A New Three Object Triangulation Algorithm for Mobile Robot Positioning

Vincent Pierlot, *Member, IEEE*, and Marc Van Droogenbroeck, *Member, IEEE*

***Abstract*—Positioning is a fundamental issue in mobile robot applications. It can be achieved in many ways. Among them, triangulation based on angles measured with the help of beacons is a proven technique. Most of the many triangulation algorithms proposed so far have major limitations. For example, some of them need a particular beacon ordering, have blind spots, or only work within the triangle defined by the three beacons. More reliable methods exist; however, they have an increasing complexity, or they require to handle certain spatial arrangements separately. In this paper, we present a simple and new three object triangulation algorithm, known as ToTal, that natively works in the whole plane and for any beacon ordering. We also provide a comprehensive comparison between many algorithms and show that our algorithm is faster and simpler than comparable algorithms. In addition to its inherent efficiency, our algorithm provides a very useful and unique reliability measure that is assessable anywhere in the plane, which can be used to identify pathological cases, or as a validation gate in Kalman filters.**

***Index Terms*—Mobile robot, positioning, triangulation.**

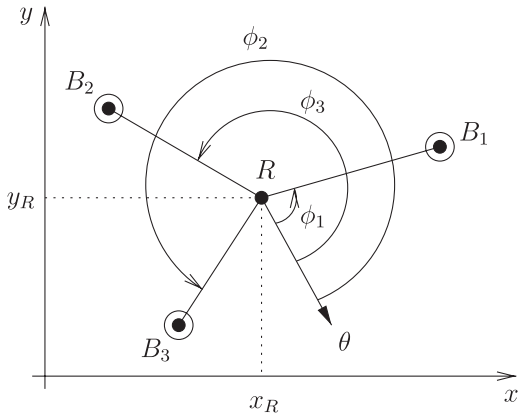


Fig. 1. Triangulation setup in the 2-D plane. R denotes the robot. B_1, B_2, and B_3 are the beacons. ϕ_1, ϕ_2, and ϕ_3 are the angles for B_1, B_2, and B_3, respectively, relative to the robot reference orientation θ. These angles may be used by a triangulation algorithm in order to compute the robot position $\{x_R, y_R\}$ and orientation θ.

I. Introduction

POSITIONING is a fundamental issue in mobile robot applications. Indeed, in most cases, a mobile robot that moves in its environment has to position itself before it can execute its actions properly. Therefore, the robot has to be equipped with some hardware and software capable to provide a sensory feedback related to its environment [3]. Positioning methods can be classified into two main groups [5]: (1) *relative* positioning (also called *dead-reckoning*) and (2) *absolute* positioning (or reference-based). One of the most famous relative positioning technique is the odometry, which consists of counting the number of wheel revolutions to compute the offset relative to a known position. It is very accurate for small offsets but is not sufficient because of the unbounded accumulation of errors over time (because of wheel slippage, imprecision in the wheel circumference, or wheel base) [5]. Furthermore, odometry needs an initial position and fails when the robot is "waken-up" (after a forced reset for example) or is raised and dropped somewhere, since the reference position is unknown or modified. An absolute positioning system is thus required to recalibrate the robot position periodically.

Relative and absolute positioning are complementary to each other [3], [6] and are typically merged together by using a Kalman filter [17], [21]. In many cases, absolute positioning is ensured by beacon-based triangulation or trilateration. *Triangulation* is the process of determining the robot pose (position and orientation) based on angle measurements, while *trilateration* methods involve the determination of the robot position based on distance measurements. Because of the availability of angle measurement systems, triangulation has emerged as a widely used, robust, accurate, and flexible technique [14]. Another advantage of triangulation versus trilateration is that the robot can compute its orientation (or heading) in addition to its position so that the complete *pose* of the robot can be found. The process to determine the robot pose based on angle measurements is generally termed triangulation. The word triangulation is a wide concept, which does not specify if the angles are measured from the robot or the beacons, nor the number of angles used. In this paper, we are interested in self position determination, meaning that the angles are measured from the robot location. Fig. 1 illustrates our triangulation setup.

Moreover, if only three beacons are used in self position determination, triangulation is also termed *Three Object Triangulation* by Cohen and Koss [10]. Here, the general term *object* refers to a 2-D point, whose location is known.

Our algorithm, which will be known as ToTal[1] hereafter, has already been presented in a previous paper [33]. In this paper, we

Manuscript received April 5, 2013; revised September 27, 2013; accepted December 2, 2013. Date of publication January 2, 2014; date of current version June 3, 2014. This paper was recommended for publication by Associate Editor P. Jensfelt and Editor D. Fox upon evaluation of the reviewers' comments.

The authors are with the INTELSIG, Laboratory for Signal and Image Exploitation, Montefiore Institute, University of Liège, Liège, Belgium (e-mail: vpierlot@gmail.com; M.VanDroogenbroeck@ulg.ac.be).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2013.2294061

[1]ToTal stands for: **T**hree **o**bject **T**riangulation **al**gorithm.

supplement our previous work with an extensive review about the triangulation topics, detail the implementation of our algorithm, and compare our algorithm with seventeen other similar algorithms. Please note that the C source code implementation, developed for the error analysis and benchmarks, is made available to the scientific community.

The paper is organized as follows. Section II reviews some of the numerous triangulation algorithms found in the literature. Our new three object triangulation algorithm is described in Section III. Section IV presents simulation results and benchmarks. Then, we conclude the paper in Section V.

II. Related Work

A. Triangulation Algorithms

The principle of triangulation has existed for a long time, and many methods have been proposed so far. One of the first comprehensive reviewing work has been carried out by Cohen and Koss [10]. In their paper, they classify the triangulation algorithms into four groups: (1) *Geometric Triangulation*, (2) *Geometric Circle Intersection*, (3) *Iterative methods* (Iterative Search, Newton–Raphson, etc.), and (4) *Multiple Beacons Triangulation*.

The first group could be named *Trigonometric Triangulation,* because it makes an intensive use of trigonometric functions. Algorithms of the second group determine the parameters (radius and center) of two (of the three) circles that pass through the beacons and the robot, then they compute the intersection between these two circles. Methods of the first and second groups are typically used to solve the three object triangulation problem. The third group linearizes the trigonometric relations to converge to the robot position after some iterations, from a starting point (usually the last known robot position). In the iterative methods, they also present *Iterative Search*, which consists in searching the robot position through the possible space of orientations, and by using a closeness measure of a solution. The fourth group addresses the more general problem of finding the robot pose from more than three angle measurements (usually corrupted by errors), which is an overdetermined problem.

Several authors have noticed that the second group (*Geometric Circle Intersection*) is the most popular to solve the three object triangulation problem [18], [31]. The oldest Geometric Circle Intersection algorithm was described by McGillem and Rappaport [30], [31]. Font-Llagunes and Battle [18] present a very similar method, however they first change the reference frame to relocate beacon 2 at the origin and beacon 3 on the X-axis. They compute the robot position in this reference frame and then, they apply a rotation and translation to return to the original reference frame. Zalama *et al.* [45], [46] present a hardware system to measure angles to beacons and a method to compute the robot pose from three angle measurements. A similar hardware system and method based on [45] and [46] is described by Tsukiyama [44]. Kortenkamp [23] presents a method which turns out to be exactly the same as the one described by Cohen and Koss [10]. All these methods compute the intersection of two of the three circles that pass through the beacons and the robot. It appears that they are all variations or improvements of older methods of McGillem and Rappaport, or Cohen and Koss. The last one is described by Lukic *et al.* [8], [27], but it is not general, as it only works for a subset of all possible beacon locations.

Some newer variations of Geometric/Trigonometric triangulation algorithms are also described in the literature. In [14], Esteves *et al.* extend the algorithm presented earlier by Cohen and Koss [10] to work for any beacon ordering and to work outside the triangle formed by the three beacons. In [15] and [16], they describe the improved version of their algorithm to handle the remaining special cases (when the robot lies on the line that joins the two beacons). They also analyze the position and orientation error sensitivity in [16]. Whereas Easton and Cameron [12] concentrate on an error model for the three object triangulation problem, they also briefly present an algorithm that belongs to this family. Their simple method works in the whole plane and for any beacon ordering. The work of Hmam [20] is based on Esteves *et al.*, as well as Cohen and Koss. He presents a method, valid for any beacon ordering, that divides the whole plane into seven regions and handles two specific configurations of the robot relatively to the beacons. In [28], Madsen and Andersen describe a vision-based positioning system. Such an algorithm belongs to the trigonometric triangulation family as the vision system is used to measure angles between beacons.

It should be noted that the "three object triangulation problem" is also known as the "three point resection problem" in the surveying engineering research area. In this field, the beacons are often called stations, and the angles (or azimuths) are measured with a goniometer. As it is a basic operation for decades in the surveying field, there are lots of procedures (more than 500) to solve this problem, numerically as well as graphically [19]. Surprisingly, there is almost no link between these two fields, except the recent work of Font-Llagunes and Batlle [19], and the older one of McGillem and Rappaport [30], [31]. Therefore, it appears that some algorithms from the two fields are similar, but wear different names. One of the most famous and oldest procedures is called the Kaestner–Burkhardt method, which is also known as the Pothonot–Snellius method [9]. This method is similar to the one described by McGillem and Rappaport [30], [31], which is a trigonometric approach. Then, there is the Collins method [9], which is a trigonometric solution and is the one described by Esteves *et al.* [16] or Cohen and Koss [10]. In addition, there is the Cassini method [9], similar to the method of Easton and Cameron [12], both being a trigonometric approach. Finally, there is the Tienstra method [9], [34], which is a completely different approach, since it makes use of the barycentric coordinates in order to express the robot position as a weighted sum of the beacons' coordinates. This method has been known for a long time; Porta and Thomas have presented a new concise proof for this method recently [34]. Despite that the three point resection problem has been known for a long time and has many solutions, some newer works are still emerging. For example, Font-Llagunes and Batlle [19] have published a new method, which uses straight lines intersection and the property of similar triangles. To our knowledge, the most recent work has been carried on by Ligas [25]. Both Ligas's method and ToTal rely on the idea of using the radical axis of two circles.

However, Ligas intersects one radical axis and one circle, whereas our algorithm intersects the three radical axes of the three pairs of circles.[2] Likewise, Ligas also uses only two trigonometric functions (like our method ToTal), and as a consequence, it is one of the most efficient methods (with ToTal), as shown in Section IV-B.

Some of the Multiple Beacons Triangulation (multiangulation) algorithms are described hereafter. One of the first work in this field was presented by Avis and Imai [1]. In their method, the robot measures k angles from a subset of n indistinguishable landmarks, and therefore they produce a bounded set a valid placements of the robot. Their algorithm runs in $\mathcal{O}(kn^2)$ if the robot has a compass or in $\mathcal{O}(kn^3)$ otherwise. The most famous algorithm was introduced by Betke and Gurvits [3]. They use an efficient representation of landmark 2-D locations by complex numbers to compute the robot pose. The landmarks are supposed to have an identifier known by the algorithm. The authors show that the complexity of their algorithm is proportional to the number of beacons. They also performed experiments with noisy angle measurements to validate their algorithm. Finally, they explain how the algorithm deals with outliers. Another interesting approach is proposed by Shimshoni [36]. He presents an efficient SVD based multiangulation algorithm from noisy angle measurements and explains why transformations have to be applied to the linear system in order to improve the accuracy of the solution. The solution is very close to the optimal solution computed with nonlinear optimization techniques, while being more than a hundred times faster. Briechle and Hanebeck [7] present a new localization approach in the case of relative bearing measurements by reformulating the given localization problem as a nonlinear filtering problem.

Siadat and Vialle [38] describe a multiangulation method based on the Newton–Raphson iterative method to converge to a solution minimizing an evaluation criterion. Lee *et al.* [24] present another iterative method very similar to Newton–Raphson. Their algorithm was first designed to work with three beacons, however it can also be generalized to a higher number of beacons. The initial point of the convergence process is set to the center of the beacons, and good results are obtained after only four steps.

Sobreira *et al.* [41] present a hybrid triangulation method working with two beacons only. They use a concept similar to the *running-fix* method introduced by Bais in [2], in which the robot has to move by a known distance to create a virtual beacon measurement and to compute the robot pose after it has stopped to take another angle measurement. In [40], Sobreira *et al.* perform an error analysis of their positioning system. In particular, they express the position uncertainty originated by errors on measured angles in terms of a surface. Sanchiz *et al.* [35] describe another multiangulation method based on *Iterative Search* and circular correlation. They first compute the robot orientation by maximizing the circular correlation between the expected beacons angles and the measured beacons angles. Then, a method similar to *Iterative Search* is applied to compute the position. Hu and Gu [21] present a multiangulation method based on Kohonen neural network to compute the robot pose and to initialize an extended Kalman filter used for navigation.

[2]Note that the paper of Ligas [25] is posterior to ours [33].

B. Brief Discussion

It is difficult to compare all the above mentioned algorithms, because they operate in different conditions and have distinct behaviors. In practice, the choice is dictated by the application requirements and some compromises. For example, if the setup contains three beacons only or if the robot has limited on-board processing capabilities, methods of the first and second groups are the best candidates. Methods of the third and fourth groups are appropriate if the application must handle multiple beacons and if it can accommodate a higher computational cost. The main drawback of the third group is the convergence issue (existence or uniqueness of the solution) [10]. The main drawback of the fourth group is the computational cost [3], [7], [36].

The drawbacks of the first and second group are usually a lack of precision related to the following elements: 1) The beacon ordering needed to get the correct solution, 2) the consistency of the methods when the robot is located outside the triangle defined by the three beacons, 3) the strategy to follow when falling into some particular geometrical cases (that induce mathematical underdeterminations when computing trigonometric functions with arguments like 0 or π, division by 0, etc.), and 4) the reliability measure of the computed position. Simple methods of the first and second groups usually fail to propose a proper answer to all of these concerns. For example, to work in the entire plane and for any beacon ordering (for instance [16]), they have to consider a set of special geometrical cases separately, that results in a lack of clarity. Finally, to our knowledge, none of these algorithms gives a realistic reliability measure of the computed position.

C. Other Aspects of Triangulation

For now, we have focused on the description of triangulation algorithms which are used to compute the position and orientation of the robot. Other aspects of triangulation have to be considered as well to achieve an optimal result on the robot pose in a practical situation. These are: 1) the sensitivity analysis of the triangulation algorithm, 2) the optimal placement of the landmarks, 3) the selection of some landmarks among the available ones to compute the robot pose, and 4) the knowledge of the true landmark locations in the world and the true location of the angular sensor on the robot.

Kelly [22] uses the famous geometric dilution of precision concept, used in GPS error analysis, and applies it to range based and angle based positioning techniques. He derives two useful formulas in the case of two beacons. Madsen *et al.* [29] perform a sensitivity analysis of their triangulation algorithm. They use first order propagation of angle measurement errors through covariance matrix and Jacobian to derive the precision of location. Easton and Cameron [12] present the same error analysis as that of Madsen *et al.*, but in addition to the angle uncertainty, they take into account the beacon location uncertainty. They also present some simulations for various beacons configurations as well as some metrics to evaluate their model's performance.

Optimal landmark placement has been extensively studied. Sinriech and Shoval [37], [39] define a nonlinear optimization model used to determine the position of the minimum number of beacons required by a shop floor to guarantee an accurate and reliable performance for automated guided vehicles. Demaine *et al.* [11] present a polynomial time algorithm to place reflector landmarks such that the robot can always localize itself from any position in the environment, which is represented by a polygonal map. Tedkas and Isler [42], [43] address the problem of computing the minimum number and placement of sensors so that the localization uncertainty at every point in the workspace is less than a given threshold. They use the uncertainty model for angle based positioning derived by Kelly [22].

Optimal landmark selection has been studied by Madsen *et al.* [28], [29]. They propose an algorithm to select the best triplet among several landmarks seen by a camera, that yields to the best position estimate. The algorithm is based on a "goodness" measure derived from an error analysis which depends on landmarks and on the robot relative pose.

To have a good sensor that provides precise angle measurements as well as a good triangulation algorithm is not the only concern to get accurate positioning results. Indeed, the angle sensor could be subject to nonlinearities in the measuring angle range (a complete revolution). Moreover, the beacon locations that are generally measured manually are subject to inaccuracies, which affect directly the positioning algorithm. In their paper, Loevsky and Shimshoni [26] propose a method to calibrate the sensor and a method to correct the measured beacon locations. They show that their procedure is effective and mandatory to achieve good positioning performance.

In the remainder of this paper, we concentrate on three object triangulation methods. Our paper presents a new three object triangulation algorithm that works in the entire plane (except when the beacons and the robot are concyclic or collinear), and for any beacon ordering. Moreover, it minimizes the number of trigonometric computations and provides a unique quantitative reliability measure of the computed position.

III. Description of a New Three Object Triangulation Algorithm

Our motivation for a new triangulation algorithm is fourfold: 1) We want it to be independent of the beacon ordering, 2) the algorithm must also be independent of the relative positions of the robot and the beacons, 3) the algorithm must be fast and simple to implement in a dedicated processor, and 4) the algorithm has to provide a criterion to qualify the reliability of the computed position.

Our algorithm, named, ToTal, belongs to the family of *Geometric Circle Intersection* algorithms (that is, the second group). It first computes the parameters of the three circles that pass through the robot and the three pairs of beacons. Then, it computes the intersection of these three circles, by using all the three circles parameters (not only two of them, to the contrary of other methods).

Our algorithm relies on two assumptions: 1) The beacons are distinguishable (a measured angle can be associated to a given beacon); and (2) the angle measurements from the beacons are taken separately, and relatively to some reference angle θ, usually the robot heading (see Fig. 1). Note that the second hypothesis simply states that angles are given by a rotating angular sensor. Such sensors are common in mobile robot positioning that uses triangulation [4], [5], [24], [31], [32], [45]. By convention, in the following, we consider that angles are measured counterclockwise (CCW), like angles on the trigonometric circle. To invert the rotating direction to clockwise (CW) would only require minimal changes of our algorithm.

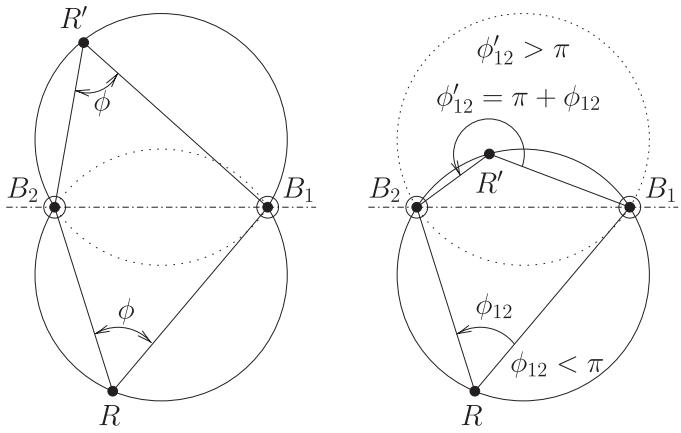


Fig. 2. (Left) Locus of points R that *see* two fixed points B_1 and B_2 with a constant angle γ, in the 2-D plane, is formed by two arcs of circle. (Right) Ambiguity is removed by taking the following convention: $\phi_{12} = \phi_2 - \phi_1$.

A. First Part of the Algorithm: The Circle Parameters

In a first step, we want to calculate the locus of the robot positions R, that *see* two fixed beacons, B_1 and B_2, with a constant angle γ, in the 2-D plane. It is a well-known result that this locus is an arc of the circle that passes through B_1 and B_2, whose radius depends on the distance between B_1 and B_2, and γ (Proposition 21 of Book III of Euclid's Elements [48]). More precisely, this locus is composed of two arcs of circle, which are the reflection of each other through the line that joins B_1 and B_2 (see the continuous lines on the left-hand side of Fig. 2).

A robot that measures an angle γ between two beacons can stand on either of these two arcs. This case occurs if the beacons are not distinguishable or if the angular sensor is not capable to measure angles larger than π (like a vision system with a limited field of view, as used by Madsen *et al.* [28]). To avoid this ambiguity, we impose that, as shown on the right-hand side of Fig. 2, the measured angle between two beacons B_1 and B_2, which is denoted ϕ_{12}, is always computed as $\phi_{12} = \phi_2 - \phi_1$ (this choice is natural for a CCW rotating sensor). This is consistent with our measurement considerations, and it removes the ambiguity about the locus; however, it requires that beacons are indexed and that the robot is capable to establish the index of any beacon. As a result, the locus is a single circle that passes through R, B_1, and B_2. In addition, the line joining B_1 and B_2 divides the circle into two parts: one for $\phi_{12} < \pi$ and the other for $\phi_{12} > \pi$. In the following, we compute the circle parameters.

The circle equation may be derived by using the complex representation of 2-D points (Argand diagram) and by

expressing angles as the argument of complex numbers. In particular, the angle of $(B_2 - R)$ is equal to that of $(B_1 - R)$ plus ϕ_{12}. Equivalently

$$\arg\left\{\frac{B_2 - R}{B_1 - R}\right\} = \phi_{12} \tag{1}$$

$$\Rightarrow \arg\left\{(B_2 - R)\,\overline{(B_1 - R)}\right\} = \phi_{12}. \tag{2}$$

Then, if we substitute R, B_1, B_2, respectively by $(x + iy)$, $(x_1 + iy_1)$, $(x_2 + iy_2)$, we have that

$$\arg\left\{(x_2 + iy_2 - x - iy)\,(x_1 - iy_1 - x + iy)\,e^{-i\phi_{12}}\right\} = 0 \tag{3}$$

$$\Rightarrow -\sin\phi_{12}\,(x_2 - x)\,(x_1 - x) + \sin\phi_{12}\,(y_2 - y)\,(y - y_1) + \cos\phi_{12}\,(x_2 - x)\,(y - y_1) + \cos\phi_{12}\,(y_2 - y)\,(x_1 - x) = 0 \tag{4}$$

where $i = \sqrt{-1}$. After lengthy simplifications, we find the locus

$$(x - x_{12})^2 + (y - y_{12})^2 = R_{12}^2 \tag{5}$$

which is a circle whose center $\{x_{12},\, y_{12}\}$ is located at

$$x_{12} = \frac{(x_1 + x_2) + \cot\phi_{12}\,(y_1 - y_2)}{2} \tag{6}$$

$$y_{12} = \frac{(y_1 + y_2) - \cot\phi_{12}\,(x_1 - x_2)}{2} \tag{7}$$

and whose squared radius equals

$$R_{12}^2 = \frac{(x_1 - x_2)^2 + (y_1 - y_2)^2}{4\,\sin^2\phi_{12}}. \tag{8}$$

The three last equations may also be found in [18]. The replacement of ϕ_{12} by $\pi + \phi_{12}$ in the above equations yields the same circle parameters (see the right-hand side of Fig. 2), which is consistent with our measurement considerations. For an angular sensor that turns in the CW direction, these equations are identical except that one has to change the sign of $\cot(.)$ in (6) and (7).

Hereafter, we use the following notations:

1) B_i is the beacon i, whose coordinates are $\{x_i,\, y_i\}$.
2) R is the robot position, whose coordinates are $\{x_R,\, y_R\}$.
3) ϕ_i is the angle for beacon B_i.
4) $\phi_{ij} = \phi_j - \phi_i$ is the bearing angle between beacons B_i and B_j.
5) $T_{ij} = \cot(\phi_{ij})$.
6) $\mathcal{C}_{ij}$ is the circle that passes through B_i, B_j, and R.
7) c_{ij} is the center of $\mathcal{C}_{ij}$, whose coordinates are $\{x_{ij},\, y_{ij}\}$:

$$x_{ij} = \frac{(x_i + x_j) + T_{ij}\,(y_i - y_j)}{2} \tag{9}$$

$$y_{ij} = \frac{(y_i + y_j) - T_{ij}\,(x_i - x_j)}{2} \tag{10}$$

8) R_{ij} is the radius of $\mathcal{C}_{ij}$ derived from

$$R_{ij}^2 = \frac{(x_i - x_j)^2 + (y_i - y_j)^2}{4\,\sin^2\phi_{ij}}. \tag{11}$$

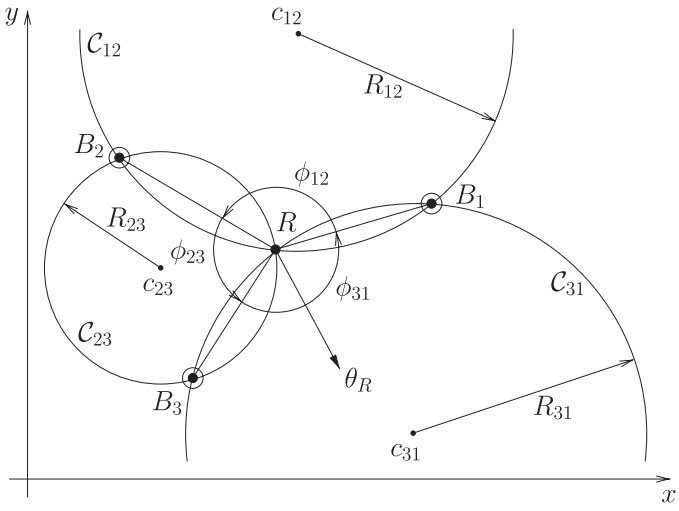


Fig. 3. Triangulation setup in the 2-D plane, using the geometric circle intersection. R is the robot. B_1, B_2, and B_3 are the beacons. ϕ_{ij} are the angles between B_i, R, and B_j. $\mathcal{C}_{ij}$ are the circles that pass through B_i, R, and B_j. R_{ij} and c_{ij} are, respectively, the radii and center coordinates of $\mathcal{C}_{ij}$. θ_R is the robot heading orientation.

All the previous quantities are valid for $i \neq j$; otherwise, the circle does not exist. In addition, we have to consider the case $\phi_{ij} = k\pi$, $k \in \mathbb{Z}$. In that case, the $\sin(.)$ and $\cot(.)$ are equal to zero, and the circle degenerates as the B_iB_j line (infinite radius and center coordinates). In a practical situation, this means that the robot stands on the B_iB_j line and measures an angle $\phi_{ij} = \pi$ when between the two beacons or $\phi_{ij} = 0$ when outside of the B_iB_j segment. These special cases are discussed later.

B. Second Part of the Algorithm: Circles Intersection

From the previous section, each bearing angle ϕ_{ij} between beacons B_i and B_j constraints the robot to be on a circle $\mathcal{C}_{ij}$, which passes through B_i, B_j, and R (see Fig. 3). The parameters of the circles are given by (9)–(11). Common methods use two of the three circles to compute the intersections (when they exist), one of which is the robot position and the second being the common beacon of the two circles. This requires to solve a quadratic system and to choose the correct solution for the robot position [18]. Moreover, the choice of the two circles is arbitrary and usually fixed, whereas this choice should depend on the measured angles or beacons and robot relative configuration to have a better numerical behavior.

Hereafter, we propose a novel method to compute this intersection by using all of the three circles and by reducing the problem to a linear problem.[3] To understand this elegant method, we first introduce the notion of *power center* (or *radical center*) of three circles. The *power center* of three circles is the unique point of equal *power* with respect to these circles [13]. The *power* of a point p relative to a circle $\mathcal{C}$ is defined as

$$\mathcal{P}_{\mathcal{C},p} = (x - x_c)^2 + (y - y_c)^2 - R^2 \tag{12}$$

[3]The idea of using all the parameters from the three circles is not new and has been used in [47]. However, they did not simplify their algorithm as far as we do in our developments.

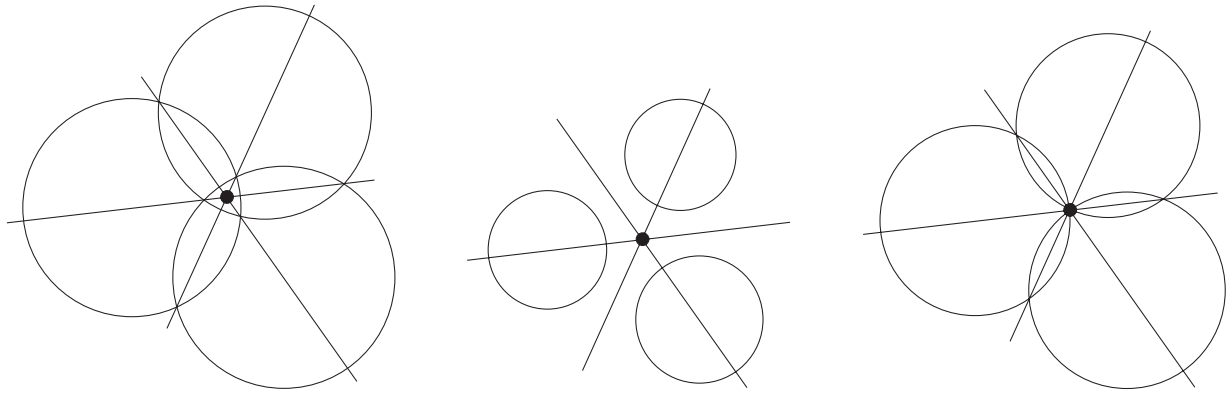


Fig. 4. Black point is the power center of three circles for various configurations. It is the unique point having the same power with respect to the three circles. The power center is the intersection of the three power lines.

where $\{x, y\}$ are the coordinates of p, $\{x_c, y_c\}$ are the circle center coordinates, and R is the circle radius. The power of a point is null onto the circle, negative inside the circle, and positive outside the circle. It defines a sort of relative distance of a point from a given circle. The *power line* (or *radical axis*) of two circles is the locus of points that have the same power with respect to both circles [13]; in other terms, it is also the locus of points at which tangents drawn to both circles have the same length. The power line is perpendicular to the line that joins the circle centers and passes through the circle intersections, when they exist. Monge demonstrated that when considering three circles, the three power lines defined by the three pairs of circles are concurring in the power center [13]. Fig. 4 shows the power center of three circles for various configurations. The power center is always defined, except when at least two of the three circle centers are equal or when the circle centers are collinear (parallel power lines).

The third case of Fig. 4 (right-hand drawing) is particular as it perfectly matches our triangulation problem (see Fig. 3). Indeed, the power center of three concurring circles corresponds to their unique intersection. In our case, we are sure that the circles are concurring since we have

$$\phi_{31} = \phi_3 - \phi_1 = (\phi_3 - \phi_2) + (\phi_2 - \phi_1) \tag{13}$$

$$= -\phi_{23} - \phi_{12} \tag{14}$$

by construction (only two of the three bearing angles are independent), even in presence of noisy angle measurements ϕ_1, ϕ_2, and ϕ_3. It has the advantage that this intersection may be computed by intersecting the power lines, which is a linear problem. The power line of two circles is obtained by equating the power of the points relatively to each circle [given by (12)]. In our problem, the power line of $\mathcal{C}_{12}$ and $\mathcal{C}_{23}$ is given by

$$(x - x_{12})^2 + (y - y_{12})^2 - R_{12}^2 = (x - x_{23})^2 + (y - y_{23})^2 - R_{23}^2 \tag{15}$$

$$\Rightarrow x\,(x_{12} - x_{23}) + y\,(y_{12} - y_{23}) = \frac{x_{12}^2 + y_{12}^2 - R_{12}^2}{2} - \frac{x_{23}^2 + y_{23}^2 - R_{23}^2}{2} \tag{16}$$

$$\Rightarrow x\,(x_{12} - x_{23}) + y\,(y_{12} - y_{23}) = k_{12} - k_{23} \tag{17}$$

where we introduce a new quantity k_{ij} which only depends on $\mathcal{C}_{ij}$ parameters:

$$k_{ij} = \frac{x_{ij}^2 + y_{ij}^2 - R_{ij}^2}{2}. \tag{18}$$

In our triangulation problem, we have to intersect the three power lines, that is to solve this linear system:

$$\begin{cases} x\,(x_{12} - x_{23}) + y\,(y_{12} - y_{23}) & = k_{12} - k_{23} \\ x\,(x_{23} - x_{31}) + y\,(y_{23} - y_{31}) & = k_{23} - k_{31} \\ x\,(x_{31} - x_{12}) + y\,(y_{31} - y_{12}) & = k_{31} - k_{12.} \end{cases} \tag{19}$$

As can be seen, any of these equations may be obtained by adding the two others, which is a way to prove that the three power lines concur in a unique point: the power center. The coordinates of the power center, that is the robot position is given by

$$x_R = \frac{\begin{vmatrix} k_{12} - k_{23} & y_{12} - y_{23} \\ k_{23} - k_{31} & y_{23} - y_{31} \end{vmatrix}}{D_\triangle} \tag{20}$$

$$y_R = \frac{\begin{vmatrix} x_{12} - x_{23} & k_{12} - k_{23} \\ x_{23} - x_{31} & k_{23} - k_{31} \end{vmatrix}}{D_\triangle}. \tag{21}$$

The denominator $D_\triangle$, which is common to x_R and y_R, is equal to

$$D_\triangle = \begin{vmatrix} x_{12} - x_{23} & y_{12} - y_{23} \\ x_{23} - x_{31} & y_{23} - y_{31} \end{vmatrix} = \begin{vmatrix} x_{12} & y_{12} & 1 \\ x_{23} & y_{23} & 1 \\ x_{31} & y_{31} & 1 \end{vmatrix} \tag{22}$$

which is the signed area between the circle centers multiplied by two. This result shows that the power center exists if the circle centers are not collinear, that is, if $D_\triangle \neq 0$. The special case ($D_\triangle = 0$) is discussed later.

C. First (Naive) Version of the Algorithm

A first, but naive, version of our algorithm consists in applying the previous equations to get the robot position. This method is correct; however, it is possible to further simplify the equations. First, note that the squared radii R_{ij}^2 only appear in the parameters k_{ij}. If we replace the expression of R_{ij}^2 [see (11)] in the

expression of k_{ij} [see (18)], we find, after many simplifications, that

$$k_{ij} = \frac{x_i x_j + y_i y_j + T_{ij}(x_j y_i - x_i y_j)}{2} \tag{23}$$

which is much simpler than (11) and (18) (no squared terms anymore). In addition, the $1/2$ factor involved in the circle centers coordinates [see (9) and (10)], as well as in the parameters k_{ij} (18), cancels in the robot position coordinates [see (20) and (21)]. This factor can thus be omitted. For now, we use these modified circle center coordinates $\{x'_{ij},\, y'_{ij}\}$

$$x'_{ij} = (x_i + x_j) + T_{ij}\,(y_i - y_j) \tag{24}$$

$$y'_{ij} = (y_i + y_j) - T_{ij}\,(x_i - x_j) \tag{25}$$

and modified parameters k'_{ij}

$$k'_{ij} = x_i x_j + y_i y_j + T_{ij}(x_j y_i - x_i y_j). \tag{26}$$

D. Final Version of the Algorithm

The most important simplification consists in translating the world coordinate frame into one of the beacons, that is solving the problem relatively to one beacon and then add the beacon coordinates to the computed robot position (like Font-Llagunes [18] without the rotation of the frame). In the following, we arbitrarily choose B_2 as the origin ($B'_2 = \{0,\, 0\}$). The other beacon coordinates become $B'_1 = \{x_1 - x_2,\, y_1 - y_2\} = \{x'_1,\, y'_1\}$ and $B'_3 = \{x_3 - x_2,\, y_3 - y_2\} = \{x'_3,\, y'_3\}$. Since $x'_2 = 0$ and $y'_2 = 0$, we have $k'_{12} = 0$, $k'_{23} = 0$. In addition, we can compute the value of one $\cot(.)$ by referring to the two other $\cot(.)$ because the three angles are linked [see (14)]

$$T_{31} = \frac{1 - T_{12}T_{23}}{T_{12} + T_{23}}. \tag{27}$$

The final algorithm is given in Algorithm 1.

E. Discussion

The ToTal algorithm is very simple: computations are limited to basic arithmetic operations and only two $\cot(.)$. Furthermore, the number of conditional statements is reduced, which increases its readability and eases its implementation. Among them, we have to take care of the $\cot(.)$ infinite values and the division by D, if equal to zero. If a bearing angle ϕ_{ij} between two beacons is equal to 0 or π, that is, if the robot stands on the B_iB_j line, then $\cot(\phi_{ij})$ is infinite. The corresponding circle degenerates to the B_iB_j line (infinite radius and center coordinates). The robot is then located at the intersection of the remaining power line and the B_iB_j line; it can be shown that the mathematical limit $\lim_{T_{ij}\to\pm\infty}\{x_R,\, y_R\}$ exists and corresponds to this situation.

Like for other algorithms, our algorithm also has to deal with these special cases, but the way to handle them is simple. In practice, we have to avoid *Inf* or *NaN* values in the floating point computations. We propose two ways to manage this situation. The first way consists in limiting the $\cot(.)$ value to a minimum or maximum value, corresponding to a small angle that is far below the measurement precision. For instance, we limit the value of the $\cot(.)$ to $\pm 10^8$, which corresponds to an angle of about $\pm 10^{-8}\ rad$; this is indeed far below the existing angular sensor precisions. With this approximation of the mathematical limit, the algorithm remains unchanged. The second way consists in adapting the algorithm when one bearing angle is equal to 0 or π. This special case is detailed in Algorithm 2, in which the indexes $\{i,\, j,\, k\}$ have to be replaced by $\{1,\, 2,\, 3\}$, $\{3,\, 1,\, 2\}$, or $\{2,\, 3,\, 1\}$ if $\phi_{31} = 0$, $\phi_{23} = 0$, or $\phi_{12} = 0$, respectively.

The denominator D is equal to 0 when the circle centers are collinear or coincide. For noncollinear beacons, this situation occurs when the beacons and the robot are concyclic; they all stand on the same circle, which is called the *critic circumference* in [18]. In that case, the three circles are equal as well as their centers, which causes $D = 0$ (the area defined by the three circle centers is equal to zero). For collinear beacons, this is encountered when the beacons and the robot all stand on this line. For these cases, it is impossible to compute the robot position. This is a *restriction common to all three object triangulation*, regardless of the algorithm used [14], [18], [31].

The value of D, computed in the final algorithm, is the signed area delimited by the real circle centers, multiplied by height[4].

Algorithm 1 Final version of the ToTal algorithm.

Given the three beacon coordinates $\{x_1,\, y_1\}$, $\{x_2,\, y_2\}$, $\{x_3,\, y_3\}$, and the three angles ϕ_1, ϕ_2, ϕ_3:

1) compute the modified beacon coordinates:

$$x'_1 = x_1 - x_2, \qquad y'_1 = y_1 - y_2,$$
$$x'_3 = x_3 - x_2, \qquad y'_3 = y_3 - y_2,$$

2) compute the three $\cot(.)$:

$$T_{12} = \cot(\phi_2 - \phi_1), \qquad T_{23} = \cot(\phi_3 - \phi_2),$$
$$T_{31} = \frac{1 - T_{12}T_{23}}{T_{12} + T_{23}},$$

3) compute the modified circle center coordinates:

$$x'_{12} = x'_1 + T_{12}\,y'_1, \qquad y'_{12} = y'_1 - T_{12}\,x'_1,$$
$$x'_{23} = x'_3 - T_{23}\,y'_3, \qquad y'_{23} = y'_3 + T_{23}\,x'_3,$$
$$x'_{31} = (x'_3 + x'_1) + T_{31}\,(y'_3 - y'_1),$$
$$y'_{31} = (y'_3 + y'_1) - T_{31}\,(x'_3 - x'_1),$$

4) compute k'_{31}:

$$k'_{31} = x'_1 x'_3 + y'_1 y'_3 + T_{31}(x'_1 y'_3 - x'_3 y'_1),$$

5) compute D (if $D = 0$, return with an error):

$$D = (x'_{12} - x'_{23})(y'_{23} - y'_{31}) - (y'_{12} - y'_{23})(x'_{23} - x'_{31}),$$

6) compute the robot position $\{x_R,\, y_R\}$ and return:

$$x_R = x_2 + \frac{k'_{31}(y'_{12} - y'_{23})}{D}, \quad y_R = y_2 + \frac{k'_{31}(x'_{23} - x'_{12})}{D}.$$

[4]Note that the quantity D computed in the final algorithm is different from the quantity $D_\triangle$ defined in Section III-B, since the center coordinates have been multiplied by two.

Algorithm 2 Special case $\phi_{ki} = 0 \vee \phi_{ki} = \pi$.

Given the three beacon coordinates $\{x_i, y_i\}$, $\{x_j, y_j\}$, $\{x_k, y_k\}$, and the three angles ϕ_i, ϕ_j, ϕ_k:

1) compute the modified beacon coordinates:

$$x'_i = x_i - x_j, \qquad y'_i = y_i - y_j,$$
$$x'_k = x_k - x_j, \qquad y'_k = y_k - y_j,$$

2) compute $T_{ij} = \cot(\phi_j - \phi_i)$,

3) compute the modified circle center coordinates:

$$x'_{ij} = x'_i + T_{ij}\, y'_i, \qquad y'_{ij} = y'_i - T_{ij}\, x'_i,$$
$$x'_{jk} = x'_k + T_{ij}\, y'_k, \qquad y'_{jk} = y'_k - T_{ij}\, x'_k,$$
$$x'_{ki} = (y'_k - y'_i), \qquad y'_{ki} = (x'_i - x'_k),$$

4) compute $k'_{ki} = (x'_i y'_k - x'_k y'_i)$,

5) compute D (if $D = 0$, return with an error):

$$D = (x'_{jk} - x'_{ij})(y'_{ki}) + (y'_{ij} - y'_{jk})(x'_{ki}),$$

6) compute the robot position $\{x_R, y_R\}$ and return:

$$x_R = x_j + \frac{k'_{ki}(y'_{ij} - y'_{jk})}{D}, \quad y_R = y_j + \frac{k'_{ki}(x'_{jk} - x'_{ij})}{D}.$$

$|D|$ decreases to 0 when the robot approaches the critic circle (almost collinear circle center and almost parallel power lines). Therefore, it is quite natural to use $|D|$ as a reliability measure of the computed position. In the next section, we show that $1/|D|$ is a good approximation of the position error. In practice, this measure can be used as a validation gate after the triangulation algorithm, or when a Kalman filter is used. Finally, it should be noted that the robot orientation θ_R may be determined by using any beacon B_i and its corresponding angle ϕ_i, once the robot position is known:

$$\theta_R = \text{atan2}(y_i - y_R, x_i - x_R) - \phi_i \tag{28}$$

where atan2(y, x) denotes the C-like two argument function, defined as the principal value of the argument of the complex number $(x + iy)$.

IV. Simulations

A. Error Analysis

The problem of triangulation given three angle measurements is an exact calculus of the robot pose, even if these angles are affected by noise. Therefore, the sensitivity of triangulation with respect to the input angles is unique and does not depend on the way the problem is solved, nor on the algorithm. This contrasts with multiangulation, which is an overdetermined problem, even with perfect angle measurements. Therefore, we do not elaborate on the error analysis for triangulation, as has been studied in many papers; the same conclusions, as found in [12], [16], [18], [22], [28], and [37], also yield for our algorithm. However, in order to validate our algorithm and to discuss the main characteristics of triangulation sensitivity, we have performed some simulations. The simulation setup comprises a square shaped area ($4 \times 4\,m^2$), with three beacons which form two distinct configurations. The first one is a regular triangle ($B_1 = \{0\,m,\, 1\,m\}$, $B_2 = \{-0.866\,m,\, -0.5\,m\}$, and $B_3 = \{0.866\,m,\, -0.5\,m\}$), and the second one is a line ($B_1 = \{0\,m,\, 0\,m\}$, $B_2 = \{-0.866\,m,\, 0\,m\}$, and $B_3 = \{0.866\,m,\, 0\,m\}$). The distance step is 2 cm in each direction. For each point in this grid, we compute the exact angles ϕ_i seen by the robot (the robot orientation is arbitrarily set to $0°$). Then, we add Gaussian noise to these angles, with zero mean, and with two different standard deviations ($\sigma = 0.01°$, $\sigma = 0.1°$). The noisy angles are then used as inputs of our algorithm to compute the estimated position. The position error Δd_R is the Euclidean distance between the exact and estimated positions:

$$\Delta d_R = \sqrt{(x_{\text{true}} - x_R)^2 + (y_{\text{true}} - y_R)^2}. \tag{29}$$

The orientation error $\Delta\theta_R$ is the difference between the exact and estimated orientations:

$$\Delta\theta_R = \theta_{\text{true}} - \theta_R. \tag{30}$$

The experiment is repeated 1000 times for each position to compute the standard deviation of the position error $\sqrt{var\{\Delta d_R\}}$ and of the orientation error $\sqrt{var\{\Delta\theta_R\}}$. The standard deviations of the position and orientation errors are drawn in Fig. 5. The beacon locations are represented by black and white dot patterns. The first and second columns provide the result for the first configuration, for $\sigma = 0.01°$, and $\sigma = 0.1°$, respectively. The third and fourth columns provide the result for the second configuration, for $\sigma = 0.01°$, and $\sigma = 0.1°$, respectively. The first, second, and third rows show the standard deviation of the position error, the standard deviation of the orientation error, and the mean error measure $1/|D|$, respectively. Note that the graphic scales are not linear. We have equalized the image histograms in order to enhance their visual representation and to point out the similarities between the position and orientation error and our new error measure.

Our simulation results are consistent with common three object triangulation algorithms. In particular, in the first configuration, we can easily spot the criticcircumference where errors are large, the error being minimum at the center of this circumference. In the second configuration, this critic circumference degenerates as the line passing through the beacons. In addition, one can see that, outside the critic circumference, the error increases with the distance to the beacons. It is also interesting to note that $1/|D|$ has a similar shape than the position or orientation errors (except in the particular case of collinear beacons). It can be proven [starting from (20) and (21)] by a detailed sensitivity analysis of the robot position error with respect to angles, that

$$\Delta d_R \simeq \frac{1}{|D|}\, \Delta\phi\, f(.) \tag{31}$$

where $\Delta\phi$ is the angle error (assumed to be the same for the three angles), and $f(.)$ is some function of all the other parameters (see the appendix for details). This confirms our claim that $1/|D|$ can be used as an approximation of the position error. Furthermore, one can observe from the graphic scales, that the position or orientation errors almost evolve

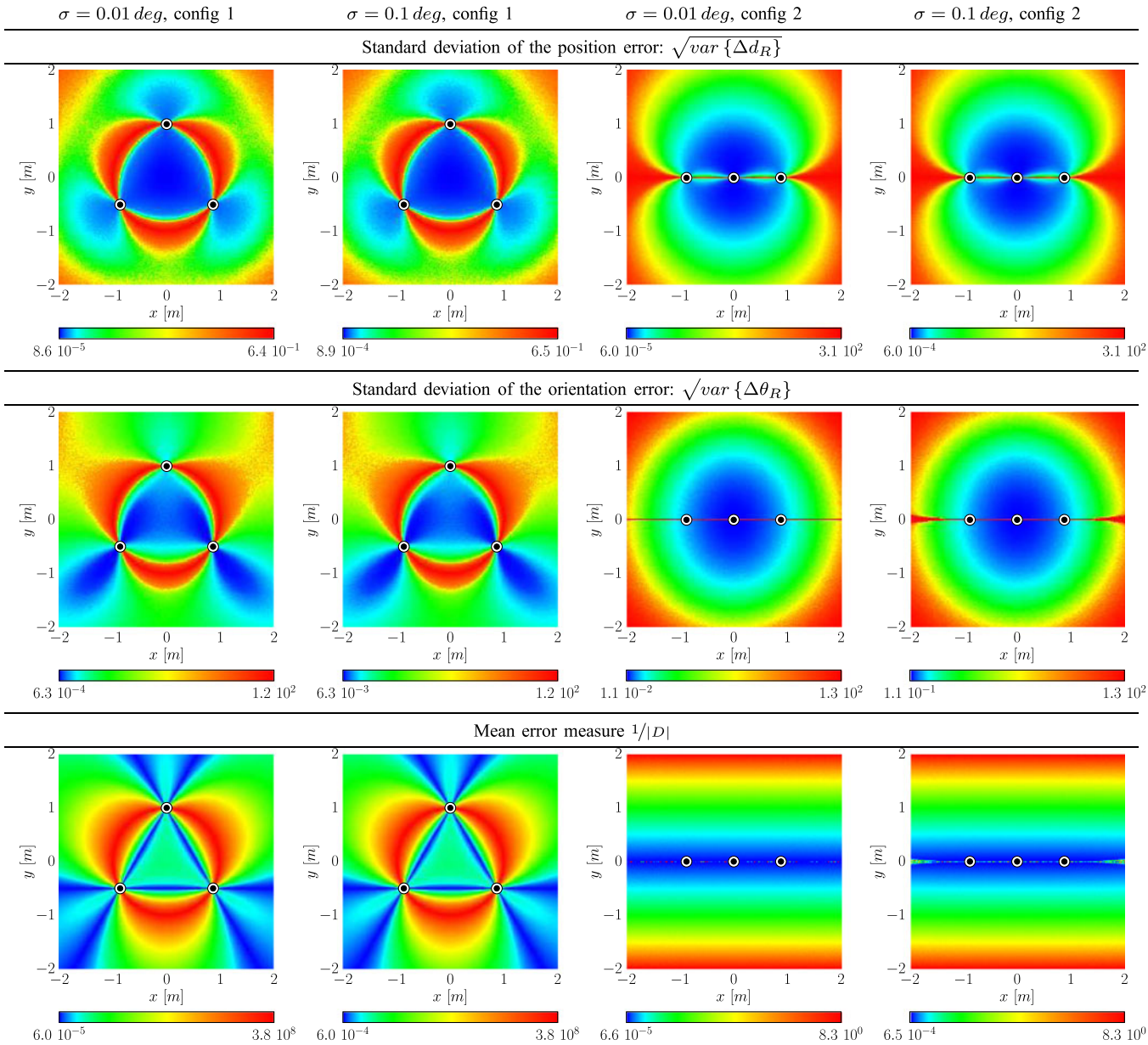


Fig. 5. Simulation results giving the position and orientation errors for noisy angle measurements. The beacon positions are represented by black and white dot patterns. The first and second columns provide the results for the first configuration for $\sigma = 0.01^\circ$ and $\sigma = 0.1^\circ$, respectively. The third and fourth columns provide the results for the second configuration for $\sigma = 0.01^\circ$ and $\sigma = 0.1^\circ$, respectively. Position errors are expressed in meters, the orientation error is expressed in degrees, and the error measure $1/|D|$ is in $1/m^2$. The graphics are displayed by using an histogram equalization to enhance their visual representation and interpretation.

linearly with angle errors, when they are small (look at the scale of the different graphics). Note that there is a small discrepancy in the symmetry of the simulated orientation error with respect to the expected behavior. This is explained because we used B_1 to compute the orientation [see (28)]. In addition, the histogram equalization emphasizes this small discrepancy.

B. Benchmarks

We have also compared the execution time of our algorithm to 17 other three object triangulation algorithms similar to ours (i.e., which work in the whole plane and for any beacon ordering). These algorithms have been introduced in Section II, and have been implemented after the author's guidelines.[5] Each algorithm has been running 10^6 times at random locations of the same square shaped area as that used for the error analysis. The last column of Table I provides the running times on an *Intel(R) Core(TM) i7 920 @ 2.67GHz* (6 GB RAM, Ubuntu 11.04, GCC 4.5.2). We used the C clock_gettime function to measure the execution times, in order to yield reliable results under timesharing. It appears that our algorithm is the fastest of all (about 30 % faster than the last best known algorithm of Font-Llagunes [18] and 5 % faster than the recent algorithm of

[5]The C source code used for the error analysis and benchmarks is available at http://www.ulg.ac.be/telecom/triangulation.

TABLE I
COMPARISON OF VARIOUS TRIANGULATION ALGORITHMS TO OUR TOTAL ALGORITHM

	Algorithm	+	×	/	$\sqrt{x}$	*trigo*	time $(s)^\dagger$
	ToTal[1]	30	17	2	0	2	0.163
[25]	Ligas[1]	29	22	2	0	2	0.171
[18]	Font-Llagunes[1]	23	17	2	0	5	0.228
[9]	Cassini[2]	19	8	3	0	4	0.249
[10]	Cohen[1]	37	15	3	2	4	0.272
[12]	Easton[2]	22	24	1	0	5	0.298
[31]	McGillem[1]	37	18	5	2	8	0.340
[20]	Hmam[2]	29	11	3	3	9	0.428
[10]	Cohen[2]	26	11	3	2	11	0.437
[16]	Esteves[2]	43	14	2	2	11	0.471
[9]	Collins[2]	34	10	2	2	11	0.485
[31]	McGillem[2]	29	9	3	2	11	0.501
[9]	Kaestner[2]	28	10	3	2	11	0.504
[44]	Tsukiyama[1]	52	22	3	5	14	0.596
[45]	Zalama[1]	52	21	4	5	14	0.609
[34]	Tienstra[2]	33	18	8	3	9	0.640
[19]	Font-Llagunes[1]	62	25	6	1	8	0.648
[28]	Madsen[2]	38	24	5	3	15	0.707

†For 10^6 executions on an Intel(R) Core(TM) i7 920 @ 2.67GHz.
[1]Geometric circle intersection. [2]Trigonometric solution.

Ligas [25]). In addition to the computation times, we have also reported the number of basic arithmetic computations, squared roots, and trigonometric functions used for each algorithm. This may help to choose an algorithm for a particular hardware architecture, which may have a different behavior for basic arithmetic computations, or complex functions such as square root or trigonometric functions. One can see that our algorithm has the minimum number of trigonometric functions, which is clearly related to the times on a classical computer architecture (see Table I). A fast algorithm is an advantage for error simulations, beacon placement, and beacon position optimization algorithms. Note that the algorithm of Ligas also uses the minimum number of trigonometric functions (two $\cot(.)$ computations) like ToTal, which explain why both algorithms are basically similar in terms of efficiency. However, the algorithm of Ligas does not provide a reliability measure, contrarily to our algorithm ToTal.

V. CONCLUSIONS

Most of the many triangulation algorithms proposed so far have major limitations. For example, some of them need a particular beacon ordering, have blind spots, or only work within the triangle defined by the three beacons. More reliable methods exist, however they have an increasing complexity or they require to handle certain spatial arrangements separately.

This paper presents a new three object triangulation algorithm based on the elegant notion of power center of three circles. Our new triangulation algorithm, which is called ToTal, natively works in the whole plane (except when the beacons and the robot are concyclic or collinear) and for any beacon ordering. Furthermore, it only uses basic arithmetic computations and two $\cot(.)$ computations. Comprehensive benchmarks show that our algorithm is faster than comparable algorithms, and simpler in terms of the number of operations. In this paper, we have compared the number of basic arithmetic computations, squared root, and trigonometric functions used for 17 known triangulation algorithms.

In addition, we have proposed a unique reliability measure of the triangulation result in the whole plane, and established by simulations that $1/|D|$ is a natural and adequate criterion to estimate the error of the positioning. To our knowledge, none of the algorithms of the same family provides such a measure. This error measure can be used to identify the pathological cases (critic circumference) or as a validation gate in Kalman filters based on triangulation.

For all these reasons, ToTal is a fast, flexible, and reliable three object triangulation algorithm. Such an algorithm is an excellent choice for many triangulation issues related to the performance or optimization, such as error simulations, beacon placement, or beacon position optimization algorithms. It can also be used to understand the sensitivity of triangulation with respect to the input angles. A fast and inexpensive algorithm is also an asset to initialize a positioning algorithm that internally relies on a Kalman filter.

APPENDIX

In this section, we detail the sensitivity analysis of the computed position. We start by computing the derivative of x_R with respect to the first angle ϕ_1

$$\frac{\partial x_R}{\partial \phi_1} = \frac{\partial k'_{31}}{\partial \phi_1}\frac{(y'_{12}-y'_{23})}{D} + \frac{\partial (y'_{12}-y'_{23})}{\partial \phi_1}\frac{k'_{31}}{D} \tag{32}$$

$$+ \frac{\partial \left(\frac{1}{D}\right)}{\partial \phi_1} k'_{31}(y'_{12}-y'_{23}) \tag{33}$$

$$= \frac{a_1(.)}{D} + \frac{a_2(.)}{D} - \frac{a_3(.)}{D^2} = \frac{1}{D}\, g_1(.) \tag{34}$$

where $g_1(.)$ is some function of all the other parameters. Similar results yield for the derivative of x_R with respect to the second and third angles, ϕ_2 and ϕ_3, respectively

$$\frac{\partial x_R}{\partial \phi_2} = \frac{1}{D}\, g_2(.), \qquad \frac{\partial x_R}{\partial \phi_3} = \frac{1}{D}\, g_3(.) \tag{35}$$

where $g_2(.)$ and $g_3(.)$ are some functions of all the other parameters. The total differential of x_R with respect to ϕ_1, ϕ_2, and ϕ_3 is given by

$$\Delta x_R = \frac{\partial x_R}{\partial \phi_1}\Delta\phi_1 + \frac{\partial x_R}{\partial \phi_2}\Delta\phi_2 + \frac{\partial x_R}{\partial \phi_3}\Delta\phi_3 \tag{36}$$

$$= \frac{1}{D}\Delta\phi\ (g_1(.) + g_2(.) + g_3(.)) \tag{37}$$

$$= \frac{1}{D}\Delta\phi\, g(.) \tag{38}$$

where we assumed that the three infinitesimal increments are equal $\Delta\phi = \Delta\phi_1 = \Delta\phi_2 = \Delta\phi_3$. A similar result yields for the total differential of y_R

$$\Delta y_R = \frac{1}{D}\Delta\phi\, h(.) \tag{39}$$

where $h(.)$ is some function of all the other parameters. Finally, we can compute Δd_R as follows:

$$\Delta d_R = \sqrt{(\Delta x_R)^2 + (\Delta y_R)^2} \tag{40}$$

$$= \frac{1}{|D|} |\Delta\phi| \sqrt{(g(.))^2 + (h(.))^2} \tag{41}$$

$$= \frac{1}{|D|} |\Delta\phi| f(.) \tag{42}$$

where $f(.)$ is some function of all the other parameters.

REFERENCES

[1] D. Avis and H. Imai, "Locating a robot with angle measurements," *J. Symbol. Comput.*, vol. 10, pp. 311–326, Aug. 1990.

[2] A. Bais, R. Sablatnig, and J. Gu, "Single landmark based self-localization of mobile robots," in *Proc. 3rd Can. Conf. Comput. Robot Vis. (CRV)*. IEEE Comput. Soc., Jun. 2006.

[3] M. Betke and L. Gurvits, "Mobile robot localization using landmarks," *IEEE Trans. Robot. Autom.*, vol. 13, no. 2, pp. 251–263, Apr. 1997.

[4] J. Borenstein, H. Everett, and L. Feng, "Where am I? Systems and methods for mobile robot positioning," Univ. Michigan, Ann Arbor, MI, USA, Tech. Rep., Mar. 1996.

[5] J. Borenstein, H. Everett, L. Feng, and D. Wehe, "Mobile robot positioning—Sensors and techniques," *J. Robot. Syst.*, vol. 14, no. 4, pp. 231–249, Apr. 1997.

[6] J. Borenstein and L. Feng, "UMBmark: A benchmark test for measuring odometry errors in mobile robots," in *Proc. Soc. Photo-Instrum. Eng.*, vol. 1001, Philadelphia, PA, USA, Oct. 1995, pp. 113–124.

[7] K. Briechle and U. Hanebeck, "Localization of a mobile robot using relative bearing measurements," *IEEE Trans. Robot. Autom.*, vol. 20, no. 1, pp. 36–44, Feb. 2004.

[8] M. Brkić, M. Lukić, J. Bajić, B. Dakić, and M. Vukadinović, "Hardware realization of autonomous robot localization system," in *Proc. Int. Conv. Inf. Commun. Technol., Elect. Microelectron.*, Opatija, Croatia, May 2012, pp. 146–150.

[9] R. Burtch, *Three point resection problem*, Surveying computations course notes 2005/2006, Ferris State Univ., Big Rapids, MI, USA, 2005, ch. 8, pp. 175–201.

[10] C. Cohen and F. Koss, "A comprehensive study of three object triangulation," in *Mobile Robots VII*. vol. 1831, Boston, MA, USA: SPIE, Nov. 1992, pp. 95–106.

[11] E. Demaine, A. López-Ortiz, J. Munro, Robot localization without depth perception, Proc. Scandinavian Workshop on Algorithm Theory (SWAT), ser. (Lecture Notes Comput. Sci.), vol. 2368. New York, NY, USA: Springer-Verlag, Jul. 2002, pp. 177–194,

[12] A. Easton and S. Cameron, "A gaussian error model for triangulation-based pose estimation using noisy landmarks," in *Proc. IEEE Conf. Robot., Autom. Mechatron.*, Bangkok, Thailand, Jun. 2006, pp. 1–6.

[13] J.-D. Eiden, *Géometrie Analytique Classique*. Paris, France: Calvage & Mounet, 2009.

[14] J. Esteves, A. Carvalho, and C. Couto, "Generalized geometric triangulation algorithm for mobile robot absolute self-localization," in *Proc. Int. Symp. Ind. Electron.*, Rio de Janeiro, Brazil, Jun. 2003, vol. 1, pp. 346–351.

[15] J. Esteves, A. Carvalho, and C. Couto, "An improved version of the generalized geometric triangulation algorithm," presented at the Eur.-Latin-Amer. Workshop Eng. Syst., Porto, Portugal, Jul. 2006.

[16] J. Esteves, A. Carvalho, and C. Couto, "Position and orientation errors in mobile robot absolute self-localization using an improved version of the generalized geometric triangulation algorithm," in *Proc. IEEE Int. Conf. Ind. Technol.*, Mumbai, India, Dec. 2006, pp. 830–835.

[17] J. Font-Llagunes and J. Batlle, "Mobile robot localization. Revisiting the triangulation methods," in *Proc. IFAC Int. Symp. Robot Control*, vol. 8, Santa Cristina Convent, Univ. Bologna, Bologna, Italy, Sep. 2006, pp. 340–345.

[18] J. Font-Llagunes and J. Batlle, "Consistent triangulation for mobile robot localization using discontinuous angular measurements," *Robot. Auton. Syst.*, vol. 57, no. 9, pp. 931–942, Sep. 2009.

[19] J. Font-Llagunes and J. Batlle, "New method that solves the three-point resection problem using straight lines intersection," *J. Surveying Eng.*, vol. 135, no. 2, pp. 39–45, May 2009.

[20] H. Hmam, "Mobile platform self-localization," in *Proc. Inf., Decis. Control*, Adelaide, Australia, Feb. 2007, pp. 242–247.

[21] H. Hu and D. Gu, "Landmark-based navigation of industrial mobile robots," *Int. J. Ind. Robot*, vol. 27, no. 6, pp. 458–467, 2000.

[22] A. Kelly, "Precision dilution in triangulation based mobile robot position estimation," *Intell. Auton. Syst.*, vol. 8, pp. 1046–1053, 2003.

[23] D. Kortenkamp, "Perception for mobile robot navigation: A survey of the state of the art," Nat. Aero. Space Admin., Tech. Rep. 19960022619, May 1994.

[24] C. Lee, Y. Chang, G. Park, J. Ryu, S.-G. Jeong, S. Park, J. Park, H. Lee, K.-S. Hong, and M. Lee, "Indoor positioning system based on incident angles of infrared emitters," in *Proc. Conf. IEEE Ind. Electron. Soc.*, vol. 3, Busan, Korea, Nov. 2004, pp. 2218–222.

[25] M. Ligas, "Simple solution to the three point resection problem," *J. Surveying Eng.*, vol. 139, no. 3, pp. 120–125, Aug. 2013.

[26] I. Loevsky and I. Shimshoni, "Reliable and efficient landmark-based localization for mobile robots," *Robot. Auton. Syst.*, vol. 58, no. 5, pp. 520–528, May 2010.

[27] M. Lukić, B. Miodrag, and J. Bajić, "An autonomous robot localization system based on coded infrared beacons," in *Research and Education in Robotics (EUROBOT)*, vol. 161, Prague, Czech Republic, Springer-Verlag, Jun. 2011, pp. 202–209.

[28] C. Madsen and C. Andersen, "Optimal landmark selection for triangulation of robot position," *Robot. Auton. Syst.*, vol. 23, no. 4, pp. 277–292, Jul. 1998.

[29] C. Madsen, C. Andersen, and J. Sorensen, "A robustness analysis of triangulation-based robot self-positioning," in *Proc. Int. Symp. Intell. Robot. Syst.*, Stockholm, Sweden, Jul. 1997, pp. 195–204.

[30] C. McGillem and T. Rappaport, "Infra-red location system for navigation of autonomous vehicles," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2, Philadelphia, PA, USA, Apr. 1988, pp. 1236–123.

[31] C. McGillem and T. Rappaport, "A beacon navigation method for autonomous vehicles," *IEEE Trans. Veh. Technol.*, vol. 38, no. 3, pp. 132–139, Aug. 1989.

[32] V. Pierlot and M. Van Droogenbroeck, "A simple and low cost angle measurement system for mobile robot positioning," in *Proc. Worksh. Circuits, Syst. Signal Process.*, Veldhoven, The Netherlands, Nov. 2009, pp. 251–254.

[33] V. Pierlot, M. Van Droogenbroeck, and M. Urbin-Choffray, A new three object triangulation algorithm based on the power center of three circles, Research and Education in Robotics (EUROBOT), ser. Commun. Comput. Inf. Sci., vol. 161. New York, NY, USA: Springer-Verlag, 2011, pp. 248–262.

[34] J. Porta and F. Thomas, "Simple solution to the three point resection problem," *J. Surveying Eng.*, vol. 135, no. 4, pp. 170–172, Nov. 2009.

[35] J. Sanchiz, J. Badenas, and F. Pla, "Control system and laser-based sensor design of an automonous vehicle for industrial environments," *Soc. Photo-Instrum. Eng.*, vol. 5422, no. 1, pp. 608–615, 2004.

[36] I. Shimshoni, "On mobile robot localization from landmark bearings," *IEEE Trans. Robot. Autom.*, vol. 18, no. 6, pp. 971–976, Dec. 2002.

[37] S. Shoval and D. Sinriech, "Analysis of landmark configuration for absolute positioning of autonomous vehicles," *J. Manuf. Syst.*, vol. 20, no. 1, pp. 44–54, 2001.

[38] A. Siadat and S. Vialle, "Robot localization, using p-similar landmarks, optimized triangulation and parallel programming," presented at the IEEE Int. Symp. Signal Process. Inform. Technol., Marrakesh, Morocco, Dec. 2002.

[39] D. Sinriech and S. Shoval, "Landmark configuration for absolute positioning of autonomous vehicles," *IIE Trans.*, vol. 32, no. 7, pp. 613–624, Jul. 2000.

[40] H. Sobreira, A. Moreira, and J. Esteves, "Characterization of position and orientation measurement uncertainties in a low-cost mobile platform," in *Proc. Portuguese Conf. Automat. Contr.*, Coimbra, Portugal, Sep. 2010, pp. 635–640.

[41] H. Sobreira, A. Moreira, and J. Esteves, "Low cost self-localization system with two beacons," in *Proc. Int. Conf. Mobile Robots Competit.*, Leiria, Portugal, Mar. 2010, pp. 73–77.

[42] O. Tekdas and V. Isler, "Sensor placement algorithms for triangulation based localization," in *Proc. IEEE Int. Conf. Robot. Autom.*, Rome, Italy, Apr. 2007, pp. 4448–4453.

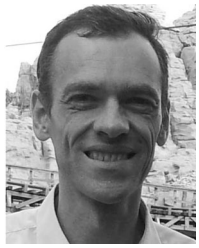
[43] O. Tekdas and V. Isler, "Sensor placement for triangulation-based localization," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 3, pp. 681–685, Jul. 2010.

[44] T. Tsukiyama, "Mobile robot localization from landmark bearings," in *World Congr. Fundam. Appl. Metrol.*, Lisbon, Portugal, Sep. 2009, pp. 2109–211.

[45] E. Zalama, S. Dominguez, J. Gómez, and J. Perán, "A new beacon-based system for the localization of moving objects," presented at the *IEEE Int. Conf. Mechatron. Mach. Vis. Pract.*, Chiang Mai, Thailand, Sep. 2002.
[46] E. Zalama, S. Dominguez, J. Gómez, and J. Perán, "Microcontroller based system for 2D localisation," *Mechatronics*, vol. 15, no. 9, pp. 1109–1126, Nov. 2005.
[47] O. Fuentes, J. Karlsson, W. Meira, R. Rao, T. Riopka, J. Rosca, R. Sarukkai, M. Van Wie, M. Zaki, T. Becker, R. Frank, B. Miller, and C. M. Brown, "Mobile robotics 1994," Tech. Rep. 588, Comput. Sci. Dept., Univ. Rochester, Rochester, NY, Jun. 1995 (see http://www.cs.cmu.edu/afs/cs.cmu.edu/Web/People/motionplanning/papers/sbp_papers/integrated2/fuentas_mobile_robots.pdf).
[48] T. Heath, *Euclid: The Thirteen Books of The Elements*. Dover, 1956.

Vincent Pierlot (M'13) was born in Liège, Belgium. He received the electrical engineering and Ph.D. degrees from the University of Liège, in 2006 and 2013, respectively.

He is the architect of a new positioning system that has been used during the EUROBOT contest for several years. His research interests are mainly focused on electronics, embedded systems, motion analysis, positioning, triangulation, and robotics.



Marc Van Droogenbroeck (M'99) received the electrical engineering and Ph.D. degrees from the University of Louvain, Louvain-la-Neuve, Belgium, in 1990 and 1994, respectively. While working toward the Ph.D., he spent two years with the Center of Mathematical Morphology, the School of Mines, Paris, France.

In April 1994, he joined the New Development Department of Belgacom. He was the Head of the Belgian Delegation within the ISO/MPEG Committee and as a Representative to the World Wide Web Consortium for two years. In 2003, he was a Visiting Scientist at CSIRO, Sydney, Australia. Since 1998, he has been a Member of the Faculty of Applied Sciences, the University of Liège, Liège, Belgium, where he is currently a Professor. His current research interests include computer vision, 3-D vision, real-time processing, motion analysis, gait analysis, telecommunications, positioning, and robotics.