# ECMI-REPORT
## Odometry for train location

Bubacarr Bah, Eva Jungabel, Monika Kowalska,
Christian Leithäuser, Ankur Pandey, Carsten Vogel

February 7, 2009

# Contents



# List of Figures



# List of Tables

# 1 Introduction

The European Train Control System (ETCS) is a signalling control and train protection system. Although ETCS have been underway for many years, there is now a desire to deploy the technology operationally, but the system costs are too high in comparison with the benefits that can be achieved. A major cost component concerns positioning equipment on the train: odometers, radars and balise readers. Additionally, there are costs arising from installation, fleet migration time and ongoing maintenance. So it has become more and more important to find a new train positioning system which uses equipment which is cost effective, i.e. easy to install and cheap to maintain, and which can help to achieve faster migration times.

Reliable Data Systems International (RDS International) has come up with a cheaper alternative for the implementation of ETCS — the Video Train Positioning System (VTPS). It is expected that, in this new train positioning system, a new speed and distance measurement sensor is used to simplify ETCS migration and reduce cost. The sensor uses real-time video from a camera mounted on the leading vehicle within the cab or externally. From the sequence of video frames, the system computes speed and distance travelled. In addition to being cheaper, another advantage of this system is that it is capable of all-weather operation. Furthermore, the technology can be housed in a portable unit for temporary installation, for example for use in proving freight train integrity. For verification of the above claims refer to the RDS website: `www.rdsintl.com`.

This system is not error free. There are bound to be errors resulting from the misalignment of frames due to the movement of the train on its suspensions and the curvature of the tracks. However, these errors are not allowed to accumulate a lot but are instead corrected based on known land marks along the tracks including "visual balises". RDS International claims to have estimated this error to be roughly 2.5% which meets the ETCS target. The work of this project is an attempt to verify this claim through a more rigorous mathematical analysis of the error.

To achieve this goal we started by using simple geometry and trigonometry to undo the projection transformation which maps the 3D world onto the 2D image to give an image which appears to look directly down onto the tracks. This is done in the next section. In Section 3, we give a demonstration of the projection transformation by numerically implementing our mapping. Section 4 entails the error analysis which is followed by a conclusion.

## 2 Projection Transformation

As afore-stated we had to find an appropriate projection transformation from $\mathbb{R}^3$ to $\mathbb{R}^2$ to give an image that appears to look directly down onto the tracks. The effect of this is shown in Figure 1. This actually summarises the image processing phase of the speed calculation.

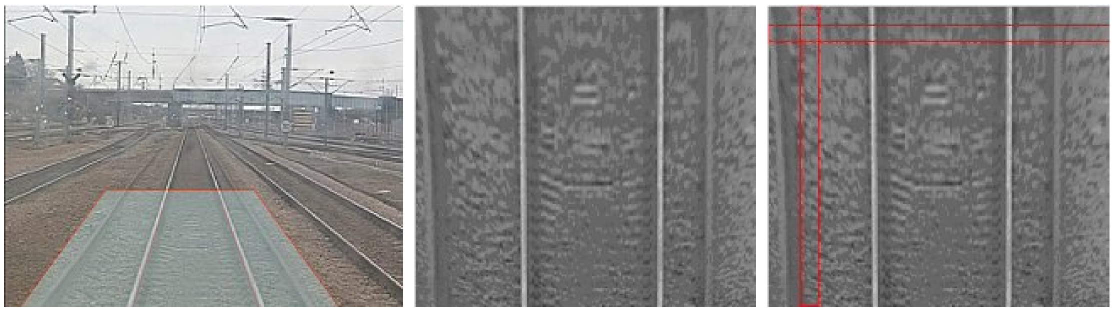


Figure 1: An image of the tracks showing the trapezoidal area processed as a single frame (left) tracks transformed (middle), and tracks with strips (right). *Courtesy of* RDS International.

We sketch a schematic diagram of the 2D image and the real tracks in Figure 2. What is labelled as the image plane is equivalent to the light gray trapezoidal area in Figure 1. This diagram incorporates extra information like the origin, $O$, of the $x$, $y$ and $z$ coordinate system; the height, $h$, at which the camera is mounted above ground; the tilt of the camera, $\theta$, the vertical and horizontal view angles of the camera, $\alpha$ and $\beta$ respectively; the width of the tracks, $2W$; the distance, $d$, of the beginning of the frame from the origin, $O$; arbitrary distances, $p$ and $q$, on the image in pixels and their corresponding distances, $p'$ and $q'$, on the ground in metres (or kilometres).

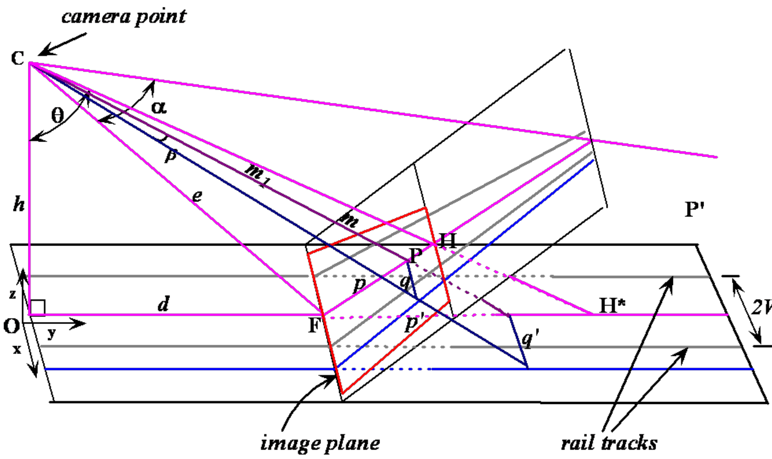



Figure 2: A schematic of the image and the real rail tracks.

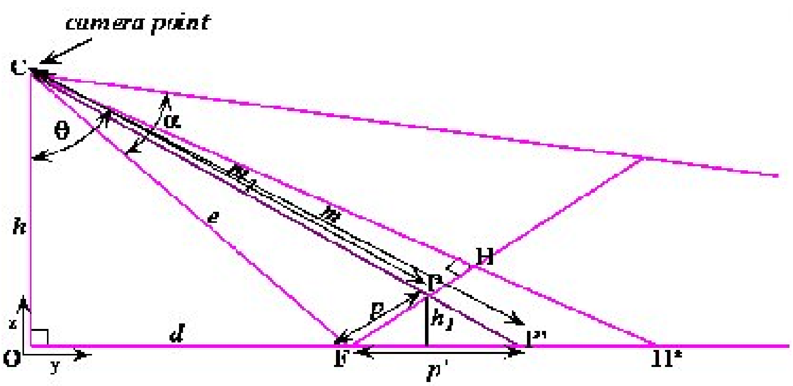



Figure 3: Vertical cross-sectional view of the schematic in Figure 2.

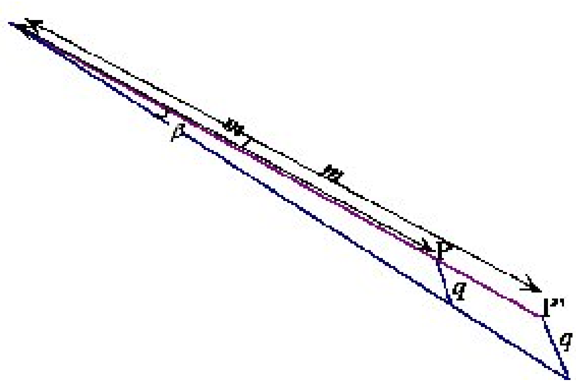



Figure 4: Horizontal cross-sectional view of the schematic in Figure 2.

Let

$$p' = a + b, \tag{1}$$

where

$$a = \sqrt{p^2 - h_1^2}, \tag{2}$$

and

$$b = \sqrt{(m - m_1)^2 - h_1^2}. \tag{3}$$

Through similar triangles we have the following equation

$$\frac{h_1}{h} = \frac{m - m_1}{m}. \tag{4}$$

Substituting (4) in (3) we have

$$b = \frac{h_1}{h}\sqrt{m^2 - h^2}. \tag{5}$$

But it could be seen from the diagram in Figure 2 and Figure 3 that

$$m^2 = h^2 + (d + p')^2, \tag{6}$$

and substituting this into (5) gives

$$b = \frac{h_1}{h}(d + p'). \tag{7}$$

With $h_1 = p \sin\theta$, from (1) we get an expression for $p'$ as

$$p' = \frac{p(h\cos\theta + d\sin\theta)}{h - p\sin\theta}. \tag{8}$$

For calculating the value for $q'$ we use the relation

$$\frac{m_1}{m} = \frac{q}{q'}. \tag{9}$$

This result is also clearly based on similar triangles as can be seen from Figure 4. So what is needed is the expression of $m$ and $m_1$ in terms of known variables $h$, $p$, $d$ and $\theta$. From Equation 6 we deduce

$$m = \sqrt{h^2 + (d + p')^2}.$$

By using the already derived value for $p'$ this becomes

$$m = \frac{h\sqrt{(h - p\sin\theta)^2 + (d + p\cos\theta)^2}}{h - p\sin\theta}. \tag{10}$$

Next we determine $m_1$ from the relation (4) to get

$$m_1 = m(1 - \frac{h_1}{h}).$$

Therefore, using (8) again we get

$$m_1 = \sqrt{(h - p\sin\theta)^2 + (d + p\cos\theta)^2}. \tag{11}$$

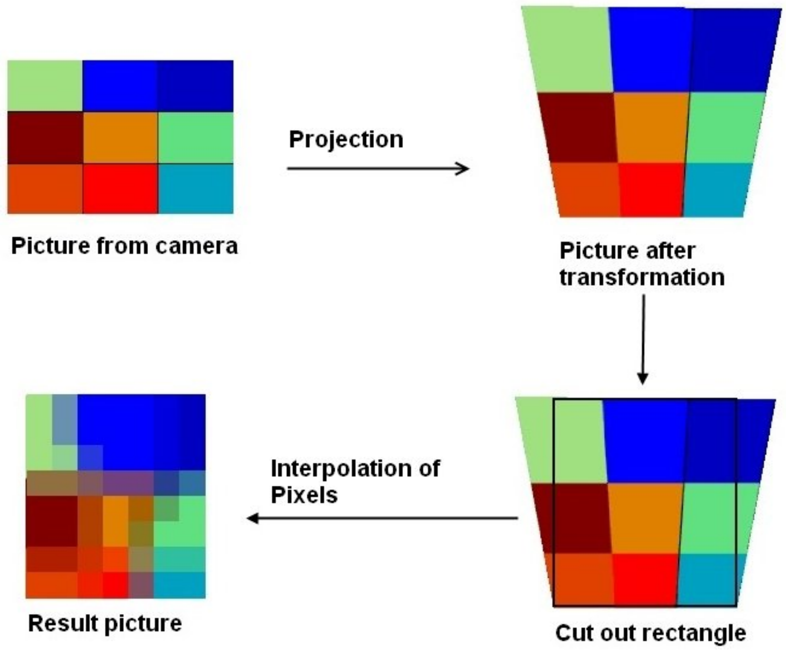
Finally substituting (10) and (11) into (9) yields

$$q' = \frac{hq}{h - p\sin\theta}. \tag{12}$$

# 3 Numerical Implementation

A program in Matlab code is used to apply the transformation of real life data, which is supplied by RDS. The data consists of 300 still images which were extracted from a movie taken during an RDS test run. For each picture the current speed, measured by the RDS system is known. We can now perform computations for the trains velocity using our own transformations and compare them to the RDS results.

## 3.1 How the algorithm works




The algorithm works in five steps:

1. The picture is loaded. The colorvalues of each pixel are stored in the matrix $C$. Their positions are stored in $X$ and $Y$.

2. Using the projective transformations, $q(X)$ and $p(Y)$ are computed. The colorvalues are not affected by this.

3. The resulting picture has the form of a trapeze and pixels are not equidistantly distributed. In order to make it possible to visualize this on the computer screen, we have to make it rectangular and pixels have to be equidistant. Therefore we cut out a rectangle and put an equidistant grid over it. The gridpoints will be the new pixel positions. But because the new pixels do not exactly correspond to the original pixels, we have to interpolate their color values.

4. The positions of the rails are identified, because their distance apart is known precisely. This is used to correct the scale in the transformed picture.

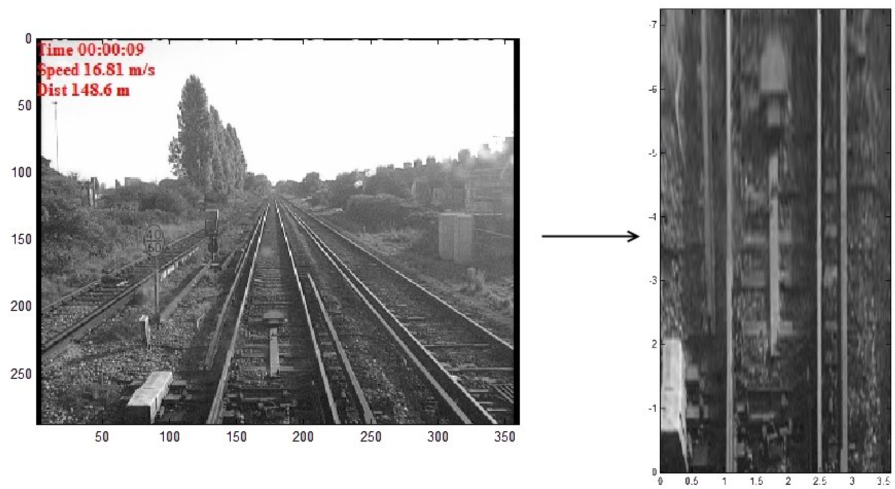
5. The previous steps are applied to two consecutive pictures. The user has to identifie a point $P_0$ which matches in both pictures (the RDS system does this automatically). Knowing the position of the point $P_0$ the distance between the two pictures is computed. From this we get the speed of the train.

## 3.2 Applying algorithm to data

In order to apply the transformations to the data provided by RDS we have to know the parameters $h$, $\alpha$, $\theta$ and the frame rate $f$. The frame rate is known to be $f = 25s^{-1}$. We did not receive any values for the other parameters. To make the transformation work, we have to use some good guesses for these values. A typical camera height and camera angle might be $h = 3\,\mathrm{m}$ and $\alpha = 35\,^{\circ}$. A reasonable value for the mounting angle $\theta$ can be obtained by looking at the relative position of the horizon in the pictures and using trigonometry. These initial guesses are calibrated by applying them to some test cases and comparing the output. The following values show good results and will be used in further computations:

$$\begin{aligned} h &= 2.9\,\mathrm{m} \\ \alpha &= 33.8\,^{\circ} \\ \theta &= 86.4\,^{\circ} \\ f &= 25s^{-1} \end{aligned}$$

Because there is no way to get more reliable values for these parameters, this is a big cause for errors.




# 4 Error Analysis

There are several causes for errors which effect the performance of the train positioning system:

- Errors in the camera position, due to the initial installation of the camera and subsequently to the shaking of the train when in motion.
- Errors in picture processing. It is not possible to measure the distance between the same point on two pictures exactly.

Our first thought was to compute the worst possible error, which can occur during a single distance measuring step. We did this by specifying individual error bounds for each input parameter and used this to compute the maximal error in the distance measurement. Predictably this error is far out of bounds and there would be no use for a positioning system with such high error ranges. But of course it is not realistic to look at the worst possible error. Because we are doing many measuring steps consecutively, the single errors in each step will cancel out each other and the real error tend to be much smaller than adding up the maximal possible errors in each step. Therefore in order to compute realistic error bounds, we have to use statistics to investigate how the error evolves after several steps. Unfortunately we had not enough time during the workshop to go really deep into this subject. So what we did instead is an empirical error analysis. For this we used real data which was provided by RDS and the goal was to compute error bounds for this data. Of course this is only an example and no mathematical proof, but it is what we managed to do in the time available.

## 4.1 Empirical error

We are going to apply our algorithm to 40 consecutive frames from the data set supplied by RDS. This enables us to see, how it might perform in reality. We can investigate, how our calculated velocity compares to the one computed by RDS. We also estimated the total distance travelled and compared it to the actual distance travelled.

The table compares the velocity measured by RDS with the velocity from our algorithm, for each picture. One can see that the velocity values are jumping between two consecutive measurements, which is due to measuring errors. When using the system in a real train, smoothing must be applied, possibly by some Kalman filter. For our case this is not necessary, since we are interested in the errors.

At an average our values for the velocity are $E_v = 0.19$ m s$^{-1}$ higher than the values from RDS. The standard deviation is $\sigma_v = 0.21$ m s$^{-1}$. Therefore the distance travelled will differ by $E_d = 0.0075\,\mathrm{m}$ with standard derivative $\sigma_d = 0.0085\,\mathrm{m}$ between two pictures from the camera. The average distance travelled between two consecutive pictures is $d = 0.67\,\mathrm{m}$.

Therefore if the train covers a distance of $D = 1000\,\mathrm{m}$, the distance measured by our algorithm will be in the following interval with 95 % probability:

Table 1: Comparison of measured and calculated velocity.

| pic | RDS [m $s^{-1}$] | algorithm [m $s^{-1}$] | pic | RDS [m $s^{-1}$] | algorithm [m $s^{-1}$] |
|---|---|---|---|---|---|
| 1255 | 16.77 | 16.32 | 1275 | 16.79 | 17.45 |
| 1256 | 16.72 | 16.96 | 1276 | 16.78 | 16.21 |
| 1257 | 16.68 | 16.81 | 1277 | 16.79 | 16.69 |
| 1258 | 16.65 | 16.67 | 1278 | 16.79 | 16.97 |
| 1259 | 16.63 | 16.34 | 1279 | 16.80 | 17.54 |
| 1260 | 16.62 | 16.36 | 1280 | 16.83 | 17.87 |
| 1261 | 16.62 | 16.59 | 1281 | 16.86 | 17.75 |
| 1262 | 16.62 | 17.25 | 1282 | 16.89 | 17.51 |
| 1263 | 16.61 | 16.82 | 1283 | 16.92 | 17.60 |
| 1264 | 16.64 | 17.48 | 1284 | 16.95 | 16.92 |
| 1265 | 16.66 | 16.72 | 1285 | 16.99 | 17.24 |
| 1266 | 16.68 | 16.66 | 1286 | 17.01 | 16.85 |
| 1267 | 16.69 | 16.94 | 1287 | 17.02 | 16.20 |
| 1268 | 16.71 | 17.53 | 1288 | 17.03 | 16.70 |
| 1269 | 16.72 | 17.21 | 1289 | 17.02 | 17.58 |
| 1270 | 16.74 | 16.40 | 1290 | 16.99 | 17.76 |
| 1271 | 16.76 | 16.97 | 1291 | 16.95 | 16.47 |
| 1272 | 16.78 | 16.90 | 1292 | 16.91 | 17.41 |
| 1273 | 16.79 | 16.89 | 1293 | 16.84 | 17.51 |
| 1274 | 16.79 | 16.28 | 1294 | 16.77 | 16.97 |

$$
\begin{aligned}
I &= 1000 + \frac{D}{d} \times [E_d - 2 \times \sigma_d \ , \ E_d + 2 \times \sigma_d] \\
&= [985 \ , \ 1037]\text{m}
\end{aligned}
$$

The maximum error is 37m $\text{km}^{-1}$. Of course this error bound has to be handled with care, because it was derived from quite a small set of data and the above computation is only valid for velocities close to 17m $\text{s}^{-1}$.

## 5 Conclusion

In order to ensure safety in such a complex system as a train network, one needs a reliable positioning system. This system has to be very accurate and it needs to be verified that it really possesses this accuracy. Of course we were not able to give a mathematical proof for the reliability of the positioning system in the short time of the workshop. But we were able to do a simple implementation for the system and tested it for a real life case. For this test case the system's performance was accurate enough to provide reliable data for train positioning. This does not guarantee that the system works under any conditions, but it does justify further research on the subject, to improve accuracy and prove reliability.