Relative Localization from Image Sequences

Brian L. Stuart
Innovation/Scanning Technology
FedEx Corporate Services
2847 Business Park Dr.
Memphis, TN 38118
blstuart@fedex.com

ABSTRACT

This paper presents a technique for vehicle localization. The technique is based on processing images taken by one or more downward pointing cameras. Motion represented by shift between successive images allows us to directly measure the velocity at two points on either side of the axis of motion. From these possibly different velocities, we can infer the vehicle speed and turning radius and from those, the vehicle orientation, position, velocity and acceleration vectors.

Categories and Subject Descriptors

L.4.8 [**Image Processing and Computer Vision**]: Scene Analysis—*motion, tracking*; I.2.9 [**Artificial Intelligence**]: Robotics—*autonomous vehicles*

General Terms

Algorithms, Localization

1. BACKGROUND

In order to make navigational decisions, an autonomous robot must know its position and orientation. Borenstein, et al[1] present a good survey of common techniques in the field of localization. Most techniques can be categorized according to whether they provide absolute position and orientation or relative position and orientation.

Common examples of absolute positioning systems include the Global Positioning System (GPS), magnetic compasses and landmark identification. All of these require external references. These external references are subject to interference. For example, vehicles traveling cross-country almost always experience intermittent loss of GPS signals. Environments with significant steel structures (such as warehouses and airports) will interfere both with GPS signals and with magnetic compasses. Landmark-based localization, on the other hand, requires that the robot have prior knowledge of the location and appearance of landmarks. In some applications, this is simply not possible. In others, anything which removes, defaces or obscures a landmark interferes with localization.

The most common forms of relative positioning are wheel odometry and inertial navigation. Just as an automobile counts axle rotations to measure distance traveled, many robotic systems measure axle or wheel rotations to determine position. Steering angle can be included to determine orientation. More sophisticated implementations measure the rotation of all wheels to determine both orientation and position. Incremental changes in wheel position are integrated to give a total distance traveled. Because we are integrating once, a constant error in the data will yield a distance error that grows linearly with time. However, wheel odometry is subject to unobserved interactions with the environment. The most common of these is wheel slip. Just because a wheel turns a certain angle about a known axis does not mean that the vehicle necessarily moves accordingly. Sometimes the wheels will lose traction and at other times external forces will cause motion not induced by the wheels.

Inertial navigation systems are based on measurements from accelerometers or gyroscopes. While they are not subject to the environmental interactions that can affect wheel odometry, they are not without a downside. Since we must integrate acceleration twice to get position, any constant error will manifest itself as a position error that grows with the square of time. Gyroscopes, on the other hand, suffer from drift over time even when the vehicle is still.

These techniques are rarely used in isolation. For example, in a constrained environment such as a factory, we may provide landmarks which can update a vehicle's absolute position when encountered but use relative positioning systems between landmarks. Similarly, in cross-country applications, we often use GPS to give an absolute position when possible. However, during loss of GPS signal, wheel odometry and inertial navigation devices are often used to augment GPS. It is quite common for the various sources of data to be combined using Kalman filters[4].

The remainder of this paper describes a new technique that shows the linear error growth of odometry without being subject to the same environmental effects. It is also not subject to the drift that affects gyroscopic systems.

2. MOTION INFERENCE

Our technique is a form of odometry based on processing images of the ground taken as the vehicle moves. Thus it is immune to issues like wheel slippage. This is similar to the technique used by optical mice as they track movement

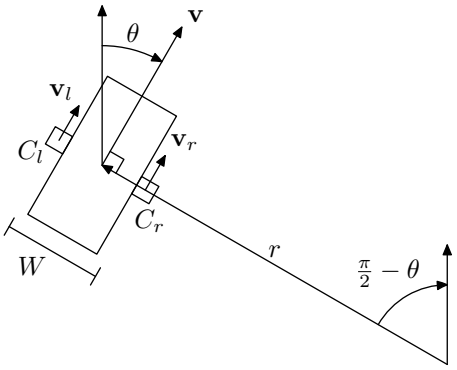

Figure 1: Vehicle Geometry

over textured surfaces. Suppose we have two images I_1 and I_2 taken Δt seconds apart. If we correlate the images and find that I_2 is shifted from I_1 by Δx pixels in the horizontal direction and Δy pixels in the vertical, then we can compute an instantaneous approximation to the vehicle velocity by

$$v = \frac{s\sqrt{(\Delta x)^2 + (\Delta y)^2}}{\Delta t} \tag{1}$$

where s is a scaling factor that relates a distance in pixels to a distance in the physical world. In the experiments reported here, it is calibrated in meters/pixel.

This computation of velocity gives us only the magnitude which would be sufficient if the vehicle always traveled in a straight line. In order to determine the direction of the velocity, however, we must compute the change in orientation. We base this calculation on the assumption that between successive images the vehicle moves in a circular arc with radius r, as illustrated in Figure 1. Before deriving an expression for r, we need to point out a simplifying observation. The vehicle's orientation is always at right angles to the radius of the turn. Consequently, the change in orientation angle is always equal to the change in angle about the center of the turn arc. This observation allows us to speak solely in terms of the change in orientation angle throughout this paper. Furthermore, throughout this derivation, we talk about a turn to the right. It should be noted that the results still follow in the case of a left-hand turn if we take $r < 0$ to indicate the left-hand direction. Referring again to Figure 1, the distance traveled and the length of the arc, l, are related by

$$l = vt = r\theta$$

where v is the magnitude of the velocity vector, $\mathbf{v}$. Thus

$$\theta = \frac{1}{r}vt$$

and the change in orientation is given by

$$\frac{d\theta}{dt} = \frac{1}{r}v. \tag{2}$$

We position two cameras (labeled C_l and C_r in Figure 1) separated by a distance W such that they lie along the turning radius and such that the center between them coincides with the center of the vehicle. Since the change in orientation of the cameras is the same as that of the vehicle, the velocity seen by the left-hand camera, C_l, is given by

$$v_l = \frac{1}{r}v\left(r + \frac{W}{2}\right) = \left(1 + \frac{W}{2r}\right)v$$

and that seen by the right-hand camera, C_r, is given by

$$v_r = \left(1 - \frac{W}{2r}\right)v.$$

Solving for v and setting the two expressions equal gives us

$$v = \frac{v_l}{\left(1 + \frac{W}{2r}\right)} = \frac{v_r}{\left(1 - \frac{W}{2r}\right)}.$$

Solving for r gives

$$r = \frac{W}{2}\left(\frac{v_l + v_r}{v_l - v_r}\right).$$

Working backwards, we find that the vehicle velocity is just the mean of the left and right camera velocities,

$$v = \frac{v_l + v_r}{2}. \tag{3}$$

From the turning radius, we can calculate the orientation angle at time T by integrating (2)

$$\theta = \int_0^T \frac{1}{r}v\ dt \tag{4}$$

$$= \int_0^T \frac{2}{W}\left(\frac{v_l - v_r}{v_l + v_r}\right)\frac{v_l + v_r}{2}\ dt \tag{5}$$

$$= \int_0^T \frac{1}{W}(v_l - v_r)\ dt. \tag{6}$$

This gives us the velocity vector

$$\mathbf{v} = (v\cos\theta, v\sin\theta). \tag{7}$$

Similarly, we obtain the position vector by integrating the velocity components

$$\mathbf{p} = \left(\int_0^T v\cos\theta\ dt, \int_0^T v\sin\theta\ dt\right). \tag{8}$$

The final component of the state vector is the acceleration which is given by the derivative of the velocity

$$\mathbf{a} = \left(\frac{d}{dt}v\cos\theta, \frac{d}{dt}v\sin\theta\right). \tag{9}$$

3. IMPLEMENTATION

In Section 2, we work from the model of two cameras, one on either side of the vehicle. Early experimental work followed this model. However, we later migrated to a model where a single camera is used. The camera is positioned at the front of the vehicle looking at the ground just in front of the vehicle as it moves. This arrangement is shown in the picture in Figure 2.

The camera we use is an Axis 2100, which is capable of streaming JPEG compressed images at a rate of approximately 10 per second. It is configured to send images which are 320 pixels wide by 240 pixels tall. Each image is decompressed into a one byte-per-pixel greyscale representation.

After decompression, we apply a correction to remove the barrel (fisheye) distortion. We use the method of Perš and

Figure 2: Test Vehicle

Kovačič[5]. For each pixel in the corrected image, we determine its distance from the center, r_l, and compute the corresponding position in the original (distorted) image by

$$r' = f \ln \left(\frac{r_l}{f} + \sqrt{1 + \frac{r_l^2}{f^2}} \right)$$

where f is the focal length of the lens in the same units as r_l. We use bilinear interpolation to find the pixel value in the new image. Figure 3 shows an example of such an image after distortion removal.

Next we extract two sub-images, each 100×100 pixels. The two images are positioned at the extreme left and right edges of the original image and are centered vertically. These two sub-images serve as the left and right cameras.

In order to find the velocity magnitude, we need to determine the relative shift between successive images on each side. We use the orientation correlation technique of Fitch, et al[2] to find this shift. Their technique involves creating a new pair of images in which each position is a complex number giving the direction of the gradient at that position of the original image. For image I_1, we compute

$$g_1(x,y) = \mathrm{sgn} \left(\frac{\partial I_1(x,y)}{\partial x} + i \frac{\partial I_1(x,y)}{\partial y} \right)$$

where

$$\mathrm{sgn}(x) = \begin{cases} 0 & \text{if } |x| = 0 \\ \frac{x}{|x|} & \text{otherwise} \end{cases}$$

and where we approximate the partial derivatives by

$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x-1,y) - f(x+1,y)}{2}.$$

Likewise, we compute g_2 from image I_2. The correlation is then computed quickly using Fast Fourier Transforms (FFTs). If $G_1(x,y)$ is the FFT of $g_1(x,y)$ and $G_2(x,y)$ is the FFT of $g_2(x,y)$, then

$$\Re\{\mathrm{IFFT}(G_1(x,y)G_2^*(x,y))\}$$

gives what Fitch, et al call the orientation correlation matching surface. The location of the peak in this surface gives the shift amount in the x and y directions. (We use the FFTW[3] library to compute the FFTs and inverse FFTs.) From the resulting shifts, we compute the camera velocities by Equation (1). Then, we compute the velocity magnitude by Equation (3), the orientation by Equation (6) and the other state vector components using Equations (7)–(8). In numerically evaluating the integrals of Equations (6) and (8), we use trapezoid rule integration. (We do not compute the acceleration in the implementation reported here.)

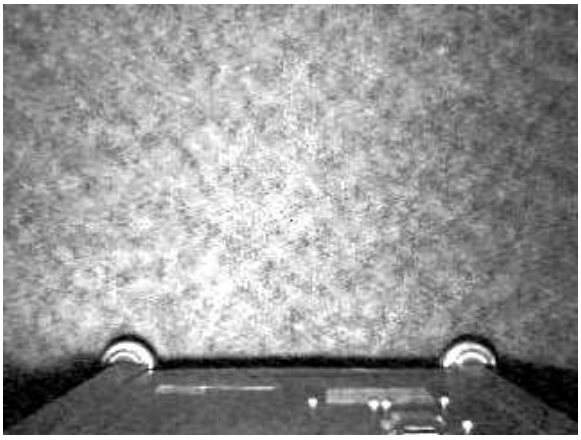

Figure 3: Example Image After Distortion Correction

4. CALIBRATION

There are two physical parameters in our model that must be calibrated. The first is the parameter s in Equation (1), and the second is the camera separation parameter W in Equation (6). In principle, both of these can be measured directly from static images. However, we can get greater precision by moving the vehicle in a known trajectory and using the results to calculate backwards to the correct parameter values.

To complicate matters, the previous two sections assume ideal camera placement and orientation. In a real configuration, however, there are errors in placement and orientation. Errors in absolute placement of the camera do not affect the results. The fact that the camera in our experiments is placed forward of the vehicle's center of gravity only means that we track the position of the camera rather than the position of the vehicle. However, because the relative position and orientation of the camera to the vehicle is fixed, a simple transformation gives us the orientation and position of the vehicle.

Errors in camera orientation do affect the results. Consider first an error in the orientation of the camera about the axis parallel to the direction of motion. If the camera is rotated about this axis, the line from the camera to the center of its imaging area will not be normal to the ground. As a result, there will be a perspective effect that differentiates the left sub-image from the right. For example, if

the camera is pointed slightly to the left, then the left sub-image will cover a greater surface area of the ground than the right. In effect, the scaling factor s will be larger for the left sub-image than for the right. A similar effect occurs for the axis which is normal to the ground. Suppose that the camera is rotated slightly counterclockwise (when viewed from above) about this axis. Between any consecutive pair of images there is diagonal movement (in addition to the rotation) of the center point between the two sub-images. When this movement is to the left in this example, the distance between the sub-images appears to be larger than when the movement is to the right.[1] The net effect is that for left-hand turns, the value of W is larger than it is for right-hand turns. The third axis is parallel to the ground and perpendicular to the direction of travel. Because any rotation about this axis affects both sub-images in the same way, we ignore it in the work reported here.

Because of these orientation effects, we use a calibration procedure that produces separate scaling values s_l and s_r for the left and right sub-images, respectively. Likewise, the procedure allows us to set values of W_l and W_r which will be used in Equation (6) depending on which direction we are turning. (We know the direction by whether v_l or v_r is the larger value.) The overall calibration procedure is:

1. Determine an initial value of $s = s_l = s_r$ directly from images of a tape measure.

2. Move the vehicle in a straight line for a known distance. Adjust the values of s_l and s_r by a factor of the distance traveled divided by the distance reported.

3. Again, move the vehicle in a straight line, accumulating the sum of v_l and v_r values. Let

$$\alpha = \frac{\sum v_l}{\sum v_r}.$$

   Update the scaling values by

$$s_l' = s_l + s_r \frac{1-\alpha}{2}$$

$$s_r' = s_r - s_r \frac{1-\alpha}{2}$$

4. Move the vehicle clockwise around a square making four right-angle turns. The resulting orientation should be 2π. Let $\beta = \frac{\theta}{2\pi}$ and adjust W_r as $W_r' = \beta W_r$.

5. Similarly move the vehicle counterclockwise around a square. Let $\beta = \frac{\theta}{-2\pi}$ and adjust W_l as $W_l' = \beta W_l$.

6. Repeat steps 3–5 until the adjustments are smaller than the statistical variation.

5. EXPERIMENTAL RESULTS

In preparation for experimental trials, the camera was positioned as accurately as we were able. We then followed the calibration procedure of Section 4, resulting in the calibration parameters shown in Table 1. These values suggest that the orientation of the camera about the axis of travel is quite accurate. However, values for W_l and W_r show more discrepancy than we would have expected. The camera orientation in the axis normal to the ground was set with respect to the frame of the vehicle. The particular vehicle involved had been damaged, however. We suspect that the difference between the values of W_l and W_r is due to this damage. In particular, it would appear that the frame is not square to the direction of travel when moving in a straight line.

[1]In the experiments reported here, we ignore the symmetric effect of foreshortening when the camera is perfectly oriented.

Table 1: Calibration Parameters

Parameter	Value
s_l	0.002602
s_r	0.002598
W_l	0.5444
W_r	0.5008

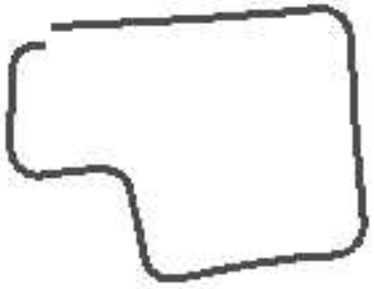

Figure 4: Typical Trial Path

The experimental trials covered a path approximately 85m in length going through the hallways of our offices. The path included five right-hand turns and one left-hand turn. Thus, not counting small course adjustments, the total $\Delta\theta = 3\pi$ while the net $\Delta\theta = 2\pi$. During the test runs, we kept the vehicle speed as close to 0.30m/s as possible. Speeds were generally observed to fall between 0.25m/s and 0.35m/s. Each of the turns had a turning radius of approximately 2m. Figure 4 shows a path followed during Trial 4. Test runs began at the endpoint above the gap in the figure and proceeded in a clockwise fashion along the path. The vehicle was returned to its starting point so that the reported final position represents the accumulated error during the test run. Results for ten trials are summarized in Table 2. The value shown

Table 2: Experimental Trials

Trial	d	θ	Error	(x, y)	Error
1	85.3	6.51	2.4%	$(-0.84, 1.01)$	1.5%
2	85.2	6.17	1.2%	$(-2.61, 0.28)$	3.1%
3	85.1	6.42	1.5%	$(-0.50, 0.73)$	1.0%
4	85.1	6.38	1.1%	$(-0.52, -1.57)$	1.9%
5	85.0	5.99	3.1%	$(-4.05, -1.67)$	5.1%
6	84.8	6.31	0.3%	$(-2.67, 0.95)$	3.3%
7	85.1	6.31	0.3%	$(-2.36, 0.06)$	2.8%
8	84.9	6.26	0.2%	$(-1.64, 0.54)$	2.0%
9	85.2	6.19	1.0%	$(-0.66, 1.22)$	1.6%
10	84.9	6.09	2.0%	$(-1.44, -0.81)$	1.9%
Mean			1.3%		2.4%
	85.1	6.26	0.2%	$(-1.73, 0.07)$	2.0%

in the second column is the total odometry $d = \sum v\Delta t$. The final orientation θ is given in the third column and the error in orientation is shown in the fourth column. This error is expressed as a percentage of the total change in orientation, i.e. $|\frac{\theta - 2\pi}{3\pi}|$. Similarly, the final position is reported in the fifth column. The sixth column shows the error as a percentage of d.

The last two rows of the table summarize the results. In the next to last row, we present the means of the error columns. These represent the amount of error that can be expected on average for a path that is about 85 meters long with six right-angle turns. In the last row, we present the means of the final odometry values and compute the errors on those means. We must be careful in interpreting these data. The 0.2% error in the average orientation represents the orientation error we would expect if we ran the ten trial laps consecutively. This figure suggests that the orientation error is random, with a mean close to 0. On the other hand, the average (x, y) position, with its 2.0% error, does not represent the error we would expect after ten consecutive laps. Instead it suggests the error that we would get if we used an orientation reference such as a magnetic compass to correct the orientation error once per lap.

6. PERFORMANCE ISSUES

As reported in the previous section, we moved at approximates 0.30m/s during testing. This speed is well below the maximum speed for this technique. Our calibration results show that each camera pixel covers approximately 0.0026m. In the calculation of image shift, we interpret shifts greater than one-half of the image to be negative shifts. Using 100×100 pixel images, this limits the motion to no more than 50 pixels, or 0.13m, per image. Since we capture approximately 10 images per second, we can travel no faster than 1.3m/s, or 4.68k/h. Allowing a factor of two margin for error, we could expect the technique to work up to 0.65m/s or 2.43k/h. To support faster speeds, we would need to increase the frame rate, increase the number of meters per pixel or increase the size of each frame.

The computational load imposed by this technique is primarily dependent on the size of the images. If we use an $n \times n$ image, then each image has n^2 pixels. Because the FFT is an $O(n \log n)$ computation and since the FFTs dominate the computation, the tracking algorithm has $O(n^2 \log n)$ complexity. While we haven't carried out extensive studies on computational load, we have observed that using 100×100 pixel images we used approximately 20% of the CPU resources on a Compaq laptop computer using a 2.4GHz Pentium IV processor running Linux. Similarly, in earlier experiments with two cameras and 240×240 pixel images, the CPU was loaded to the point where images had to be dropped at times. While such a load is significantly higher than that of other localization techniques, it is still well within the capabilities of modern hardware.

It would be appropriate to compare the accuracy of image-based localization with other techniques. We have not yet done so experimentally. However, we can draw some rough comparisons to other reported results. In [1], the most applicable comparison is to the reported characteristics of the ENV-O5S Gyrostar inertial navigation system. They report a positional drift of between 1 and 8cm/s and an orientation drift of 0.25°/s. This orientation drift over our 85m course at 0.3m/s would result in an error of 1.24rad or 13%. However, Borenstein, et al also report on a drift compensation that reduces the error rate to 0.0125°/s which would result in a 0.7% error. It is difficult to directly determine the implications for the positional drift on our circuit. However, over the approximately 283 seconds that it takes to complete the course, the inertial navigation system would accumulate between 2.83m and 22.64m of drift which could represent as much as 3.3% to 26.6% positional error.

7. CONCLUSION

We have presented a method for tracking a vehicle's motion in order to augment absolute positioning systems. The technique can be used to fill in the gaps during GPS outages and during times when magnetic compasses do not provide reliable data. Initial results look promising.

Several additional avenues for further research are apparent. First, we would like to investigate the sources of error with an eye toward improving the system's accuracy and toward creating a more automatic calibration process. At this point, there is little data on how the error relates to vehicle speed and to turning radius. A second area of development aims at advancing the motion inference technique to support full-sized road vehicles. The camera used in these experiments substantially limits the speed at which the vehicle can travel. The third area of further research is combining this vision-based technique with other localization techniques. Finally, we seek to adapt the technique to allow motion inference from images taken from a forward-looking camera. This would allow us to piggy-back on infrastructure already in place for other purposes (e.g. obstacle detection and landmark identification). Imaging in a forward direction would also place the imaged ground farther from the camera resulting in a larger meters per pixel scaling factor which, in turn, would allow for higher velocities.

8. REFERENCES

[1] J. Borenstein, H. Everett, L. Feng, and D. Wehe. Mobile robot positioning: Sensors and techniques. *Journal of Robotic Systems*, 14(4):231–249, 1997.

[2] A. Fitch, A. Kadyrov, W. Christmas, and K. J. Orientation correlation. In P. Rosin and D. Marshall, editors, *British Machine Vision Conference*, volume 1, pages 133–142, 2002.

[3] M. Frigo and S. G. Johnson. FFTW: An adaptive software architecture for the FFT. In *Proc. 1998 IEEE Intl. Conf. Acoustics Speech and Signal Processing*, volume 3, pages 1381–1384. IEEE, 1998.

[4] R. Negenborn. Robot localization and Kalman filters. Master's thesis, Utrecht University, 2003.

[5] J. Perš and S. Kovačič. Nonparametric, model-based radial lens distortion correction using tilted camera assumption. In H. Wildenauer and W. Kropatsch, editors, *Proceedings of the Computer Vision Winter Workshop 2002, Bad Aussee, Austria*, pages 286–295, 2002.