SafeWalk: Early Warning System for Pedestrians

Diogo Dias
INESC-ID, Instituto Superior Técnico
Universidade de Lisboa
Lisbon, Portugal
diogo.g.dias@tecnico.ulisboa.pt

Pedro Rosa
INESC-ID, Instituto Superior Técnico
Universidade de Lisboa
Lisbon, Portugal
pmgsrosa@tecnico.ulisboa.pt

Miguel L. Pardal
INESC-ID, Instituto Superior Técnico
Universidade de Lisboa
Lisbon, Portugal
miguel.pardal@tecnico.ulisboa.pt

***Abstract*—Pedestrians on streets are exposed to dangerous situations, particularly when large vehicles cross urban areas and visibility is limited. However, there is a potential way to mitigate these dangers, as sensor technologies are increasingly ubiquitous in vehicles and nearly everyone carries a smartphone.**

This paper presents a safety application for pedestrians that uses Bluetooth Low Energy (BLE) in smartphones to broadcast Personal Safety Messages (PSMs) with the position, speed and heading of walkers. We started by evaluating the performance of BLE in some smartphone models and obtained very promising results. This technology can provide information even without a direct line of sight between the driver and the pedestrian. Then we developed an Android application with an algorithm for managing the data exchanged between the two parties. The PSMs received by the vehicles are compared with their current location to determine whether a collision will occur, and the driver is warned accordingly. We evaluated in different scenarios, two of which are also used in Euro NCAP tests since 2020. The results are very encouraging because detections were achieved in a timely manner, resulting in a practical early warning system.

***Index Terms*—Safety, V2P, PSM, Pedestrians, BLE, Bluetooth**

I. Introduction

Sensors in cars have become commonplace, providing safety warnings about immediate dangers. Furthermore, there are growing concerns about pedestrians safety, particularly in cities, due to the higher pedestrian and traffic density. Safety tests in cars now recognize the importance of sensors for the protection of vulnerable road users (VRU). In Europe, Euro Ncap [1] tests cars both for vehicle and VRU safety. However, in-car sensors have a limited range and can only detect within their line of sight. For traffic safety, timely communication between cars and pedestrians can add early awareness.

To widely deploy safety applications, the communication technology must already exist in devices used by pedestrians and cars. Bluetooth technology fits this requirement, being both inexpensive and present in nearly all mobile devices.

This paper introduces *SafeWalk*, an application that uses Bluetooth to improve collision detection systems and provide early warnings for pedestrians. Vehicles can already fuse data from multiple sensors [2]. Bluetooth can extend this sensor fusion with inputs from pedestrian mobile phones, providing additional information for collision detection systems, even in non-line-of-sight (NLOS) situations.

This document is organized as follows. Section II describes test experiments that substantiate the development of the application. Section III reviews other solutions that solve similar problems. Section IV provides an overview of the proposed solution. Section V presents the results of the solution, discussed in Section VI. Section VII outlines potential extensions and improvements, and Section VIII concludes the document.

INESC-ID is supported by national funds through FCT, Fundação para a Ciência e a Tecnologia, under project UIDB/50021/2020. This work is financially supported by Project Blockchain.PT – Decentralize Portugal with Blockchain Agenda, (Project no 51), WP 1: Agriculture and Agri-food, Call no 02/C05-i01.01/2022, funded by the Portuguese Recovery and Resilience Program (PPR), The Portuguese Republic and The European Union (EU) under the framework of Next Generation EU Program.

II. Background Experiments

Before designing our proposal, we performed preliminary experiments to ensure that Bluetooth Low Energy (BLE) would be effective in terms of both distance and latency and that Personal Safety Messages (PSM) could be encoded within the available payload size.

A. Bluetooth Coverage

To evaluate the effectiveness of BLE we performed some early tests to verify if a receiver device could capture BLE advertising packets at various fixed distances and also how fast they were detected if either the receiver or the transmitter were moving.

The receiver in these experiments was a laptop with Bluetooth (BT) v5 running in Ubuntu Linux 22.04 with an Intel i5-1135G7 CPU and 12GB of RAM. The laptop was running a Python script that used the Bleak [3] software to scan for BLE advertising packets and log them.

We used multiple smartphones acting as transmitters. They were running the nRF Connect for Mobile application from Nordic Semiconductor [4] to advertise BLE packets in the 1M PHY (physical layer) mode with a predefined transmission power (Tx Power) and advertising interval. Packets included the Tx Power and the device name for easier distinction. The receiver could detect these packets and also determine the strength of the received signal (RSSI). In all these experiments the transmitters used the maximum advertising frequency allowed, as well as the maximum Tx Power allowed.

For the first experiment we positioned the transmitters at various distances and checked if the receiver could detect the advertising packets, along with the calculated RSSI. Table I shows the different smartphone models used along with the

TABLE I
RSSI READINGS FROM VARIOUS DISTANCES

Smartphone model	Distance from receiver (m)	Average detected RSSI (dBm)
iPhone 14	140	-97
Galaxy A23 5G	128	-92
Galaxy S20	84	-95
HUAWEI P smart 2019	99	-91

TABLE II
TIME UNTIL PEDESTRIANS WOULD COLLIDE AT DIFFERENT SPEEDS

Smartphone model	TTC (s) 20 km/h	TTC (s) 40 km/h	TTC (s) 50 km/h
Galaxy A23 5G	29	9	7
Galaxy S20	17	16	5
HUAWEI P smart 2019	18	10	8

distance from the receiver at which they were positioned and the RSSI value calculated at the receiver. We got a signal from every transmitter, the furthest away being 140m from the receiver. As expected, the RSSI was generally lower the farther away it was from the receiver.

For the second experiment, the transmitters were placed in a crosswalk while the receiver was inside a vehicle. We treated the smartphones as if they were stationary pedestrians in the middle of the crosswalk. We then accelerated the vehicle, using various constant speeds, to see how long it took for the receiver to detect the advertising packets before the vehicle crossed the crosswalk. The receiver timestamped the moment the vehicle crossed the crosswalk and we compared it with the timestamps of the first received advertising packets from each transmitter. By subtracting those timestamps, we got the time that would take for the vehicle to collide with a stationary pedestrian from the moment the vehicle received the pedestrian's advertising packet. Since the vehicle was moving at a constant speed, we could calculate the distance between the transmitter and receiver when the first advertising packet was received. Table II shows the results of this experiment, indicating the time until the vehicle would collide (TTC) with a pedestrian at different speeds. We tested a speed of 50 km/h to include values above the speed limit.

For the third experiment, we repeated the second experiment with the receiver and transmitters swapped: the receiver sat besides the crosswalk acting as a Road Side Unit (RSU) receiving advertisements from vehicles, while the transmitters sat inside the vehicle acting as multiple vehicles broadcasting advertisements. We accelerated the vehicle towards the crosswalk, maintaining various constant speeds. The receiver measured the time to detect the advertising packets (we did not test at 50 km/h as it was too fast for an urban street). Table III shows the results of this experiment, indicating the time until the vehicle would cross the RSU at different speeds.

These results provided a good estimate of the time available to notify each entity. Notably, since the experiments were conducted on a slightly curved road, the receiver detected most of the advertising packets before there was a direct line of sight to the transmitters.

TABLE III
TIME UNTIL EACH VEHICLE CROSSES THE CROSSWALK AFTER THE RSU RECEIVES THE FIRST ADVERTISEMENT

Smartphone model	TTC (s) 20 km/h	TTC (s) 40 km/h
Galaxy A23 5G	27	11
Galaxy S20	15	6
HUAWEI P smart 2019	23	13

B. Message Encoding

In BLE advertising, an advertisement data structure can occupy up to 31 bytes [5]. However, depending on the type of the structure, some bytes are reserved for the header. This leaves us with 27 bytes for useful data. We chose the Manufacturer-Specific Data type, where the first byte indicates the length of the structure, the second byte indicates the type of advertisement, and the next two bytes indicate the manufacturer ID. This leaves us with 27 bytes for useful data.

PSM, defined in the SAE J2735 standard [6], are used to broadcast safety data regarding VRU, including the positions of pedestrians. Depending on the encoding of the messages, they can occupy as little as 26 bytes if only the required fields are utilized. As such, it is possible to incorporate PSM in BLE advertising packets.

The PSM data structure is defined using ASN.1, a standard interface description language for defining data structures, using the Unaligned Packed Encoding Rules (UPER), the most compact encoding [7]. UPER uses the fewest bits necessary for fields, concatenating them in order without padding.

It is important to note that with sequences, if there is an extension marker, an optional field or a default field, we need a fixed size bit-field preamble to record the presence or absence of optional fields in the sequence and whether or not the extension marker is present. This is most noticeable on the top-level sequence which has a lot of optional fields and an extension marker and, therefore, takes up 19 bits. The only other sequence that also takes up space is the `position`.

In total, the encoding of the PSM takes up 204 bits which, after padding the last octet, takes up 26 bytes. 26 bytes plus the 4 bytes required for the advertising packet headers totals 30 bytes, which fits under the maximum advertisement data size of 31 bytes. It is still possible to include optional information on the PSM as we still have 12 bits left before exceeding the maximum size of advertising data. A representation of the advertisement packet structure is shown in Figure 1.

III. RELATED WORK

Wu et al. developed BLE-Horn [8], an Android application which uses BLE for bidirectional communication between vehicles and pedestrians at intersections. This is accomplished by including compressed GPS data in the BLE advertising packets. Both pedestrians and vehicles broadcast their GPS data and a collision avoidance algorithm predicts the probability of collision based on the data. Our solution includes a standardized PSM in the advertising packets and reduces packet loss probability by having only pedestrians send them.

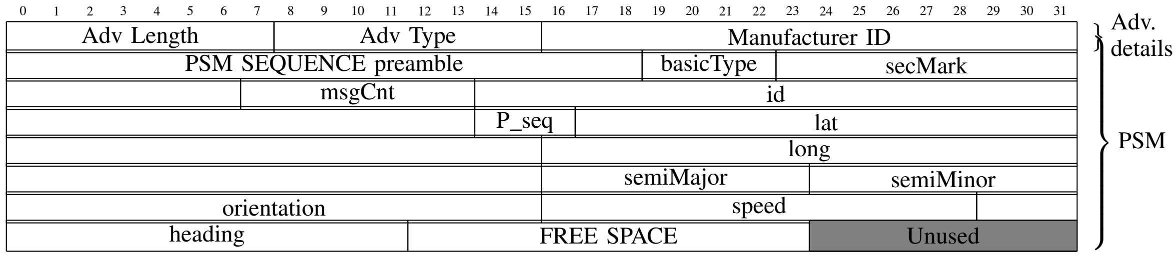

Fig. 1. Advertisement packet structure

Ho and Chen [9] present a system similar to BLE-Horn, but using Wi-Fi for communication. To broadcast GPS information, the pedestrian's device is set to AP (Access Point) mode and the SSID (service set identifier) is replaced by the GPS information. The SSID is broadcast in beacon messages every 100ms, similar to Basic Safety Messages (BSM) in vehicles, and can be received by low-cost Wi-Fi cards or standard smartphones. Although this approach uses Wi-Fi for communication with low power consumption similar to BLE, Wi-Fi still generally consumes more energy than BLE [10].

Zhou et al. [11] use Lidar sensors in RSUs to detect the position of pedestrians and combine it with information gathered from pedestrian smartphones. This data includes attributes such as whether the pedestrian is a child, older, or visually impaired. Pedestrian smartphones wirelessly transmit attribute and GPS information to the network, which is corrected using information from the high-precision Lidar sensors. Vehicles can then access the network to get this data.

Quack et al. [12] propose an algorithm to track road users, including vehicles and pedestrians, using multiple sensors like Lidar, cameras, ultrasonic sensors on infrastructure, and GPS on smartphones and vehicles. Vehicles additionally send their own sensor data from Lidar and camera images. Road users send their data to a processing center where an algorithm corrects for latency.

These works on sensor fusion highlight the potential of integrating BLE communications between smartphones and RSU. This integration could provide additional information on pedestrian positions, enhancing the accuracy of pedestrian positioning algorithms.

IV. Proposed solution

Our preliminary experiments showed that we could get data in a timely manner and encode it following standards. The next step was to calculate the probability of impact with a VRU. For this purpose, we adapted an algorithm by Qu et al. [13]. It first uses the relative position and distance between the vehicle and pedestrian to calculate the point where they will intersect and possibly collide. Then it uses the speed of the vehicle and pedestrian to calculate how much time it takes until each of them enters and exits the point of intersection. Depending on the accuracy of the GPS locations, the algorithm issues a "high" or "medium" level warning if a collision is predicted. If there is no collision predicted at the current speeds of the vehicle and pedestrian, but still some proximity, it issues a "low" level warning. If there is no intersection in the first place but a pedestrian is nearby, the algorithm simply notifies the vehicle.

The original algorithm is used to detect collisions between two vehicles, while our algorithm detects collisions between a vehicle and a pedestrian. This algorithm differs from the original in the way it determines times on T_{sv}, T_{sp}, T_{ev} and T_{ep}. The method to calculate TTC is also simplified: it simply takes the value of when the time intervals first intersect.

The algorithm can produce the following outcomes:

- `NO_COLLISION` means that the paths of the vehicle and the pedestrian will never intersect.
- `PEDESTRIAN_NEARBY` means that no collision will occur, but there is a pedestrian near the vehicle.
- `PEDESTRIAN_LOS` means that the paths of the vehicle and the pedestrian will intersect at some point, but at their current speeds either the vehicle or the pedestrian will have already left the point of intersection.
- `COLLISION_PROBABLE` means that a collision will occur and that the accuracy from the vehicle and pedestrian locations are more inaccurate than the accuracy threshold.
- `COLLISION_IMMINENT` means that a collision will occur and that the accuracy from the vehicle and pedestrian locations are more accurate than the accuracy threshold.

Regarding Algorithm 1:

- Loc represents the GPS location information which includes latitude, longitude, speed and bearing;
- W and L represent width and length;
- ϕ is the bearing in degrees east of true north when traveling along the shortest path between the vehicle and the pedestrian locations;
- α is the angle formed between the vehicle's heading and the pedestrian's position;
- β is the angle between the vehicle's heading and the pedestrian's heading;
- X_{vp} is the distance between the vehicle's location and the pedestrian's location in meters;
- X_{cv} and X_{cp} represent the distance until the vehicle and the pedestrian reach the point of collision, respectively;
- T_{sv} and T_{sp} represent the time until the front of the vehicle and the pedestrian starts to enter the point of collision, while
- T_{ev} and T_{ep} represent the time until the back of the vehicle and the pedestrian exit the point of collision;
- M represents whether the time intervals intersect.

The default values we chose to use for testing are the

Algorithm 1 Collision estimation algorithm.

Require: Loc_v, Loc_p, W_v, L_v, W_p, L_p

$\phi = computeAngle(Loc_v, Loc_p)$
$\alpha = \phi - heading(Loc_v)$
$\beta = heading(Loc_p) - heading(Loc_v)$
$X_{vp} = computeDistance(Loc_v, Loc_p)$
$X_{cv} = X_{vp} * (\cos\alpha - \sin\alpha/\tan\beta)$
$X_{cp} = -X_{vp} * \sin\alpha/\sin\beta$
if $X_{cv} < 0$ or $X_{cp} < 0$ **then**
    if $X_{vp} <$ threshold **then**
        return PEDESTRIAN_NEARBY
    else
        return NO_COLLISION
    end if
end if
$T_{sv} = (X_{cv} - L_v/2 - W_p/2)/speed(Loc_v)$
$T_{ev} = (X_{cv} + L_v/2 + W_p/2)/speed(Loc_v)$
$T_{sp} = (X_{cp} - L_p/2 - W_v/2)/speed(Loc_p)$
$T_{ep} = (X_{cp} + L_p/2 + W_v/2)/speed(Loc_p)$
$M = (T_{sv} - T_{ep}) * (T_{sp} - T_{ev})$
if $T_{sp} > T_{sv}$ and $T_{sp} < T_{ev}$ **then**
    $TTC = T_{sp}$
else
    $TTC = T_{sv}$
end if
if $M > 0$ **then**
    if Loc_v and Loc_p above accuracy threshold **then**
        return COLLISION_IMMINENT
    else
        return COLLISION_PROBABLE
    end if
else
    return PEDESTRIAN_LOS
end if

following: W_v = 2; L_v = 5; W_p = 1; L_p = 1; X_{vp} threshold = 5; Accuracy threshold = 4.5 The values for vehicle and pedestrian width and length were chosen to be above the average values in an effort to increase true positive results where a real collision will occur, even at the cost of false positives where one might not. Values for X_{vp} and accuracy thresholds were chosen arbitrarily, as these might later be optimized with further testing.

We developed an Android application with two main screens: the *Pedestrian* screen, where the user can start or stop transmitting the smartphone's position through BLE advertising packets; and the *Driver* screen, where the user can start or stop scanning for those packets.

The application frequently updates the current GPS position with the fastest interval permitted by the Android and broadcasts the advertising packets every 100ms. As soon as the application detects a change in the current GPS position, it generates a new PSM advertising the position update.

On the Driver screen, every time the application receives an advertising packet, it decodes the PSM and generates a Location object based on the decoded information. Then the application runs the collision detection algorithm with the pedestrian's location and the driver's most recent known location. Depending on the output of the algorithm, the screen flashes a certain color and outputs text to the screen.

TABLE IV
APPLICATION EVALUATION - MAXIMUM DISTANCE READINGS

Smartphone model	Bluetooth Version	Distance from receiver (m)
Galaxy A52 5G	5.0	205
Xperia X	4.2	88
Zenfone 7 PRO	5.0	53

TABLE V
TIME TO COLLISION WITH THE VEHICLE AT VARYING SPEEDS

Smartphone model	TTC (s) 20km/h	TTC (s) 40km/h	TTC (s) 50km/h
Galaxy A52 5G	24.7	10.2	3.4
Xperia X	3.8	5.2	4.2
Zenfone 7 PRO	3.6	1.1	0.6

V. EXPERIMENTAL RESULTS

A. Initial application tests

We first evaluated our application through field tests in real-life scenarios. Some tests assessed how far a driver using the application could capture BLE advertising packets from pedestrians and how quickly they were detected if the vehicle was moving. The application used the maximum advertising frequency and Tx power allowed by Android: 100 ms and 1 dBm, respectively.

Table IV shows the different smartphone models used along with their BT version and the distance at which they were first detected while slowly approaching the vehicle running the application. To test how fast the application captured the advertising packets, first we placed 3 smartphones besides a crosswalk running the application and started broadcasting. Then we used a smartphone running the application inside a vehicle to scan for these advertisements. We accelerated the vehicle and maintained various constant speeds to see how long it took for the application to detect the advertising packets before the vehicle crossed the crosswalk. A person in the vehicle timestamped the moment the vehicle crossed the crosswalk and we compared it with the timestamps of the first received advertising packets from each broadcasting smartphone. By subtracting those timestamps, we got the time that would take for the vehicle to collide with a stationary pedestrian from the moment the vehicle received the pedestrian's advertising packet. The smartphone scanning for advertisements inside the vehicle was a Samsung Galaxy S20, Android 13, BT 5.0.

Table V shows the results of this experiment, indicating the time until the vehicle would collide with a pedestrian at different speeds. These tests were performed in the same slightly curved road as the initial BT tests, with no direct line of sight, but this time under heavy rain.

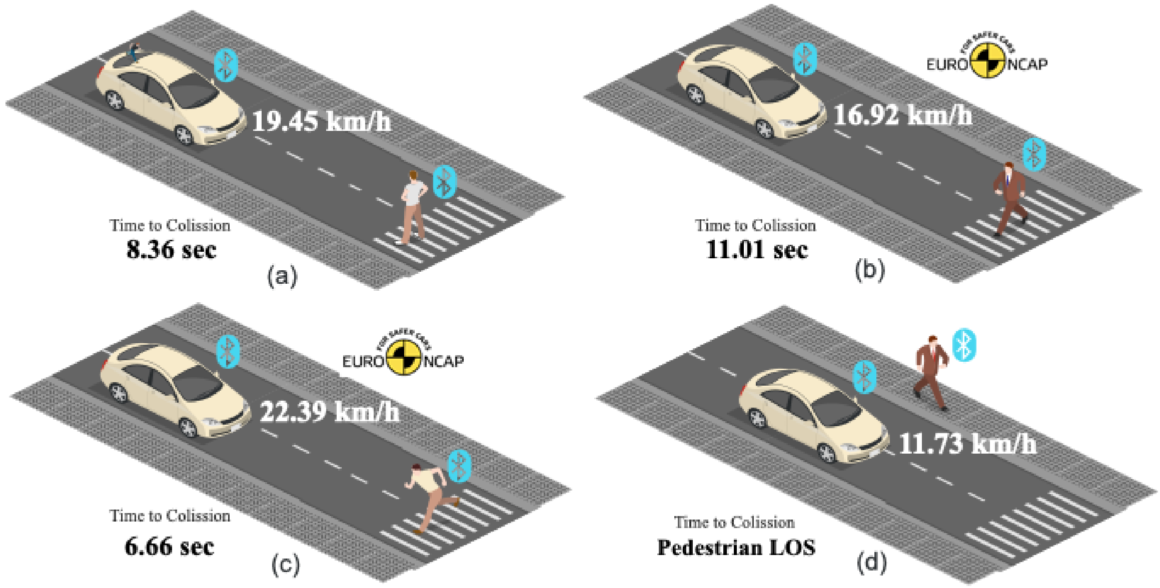

Fig. 2. Test Scenarios (a) Pedestrian stands still on path (b) Pedestrian walks across path (c) Pedestrian runs across path (d) Pedestrian walks along path

B. Algorithm results

For the main test we used seven different scenarios, four of which are shown in Figure 2:

(a) pedestrian stands still on the path of the vehicle;
(b) pedestrian walks to cross the path of the vehicle;
(c) pedestrian runs to cross the path of the vehicle;
(d) pedestrian walks alongside the path of the vehicle;
(e) pedestrian runs alongside the path of the vehicle;
(f) pedestrian walks directly towards a vehicle backing up;
(g) pedestrian walks to the path of a vehicle backing up.

Scenarios (b) and (c) are included in Euro NCAP tests for Autonomous Emergency Breaking (AEB) [1]. Results:

(a) Earliest collision detected with TTC of 8.36s with the vehicle going at 5.48 m/s;
(b) Earliest collision detected with TTC of 11.01s with the vehicle going at 4.70 m/s;
(c) Earliest collision detected with TTC of 6.66s with the vehicle going at 6.22 m/s;
(d) No collisions were detected;
(e) No collisions were detected;
(f) Earliest collision detected with TTC of 5.89s with the vehicle going at 2.0 m/s;
(g) No collisions were detected.

The vehicle smartphone was a Samsung Galaxy A23 5G running Android 14 with BT 5.1, while the pedestrian's was a Galaxy A52s 5G, running the same Android with BT 5.0.

The following condensed logs show some scenarios where each entry in a log represents a change in the collision result.

(a) Pedestrian standing in middle of road

```
18:50:27.957 pedestrian_los V: 5.68 TTC: -1.00 Xvp: 53.88
18:50:28.921 collision_imminent V: 5.48 TTC: 8.36 Xvp: 48.49
18:50:30.964 pedestrian_los V: 5.27 TTC: -1.00 Xvp: 38.43
18:50:32.037 collision_imminent V: 4.71 TTC: 6.22 Xvp: 33.51
```

(b) Pedestrian walking to cross vehicle path

```
18:52:18.049 pedestrian_los V: 5.17 TTC: -1.00 Xvp: 65.28
18:52:21.018 collision_probable V: 4.70 TTC: 11.01 Xvp: 51.69
18:52:21.554 pedestrian_los V: 4.60 TTC: -1.00 Xvp: 48.53
18:52:21.960 collision_probable V: 4.59 TTC: 9.01 Xvp: 46.30
18:52:27.959 collision_imminent V: 6.22 TTC: 1.37 Xvp: 11.59
```

(c) Pedestrian running to cross vehicle path

```
18:54:58.984 pedestrian_los V: 6.19 TTC: -1.00 Xvp: 37.70
18:54:59.947 collision_imminent V: 7.06 TTC: 3.53 Xvp: 31.47
18:55:01.026 pedestrian_los V: 6.68 TTC: -1.00 Xvp: 24.92
18:55:03.061 collision_imminent V: 2.22 TTC: 3.97 Xvp: 15.96
```

(d) Pedestrian walks the same direction as vehicle

```
18:56:49.782 no_collision V: 0.39 TTC: -1.00 Xvp: 27.72
18:56:53.043 pedestrian_los V: 3.26 TTC: -1.00 Xvp: 25.73
```

(e) Pedestrian runs the same direction as vehicle

```
18:58:31.970 pedestrian_los V: 6.74 TTC: -1.00 Xvp: 12.30
18:58:36.611 pedestrian_nearby V: 6.81 TTC: -1.00 Xvp: 4.18
```

(f) Pedestrian walks towards vehicle that is backing up

```
19:00:56.923 pedestrian_los V: 1.89 TTC: -1.00 Xvp: 25.71
19:00:57.464 no_collision V: 1.89 TTC: -1.00 Xvp: 25.75
19:00:58.977 collision_imminent V: 2.00 TTC: 5.89 Xvp: 20.50
```

(g) Pedestrian walks to cross vehicle that is backing up

```
19:02:53.217 pedestrian_los V: 2.06 TTC: -1.00 Xvp: 29.51
19:03:00.612 no_collision V: 2.55 TTC: -1.00 Xvp: 11.82
19:03:02.945 pedestrian_nearby V: 1.20 TTC: -1.00 Xvp: 3.80
```

VI. Discussion

The default values for vehicle and pedestrian width and length were set above average, prioritizing safety by increasing detection likelihood, despite the risk of more false positives.

In scenarios (a), (b), (c) and (f), where a collision was to be expected, the algorithm performed correctly in returning either `COLLISION_PROBABLE` or `COLLISION_IMMINENT` at some point. Fortunately in scenarios (d) and (e), which involve the pedestrian walking and running alongside the vehicle, no collisions were detected, therefore no false positives.

Only in scenario (g), where a collision was to be expected, the algorithm never returned a collision warning, only having returned `NO_COLLISION`, `PEDESTRIAN_LOS` and `PEDESTRIAN_NEARBY`. This false negative that was caused by inaccuracies in the bearing reported by the pedestrian's smartphone. While the bearing of the pedestrian should have been around the bearing of the vehicle plus 90º, as in scenarios (b) and (c), in this case the bearing values varied a lot. We observed that the bearing accuracy tends to increase with speed, so the problem may be due to the pedestrian walking at a low speed.

In most of the tests, `COLLISION_PROBABLE` occurred more often than `COLLISION_IMMINENT`, meaning that the position accuracy value of the smartphones was more than 4.5m most of the time, as that was the accuracy threshold we chose for the tests. The biggest hurdle to achieving better results with the current algorithm is GPS accuracy, as we already have good results with how fast the first BLE advertising packets can be detected.

VII. Future Work

The PSM also has a lot of optional fields which our application could take advantage of, seeing as there are still 12 bits of free space in the advertising packet. Some interesting fields might include `crossRequest` or `crossState`, which represent the intent or state of crossing the road, `assistType` which indicates special needs of pedestrians (e.g. vision, hearing and movement). It is of particular interest knowing whether the pedestrian has slower movement, such as an elderly person, or more erratic movement, such as a child, as this significantly affects bearing accuracy.

We are using PSM, that are a SAE standard, but our system is agnostic to the type of messages used. This means that we could also use VAM awareness messages from the ETSI standard [14]. VAM could prove useful because the heading and speed fields also contain confidence intervals which we could adapt to make the algorithm more accurate.

BLE has different physical layers to choose from. While the one we chose was 1M PHY due to it being more compatible with older smartphones and having longer range than 2M PHY, it could prove useful to experiment with the newer Coded PHY which should have longer communication range [15].

Our work allows for important safety features to be developed in user interfaces. For example, joggers wearing earbuds can receive audio warnings when a vehicle is approaching. Additionally, since joggers usually move at higher speeds, the bearing sensor data is generally more accurate.

VIII. Conclusion

A fact of modern life is that most people carry smartphones or similar devices. This presents the opportunity to exchange information between pedestrians and vehicles to enhance safety. We evaluated the effectiveness of BLE for communication between vehicles and pedestrians, having smartphones broadcast their position and trajectory to nearby vehicles through an app that sends PSM messages using BLE. We demonstrated that this information can enhance collision detection systems and provide early pedestrian warnings. The tests yielded promising results, with the algorithm consistently issuing correct warnings. We anticipate that this technology could lead to significant improvements in road safety.

References

[1] Euro NCAP, "Euro NCAP AEB LSS VRU Test Protocol - v4.5.1," Feb. 2024. [Online]. Available: https://www.euroncap.com/media/80156/euro-ncap-aeb-lss-vru-test-protocol-v451.pdf

[2] Z. Xu, H. Deng, R. Zhang, and L. Yang, "An Efficient Multi-Sensor Fusion Protocol in a Vehicle-Road Collaborative System," in *2021 IEEE/CIC International Conference on Communications in China (ICCC)*, Jul. 2021, pp. 1161–1165, iSSN: 2377-8644.

[3] H. Blidh, "Bleak," Oct. 2023, original-date: 2018-04-26T21:15:45Z. [Online]. Available: https://github.com/hbldh/bleak

[4] Nordic Semiconductor ASA,, "nRF Connect for Mobile." [Online]. Available: https://play.google.com/store/apps/details?id=no.nordicsemi.android.mcp

[5] J. Tosi, F. Taffoni, M. Santacatterina, R. Sannino, and D. Formica, "Performance Evaluation of Bluetooth Low Energy: A Systematic Review," *Sensors*, vol. 17, no. 12, p. 2898, Dec. 2017, number: 12 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/1424-8220/17/12/2898

[6] SAE International, "V2X Communications Message Set Dictionary," Sep. 2023.

[7] ITU-T, "Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)," Feb. 2021. [Online]. Available: https://www.itu.int/ITU-T/studygroups/com17/languages/X.691-0207.pdf

[8] M. Wu, B. Ma, Z. Liu, L. Xiu, and L. Zhang, "BLE-horn: A smartphone-based bluetooth low energy vehicle-to-pedestrian safety system," in *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, Oct. 2017, pp. 1–6, iSSN: 2472-7628. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8171180

[9] P.-F. Ho and J.-C. Chen, "WiSafe: Wi-Fi Pedestrian Collision Avoidance System," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 4564–4578, Jun. 2017, conference Name: IEEE Transactions on Vehicular Technology. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7539332

[10] A. Abedi, O. Abari, and T. Brecht, "Wi-LE: Can WiFi Replace Bluetooth?" in *Proceedings of the 18th ACM Workshop on Hot Topics in Networks*, ser. HotNets '19. New York, NY, USA: Association for Computing Machinery, Nov. 2019, pp. 117–124. [Online]. Available: https://doi.org/10.1145/3365609.3365853

[11] Z. Zhou, S. Kitamura, Y. Watanabe, S. Yamada, and H. Takada, "Extraction of Pedestrian Position and Attribute Information Based on the Integration of LiDAR and Smartphone Sensors," in *2021 IEEE International Conference on Mechatronics and Automation (ICMA)*, Aug. 2021, pp. 784–789, iSSN: 2152-744X. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9512649

[12] T. Quack, F.-J. Heßeler, and D. Abel, "Infrastructure-Based Tracking of Road Users in Urban Intersections for Partially Available, Variable-Delay Sensor Data," in *2018 26th Mediterranean Conference on Control and Automation (MED)*, Jun. 2018, pp. 1–7, iSSN: 2473-3504. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8442691

[13] C. Qu, W.-Y. Qi, and P. Wu, "A High Precision and Efficient Time-to-Collision Algorithm for Collision Warning Based V2X Applications," in *2018 2nd International Conference on Robotics and Automation Sciences (ICRAS)*, Jun. 2018, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8443265

[14] ETSI, "TS 103 300-3 - V2.2.1 - Intelligent Transport Systems (ITS); Vulnerable Road Users (VRU) awareness; Part 3: Specification of VRU awareness basic service; Release 2," Feb. 2023.

[15] J. C. G. Ortiz, J. Silvestre-Blanes, V. M. Sempere-Payá, and D. C. Frau, "Evaluation of improvements in BLE Mesh through CODED PHY," in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2021, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9613315