Proceedings of the 2004 IEEE
Conference on Robotics, Automation and Mechatronics
Singapore, 1-3 December, 2004

Visual Odometry for an Outdoor Mobile Robot

David Fernandez and Andrew Price
Department of Electrical & Computer Systems Engineering
Monash University
Clayton, Victoria 3168, Australia
David.Fernandez@eng.monash.edu.au

***Abstract*— This paper presents an alternative solution to the problem of estimating the motion of an outdoor mobile robot through odometry. The solution consists of the use of vision and *pseudo*-optical flow techniques to make motion estimates of a camera attached to the robot chassis. The use of vision makes the system independent of locomotion (for example, the differences between a robot with wheels and one with legs) and is shown to be applicable to a variety of outdoor ground surfaces, such as tarmac, gravel, grass and dirt. Finally, the simplified calculations result in high processing speeds on moderate equipment, with experiments showing that high speed motion (approximately 300 degree per second rotations and 350 millimetre per second translations) can be reliably estimated.**

I. Introduction

The word odometry has its roots in the Greek *hodos* (journey) and *metron* (measure). That is, odometry is the measurement of a journey. In a localised and relative sense, the measurement of small-scale movements occurring over short time intervals is known as incremental odometry. The highly common use of wheels for locomotion has resulted in the equally common use of wheel encoders for such movement measurement. One of the biggest problems with the use of wheel encoders is the error associated with wheel slippage, that is, a difference between the rotation of a wheel and the actual motion of the platform. Robots utilising differential drives for steering in particular are prone to high degrees of slippage, and those employing skid-steering techniques even more so. For an outdoor mobile robot traversing grass, gravel, dirt and other highly non-uniform surfaces, slippage is even more a problem than it is for indoor robots that have the relative luxury of carpets or tiles. A more reliable and robust method of incremental odometry estimation will no doubt lead to more accurate and precise motion control.

Examples of possible solutions have included the use of radar [1], ABS sensors [2] and fusion with gyros [3]. The evaluation [4] and correction [5] of odometry errors has also been examined. A relative of the work presented here is that undertaken by Kelly [6], who's methods utilise an existing "map" of the environment (an image mosaic) and attempts to estimate a robot's position (that is, perform localisation). It does this by matching a current camera view to an area within the stored mosaic. Conversely, the work proposed here has no reliance on the existence of a pre-recorded map and instead aims to make incremental odometry estimations. For reliable operation, all that is required is that the scene viewed undergoes only relatively small changes from frame to frame, (a nominal time difference of 40 milliseconds for 25 frame-per-second PAL video), allowing an estimate of the motion between those frames to be extracted.

Nistér [7] presents an alternative system for performing visual odometry. It utilises a feature-tracking front-end (Harris corners are the features detected) that matches point features between frames and creates trajectories by matching features across pairs of frames. These trajectories are then used to estimate the camera pose and hence recover motion estimates. On tests using a stereo vision-equipped platform using DGPS (differential global positioning system) as ground truth, errors of between one and five percent over approximately 200 to 400 metres were reported for translations.

Bunschoten [8] describes a method for estimating the translation and rotation of an indoor office robot using an omnidirectional, catadioptric vision sensor. The omnidirectional image is first projected onto a vertically-aligned cylinder giving a panoramic image whose base is parallel with the floor. A Kanade-Lucas-Tomasi feature tracker is used to track features in this panoramic image across frames and subsequently platform motion is estimated. A method of global trajectory error correction has been suggested, whereby the first and last frame of a trajectory are used to estimate the motion between them and hence correct and close a loop. Clearly this has limited usefulness in a more general setting where points of loop closure are not known and hence knowing which additional frames to compare is difficult if not impossible.

Nagatani [9] proposed a similar idea to that presented here for an omnidirectional Mecanum-wheel robot. It was stated, however, to be inaccurate compared to traditional odometry and instead was used in combination with traditional forms of measurement. The work presented here aims to improve upon their results by providing more detailed odometry estimates (a full set of 2D translation and rotation, compared to only translations) that could ideally be used as a sole form of odometry, including cases where the platform would not allow other methods to be utilised (such as legs).

The rapid advance of computing technology has opened many doors in terms of practical data processing. Faster processors and higher memory capacities have been particularly kind to the area of image processing, where the usefulness of many ideas and algorithms has been limited by what a computer can do in what is considered a reasonable amount of time. The work presented here suggests a solution to the odometry problem that makes use of such computational

0-7803-8645-0/04/$20.00

advances, allowing it to be used with high performance on common equipment with average specifications.

The system uses a robot-mounted camera that faces down, with the image plane parallel to the ground surface. *Pseudo*-optical flow techniques are used to calculate motion vector estimates from the images captured and the rotation and two-dimensional translation of the robot can be measured. The visual nature of the system means that the only motion measured is that of the camera relative to the ground and hence the motion of the robot. Slippage of wheels is no longer an issue and indeed, the system can be used independently of any particular method of locomotion.

II. System Overview

The visual odometry system being developed uses *pseudo*-optical flow techniques to estimate movement of the camera relative to the ground surface being observed. The idea of using optical flow for positioning is not new (see [9] for example), but the amount of information garnered has in the past been limited (typically to translation estimates only). The system presented here aims to provide a complete set of odometry estimates (two-dimensional translation and rotation) in real-time using standard and commonly available equipment.

Motion estimation is performed at a number of discrete points on the captured image allowing the calculation of both a two-dimensional translational motion vector estimate and an angle of rotation (measured about an axis perpendicular to the image plane). The term *pseudo*-optical flow is used because, while the central idea (the estimation of motion from a sequence of images) is the same as that used in more rigorous optical flow calculations (for example, those involving image derivatives), the methods employed are highly simplified in order to achieve acceptable, real-time performance on average desktop equipment.

The resultant rotation and translation measurements are calculated based upon the assumption that the robot (and hence the points in the image being considered) has undergone a constrained affine transformation consisting of a rotation about the camera's optical axis followed by a translation in a plane parallel to the image plane. In the mathematics presented, the coordinate system used is centred on the image plane with x and y axes extending parallel to the plane and z extending perpendicular to it (and hence parallel to the optical axis).

The rotation of a point by angle θ about the z-axis can be represented as

$$x' = x.cos\theta - y.sin\theta$$
$$y' = x.sin\theta + y.cos\theta$$

and a translation by (dx, dy) as

$$x' = x + dx$$
$$y' = y + dy$$

Combined these give

$$x' = x.cos\theta - y.sin\theta + dx$$
$$y' = x.sin\theta + y.cos\theta + dy$$

More generally the transformation can be represented as

$$p' = \Phi p \tag{1}$$

where

$$p' = [x', y', z', w']^T$$

is the generalised position of the point in the current frame (i.e. the transformed point),

$$\Phi = \begin{pmatrix} cos\theta & -sin\theta & 0 & dx \\ sin\theta & cos\theta & 0 & dy \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

is the transform matrix for a rotation of θ about the z-axis followed by a translation of (dx, dy) in the image plane and

$$p = [x, y, z, w]^T$$

(with $w = 1$ for unit-scaled translations) is the position of the point in the original (previous) frame.

The transform of two such points, $p_1 = [x_1, y_1, 0, 1]^T$ and $p_2 = [x_2, y_2, 0, 1]^T$ can be represented as follows:

$$p'_c = P_c\phi \tag{2}$$

where

$$p'_c = [x'_1, y'_1, x'_2, y'_2,]^T$$

is a vector containing the combined relevant p'_1 and p'_2 coordinates,

$$P_c = \begin{pmatrix} x_1 & -y_1 & 1 & 0 \\ y_1 & x_1 & 0 & 1 \\ x_2 & -y_2 & 1 & 0 \\ y_2 & x_2 & 0 & 1 \end{pmatrix}$$

is the matrix containing the relevant coordinates of p_1 and p_2, and

$$\phi = [cos\theta, sin\theta, dx, dy]^T$$

is a vector of the unknown transform parameters (θ can simply be calculated from $\theta = Tan^{-1}(\frac{sin\theta}{cos\theta})$).

With p' and p (and hence p'_c and P_c) known, 2 can be solved for ϕ and the rotation angle and translation motion vector can be found. Using these results, the overall rotation and translation can be found from a discrete number of p, p' pairs taking two at a time and averaging the resultant θs and (dx, dy)s.

III. Software

The software system can be split into two distinct components: image capture and processing for odometry. The capture system was developed for a larger project with high-performance and modular usage being key goals, the result of which is a self-contained, high-speed image capture black-box module. Calculation of motion vector estimates use a highly simplified, texture-matching technique that simply aims to find out where a small patch of the previous image (the *target patch*) appears in the current image within a specified *search window*. This is performed at multiple locations on the

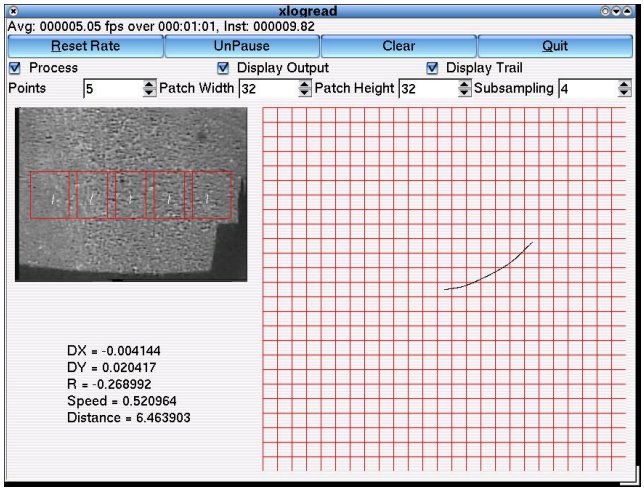

Fig. 1. Software screen capture, showing image, target patches and a traced path

image allowing rotation as well as translation to be estimated. Additionally, the matching process uses sub-sampling to further reduce the overall number of calculations. (Section III-B describes this process in more detail.)

Figure 1 shows a screen capture of the software system's display. The left side contains an input image (top) and numerical results of motion estimation (below). The image has borders drawn around the target and matched patches. To the right a simple trace of the path taken is shown over a grid. As shown by the controls at the top, the number of estimation points (target patches), their width and height and the level of sub-sampling can all be dynamically altered. While the search window can also be modified, it is currently set to be simply twice as wide and twice as high as the target patch. The positions of the target patches are automatically set by simply making them evenly spaced across the image.

A. Multi-threaded Image Capture

As stated previously, the image capturing module was developed as part of a larger robotic vision system that utilised multiple cameras. It was found during its development that unavoidable delays associated with a linear capture, wait-for-frame, capture, etc. approach were a significant cause of performance degradation. The lack of synchronisation across the multiple frame-grabbers gave rise to delays between capturing the image from one card and the next. The solution to this was the development of a threaded capture system that took advantage of the task-sharing abilities of the operating system.

A child thread is spawned for each active card, and each such child sits in its own capture-wait loop. A new frame is obtained when one is ready and, through the operating system's process scheduling, processor time is surrendered while it waits for a new frame. When a new frame is captured it is timestamped, a flag is set indicating that a new frame is available from that card, and the old frame data is replaced by the new image. A parent thread creates and manages the children and serves as the interface to the whole image capturing module. When a request is received for new images the parent simply copies frames from each of the children into its own buffer and resets the appropriate child's new-image flag. Semaphores are used for synchronisation and to provide clean termination of the children. Only new frames (as indicated by each child's flag) are actually copied, eliminating superfluous and potentially costly operations. The children's replacement of old frames with new ensures that when requests for images are made, the most recently captured images are provided.

B. Pseudo Optical Flow

Optical flow is a term used to describe the analysis of an image or images in order to estimate the motion of the scene viewed. *Pseudo* optical flow is a term used here to describe a simplified (and hence less detailed) analysis used to extract a small subset of motion information from a pair of images. The motion estimate data obtained is not as detailed or extensive as that which would be obtained were a traditional and rigorous optical flow calculation to be used, but it is sufficient for the task.

The method used is that of texture or pattern matching. A small sub-image (target patch) from one image is sought within a given search window in a second image. The position of the target patch in the second image, relative to its position in the initial image, is then taken to be the displacement of the image at the target patch's centre. While traditional optical flow would make use of a correlation calculation to find matching patches, pseudo-optical flow uses a *normalised sum of absolute differences* calculation. For two patches, P_A and P_B each of width W and height H pixels and a pixel value in the range $[0, p_{MAX}]$, the normalised sum of absolute differences (SAD) is calculated as follows:

$$SAD = \frac{1}{W.H.p_{MAX}} \sum_{j=0}^{H} \sum_{i=0}^{W} |P_A[j,i] - P_B[j,i]| \quad (3)$$

The result is a value between zero and one, zero corresponding to an exact match between target and potential, one corresponding to the highest possible difference between patches. The important difference between a SAD and a correlation calculation is the minimal use of costly multiplications, a factor of particular importance when aiming to achieve real-time performance. Additionally, the SAD calculation (in an un-normalised form) could be implemented on a much simpler processing device (such as a microcontroller without extended mathematics abilities) making it potentially more widely applicable.

In order to reduce errors and increase accuracy, a weighted averaging is used to calculate the final (dx, dy, θ) estimate. Each component set of (dx, dy, θ) (calculated from equation 2 for a (p_1, p_1'), (p_2, p_2') pair) is weighted by a (1-SAD) factor as a way of suppressing poor matches (that is, matches with high SADs) that would otherwise disturb the results.

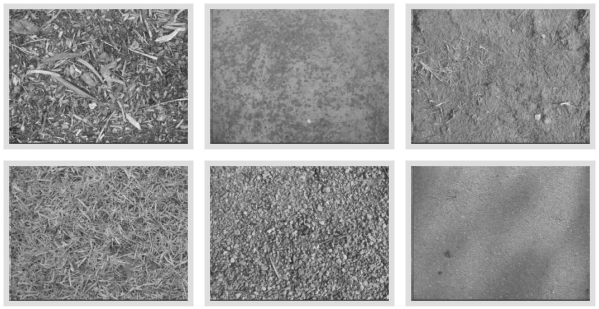

Fig. 2. A sample of the test images used

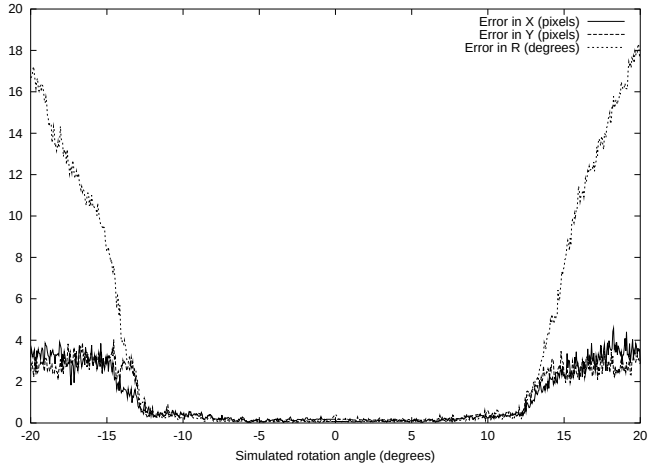

Fig. 3. Rotation Simulation: Errors

IV. Experimental Results and Discussion

A. Simulation

Simulations have been performed to test the implemented algorithms under controlled conditions. Testing consisted of taking a captured image, performing a translation/rotation transformation on the image and using the original and transformed images as inputs to the odometry estimation routine. The source image is captured at higher-than-normal dimensions before being rotated and translated by known amounts. Finally, the source and transformed images are both cropped down to standard dimensions. These two steps - large-size capture and final cropping - are done to ensure edge-effect distortions that result from the transformation do not appear in the transformed image being used.

To get a true idea of the performance, a sequence of fourteen input images were used, consisting of images taken of a variety of ground types, including concrete, tarmac, mud/dirt, gravel, grass and combinations. A selection of the images used is presented in Figure 2. Transformations were applied to all images and the results shown here are the averages over the fourteen images.

Figures 3, 4 and 5 show results of test-runs that investigated rotations over a range of ± 20 degrees with 0.05 degree increments, Y-translations over ± 30 pixels with 0.5 pixel increments and X-translations over ± 30 pixels with 0.5

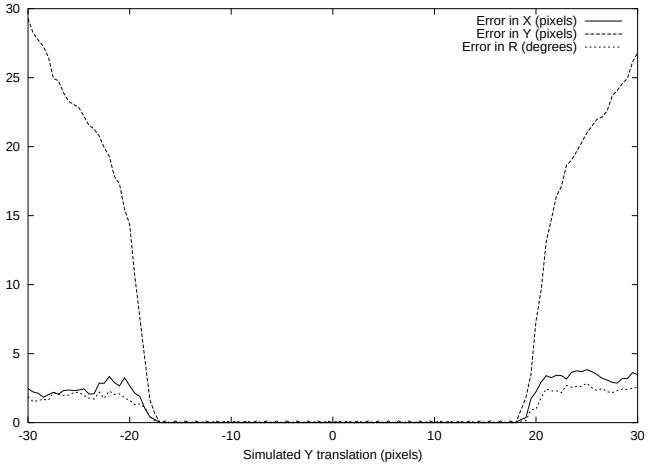

Fig. 4. Y-Translation Simulation: Errors

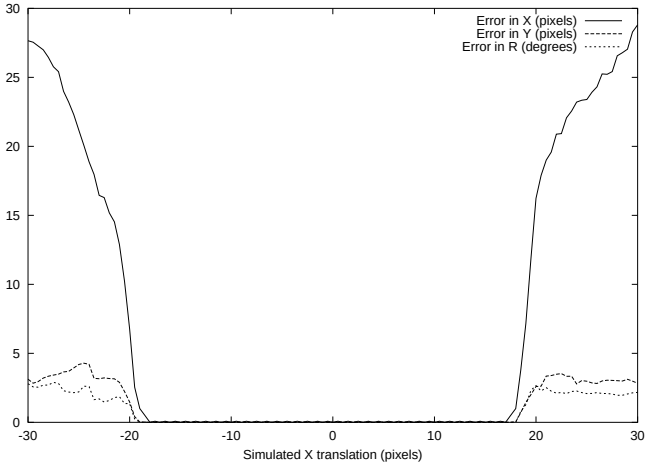

Fig. 5. X-Translation Simulation: Errors

Simulated Rotation		
	Mean absolute error	Standard Deviation
R (degrees)	0.15771	0.11221
Y (pixels)	0.22469	0.15588
X (pixels)	0.20764	0.13196
Input ranges: R = $[-12.35\,^\circ, 12.00\,^\circ]$, Y = 0, X = 0		

Simulated Y-Translation		
	Mean absolute error	Standard Deviation
R (degrees)	0.0031212	0.0031943
Y (pixels)	0.056777	0.058106
X (pixels)	0.0	0.0
Input ranges: R = $0\,^\circ$, Y = [-17, 15], X = 0		

Simulated X-Translation		
	Mean absolute error	Standard Deviation
R (degrees)	0.020898	0.021387
Y (pixels)	0.0	0.0
X (pixels)	0.039354	0.040275
Input ranges: R = $0\,^\circ$, Y = 0, X = [-18, 14]		

TABLE I
Simulation Results: Stable Regions

pixel increments respectively. The resultant absolute errors averaged over the fourteen test images are shown for each of the estimated X, Y and R values. For these tests, eight points of motion estimation were used, each utilising an 18×18 pixel target region sub-sampled by a factor of 2, with searching performed over a 36×36 pixel window. The average processing rate (on a Pentium 3, 800MHz with 128MB of RAM and with minimal code optimisation) was approximately 41 frames per second.

The breakpoints for all three measurements can be seen quite clearly as can the level of degradation (that is, the high errors) outside the stable region. Table I gives a more detailed outline of the performance within the stable regions. As intuitively anticipated, errors associated with pure translations are particularly low (the highest average being approximately 0.06 pixels for Y-translations) and the usable ranges are all at least ± 14 pixels. The results of rotations tests are particularly pleasing. Over the highly useful range of $\pm 12\,^{\circ}$, the average rotational error is less than one fifth of a degree, with a standard deviation of approximately one tenth of a degree. The corresponding translational errors are also very low, less than a quarter of a pixel.

When coupled with the realisation that these estimations are being made between frames (and are able to be made at a high rate, well over the anticipated input rate of 25 fps), it can be seen that the performance allows relatively high speed motion to be estimated with high accuracy. For example, rotational rates of over 300 degrees per second can be estimated assuming a standard PAL input frame-rate of 25 frames per second. A reasonable camera mounting height of about 30 centimetres off the ground would result in a linear mapping of about 1 millimetre per pixel, giving a maximum reliable speed measurement of $\pm$ 0.35 metres per second.

It should be noted that these stated maximums are for isolated rotations or translations. Realistic movement that combines rotations and translations will reduce these maximums but, given that the maximum speeds are already quite high, slight reductions to accommodate combinations of them will still leave a highly useful range of reliable estimation, especially given that for most tasks, a robot would not be expected to be travelling and turning at high rates simultaneously.

B. Field-Tests

Field-testing has initially consisted of the collection of data over a number of surfaces (including gravel but predominantly tarmac and grass) and post-collection processing. The data captured is simply uncompressed (raw) image frames as captured by the camera at the desired resolution of 320 x 240 pixels. These image sequences are then fed through the odometry routines and the resultant motion is plotted on-screen. The combination of raw-capture and post-processing allows the odometry software and its parameters to be modified as necessary with the results of modifications able to be seen rapidly, without the need to repeat potentially time-consuming field-tests immediately. As improvements are made new data can be collected to further test the system.

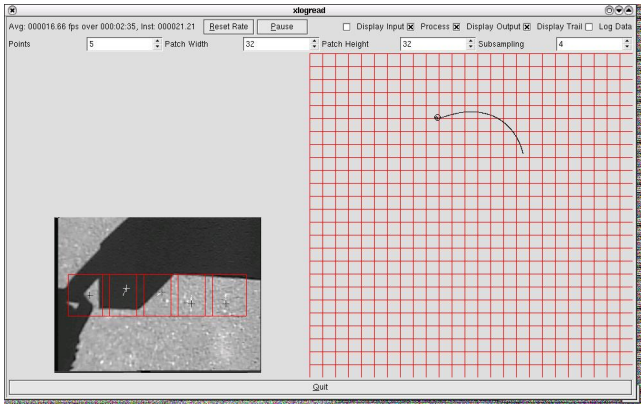

Fig. 6. Odometry field-tests: The lighting problem

To date, field-tests have consisted of three simple test variants. The first is a test of texture-matching stability over different surfaces. This involves placing the platform on a variety of surfaces and observing the behaviour when stationary and when undergoing simple movements. The second test involves moving the platform through simple, predetermined paths (straight lines, gentle, smooth curves and U-turns for example) and visually observing the results of motion-estimation. This is to qualitatively confirm that the motions estimated are consistent with the paths travelled. The third test uses simple paths with known dimensions (thus-far only straight-line segments have been used) to provide quantitative verification.

These tests have brought an important issue to light. This concerns the stability of the system under varying lighting conditions. In particular, heavy shadows cast under strong sunlight can be troublesome. The camera's automatic gain control means that both (reasonably) consistent bright lighting and dark shadow can be tolerated individually, but a mix of the two (small patches of dark shadow on an otherwise bright image or small patches of brightness on an otherwise dark image), can result in a large number of false patch-matches where the image is either under-exposed or saturated. The effects are most significant over relatively smooth surfaces that exhibit low texture detail. Figure 6 is a screen capture that shows the lighting issue. Here the heavy shadow cast by the platform itself causes erroneous patch-matching. The surface here is fairly smooth road tarmac.

Software-based methods (such as intensity normalisation) could be easily incorporated here to alleviate the lighting issue. The preferred solution, though, is the physical control of the lighting through the use of an opaque or semi-transparent skirt and an array of high-intensity LEDs to ensure a consistently bright area. This is preferred because a consistently bright image area will result in high camera shutter speeds (the camera's shutter speeds are variable but automatic), which will reduce (if not remove) motion blur that can result from slower shutter speeds combined with camera motion. This also has the useful side-effect that the system could be used in all manner of light, including bright sunlight, cloudy overcast conditions

and even the darkness of night. The use of infrared (IR) LEDs (combined with an IR-sensitive camera) over visible spectrum LEDs could further improve the usefulness of the system by making it undetectable to the naked eye (especially useful if used on robots operating in nighttime surveillance operations, for example).

V. Proposed Extensions and Future Work

Dynamic control of the system, making it able to adapt to environmental conditions (notably ambient light) is highly desirable. One proposed extension of particular interest is the use of a dynamic lighting system whereby the intensity of a lighting grid is adjusted to compensate for varying ambient light. This can be achieved, for example, by the introduction of a digital potentiometer into the LED circuit, the resistance of which is regulated by the value of the captured image's average intensity.

The idea of using motion vector history to predict the current vector (primarily to provide a search window that is more useful than the current "all-possible-directions" one) is also being considered.

Steps could also be taken towards creating a complete and self-contained, high-performance odometry unit by using a high-speed (that is, high frame-rate) camera linked to a dedicated embedded microprocessor.

Given the dependence of the measurable vector's range and resolution upon the size and coarseness of both the target patch and search window, a modification that uses the current vector's magnitude to adapt these parameters (for example, a coarse patch and a wide window at high speeds and a finer patch and smaller window at low speeds) could be of great use in increasing the range of conditions under which the system is operable.

VI. Conclusions

The algorithms used within the system have been shown through simulation to be highly accurate for the intended purpose of robot motion estimation. The estimated angle of rotation (the component most able to degrade usefulness if erroneous) displayed an average error of less than 0.2 degrees over a ± 12 degree range with a standard deviation of less than 0.12 degrees. Translation estimations are similarly reliable, with average errors of no more than about 0.06 pixels and standard deviations below 0.06 pixels over a ± 14 pixel range in both X and Y axes. While real-world values will depend on the exact camera location, at a height of about 30 centimetres off the ground, a 1 millimetre per pixel mapping is possible. Both simulations and field tests of the system have shown it to be quite robust with regards to terrain. Grass, gravel, concrete, tarmac and dirt were all handled consistently. A simplification of the mathematics involved has allowed high processing rates, making it a valid alternative for platforms moving at a moderate pace (up to 300 degrees per second rotational rates and 0.35 metres per second translations with a camera height of 30 centimetres). The problem associated with inconsistent image lighting has been highlighted and solutions involving the physical control of scene illumination - namely the use of a lighting array and a semi-transparent skirt - have been suggested for further investigation.

References

[1] R. Rogers, "Improved heading using dual speed sensors for angular rate and odometry in land navigation," in *IEEE 1998 Position Location and Navigation Symposium*, Apr 1998, pp. 20–23. [Online]. Available: http://ieeexplore.ieee.org/iel4/5490/14769/00670039.pdf

[2] P. Bonnifait, P. Bouron, P. Crubille, and D. Meizel, "Data fusion of four abs sensors and gps for an enhanced localization of car-like vehicles," in *Proceedings of the 2001 ICRA. IEEE International Conference on Robotics and Automation*, 2001, pp. 1597–1602. [Online]. Available: http://ieeexplore.ieee.org/iel5/7423/20183/00932839.pdf

[3] J. Borenstein and L. Feng, "Gyrodometry: a new method for combining data from gyros and odometry in mobile robots," in *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, vol. 1, Apr 1996, pp. 22–28. [Online]. Available: http://ieeexplore.ieee.org/iel3/3678/10815/00503813.pdf

[4] A. Martinelli, "A possible strategy to evaluate the odometry error of a mobile robot," in *2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 4, 2001, pp. 1946–1951. [Online]. Available: http://ieeexplore.ieee.org/iel5/7677/21067/00976358.pdf

[5] J. Borenstein, "Experimental results from internal odometry error correction with the omnimate mobile robot," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 963–969, Dec 1998. [Online]. Available: http://ieeexplore.ieee.org/iel4/70/15903/00736779.pdf

[6] A. Kelly, "Contemporary feasibility of image mosaic based vehicle position estimation," in *Proceedings of the 1999 International Conference on Robotics and Applications*, Oct 1999.

[7] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2004, pp. 652–659. [Online]. Available: http://ieeexplore.ieee.org/iel5/9183/29133/01315094.pdf

[8] R. Bunschoten and B. Kröse, "Visual odometry from an omnidirectional vision system," in *Proceedings of the IEEE International Conference on Robotics and Automation, 2003*, vol. 1, September 2003, pp. 577–583. [Online]. Available: URL:http://ieeexplore.ieee.org/iel5/8794/27829/01241656.pdf

[9] K. Nagatani, S. Tachibana, M. Sofne, and Y. Tanaka, "Improvement of odometry for omnidirectional vehicle using optical flow information," in *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2000, pp. 468–473. [Online]. Available: http://ieeexplore.ieee.org/iel5/7177/19350/00894648.pdf