Rail vehicle localization exploiting rail track georeferenced coordinates

António João Ferreira[1], José Miguel Almeida[2], Eduardo Silva

INESC TEC – INESC Technology and Science (formerly INESC Porto) and ISEP/IPP – School of Engineering, Polytechnic Institute of Porto [1](aferreira@lsa.isep.ipp.pt), [2](jma@lsa.isep.ipp.pt)

Abstract

A novel dead reckoning algorithm conceived for localization of small inspection rail vehicles in Global Navigation Satellite System (GNSS) denied environments is presented. This work focus on simplifying the rail vehicle localization task, taking into account restrictions on movement imposed by the railroad tracks. Considering that dead reckoning techniques accumulate errors over time, leading to increasing global uncertainty, a method was designed to correct the estimates and also smooth trajectory errors backwards in time, through visualization of global landmarks. Results show the effectiveness of this approach in reducing long-term position errors. The current document reports real railroad experiments, featuring a specially designed non-motorized mobile modeling vehicle.

Subject Headings. Robotics, Data Processing, Railway Track.
Author Keywords. Sensor Fusion, Autonomous Localization, Rail Vehicle, Modeling, GNSS-denied.

1. Introduction

Rail vehicles may travel along several kilometers of tunnels and canyons without being able to receive GNSS signal. Nowadays long-term precise localization in GNSS-denied environments is still a challenging problem, and even more difficult to achieve through low-cost sensors. In rail inspection systems (Glaus 2006) total stations are usually employed to track the surveying vehicle. This is not a versatile solution since the total stations have to be moved constantly to maintain line-of-sight with the vehicle. By this reason, for long traveling distances, on-board localization systems would be much more convenient.

Over the last few years, Simultaneous Localization and Mapping (SLAM) became an effective solution to the robotic exploration problem. Some interesting methods where evaluated in extreme scenarios, including mines (Nüchter et al. 2004; Thrun et al. 2004), however the obligation to close loops in order to reduce the global uncertainty remains a huge limitation. Loop detection is usually a hard task and in some situations the system may not be able to revisit previously explored areas, due to logistic restrictions or, in the particular case of rail vehicles, due to the absence of loops in the tracks. Furthermore, smooth and uniform geometry of the surrounding scene, especially inside tunnels, may lead to sensor ambiguity, which makes the localization process even harder.

Since rail vehicles follow the fixed course enforced by the railroad, whenever a detailed description of the tracks is available, in the form of global coordinates, a model of the path can be instantiated and used to help solving the localization problem. In our case the railroad model relates two different domains, namely the path length along the rail and the vehicle localization, defined in the same global coordinate system used for the railroad

model. This capability of translating motion along the tracks into 6 degrees-of-freedom localization and vice-versa is a major breakthrough towards a simplified, flexible and real-time algorithm for localization of rail vehicles.

After proper initialization, the distance traveled over the rails is computed through integration of displacement measurements retrieved from rotary encoders attached to the wheels. Note that only one dimension is subjected to this dead reckoning algorithm – the displacement along the track. During motion all computations can be done exclusively in the railroad domain, but at any instant the railroad model can be invoked to jump to the meaningful three-dimensional domain.

Monotonically increasing uncertainty is an intrinsic feature of dead reckoning solutions, yet unbounded uncertainty is not admissible in mobile modeling applications. In order to keep uncertainty at reasonable levels, known laser targets are captured by a LIDAR and used to correct the global error. Moreover, in a post processing stage all target observations contribute to smooth the whole trajectory and so achieve global consistency.

This document is organized as follows: Section 2 describes the railway modeling process. Section 3 is devoted to the explanation of the localization algorithm. Results are presented and discussed in section 4 and finally, section 5 provides a conclusion and sets some future goals.

2. Railway Modeling

In railways design, the route of a given track is referred by the term alignment. The horizontal alignment defines the track shape in the XY plane, while the vertical alignment consists on the elevation along the Z component. Since trains have no control over lateral movement the alignment restrictions are of extreme importance to ensure both security and comfort. The overall alignment results from the combination of three basic shapes: straight lines, circular arcs and transition curves. The Euler spiral – clothoid – is the most common shape of transition curve. The curvature of a clothoid changes linearly with his curve length, resulting in a constant value of centripetal acceleration applied to a vehicle moving at constant velocity.

Usually, transition curves provide a smooth change between curved and straight segments in the horizontal alignment. For the vertical alignment only straight segments and transition curves are used. Nevertheless, in either case, the resulting track shape is smooth from the mathematical point of view, i.e., the piece-wise function representing the global track configuration has continuous derivatives. Splines preserve the same properties, which makes the method ideal to construct the railroad model.
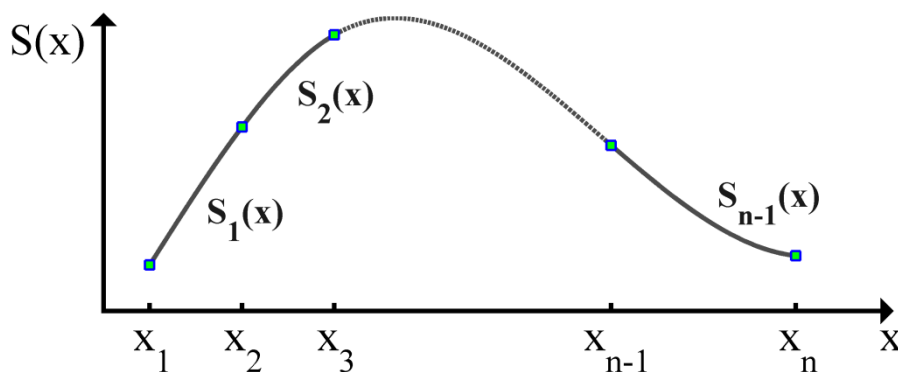

Figure 1: Interpolation of n points using cubic splines

2.1. Spline Interpolation Background

The rail model is obtained by interpolation of georeferenced coordinates using cubic splines. For the two-dimensional case, assume a set of n control points, $\vec{p} = \{\vec{p}_1, \vec{p}_2, \dots, \vec{p}_n\} = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, with $x_1 < x_2 < \cdots < x_n$. Each sub-interval $[x_i, x_{i+1}]$, with $1 \leq i \leq n-1$, is represented by a distinct third-order polynomial $s_i(x)$ (See Figure 1), so the interpolation function is a collection of $n-1$ polynomials expressed in a piece-wise manner as:

$$S(x) = \begin{cases} S_1(x) : & x \in [x_1, x_2] \\ S_2(x) : & x \in [x_2, x_3] \\ S_3(x) : & x \in [x_3, x_4] \\ \qquad \vdots \\ S_{n-1}(x) : & x \in [x_{n-1}, x_n] \end{cases} \tag{1}$$

Each cubic polynomial piece $S_i(x)$ has 4 unknown coefficients - $\{a_i, b_i, c_i, d_i\}$:

$$S_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \tag{2}$$

The goal is to find the $4(n-1)$ unknowns so that polynomials join at the control points as smoothly as possible. Therefore, the following restrictions are applied so that $S'(x)$, $S'(x)$ and $S''(x)$ preserve continuity (Burden et al. 2005):

$$S_i(x_i) = y_i : \ i = 1, 2, \dots, n-1 \tag{3a}$$

$$S_i(x_{i+1}) = y_{i+1} : \ i = 1, 2, \dots, n-1 \tag{3b}$$

$$S_i'(x_{i+1}) = S_{i+1}'(x_i + 1) : \ i = 1, 2, \dots, n-1 \tag{3c}$$

$$S_i''(x_{i+1}) = S_{i+1}''(x_i + 1) : \ i = 1, 2, \dots, n-1 \tag{3d}$$

From the above constraints $4(n-1) - 2$ equations can be instantiated. The system is still under-determined, hence two more constraints are needed to reach as many equations as the total number of unknowns. There are several ways to select the two additional constraints (Bartels et al. 1987), the one used here is the not-a-knot condition. It stipulates continuity for the third derivative at the points x_2 and x_{n-1}:

$$S_1'''(x_2) = S_2'''(x_2) \tag{3e}$$

$$S_{n-2}'''(x_{n-1}) = S_{n-1}'''(x_{n-1}) \tag{3f}$$

Finally, the spline coefficients are computed by simply solving the system of linear equations defined by all previous constraints (equations 3).

2.2. Expanding Spline Interpolation to the 3D Space

One way of performing cubic spline interpolation of 3D points is to decouple each component and solve the problem independently for each dimension, ending up with three piece-wise polynomials – $S^x(\lambda)$, $S^y(\lambda)$ and $S^z(\lambda)$ (See Figure 2).

$$S^{xyz}(\lambda) = \begin{bmatrix} S^x(\lambda) \\ S^y(\lambda) \\ S^z(\lambda) \end{bmatrix} \tag{4}$$

For simplicity reasons a common domain λ must be used in all spline computations. Remember, from section 2.1, that domain points have to be monotonically increasing, so this must be taken into consideration when choosing the calculation method for λ. Our solution is to define λ to be the cumulative sum of Euclidean distances between consecutive points:
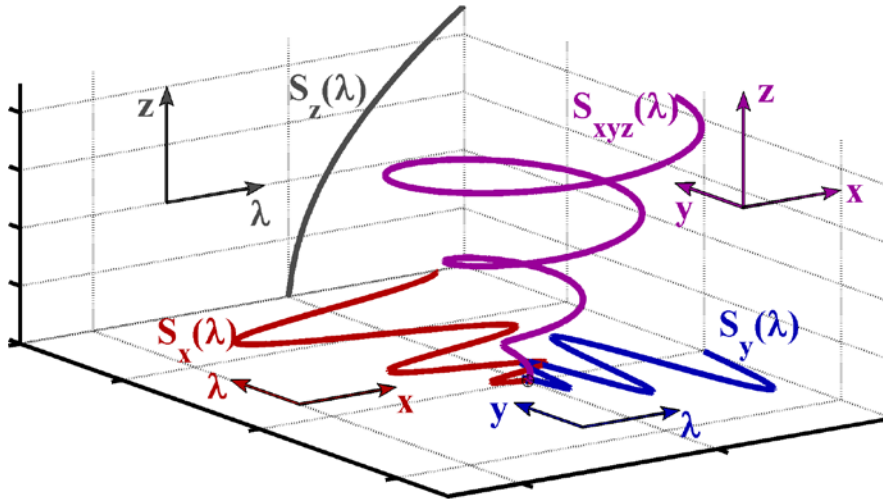
Figure 2: Together, the three cubic splines $S^x(\lambda)$, $S^y(\lambda)$ and $S^z(\lambda)$, defined in the same domain λ, encode the complexity of a three-dimensional spiral $S^{xyz}(\lambda)$

$$\lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} 0 \\ \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \\ \lambda_2 + \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2 + (z_3 - z_2)^2} \\ \vdots \\ \lambda_{n-1} + \sqrt{(x_n - x_{n-1})^2 + (y_n - y_{n-1})^2 + (z_n - z_{n-1})^2} \end{bmatrix} \quad (5)$$

This is a practical solution since λ can be computed directly from the initial set of points, no extra information is needed. Moreover, for a set of non-repeated points, a monotonically increasing result will always be achieved.

3. Localization Method

The localization algorithm operates in two different domains, particularly:

- The railroad domain – One state corresponding to the vehicle displacement along the rail tracks: $ds(t)$, plus two additional parameters λ^v and spline segment index, used to relate the displacement with the railroad model;
- The global domain – Position $\{x(t), y(t), z(t)\}$ and orientation $\{\omega_x(t), \omega_y(t), \omega_z(t)\}$ defined in the ECEF (Earth-Centered Earth-Fixed) Cartesian coordinate system.

Figure 3 illustrates the high level concept for the purposed localization approach. After building the railroad model, as described in the previous section, the vehicle's initial position is projected to the closest point in the splines model, to specify the localization starting point in the railroad domain. The dead reckoning method then runs exclusively in the railroad domain, integrating distance increments, provided by wheel encoders, to retrieve the vehicle position along the tracks.

The domain change operation is a crucial piece that influences the design of all other blocks. It manipulates the railroad model to relate the global domain with the railroad domain and perform conversion in both directions. Two distinct algorithms deal with each direction of conversion, its details are presented next.
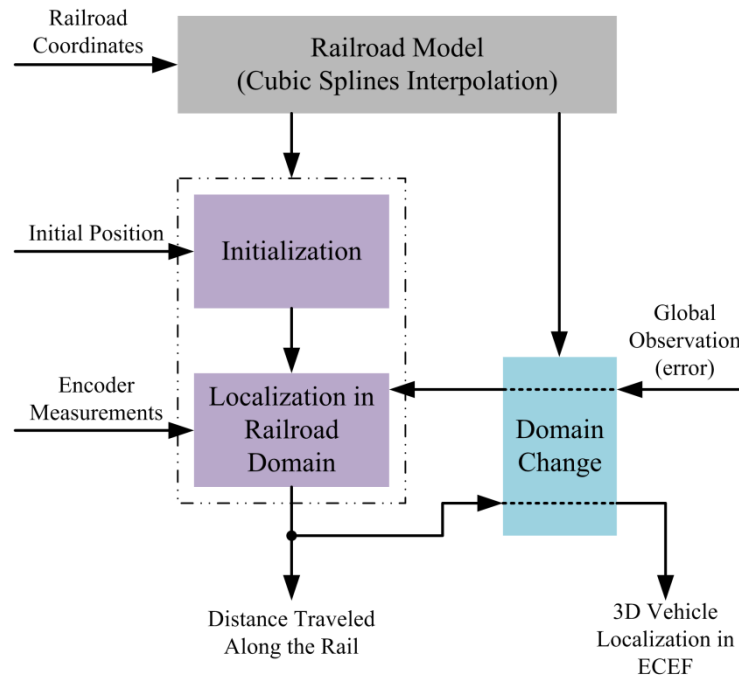
Figure 3: Conceptual diagram of the localization system

3.1. Domain Change

Through the railroad model global coordinates are projected to the railroad domain and vice-versa.

In addition to the displacement along the tracks, the domain change operation demands for additional redundant information in the railroad domain, such as the value of λ^v corresponding to the current vehicle position and the index of the spline segment in which the vehicle is located. With that knowledge, coordinates in the global domain can be easily recovered by replacing λ^v in spline equations (equation 4).

The conversion in the other direction is slightly more complex. A sequence of operations is executed to project global coordinates to the railroad domain. The conversion is applied either to the initial global position and also global observations to find the corresponding projections on the railroad model. Throughout the procedure, relevant information, including λ^p, and the spline segment in which the point is projected, are determined and used to finally compute the corresponding displacement in relation to the origin of the railroad model.

3.1.1. From Global to Railroad Domain – Spline segment and λ^{p} computation

The goal is to project a global point $P^w = (x^w, y^w, z^w)$ into the closest point along the railroad model, to determine three parameters: λ^p, the spline segment index i and the displacement ds, defined as the rail track length between the projected point and the point $\vec{p}_1$ – the origin of the railroad model.

1:      $r \leftarrow Find\ Closest\ Control\ Point(\vec{p}, P^w)$

$\triangleright\ Start\ working\ now\ on\ the\ adjacent\ segments$:

2:      $Hyp1 \leftarrow Find\ Closest\ \lambda\ (S_{r-1}^{xyz}, P^w)$

3:      $Hyp2 \leftarrow Find\ Closest\ \lambda\ (S_{r}^{xyz}, P^w)$

4:      $[\lambda^p, i] \leftarrow Check\ Domaind\ Constraints\ (\lambda, Hyp1, Hyp2, r)$

Algorithm 1 - Compute segment index and λ^p

Following Algorithm 1, the search for the closest spline segment and the computation of λ^p are carried out simultaneously. First, the nearest spline node to P^w is selected from the set of georeferenced points $\vec{p}$. The segments adjacent to the node selected establish the two possibilities for the closest spline segment. The algorithm focus then on those two spline segments, searching for the points in each one that minimize the Euclidean distance to P^w. Euclidean distance and squared Euclidean distance share the same minima, so the squared Euclidean distance is minimized instead to simplify the calculations. To formalize the problem let's consider the arbitrary spline segment r, the objective is to find the value of λ^p that minimizes the squared Euclidean distance:

$$\arg\min_{\lambda \in [\lambda_{r-1}, \lambda_r]} d^2(\lambda) \tag{6}$$

Where:

$$d^2(\lambda) = (S_r^x(\lambda) - x^w)^2 + \left(S_r^y(\lambda) - y^w\right)^2 + (S_r^z(\lambda) - z^w)^2 \tag{7}$$

The minimum of d^2 is found by computing the roots of $(\partial d^2 / \partial \lambda)$ as follows:

$$\frac{\partial d^2}{\partial \lambda} = 0 <=> \frac{\partial d_x^2}{\partial \lambda} + \frac{\partial d_y^2}{\partial \lambda} + \frac{\partial d_z^2}{\partial \lambda} = 0 \tag{8}$$

With:

$$\begin{cases} \dfrac{\partial d_x^2}{\partial \lambda} \\ \dfrac{\partial d_y^2}{\partial \lambda} \\ \dfrac{\partial d_z^2}{\partial \lambda} \end{cases} = \begin{cases} \dfrac{\partial((S_r^x(\lambda) - x^w)^2)}{\partial \lambda} \\ \dfrac{\partial\left(\left(S_r^y(\lambda) - y^w\right)^2\right)}{\partial \lambda} \\ \dfrac{\partial((S_r^z(\lambda) - z^w)^2)}{\partial \lambda} \end{cases} \tag{9}$$

Equation 8 produces five solutions, although only one real solution is expected. It should be considered the hypothetical value of λ^p for that given segment. More than one real solution is an indication of poor model resolution, hence more georeferenced rail coordinates must be used to build the railroad model.

Finally, as stated in the last line of Algorithm 1, the hypotheses are evaluated through a domain check operation. Each hypothesis must comply with the corresponding segment limits, in other words, for segment r the hypothetical value must be between λ_{r-1} and λ_r. Again, when dealing with a congruent model, only one valid hypothesis is expected. The valid hypothesis reveals also the initial spline segment.

3.1.2. From Global to Railroad Domain – Displacement in railroad domain

The displacement calculation (ds) can proceed by computing the splines arc-length between the models origin and the new point, defined by the previously computed λ^p and spline segment index. The arc-length (L) of an arbitrary spline segment r is evaluated analytically by the next formula.

$$L_r = \int_{\lambda_{r-1}}^{\lambda_r} \sqrt{\left(\frac{\partial S_r^x(\lambda)}{\partial \lambda}\right)^2 + \left(\frac{\partial S_r^y(\lambda)}{\partial \lambda}\right)^2 + \left(\frac{\partial S_r^z(\lambda)}{\partial \lambda}\right)^2} \, d\lambda \tag{10}$$

The integration limit may be simply adjusted to compute the arc-length for intermediate values of $\lambda \in [\lambda_{r-1}, \lambda_r]$. This capability is essential to compute the displacement portion of the spline segment i in which the point is projected. Accordingly, the displacement will be the sum of all spline arc-lengths from the models origin to segment $i - 1$ plus a portion of the spline segment i^{th} arc-length:

$$ds(0) = \sum_{t=1}^{i-1} L_i + \int_{\lambda_{i-1}}^{\lambda^p} \sqrt{\left(\frac{\partial S_i^x(\lambda)}{\partial \lambda}\right)^2 + \left(\frac{\partial S_i^y(\lambda)}{\partial \lambda}\right)^2 + \left(\frac{\partial S_i^z(\lambda)}{\partial \lambda}\right)^2} \, d\lambda \tag{11}$$

3.2. Dead Reckoning Method and Displacement Calculation

Now with the capability of jumping between domains, the localization starting point in the railroad domain can be determined though projection of the initial global position. After that the system is ready to perform the dead reckoning algorithm to estimate the vehicle position in the railroad domain.

First wheel encoder measurements are converted into distance increments and subsequently added to the previously estimated displacement. This simple operation computes the new vehicle position in the railroad domain, nevertheless the update of λ^v is also convenient, since it is a crucial parameter when jumping from the railroad domain to the global domain. The new λ^v value is successively approximated by iteration according to Algorithm 2.

|	
1:	**function** $(S^{xyz}, \lambda^v(k-1), ds(k), ds(k-1), threshold, i)$
2:	$error \leftarrow ds(k) - ds(k-1) \quad \triangleright displacement\ increment$
3:	$ds(k) \leftarrow ds(k-1)$
4:	$\lambda(k) \leftarrow \lambda(k-1)$
5:	**while** $error > threshold$ **do**
6:	$\lambda(k) += gain * error \quad \triangleright update\ \lambda(k)$
7:	$ds(k) += \int_{\lambda(k-1)}^{\lambda(k)} \sqrt{\left(\frac{\partial s_i^x(\lambda)}{\partial \lambda}\right)^2 + \left(\frac{\partial s_i^y(\lambda)}{\partial \lambda}\right)^2 + \left(\frac{\partial s_i^z(\lambda)}{\partial \lambda}\right)^2} \, d\lambda$
8:	$error \leftarrow ds(k) - ds(k-1)$
9:	$\lambda(k-1) \leftarrow \lambda(k)$
10:	**end while**
11:	**return** $\lambda(k)$
12:	**end function**

Algorithm 2 - Compute λ^v from total displacement ds

To increase readability, Algorithm 2 leaves out some details regarding transitions between consecutive spline segments and does not deal with movement in the reverse direction. The algorithm implements a proportional control scheme to refine the value of $\lambda^v(k)$ until the arc-length between $\lambda^v(k-1)$ and $\lambda^v(k)$ resembles the actual displacement measured by the sensors. To compute the predicted displacement the arc-length equation is used (equation 10). The iteration cycle stops when an error below $threshold = 10^{-10}$ is achieved. The threshold value is more than 10.000 times smaller than the resolution of our measuring system, so the error introduced here is insignificant.
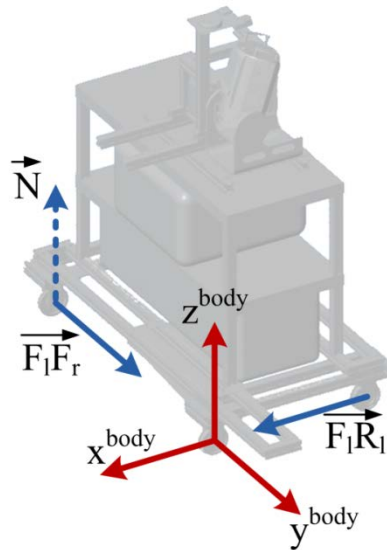
Figure 4: Representation of the body frame and vectors used to compute the vehicle orientation

3.3. Orientation Calculation

The vehicle orientation with respect to the ECEF reference frame is derived from the global position of three wheels: the two front wheels and the left rear wheel. However, only the position of the front left wheel is given by the dead reckoning algorithm. The position of the front right wheel results from the projection of the front left wheel into the closest point in the right rail track, while the rear left wheel position is estimated using a strategy similar to Algorithm 2, but, instead of searching for a given displacement, takes the Euclidean distance between those wheels to be the goal.

As illustrated in Figure 4, the coordinates of those three points define two vectors, aligned with booth x^{body} and y^{body} axes of the body frame. Tacking the cross product of the two normalized vectors will result in a direction vector perpendicular to the plane defined by $\overrightarrow{F_lF_r}$ and $\overrightarrow{F_lR_l}$. There is no guarantee that vectors $\overrightarrow{F_lF_r}$ and $\overrightarrow{F_lR_l}$ are orthogonal so vector $\overrightarrow{F_lF_r}$ is re-estimated by tacking the cross product between $\overrightarrow{F_lR_l}$ and $\vec{N}$. After this set of operations we end up with three directional vectors that define a three-dimensional coordinate system. The vectors can be organized in a matrix form to define the rotation matrix between the ECEF coordinate system and the body frame:

$$R_{ECEF}^{body} = \left[\overrightarrow{F_lR_l}^T \quad \overrightarrow{F_lF_r}^T \quad \vec{N}^T \right]$$

(12)

Finally, the Euler angles are extracted from the rotation matrix using a standard algorithm.

3.4. Post-processing Method

As explained before, at least one global reference is needed to initialize the system in the railroad domain. Any additional global observation, taken during the vehicle operation, can be projected to the railroad model to stipulate precisely the current vehicle displacement. The displacement error is then used to compensate past encoder measurements until the instant of the previous observation.

The proportion between the displacement error d_{err} accumulated between observations at time t_{obs1} and t_{obs2}, and the error expressed as a function of the displacement $ds_{t_{obs1} \to t_{obs2}}$ between those observations becomes:

$$prop = \frac{d_{err}}{ds_{k-1 \to k}}$$

(13)

For a vehicle moving only in one direction, the displacement error can be compensated by applying the above correction constant to the encoder measurements:

$$enc(t) = enc(t) + enc(t) * prop \tag{14}$$

Where $t \in [t_{obs1}, t_{obs2}[$.

This procedure stretches or compresses the displacement measured by the wheel encoders. It compensates various sources of error, namely the lack of adherence between the wheels and the rail, inaccuracy in the wheel radius measurement, and the rail model error.

The dead-reckoning algorithm must be executed again to process the corrected measurements between instants t_{obs1} and t_{obs2}. After that, the process can be repeated for the next observation $obs3$ and so on.

4. Results and Discussion

The dataset was collected in a disabled rail line with an extension of near 300 meters, see Figure 5. The non-motorized modeling vehicle used for data collection carried a variety of sensors along with a computational system and a power unit. The two front wheels were equipped with encoders with a resolution of 16000 ticks per revolution, which associated to a wheel with a 10 centimeters radius assure a metric resolution of 0.039 mm. Measurements provided by a INS/GPS iMAR iNAV-FMS-E, further processed in a tightly-coupled fashion using a commercial software (Waypoint Inertial Explorer), produce the ground truth used to evaluate the performance of the purposed localization method. The railroad model was built using 10 rail points extracted by GPS, with approximate distance of 30 meters between each other. Finally, the landmarks introduced in the post-processing stage consist on reflective targets detected by a RIEGL VZ-400 laser scanner.

Figure 5: Picture of the mobile modeling vehicle during data acquisition in a railroad. A reflective target can be seen at the right of the vehicle wheels
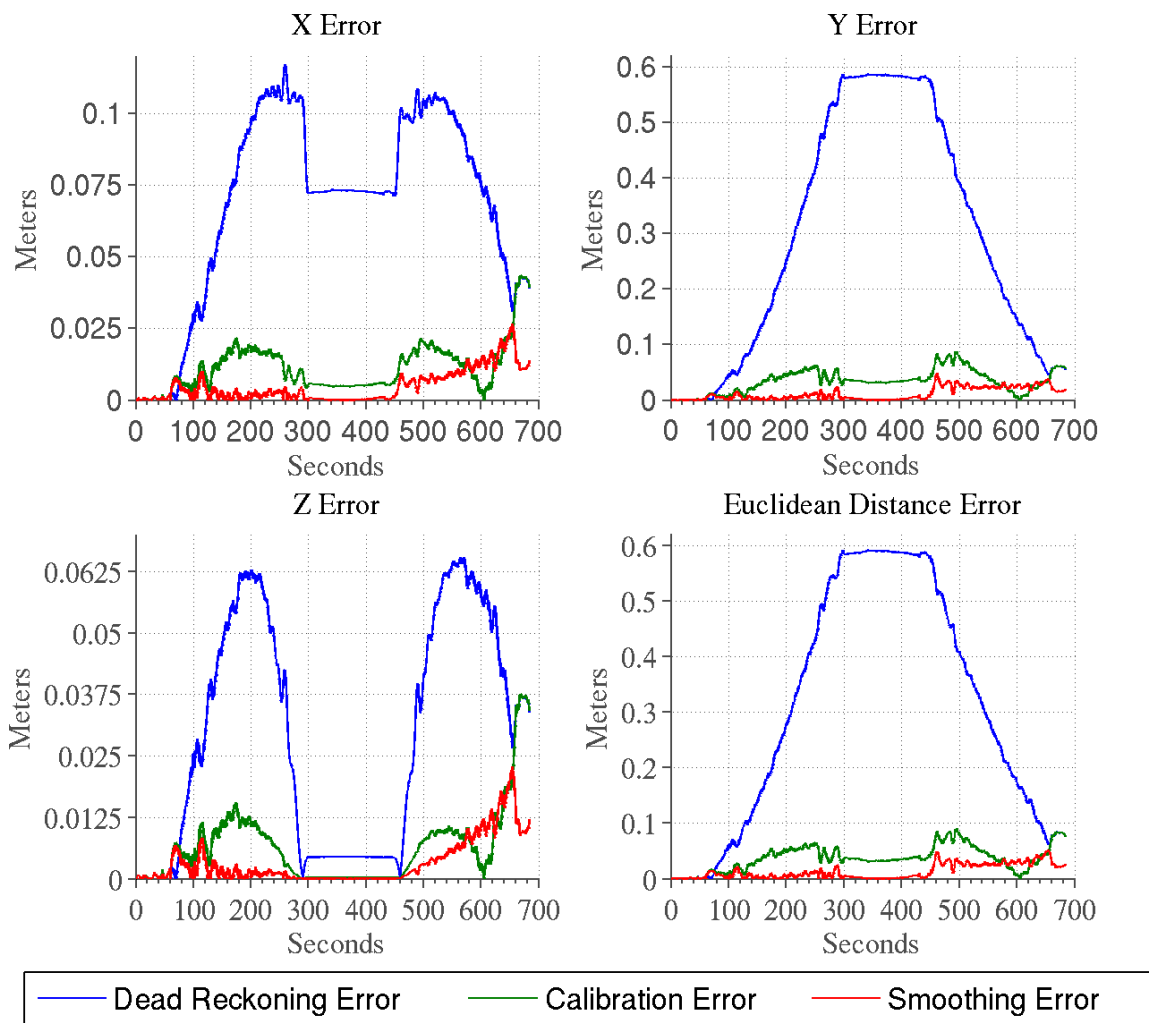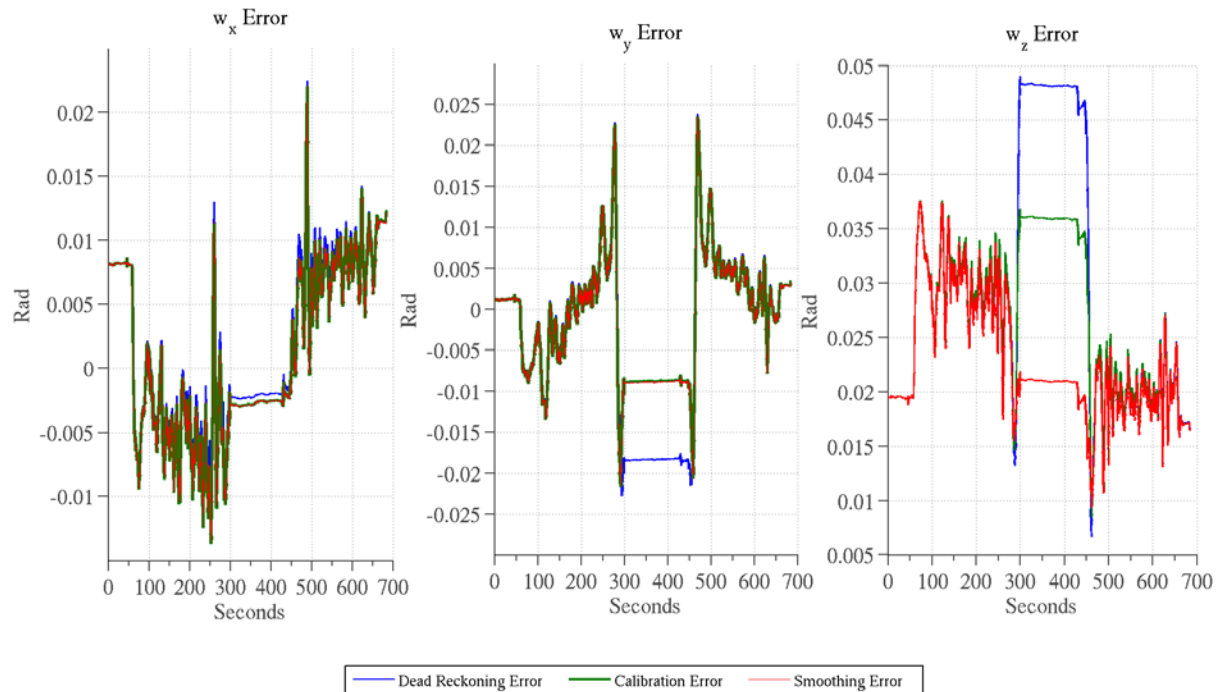
Figure 6: Position errors in each component and Euclidean distance error of the three localization variants regarding the ground truth.

Two variants of the post-processing algorithm were tested, one – denominated as smoothing – processes a series of landmarks placed across the rail. The other methodology behaves as a calibration procedure, where a short path, with approximately 40 meters, is processed with only two global observations, at the beginning and at the end of the path. The objective is to compute the overall error and apply the compensation to the measurements collected in a longer experiment.

The results shown below characterize the performance of the dead reckoning approach and the two post-processing methods. In the test presented the vehicle traveled forward for about 220 meters stopped for a while and traveled back towards the starting point. During motion, five landmarks are observed twice – one observation in each direction. Position errors in all components (see Figure 6) clearly show the advantage of the post-processing methods in improving the dead reckoning results. The calibration method takes less information than the smoothing post-processing solution, this justifies the higher errors of the calibration approach. Still, the outcome is notably superior when compared with the plain dead reckoning method. Analysis of the absolute error (Euclidean distance to the ground truth), indicates once more the ability of the post-processing solutions to reduce the error of the dead reckoning approach.

Interesting error symmetry is observed between forward and reverse directions. At the same rail zones a similar error behavior, regardless the traveling direction, indicates rail model deficiencies in those particular spots. The rail model errors are intrinsic to the system and cannot be totally smoothed through the post-processing technique, nevertheless the gain in accuracy given by the post-processing phase is obvious.

Figure 7: Orientation errors in each component of the three localization variants regarding the ground truth.

The orientation errors depicted in Figure 7 reflect mainly the railroad model deficiencies, due to the heavy influence of the railroad model in the orientation calculation. Therefore, the orientations computed based on the post-processed trajectories do no achieve a significant improvement regarding the ones computed through dead-reckoning. In the worst case the post-processing methods present an error inferior to 2.5 degrees, which is still quite high and unacceptable for 3D mobile modeling applications.

5. Conclusions and Future Work

A localization method for rail vehicles was presented. It relies on previous geometric measurements to establish rigid rules for the rail vehicle movement. Those restrictions complement a simple and inexpensive perception system, and enable the localization estimation in six degrees of freedom. The post-processing solutions, purposed to reduce the global position errors, accomplished meritorious results with errors almost ten times smaller than the ones achieved by the dead reckoning solution. Moreover its application prevents the unbounded error growth over time.

The orientation calculation method revealed itself highly sensitive to the railroad model error, and neither the dead reckoning nor the post-processing trajectories led to acceptable orientation estimates. In the future, the inclusion of orientation observations, provided for instance by cameras, may be considered to reduce orientation and railroad model errors.

References

Glaus, Ralph. 2006. "The Swiss Trolley - A Modular System for Track Surveying Geodätisch-geophysikalische Arbeiten in der Schweiz", Schweizerische Geodätische Kommission, ISBN 3-908-440-13-0.

Nüchter, Andreas, Hartmut Surmann, Kai Lingemann, Joachim Hertzberg and Sebastian Thrun. 2004. "6D SLAM with an Application in Autonomous Mine Mapping". IEEE International Conference on Robotics and Automation (ICRA), pages 1998-2003.

Thrun, Sebastian, Scott Thayer, Warren Whittaker, Christopher R. Baker, Wolfram Burgard, David Ferguson, Dirk Haehnel, Michael Montemerlo, Aaron Christopher Morris, Zachary Omohundro, Carlos Reverte and William L. Whittaker. 2004. "Autonomous Exploration and Mapping of Abandoned Mines". IEEE Robotics and Automation Magazine, pages 79-91.

Burden, Richard L. and J. Douglas Faires. 2005. "Numerical Analysis", 7th edition Brooks/Cole, ISBN 9780534392000.

Bartels, Richard H., John C. Beatty, and Brian A. Barsky. 1987. "An introduction to splines for use in computer graphics & geometric modeling". Morgan Kaufmann Publishers Inc., ISBN 0934613273.

Acknowledgment