on.* IEEE, 2011, pp. 1609–1615.

[7] J. Dai, J. Teng, X. Bai, Z. Shen, and D. Xuan, “Mobile phone based drunk driving detection,” in *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*, March 2010, pp. 1–8.

[8] S. Patra, C. Calafate, J.-C. Cano, and P. Manzoni, “An its solution providing real-time visual overtaking assistance using smartphones,” in *Local Computer Networks (LCN), 2015 IEEE 40th Conference on*, Oct 2015, pp. 270–278.

[9] S. M. Tornell, S. Patra, C. T. Calafate, J.-C. Cano, and P. Manzoni, “GRCBox: Extending Smartphone Connectivity in Vehicular Networks,” *International Journal of Distributed Sensor Networks*, 2014.

[10] “Raspberry Pi official website,” https://www.raspberrypi.org/, accessed: 2015-04-20.