A New Positioning Scheme for Pen-Like Handwriting Input Devices

Xu Jin, Okada Kanzo and Sun Xiao Bing
Singapore Research Laboratory
Sony Electronics Singapore Pte Ltd
Singapore
Jin.Xu@ap.sony.com

Abstract—This paper introduces a new positioning scheme for pen-like handwriting input devices. With the use of passive references, positioning can be done in a more standalone manner with lower system complexity. In this scheme, the system is made up of a pen-like device with a single omni-directional wireless transceiver located near the tip and a writing tablet with several reflectors embedded at its edges as references. The omni-directional transceiver keeps transmitting short wireless pulses and receiving echoes from the passive references. Using the proposed simple self-localization method, the 2D position of the pen-like device can be obtained solely from the received echoes that contain no directional information. Simulation results using ultrasound as the wireless medium have shown that it is capable of recognizing handwritten input characters accurately.

Keywords—pen-like handwriting input device; passive reference; self-localization; omni-directional transceiver

I. INTRODUCTION

As human beings have long used pen and paper in the history, handwriting becomes a natural way to store and transfer information. Pen-like devices that can digitize the user’s handwriting trajectory have attracted much attention as it provides a more user-friendly man/machine interface, especially for mobile applications. Advances in sensors and miniaturized processors enabled many technologies to implement this function [1]. Table I gives a comparison of standaloneness (everything done within the pen-like device) and writing mode (whether multiple discontinuous strokes or only piecewise continuous stroke is supported) for three typical products available in the market.

The proposed scheme aims at features similar to product B, but without constraint of the specially designed paper. It is a two unit system made up of a pen-like device and a tablet as shown in Fig. 1. It is different from normal tablet-based methods in that there are no sensors embedded in the tablet and positioning is performed solely in the pen-like device. In the tablet, only several passive references (small-sized reflectors) are embedded to serve as “landmarks” when the pen-like device is moving on it.

With this configuration, the positioning problem is similar to the self-localization problem encountered in the field of robotics. But the handwriting input application requires for a simple method to be implemented within the practical size of a pen. The proposed positioning method and system provide such a solution. Instead of using multiple directional sensors or a bulky rotation mechanism commonly adopted in robots, a single omni-directional transceiver (e.g. for ultrasound, RF or infrared, etc.) is used. The transceiver keeps transmitting short pulses and receiving the echoes from the references to monitor its own movement. As an omni-directional transceiver is used, the received echoes form a time series gathered from all directions. Associating the references with the corresponding range data thus becomes a major problem to deal with. But our findings on the geometric constraints posed by no less than three references can solve this problem. A simple positioning method based on this approach is proposed in this paper, as shown in Fig.2.

![Figure 1. Illustration of the proposed system](image-url)
II. A SIMPLE SELF-LOCALIZATION METHOD

This section formulates the problem and introduces the proposed positioning method. The methodology is similar to self-localization technique in robotics and it provides a very simple solution with the use of a single omni-directional transceiver and several passive references.

A. Problem Statement

Self-localization requires the system to determine its own movement by monitoring the sensor data. It is a hot topic in robotics field and many methods are available [2], [3], [4]. The problem in our positioning scheme falls right into this category. But different from robotics applications that normally involve both translation and rotation, we are only interested in the translational movement.

For sensor data collection, a panoramic view is advantageous as it can provide information all around the robot. A typical solution is to use a ring of directional sensors (e.g. ultrasound in [2]) or a rotation mechanism of a single directional sensor (e.g. laser range finder in [3]). A single omni-directional sensor is normally avoided, not only for its lack of direction/orientation information, which is important to robot’s navigation problem, but also due to the introduction of severe multi-path problem. Use of omni-directional cameras in robot navigation exists [5], but their principle based on optical flow requires heavy computation.

With sensor data collected, they need to be correctly associated with corresponding objects in a static reference model, simple or complex, pre-known or progressively-learned, before they can be used in position calculation. In some cases, this association process alone solves the positioning problem. The simplest way is to use a set of active references (e.g. transponders) whose identity can be recognized via wireless communication (e.g. a tag ID encoded in the wireless signal). Thus the range data can be easily associated with their pre-known positions and then triangulation to calculate position is trivial. But this requires costly deployment and maintenance of the active references. Another way is to use the static/quasi-static surroundings as the reference and to find the most probable “matching” by use of geometric constraints (either explicitly [3], [4] or implicitly [2]) based on pre-known world model, such as a floor plan. For robotics application, this is normally computationally demanding and usually requires the aids from gyro-sensors and odometers onboard the robots.

The proposed scheme is a unique and simple implementation. Use of a single omni-directional sensor enables a panoramic view with the minimum hardware complexity and passive references are cheaper and easier to deploy and maintain compared to active references. However, with this combination, echoes from all the references mix up in the received signal and the detected range data will be in ascending order (these ordered range data are referred to as a “distance vector” in the rest of the paper). No direction/orientation information is available to build up the correspondence between each range value and its corresponding reference (referred to as “correspondence” in the rest of the paper).

In a setup as shown in the left side of Fig.3, assuming that all the range data can be recovered precisely in the pre-processing step, if the locations of the \( N \) passive references are pre-known values of \( X_i = (x_i, y_i) \) \((i = A, B, \ldots)\) and the distance vector detected from the echo signal is \((d_1, d_2, \ldots, d_N)\) with \( d_1 \leq d_2 \leq \ldots \leq d_N \), the problem can be posed as follows: is it possible to uniquely and precisely determine the transceiver position denoted by \( P(x_P, y_P) \)?

B. Principle of the Positioning Method

The finding in the Appendix forms the basis for the proposed method as it assures the existence of a solution under certain circumstances. With block diagram shown in Fig.2, a practical implementation of the proposed method for the case of \( N = 3 \) is introduced in the following part. This setup is chosen to minimize the complexity of system and the algorithms. Although there are “ambiguity points” shown in the Appendix to exist on the loci of hyperbolic curves, it will be shown later in Sec. IV that they can be effectively filtered out by monitoring the movement continuously.

The core of this positioning method is the restoration of “correspondence”, after which the calculation of position is straightforward. If \( N = 3 \), there are six possible correspondences for a distance vector as listed in Table II with the illustration in the right side of Fig.3, where \( O \) is the circumcenter of the triangle and the sections are divided by bisectors of each side. Thus the problem is reduced to a classification problem. A classifying criterion needs to be determined to find out the correct correspondence from the six possible ones solely based on the input of a distance vector itself. The classifying criterion we found turns out to be a fourth-order polynomial as explained afterwards.

![Figure 2. Block diagram of the self-localization method](image)

![Figure 3. Illustration of the geometry](image)
and 2. \( C_y \) is also depending on the index \( B = \delta_d \). \( d_P \) and \( X \) are the \( d_P \times C_2 \) and \( X \) is currently in. Its \( d - d_P \), \( X \) is pre-stored possible values of \( \delta_d \), and the according values are shown in Table II.

Equation (2) can be expressed in terms of \( (d_1^2 - d_2^2) \) and \( (d_3^2 - d_4^2) \). Substituting it into (1), a constraint on the three elements of the distance vector is obtained as:

\[
\begin{align*}
(d_1^2 - d_2^2) & = T_{2:0:3} \left[ \begin{array}{c} x_p \\ y_p \\ 1 \end{array} \right] (X = 1, 2, ..., 6) \\
(d_3^2 - d_4^2) & = T_{2:0:3} \left[ \begin{array}{c} x_p \\ y_p \\ 1 \end{array} \right] (X = 1, 2, ..., 6)
\end{align*}
\]

where the matrix \( T_{2:0:3} \) is also depending on the index \( X \) and the according values are shown in Table II.

With (2), \( (x_p, y_p) \) can be expressed in terms of \( (d_1^2 - d_2^2) \) and \( (d_3^2 - d_4^2) \). Substituting it into (1), a constraint on the three elements of the distance vector is obtained as:

\[
\begin{align*}
d_1^2 & = B_{1:0:3} \left[ \begin{array}{c} (d_1^2 - d_2^2)^2 \\ (d_3^2 - d_4^2)^2 \\ (d_3^2 - d_4^2)(d_1^2 - d_2^2) \\ (d_1^2 - d_2^2) \\ (d_3^2 - d_4^2) \\ 1 \end{array} \right] \\
\end{align*}
\]

where \( B_{1:0:3} \) is fully determined by the coordinates of the three references and the region \( (x_p, y_p) \) is currently in. Its six possible values can be calculated based on above deduction and pre-stored in the memory of the pen.

Equation (3) can serve as a way to classify an input distance vector into one of the six regions by choosing the index \( X \) that gives the minimum value of (4), with the six pre-stored possible values of \( B_{1:0:3} \). After classification, (2) can then be used to calculate the 2D coordinate.

\[
\begin{align*}
abs(d_1^2 - B_{1:0:3}) & = \left\{ \begin{array}{l} (d_1^2 - d_2^2)^2 \\ (d_3^2 - d_4^2)^2 \\ (d_3^2 - d_4^2)(d_1^2 - d_2^2) \\ (d_1^2 - d_2^2) \\ (d_3^2 - d_4^2) \\ 1 \end{array} \right\} \\
\end{align*}
\]

III. PERFORMANCA ANALYSIS

In this section, the positioning performance will be analyzed in terms of several factors. The performance is described in terms of the root-mean-square error (RMSE). Due to the principle of our method, the positioning errors can be divided into two groups based on the correctness of the classifying step. Theoretically, (4) generate perfect classifying results except for the “ambiguous points”. However, in real situations the distance vector cannot be obtained without errors and this may lead to wrong classifications at other locations as well. Base on this dividing method, RMSE can be expressed as:

\[
\begin{align*}
\sqrt{\frac{\sum \int (P(g(x,y)) \delta_d \Delta \Omega(x,y) f(g(x,y)))^2}{\text{Total area}}} \\
\end{align*}
\]
classification may happen at positions more than the loci discussed in the Appendix, because detected distance vectors are imperfect. For example, it is easy to reason that a configuration very “close” to being symmetric will cause more positioning errors than a configuration “far” from being symmetric. Simulations have been done for choosing the optimum geometry in a trying and comparing manner. However, rigorous theoretical analysis is not covered in this paper. The measurement of symmetry in [7] might be a starting point for further studies.

B. Ranging Accuracy

The ranging errors are modeled as additional-white-Gaussian-noise (AWGN) and the measured distance vector \((\vec{d}_1, \vec{d}_2, \vec{d}_3)\) can be described as (with perfect resolution assumed):

\[
\begin{bmatrix}
\vec{d}_1 \\
\vec{d}_2 \\
\vec{d}_3
\end{bmatrix} =
\begin{bmatrix}
d_1 \\
d_2 \\
d_3
\end{bmatrix} +
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\delta_3
\end{bmatrix}
\]

where \((d_1, d_2, d_3)\) is the exact value of distance vector, \(\delta_1, \delta_2\) and \(\delta_3\) are three independent AWGN random variables with zero-mean and same standard deviation (determined by the achievable ranging accuracy). Then the error in positioning results caused by ranging errors can be calculated based on (2).

If the classifying result is correct, the error (with higher-order terms of \(\delta_1, \delta_2\) and \(\delta_3\) ignored) is:

\[
\begin{bmatrix}
\delta x \\
\delta y \\
\delta z
\end{bmatrix} =
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
2d_1 - 2d_2 \\
2d_1 - 2d_3
\end{bmatrix} +
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\delta_3
\end{bmatrix}
\]

(7)

where \(T_{ij}\) is the element at \(i-th\) row and \(j-th\) column of the matrix \(T\) (refer to Table II for its six possible values).

When the classifying result is incorrect, wrong transformation will be applied in position calculation. If instead of \(T_{21,1}, \ T_{22,1}\) is applied in the calculation, the error for position estimation in this case can be calculated as:

\[
\begin{bmatrix}
\delta x \\
\delta y \\
\delta z
\end{bmatrix} =
\begin{bmatrix}
T_{11,1} & T_{12,1} \\
T_{21,1} & T_{22,1}
\end{bmatrix}^{-1}
\begin{bmatrix}
T_{11,1} & T_{12,1} \\
T_{21,1} & T_{22,1}
\end{bmatrix}^{-1}
\begin{bmatrix}
-2d_1 + 2d_2 \\
-2d_1 + 2d_3
\end{bmatrix} +
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\delta_3
\end{bmatrix}
\]

(8)

Equations (7) and (8) are quite different. Equation (7) describes an un-biased linear transformation on an AWGN vector. But in (8), the errors are biased by the first term which is determined purely by the geometry and the second term which depends on both the geometry and the current position. Combined with formulas of functions \(P_1\) and \(P_2\), (7) and (8) can be used to predict the performance using (5).

C. Resolution Limit

Due to the resolution limit, some of the echo components cannot be detected due to overlapping effect shown in Fig.4. As a result, the according range data can only be “filled-in” with the value of the detectable echo component right before it. It is assumed that the echo component that “overshadows” it can be correctly determined. For example, with a waveform in Fig.4, it can be told (e.g., from the fact that \(W_1 > W_2\) ) that the “missing” range value should be “filled-in” with \(d_i\) but not with \(d_j\).

With this assumption, there are three different cases: a) the first echo overshadows the second, b) the second echo overshadows the third and c) the first echo overshadows both the second and third. In these cases, the detected distance vectors are “filled-in” as \((\vec{d}_1, \vec{d}_3, \vec{d}_2)\), \((\vec{d}_2, \vec{d}_1, \vec{d}_3)\) and \((\vec{d}_3, \vec{d}_2, \vec{d}_1)\) respectively, where \(\vec{d}_j\) \((j=1,2)\) is the measured value of a component in the distance vector. The relationship between them and the true distance vectors can be accordingly modeled respectively as (with perfect ranging assumed):

\[
d_1 = \vec{d}_1, \ d_2 = \vec{d}_2 + \delta_\lambda\text{ and } d_3 = \vec{d}_3
\]

\[
d_1 = \vec{d}_1, \ d_2 = \vec{d}_2 + \delta_\lambda\text{ and } d_3 = \vec{d}_3 + \delta_\lambda + \delta_\lambda'
\]

where \(\delta_\lambda\) and \(\delta_\lambda'\) are random variables with uniform distribution probability over \([0, \Delta]\) and \([-\Delta, \Delta - \Delta]\).

If the classifying result is correct, the positioning error is only caused by substituting the “hidden” echo component with the one that “overshadows” it. From (2), it can be calculated as (9), (10) and (11), for the above three cases respectively.

\[
\begin{bmatrix}
\delta x \\
\delta y \\
\delta z
\end{bmatrix} =
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
2d_1 - 2d_2 \\
-2d_1 + 2d_3
\end{bmatrix} +
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\delta_3
\end{bmatrix}
\]

(9)

\[
\begin{bmatrix}
\delta x \\
\delta y \\
\delta z
\end{bmatrix} =
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
2d_1 - 2d_2 \\
2d_1 - 2d_3
\end{bmatrix} +
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\delta_3
\end{bmatrix}
\]

(10)

Figure 4. Illustration of the “overlapping” problem.
\[
\begin{bmatrix}
\delta_x \\
\delta_y \\
\delta_z
\end{bmatrix} =
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}
\begin{bmatrix}
d_1 \\
d_2 \\
de_3
\end{bmatrix}
\]

If the classification is wrong, instead of \( T_{2x3} \), \( T_{2x3} \) is applied in the calculation. The positioning errors can be calculated as:

\[
\begin{bmatrix}
\delta_x \\
\delta_y \\
\delta_z
\end{bmatrix} =
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
d_1 \\
d_2 \\
de_3
\end{bmatrix}
\]

\[
+ T_{11} T_{12} T_{21} T_{22} \begin{bmatrix} 2d_1 \\ 2d_1 \\ 2d_1 \end{bmatrix}
\]

(12)

\[
\begin{bmatrix}
\delta_x \\
\delta_y \\
\delta_z
\end{bmatrix} =
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
d_1 \\
d_2 \\
de_3
\end{bmatrix}
\]

\[
+ T_{11} T_{12} T_{21} T_{22} \begin{bmatrix} 2d_1 \\ 2d_1 \\ 2d_1 \end{bmatrix}
\]

(13)

\[
\begin{bmatrix}
\delta_x \\
\delta_y \\
\delta_z
\end{bmatrix} =
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}
\begin{bmatrix}
T_{11} & T_{12} \\
T_{21} & T_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
d_1 \\
d_2 \\
de_3
\end{bmatrix}
\]

\[
+ T_{11} T_{12} T_{21} T_{22} \begin{bmatrix} 2d_1 \\ 2d_1 \\ 2d_1 \end{bmatrix}
\]

(14)

These results are similar to those of ranging errors. The two factors discussed above co-exist in real situations and their effects are actually superimposed with each other. However, from the above results, we are able to predict that the final errors will be clustered into two groups: unbiased small errors (when the classifying is correct) and biased big errors (when the classifying is wrong).

IV. SIMULATION RESULT

This section demonstrates the simulation results of the proposed scheme based on a Monte-Carlo method. In all the simulations, ultrasound is employed as the medium for its high ranging accuracy (a value of 0.5mm is commonly reported and used in the simulation). Since the wavelength of ultrasound is much shorter than that of normal RF signals, it can also provide higher ranging resolution. In our simulation the resolution limit is set to 0.85cm, which corresponds to the wavelength of 40 kHz ultrasound (a higher frequency may produce even better resolution). The writing tablet is assumed to be of an A4-size with passive references located at three corners.

The first simulation was done as follows. At a certain position, the distances to the three references are first calculated based on their known positions. These values were perturbed by an AWGN noise vector (with a standard deviation of 0.5mm) to simulate the effect of imperfect ranging. Then the overlapping effect was simulated with the assumption introduced in Sec. III. Lastly, the resulting distance vector is input into the positioning algorithm to generate an estimation of the position. The output value was compared to the true value to calculate the positioning error. This process was performed repeatedly on thousands of points randomly selected on the writing tablet and the statistical result (histogram and cumulative distribution probability - CDP) is shown in Fig. 5. The result supports the findings in Sec. III: except for a small number (around 3% in this simulation) of sporadic big errors, the positioning is quite accurate with RMSE=0.6mm for errors up to 97% CDP. This fact provides the possibility of filtering out the big errors by detecting abnormal output during a continuous movement. For example, assuming a sampling frequency of 200Hz and writing speed limit of 1m/s yields a maximum possible movement of 0.5cm within a sampling period. Any output that is more than 0.5cm away from the previous sampling can be regarded as wrong and should be removed.

The second simulation was to directly emulate the application of handwriting input with a simplest filtering: if the current output is detected as a result of wrong classification, it is replaced by the previous one and the positioning continues onto next sampling. In the left of Fig.6, real input was obtained by capturing the movement of a mouse and connecting them with straight lines. In the right side of Fig.6, the real-time positioning result is illustrated by joining the “filtered” output with straight lines. Three red circles denote the passive references.

![Figure 5. Histogram and cumulative distribution probability of errors](image)

![Figure 6. Simulation result with handwriting input](image)
V. CONCLUSIONS AND FUTURE WORK

A passive-reference based self-localization method has been described in details in this paper. This method is proposed for a new positioning scheme of pen-like devices. Simulation results show that with current ultrasonic sensor and electrical circuit technologies, sub-mm level accuracy can be achieved, which is adequate for handwriting input applications.

The proposed method is unique in the use of a single omni-directional transceiver in self-localization. It has been shown that it is possible to perform positioning without any directional information in the received signal by utilizing the geometric constraints within the range data. This method can be readily applied to other fields like robotics.

Future works include more rigorous theoretical study on the performance analysis and optimization of the method as well as the verification of the method using a working prototype.

ACKNOWLEDGMENT

The authors would like to thank their colleagues Dr. Lu Wei, Dr. Gao Peng and Dr. Ma Yu Gang for enlightening discussions.

REFERENCES


APPENDIX: ONE-TO-ONE PROPERTY OF MAPPING BETWEEN A DISTANCE VECTOR AND A 2D POSITION

Given three reference points not lying on a line with unequal distances among them, which is equivalent to saying that the triangle formed by joining them with straight lines is not isosceles triangle, and a point within the plane of the triangle (not necessarily within the triangle), the distance vector is defined by ordering the distances from the point to the three reference points in the ascending manner. It will be discussed below whether the mapping between the position of the point and the distance vector is one-to-one, with the extension of the result to cases with more than three references.

We call the three reference points $A$, $B$ and $C$ and denote the distance from the given point to each of them as $d_A$, $d_B$ and $d_C$ respectively. Assuming without loss of generality that there are two points $P(x_1,y_1)$ and $Q(x_2,y_2)$ giving the same distance vector, there exist only three possibilities:

- \[
\begin{aligned}
&d_A = d_B, \\
&d_A = d_C, \\
&d_B = d_C,
\end{aligned}
\]

where the subscript 1 or 2 indicates which point, $P$ or $Q$, is used to measured the distance to each reference point.

For cases a) and b), it is easy to show that $P$ and $Q$ have to be identical. Due to the limited space of the paper, the proof is omitted here. For case c), writing the distances in terms of 2D coordinates, we have:

\[
\begin{aligned}
&d_A = d_B, \\
&d_A = d_C, \\
&d_B = d_C.
\end{aligned}
\]

Subtracting each of the lower two equations from the top one in (A.1), we have:

\[
\begin{aligned}
&x_1 - x_2 = (x_2 - x_1)^2 + (y_2 - y_1)^2, \\
&y_1 - y_2 = (x_2 - x_1)^2 + (y_2 - y_1)^2.
\end{aligned}
\]

Equation (A.2) shows that there is a linear transformation between the coordinates of $(x_1, y_1)$ and $(x_2, y_2)$, thus $P$ and $Q$ can be expressed as a linear transformation of $(x_1, y_1)$ and the resulting expressions are substituted into (A.1) to obtain a second-order equation in the coordinates of $P$. It is easy to show that $P$ must lie on a hyperbola with the parameters determined by the coordinates of the three points $A$, $B$ and $C$. Accordingly, $Q$ lies on another hyperbola, as a linear transformation of the locus of $P$.

Therefore, except for two pairs of hyperbola loci (since there is another equivalent case of:

\[
\begin{aligned}
&d_A = d_C, \\
&d_B = d_B, \\
&d_C = d_B,
\end{aligned}
\]

is actually one-to-one.

With four references, it can be similarly deduced that there are at most two pairs of single points with equivalent distance vector. And with no less than five reference points, the mapping between 2D coordinates of a point and the distance vector is one-to-one over the whole plane. Of course, in all these cases, there is an assumption that the polygon formed by joining the reference points is not symmetric and without any parallel sides.