Automatic Triangulation Positioning System for wide area coverage from a Network of Stations in Fixed Positions

Marios Sfendourakis   Rajagopal Nilavalan   Emmanuel Antonidakis

***Abstract*—**This paper examines the triangulation problem approached from the perspective of a fixed stations network. An initial state A of n sensors (SRs) and m transmitters (TRs).might change from its initial state to a new state B and m +1 transmitters The fixed stations /sensors should recognize the new state B and find the new transmitters that entered in the area by finding with triangulation procedures the positions of the new transmitters. Cases of misinterpretation of data and accuracy problems when more than three lines intersect are explored.

It is also shown that when there is a high number of transmitters and a lot of readings the data analysis procedure becomes more complex and software architecture needs adaptation.

***Keywords*—** Fixed Stations Network, Sensors, Intersection lines Triangulation, Polygons triangulation. Pseudo-Triangulation Transmitters.

1 .INTRODUCTION

Triangulation problem is still under examination and a lot of research is still ongoing as triangulation is used in many applications like GPS positioning, Areas of Mobile phone technology, Robotics position finding etc. Yet the ability to detect the existence of a possible intersection between pairs of objects can be important in a variety of problem domains such as geographic information systems, CAD/CAM geometric modelling ,networking and wireless computing.[1]

In many cases we have a number of transmitters and a number of sensors which have to acquire and interpret data. But relative positioning of the transmitters is strongly related with a right triangulation procedure in order to have correct data analysis and extract right information. Intersection detection is complex problem and algorithms are used to speed up the process are still being explored for various applications, [2].

The software which has been designed allows the user to enter an amount of sensors and a set of data for each sensor. Those data are the bearings that each sensor detects a transmitter in the area. In addition an extra parameter of accuracy is entered which is common for all sensors, meaning that this accuracy is fixed with the assumption that it is the average error of detection for the Fixed Sensors Network.

2. Triangulation Problem

The triangulation problem is not new in the computing technology as it is used in many applications. Relative bearings between sensors will intersect at certain points. A Network of SRs and TRs is depicted in Fig 1

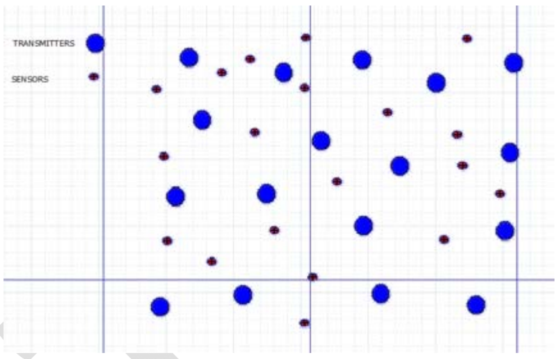


Fig 1

where this status is before the application of the triangulation procedure. By depicting the relative set of bearings for each sensor we see a complex web like the one that appears in Fig 2.

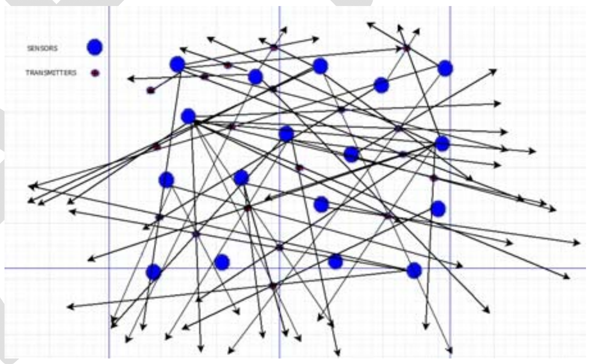


Fig 2

After applying the triangulation procedure we will have many correct triangulations (TNs) and many pseudo triangulations – (PTNs) as at long distances the segments of intersection have areas of uncertainty like in Fig 3.In this depiction SRs areas of bearing appears as sectors with the assumption that there will be a minus –plus amount of accuracy in relative bearings, Fig.4.This assumption is applied in the software which is designed.

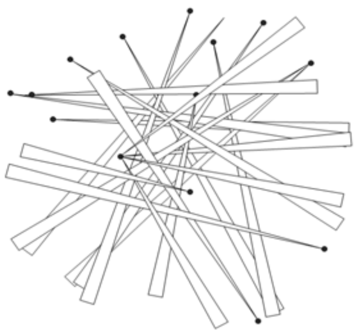


Fig 3

University of Brunel London
TEI of CRETE

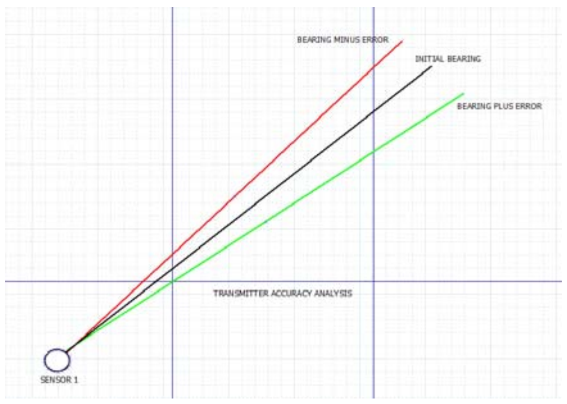


Fig 4

which doesn't allow to exact correct data. In addition minimum errors at long distances can create areas of uncertainty like in Fig 5 where we can't have accurate results of positioning.

2.1 Intersection Area

Intersection area of two SRs is depicted in Fig.4

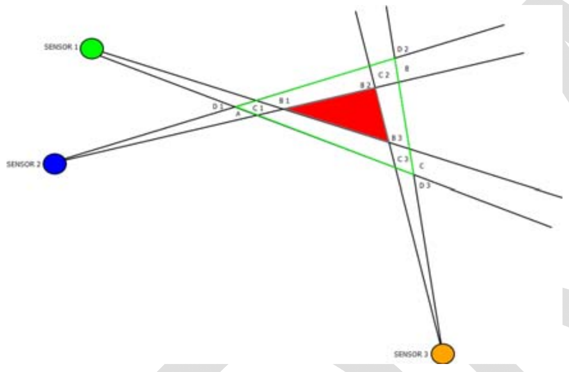


Fig 5

2.2 Triangulation rejection Code

For the rejection of a triangulation we use the following hypothesis which uses Polygons centroids in combination with triangles centroids.

Hypothesis
-C 1 C 2 C3 are the Centroids of the three Polygons A,B,C respectively Fig.5
-Triangle C1 C2 C3 Centroid Point is D
B1 B2 B3 are the closest points of polygons C1 C2 C3 to the point D and they form the triangle B1 B2 B3.

Triangle B1 B2 B3 Centroid is E
Condition 1
POLYGONS A, B, C - ''HAVE NOT COMMON POINTS''
Condition 2
D AND E - ''ARE BOTH INSIDE '' Triangle B1 B2 B3
Condition 3
D AND E ARE NOT COMMON POINTS OF POLYGONS A, B AND C

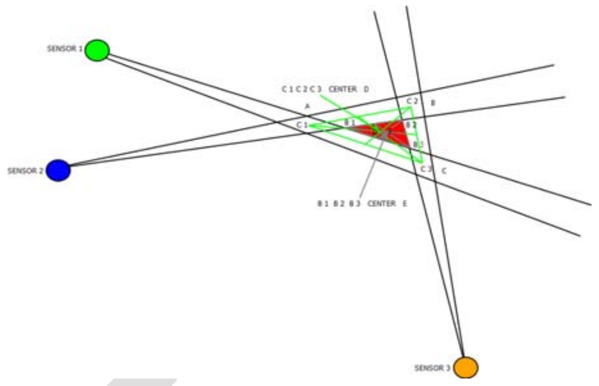


Fig 6

We define the triangle D1 D2 D3 Fig.7 which is formed by the following rule:
-D 1 is the maximum diagonal distance point in Polygon A and point B 1
-D 2 is the maximum diagonal distance point in Polygon B and point B 2
-D 3 is the maximum diagonal distance point in Polygon C and point B 3
We divide the area of the two triangles, Formula 1.

$$\frac{\textbf{Triangle B 1 B 2 B 3}}{\textbf{Triangle D 1 D 2 D 3}} = \boldsymbol{\lambda}$$

Formula 1

-If **0< λ <** 1 and λ close to 1, then the triangle B1 B2 B3 lies within the triangle D1 D2 D3 and we don't have triangulation Fig.5

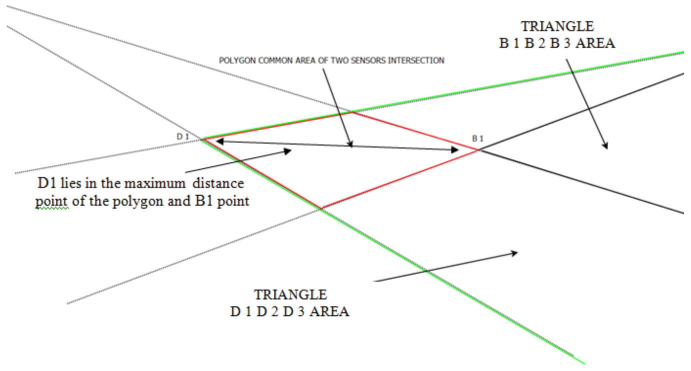

TRIANGLE
B 1 B 2 B 3 AREA
POLYGON COMMON AREA OF TWO SENSORS INTERSECTION
D 1
B 1
D1 lies in the maximum distance point of the polygon and B1 point
TRIANGLE
D 1 D 2 D 3 AREA

Fig 7

-If **λ is close to 0** The triangle B1 B2 B3 lies within the triangle D1 D2 D3 and the Centroid point E lies within the common area of triangulation and we have a triangulation Fig. We also see that the area of triangle B1 B2 B3 is much lower than the area of the triangle D 1 D 2 D 3.
Condition three should also be true or D will be very close to E.

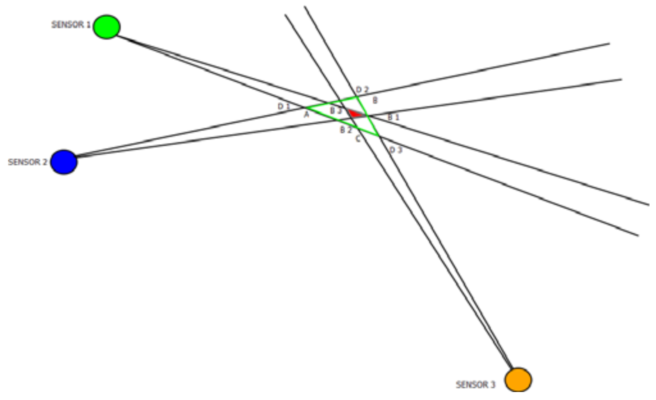


Fig.6

2.3 Real triangulation

In this case all three polygons intersect and they have common area which is shown in following fig.6.This case is considered as a real triangulation.

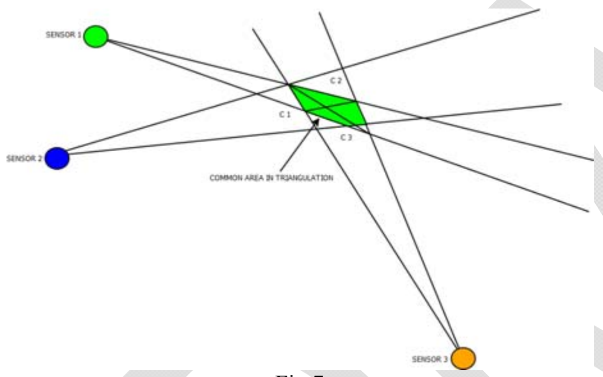


Fig 7

3. Software architecture

The Software starts with the acquirement and storage of the data in a set of arrays. This is achieved by prompting the user to enter the number of sensors and their coordinates and then the data for each transmitter that has been detected in the area. The language used is JAVA and the arrays are two dimensional and dynamic. By that way the user can create a Network of the scale that he wants. At this point for each Sensor the maximum number of allowed bearings detected is defined by the programmer. This parameter can change if more data are required for analysis. Details concerning the software code are beyond the scope of this paper. Software architecture is shown in fig.8

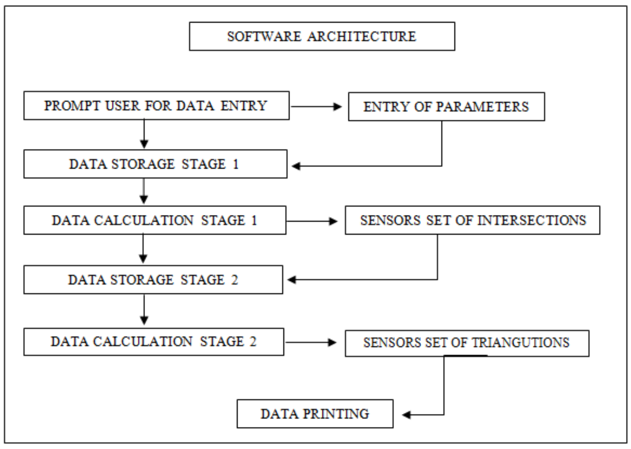


Fig. 8

3.1 Software functions

The Software uses Veness [3] formulas in order to calculate relative bearing and distance between SRs and TRs. Coordinates of intersection points are calculated as pairs between SRs and then the software search for triangulations were three, or more than three lines converge from different SRs.

3.3 JavaScript Code for calculations

Veness [3] provides code for implementation and calculations between geographical points. During this research the following code has been used:

- Intersection of two paths given start points and bearings
- Bearing between two points when their coordinates are known.

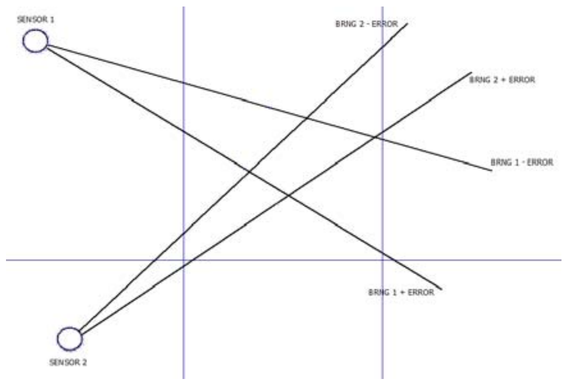
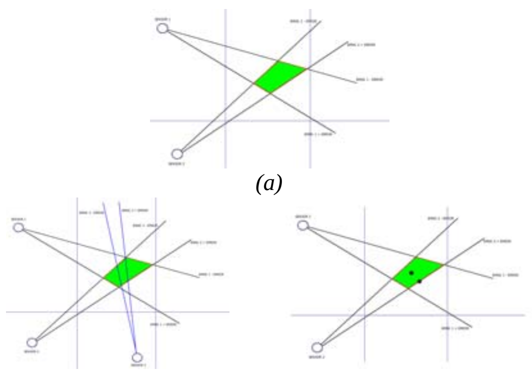


Fig. 9


(a)

(b) *(c)*

Fig. 10

3.3 Intersection between Sensors Routine

The routine which used is the following:

```
    if (CheckBrng1 == $\theta13$ - x  || CheckBrng1 == $\theta13$ + x
        ||
        ($\theta13$ - x < CheckBrng1) && (CheckBrng1 < $\theta13$ + x)

    &&
        (CheckBrng2 == $\theta23$ - x  || CheckBrng2 == $\theta23$ + x ||
        ($\theta23$ - x < CheckBrng2) && CheckBrng2 < $\theta23$ + x ) )
```

where CheckBrng1 and CheckBrng2 are the bearings θ13 , θ23 from Sensor 1 Coordinates ,Point 1 and Sensor 2 Coordinates Point 2 related with the intersection point Point 3 as it is depicted in Fig 9, Fig 10 (a),(b),(c). Point 3 is defined as an intersection between a bearing of SR1 or a bearing from SR2 and a bearing from another SR's set. That point is on the boundaries defined by the green colored polygon Fig 10.That polygon is the common area of intersection between the two SRs.

4. Pseudo-triangulation cases

Triangulations are shown with a circle with a cross inside. The colorless ones are considered pseudo-triangulations.

4.1 Pseudo-triangulation case 1

A set of sensors are used for each triangulation. A sensor located on the direction between another sensor in the set and the intersection point must be excluded from the set. The following figure shows that SR4 should be excluded from the triangulation of TR2 since SR2 is included. Also SR 4 should be excluded from the triangulation of TR1 since SR7 is included, see Fig 11.

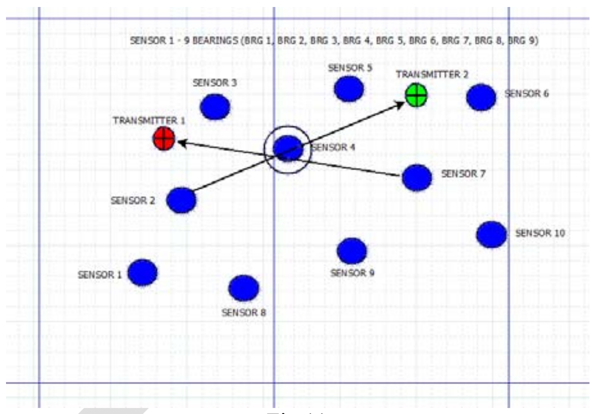


Fig.11

4.2 Pseudo-triangulation case 2

There is false transmitter detection due to the extending of intersection lines. Not all crossings of 3 lines are considered triangulation points, Fig 12

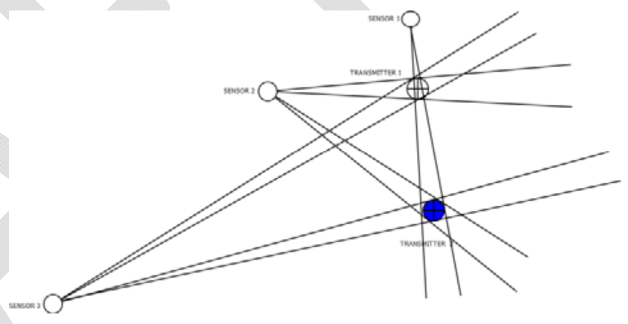


Fig.12

4.3 Pseudo-triangulation case 3

When two TRs fall into the intersection polygon of two SRs then it cannot be differentiated that there are two TRs. Only when there are different sensors near the two TRs then they may be able to be differentiated, Fig 13 (a), (b).

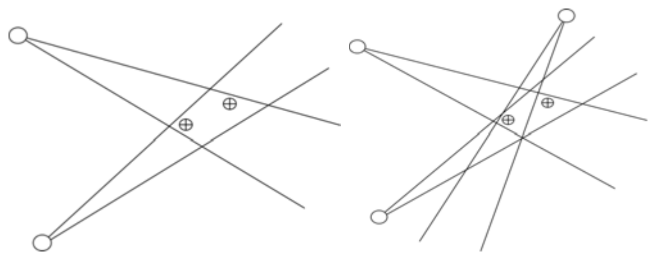

(a) *(b)*

Fig. 13

4.4 Pseudo-triangulation case 4

There are more complicated cases where a mixture of the above and more may happen as the scale of the NFSs increases Fig 14.

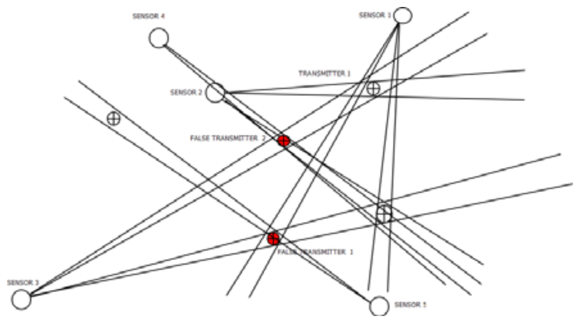


Fig. 14

4.4 Pseudo-triangulation case 5

In the following Fig 15 SR 1 and SR 2 detect on TR at the same bearing. Other TRs should reveal that at that bearing there are more than on TRs on the same line.

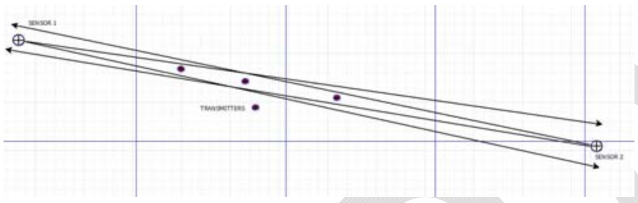


Fig. 15

4.5 Ways to tackle the Pseudo-triangulation cases

As it was obvious in all the pre-mentioned cases of PT the software should tackle with them with a combination of ways in order to minimize false triangulations and PTRs. Ways to tackle these cases are under testing and already are showing positive results.

4.5.1 Partioning of the search area

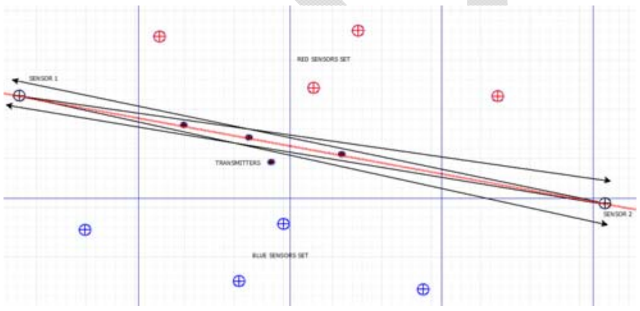


Fig. 16

4.5.2 Rejection of Sector to Sector bearings

By adequate software programming in order to avoid sector to sector bearings the results are more promising while false triangulations of SR to SR are rejected Fig 17. In other words the SRs search for bearings which are out from other SRs region.

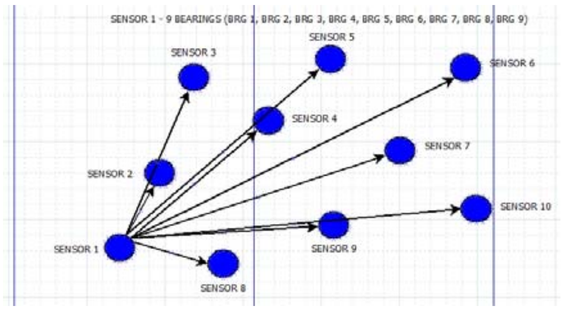

SENSOR 1 - 9 BEARINGS (BRG 1, BRG 2, BRG 3, BRG 4, BRG 5, BRG 6, BRG 7, BRG 8, BRG 9)
SENSOR 5
SENSOR 3
SENSOR 6
SENSOR 4
SENSOR 7
SENSOR 2
SENSOR 10
SENSOR 1
SENSOR 9
SENSOR 8

Fig. 17

Conclusion

As a conclusion we can mention that a triangulation problem is becoming high complicated when we have a large number of sensors and transmitters. The software of the FSN which will have to deal with the triangulation problem should be designed to deal with cases of inaccuracies and false detections. It needs to be tested and configured in order to avoid triangulation cases that are problematic and lead the user to misinterpretation of readings. This research shows that there are relative sensors bearings and data readings that have to be checked in a case by case basis.

Future work might involve the inclusion of more parameters of the sensors like power and antenna size which will lead to new sets of data and additional software architecture in order to acquire correct interpretation of acquired data and avoid cases of false triangulations.

Acknowledgment

M.S. thanks his Prof Nila Nilavalan and Prof. Emmanuel Antonidakis for their support and their guidance in this research. Their contribution added valuable knowledge to my research and

References

[1] Chaman L Sabharwal1, Jennifer L Leopold2, Douglas McGeehan3 Triangle-Triangle Intersection Determination and Classification to Support Qualitative Spatial Reasoning Missouri University of Science and Technology Polibits, Research Journal on Computer Science and Computer Engineering with Applications Issue 48 (July–December 2013), pp. 13–22, 2013

[2] N. Eloe, J. Leopold, C. Sabharwal, and Z. Yin,"*Efficient Computation of Boundary Intersection and Error Tolerance in VRCC-3D+* ", Proceedings of the 18h International Conference on Distributed Multimedia Systems (DMS'12), Miami, FL, Aug. 9- 11, 2012, pp. 67 – 70, 2012.

[3] Veness, Chris.2007a. Calculate distance, bearing and more between two latitude/longitude points. Available at http://www.movable-type. co.uk/scripts/latlong.html