Virtual raw anchor coordinates: A new localization paradigm

Huc, Florian; Jarry, Aubin; Leone, Pierre; Rolim, Jose

How to cite

HUC, Florian et al. Virtual raw anchor coordinates: A new localization paradigm. In: Theoretical computer science, 2012, vol. 453, p. 44–53. doi: 10.1016/j.tcs.2011.11.023

This publication URL: https://archive-ouverte.unige.ch//unige:32261
Publication DOI: 10.1016/j.tcs.2011.11.023

Virtual Raw Anchor Coordinates: A New Localization Paradigm*

Florian Huc, Aubin Jarry, Pierre Leone, and José Rolim

Computer Science Department
University of Geneva
Battelle A, route de Drize 7
1227 Carouge, Switzerland
firstname.name@unige.ch

Abstract. A wide range of applications in wireless sensor networks rely on the location information of the sensing nodes. However, traditional localization techniques are dependent on hardware that is sometimes unavailable (e.g. GPS), or on sophisticated virtual localization calculus which have a costly overhead.

Instead of actually localizing nodes in the physical two-dimensional Euclidean space, we use directly the raw distance to a set of anchors to produce multi-dimensional coordinates. We prove that the image of the physical two-dimensional Euclidean space is a two-dimensional surface, and we show that it is possible to adapt geographic routing strategies on this surface, simply, efficiently and successfully.

1 Introduction

Localization plays an important role in wireless sensor networks. Indeed, if the identity of each sensor is used in the MAC layer to differentiate the neighbors of each node, what is important at the application level is the locations inside the monitored area, not individual sensors. Indeed, many applications need topological information for internal interventions such as tracking, or for external interventions such as the shipment of supplies or rescue team intervention. As such, information is retrieved from specific locations; communications are sent between locations; and network actions take place at specific locations, be it the movement of sensors (if they are so equipped) sleep schedule reconfigurations, or generally reprogramming to adapt to a new situation in the network. From the point of view of sensors, topology awareness enables them to know in which area of the network they are, and to appreciate the distance to and from particular areas of interest. Since sensors generally do not have routing tables that are costly to maintain, it also allows the use of the topological properties of the network for routing. This is generally referred to as geographic routing. Of the many efficient geographic routing algorithms that have been devised, we cite GFG/GPSR [5,12] and OAFR [15] which use greedy routing and face routing on

* Work partially funded by project FRONTS 215270.

C. Scheideler (Ed.): ALGOSENSORS 2010, LNCS 6451, pp. 161–175, 2010.

a planarized connectivity graph. When authorizing the use of a bit of memory at each node, early obstacle detection algorithms have been proposed [11,19].

In order to obtain coordinates, the nodes may rely on interferometry [17] or on an external source of knowledge, such as a GPS or Galileo unit, pressure or magnetic field measurements, and so on. Coordinates may be also manually assigned by men, robots or unmanned aerial vehicles (UAV). This dependency on hardware or on external intervention entails a lot of drawbacks for wireless sensor networks. First of all, hardware devices have a monetary cost, take up space and weight, and consume energy, all of which are critical resources when designing miniaturized motes that will be dispatched in the thousands. Secondly, external intervention is not self-contained and thus may not be available. As an illustration, GPS systems are not available underground, inside parts of buildings, under sea, in case of satellite failure or if sensors are deployed on a planet not equipped with satellites.

In order to reduce the dependency on external positioning, only a handful of sensors – usually called anchors – may be positioned at start, whereas regular sensors have access to relative spatial information using optional hardware (angle measurements, distance measurements by time difference of arrival between sound and radio signals, etc.) or using their access to the wireless medium: distances may be measured with the strength of received signals, or more simply, by hop-count. A nice introduction on the various positioning methods for networks may be found in [21]. Positioning methods can be classified into three main types, whether one achieves absolute positioning, relative positioning or only local positioning. In absolute positioning, the coordinate system has a global coherence within the system but also with respect to exterior coordinates. Relative positioning is only coherent within the network, whereas local positioning just asks for local coherence. In [16], three localization algorithms are compared, namely Ad-hoc positioning, Robust positioning, and N-hop multilateration. These three algorithms have a common three phase structure: they first determine node to anchor distances, then compute node positions, and optionally refine the positions through an iterative procedure. Some authors improved accuracy by using angles measurement [6,25,26].

Dependency on external intervention or hardware is further reduced by having no sensor with extra capabilities or information. Some authors thus propose to compute virtual coordinates instead of real ones. Indeed, many algorithms do not need actual two dimensional coordinates, but the relative position of the nodes. In [3,4] the authors call this problem the training problem and propose an algorithm allowing the sensors (which are asynchronous) to estimate their distance to a central sink. This algorithm needs the sink to be able to emit to all the nodes of the network, and its output is a partition of the network in rings. If the sink is also equipped by directional antennas, it is also feasible to partition the networks in slices. Hence the authors propose to use the ring number and the slice number of each nodes as coordinates. Other papers [7,18,23,24] propose to compute virtual coordinates from the distance between nearby nodes and have the advantage of not needing anchors. Still this approach may lead to

unsolvable issues if the network is not dense enough. To avoid this, in [23], the authors use a mobile unit to assist in measuring the distance between nodes. It also helps to improve accuracy. For these papers, the key point is to obtain sufficient data on inter node distances. In [14], the authors study the problem of computing missing inter-nodes distances from known ones. They propose an algorithm, which given distances from all nodes to some anchors, recompute the unknown distances with an arbitrary precision. They also discuss complexity and non approximability issues.

Virtual coordinates have also been discussed in other contexts. In the context of peer-to-peer networks embedded in the Internet, Hotz proposed in [9] to use the distance to anchors as virtual coordinates while using only triangle inequalities to estimate distances. Following this trend, Ng and Zhang proposed in [20] to first compute coordinates for the anchors (called landmarks in their paper) by using linear system resolution tools, and then to compute locally coordinates for the nodes (by solving smaller linear systems). Not only were the experimental results quite good, it was theoretically proved in [14] that provided that the anchors were randomly selected, in a sufficiently large number, and provided that the distance between anchors was respected in the new coordinate system, the distortion of distances in the new coordinate system could be arbitrarily low. In the context of air navigation, Farell et all [8] considered the idea of using distances rather than coordinates and proposed that collision avoidance and other time-critical algorithms used GPS pseudo-ranged rather than derived coordinates. In this paper, we discard any preprocessing technique and propose to directly use raw distance information. We study routing algorithms using directly the distance to the anchors as coordinates, as first proposed in [10], without computing from them 2-dimensional coordinates. In Section 2 we precisely describe how the idea is implemented, in Section 3 we analyze how a message sent towards a destination performs in the new coordinate system, and we present some simulation results in Section 4.

2 Implementation

Current localization methods rely on raw information computed externally from normal sensing nodes (exact location of some anchors), and on raw information computed locally in normal sensing nodes (distance to anchors, angle measurements). In this paper, we do use the information about the distance to some anchors, but we completely discard any physical information that the anchors might have. This gives much more flexibility in the way sensor networks are deployed: anchors might be external entities, as planes or robots; anchors might be specialized nodes whose only purpose is to emit a strong signal, or they might be randomly chosen sensors which advertise their distance to the other nodes.

We build a multi-dimensional coordinate system using directly the raw information, i.e. the distance to the anchors. Given a node at location X, we define the multi-dimensional coordinates $f(X)$ of this node as its distance to the anchors at location $A_1, A_2, \ldots A_n$:

$$f : X \to \begin{pmatrix} d(X, A_1) \\ d(X, A_2) \\ \dots \\ d(X, A_n) \end{pmatrix}.$$

We call this function the anchor coordinates function, and we call these multi-dimensional coordinates the anchor coordinates. Whereas any distance function, such as hop count, may be used [10], in section 3 we pay a special attention to the properties of f when d is the Euclidean distance.

In the next subsection we discuss the computation costs that are specific to using multi-dimensional coordinates. We then go into the details of greedy routing implementation, and into the details of rotating multi-dimensional vectors.

2.1 Computation Cost

While saving on initialization overhead, multi-dimensional routing causes some additional computation costs when sending messages compared to traditional two-dimensional routing. Here is a break-down of various vector operations:

Operation	n**-dimensional**	**2-dimensional**
$\overrightarrow{u} + \overrightarrow{v}$	n additions	2 additions
$k\overrightarrow{u}$	n multiplications	2 multiplications
$\overrightarrow{u} \cdot \overrightarrow{v}$	n multiplications $n-1$ additions	2 multiplications 1 addition
$\frac{\overrightarrow{u}}{\|\|\overrightarrow{u}\|\|}$	1 sqrt extraction 1 inversion $2n$ multiplications $n-1$ additions	1 sqrt extraction 1 inversion 2 multiplications 1 addition

Note that additions and multiplications typically use 1 CPU cycle, whereas the expensive operations (square root extraction, inversion) stay the same in multi-dimensional routing as in traditional two-dimensional routing. We also point out that these computation costs are not communication costs and are lower in terms of energy consumption by some order of magnitude.

2.2 Greedy Routing

Greedy routing is the most basic geographic routing algorithm. It consists in following the direction to the destination. This basic strategy is widely used as a default mode in most geographic routing protocols. When a node at location X which wants to send a message towards a final destination at location D, three implementations of greedy routing are routinely used:

1. *(canonical)* for each neighbor location X', compute the distance $d(X', D)$ and send the message to the neighbor which is closest to D. Alternatively, compute $\overrightarrow{X'D} \cdot \overrightarrow{X'D}$ instead of $d(X', D)$.

2. for each neighbor location X', compute the scalar product $\overrightarrow{XX'} \cdot \overrightarrow{XD}$ and select the neighbor with the best result.
3. for each neighbor location X', compute the scalar product $\frac{\overrightarrow{XX'}}{||\overrightarrow{XX'}||} \cdot \overrightarrow{XD}$ and select the neighbor with the best result.

These three implementations are valid for any number of coordinates.

2.3 Rotation

When greedy strategies fail, a number of two-dimensional routing algorithms fall back on more sophisticated routing modes that use rotations or angle computations [5,12,22]. When using two dimensions, a rotation is typically defined by $rot_\alpha : (x, y) \rightarrow (x \cos\alpha + y \sin\alpha, y \cos\alpha - x \sin\alpha)$. We can't define such a rotation in n dimensions ($n \geq 3$). However, if we assume that our sensors were on a two-dimensional physical plane in the first place, then they are distributed over a two-dimensional surface in the multi-dimensional space (more on this in section 3). We do the following:

1. compute an orthonormal basis $(\overrightarrow{i}, \overrightarrow{j})$ of the tangent plane in $f(X)$ (see section 3).
2. express vectors $\overrightarrow{u}$ as $x_u \overrightarrow{i} + y_u \overrightarrow{j} + \overrightarrow{\epsilon_u}$ by computing $x_u = \overrightarrow{u} \cdot \overrightarrow{i}$ and $y_u = \overrightarrow{u} \cdot \overrightarrow{j}$. We assume that $\overrightarrow{u}$ is close to the tangent plane in $f(X)$, which means that we ignore in fact $\overrightarrow{\epsilon_u}$.

Rotations are then normally carried out on the tangent plane. The sensitive part is to compute $(\overrightarrow{i}, \overrightarrow{j})$ and to make sure that the orientation of the surface is preserved when routing the message (taking the surface upside-down has the undesirable effect of negating angles). Given a node at location X, a destination at D, and a basis $(\overrightarrow{i_{old}}, \overrightarrow{j_{old}})$ inherited from a previous node, we do the following:

1. choose two neighbors at position X_1 and X_2
   - either arbitrarily (low quality, inexpensive)
   - or such that $\frac{|\overrightarrow{XX_1} \cdot \overrightarrow{XX_2}|}{||\overrightarrow{XX_1}||||\overrightarrow{XX_2}||}$ is minimal (i.e. choose $\overrightarrow{XX_1}$ and $\overrightarrow{XX_2}$ as orthogonal as possible)
2. compute $\overrightarrow{i} = \frac{\overrightarrow{XX_1}}{||\overrightarrow{XX_1}||}$
3. compute $\overrightarrow{u} = \overrightarrow{XX_2} - (\overrightarrow{i} \cdot \overrightarrow{XX_2})\overrightarrow{i}$
4. compute $\overrightarrow{v} = \frac{\overrightarrow{u}}{||\overrightarrow{u}||}$
5. compute $\sigma = (\overrightarrow{i} \cdot \overrightarrow{i_{old}})(\overrightarrow{v} \cdot \overrightarrow{j_{old}}) - (\overrightarrow{i} \cdot \overrightarrow{j_{old}})(\overrightarrow{v} \cdot \overrightarrow{i_{old}})$.
6. if $\sigma \geq 0$ then set $\overrightarrow{j} = \overrightarrow{v}$, else set $\overrightarrow{j} = -\overrightarrow{v}$.

Note that many algorithms using angles use normalized vectors. Therefore, most of the normalization cost when computing the basis $(\overrightarrow{i}, \overrightarrow{j})$ is not an additional cost of multi-dimensional routing.

2.4 Cross Link Detection Protocol

Many geographic routing algorithms rely on a planarized version of the communication graph, and various techniques exist to compute that graph. A common assumption is that the network forms a Unit Disc Graph or an approximation of it, which enables the computation of Gabriel Graphs (see for instance [2]). It also has been argued that this assumption is unrealistic [13], and anyhow, the utilization of virtual coordinates can distort the length of communication links in such a way that UDG properties are not preserved. We chose to adapt CLDP [13], a distributed planarization algorithm where no assumption is made on the communication graph. CLDP works in a distributed manner: it tests each link uv by computing a circuit from one node to the other and looking if a link of the circuit crosses uv. If this is the case, one of the crossing links is deleted. Links are tested until no crossing is detected.

When a network is embedded in a k dimensional space with $k \geq 3$, its links will generally not cross each other, but this has no bearing on whether the communication graph is planar or not[1]. Therefore, in order to implement CLDP using our virtual coordinates we have to understand the crossing of links according to some projection on a surface. In particular, given a 2 dimensional plane, we can assess the planarity of the communication graph by projecting the links of the network into to this plane. We implemented CLDP using virtual raw anchor coordinates as follows (when testing an edge uv):

1. Compute a plane P approximately tangent to the surface $f(\mathbb{R}^2)$ at v. P is computed by choosing a third node among the neighbors of v.
2. Create a circuit from v to u using the right hand rule in the projected image of the network on P.
3. If the projection of the circuit on P intersects the projection of uv on P, delete uv.

2.5 Greedy Perimeter Stateless Routing

GFG/GPSR, initially proposed in [5,12], is a geographic routing algorithm which guarantees a 100% message delivery. Its default mode is to use greedy routing. However it has a secondary mode which allows messages to get out of a local minimum. This secondary mode uses a planarized version of the communication graph. In this planarized graph, an obstacle inducing a local minimum is also a face. The secondary mode of GFG/GPSR is then the following: the local minimum is called the entry point, and the message is routed along the face corresponding to the obstacle using the right hand rule until it reaches a node closer to the destination than the entry point. Greedy mode routing is then resumed. Greedy mode routing implementation with multidimensional coordinates is straightforward (see Subsection 2.2). The implementation of the secondary mode is done as follows:

[1] For instance any graph can be represented in a 3 dimensional space without any edge intersection.

1. Compute a plane P tangent to the surface at the entry point.
2. Choose the next node using the right hand rule in the projection of the planarized network on P.
3. If the next node is closer to the destination than the entry point, resume greedy routing.

It is also possible to compute a new tangent plane at each step, and preserve a coherent orientation between tangent planes (so that the right hand rule has a meaning), as done for the implementation of GRIC in [10]. The simulations results in Section 4 were done with a single tangent plane per secondary mode, but using multiple tangent planes nevertheless yielded nearly identical results.

3 Algebraic Analysis

In the plane with Euclidean distance, any node has a pair of physical coordinates $X = (x, y)$. We denote by $A_i = (x_i, y_i)$ the physical coordinates of the i^{th} anchor. The anchor coordinates function is a function from $\mathbb{R}^2 \to \mathbb{R}^n$ defined by

$$f : (x, y) \to \begin{pmatrix} \sqrt{(x-x_1)^2 + (y-y_1)^2} \\ \sqrt{(x-x_2)^2 + (y-y_2)^2} \\ \cdots \\ \sqrt{(x-x_n)^2 + (y-y_n)^2} \end{pmatrix}.$$

Since the functions $f_i : (x, y) \to \sqrt{(x-x_i)^2 + (y-y_i)^2}$ are continuous and C^∞ except in (x_i, y_i), we show that with three or more anchors that are not on the same line, the image $f(\mathbb{R}^2)$ in $\mathbb{R}^n$ is a *continuous surface* (claim 3.1). Figure 1 represents the image of f, when there are three anchors at location $(0, 0)$, $(0, 1)$ and $(1, 0)$.

First, we describe in subsection 3.1 the vector spaces that are tangent to $f(\mathbb{R}^2)$. Next, we express in subsection 3.2 what is the physical direction of messages that use the greedy strategy with virtual coordinates. This physical direction produces a curve that approximates the paths followed by messages. We discuss in subsection 3.3 what are the convergence conditions on $f(\mathbb{R}^2)$ under which the curve ends at the destination, and prove a bound on the length of this curve. Finally, we study in subsection 3.4 how the placement of anchors affect the convergence conditions and how we can guarantee that they are met.

3.1 Tangent Space

At any point $f(x, y)$, the surface $f(\mathbb{R}^2)$ has a tangent vector space spanned by the two vectors $\frac{\partial f}{\partial x}(x, y)$ and $\frac{\partial f}{\partial y}(x, y)$. We have

$$\frac{\partial f}{\partial x}(x, y) = \begin{pmatrix} \frac{x-x_1}{\sqrt{(x-x_1)^2+(y-y_1)^2}} \\ \frac{x-x_2}{\sqrt{(x-x_2)^2+(y-y_2)^2}} \\ \cdots \\ \frac{x-x_n}{\sqrt{(x-x_n)^2+(y-y_n)^2}} \end{pmatrix} \quad \text{and} \quad \frac{\partial f}{\partial y}(x, y) = \begin{pmatrix} \frac{y-y_1}{\sqrt{(x-x_1)^2+(y-y_1)^2}} \\ \frac{y-y_2}{\sqrt{(x-x_2)^2+(y-y_2)^2}} \\ \cdots \\ \frac{y-y_n}{\sqrt{(x-x_n)^2+(y-y_n)^2}} \end{pmatrix}.$$

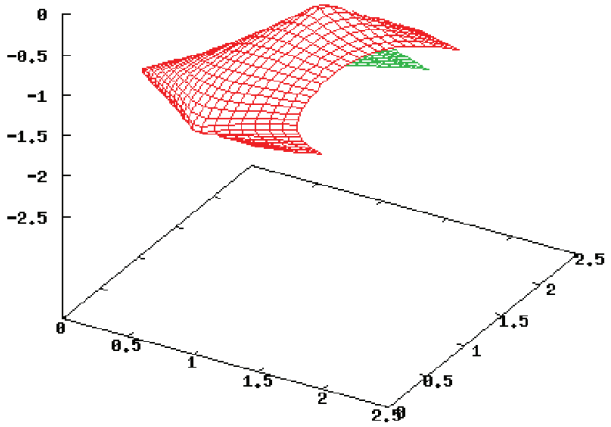


Fig. 1. Representation of the distance to three anchors

Claim. The vector space that is tangent to the surface $f(\mathbb{R}^2)$ in $f(X)$ is two-dimensional if and only if the node X and the anchors $A_1, A_2, \ldots, A_n$ are not situated on a single line in the physical space.

Proof. The tangent vector space is two-dimensional if and only if $\frac{\partial f}{\partial x}(x,y)$ and $\frac{\partial f}{\partial y}(x,y)$ are not collinear. Conversely $\frac{\partial f}{\partial x}(x,y)$ and $\frac{\partial f}{\partial y}(x,y)$ are collinear if and only if there is $\alpha \in [0, 2\pi[$ such that $\frac{\partial f}{\partial x}(x,y)\cos\alpha + \frac{\partial f}{\partial y}(x,y)\sin\alpha = 0$. By changing the physical coordinates into $u = x\cos\alpha + y\sin\alpha$ and $v = y\cos\alpha - x\sin\alpha$ (we also set $u_i = x_i\cos\alpha + y_i\sin\alpha$ and $v_i = y_i\cos\alpha - x_i\sin\alpha$), we express the tangent vector space with the two vectors

$$\frac{\partial f}{\partial u}(X) = \begin{pmatrix} \frac{u-u_1}{\sqrt{(u-u_1)^2+(v-v_1)^2}} \\ \frac{u-u_2}{\sqrt{(u-u_2)^2+(v-v_2)^2}} \\ \cdots \\ \frac{u-u_n}{\sqrt{(u-u_n)^2+(v-v_n)^2}} \end{pmatrix} \quad \text{and} \quad \frac{\partial f}{\partial v}(X) = \begin{pmatrix} \frac{v-v_1}{\sqrt{(u-u_1)^2+(v-v_1)^2}} \\ \frac{v-v_2}{\sqrt{(u-u_2)^2+(v-v_2)^2}} \\ \cdots \\ \frac{v-v_n}{\sqrt{(u-u_n)^2+(v-v_n)^2}} \end{pmatrix}.$$

Therefore, we have $\frac{\partial f}{\partial u}(X) = 0$ if and only if for all $i \in \{1 \ldots n\}, u = u_i$.

When the two vectors $\frac{\partial f}{\partial x}(x,y)$ and $\frac{\partial f}{\partial y}(x,y)$ are not collinear, then the Jacobian matrix

$$J_f(X) = J_f(x,y) = \begin{pmatrix} \frac{x-x_1}{\sqrt{(x-x_1)^2+(y-y_1)^2}} & \frac{y-y_1}{\sqrt{(x-x_1)^2+(y-y_1)^2}} \\ \frac{x-x_2}{\sqrt{(x-x_2)^2+(y-y_2)^2}} & \frac{y-y_2}{\sqrt{(x-x_2)^2+(y-y_2)^2}} \\ \cdots & \cdots \\ \frac{x-x_n}{\sqrt{(x-x_n)^2+(y-y_n)^2}} & \frac{y-y_n}{\sqrt{(x-x_n)^2+(y-y_n)^2}} \end{pmatrix}$$

defines a morphism of the physical plane into the vector space tangent to $f(\mathbb{R}^2)$ at $f(x, y)$. Given a node at position X in the physical space and its neighbors at position $X_1, X_2, \ldots, X_\delta$, it is not unreasonable to assume that for all i, $f(X_i)$ is close to the Taylor expansion $f(X) + J_f(X)(\overrightarrow{XX_i})$ in the affine space tangent to $f(\mathbb{R}^2)$ in $f(X)$.

3.2 Directional Vector

In a greedy routing strategy using virtual coordinates, the neighbor X' of choice will be a maximum for some scalar product $\overrightarrow{f(X)f(X')} \cdot \overrightarrow{f(X)f(D)}$.

Claim. Given two physical positions $X, D \in \mathbb{R}^2$, the function $s_X : \mathbb{R}^2 \to \mathbb{R}$ such that for any vector $\overrightarrow{XX'} \in \mathbb{R}^2$, $s_X(\overrightarrow{XX'})$ is the scalar product of $J_f(X)(\overrightarrow{XX'})$ by $\overrightarrow{f(X)f(D)}$ is a linear form that can be expressed as

$$\overrightarrow{XX'} \to \overrightarrow{XX'} \cdot \sum_i \alpha_i \overrightarrow{XA_i}$$

where $\alpha_i = \frac{d(X,A_i)-d(D,A_i)}{d(X,A_i)}$.

Proof. The transformation $\overrightarrow{XX'} \to J_f(X)(\overrightarrow{XX'})$ is a linear function. Since the scalar product by $\overrightarrow{f(X)f(D)}$ is a linear form, s_X is also a linear form. We may decompose the vector $\overrightarrow{f(X)f(D)}$ into $\sum_i (d(D, A_i) - d(X, A_i))\mathbf{1}_i$ where $\mathbf{1}_i$ is the multi-dimensional vector with 1 as its i^{th} coordinate and zeroes everywhere else. In this manner, $s_X = \sum_i s_{X,i}$ where

$$s_{X,i}(\overrightarrow{XX'}) = (d(D, A_i) - d(X, A_i))J_f(X)(\overrightarrow{XX'}) \cdot \mathbf{1}_i$$

$$J_f(X)(\overrightarrow{XX'}) \cdot \mathbf{1}_i = \frac{(x - x_i)(x' - x) + (y - y_i)(y' - y)}{\sqrt{(x - x_i)^2 + (y - y_i)^2}}.$$

Thus $s_{X,i}$ can be expressed as

$$\overrightarrow{XX'} \to \overrightarrow{XX'} \cdot \frac{d(X, A_i) - d(D, A_i)}{d(X, A_i)} \overrightarrow{XA_i}.$$

Given a node at physical location X and a destination $D \in \mathbb{R}^2$, we call *apparent destination* related to D in X the location

$$D' = X + \sum_i \alpha_i \overrightarrow{XA_i} = X + \sum_i \frac{d(X, A_i) - d(D, A_i)}{d(X, A_i)} \overrightarrow{XA_i}.$$

3.3 Virtual Consistency

We say that the anchor coordinate system is virtually consistent at distance r for a physical destination $D \in \mathbb{R}^2$, if at every point $X \neq D$ such that $f(X)$ is in a closed metric ball of center $f(D)$ and radius r, $s_X \neq 0$. Note that $s_x = 0$ if and only if the multi-dimensional vector $\overrightarrow{f(X)f(D)}$ is orthogonal to the vector space tangent to $f(\mathbb{R}^2)$ in $f(X)$. It is also equivalent to state that the anchor coordinate system is virtually consistent at distance r for a physical destination $D \in \mathbb{R}^2$, if no closed metric ball centered on $f(D)$ and of radius $0 < r' \leq r$ is tangent to $f(\mathbb{R}^2)$.

Claim. If the anchor coordinate system is virtually consistent at distance r for a physical destination $D \in \mathbb{R}^2$, then there is $\lambda \in \mathbb{R}^+$ such that for any point X_0 with $f(X_0)$ in a closed metric ball of center $f(D)$ and radius r we have a curve $c[0,1] \in \mathbb{R}^2$ that verifies:

- $c : [0,1] \to \mathbb{R}^2$ is a derivable function,
- $c(0) = X_0$ and $c(1) = D$,
- At any point $t \in [0,1($, the vector $\frac{\partial c}{\partial t}(t)$ is collinear with the vector $\overrightarrow{c(t)D'_t}$ where D'_t is the apparent destination related to D in $c(t)$.
- $\int_0^1 ||\frac{\partial c}{\partial (t)}||dt \leq \lambda d(X_0, D)$.

Proof. Let k be the largest positive number such that for any point $X = (x, y)$ with $f(X)$ in a closed metric ball of center $f(D)$ and radius r, the orthogonal projection of $\overrightarrow{f(X)f(D)}$ on the vector space defined by the two vectors $\frac{\partial f}{\partial x}(x, y)$ and $\frac{\partial f}{\partial y}(x, y)$ has a norm greater than or equal to $kd(X, D)$. Since f is a continuous function, the set of physical positions X such that $d(f(X), f(D)) \leq r$ is compact subset of $\mathbb{R}^2$. Therefore, if k was equal to zero, then there would be a point $X \neq D$ in the ball such that $\overrightarrow{XD}$ is orthogonal to the surface $f(\mathbb{R}^2)$, which we excluded in our assumptions.

Let $c : [0,1] \to \mathbb{R}^2$ be the function defined by $c(0) = X_0$ and such that $\frac{\partial (f \circ c)}{\partial t}(t)$ is the orthogonal projection of $\frac{k^{-2} d(f(X_0), f(D))}{d((f \circ c)(t), f(D))} \overrightarrow{(f \circ c)(t) f(D)}$ on the vector space defined by the two vectors $\frac{\partial f}{\partial x}(c(t))$ and $\frac{\partial f}{\partial y}(c(t))$. Since

$$\frac{\partial (f \circ c)}{\partial t}(t) \cdot \frac{\overrightarrow{(f \circ c)(t) f(D)}}{||\overrightarrow{(f \circ c)(t) f(D)}||} \geq k || \frac{\partial (f \circ c)}{\partial t}(t) ||$$

we can see that

$$\frac{\partial d((f \circ c)t), f(D))}{\partial t}(t) \geq d(f(X_0), f(D))$$

which implies that $c(1) = D$. The norm of $\frac{\partial c}{\partial t}(t)$ is smaller than or equal to $||(J_f(c(t)))^{-1}|| d(f(X_0), f(D))$, which means that

$$\int_0^1 ||\frac{\partial c}{\partial(t)}||dt \leq \max_{t\in[0,1]} ||(J_f(c(t)))^{-1}||d(f(X_0, f(D)))$$

$$\int_0^1 ||\frac{\partial c}{\partial(t)}||dt \leq \sqrt{n} \max_{t\in[0,1]} ||(J_f(c(t)))^{-1}||d(X_0, D).$$

3.4 Physical Consistency

We say that the anchor coordinate system is physically consistent at position X for the destination D if $\overrightarrow{XD'} \cdot \overrightarrow{XD} > 0$, where D' is the apparent destination related to D' in X. Observe that if the anchor coordinate system is physically consistent for the destination D in a ball $\mathcal{B}$ around D, then it is virtually consistent at distance r for the physical destination D, where r is the radius of the biggest multi-dimensional ball Ω such that $\Omega \cap f(\mathbb{R}^2) \subset f(\mathcal{B})$.

To study the physical consistency of the system at position X for the destination D, we split the physical plane in four parts P_1, P_2, P_3, P_4 with $P_1 = \{X'|\overrightarrow{XX'} \cdot \overrightarrow{XD} \leq 0\}$, $P_2 = \{X'|\overrightarrow{XX'} \cdot \overrightarrow{XD} > 0$ and $d(X, X') < d(X, D)\}$, $P_3 = \{X'|\overrightarrow{DX'} \cdot \overrightarrow{DX} > 0$ and $d(X, X') \geq d(X, D)\}$, $P_4 = \{X'|\overrightarrow{DX'} \cdot \overrightarrow{DX} \leq 0\}$. Since the apparent destination D' is defined by

$$D' = X + \sum_i \frac{d(X, A_i) - d(D, A_i)}{d(X, A_i)} \overrightarrow{XA_i}$$

we see as illustrated in Figure 2 that only the anchors in P_2 give a negative contribution to $\overrightarrow{XD'} \cdot \overrightarrow{XD}$.

If anchors are randomly distributed in the network, the negative contribution will most probably be small enough for the system to be consistent, unless P_1 and P_4 are almost void of nodes, which happens when X and D are located on opposite borders of the network (so that all the anchors are between them). This situation did not occur in the experiments we carried out. Nevertheless, physical inconsistency may be avoided by selecting anchors when the destination D of a message originating from X_0 is far away:

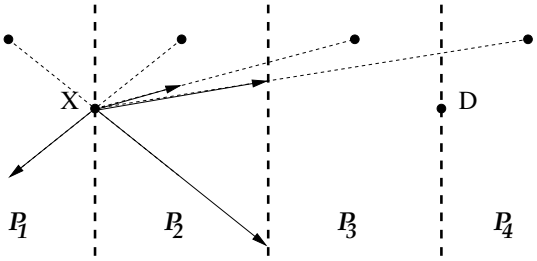


Fig. 2. Contribution of anchors in P_1, P_2, P_3, P_4

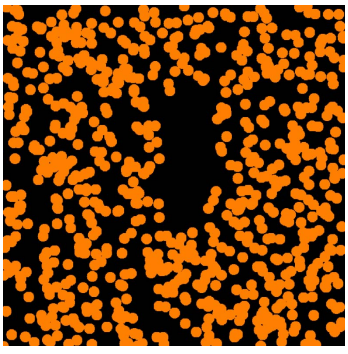


Fig. 3. Experimental settings

1. by default, use all the anchors.
2. compute $l_A = \max_{i \in \{1,..,n\}} \max(d(D, A_i), d(X_0, A_i))$. l_A gives an idea of the diameter of the network.
3. for each node X along the path of the message
   (a) compute $l_X = \max_{i \in \{1,..,n\}} |d(X, D) - d(X, A_i)| = ||f(D) - f(X)||_\infty$. l_X is smaller than $d(X, D)$.
   (b) if using all the anchors and if $l_X > \frac{2l_A}{3}$ then use only the anchors A_i such that $d(D, A_i) < \frac{l_A}{3}$.
   (c) if using a subset of anchors and if $l_X < \frac{l_A}{2}$ then use all the anchors.

In this way, physical inconsistency can be completely avoided in the network, at the cost of using a different coordinate system when $d(X, D)$ is comparable to the diameter of the network.

4 Experiments

4.1 Settings

We implemented CLDP and GFG/GPSR with multiple coordinates on Algo-Sensim [1] to compare the use of virtual coordinates versus real coordinates. To run our simulations, we considered a 15×15 square zone with a rectangular obstacle in the middle (cf Figure 3). We considered a density ranging from 10 to 30 which corresponds to 750-2250 nodes with a circular communication range of 1. We made simulation on one hundred different networks for each settings, over a duration of 1000 steps. At each step, one message (defined by its source and its destination) is generated.

Concerning the coordinates, we made the experiments under two scenarios : without errors and with errors. In each of them we considered three cases: the nodes know their Euclidean coordinates, the nodes know their distances to four anchors positioned at the four corners of the network or the nodes know their distance to six anchors positioned at random in the network.

4.2 Errors on Coordinates

For the Euclidean coordinates, we added an uncorrelated error to both x and y coordinates whose value is uniformly distributed in-between -0.5 and 0.5. This error represents the incertitude of the positioning using devices such as GPS. Hence a node X with exact coordinates (x, y) is considered to have coordinates $(x + b_1, y + b_2)$ where $b_1 \in (-0.5; 0.5)$ and $b_2 \in (-0.5; 0.5)$.

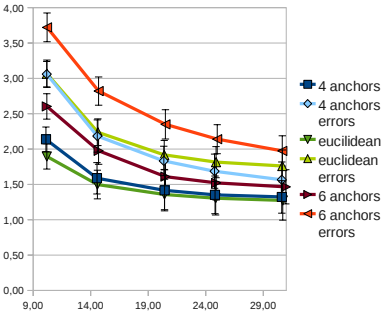
For the virtual coordinates, we added two types of error. To explain them, let us first describe the scenario we consider. We suppose that the nodes estimate their distance to the anchors using signal measurements. The first error represents the node's calibration offset, which is the same whichever signal is measured. To represent this, we chose a multiplicative factor uniformly distributed in-between 0.95 and 1.05. We chose a single value per node and each exact distance to anchors is multiplied by this value. A second error representing signal

distortion is chosen uniformly distributed in-between -0.5 and 0.5 for each coordinate. A value is chosen independently for each coordinate. Hence if a node X has exact distances $(d_i)_1^n$ to anchors $(A_i)_1^n$, we choose $n+1$ random variables, $a \in (0.95; 1,05)$ and $b_i \in (-0.5; 0.5)$ $1 \leq i \leq n$, and the virtual coordinates of X are $(a \cdot d_i + b_i)_1^n$.

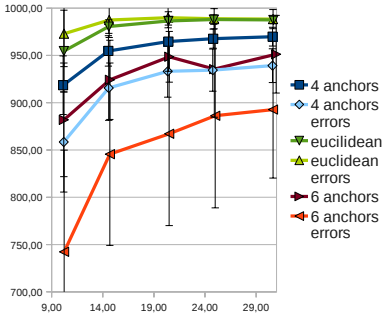
4.3 Experimental Results

We compare the efficiency of using virtual coordinates and Euclidean coordinates. We outline three different experimental results: first, the stretch of computed path (Fig. 4(a), where the stretch is the length of the computed path divided by the length of the shortest path), then the number of times the algorithm CLDP checks each link before the graph is planar (Fig. 4(c)) and finally the number of delivered messages (Fig. 4(b)).

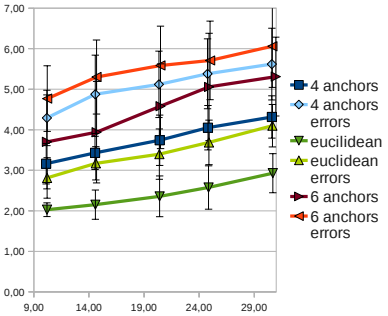
The results of Figure 4 show that the efficiency of using virtual coordinates is the same as the efficiency of using Euclidean coordinates when we use four



(a) Stretch of computed paths depending on density



(b) Delivery rates per thousand messages, depending on density



(c) Average number of times CLDP checks each edge, depending on density

Fig. 4.

anchors placed at the corners. Interestingly, the use of virtual coordinates makes the routing more resilient to errors. The delivery rates are comparable in both settings. When we use six anchors positioned at random in the network, the stretch and the delivery rate are slightly worse. This decrease in efficiency could be explained by the fact that some anchors are positioned in between sources and destinations, thus forcing the message to take a detour (cf Section 3.4, where anchors in P_2 and P_3 penalize the routing). This situation illustrates the trade-off of positioning the anchors at random, which is otherwise a great operational advantage.

5 Conclusion

Geographic routing is an essential component in connecting sensor networks. Foregoing the previously necessary localization phase where physical Cartesian coordinates are produced is an important step into making networks more robust and totally independent from external hardware. Sensor network applications that use localization information exclusively inside the network may transparently use virtual coordinates, whereas sophisticated physical localization may still be performed at some external base station from the virtual coordinates whenever localization must be used externally. In this way, directly using raw distance information without any costly or sophisticated localization calculus is a simple, viable, and efficient way to perform geographic routing.

References

1. AlgoSensim simulator, `http://tcs.unige.ch/code/algosensim/overview`
2. Barriere, L., Fraignaud, P., Narayanan, L.: Robust position based routing in wireless ad hoc networks with unstable transmission ranges. In: Proc. DialM, pp. 19–27 (2001)
3. Barsi, F., Bertossi, A., Sorbelli, F.B., Ciotti, R., Olariu, S., Pinotti, M.: Asynchronous Training in Wireless Sensor Networks. In: Kutyłowski, M., Cichoń, J., Kubiak, P. (eds.) ALGOSENSORS 2007. LNCS, vol. 4837, pp. 46–57. Springer, Heidelberg (2007)
4. Barsi, F., Navarra, A., Pinotti, M., Lavault, C., Ravelomanana, V., Olariu, S., Bertossi, A.: Efficient binary schemes for training heterogeneous sensor and actor networks. In: Proc. HeterSanet 2008, pp. 17–24 (2008)
5. Bose, P., Morin, P., Stojmenovic, I., Urrutia, J.: Routing with guaranteed delivery in ad hoc wireless networks. Wireless Networks 7(6), 609–616 (2001)
6. Bruck, J., Gao, J., Jiang, A.: Localization and routing in sensor networks by local angle information. In: Proc. MOBIHOC 2005, pp. 181–192 (2005)
7. Caruso, A., Chessa, S., De, S., Urpi, A.: GPS free coordinate assignment and routing in wireless sensor networks. In: Proc. INFOCOM 2005, pp. 150–160 (2005)
8. Farrell, J., Conkey, E.M., Stephens, C.: Send measurements, not coordinates. Navigation 46(3), 203–215 (1999)
9. Hotz, S.: Routing information organization to support scalable interdomain routing with heterogeneous path requirements PhD Thesis. University of Southern California, Los Angeles, CA (1996)

10. Huc, F., Jarry, A.: Vrac: Virtual routing with raw anchor coordinates in sensor networks. In: Proc. WONS 2010, pp. 106–112 (2010)
11. Huc, F., Jarry, A., Leone, P., Moraru, L., Nikoletseas, S., Rolim, J.: Early obstacle detection and avoidance for all to all traffic pattern in wireless sensor networks. In: Dolev, S. (ed.) ALGOSENSORS 2009. LNCS, vol. 5804, pp. 102–115. Springer, Heidelberg (2009)
12. Karp, B., Kung, H.: GPSR: greedy perimeter stateless routing for wireless networks. In: Proc. MOBICOM 2000, pp. 243–254 (2000)
13. Kim, Y., Govindan, R., Karp, B., Shenker, S.: Geographic routing made practical. In: Proc. NSDI 2005, pp. 217–230 (2005)
14. Kleinberg, J., Slivkins, A., Wexler, T.: Triangulation and embedding using small sets of beacons. In: Proc. FOCS 2004, pp. 444–453 (2004)
15. Kuhn, F., Wattenhofer, R., Zollinger, A.: An algorithmic approach to geographic routing in ad hoc and sensor networks. IEEE/ACM Transactions on Networking 16(1), 51–62 (2008)
16. Langendoen, K., Reijers, N.: Distributed localization in wireless sensor networks: a quantitative comparison. Computer Networks 43(4), 499–518 (2003)
17. Maroti, M., Völgyesi, P., Dora, S., Kusy, B., Nadas, A., Ledeczi, A., Balogh, G., Molnar, K.: Radio interferometric geolocation. In: Proc. SenSys 2005, pp. 1–12 (2005)
18. Moore, D., Leonard, J., Rus, D., Teller, S.: Robust distributed network localization with noisy range measurements. In: Proc. SENSYS 2004, pp. 50–61 (2004)
19. Moraru, L., Leone, P., Nikoletseas, S., Rolim, J.: Geographic Routing with Early Obstacles Detection and Avoidance in Dense Wireless Sensor Networks. In: Proc. AdHocNets 2008, pp. 148–161 (2008)
20. Ng, T., Zhang, H.: Predicting Internet network distance with coordinates-based approaches. In: Proc. INFOCOM 2002, pp. 170–179 (2002)
21. Niculescu, D.: Positioning in ad hoc sensor networks. IEEE Network, 24–29 (2004)
22. Powell, O., Nikoletseas, S.: Simple and efficient geographic routing around obstacles for wireless sensor networks. In: Demetrescu, C. (ed.) WEA 2007. LNCS, vol. 4525, pp. 161–174. Springer, Heidelberg (2007)
23. Priyantha, N.B., Balakrishnan, H., Demaine, E.D., Teller, S.: Mobile-assisted localization in wireless sensor networks. In: Proc. INFOCOM 2005, Miami, Florida, pp. 172–183 (2005)
24. Rao, A., Papadimitriou, C.H., Shenker, S., Stoica, I.: Geographic routing without location information. In: Proc. MOBICOM 2003, pp. 96–108 (2003)
25. Saad, C., Benslimane, A., König, J.: AT-DIST: A Distributed Method for Localization with high accuracy in Sensor Networks. Technical Report lirmm-00270283, lirmm (2008)
26. Stefano, G.D., Petricola, A.: A distributed AOA based localization algorithm for wireless sensor networks. Journal of Computers 3(4), 1–8 (2008)