IMU Aided 3D Visual Odometry for Car-Like Vehicles

Navid Nourani-Vatani\textsuperscript{1,2}, Jonathan Roberts\textsuperscript{1}, and Mandyam V. Srinivasan\textsuperscript{2}

\textsuperscript{1}Autonomous Systems Laboratory
CSIRO ICT Centre
PO Box 883, Kenmore, QLD 4069, Australia
\{Navid.Nourani,Jonathan.Roberts\}@csiro.au

\textsuperscript{2}Queensland Brain Institute
University of Queensland
QBI Building 79, Sct. Lucia, QLD 4072, Australia
\{N.Nourani,M.Srinivasan\}@uq.edu.au

Abstract

We present a method for calculating odometry in three-dimensions for car-like ground vehicles with an Ackerman-like steering model. In our approach we use the information from a single camera to derive the odometry in the plane and fuse it with roll and pitch information derived from an on-board IMU to extend to three-dimensions, thus providing odometric altitude as well as traditional x and y translation. We have mounted the odometry module on a standard Toyota Prado SUV and present results from a car-park environment as well as from an off-road track.

1 Introduction

Traditionally, when dealing with mobile robots, researchers use odometry as a primary source of localisation information. The majority of mobile robots to date have some kind of wheel or drive shaft encoder and some type of steering sensor from which to derive the distance travelled. Even in the absence of such sensors, odometry information can be estimated from the commanded motions of the vehicle. The use of along-track (drive shaft or wheel encoders) combined with steering information leads researchers to assume that the driving surface is roughly planar and hence an x and y translation estimate (the odometry) may be calculated. However, there are many applications for mobile robots where the driving surface is non-planar and in fact quite rough and undulating. Traditional two-dimensional (2D) odometry may be useful and in fact, adequate, in many of these applications, but nevertheless there are applications where simple 2D odometry is not good enough and a more realistic three-dimensional (3D) odometry system is required - where altitude is the third element estimated along with the x and y translations. Obvious applications for 3D odometry are Mars/Moon rover type scenarios and general off-road/cross-country mobile robotics. Less obvious applications lie in the area of Simultaneous Localization and Mapping (SLAM) in standard road driving scenarios. [Bosse and Zlot, 2008] showed that the lack of altitude estimate resulted in non-ideal performance of an urban SLAM algorithm. Even in city streets, it is common for one carriageway to be slightly higher/lower than the oncoming one and hence when driving in opposite directions on such stretches of roadway, any environmental scanning systems (lasers, radar or vision systems) will be collecting data at different heights (due to the different altitude of the vehicle). Other pathological cases for 2D odometry are the bridge scenario, where one road crosses over another, and the multi-story carpark scenario, where multiple ‘roads’ appear on top of one another and have common x and y translational components.

In this paper we describe an approach to derive two-dimensional odometry from a single camera and extend it to three-dimensions, thus providing odometric altitude as well as traditional x and y translation. The method has been developed for car-like vehicles (Figure 1), which use an Ackerman-like steering model [Borenstein et al.,

Figure 1: The test vehicle used for these tests is a standard Toyota Prado SUV.
This is the same vehicle as used by [Bosse and Zlot, 2008] and so our approach directly addresses their issues.

As mentioned previously, there has been little work reported on altitude estimation using odometry. Until recently, the main focus in this area was in Mars/Moon rover applications. An example of this is given in [Helmick et al., 2004], where they describe a visual odometry system, based on stereo vision, and developed for possible deployment on a future Mars rover. They show how, with the use of an Inertial Measurement Unit (IMU) and an Extended Kalman Filter (EKF), the planar world assumption can be removed and the full pose and position of the rover estimated. They also present graphic results of how poor an estimate is obtained when the planar assumption is used. Their work incorporates modelling of slip and achieves impressive results. Similarly, [Nister et al., 2005] use a vision-based odometer on an off-road all-terrain mobile robot. However, they use the stereo system alone and only use an IMU (which is combined with DGPS) as ground truth to compare the vision system’s performance. Unfortunately, the altitude results are not shown in [Nister et al., 2005], although estimates must have been available in their system.

More recently, SLAM has been a major focus of mobile robotics research. As before, SLAM researchers initially assumed a planar world and neglected the issues of altitude and pose changes of the mobile vehicle. More recently, this assumption is being removed and there are now examples of full six-dof SLAM. Of note is research reported in [Stasse et al., 2006] where stereo vision-based full 6-dof SLAM results are presented for a humanoid robot. In outdoor applications, [Zhou et al., 2007] report an Extended Information Filter-based SLAM system that is capable of 3D position estimation. However, only simulation results are presented.

3D SLAM algorithms using laser scanners are also now emerging. [Weingarten and Siegwart, 2005] describe such a system that uses an EKF and estimates altitude along with x and y translations.

In our approach we decouple the calculation of odometry in the plane, (x, y, θ), from the calculation of the altitude, z. We do so by calculating the vehicle’s 2D relative displacement using a single camera and combining it with the instantaneous vehicle roll and pitch estimate received from an IMU to determine the change in altitude. The result is global odometry - off the mythical planar world. As will be shown, the altitude calculation is decoupled from the vehicle displacement calculation. Therefore, this method can be applied to other two-dimensional odometry methods to extend them to three-dimensions.

In the following section we describe our methodology in depth. In § 3 we present and analyze the results from a series of experiments on our test vehicle. Finally, § 4 concludes and discusses the findings.

2 Methodology

In this section we describe a methodology for deriving a vehicle’s motion using a single camera. This method is designed for vehicles applying an Ackerman-like steering method [Borenstein et al., 1996] which is the most common steering model for car-like vehicles. The kinematic motion constraints introduced by this model allow us to make the following assumptions which simplifies the odometry calculations considerably: 1) the vehicle is car-like and its motion is constrained to Ackerman’s steering model; its motion is comprised of a forward translation and a rotation around the center-of-motion of the rear axle, 2) there is no side-ways translation so we ignore any side-ways slippage, and 3) the vehicle motion can be viewed as piece-wise straight motion. This more simplified vehicle motion allows for easier odometry calculation. Note that we do not make the assumption of a flat ground plane. On the contrary, we use the unevenness of the ground to calculate change in altitude.

The motion of the vehicle is derived from the observed image motion in the camera mounted at the front of the vehicle at a fixed height looking at the ground straight under it. The motion in the camera is determined by calculating the pixel displacement, ∆U and ∆V, in consecutive frames. This displacement is translated to the vehicle-relative motion, ∆x and ∆θ, as shown in Figure 2 and described later below.

After calculating the two-dimensional vehicle-relative motion, the vehicle roll and pitch, gamma and beta, measurements are used to estimate the altitude, z. The vehicle’s roll and pitch have to be determined by other means. Here we show how the roll and pitch angles derived from an on-board IMU can be incorporated to get a full 3D odometry calculation.

Next we explain the method used for calculation of the pixel displacement in the camera and from that the determination of the vehicle odometry.

2.1 Pixel Displacement

A means by which the displacement of two frames can be computed is by calculating the optical flow. Standard optical flow methods require features to stand out in the image, however we are faced with very smooth and almost textureless surfaces; see Figure 3. Results using OpenCV’s implementation of Harris corners and pyramidal Lukas-Kanade optical flow calculation [Intel, 2001] from the concrete and asphalt surfaces were very poor. This has also been reported by Campbell et all [Campbell et al., 2005]. Other feature trackers such as Canny edge detector [Intel, 2001] and the Speeded-Up Robust Features (SURF) [Bay et al., 2006] were also
implemented without any significant improvements.

[Srinivasan, 1994] used an interpolation method to calculate the optical flow where by the translation in a two-dimensional plane, as well as rotation about a previously unspecified axis perpendicular to the plane can be measured. Unfortunately, this method cannot deal with large displacement.

Another method is correlation matching, which can be applied to the entire frame or a small area of one frame can be found in a larger frame with a similar brightness pattern [Horn, 1986]. This method is also called template matching. Template matching has proven much more reliable and is able to produce precise displacements on all the surfaces we have tested: concrete, asphalt, gravel, and grass. We therefore use this method.

One shortcoming of the template matching is its deficiency in dealing with foreshortening. It is therefore expected that less accurate or wrong results can be produced on uneven surfaces where two consecutive frames have a slightly different perspective.

The processing time of the correlation matching method is highly dependent on the size of the image. Our experiments have shown that the processing time is decreased as the inverse of the size of the image. This is one important design parameter. Another parameter is the size of the correlation template. The size of the template determines the maximum shift possible before the area is shifted outside the image. We use the OpenCV implementation of template matching using CV_TM_CCOEFF_NORMED as the matching method [Intel, 2001]. For our range of image velocities, good results are obtained if the length of the template is one-third to one-fourth the height of the image.

2.2 Odometry Calculation

In order to calculate the odometry estimate, the pixel displacement observed by the camera must be translated into vehicle motion and combined with the IMU measurements. This translation is a five step process and is explained below. In the following X defines the vehicle pose, which is decomposed into a 3 × 1 translational matrix, t, and a 4 × 1 quaternion rotation vector, Q:

\[
X = \begin{bmatrix} t \\ Q \end{bmatrix}
\]

\[
t = \begin{bmatrix} x \\ y \\ z \end{bmatrix}
\]

\[
Q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix}
\]

Converting pixels to distance

Assuming constant known distance between the camera and the ground, d, the displacement in pixels can easily be converted to actual displacement, D_{\{C\}}. Using the pinhole camera model we can recover the projection of the pixels on the ground [Horn, 1986]:

\[
D_{\{C\}} = \begin{bmatrix} \Delta x_{\{C\}} \\ \Delta y_{\{C\}} \end{bmatrix} = \begin{bmatrix} \Delta U \\ \Delta V \end{bmatrix} \cdot \begin{bmatrix} 4 \\ 7 \end{bmatrix}
\]

where \(\Delta U\) and \(\Delta V\) are the pixel displacements and \(f\) is the focal length of the camera.

To calibrate our camera and retrieve the focal length, \(f\), we use a standard radial distortion model [Zhang, 1999]. The camera-ground-distance, d, is retrieved manually using a measurement tape and later determined through a test run. This step is described in § 3.
A transformation is then necessary to convert from a camera coordinate frame, C, to the vehicle bumper centered coordinate frame, B, D_{BC} = T \times D_{CcR}

**Displacement in the plane**

Dealing with an Ackerman-steered vehicle we can take advantage of the motion restrictions described earlier and simplify the calculation of the vehicle displacement. The displacement in the plane seen by the vehicle then becomes:

\[
\Delta x = \Delta x_{[B]} \\
\Delta y = 0 \\
\Delta \theta = \text{atan2} (\Delta y_{[B]}, x_{\text{cam}})
\]

where \(\Delta x_{[B]}\) and \(\Delta y_{[B]}\) are the displacement in the vehicle bumper frame, and \(x_{\text{cam}}\) is the mounting position of the camera along the x-axis of the vehicle, i.e. the rear wheel axle to bumper distance. Here we set \(\Delta y = 0\) as we assume no sideways motion. See Figure 2.

**Integration of Pitch and Roll**

To extend the displacement to three-dimensions it is necessary to estimate and include the roll and pitch, \(\gamma\) and \(\beta\), that the vehicle is experiencing. We do so by reading the instantaneous pitch and roll values from the on-board IMU. These values are used as the current robot state and are used directly in the rotation quaternion, \(Q_p\). Incrementing the vehicle state with these values will produce the displacement in 3D. (5) shows the calculation of the rotation quaternion using the IMU roll and pitch readings.

\[
Q_p = \begin{bmatrix} q_{1p} \\ q_{2p} \\ q_{3p} \\ q_{4p} \end{bmatrix} = \begin{bmatrix} \cos(\text{roll}/2) \cdot cPh \cdot sYh + sRh \cdot sPh \cdot sYh \\ sRh \cdot cPh \cdot cYh - cRh \cdot sPh \cdot sYh \\ cRh \cdot sPh \cdot cYh + sRh \cdot cPh \cdot sYh \\ sRh \cdot cPh \cdot sYh - sRh \cdot sPh \cdot cYh \end{bmatrix}
\]

where \(cRh = \cos(\text{roll}/2)\), \(sRh = \sin(\text{roll}/2)\), etc.

Notice, that the yaw, \(\alpha\), is not read from the IMU, but is kept from the odometry in-the-plane calculation described above. This estimation of the heading is a more precise and reliable estimate compared to the IMU yaw calculation. If a different sensor with even more reliable approximation of the heading was available that estimate could have been incorporated here.

**Increment vehicle displacement**

Finally, these incremental pose displacements are integrated to derive the global robot pose, \(X\):

\[
t_n = t_p + R_p \cdot t_i
\]

\[
Q_n = \begin{bmatrix} q_{1p} \cdot q_{1i} - q_{2p} \cdot q_{2i} - q_{3p} \cdot q_{3i} - q_{4p} \cdot q_{4i} \\ q_{1p} \cdot q_{2i} + q_{1i} \cdot q_{2p} + q_{3i} \cdot q_{4p} - q_{4i} \cdot q_{3p} \\ q_{1p} \cdot q_{3i} + q_{1i} \cdot q_{3p} + q_{4i} \cdot q_{2p} - q_{2i} \cdot q_{4p} \\ q_{1p} \cdot q_{4i} + q_{1i} \cdot q_{4p} + q_{2i} \cdot q_{3p} - q_{3i} \cdot q_{2p} \end{bmatrix}
\]

where subscripts \(n\), \(p\), and \(i\) denote the new, previous, and incremental poses, respectively, \(q_i\) denotes the \(i\)-th quaternion vector element, and \(R\) is the \(3 \times 3\) rotation matrix.

In the next section we describe a method for calculation of the performance of the odometer and analyze our findings.

### 3 Results

Our visual odometry system has been tested on a 20 tonne autonomous Hot Metal Carrier (HMC) forklift [Roberts et al., 2007] and a Toyota Prado SUV. We present results in the plane using the HMC in a future publication. Here, results in three-dimensions using the SUV test vehicle are presented.

The web cam used has a resolution of \(640 \times 480\) pixels and maximum frame rate of 30 fps. It has been mounted on the front bumper of the SUV looking straight at the ground beneath. The IMU, a MicroStrain 3DM-GX1, was also mounted on the bumper. The odometry calculation and data logging was done in real-time on a MacBook Santa Maria Intel Core 2 Duo 2.4 GHz with 2 GB RAM running Ubuntu Hardy Heron.

As ground truth an RTK-GPS with an estimated error of 2 cm is used. The GPS roaming antenna is mounted on the roof of the SUV straight over the rear axle. Figure 1 shows this setup on the SUV.

#### 3.1 System Calibration

To obtain the IMU offsets and the camera height over ground and any misalignment with the vehicle axis due to mounting, a calibration run is necessary. We do this by driving a loop over a relatively flat ground and calculating the system parameters and finding the offsets. The outcome from one of our calibration run is shown in Table 1.

Note that this method suffers from velocity scaling error detection and is only used to derive the mentioned parameters. This shortcoming is discussed further next.

#### 3.2 Odometry Performance Measurement

The systematic velocity error in odometry modules will increase with the distance from origin but decrease as we loop around and head back towards the origin [Kelly,
Table 1: System parameters retrieved from a calibration run.

<table>
<thead>
<tr>
<th>System Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cam X</td>
<td>3.77 m</td>
</tr>
<tr>
<td>Cam Y</td>
<td>0.00 m</td>
</tr>
<tr>
<td>Cam Z</td>
<td>0.79 m</td>
</tr>
<tr>
<td>Cam $\theta$</td>
<td>-0.4 rad</td>
</tr>
<tr>
<td>IMU Roll</td>
<td>0.013 rad</td>
</tr>
<tr>
<td>IMU Pitch</td>
<td>0.099 rad</td>
</tr>
</tbody>
</table>

Table 2 shows the three-dimensional Euclidean distance error and standard deviation for the off-road path, $\sim 415$ m.

<table>
<thead>
<tr>
<th>Distance traveled</th>
<th># of cases</th>
<th>Position Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$\mu$</td>
</tr>
<tr>
<td>10 m</td>
<td>383</td>
<td>0.47 m (4.7%)</td>
</tr>
<tr>
<td>20 m</td>
<td>374</td>
<td>1.00 m (5.0%)</td>
</tr>
<tr>
<td>50 m</td>
<td>346</td>
<td>2.75 m (5.5%)</td>
</tr>
<tr>
<td>100 m</td>
<td>298</td>
<td>6.67 m (6.7%)</td>
</tr>
</tbody>
</table>

Table 3: Two-dimensional Euclidean distance error using three-dimensional or two-dimensional odometry calculation over 10 m and 100 m paths.

<table>
<thead>
<tr>
<th>Odometry Method</th>
<th>Position Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\mu$</td>
</tr>
<tr>
<td>3D - 10 m</td>
<td>0.47 m (4.7%)</td>
</tr>
<tr>
<td>2D - 10 m</td>
<td>0.47 m (4.7%)</td>
</tr>
<tr>
<td>3D - 100 m</td>
<td>6.79 m (6.8%)</td>
</tr>
<tr>
<td>2D - 100 m</td>
<td>7.06 m (7.1%)</td>
</tr>
</tbody>
</table>

2004]. This implies, that we should not measure the performance of odometry modules by driving in a loop and measuring the overall global error and calculating the Euclidean distance from origin. Instead, we suggest, the performance of an odometry module to be measured as the error induced as a function of the distance travelled.

We analyze the performance of our system by showing the three-dimensional Euclidean distance errors for 10 m, 20 m, 50 m, and 100 m intervals to show the tendencies of the error increment in the odometer as the distance driven is increased.

Next we analyze the performance of our odometry module using the method described here.

3.3 Analysis

We present results from two different environments: The first environment is a hilly off-road terrain containing gravel, large stones and high uncult grass. The test sequence here is approximately 415 m, contains a more than 10° accent and descent, 180 and 360° turns, two three-point turns, and loops back to the start position. This test demonstrates the algorithm’s robustness to uneven ground and large roll and pitch motion. The outcome of this test is also used to confirm the usability of the acquired 3D odometry.

The second test area is a car park with relatively flat smooth asphalt roads but with a 10 m change in altitude. Due to the featureless structure of the road, lighting is very important as incorrect exposure and sun-shade regions, especially the vehicle’s own shadow, disturbs the pixel displacement calculation from the correlation matching. The chosen path is approximately 770 m, contains two roundabouts, but does not loop back to the beginning. Since the path does not loop back to the beginning we can be sure that the systematic linear error propagation will not cancel out [Kelly, 2004].

Figure 4 shows the three-dimensional paths calculated using odometry together with the RTK-GPS paths. The path estimation for the off-road track is visualized in Video 1 which is accessible from http://nourani.dk/robotics/files/nourani08_visual_pdo.divx. Table 2 shows the three-dimensional Euclidean distance error between the odometer and the GPS for the off-road data and Figure 5 visualizes this. Due to the very noisy and unstable GPS readings during the car park test, it has not been possible to compare our odometry calculation to the GPS data. The data presented here are only useful for visual inspection.

Figure 6 shows the altitude calculation for the two test runs compared to the GPS. In the car park test the GPS altitude has discontinuities of more than 5 m, which shows clearly why the data cannot be used. In the hill test in the time intervals 240 – 290 sec and 470 – 510 sec the IMU returns higher pitch angle during the descent than encountered. The reason for this is unknown, but the outcome is a 2 x 1 m error in the altitude calculation, which we carry for the rest of the path. Otherwise the error is within $\pm0.25$ m.

One interesting result is the improvement in the $x$ and $y$ position calculation with and without the knowledge of vehicle pitch and roll angles and calculation of altitude. In Figure 7 we plot the x-y-coordinates for the odometry paths calculated using vision only and using vision + IMU. In Figure 7A the difference between the two calculation methods is minor. The 10 m and 100 m two-dimensional Euclidean distance errors are shown in Table 3, which confirm the minor improvement achieved by calculating the odometry in three-dimensions over short distances. As expected, the error grows more when the path does not loop back. This is shown clearly in Figure 7B where the two-dimensional odometry error is growing faster than the three-dimensional odometry error. Unfortunately, we cannot put numbers on the error growth for this test.
Figure 4: Comparison of the calculated 3D odometry (solid blue line) with the RTK-GPS (dashed red line). Please note that the z-axis is stretched to exaggerate the effect of the altitude for better comparison. A) Hilly off-road terrain and B) Car park environment.

Figure 5: Three-dimensional odometry error growth compared to the traveled distance.

Figure 6: Comparison of altitude calculated by odometer with the RTK-GPS. A) Hill run, B) Car park run.

4 Conclusion & Discussion

We have presented a method to derive a vehicle’s planar translation and rotation simultaneously from a monocular vision system. Using an on-board IMU, we have shown how any such two-dimensional odometry calculation can be expanded to also include vehicle altitude calculation. The main advantages of the proposed method are its simplicity, ease of implementation, and practicality. We have shown this by implementing it on our Hot Metal Carrier and a standard Toyota Prado SUV.

Analysis of the presented data from the SUV driving on an off-road track shows an average three-dimensional Euclidean distance error of approximately 5%. From the results it is also evident that the mean error increases linearly with the distance driven, Figure 5. Although we used an RTK-GPS unit with an estimated error of 2 cm, the signal was still so noisy on the test day due to multipathing that we were not able to use it as ground truth over large areas of our test site, which includes the majority of the car park area and the building canyons. We have therefore only been able to perform visual inspection of these test results. The odometry altitude calculation follows the GPS altitude very closely but is highly dependent on the precision of the roll and pitch values from the IMU. We also compared the x-y-positions calculated from the tree-dimensional odometry with the two-dimensional method. Our conclusion is that the gain in precision in x-y-coordinate starts to show only over longer distances. Depending on the application where the odometry module is being used the IMU sensor and
Figure 7: Difference in x-y-coordinates with and without the knowledge of vehicle pitch and roll. A) Hill run, B) Car park run.

the extra calculations necessary to estimate the vehicle altitude can therefore be saved. As mentioned in the introduction, though, in bridge scenarios, multi-story car parks and on roads with different heights, having knowledge of vehicle height will improve the performance of other localization methods.

It is commonplace for scientists to ignore the lighting issues they encounter in outdoor environments. We have addressed this matter several times and have also explained different methods for better exposure control in outdoor ungovernable lighting conditions [Nuske et al., 2006][Nuske et al., 2008][Nourani-Vatani and Roberts, 2007]. Yet we haven’t been able to control the exposure encountered during many of these tests. The issue here arises when the frame is divided between the harsh Queensland sun and the shade cast by trees, buildings, and the vehicle itself on the bright featureless concrete and asphalt roads. In this situation the lighting difference, i.e. the contrast, between the shade and sun exceeds the dynamic range of standard digital cameras of five f-stops. It is therefore not possible to expose the image to have details in both the dark and highlight of the image simultaneously. Therefore, if the correlation template area is in the sunny area of the image and the matched area is in the shade, or vice versa, the correlation will fail as one or the other area will have no feature due to the exposure. We cannot control the ambient light, but we can try to minimize its effect. Of course, if we had the time to stop and take two or more images with different exposures this would not be a problem, but that is not the case in real-time operation. A better placement of the camera would help. Repositioning the camera under the vehicle between the front wheels looks promising as this area is normally in full shade. Positioning the camera there requires it to be very close to the ground, which in turn requires much higher frame rate to keep up with the displacement, from (1). An optical imaging device that has the necessary frame rate, can perform image displacement, and is still very affordable is an optical mouse. We are currently producing lenses for an optical mouse and hope to show good results from that unit in the near future.

Acknowledgments

This work is funded jointly by the Australian government through the Endeavour IPRS scheme, the University of Queensland and the Queensland Brain Institute, the CSIRO ICT Centre and the ARC Centre of Excellence in Vision Science (grant CE0561903). The authors would like to thank their colleagues at the CSIRO ICT Centre’s Autonomous Systems Lab and the UQ QBI Biorobotics lab for their help and support.

References

[Campbell et al., 2005] J. Campbell, R. Sukthankar, I. Nourbakhsh, and A. Pahwa. A robust visual odom-


[Weingarten and Siegwart, 2005] J. Weingarten and R. Siegwart. EKF-based 3D SLAM for structured environment reconstruction. *Intelligent Robots and