AL-S^{2}m: Soft road traffic Signs map for vehicular systems

Ammar LAHLOUHI
Department of Computer Science,
University of Batna 2,
05 000 Batna, Algeria

***Abstract*—In this paper, we describe AL-S^{2}m, a roadmap with traffic signs to be used in vehicular systems. AL-S^{2}m is part of a more general system of traffic signs (TSs) management, called AL-S^2, which includes two sides: central map server and client vehicular system. The server allows establishing, maintaining and disseminating AL-S^{2}m. The client localizes the vehicle in AL-S^{2}m and detects TSs. In this paper, we focus on the AL-S^{2}m establishment. AL-S^{2}m can handle variable TSs and its update is easy, which keeps it coherent with the reality. Also, it improves the map-matching algorithm. We implemented AL-S^{2}m easily using an Android device.**

Keywords*—Road Traffic Signs, Roadmap, Map-Matching, Driver Assistance Systems, Autonomous vehicles

I. Introduction and Motivation

Advanced Driver Assistance Systems (ADAS) have the capacity to increase driver safety by observing the driver and its environment and providing information, warnings, or even taking actions. Integrating ADAS into adaptive cruise control system can reduce the cognitive load of the driver and supports safe driving. Completely automated tasks of the driver lead to autonomous vehicles AV. Both ADAS and AV need to be aware of local traffic laws, such as the right of way and speed limits, as well as information such as city limits to comply with traffic rules. The transport authority specifies such rules in traffic signs (TSs).

Several important works tried to make vehicular systems aware of TSs. The well known of such works are those based on TSs detection and recognition (DRS) (see [8] for instance) using a camera onboard of the vehicle. Such systems are more consistent with the reality since they report the state of the reality as it is when the vehicle moves by detecting, particularly, variable and temporary TSs.

High precision real-time DRS is a hard task [11]. In spite of several works on DRS, such as [1], [3], [5], [6], [12], [14], current DRSs are lengthy in recognition time, sometimes, deduce an incorrect detection, and require a special powerful processing devices and a material for Nocturne vision during darkness time. Also, bad weather conditions (snowing, raining and fog) and direct sunlight or sunlight reflections by TSs can affect camera sign detection. Even with the best design and under perfect weather conditions, cameras are still far away from performing as accurately as the drivers vision. The limited field of view and fixed related position on the vehicle may cause the camera system to miss a TS, specifically in sharp curves. In some occasions, other vehicles may hide TSs. Even with a very good image processing system, it may not be easy to determine that the TS does not belong to the road in which the vehicle travels.

Several works use digital maps for navigation purposes that the vehicular system can exploit, such as the ones proposed in [9], [10], [18]. Firstly, we must build and store a detailed roadmap including TSs. Next, the estimator of the vehicular system filters online measurements and gives the position of the vehicle in the roadmap. Then, one can make TSs detection independently of atmospheric disturbances and traffic situations. An interesting approach [7], applied to speed limit TSs only, combines a camera based with a map-based system to provide information that is more precise. The first detects variable and temporary speed limits while the second detects implicit ones, such as when entering from a country road into a town when there is no TS indicating a speed limit.

Two difficulties remain not issued with the important system proposed in [7]. Firstly, the TSs detection with a camera-based system remains problematic. Secondly, with the static nature of current roadmaps where their design aims updating them for long periods, they cannot satisfactorily handle temporary and variable TSs.

Given the previous weaknesses, a map-based solution can be more significant. However, current roadmaps have particularly two difficulties: (1) They are usually incoherent with the reality since they are often outdated, and (2) they cannot handle variable TSs (vTSs). Firstly, the roadmaps incoherence is due to the difficulty of their updating, which includes a lot of information at once. A solution resides in specialized (simplified) roadmaps that one can update easily. Secondly, the problem of vTSs is due to static nature of the current maps. A solution for vTSs resides in dynamic maps.

In this paper, we propose an entirely map based system, called AL-S^2 (Augmented Lighted – Soft Signs). AL-S^2 takes into account the two previous solutions. Its achievement includes two parts: central map server and client vehicular system. The server establishes, maintains and disseminates a specific roadmap (AL-S^{2}m). The client localizes the vehicle in AL-S^{2}m and detects TSs. AL-S^2 tracks then the movement of the vehicle and when it reaches the visibility field of a TS, it displays it. In this paper, we focus on the specific roadmap AL-S^{2}m only.

In AL-S^{2}m, we represent the road network as a simple topology of roads with their associated TSs. Updating AL-

S^{2}m consists in simple operations (adding, deleting and modifying) on the representations of the roads and TSs. Such operations can be broadcasted timely and wirelessly by the central map server to AL-S^{2}m. Broadcasting such information is not problematic with current generations of communication systems. Timely broadcasting such operations allows handling temporary TSs.

The remainder of the paper describes AL-S^{2}m representation (virtualizing roads and TSs). We organized it in four sections, in addition to a conclusion. Section II describes the roadmap representation. It focuses mainly on representing the elements of AL-S^{2}m, its achievement, and the map-matching method. Section III focuses on traffic signs. It consist of TSs representation (including the variable TSs), their insertion in AL-S^{2}m, and their detection during the vehicle movement. Section IV describes an implementation of AL-S^{2}m using an Android device. Section V presents an experimentation of AL-S^{2}m.

II. Roadmap

Localizing a vehicle on a roadmap is made by using a map-matching algorithm (see [15], for a survey). The general purpose of such algorithm is to identify the correct segment of the road on which the vehicle is traveling at a given time and to determine the position of the vehicle on such segment [16], [17]. In spite of their differences, the map-matching algorithms consider a linear representation of the road regardless of its width (see Fig. 1, for an illustration). Such representation leads to a confusion in determining the position of the vehicle among several roads, especially when it comes to parallel or near roads, such as the problem at Y-junctions [15]. Matching the road direction (see Fig. 2, for such direction) with that of the vehicle has improved the situation, but the problem of parallel roads close to the same direction remains unsolved.

AL-S^{2}m uses a special map-matching algorithm, which is based on representing the road as a polygon (see Table 1, for an illustration). The polygon representation is more coherent with the reality of the road than the linear one. Such representation leads to a more precise map matching. This section describes such representation and the different configurations, adopted in AL-S^{2}m. To simplify the discussion, we illustrate and summarize the design elements in equations.

AL-S^{2}m design uses the following scenario. We consider the vehicle as a mobile point V; see (1). Initially, AL-S^2 determines the coordinates of V, and locates V on a given road cR (Current road) in AL-S^{2}m using a map-matching algorithm. Then, AL-S^{2}m tracks the motion of V until it leaves cR. While tracking V, AL-S^2 searches if there is a TS on cR, which is in the visibility-field vf of the driver, to identify it. By leaving cR, V either leaves AL-S^{2}m entirely or enters a new cR to follow the same scenario.

$$V = \langle x, y, z \rangle \tag{1}$$

The rest of this section explains the representation of AL-S^{2}m elements (Subsection II-A) and its achievement (Subsection II-C), and then describes the used map-matching method (Subsection II-B).

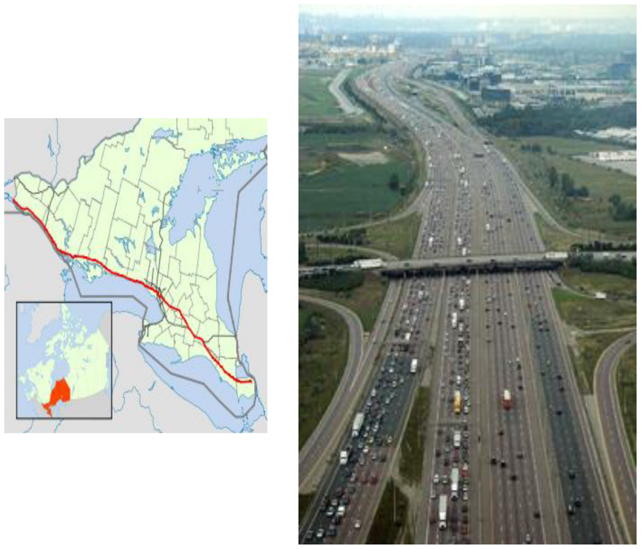


Fig. 1: Highway 401 within Ontario, Canada. In spite of its width, the road is represented in the map by a single line.

A. Representation of Roadmap Elements

AL-S^{2}m (see (2)) combines the topological localization with the geometric accuracy of metric methods. We consider AL-S^{2}m as a directed graph (N, E) where N is a set of nodes and E is a set of arcs. A node can be a road, an intersection or a roundabout. An arc from a node A to a node B specifies that when the vehicle leaves A, it can enter B. E will be distributed on the nodes of the graph such that each node A will have its own exit nodes E. For this, we associate B to A if there is an arc from A to B. Consequently, E will not appear in (2) but rather it appears in the road elements, see (3). AL-S^{2}m includes also the Vehicle (V) and the current Road (cR) on which V moves.

$$ALS2m = \langle N, V, cR \rangle \tag{2}$$

In the following subsections, we describe the different road configurations, and their virtualization in AL-S^{2}m: Single One-Way, Single Two-way, Multiple Channels and Interchange.

1) Single One-way Road: We represent a single one-way road R as a triplet; see (3), where:

- P: a polygon delimiting R,
- G: a rectangular cuboid (see explanation above),
- E: a set of roads, which are in exit relation with R.

$$R = \langle P, G, E \rangle \tag{3}$$

To take into account the fact that the road is not planar but rather a three-dimensional shape, we represent it as a skew polygon, referred below simply by polygon. A skew polygon is a polygon whose vertices are not all coplanar, see [19] for more details, and Fig. 3 for an illustration of a simple skew polygon.

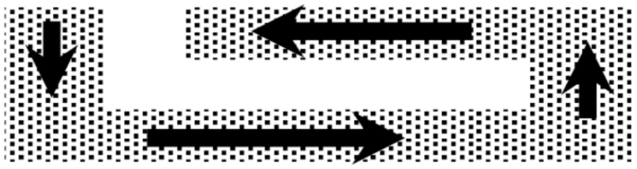


Fig. 2: A road with several directions that can be opposite. The arrows indicate the roads directions.

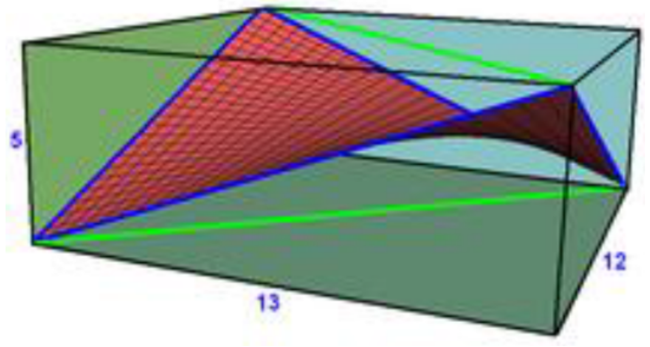


Fig. 3: An illustration of a skew polygon and its minimum bounding box.

In a polygon representation, we made the map matching simply by a point-in-the-box which determines if a given point is inside the box (rectangular cuboid) bounding the polygon (see Subsection II-B, for our map-matching method). For this reason, we add a rectangular cuboid (G) to (3). We compute G using an algorithm of a minimum bounding box. Such an algorithm proposed firstly in [19] and then improved in [13], and [4] describes a recent interesting heuristic. The description of such algorithms is outside this paper. Note that its performance is not important since we use it once at the map creation only.

For a road R with various directions, we consider it as a sequence of roads (four roads, in the example of Fig. 2), one road by direction. Each of them has its own direction. We made simply the transition between such roads by their exit relations E of (3).

For a curved road R (as illustrated in Fig. 4-a), we approximate it by a polygon, see Fig. 4.b. However, if such a form contains several directions we consider each portion of a given direction as a separate road and the road will consist of a sequence of roads, as shown in Fig. 4-c. As handled, we modeled easily parallel curved roads, as given in Fig. 4-d.

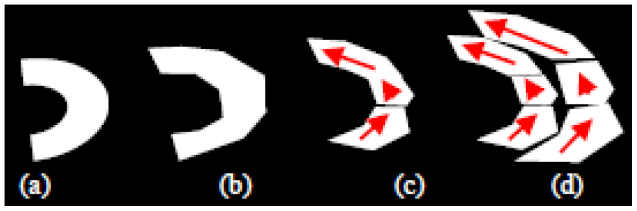
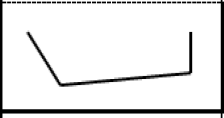
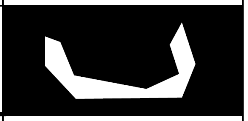
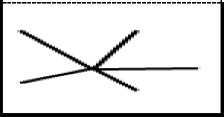
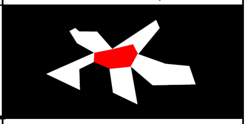
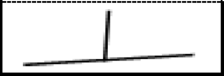
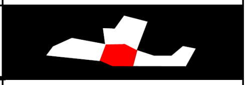
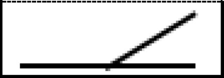
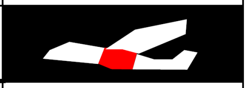
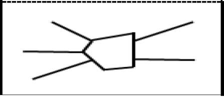
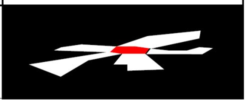


Fig. 4: Curved roads and their representation in AL-S^{2}m.

c

TABLE I: LINEAR (SPS) vs POLYGONAL (AL-S^{2}M) REPRESENTATIONS OF ROADS.

Road configuration	SPS	AL-S²m
Sample road	Three road segments as Lines	A polygon
		
Intersection	Five road segments	Six polygons including one for the intersection part (red colored)
		
T – junction	Three road segments	Four polygons including one for the intersection
		
Y – junction	Three road segments	Four polygons including one for the intersection
		
Roundabout	Ten road segments including five segments (constituting visually a polygon) constituting the roundabout	Five polygons (one for each road leaving the roundabout) and a sixth for the roundabout itself
		

2) Single Two-way Channel and U-turn Detection Problem: We consider the two-way road as two roads, R and its inverse R^{-1}. When the vehicle is in the direction of R and makes a U-turn, it leaves R and enters R^{-1}. The exit relation of R^{-1} will then include R and vice-versa, see (4).

$$\begin{aligned} R &= \langle P, G, \theta_R, E \cup \{R^{-1}\} \rangle \\ R^{-1} &= \langle P, G, \theta_R + \pi, E \cup \{R\} \rangle \end{aligned} \tag{4}$$

The definition of a vehicle V, given in (3), does not allow the vehicle to be aware of having made a U-turn. Then, we add a direction angle θ_R to the definition of the road R, as given in (4). Such angle for R^{-1} will be $\theta_R + \pi$. When V (with θ_V as its direction) is in R and the expression U ($U = (|\theta_V - \theta_R| > (\pi/2))$) is true, V has made a U-turn. Since U-turn uses the direction of the vehicle, we add θ_V to the definition

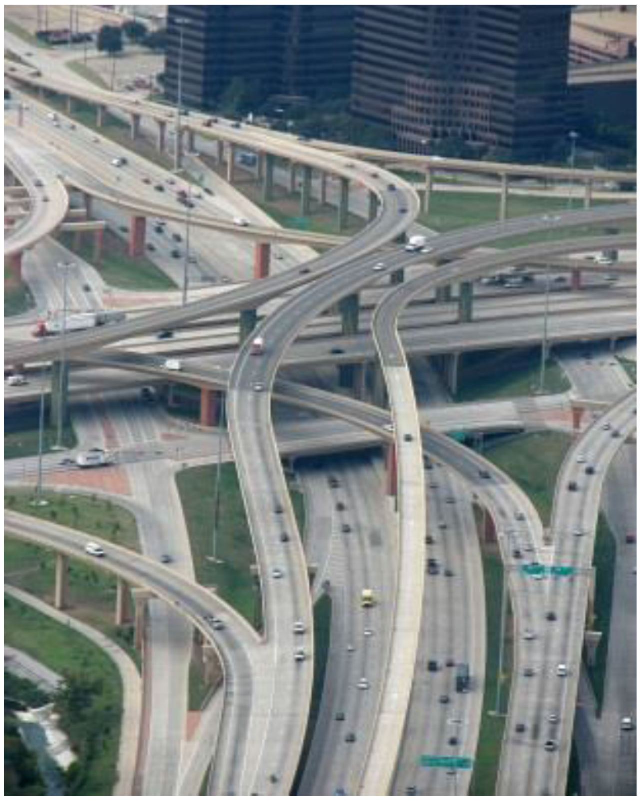


Fig. 5: A five-level interchange (The High Five Interchange in Dallas, Texas, United States).

of the vehicle of (1), as given in (5).

$$V = \langle x, y, z, \theta_V \rangle \tag{5}$$

When the vehicle makes a U-turn, it leaves R to enter R^{-1}. We search for R^{-1} in the exit roads of R ($E \cup \{R^{-1}\}$) as any other exit road of R. Note that the exit roads of R^1 (noted E^{-1}) are different from those of R; they are those of its opposite direction.

$$\begin{array}{l} C1 = \langle P_{C1}, G_{C1}, \theta_{C1}, E_{C1} \cup \{C2\} \rangle \\ C2 = \langle P_{C2}, G_{C2}, \theta_{C2}, E_{C2} \cup \{C1, C3\} \rangle \\ C3 = \langle P_{C3}, G_{C3}, \theta_{C3}, E_{C3} \cup \{C2\} \rangle \end{array} \tag{6}$$

3) Multiple Channels Road: We consider a multiple channels road essentially as a sequence of roads (channels). We made the transition between channels as that between other roads, by using the exit relation E of (4), i.e., E will include the side channels. For example, in (6), we give a representation of a road with three channels C1, C2 and C3, where C2 is in the middle.

4) Interchange: An interchange (see Fig. 5, for an illustration) is a road junction that typically uses grade separation, and one or more ramps, to permit traffic on at least one highway to pass through the junction without directly crossing any other traffic stream. In the case of a representation by a polygon, the vehicle V belongs to a given ramp RP before it will be above other ramps. Consequently, in AL-S^{2}m, we just check if V still belongs to RP without considering other ramps. This is not possible with the linear representation (especially in two-dimensions) where the map-matching is done by comparing the distances between the vehicle and the segments representing the ramps.

B. Roadmap Achievement

Several professional technics, methods, tools and devices are currently useful in determining the elements of the road, given in (4), such as LIDAR (Laser Interferometry Detection and Ranging) and roads aerial images. Such operations are not problematic tasks for road professionals. Their description is outside this paper. However, we describe in the experimentation (see Section V) a simplistic method that we used to determine polygons of roads. In addition to shaping the polygon delimiting an element of the road, we specify an approximation of the road direction (see (4)).

C. Map-Matching

Recall that in AL-S^{2}m:

- a vehicle (see (5)) is represented by a point (x_V, y_V, z_V) and a direction θ_V,
- a road element R (see (4)) is represented by: (1) a skew polygon (P), (2) surrounded by a rectangular cuboid (G), (3) its direction θ_R, and (4) roads E in exit relation with R.

A map matching consists in determining the road element of AL-S^{2}m to which belongs the point (x_V, y_V, z_V) representing the vehicle. Initially, we must seek the road element to which the vehicle belongs among all the road elements of AL-S^{2}m. However, when the vehicle leaves a particular road element R, it can enter another road element among those of its exit relation E. The map-matching algorithm will seek then such element among those of E only.

The problem of map matching consists in verifying if (x_V, y_V, z_V) belongs to a rectangular cuboid (G) of a given road, with the condition NU is true (NU=($\mid \theta_V - \theta_R \mid \leq \pi/2$)); i.e., the vehicle is in the road direction, not in its opposite. In the following paragraphs, we explain the map-matching method used in AL-S^{2}m in order to determine the rectangular cuboid to which the point (x_V, y_V, z_V) belongs; such method is an exercise of elementary geometry. We briefly describe it in the following paragraphs.

A plane P (see (7)) divides the space into three parts. The first comprises the points q with e_P (x,y,z)>0 and the second includes the points r with e_P (x,y,z)<0, while the third consists of the points p of the plane P itself with e_P (x,y,z)=0. In order for the point (x_V, y_V, z_V) to be located between two plans P1 and P2, it must verify the inequality given in (8).

$$e_P(x, y, z) = a_P \times x + b_P \times y + c_P \times z + d_P = 0 \tag{7}$$

$$(e_{P1}(x_V, y_V, z_V) \times e_{P2}(x_V, y_V, z_V)) < 0 \tag{8}$$

A rectangular cuboid is defined by six rectangular planes Pi of equations e_Pi (x,y,z), i $\in$ [1, 6]. A point (x_V, y_V, z_V)

belongs to a rectangular cuboid if it is located between the six planes Pi, i $\in$ [1, 6]. To determine whether the point (x_V, y_V, z_V) belongs to a given rectangular cuboid, we must determine that such point is between the parallel planes Pi, i $\in$ [1, 6]. Assuming that the parallel planes of the rectangular cuboid are arranged according to the following pairs: (P1, P2), (P3, P4) and (P5, P6). In this case, the point (x_V, y_V, z_V) belongs to such a road if (In(x_V, y_V, z_V)=true), see (9). Such map-matching method is very easy to implement, and consumes fewer resources, i.e.; it does not require a powerful device for its execution and responds timely.

$$In(x_V, y_V, z_V) = \begin{pmatrix} (e_{P1}(x_V, y_V, z_V) \times e_{P2}(x_V, y_V, z_V)) < 0 \\ \bigwedge(e_{P3}(x_V, y_V, z_V) \times e_{P4}(x_V, y_V, z_V)) < 0 \\ \bigwedge(e_{P5}(x_V, y_V, z_V) \times e_{P6}(x_V, y_V, z_V)) < 0 \end{pmatrix} \quad (9)$$

III. Traffic Signs

In this section, we describe a representation of traffic signs (Subsection III-A), their placement on the roadmap AL-S^{2}m (Subsection III-B), and the method used for their detection (Subsection III-C).

A. *Traffic Signs Representation*

A hard (material) traffic sign (HTS) has an image Im, it is fixed on a specific place with coordinates $(x_{HTS}, y_{HTS}, z_{HTS})$ on a given road R, and its effect starts from some straight segment that passes by HTS coordinates (see Fig. 7). HTS will be seeable by the driver at some distance d, from which its visibility-field (vf) starts. In the following, we describe the particular representation of TSs in AL-S^{2}m and then we show how to handle variable TSs. However, before considering TSs representation, we modify the road equation (see (4)) so it includes a list S of TSs that belongs to R, see (10).

$$R = \langle P, G, \theta_R, S, E\rangle \quad (10)$$

1) Traffic Sign Representation in AL-S^{2}m: In a similar way to HTSs, we can represent a soft TS by its coordinates H=$(x_{HTS}, y_{HTS}, z_{HTS})$ and its image Im. In such representation, different drivers cannot detect the TSs with the same distance from their effect area. For instance, in Fig. 7, only the vehicle V3 can detect the TS whereas V1 and V2 cannot in spite of their same distance from the beginning of the effect field of the TS. Therefore, the reaction of the drivers will not be the same at the same distance from the TS effect. Such imprecision is already a weakness of HTSs.

To overcome such detection problem, we represent the TS by straight segment ts, in two dimensions space, as illustrated in Fig. 7. The ts segment passes by the point H=$(x_{HTS}, y_{HTS}, z_{HTS})$.

In AL-S^{2}m, the point (x_V, y_V, z_V) representing the vehicle moves inside a rectangular cuboid G, representing the road, see Fig. 6. For a better coherence, we consider a TS as a plan, termed pTS, rather than a simple straight line. The pTS equation will be determined by considering the two following facts: (1) pTS contains the point H=$(x_{HTS}, y_{HTS}, z_{HTS})$ and (2) pTS is parallel to a given plane of the rectangular cuboid G, say P1 of equation e_{P1}(x,y,z)=0. This leads to the pTS equation, see (11). Finally, a soft TS will be represented by: (1) its image Im, and (2) the coefficients of pTS $co_{TS} = (a_{TS}, b_{TS}, c_{TS}, d_{TS})$, see (12), where we discarded the constant k.

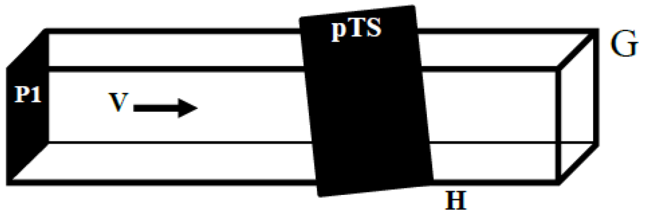


Fig. 6: The TS represented as a plan (pTS) in the rectangular cuboid G. The vehicle (point V) moves inside G (containing the road polygon) in a given direction from the plan P1.

$$\begin{aligned} e_{TS}(x, y, z) = k \times a_{P1} \times (x - x_{HTS}) + k \times b_{P1} \\ \times(y - y_{HTS}) + k \times c_{P1} \times (z - z_{HTS}) = 0 \end{aligned} \quad (11)$$

$$\begin{aligned} TS = \langle Im, co_{TS}\rangle = \langle Im, a_{TS}, b_{TS}, c_{TS}, d_{TS}\rangle \\ =< Im, a_{P1}, b_{P1}, c_{P1}, \\ -(a_{P1} \times x_{HTS} + b_{P1} \times y_{HTS} + c_{P1} \times z_{HTS}) > \end{aligned} \quad (12)$$

2) Traffic Signs Out of Their Roads: Some TSs must be seeable when the vehicle travels on a road element R while they belong to an element of its exit relation E, such as those signaling a roundabout. In AL-S^{2}m, we assign such TSs to the all roads where they must be visible, including the road where we made effectively the HTS.

3) Variable Traffic Signs: A variable TS (vTS) is a TS that changes according to some conditions (traffic, weather, time). vTSs can be seen as several conditioned simple TSs (sTSs) as follows: $(C_1 : sTS_1, \cdots, C_n : sTS_n)$, see (13). In the latter, each sTS$_i$ is represented by its image Im$_i$ while the TSs coordinates co$_{TS}$ are common to the all sTSs.

$$TS = \langle [C_1 : Im_1, \cdots, C_n : Im_n], co_{TS}\rangle \quad (13)$$

To handle the conditions, we associate to a vTS a behavior that verifies such conditions to display the image of a sTS$_i$ with the verified condition. We ranked vTS conditions into three categories:

1) Timed (such as tricolor lights, daytime, nighttime): The conditions are time intervals. AL-S^{2}m verifies the time (as a condition) and identifies the associated sTS,
2) Surrounding physical environment (such as traffic density, weather): It depends on the sensors available on the vehicle and the communication protocol between the vehicular system and AL-S^{2}m,
3) Remote (modifiable by transport authority): It depends on the communication protocol between the transport authority and AL-S^{2}m.

To take into consideration vTSs, we replace (12) of TS by (13). To illustrate the use of (13) in modelling the TSs, we

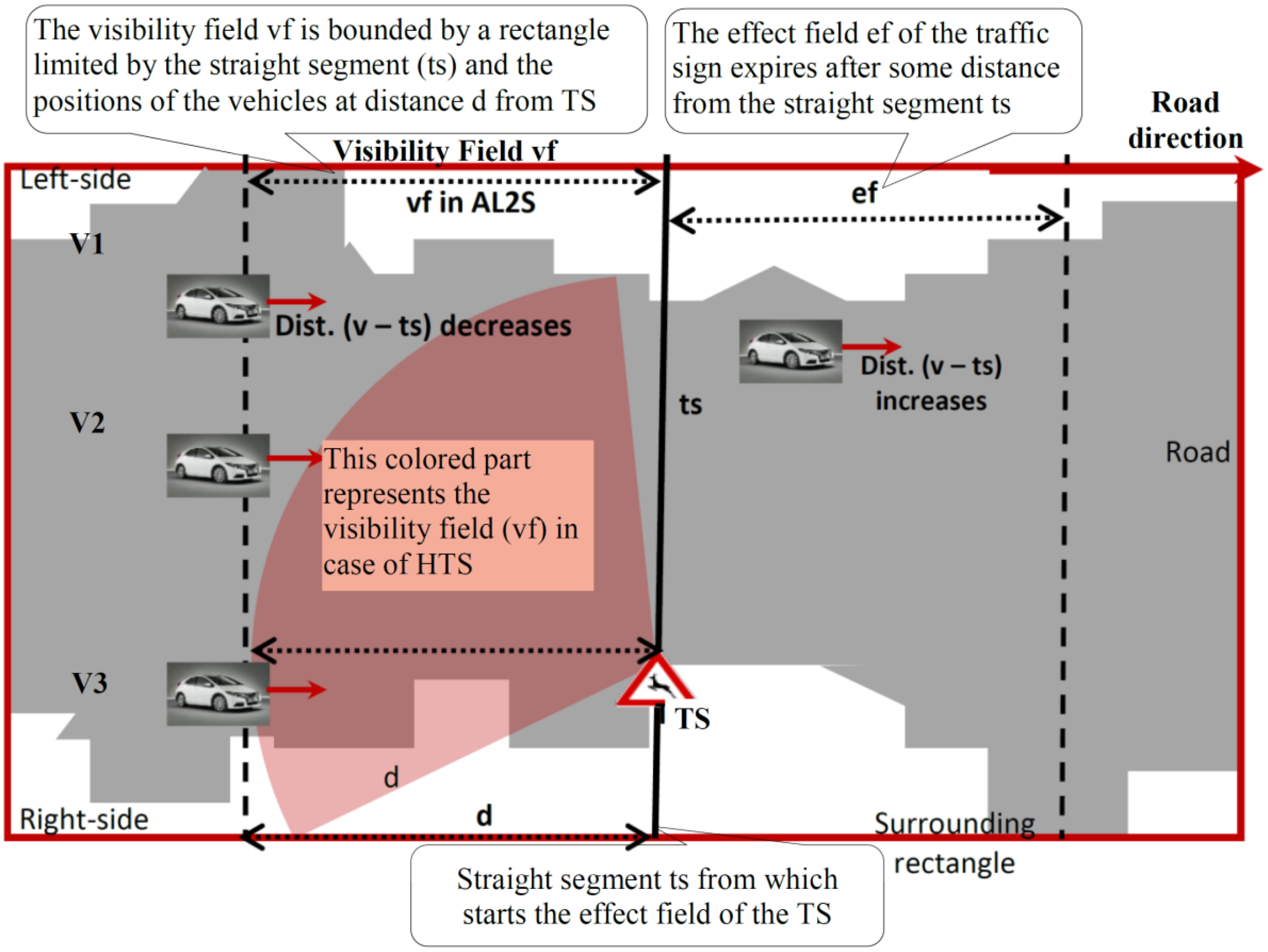

The visibility field vf is bounded by a rectangle limited by the straight segment (ts) and the positions of the vehicles at distance d from TS
The effect field ef of the traffic sign expires after some distance from the straight segment ts
Road direction
Visibility Field vf
Left-side
vf in AL2S
ef
V1
Dist. (v – ts) decreases
Dist. (v – ts) increases
ts
V2
This colored part represents the visibility field (vf) in case of HTS
Road
V3
TS
d
d
Right-side
Surrounding rectangle
Straight segment ts from which starts the effect field of the TS

Fig. 7: TS as a plan. The road and a TS projected into a two dimensions space. The road is colored in grey, the vf of the HTS is colored in orange and vf of the soft TS (in AL-S^{2}m) is indicated by a discontinuous arrow.

give two cases: simple TS (sTS) and a Tricolor Lights (TL). For simple TS, see (14), it is sufficient to put with the TS image the true value as its condition.

$$sTS = \langle [true : Im], co_{TS} \rangle \tag{14}$$

The tricolor lights TL modeling needs some explanation. The TL sign changes regularly over the time. For this aim, we take into consideration the starting time tg (hour, minute and second) of the green light. Let us consider the transition time between colors as follows: gy (green-yellow), yr (yellow-red) and rg (red-green). The AL-S^{2}m system determines the TL color (TS image) at time t according to the conditions given in the equation of TL; see (15).

$$TL = \left\langle \begin{bmatrix} (0 \leq (t-t_g)mod(gy+yr+rg) \\ < gy) : green, \\ (gy \leq (t-t_g)mod(gy+yr+rg) \\ < gy+yr) : yellow, \\ (gy+yr \leq (t-t_g)mod(gy+yr \\ +rg) < gy+yr+rg) : red \end{bmatrix}, co_{TS} \right\rangle \tag{15}$$

B. Placing Traffic Signs on the Roadmap AL-S^{2}m

In AL-S^{2}m, we can place a TS by two ways (see Subsection V-B, for more information). In the first (made on the ground), a traffic agent equipped with AL-S^{2}m system (hardware and software) moves to the TS location and requests AL-S^{2}m system to put such TS. AL-S^{2}m system proceeds then as follows: (1) it determines the position of the traffic agent (the TS coordinates) using a positioning system, (2) it determines the road R to which the TS belongs using the map-matching algorithm, and (3) it inserts the TS in AL-S^{2}m.

To insert a TS, AL-S^{2}m system: (1) requests TS information from the traffic agent (TS image, time of green light if any, $\cdots$), (2) computes the coefficients of the plan equation associated to the TS (co$_{TS}$, see (12)), and, finally, (3) adds TS information to AL-S^{2}m.

In the second way (made directly on AL-S^{2}m), an agent specifies a given road R and the information allowing to determine the coordinates of TS, such as a distance from a given point in R. Then, AL-S^{2}m determines the coordinates where it must place the TS. Finally, it inserts such TS in AL-S^{2}m as described in the previous paragraph.

C. Detecting Traffic Signs

AL-S^{2}m system detects a given TS when the vehicle is located in the visibility-field vf of such TS. vf is specified by the distance d (see Fig. 7) between the vehicle (point (x_V, y_V, z_V)) and the plan TS. We continue considering the detected TS while d decreases (the vehicle approaching TS). When d starts increasing (the vehicle moves away from TS), then the detected TS can be forsaken.

TSs detection involves two parts: tracking the vehicle and displaying the TS. For the first part, the vehicle, equipped with AL-S^{2}m, starts from a point SVP and moves on the road. AL-S^{2}m system determines the SVP coordinates using a positioning system. Then, AL-S^{2}m follows the movement of the vehicle and determines its successive positions following a dead reckoning process (see [2], for dead reckoning). However, such process has the disadvantage of errors accumulation that requires correction of the position for its use over long distances. For this aim, we used sensors fusion to correct the position of the vehicle. In this paper, we focus on the roadmap. The correction of the position is then outside this paper.

To detect a TS, AL-S^{2}m system determines firstly the road R on which the vehicle is located (to which belongs the starting point SVP). Each road R contains a list of TSs (rTSs) ordered according to their distances from a given side, say P1, of the rectangular cuboid G of the road R. P1, see Fig. 6, is the side situated at the opposite of the road direction θ_R.

AL-S^{2}m system determines then the first TS in the visibility field of the vehicle, in order to display it. We must search such first TS, as explained in the next paragraph. When a given TS is displayed, AL-S^{2}m system takes the next TS from rTSs to do the same as the previous TS and so on.

To seek the first TS, we use the following process. When the vehicle enters a road element (road, intersection or roundabout) coming from another element, we consider the first TS of rTSs and check if it is in the visibility field of the vehicle. We continue the verification until such TS becomes in the visibility field where we display it. It is not necessary to check other TSs since the vehicle must enter firstly the visibility field of the first TS of the list rTS. However, when the vehicle enters a road element from the outside of AL-S^{2}m, we must seek the nearest TS, which can be not the first element of rTSs since the vehicle can enter between TSs. To search such nearest TS, we follow the process given in Fig. 8. In the next paragraphs, we give some clarifications on such process.

At time intervals dt, the vehicle crosses different successive points (positions). We termed two successive points by old position (OP) and new position (NP). For readability reason, we shortened by TS the plan containing the TS (pTS).

For a given TS, if the distance d(NP, TS) between NP and TS is less than d(OP, TS) between OP and TS, i.e., the distance between the vehicle and such TS decreases, the vehicle approaches the TS. In this situation, the current TS is ahead of the vehicle and the searched TS is found.

In contrast, if the distance between the vehicle and TS increases, the vehicle moves away from TS. In such situation, the current TS is behind the vehicle and we must search for another TS.

A special case is when the distance between the vehicle and the TS remains unchanged. This happens when the vehicle is stopped or moves in parallel to TS. In such situation, we cannot decide. We will then get a new position of the vehicle after dt and then reiterate the same process.

To compute the distances d(OP, TS) and d(NP, TS), we use the formula of elementary geometry of a distance between a given point and a plan. For d(OP, TS), we obtain the equation (16), in which we can substitute aTS, bTS, cTS and dTS by their associated expressions, as given in (12), containing the coefficients of pTS. We can obtain the d(NP, TS) expression by substituting OP by NP.

IV. Implementation

To implement AL-S^{2}m, we used an Android device, endowed with accelerometers, gyro-meters and a GPS sensor. Android is a Java based language. The SensorManager class of API of Android sensors allows registered applications to receive data from sensors. After registration, changed data of the sensor are sent to SensorEventListener as a SensorEvent containing information generated from the given sensor (for Android programming and sensors use, see its official web site http://developer.android.com/ (Accessed 2015 11 26)).

For AL-S^{2}m, we identified four main object classes (AL-S^{2}m, R, V and TS). We already described their attributes in the associated equations. We recall them in (17).

Here, we describe some significant aspects of implementing such classes according to the following AL-S^2 implementation scenario. AL-S^{2}m identifies the current road cR on which the vehicle runs initially (see Subsection II-B) and then passes the control to cR by invoking the method cR.in. The latter tracks the vehicle in cR by verifying if it belongs to the cuboid rectangle G of cR. While the vehicle is in cR, cR.in checks whether the vehicle is on vf of some TS to detect it (see Subsection III-C). After leaving cR by the vehicle, cR.in returns the control to AL-S^{2}m. This latter seeks another road cR and iterates such process until the vehicle leaves completely AL-S^{2}m.

We implemented the vehicle by a thread. Its run() method continually updates the coordinates of the vehicle, by using the accelerometers of the Android device, and its direction, by using its gyro-meters.

Both AL-S^{2}m and cR objects use the coordinates of the vehicle and its direction. Several methods of TS and R can also access the vehicle attributes. Then, we declared the getters of

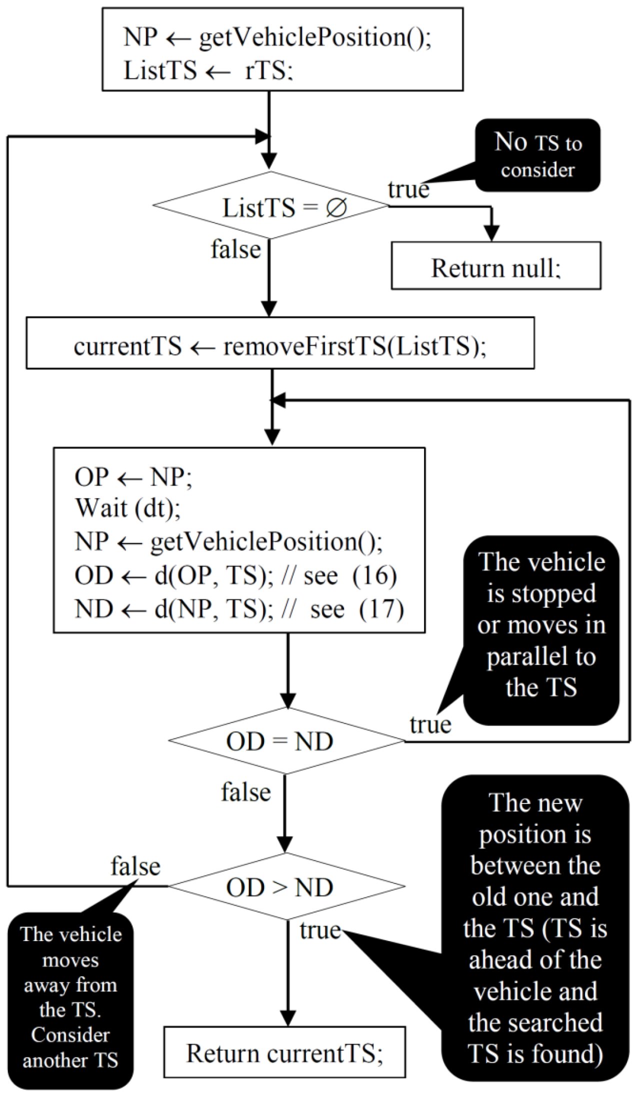

NP ← getVehiclePosition();
ListTS ← rTS;
No TS to consider
true
ListTS = ∅
false
Return null;
currentTS ← removeFirstTS(ListTS);
OP ← NP;
Wait (dt);
NP ← getVehiclePosition();
OD ← d(OP, TS); // see (16)
ND ← d(NP, TS); // see (17)
The vehicle is stopped or moves in parallel to the TS
true
OD = ND
false
The new position is between the old one and the TS (TS is ahead of the vehicle and the searched TS is found)
false
OD > ND
true
The vehicle moves away from the TS. Consider another TS
Return currentTS;

Fig. 8: Seeking the first TS. The symbols used in the diagram are well known (rectangle: processing; lozenge: testing; arrow: control flow). However, we added a bullet symbol for comments.

the vehicle (getx(), gety(), getz(), get-teta(), getvx(), getvy(), getvz()) as synchronize.

We also implemented TS as a Java thread. Its run() method allows displaying and dropping the TS image.

$$d(OP,TS) = \frac{|x_{OP} \times a_{TS} + y_{OP} \times b_{TS} + z_{OP} \times c_{TS} + d_{TS}|}{\sqrt{(a_{TS}^2 + b_{TS}^2 + c_{TS}^2 + d_{TS}^2)}} \tag{16}$$

$$\begin{array}{l} ALS^2m =< N, V, cR > \\ R =< P, G, S, \theta_R, E > \\ V =< x, y, z, \theta_V > \\ TS = \langle [c_1 : i_1 \cdots c_n : i_n], co_{TS} \rangle \end{array} \tag{17}$$

V. Experimentation

As mentioned in Subsection III-B, several professional techniques, methods, tools and devices are currently useful in determining the road elements. Such operations are not problematic tasks for road professionals. To show the easiness of AL-S^{2}m creation, we describe a simplistic experimentation for AL-S^{2}m creation. AL-S^{2}m consists of two parts: (1) delimitation of the nodes or road elements (simple roads, roundabouts and intersection) as polygons (described in Subsection V-A), (2) and the placement of TSs on AL-S^{2}m (given in Subsection V-B).

A. Delimitating Road Elements

For delimitating a road element (node), we operate with a pedestrian to bind such a road. A pedestrian equipped with AL-S^{2}m system (Android device endowed with AL-S^{2}m software) operates as follows: (1) starts from an initial point SP on the road edge, (2) moves then to surround the road, (3) to return finally to the starting point SP. AL-S^{2}m system tracks the pedestrians movement and saves the coordinates of its different positions at time intervals. Firstly, AL-S^{2}m determines the coordinates of the point SP using GPS (Global Positioning System). Then, using the accelerometers and the gyro-meters of the android device, AL-S^{2}m system uses a dead reckoning process to compute, and record, the coordinates of such positions. These positions constitute the vertices of the polygon surrounding the road.

The main challenge in tracking by using the dead reckoning lies in the inaccuracies of the values provided by MEMS sensors especially when the transporter of the device is a pedestrian. In our case, we used an approach based on the pedestrian steps. The traditional approach to measure distances using pedestrian steps uses a constant length of such steps. It counts the number of the steps and multiplies it by a given constant measure (one meter, for instance). Such approach increases the errors in the computed distances especially with false steps. To detect steps, such approaches use the accelerometers variations. However, some variations (when sensors return non-zero values whereas there is no movement) are not causes of steps (see Fig. 9).

In contrast, AL-S^{2}m detects false steps. It considers only the variations of the accelerations between two specified values (noted top and bottom, in Fig. 9). The top and bottom values depends on the used device that must be determined by experimenting such device. AL-S^{2}m discards the variations that do not spend from bottom to top as false steps.

In addition, AL-S^{2}m uses steps with variable lengths. It does not count the number of steps to multiply it by a constant length but it rather computes the length of each step (See Fig. 10) and sums such lengths. To compute the step length, we use the values given by the accelerometers and the time interval dt.

Steps with variable lengths and the detection of false steps reduced significantly the errors in the pedestrian path. Minimizing the errors in addition to the small distances covered by the pedestrian associated to the fact that our application does not require high precision, led to acceptable results (see Fig. 10).

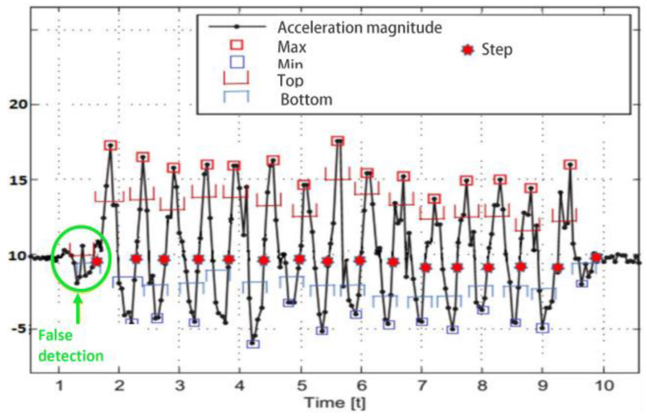


Fig. 9: Steps detection by an Android device. A higher acceleration (greater than or equal to Top) followed by a lower one (less than or equal to bottom) indicates a one-step whereas other variations between such values indicate false steps.

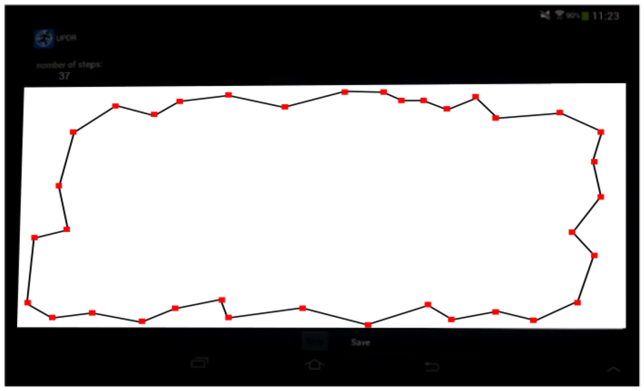


Fig. 10: A screenshot (Tablet with Android operating system) containing a projection of the skew polygon into two dimensions: each two consecutive red points represent a side of the polygon. Note that, at the closing point of the polygon, we have a relatively long segment, which is due to cumulating errors of the sensors.

B. Placement of Traffic Signs

Placing a given TS in AL-S^{2}m uses two interfaces (see Subsection III-B). For the first, a traffic agent moves to a position on the right side of a given road and requests AL-S^{2}m (by typing on the Android device equipped with AL-S^{2}m system) to insert the given TS. By using GPS, AL-S^{2}m determines the position of the traffic agent (indeed, that of the device) which will constitute the TS coordinates as in the case of the hard TS (HTS). Such position allows determining the road R to which belongs TS. Finally, the system inserts the given TS in AL-S^{2}m.

To insert a TS, AL-S^{2}m firstly derives the coefficients of the plan TS (pTS, see (13) and Fig. 6) using the TS coordinates and the rectangular cuboid (G) of R (see (3)). Then, AL-S^{2}m displays a list of TSs and requests the traffic agent to choose the one he (she) would insert. Then, it inserts, in AL-S^{2}m, the new TS as part of R. If the traffic agent chooses a variable TS, AL-S^{2}m demands the parameters of such TS: the conditions and the associated simple TSs. Otherwise, it adds the true value as the TS condition.

The second is made directly on AL-S^{2}m. An agent requests AL-S^{2}m to insert a given TS at a given position of a road R. The traffic agent can specify a distance from a given point (start, end) in R. AL-S^{2}m determines then the coordinates where it must insert TS. It inserts then such TS into AL-S^{2}m as described in the previous paragraph (deriving the coefficients of pTS, choosing a TS, specifying the conditions).

VI. Conclusion and Further Work

In this paper, we described and discussed AL-S^{2}m, a specific roadmap. We presented a virtualization of the road elements. Also, we proposed an original map-matching method and improved TSs detection by an original representation of TSs as plans. Finally, we showed how one can implement AL-S^{2}m, and we indicated how it can create AL-S^{2}m. Our experiments show that AL-S^{2}m implementation is easy.

Creating and exploiting as well as updating AL-S^{2}m are simple, thanks to the simplicity of the used representation (a topology of polygons). The exploitation of AL-S^{2}m needs a simple device with accelerometers, gyro-meters and GPS sensors only. Unlike other roadmaps, AL-S^{2}m takes into account variable TSs and greatly improves the map-matching thanks to the road representation as a polygon. The simplification of AL-S^{2}m updates and the possibility of timely broadcasting such updates allow also to take into account temporary TSs.

In this paper, we have not detailed the AL-S^{2}m dissemination. We intend to explore such problem shortly.

Acknowledgment

The author would like to thank Mohamed Amine LAAMARI, Yasser Moussa BERGHOUT, Mohamed Salah ABDOU and Walid Guerza that implement some parts of the experimentation as part of their master dissertations. I would like also to thank Yazid LAHLOUHI for helping me correcting some mistakes.

References

[1] Bahlmann, C.; Zhu, Y.; Ramesh, V.; Pellkofer, M.; & Koehler, T. A System for Traffic Sign Detection, Tracking and Recognition Using Color, Shape, and Motion Information Proceedings of the IEEE on Intelligent Vehicles Symposium, 6-8 June 2005 pages 255-260, 2005.

[2] Borenstein, J. & Feng, L. Measurement and correction of systematic odometry errors in mobile robots IEEE Transactions on Robotics and Automation 12(6) pages 869-880, 1996.

[3] Broggi, A.; Cerri, P.; Medici, P.; Porta, P. P. & Ghisio, G. Real time road signs recognition Proceedings of IEEE Intelligent Vehicles Symposium, Istanbul, Turkey, June 13-15, 2007 pages 981-986, 2007.

[4] Chang, C. T., Gorissen, B., & Melchior, S. Fast oriented bounding box optimization on the rotation group SO (3,) ACM Transactions on Graphics 30(5) pages 122:1-122:16, 2011.

[5] de la Escalera, A.; Moreno, L. E.; Salichs, M. A. & Armingol, J. M. Road traffic sign detection and classification IEEE Transactions on Industrial Electronics 44(6) pages 848-859, 1997.

[6] Fang, C.-Y.; Chen, S.-W. & Fuh, C.-S. Road-Sign Detection and Tracking IEEE Transactions on Vehicular Technology 52(5) pages 1329-1341, 2003.

[7] Jamshidi, H.; Lukaszewicz, T.; Kashi, A.; Berghuvud, A.; Zepernick, H. & Khatibi, S. Fusion of digital map traffic signs and camera-detected signs Proceedings of 5th International Conference on Signal Processing and Communication Systems (ICSPCS), Outrigger Waikiki on the Beach, Honolulu, HI, USA, December 12-14, 2011 pages 1-7, 2011.

[8] Kim, J.; Kim, S.; Lim, K.; Choi, Y. & Byun, H. Real-time traffic sign detection with vehicle camera images Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication (ICUIMC13), Kota Kinabalu, Malaysia, January 1719, 2013 pages 20:1-20:4, 2013.

[9] Lauer, M. & Stein, D. A Train Localization Algorithm for Train Protection Systems of the Future IEEE Transactions on Intelligent Transportation Systems 16(2) pages 970-979, 2015.

[10] Ficco, M.; Palmieri, F. & Castiglione, A. Supporting First Responders Localization during Crisis Management The 9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, Blumenau, Brazil, July 8-9, 2015.

[11] Mathias, M.; Timofte, R.; Benenson, R. & Van Gool, L. Traffic sign recognition How far are we from the solution? Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN13), Dallas, USA, August 4-9, 2013 pages 1-8, 2013.

[12] Miura, J.; Kanda, T.; & Shirai, Y. An active vision system for real-time traffic sign recognition Proceedings of the IEEE conference on intelligent Transportation Systems, Dearborn, USA, October 1-3, 2000 pages 5257, 2000.

[13] OROURKE, J. Finding minimal enclosing boxes International Journal of Computer & Information Sciences, 14(3) pages 183199, 1985.

[14] Paclk, P.; Novoviov, J.; Pudil, P. & Somol, P. Road sign classification using Laplace kernel classifier Pattern Recognition Letters 21(13-14) Pages 1165-1173, 2000.

[15] Quddus, M. A.: Ochieng, W. Y. & Noland, R. B. Current map-matching algorithms for transport applications: State-of-the art and future research directions Transportation Research Part C: Emerging Technologies 15(5) pages 312-328, 2007.

[16] Quddus, M. A.; Ochieng, W. Y.; Zhao, L. & Noland, R. B. A general map matching algorithm for transport telematics applications GPS solutions 7(3) pages 157-167, 2003.

[17] Smaili, C.; El Najjar M. E. & Charpillet F. A Road Matching Method for Precise Vehicle Localization Using Hybrid Bayesian Network Journal of Intelligent Transportation Systems 12(4) pages 176-188, 2008.

[18] Tao, Z.; Dian-Ge, Y.; Jiang-Tao L. & Xiao-Min, L. A Trajectory-Based Map-Matching System for the Driving Road Identification in Vehicle Navigation Systems Journal of Intelligent Transportation Systems: Technology, Planning, and Operations, Article in Press, DOI: 10.1080/15472450.2015.1015847, 2015.

[19] Williams, R. The Geometrical Foundation of Natural Structure: A Source Book of Design Dover Publications, Inc., pages 34-51, 1979.