Novel Techniques to Handle Rectangular Areas in Car-to-X Communication Applications

Attila Jaeger and Sorin A. Huss
Integrated Circuits and Systems Lab, Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt, Germany

Keywords: Car-to-Car communication, Rectangular Area.

Abstract: Car-to-X communication is one of the main technologies to enable a various set of ITS use-cases. Thereby, among others, messages about local dangers are exchanged. According to message specification, the relevance area of such dangers may be encoded as a rectangle. In this work we present several techniques to handle such rectangular areas. These techniques are already included and, hence, extensively tested in the Weather Hazard Warning applications in current vehicle communication field operations tests in Germany and Europe.

1 INTRODUCTION

Car-to-X (C2X) communication, the combination of Car-to-Car (C2C), Car-to-Infrastructure (C2I), and Infrastructure-to-Car (I2C) communication, is considered as one of the most important technologies to improve traffic efficiency and traffic safety in the near future (C2C-CC, 2007). Such vehicle communication, i. e., based on 802.11p (IEEE, 2010), is the basic technology for various ITS use-cases (ETSI, 2010b). Although, current development of applications and system components is quite far and multiple field operational trials are finished, research is still needed until upcoming market launch.

In C2X communication, the Decentralized Environmental Notification Message (DENM) (ETSI, 2010a) is the major message to notify ITS participants about the presence of a specific and potential dangerous situation nearby.

Since these situations are relevant in a bounded geographical region, vehicle communication utilizes geo-cast (C2C-CC, 2007), a location based addressing scheme, whereby messages are forwarded to a specific geographical area and delivered to every ITS station within that destination area. Thereby, the area might be defined as a rectangle. Additionally to the destination area, a relevance area, which might also have rectangular shape, is defined in DENMs. In contrast to, e. g., circular relevance areas, rectangles may cover more precisely the shape of a road.

While handling of circular areas is relatively common and related techniques are well-known in the vehicle communication community, handling of rectangular areas is rather unfamiliar. Hence, in this work we collect known and present new techniques for the major basic operations a C2X application has to solve while handling rectangular areas. These operations are

1. to define a rectangle around a given set of points, e. g., a trace,
2. to determine if a given position is located inside or outside a rectangle,
3. to calculate the size of the overlapped area of two partial overlapping rectangles, and
4. to achieve the distance between a point, e. g., the vehicles current position, and the nearest border of a rectangle.

To demonstrate the usage of the introduced rectangular arithmetic by means of an application example, we utilize the Weather Hazard Warning application as presented in (Stübing and Jaeger, 2010) and more detailed in (Jaeger and Huss, 2011), since this C2X application uses rectangular areas to determine the location of the weather situation in DENMs and within function internals.

2 REPRESENTING RECTANGLES AND COORDINATES

While a circular area is commonly defined by a center point and a radius according to (ETSI, 2010c), rect-

Jaeger A. and A. Huss S. (2013).
Novel Techniques to Handle Rectangular Areas in Car-to-X Communication Applications.
In *Proceedings of the 10th International Conference on Informatics in Control, Automation and Robotics*, pages 575-582
DOI: 10.5220/0004633205750582

angles in DENMs, i. e., the destination and relevance area, each are defined by the following parameters:

M	the center of the rectangle
a	the half of the length (distance from the center to the shorter side)
b	the half of the width (distance from the center to the longer side)
θ	the clockwise rotation against north (azimuth angle).

Thereby, M is given as a WGS-84 coordinate, a and b are given in meters, and θ is an angle in degrees. With these parameters, not only axis-oriented, but also arbitrarily oriented rectangles can be defined. Such a rectangle and its parameters is depicted in Figure 1.

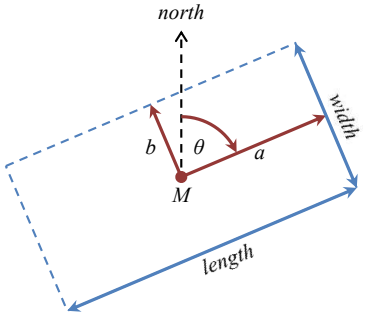

Figure 1: Parameters of a rectangle as given in a DENM.

As mentioned, points in DENMs are given in WGS-84 coordinates. However, for most mathematical operations, it is advantageous to deal with coordinates in a 2-dimensional Cartesian coordinate system. Additionally, units within the coordinate system should be interpreted as meters, which simplifies transferring sizes, like distances, lengths, and areas, into the real world. Finally, such a coordinate system can be interpreted as a two dimensional vector space $\mathbb{R}^2$ by using the coordinates of each point as components of the position vector for that point.

For example, the UTM coordinate system, whereas easting and northing can be interpreted as x- and y-coordinates, respectively, is a possible choice.

However, handling of UTM coordinates is rather challenging near the borders of UTM zones. Hence, a possible solution is to use the distance from Greenwich and the Equator as x- and y-coordinates, respectively. Thus, assuming a radius r of the earth of, e. g. 6371007.2 meters, a WGS-84 coordinate with longitude λ and latitude φ in radians is converted into a point with coordinates (x,y) by applying

$$x = r \cdot \arccos\left(\sin\varphi^2 + \cos\varphi^2 \cdot \cos\lambda\right) \quad (1)$$

$$y = r \cdot \varphi \quad (2)$$

whereas the sign of longitude has to be applied to x. Consequently, such a point is converted back into a WGS-84 coordinate by applying

$$\varphi = \frac{y}{r} \quad (3)$$

$$\lambda = \arccos\left(\cos\frac{x}{r} - \frac{\sin\varphi^2}{\cos\varphi^2}\right) \quad (4)$$

and, respectively, applying the sign of x to longitude. This coordinate system, of course, cannot be used in a global manner, but in local scale, as needed for C2X applications. It provides, in addition to fast position transformation, high accuracy.

Hence, for the following, we assume that points are given in or converted into UTM coordinates, or any other coordinate system, that meets the mentioned requirements.

3 FINDING A RECTANGLE SURROUNDING A GIVEN SET OF POINTS

Many dangerous situations, e. g., a critical weather situation such as heavy rains, are spread over a large geographical area. Whenever a C2X application detects such a situation the affected area has to be determined.

An intuitive way to do so is to record the driven way by means of waypoints while passing through the situation. After the whole situation is passed or even after a certain time, a rectangle has to be calculated that includes all of these waypoints. In Figure 2 such a use-case is depicted. This rectangle can be used as relevance area in a corresponding DENM for notifying other ITS stations about the detected situation. However, an appropriate destination area is, e. g., given by just adding a suitable distribution offset to a and b, respectively.

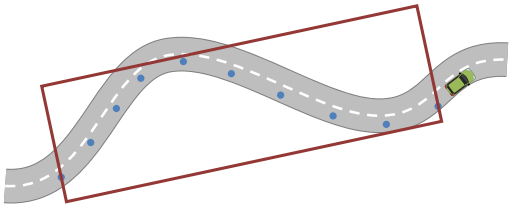

Figure 2: A rectangle surrounding a recorded trace.

To calculate a rectangle around a given set of points, the following steps have to be performed:

1. Since the sets of points may follow the course of a street, it is most likely that they are ordered in a line. Hence, the orientation of that chain of points has to be determined.

2. Since the line will not be perfectly strait, the width of the surrounding rectangle has to be calculated.
3. Afterwards, the length of the rectangle has to be determined.
4. Finally, out of these data, the parameters M, a, b, and θ as used in a DENM are to be achieved.

These steps are detailed in the following.

To obtain the main orientation of the arbitrary set of points, a regression line through all points can be calculated by using the method of least squares.

Thereby, for n points $P_i = (x_i, y_i)$ a regression line g is defined by

$$g : y = m \cdot x + \gamma \tag{5}$$

If $\overline{x}$ and $\overline{y}$ are the means of the x- and y-coordinates of all given points, respectively, m can be obtained by applying

$$m = \frac{\sum_i^n (x_i y_i) - n \cdot \overline{xy}}{\sum_i^n (x_i^2) - n \cdot \overline{x}^2} \tag{6}$$

and γ by

$$\gamma = \overline{y} - m\overline{x} \tag{7}$$

The main orientation of the points is now in the direction of this line, with the regression line in the middle of the set, as depicted in Figure 3.

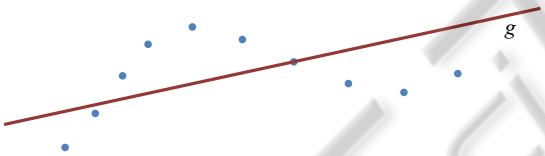

Figure 3: Regression line g in the middle of a given set of points, e. g., a recorded trace.

To determine the width and the length of the rectangle we transfer the task into the vector space $\mathbb{R}^2$. Therefore, while using the numerator and denominator of m, the vector $\vec{v}$ is given by

$$\vec{v} = \begin{pmatrix} -(\sum_i^n (x_i^2) - n \cdot \overline{x}^2) \\ \sum_i^n (x_i y_i) - n \cdot \overline{xy} \end{pmatrix} \tag{8}$$

and thus the vector $\vec{n}$ by

$$\vec{n} = \frac{\vec{v}}{|\vec{v}|} \tag{9}$$

Hence, with constant c given by

$$c = \begin{pmatrix} 0 \\ \gamma \end{pmatrix} \cdot \vec{n} \tag{10}$$

the distance d from an arbitrary point P, with position vector $\vec{P}$, to line g can be achieved by

$$d = \left| \vec{P} \cdot \vec{n} - c \right| \tag{11}$$

Thus, the width of the rectangle is the double of the largest distance d_i of any point P_i of the given set of points to the regression line g.

To achieve the length of the rectangle, we calculate for every point P_i the corresponding point F_i given by

$$\vec{F}_i = \vec{P}_i - d \cdot \vec{n} \tag{12}$$

Hence, the length of the rectangle is equal to the length of the vector $\vec{u}$ given by

$$\vec{u} = \vec{F}_{max} - \vec{F}_{min} \tag{13}$$

whereas, F_{max} and F_{min} are the two points with maximal and minimal x-component values, respectively, as depicted in Figure 4.

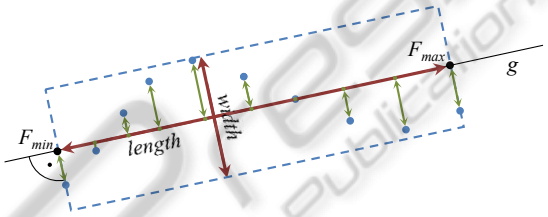

Figure 4: Calculation of a surrounding rectangle.

Since the rectangle is known now, the parameters as used in a DENM can be calculated by

$$M = F_{min} + \frac{1}{2} \cdot \vec{u} \tag{14}$$

$$a = \frac{1}{2} \cdot length \tag{15}$$

$$b = \frac{1}{2} \cdot width \tag{16}$$

$$\theta = \left(\frac{\pi}{2} - \text{atan2}(u_2, u_1) \right) \cdot \frac{180}{\pi} \tag{17}$$

whereas it has to be ensured that $0 \leq \theta < 360$ holds, by adding 360 to θ if θ is negative. Additionally, if needed, the corners E_1 to E_4 of the rectangle can be calculated by each adding and subtracting $d_i \cdot \vec{n}$ to F_{min} and F_{max}, respectively.

Although the presented algorithm is motivated by the need to calculate a rectangle around a recorded trace, the steps are applicable for any arbitrary set of points.

4 VERIFYING THE POSITION OF A POINT

In many cases a C2X application needs to calculate, if some arbitrary point, e. g., the vehicles position, is located inside or outside a given area.

If the area is, e. g., a circle, it has to be verified whether the distance to the center of the circle

is smaller or larger than the radius of the circle. However, C2X use-cases like, e. g., the Weather Hazard Warning application, uses arbitrarily oriented rectangles, whereas the mentioned calculation is not given by just a simple comparison. Consequently, in this section a method is presented to verify whether a given point P is located inside or outside a given rectangle.

According to Section 2, the point P is element of the two dimensional vector space $\mathbb{R}^2$. Hence, its components are given according to a standard orthonormal basis E with origin O and its two basis vectors

$$\vec{e}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \vec{e}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{18}$$

The basic idea behind the presented method is to transform the components of P into another coordinate system B, whereas the verification can be done by a simple comparison.

One possible solution for such a basis B is given by the rectangle and its parameters, i. e., M, a, b, and θ, to which the point has to be compared to. Thereby, the origin of B is set to the center M of the rectangle, whereas the two basis vectors

$$\vec{\beta}_1 = \begin{pmatrix} b_{11} \\ b_{21} \end{pmatrix}, \vec{\beta}_2 = \begin{pmatrix} b_{12} \\ b_{22} \end{pmatrix} \tag{19}$$

are oriented in parallel to the borders of the rectangle with a length of a and b, respectively. Figure 5 depicts such a basis B in comparison to E.

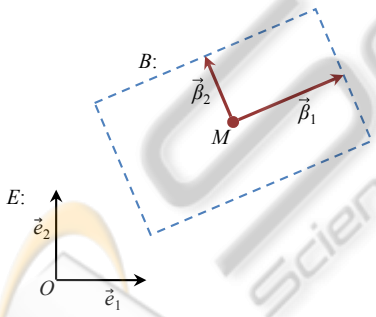

Figure 5: Standard orthonormal basis E in comparison to a basis B defined by a rectangle.

Now, the origin of basis B is given directly by the center point M, but the basis vectors $\vec{\beta}_1$ and $\vec{\beta}_2$ have to be determined accordingly.

Therefore, let φ be the radian of θ such that

$$\varphi = \theta \cdot \frac{\pi}{180} \tag{20}$$

Accordingly, the vector $\vec{v}$ is given by

$$\vec{v} = \begin{pmatrix} cos\varphi \\ sin\varphi \end{pmatrix} \tag{21}$$

Thus, the vectors $\vec{n}$ and $\vec{u}$ can be achieved by

$$\vec{n} = \frac{\vec{v}}{|\vec{v}|} \tag{22}$$

and

$$\vec{u} = \frac{-2 \cdot a \cdot \begin{pmatrix} -v_2 \\ v_1 \end{pmatrix}}{\left| \begin{pmatrix} -v_2 \\ v_1 \end{pmatrix} \right|} \tag{23}$$

respectively.

However, if the rectangle is generated as depicted in Section 3 according to the used nomenclature and definitions, vectors $\vec{n}$ and $\vec{u}$ are known already and do not have to be calculated again.

Despite the way $\vec{n}$ and $\vec{u}$ are achieved, the basis vectors of B are determined by applying

$$\vec{\beta}_1 = \frac{1}{2} \cdot \vec{u} \tag{24}$$

$$\vec{\beta}_2 = b \cdot \vec{n} \tag{25}$$

Consequently, to transform the position vector of an arbitrary point P given according to E into its equivalent in B, a transformation matrix T_B^E

$$T_B^E = \begin{pmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{pmatrix} \tag{26}$$

is needed. Since that matrix is identical to the inverse of the concatenation of the two basis vectors of B

$$T_B^E = \begin{pmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{pmatrix} = (\beta_1 \beta_2)^{-1} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}^{-1} \tag{27}$$

the system of equations

$$\vec{e}_1 = t_{11} \cdot \vec{\beta}_1 + t_{12} \cdot \vec{\beta}_2 \tag{28}$$

$$\vec{e}_2 = t_{21} \cdot \vec{\beta}_1 + t_{22} \cdot \vec{\beta}_2 \tag{29}$$

has to be solved to achieve the four components of the 2×2 transformation matrix T_B^E.

However, with q defined as

$$q = b_{11} \cdot b_{22} - b_{12} \cdot b_{21} \tag{30}$$

the solution is given by just applying

$$T_B^E = \frac{1}{q} \cdot \begin{pmatrix} b_{22} & -b_{21} \\ -b_{12} & b_{11} \end{pmatrix} \tag{31}$$

Hence, each point P, given according to E, can be transformed into B by multiplying the position vector with T_B^E.

Due to the definition of B, the point P is located inside the rectangle, if the absolute value of each of its components is lower or equal to 1

$$|p_1| \leq 1 \text{ and } |p_2| \leq 1 \tag{32}$$

with respect to B. Thus, it is located on the border of the rectangle if at least one of the absolute values of its components is equal to 1

$$|p_1| = 1 \text{ or } |p_2| = 1 \tag{33}$$

Otherwise, P is located outside of the rectangle.

5 OVERLAPPING AREA OF TWO RECTANGLES

Since many C2X applications, like the Weather Hazard Warning application, may detect and report situations successively, a receiving ITS station has to find such a relation in order to set up the whole event out of multiple DENMs, each concerning just a small part of the whole area. Additionally, since multiple ITS stations may detect the same situation independently from each other, neighboured ITS stations receive multiple messages regarding either the same event or the same part of the event. Such a situation is illustrated in Figure 6, whereas two vehicles successively report partially overlapping areas of the same event.

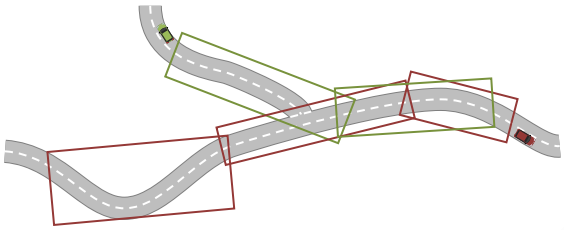

Figure 6: Multiple successively detected relevance areas of one large event reported by two ITS stations.

Thereby, according to the two mentioned cases, the areas may only overlap slightly, if they belong to a successive reporting or may heavily overlap in the case of the reporting from multiple ITS stations. Within C2X applications, this distinction has to be considered in order to decide whether the area of an event has to be expanded or if a situation is confirmed by another ITS station.

However, the algorithm used to define the overlapping area between two rectangles is the same in both cases mentioned above. Thereby, the following steps have to be performed.

1. Find all intersection points S_j of each of the four borders of one rectangle with each of the four borders of the other rectangle.
2. Find all corners E_j of each rectangle, that are located inside the other rectangle, respectively.
3. The set of all found corners and intersection points build the vertices of a non-self-intersecting polygon of the overlapping area if they are ordered the right way.
4. Finally, the area of the polygon can be calculated using the Gaussian trapezoid formula.

Such a polygon is illustrated in Figure 7, whereas the enumerated steps are detailed in the following.

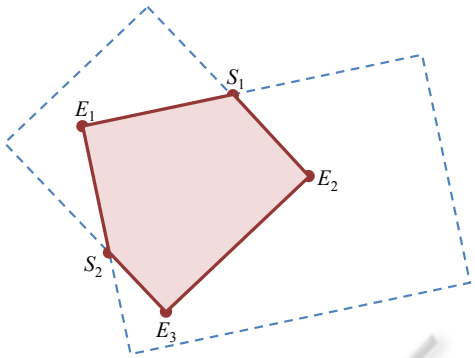

Figure 7: The overlapping area of two rectangles.

Finding the intersection points S_j of the borders of two rectangles is based on finding the intersection points of lines in basic geometry. Thereby, since every two points P and Q define a vector $\vec{u}$ with

$$\vec{u} = \vec{Q} - \vec{P} \tag{34}$$

and, thus, a line h with

$$h : \vec{x} = \vec{P} + \lambda \vec{u} \tag{35}$$

for each border of a rectangle, a corresponding line is defined by the adjacent corners.

Hence, for each two lines h and h' each defined by the adjacent corners of two borders of two different rectangles, an intersection point S is given by

$$\vec{S} = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} + \lambda \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \tag{36}$$

if there is a solution for

$$\lambda = \frac{p_1 u'_2 - p_2 u'_1 - p'_1 u'_2 + p'_2 u'_1}{u'_1 u_1 - u'_2 u_2} \tag{37}$$

Hence, with λ' given by

$$\lambda' = \frac{s_1 - p'_1}{u'_1} \tag{38}$$

S is an intersection point of the two borders, if

$$0 < \lambda < 1 \quad \text{and} \quad 0 < \lambda' < 1 \tag{39}$$

holds.

Thus, all needed intersection points S_j can be calculated by crosschecking all borders of two rectangles for intersection points. This may into an empty set if the rectangles do not overlap, or into at least two and up to eight points if they do. However, the case the rectangles do not overlap is not further regarded.

In Section 4 a method to determine whether a point is located internal to a rectangle is presented. This method is to be applied to all corners of the two rectangles to achieve the corners E_j that are located inside the other rectangle and therefore form vertices

of the polygon. This procedure may result in no or in up to four points.

Thus, a total number of three up to eight points can be found. These k points are the vertices V_1 to V_k of the overlapping polygon, if they are ordered accordingly as depicted in Figure 8.

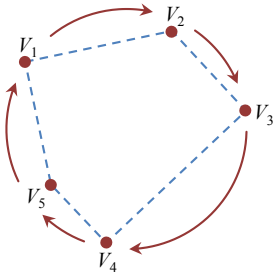

Figure 8: A polygon with vertices ordered in turn.

Since the points S_j and E_j are arbitrarily ordered if they are collected as described, the right ordering can be achieved by ordering all points according to their x-coordinate in the first place.

Afterwards, the points with the smallest and the largest x-coordinate V_{min} and V_{max}, respectively, again define a line. That line separates the set of remaining points into two groups, G_1 and G_2. Thereby, by applying for every point V_i

$$\delta_i = \vec{V}_i \cdot \vec{n} - c \tag{40}$$

with

$$\vec{n} = \frac{\begin{pmatrix} v_{min,2} - v_{max,2} \\ v_{max,1} - v_{min,1} \end{pmatrix}}{\left| \begin{pmatrix} v_{min,2} - v_{max,2} \\ v_{max,1} - v_{min,1} \end{pmatrix} \right|} \tag{41}$$

and

$$c = v_{max,1} \cdot n_1 + v_{max,2} \cdot n_2 \tag{42}$$

each vertex featuring $\delta_i \geq 0$ is located on the one side of the line and, hence, it belongs to G_1. In contrast, each vertex with $\delta_i < 0$ is located on the other side of the line and, therefore, it belongs to G_2.

Finally, by sorting the elements of both groups again according to their x-coordinate, but reversing the order of the elements of G_2, the order of the vertices V_1 to V_k of the overlapping polygon is given by appending the reversed list G_2 to the end of the ordered G_1. This ordering is depicted in Figure 9.

Thus, the overlapping polygon and the order of its vertices is known now. Consequently, all k vertices $V_j = (x_j, y_j)$ with

$$V_{k+1} = V_1 \tag{43}$$

have to be applied to the Gaussian trapezoid formula

$$A = \left| \frac{\sum_j^k (x_j + x_{j+1}) \cdot (y_{j+1} - y_j)}{2} \right| \tag{44}$$

to achieve the size A of the overlapping area.

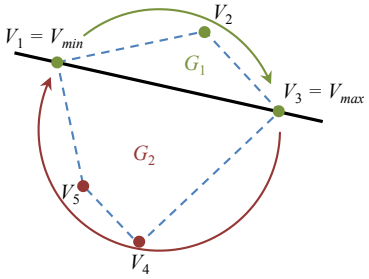

Figure 9: Finally ordered vertices.

6 DISTANCE BETWEEN A POINT AND THE BORDER OF A RECTANGLE

After detecting a situation and its relevance area denoted by a rectangle, a corresponding DENM is sent out and then received by neighboured ITS stations. Hence, the receiving station may display a warning to the driver while approaching the situation. Consequently, the distance from the vehicle's current position to the border of the rectangle has to be calculated.

Thereby, of course, the distance has to be set to 0 if the vehicle is already located inside the rectangle, which may be verified as detailed in Section 4. However, in other cases the distance can be calculated as follows.

Since warnings are presented for upcoming events only, the direction of driving, which is defined the same way as θ, the clockwise angle to north, has to be considered. Let ω be the radian of that direction of driving. Additionally, a maximum warning distance d_{max} is assumed to be given by the use case or, e. g., according to the current vehicle speed. By applying

$$\vec{w} = d_{max} \cdot \begin{pmatrix} sin\omega \\ cos\omega \end{pmatrix} \tag{45}$$

a vector $\vec{w}$ is achieved, pointing into the direction defined by the direction of driving and with a length of d_{max}.

Thus, the current vehicle position P and $\vec{w}$ define a strait line g by

$$g : x(\lambda) = \vec{P} + \lambda \cdot \vec{w} \tag{46}$$

According to Section 5, each border of a rectangle shares a line h. Following the detailed steps in Section 5, the intersection points S_j of all four borders of the rectangle with the line g can be determined.

For each border sharing no intersection point with g under the given constrains for λ, the distance can be assumed to be infinite. However, for all detected

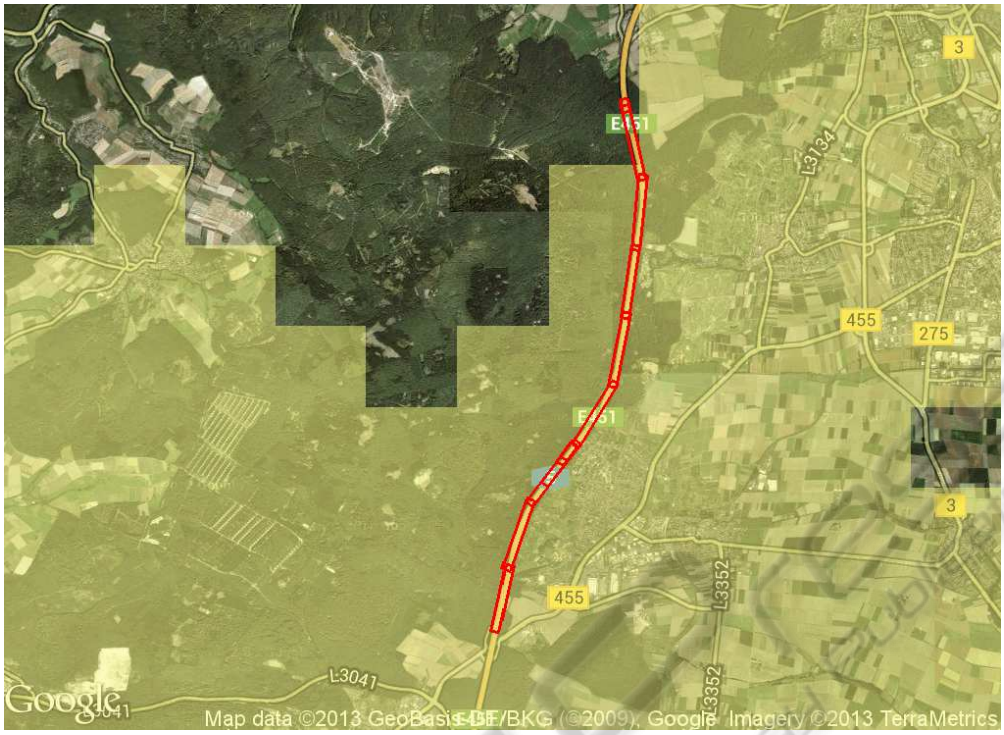

Figure 10: Result of a field operational trial test run exercising the proposed techniques.

intersection points S_j, the distance d_j to the vehicle's position is given by

$$d_j = \left| \vec{P} - \vec{S}_j \right| \tag{47}$$

Hence, the distance from the vehicle to the nearest border of the rectangle is the minimum of all calculated distances d_j.

7 APPLICATION RESULTS

To prove functional correctness and accuracy of the techniques, as presented in this work, under realistic conditions, they are implemented and tested within the Weather Hazard Warning applications in current German, i. e., sim^{TD} (sim^{TD}, 2013), and European, i. e., DriveC2X (DRIVE, 2013), large scale C2X communication field operations tests. In Figure 10 some results of a test drive near Frankfurt am Main, Germany, carried out during the field operational test of sim^{TD} are visualized. While the colored part is a heavy rain situation as reported by a rain radar service, the rectangles of a successive reported rain event are depicted along the highway.

During the test drive, a group of vehicles had driven from north to south on the highway and are, consequently, passing the rain weather condition in that area. Thereby, the first vehicle serves as the detecting vehicle, which initially detects the weather situation for the ITS system. The rectangular relevance areas reported in DENMs by that vehicle are determined according to Section 3. In all following vehicles the successive reported rectangular relevance areas are joined to one single event by calculating the overlapping area as described in Section 5 and, hence, in each following vehicle a warning was displayed by checking the distance to the area according to Section 6. The warning is finally canceled, when the vehicle is entering the nearest situation where a verification shows that the current vehicle position is inside the rectangle according to Section 4.

8 CONCLUSIONS

In this work, we firstly discussed some methods to represent geographical positions with respect to efficient calculations. We then presented techniques to determine a rectangle around a given set of arbitrary points, to verify whether a point is inside or outside an arbitrary oriented rectangle, to calculate the overlapping area of two rectangles, and finally to determine the distance between an arbitrary point and the nearest border of a rectangle. All these tasks have to be solved by a C2X application while handling rectangular rel-

evance areas. Hence, the algorithms described in this work can be used by function developers, researchers, especially with scope on C2X communication, and, maybe, other interested parties. Additionally, the provided ideas, methods, and implementations will serve as a basis for standardization and the upcoming market launch of C2X systems in near future.

REFERENCES

C2C-CC (2007). *C2C-CC Manifesto*. CAR 2 CAR Communication Consortium.

DRIVE (2013). *Project Website*. DRIVE C2X consortium. `http://www.drive-c2x.eu/`.

ETSI (2010a). Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Application; Part 3: Specification of Decentralized Environmental Notification Basic Service. *ETSI TS 102 637-3*. European Telecommunications Standards Institute.

ETSI (2010b). Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 1: Functional Requirements. *ETSI TS 102 637-1*. European Telecommunications Standards Institute.

ETSI (2010c). Intelligent Transport Systems (ITS); Vehicular Communications; Geographical Area Definition *ETSI EN 302 931*. European Telecommunications Standards Institute.

IEEE (2010). Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 6: Wireless Access in Vehicular Environments. *IEEE Std. 802.11p*. Institute of Electrical and Electronics Engineers, Inc.

Jaeger, A. and Huss, S. A. (2011). The weather hazard warning in simtd: A design for road weather related warnings in a large scale car-to-x field operational test. *11th IEEE International Conference on Telecommunications for Intelligent Transport Systems (ITST-2011)*. St. Petersburg, Russia.

sim^{TD} (2013). *Project Website*. Safe and Intelligent Mobility – Test Field Germany. http://simtd.de/index.dhtml///enEN/.

Stübing, H. and Jaeger, A. (2010). Secure beam-forming for weather hazard warning application in car-to-x communication. *Design Methodolo-gies for Secure Embedded Systems*, Lecture Notes in Electrical Engineering 78:187–206.