Optical Flow Algorithm for Velocity Estimation of Ground Vehicles: A Feasibility Study

Savan Chhaniyara, Pished Bunnun, Lakmal D. Seneviratne and Kaspar Althoefer
King's College London, Strand, London, WC2R 2LS
Department of Mechanical Engineering
savan.chhaniyara@kcl.ac.uk, pished.bunnun@nectec.or.th, k.althoefer@kcl.ac.uk,
lakmal.seneviratne@kcl.ac.uk

Abstract- This paper presents a novel velocity estimation method for all terrain ground vehicles. The technique is based on a camera that scans the ground and estimates the velocity by using an optical flow algorithm. The method is tested and validated for different types of terrains such as fine sand, coarse sand, gravel as well as a mixture of coarse sand and gravel. Measured velocities from precise encoders are compared with the velocities predicted by the optical flow algorithm, showing promising potential for implementation of the suggested approach in ground vehicles. Investigations have been carried out to determine the optimal feature window size and the influence of camera height on optical flow velocity estimates. Detailed laboratory experiments were carried out to validate the velocity estimation technique and results indicate the usefulness of the proposed method for velocity estimation of ground vehicles.

Index terms: optical flow, velocity estimation, visual odometry, optimization.

I. INTRODUCTION

It is important to estimate the velocity of autonomous robots within certain tolerances in order to achieve higher level motion and navigation tasks accurately. This is a very complex task in the presence of unstructured environments and often unpredictable vehicle behaviour such as wheel slip. One of the important parameters for ground vehicle slip estimation is vehicle velocity [1]. In this paper, a novel technique to estimate ground vehicle velocity is presented.

Traditionally, wheel odometry, GPS, DGPS and inertial sensors have been used to obtain a vehicle's speed and possibly its trajectory [2]. Despite the popularity and usefulness of above sensing techniques, they suffer from drift, low resolution or limited applicability. Wheel odometry performance degrades in presence of vehicle slip and skid. GPS and DGPS suffer from low resolution and low update rates and are thus unsuitable for highly dynamic mobile robotic applications. Whereas, inertial sensors are prone too high noise levels, especially, at low speeds and the accuracy of these sensors is affected if velocity and position estimates are required due to the needed single and double integration with respect to time [3-5].

Over the past decade, several researchers have made significant advances in utilising various vision and image processing techniques for navigation, obstacle avoidance and control of ground vehicles. Most researchers use image processing techniques for generating 3D space from 2D images, detecting obstacles or extracting motion from the acquired image sequences [6]–[11]. Many navigation algorithms that can compute a robot's path and can generate maps of unstructured environments relatively robustly have been developed in the recent past. Most notable are the advances in navigation methods based on SLAM [6], [12-13]. However, this paper focuses on velocity estimation which is applicable to slip estimation.

A numbers of approaches based on visual odometer have emerged recently [14-19]. Visual odometry is an image processing technique for incremental, online estimation of vehicle position and velocity from spatiotemporal image sequences. In this paper, a visual odometry based velocity estimation technique similar to that in [20-21] is presented using monocular camera configuration. Stereo visual odometry demonstrated by Jet propulsion laboratory's Mars rovers "Spirit" and "Opportunity" is one of the best examples. It has clearly demonstrated the potential of visual odometry systems for unmanned ground vehicles [22]. In [20], research on visual odometry for an outdoor mobile robot is proposed. Visual odometry is able to estimate both translational and rotational velocities based on a pseudo–optical flow technique. Based on simulation results, up to 300 degrees per second rotational and 350 millimetres per second translation velocities can be estimated using standard cameras.

In this paper, an accurate and reliable vehicle velocity estimation algorithm is presented for ground vehicles using the optical flow technique. This technique is based on a camera that scans the terrain and predicts the velocity of a vehicle accurately; the measured signals then can be used to estimate on-line vehicle slip parameters. It is proposed that the vehicle is equipped with a camera facing the ground and acquiring images of the surface the vehicle is manoeuvring over. Vehicle velocity is then calculated based on a camera model and the tracking of features between frames of the spatiotemporal image sequence. The proposed system is thoroughly tested for linear motion and validated under laboratory conditions for different terrain conditions such as fine sand, coarse sand, gravel and mixed coarse sand and gravel.

In the Section II, the camera velocity estimation algorithm is presented. Experimental results are presented in Section III and conclusions are given in Section IV.

II. OPTICAL FLOW AND VELOCITY ESTIMATION ALGORITHMS

A great deal of information can be extracted by recording time-varying image sequences using a moving camera. An image sequence (or video) is a series of 2-D images that are sequentially ordered with respect to time. Motion estimation is defined here as the estimation of the displacement and velocity of features in image frame with respect to the previous frame in a time sequence of 2D images. The proposed system layout for detecting the motion parameters for an all terrain unmanned ground vehicle is shown in the Figure 1. As shown, the camera is attached to the ground vehicle facing the ground. Ideally the plane of image is perpendicular to the terrain. Camera arrange parallel to the direction of motion (forward looking camera) can have better field of view then camera looking to the ground or at an angle. But in this arrangement post processing of an image can be difficult and it could require precise camera calibration. This is due to fact that features at distance moves different speed then features near to camera. As this arrangement has wide field of view it also requires numbers of iterations to track and sort suitable features in a particular image based on their location at horizon, near or far. Camera arranged perpendicular to direction of motion (downward looking camera) eliminates the drawbacks of forward looking camera but it can have limiting field of view. This arrangement also can pose restriction to placement of camera on particular robot or vehicle. As it has limiting field of view it should be fix fairly high from the ground in order to achieve sensing at higher speeds. The downward looking camera configuration is adopted here due to its simplicity as shown in Figure 1.

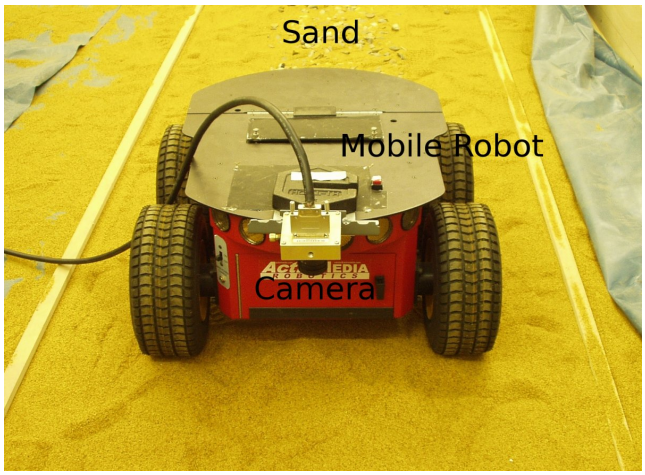

Figure 1. Visual Odometry Sensor/Camera Configuration

There are two phases for estimating the vehicle velocity. In the first phase, the optical flow algorithm is used to estimate an optical flow from an image pair. The optical flow is then used to estimate the vehicle velocity.

a. Optical flow algorithm

The optical flow algorithm determines the movement of brightness patterns which are assumed to move as the objects move in the sequence of images. The output of optical flow is obtained in the form of u_x and u_y component velocity as shown in Figure 2. Generally, we assume that the optical flow pattern will correspond to the motion field, although this is not always true in practice. Using some constrains such as geometrical (shape) comparison, comparison of grey levels and intensities, a feature found in one image can be usually matched with a feature in the next image. The main assumption for optical flow is that the appearance of a scene does not change significantly between frames, and that the scene moves smoothly from one image to the next. The flow is often locally uniform, however discontinuities may occur at scene boundaries. Other assumption is that the scene is made up of extended reasonably smooth surfaces, which move rigidly or at least distort smoothly when moving. The amount of motion between frames is small, and compared to the size of the image which means the optic flow vectors are small for pair of images. Horn [23] did pioneering work on the development of optical flow techniques based on computing spatiotemporal differences from an image sequence. Since then, many methods and algorithms for determining optical flow were developed like differential methods, region-based matching methods, energy based methods and phase-based methods [24].

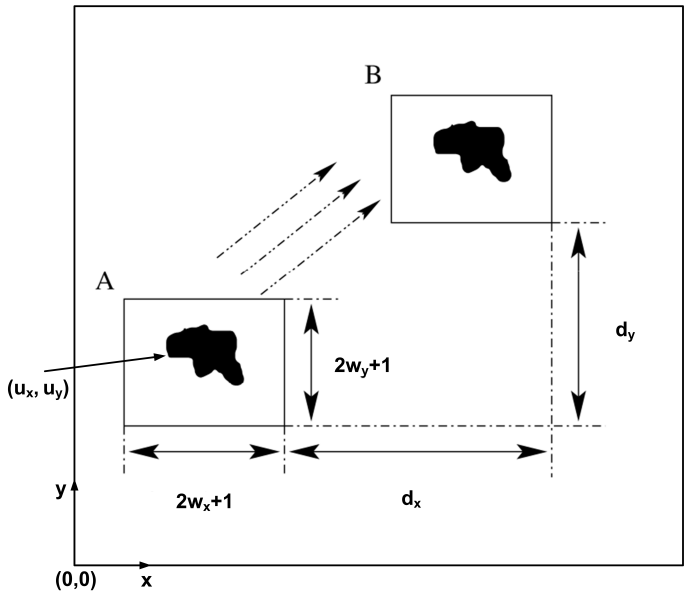

Figure 2. Optical flow features and flow field

The Kanade-Lucas optical flow algorithm is used because it is robust, accurate, insensitive to noise and non-uniform light intensity sources, and suitable for real-time computation [24-25].

In this method, let I_i be the greyscale image at time t_i and I_{i+1} be the greyscale image at time t_{i+1}. During this time interval, let the image be translated by distance $=(\Delta x, \Delta y)$. If A is a feature window in I_i and B be the same feature window in I_{i+1} as shown in Figures 2 and 4. Then the objective is to find d by minimising a residual function $\varepsilon(d)$ in (1):

$$\varepsilon(d) = \iint_w (I_i(p) - I_{i+1}(p+d))^2 \, dxdy, \quad (1)$$

where,

- $P_0 = (x, y)$ is the pixel coordinate of a generic image point as shown in Figure 4. The upper left corner pixel coordinate is $(0,0)$ and the lower right corner pixel coordinate is $(n_x - 1, n_y - 1)$, where n_x and n_y are the width and height of the image respectively.
- I_i and I_{i+1} are the greyscale values of the first image and the second image respectively.
- W is the feature window area, of size equal to $(2w_x + 1, 2w_y + 1)$.
- $d = (\Delta x, \Delta y)$ is the optical flow output or distance between features of two subsequent image frames.

In practice, the solution of minimising (1) can be achieved by using an iterative algorithm like the Newton-Raphson method. The algorithm is implemented as follows [26]:

STEP 1 Select a feature window to track in image $I_i(p)$.

STEP 2 Let $u = (ux, uy)$ be the centre of the feature window in image $I_i(p)$, Figure 2.

STEP 3 Window width and length, w_x and w_y, are user defined, with typical values of 2 to 7 pixels.

STEP 4 Let $d = (\Delta x, \Delta y)$, where $(\Delta x, \Delta y)$ is an initial guess.

STEP 5 Calculate $$G = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{u_y-w_y}^{u_y+w_y} g(p) g^T(p) \quad (2)$$

$$b = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{u_y-w_y}^{u_y+w_y} \left(I_i(p) - I_{i-1}(p+d) \right) g(p) \qquad (3)$$

$$\text{Where,} \qquad g(p) = \begin{bmatrix} \frac{\partial}{\partial x}(I_i(p) + I_{I+1}(p+d)) \\ \frac{\partial}{\partial y}(I_i(p) + I_{i+1}(p+d)) \end{bmatrix} \qquad (4)$$

STEP 6Solve linear equation $Gd_l = b$.

STEP 7Update $d = d + d_l$.

STEP 8Compute the following equation

$$\varepsilon(d) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} \left(I_{i+1}(p+p) - I_i(p) \right)^2 \qquad (5)$$

STEP 9If $\varepsilon(d) < \varepsilon$, where ε is a predefined threshold, go to 5.

STEP 10The optical flow is d .

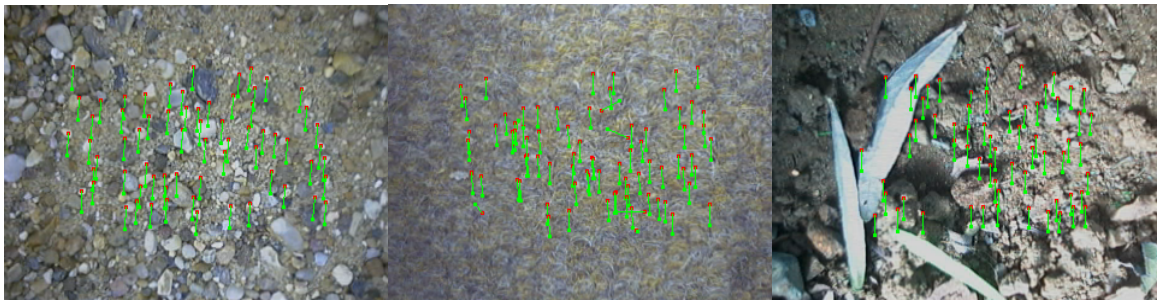

Figure 3. Processed Optical Flow Field Different Terrain Type

b. Camera model

A camera model giving the relationship between an object position and its pixel position is presented. In this study we have utilized a pinhole camera model to establish the calibration between points in 3D space to points in the image. In this model it is considered that rays of light enter the camera through a very tiny aperture. This imaging model captures accurately enough the geometry of the perspective projection [28]. Main assumptions in this camera model are that the centre of projection of the image coincides with the world coordinate and the optical axis (z axis) of the camera is perpendicular to the terrain surface. Moreover it is assumed that the image is placed is in front of the centre of projection as shown in Figure 4.

The pinhole model includes intrinsic characteristics, such as the image centre and the focal length, as well as the extrinsic characteristics providing the location of the centre and 3D orientation of the projective device.

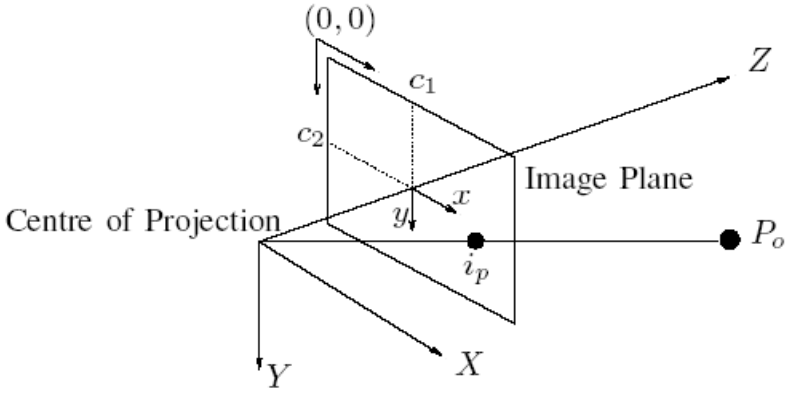

Figure 4. The Standard Coordinate System Camera

There are two sets of parameters. The first set, called extrinsic parameters are not constant and depend on the camera orientation with respect to the environment. The second set, called intrinsic or internal parameters, are camera dependent. Intrinsic parameters are the important parameters and are used to relate pixel position of the corresponding object in real space. The intrinsic parameters are determined by using the method described in [27]. Figure 4 shows the standard coordinate system of a camera. The intrinsic parameters are listed below.

1. Focal length: $f_c = \begin{bmatrix} f_1 & f_2 \end{bmatrix}'$
2. Principal point: $c_c = \begin{bmatrix} c_1 & c_2 \end{bmatrix}'$
3. Skew coefficient: α_c
4. Distortion: $k_c = \begin{bmatrix} k_1 & k_2 & k_3 & k_4 & k_5 \end{bmatrix}'$

Suppose $P_o(X_c, Y_c, Z_c)$ is the coordinate of P_o in the camera reference frame where P_0 is the generic image point. The point is projected onto an image plane normal to the optical axis. The normalised (pinhole) image projection $i_n(x_n, y_n)$ is defined by (6). From (6), after including lens distortion k_c the new normalised point $i_d(x_d, y_d)$ is defined as [27]:

$$i_n(x_n, y_n) = \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} X_c / Z_c \\ Y_c / Z_c \end{bmatrix} \qquad (6)$$

$$i_d = \begin{bmatrix} x_{d1} \\ x_{d2} \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_5 r^6) i_n + t_d \qquad (7)$$

$$t_d = \begin{bmatrix} 2k_3 x_n y_n + k_4 (r^2 + 2x_n^2) \\ k_3 (r^2 + 2y_n^2) + 2k_4 x_n y_n \end{bmatrix} \qquad (8)$$

where t_d is the tangential distortion vector and $r^2 = x_n^2 + y_n^2$. Finally, the projected point $i_p(x_p, y_p)$ of $P_o(X_c, Y_c, Z_c)$ on the image plane is represented by (9).

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} f_1 (x_{d1} + \alpha_c x_{d2}) + c_1 \\ f_2 x_{d2} + c_2 \end{bmatrix} \qquad (9)$$

b. Camera velocity estimation

The camera system setup is as shown in Figure 5. The optical axis of the camera is perpendicular to the ground plane. If the camera can be assumed to have negligible distortion and skew coefficients, then $k_c = [0 \quad 0 \quad 0 \quad 0 \quad 0]'$, and $\alpha_c = 0$.

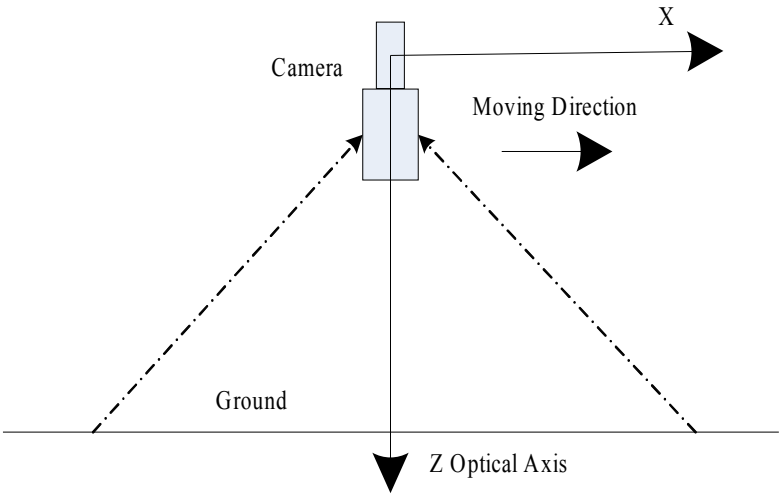

Figure 5: The Camera System Setup

Substituting (6), (7) and (8) into (9), gives:

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} f_1 \dfrac{X_c}{Z_c} + c_1 \\ f_2 \dfrac{Y_c}{Z_c} + c_2 \end{bmatrix} \qquad (10)$$

Differentiating (10) with respect to time; the velocity of the camera in $X-Y$ plane is given as:

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \frac{f_1}{Z_c^2}(Z_c V_x - X_c V_z) \\ \frac{f_2}{Z_c^2}(Z_c V_y - Y_c V_z) \end{bmatrix}; \quad \begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \frac{Z_c v_x}{f_1} + \frac{X_c V_z}{Z_c} \\ \frac{Z_c v_y}{f_2} + \frac{Y_c V_z}{Z_c} \end{bmatrix} \qquad (11)$$

Rearranging (10) to get $\frac{X_c}{Z_c}$ and $\frac{Y_c}{Z_c}$ and substituting in (11), yields (12).

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \frac{Z_c v_x}{f_1} + \frac{1}{f_1}(x_p - c_1)V_z \\ \frac{Z_c v_y}{f_2} + \frac{1}{f_2}(y_p - c_2)V_z \end{bmatrix} \qquad (12)$$

Where

- V_x, V_y and V_z are the camera velocities in *x*, *y* and *z* directions respectively.
- Z_c is the distance between the feature point on the ground and the centre of projection of the camera.
- v_x and v_y are the image velocity, $v_x = \Delta x F$ and $v_y = \Delta y F$, where F is camera's frame rate and $(\Delta x, \Delta y)$ is the optical flow.
- (x_p, y_p) is the centre point of the feature window in the first image.
- Focal length: $f_c = \begin{bmatrix} f_1 & f_2 \end{bmatrix}'$, (Obtained through camera calibration) .
- Principal point: $c_c = \begin{bmatrix} c_1 & c_2 \end{bmatrix}'$. (Obtained through camera calibration).

If the ground is assumed flat, i.e. Z_c is constant, eq. (12) becomes

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \frac{Z_c v_x}{f_1} \\ \frac{Z_c v_y}{f_2} \end{bmatrix}. \qquad (13)$$

II. VELOCITY ESTIMATION PROCEDURE

For 1D velocity estimation, every feature window in the same image would move with the same speed. If only one feature window is considered, the estimated velocity may fluctuate due to discretization errors. Therefore, 20 feature windows are selected and the average optical flow d is used to estimate the velocity for each image pair. Then, the estimated velocity is filtered using a low pass IIR-filter, to further eliminate noise. The velocity estimation procedure is as follows:

1. Calibrate camera to get the intrinsic camera parameters focal length f_c, distortion k_c, principle point c_C and skew coefficient α_c. Also set initial variable defining the numbers of feature windows and initial camera height
2. Acquire two consecutive images I_1 and I_{i+1}.
3. The numbers of good features to track are identified by the method proposed in [35]. This method tends to select corner features in the image which reduces the computational burden to find the optical flow of every pixel in the whole image.
4. Apply the optical flow algorithm to I_1 and I_{i+1} to get d which is the average optical flow from $n(=20)$ feature windows.
5. Calculate the velocity using eq. (13). The estimated velocity is obtained after applying a low pass IIR-filter..
6. Repeat above procedure for image sequence. This procedure is further summarized in the flow chart shown in Figure 6.

III. EXPERIMENTAL STUDY

Experiments are performed to check the robustness, precision and feasibility of the proposed approach using different soil types, namely coarse sand, fine sand, gravel as well as a mixture of sand and gravel. Two sets of experiments are performed using a test rig and a mobile robot respectively. First, the camera intrinsic parameters are determined and then velocity estimation is performed. Further, we conduct an experimental study in order to determine the most appropriate number of optical flow feature windows and their size as a function of camera height with a view on optimising velocity estimates.

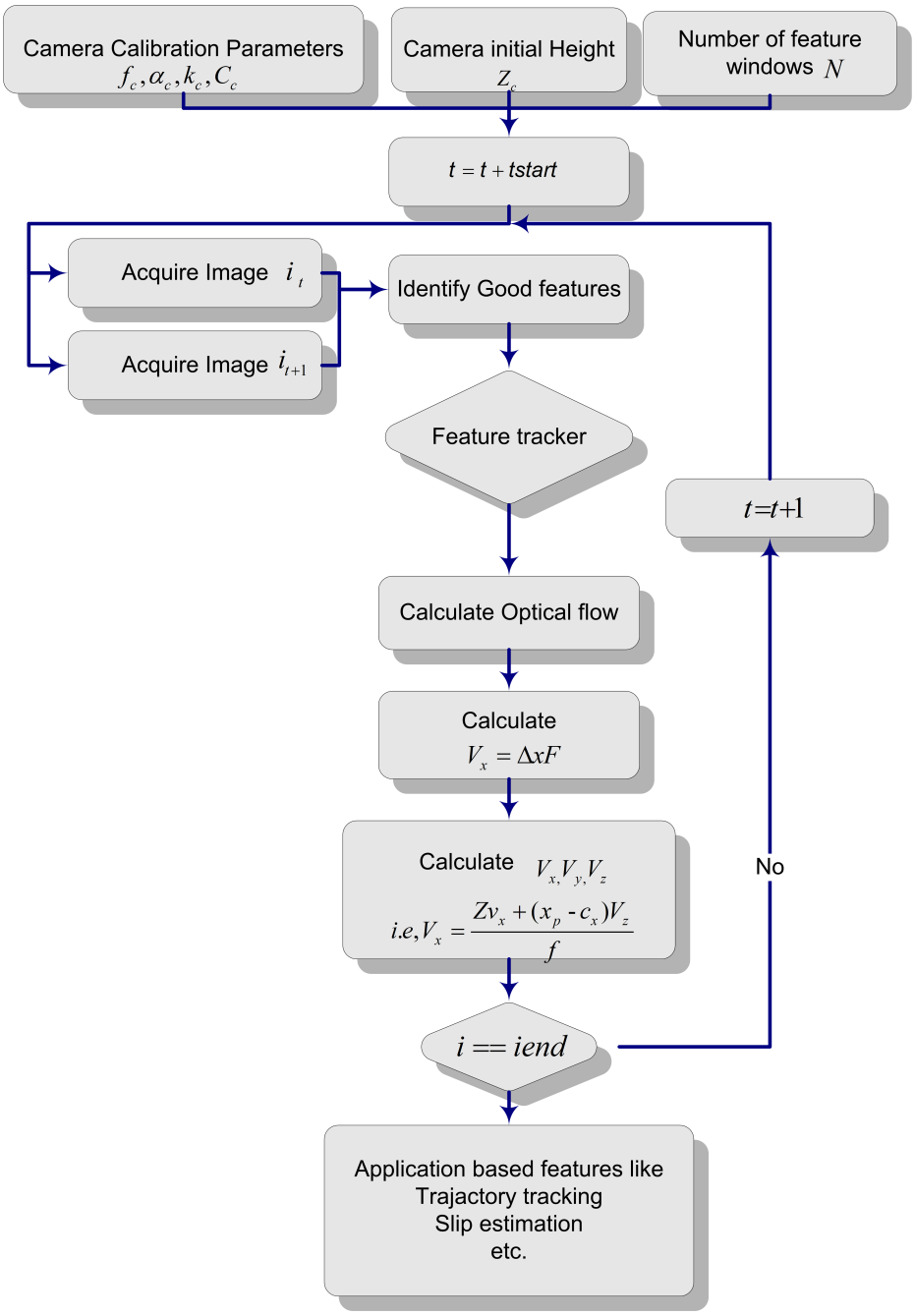

Figure 6. Optical flow based velocity estimation technique flow chart

a. Camera calibration

Before velocity estimation can commence, the intrinsic parameters of the employed camera need to be determined through a calibration procedure. Here, the intrinsic parameters of the camera are determined using the Matlab camera calibration toolbox [27] and chessboard-type

images like those shown in Figure 11. The intrinsic parameters obtained for this particular camera are as follows,

$$f_c = \left[1731.52\ 51729.896\right] \pm \left[14.150\ 14.098\right] \quad (14)$$

$$c_c = \left[598.587\ 503.540\right] \pm \left[17.324\ 16.506\right] \quad (15)$$

$$\alpha_c = 0.0 \pm 0.0 \quad (16)$$

$$k_c = \left[-0.131\ 0.351\ 0.004\ -0.002\ 0.0\right] \pm \left[0.045\ 0.325\ 0.002\ 0.0\right] \quad (17)$$

b. Determining feature window size

Figure 2 shows the basic concept of the optical flow algorithm where 'A' is a feature window in the first image and 'B' is the corresponding feature window in the second image. The "optical flow" or distance (d_x, d_y) between those two points is computed using the optical flow approach. In the example shown in Figure 7, there is only a change in *x* direction; i.e. only d_x is changing. Finding the optimal feature window size is critical to the efficiency of the optical flow algorithm as this largely affects both computational time and accuracy of the results. Simulated image sequences of all soil types are used in the tests described here. In order to test the capabilities of the proposed algorithm in a controlled environment, small-sized images are generated from one large image. This allows simulating the "movement" of the small-sized images across the larger image with varying amounts of overlap between subsequent images using. The small-sized images are generated by cropping larger camera images with a size of $1280 \times 1024\, pixel^2$ generating small images with a size of $640 \times 512\, pixel^2$. Artificial motion between two images is achieved by selecting a first image at the most left location of the larger image and a second image at a distance of *x* shifted to the right, Figure 7. During this experimental study, distance *x* was increased for a given feature window size until the optical flow algorithm failed to correctly estimate the distance by which image I2 was shifted. The optical flow output before failure occurred was recorded as the maximum optical flow achievable for a given feature window size for a particular image type. Four different image types, including fine sand, course sand, stone/gravel and a mixture of course sand and gravel were investigated.

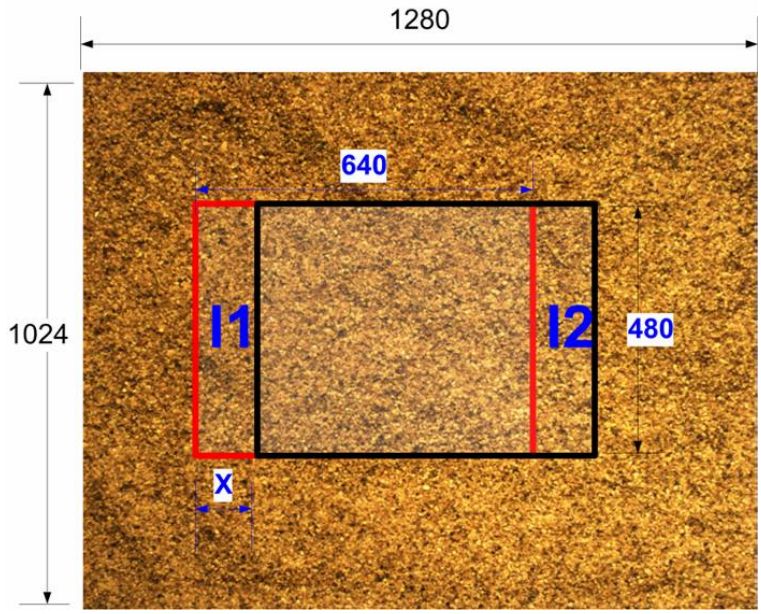

Figure 7. Simulated image sequence; Image I2 is shifted by *x* pixels.

Based on recommendations described in [26], square feature windows are chosen such that each window's width (and height) is an odd number of pixels long and at least 3 pixels long; each feature window thus has a clearly identifiable center pixel [26]. Simulation experiments were performed by increasing the feature windows width from 3 pixels in 1 pixel steps until the optical flow algorithm fails, Figure 8 (left). Based on practical experience with optical flow techniques, the number of feature windows was set to twenty. The feature windows chosen in the first image are then tracked in the second image by the optical flow algorithm. The resulting optical flow outputs and the computational time required to process different window sizes were investigated, Figure 8 (left) and (right). Figure 8 (left) shows that the optical flow not only depends on the feature window size but also on the terrain type. The experiments show that the maximum optical flow computed for gravel/stone images is greater than that for the other image types, while the optical flow computed for fine sand images is comparatively low. This means that the algorithm is capable of estimating larger image shifts when applied to images with a coarser texture.

The feature window size also affects the computational time of the optical flow algorithm. The optical flow algorithm fails when the feature window width is equal to or greater than 27 pixels, Figure 8 (right). Since the proposed approach is to estimate vehicle velocities over a wide range, it is desirable to optimize it for large image shifts or, in other words, large optical flows. In addition, a short execution time of the optical flow algorithm is preferred, to realize

real-time applications. In order to obtain the optimal window size which maximises the optical flow and, at the same time, favours the least computational time of the velocity estimation scheme, the following cost function has been defined,

$$Cost(w) = \sum_{i=1}^{n} \frac{mo_i(w)}{ct_i(w)}, \quad (18)$$

where,

- w is the feature window width
- n is the number of image types, here: $n = 4$.
- mo_1, mo_2, mo_3, mo_4 are the maximum optical flows for fine sand, coarse sand, gravel and mixed coarse sand - gravel respectively
- ct_1, ct_2, ct_3, ct_4 are the computational times of fine sand, coarse sand, gravel and mixed coarse sand and gravel respectively.

The appropriate feature window sizes are chosen such that $Cost(w)$ is maximized. For a maximum velocity of 50 mm/s at a camera height Z_c = 250 mm, employing a camera with a frame rate of 30 fps and focal length f_1 = 1731.5 and by substituting $V_x = mo_i(w)F$ into equation (13), the following inequality is obtained,

$$mo_i(w) > \frac{V_x f_1}{F Z_c} > 11.54. \quad (19)$$

From inequality (19), we find that a maximum optical flow ($mo_i > 11.54$ pixels) should be chosen for all soil types to optimize velocity estimation for the given constraints. Figure 8(a) shows that the window width w should not exceed 23 pixels, since the optical flow algorithm fails for higher feature window sizes. The cost function for window width w in the range of 13 to 23 pixels is shown in figure 9. The optimal window width w is 15 pixels as it maximises the used cost function. This window width is chosen for all further experiments.

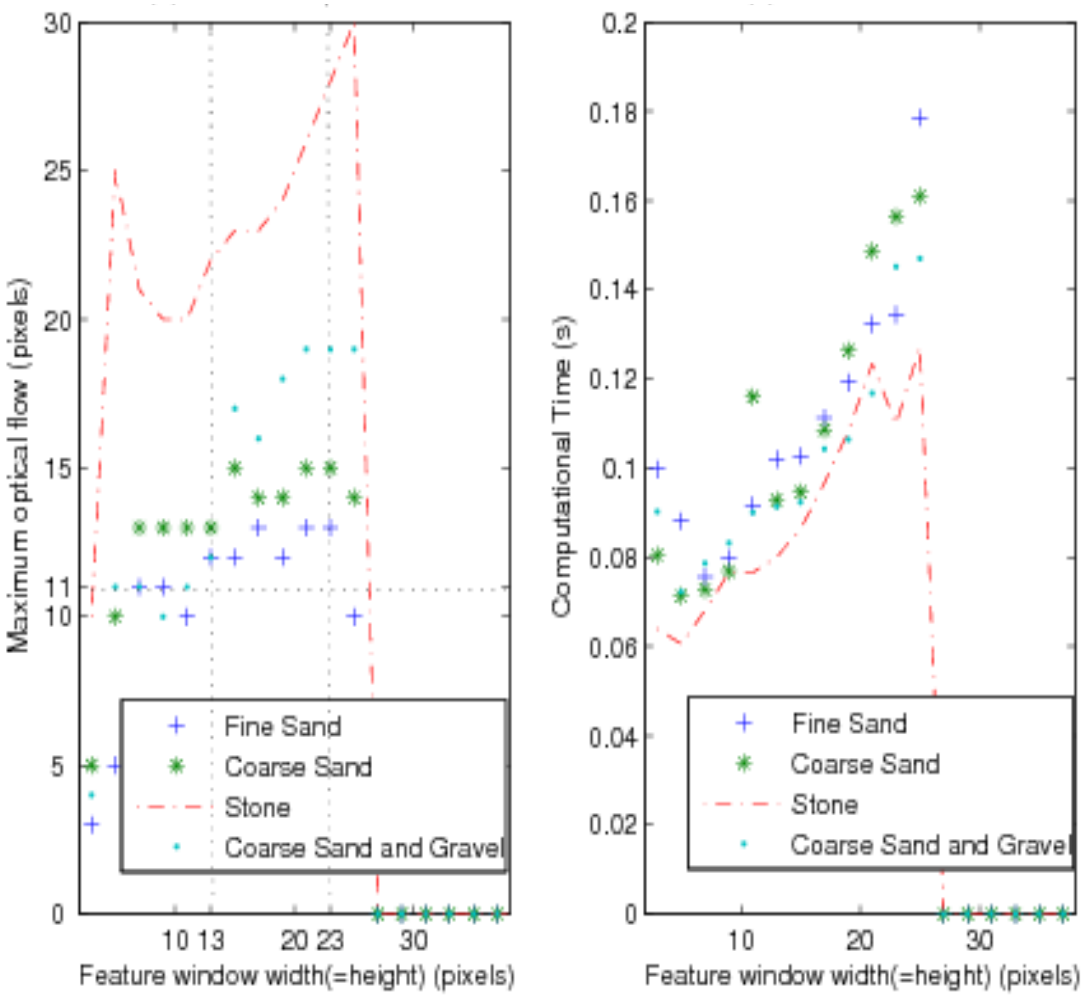

Figure 8. (Left) Optical Flow and (right) Computational Time.

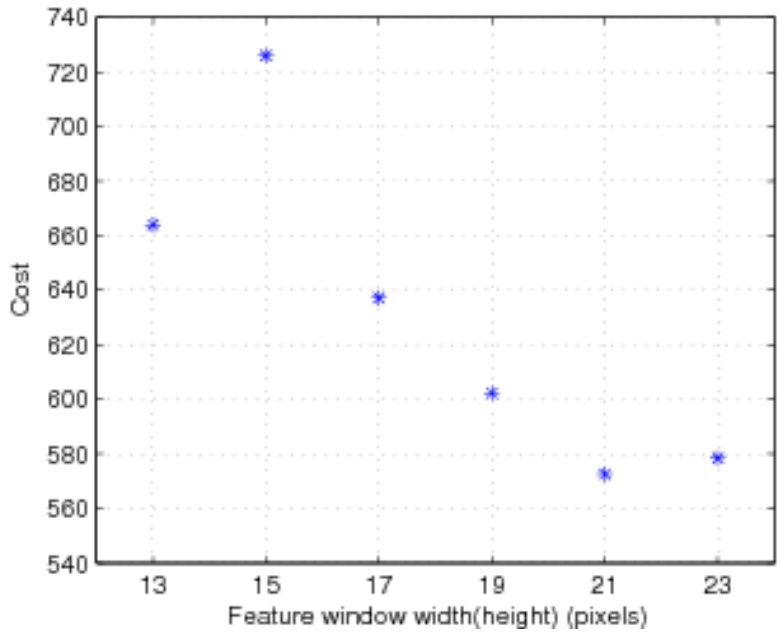

Figure 9. Cost Function

c. Test rig

The velocity measurement test rig is shown in Figure 10 and a schematic of the test rig is shown in Figure 11. The test rig consists of a DC motor driven carriage to which a camera is attached. During experimentation, the motor drives a chain attached to the carriage and causes the camera to move at different speeds. The measured velocity is calculated from a high resolution encoder attached to the motor. The camera is a Silicon Video 1281 CMOS Cameras. The camera can be set to different frame rates at specific frame sizes; at higher frame rates the frame size that can be captured is reduced and vice versa. The maximum frame rate is 6,460 fps at an image size of $50X50$ pixels^2 .

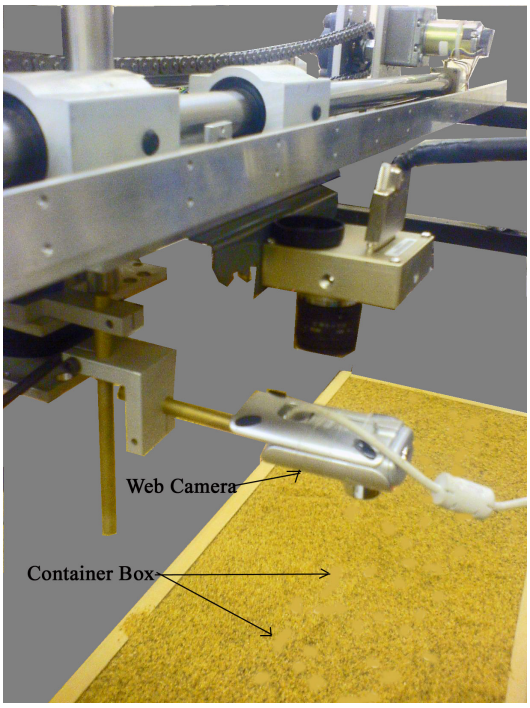

Figure 10. The Velocity Measurement Test Rig

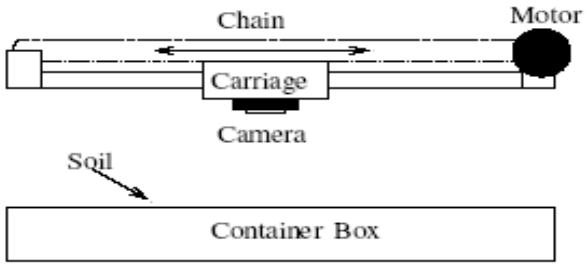

Figure 11: Schematic Of Velocity Measurement Test Rig

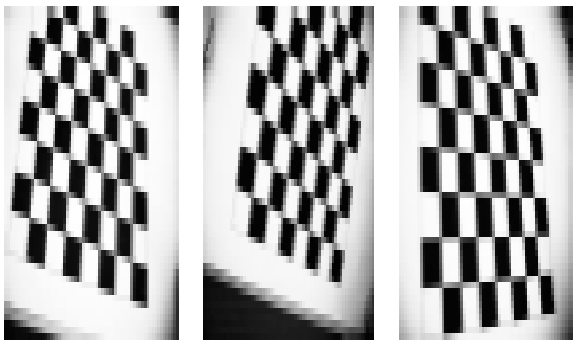

Figure 12: Chessboard Images For The Cameras Calibration

e. Velocity estimation experiments

Experiments are carried out by moving the camera with various constant velocities ranging between 0 and 50 mm/sec over 4 different types of ground (coarse sand, fine sand, gravel, and a mixture of coarse sand and gravel). During all experiments, the camera is set to capture images at 30 fps at an image size of 1280×1024 pixels^2. Camera image distortion is

compensated for employing the Matlab camera calibration toolbox and, further, images are cropped to a size of 640×512 pixels^2 allowing the optical flow algorithm to be executed in a reasonable time. The optical flow algorithm based on eq. (13) is applied. Results employing the optical flow algorithm on fine sand are shown in Figure 13. The camera velocity is estimated using eq. (13); noise is filtered by a 3rd-order IIR digital filter. The results shown in tables 1–4 are average estimated velocities at steady state. Velocity v is the measured velocity using the test rig encoder, and $\hat{v}$ is the velocity estimated using the optical flow algorithm. Figure 13 compares the predicted and measured velocities. The results (tables 1-4 and Figures 13-16) show that there is good agreement between the predicted and measured velocities, with a mean percentage error of 1.489%. The results shown in Figures 13-16 suggest that the maximum estimation errors are in the region of 2% at steady state over a wide range of different scenarios, excluding the occasional outliers which could be eliminated employing a dynamic model of the vehicle suppressing implausible acceleration transients.

Table 1: Average Estimated Velocities Over Coarse Sand

v (mm/s)	$\hat{v}$ (mm/s)	$e=\lvert v-\hat{v}\rvert$ (mm/s)	%Error
2.806	2.612	0.194	6.903
7.047	6.969	0.078	1.110
11.111	10.994	0.117	1.052
15.565	15.553	0.012	0.078
19.672	19.399	0.273	1.389
24.194	24.282	0.088	0.366
28.726	28.797	0.071	0.247
33.074	33.104	0.029	0.090
37.499	37.563	0.063	0.169
41.926	41.911	0.015	0.035
46.259	46.048	0.211	0.456
50.447	50.274	0.173	0.344

Table 2: Average Estimated Velocities Over Fine Sand

v (mm/s)	$\hat{v}$ (mm/s)	$e=\lvert v-\hat{v}\rvert$ (mm/s)	%Error
2.577	2.479	0.098	3.821
6.748	6.769	0.022	0.322
10.809	10.879	0.069	0.647
15.226	15.362	0.136	0.895
19.676	19.915	0.239	1.214
23.796	24.077	0.281	1.179
28.268	28.503	0.234	0.831
32.518	32.599	0.082	0.251
36.984	37.016	0.031	0.084
41.269	40.972	0.297	0.719
45.720	45.269	0.450	0.986
49.984	47.967	2.017	4.036

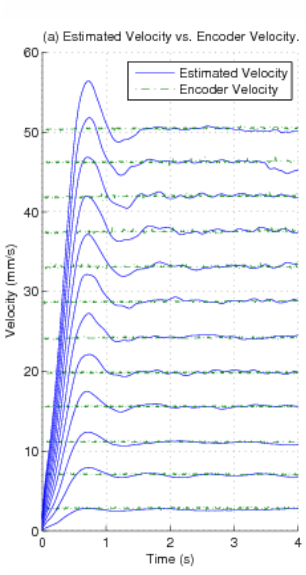

Figure 13. The Estimated Velocities Over Coarse Sand

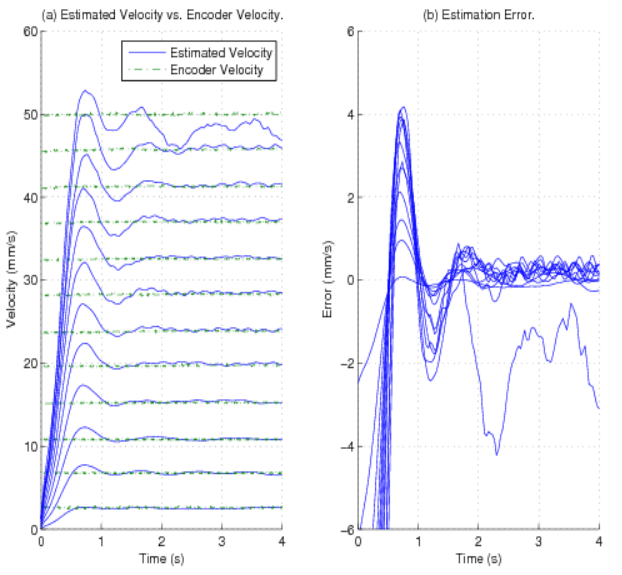

Figure 14. The Estimated Velocities Over fine Sand

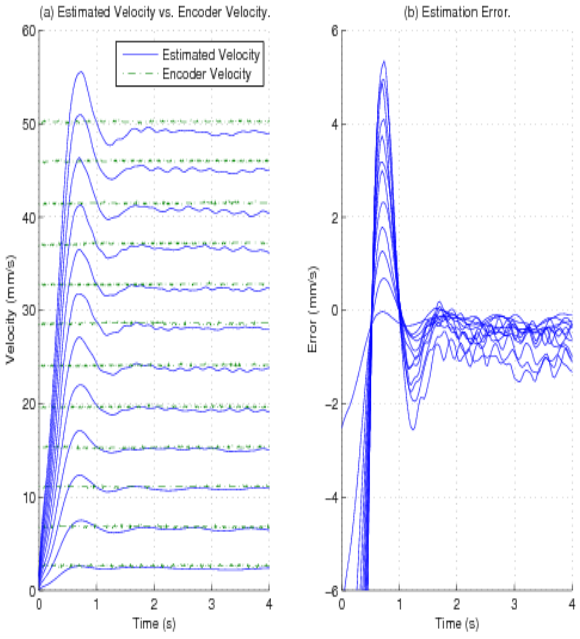

Figure 15. The Estimated Velocities Over Gravel

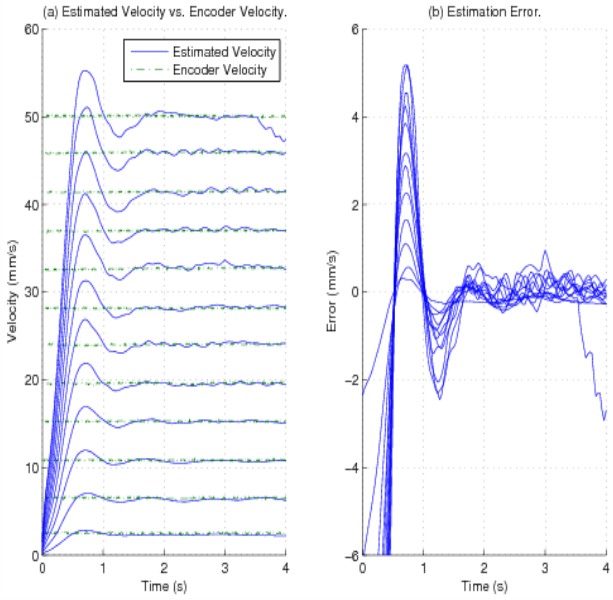

Figure 16. The Estimated Velocities Over Mixture of gravel and sand

Table 3: Average Estimated Velocities Over Gravel

v (mm/s)	$\hat{v}$ (mm/s)	$e=\lvert v-\hat{v}\rvert$ (mm/s)	%Error
2.612	2.325	0.286	10.939
6.802	6.539	0.263	3.870
11.075	10.883	0.192	1.733
15.319	15.097	0.223	1.454
19.672	19.399	0.273	1.389
24.127	23.854	0.273	1.136
28.583	28.163	0.420	1.471
32.791	32.380	0.411	1.253
37.158	36.640	0.518	1.395
41.483	40.811	0.672	1.620
45.992	45.188	0.804	1.748
50.211	49.176	1.0335	2.062

Table 4: Average Estimated Velocities Over Coarse Sand And Gravel

v (mm/s)	$\hat{v}$ (mm/s)	$e=\lvert v-\hat{v}\rvert$ (mm/s)	%Error
2.744	2.315	0.159	6.458
6.498	6.299	0.199	3.056
10.839	10.685	0.155	1.426
15.278	15.139	0.139	0.909
19.594	19.496	0.098	0.498
23.997	24.003	0.006	0.023
28.117	28.146	0.029	0.103
32.652	32.796	0.144	0.442
36.983	36.945	0.039	0.0105
41.426	41.275	0.153	0.369
45.904	45.858	0.045	0.099
50.079	49.504	0.575	1.149

f. Velocity estimation at different heights Z_c

The height of the camera with respect to the ground over which the vehicle is moving is key factor in order to estimate the vehicle's velocity, eq. (12). Here, this variation in height is analysed and also its effect on accuracy of velocity estimation is investigated. During these experiments, height Z_c between the camera and the ground varies between 220–280 millimetres. At the mean height value of $Z_n = 250$ millimetres, the image is in focus. The camera moves at 40 mm/s in *x* direction and height Z_c is varied by inserting or removing 6 millimetre thick plates below the soil container box.

Resulting velocity estimates can then be compared to the velocity readings of the test rig encoder. The velocity error is the difference between the mean value of the estimated velocity and the velocity measured using the test rig encoder. Errors at different heights when

travelling over different types of soils are shown as percentage values in Figures 17-20. Uncompensated height values Z_c are calculated by keeping $Z_c = Z_n = const.$ in eq. (12), while compensated values Z_c are calculated by inserting the actual camera height value in eq. (12). The measured velocities for different soil types are in good agreement with estimated ones for height values close to Z_n. For large deviations of the camera height from Z_n the error increases. If knowledge of the camera height is available the error can be reduced considerably (see blue lines in Figures 17 – 20). However, the error depends not only on the Z_c value but also on to what extend the image is out of focus; any image blur reduced the ability of the optical flow algorithm to provide accurate estimates.

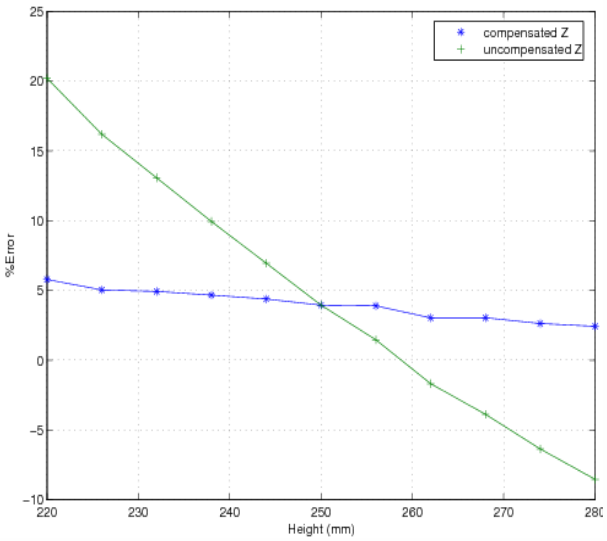

Figure 17. Estimated Error Using Gravel Varying Zc

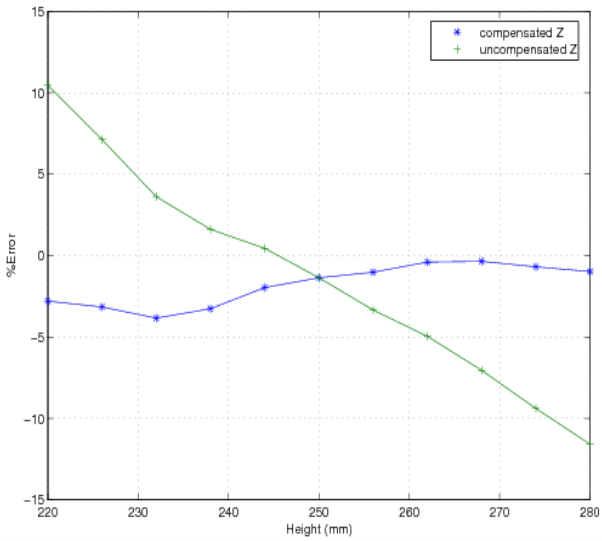

Figure 18. Estimated Error Using Fine Sand Varying Zc

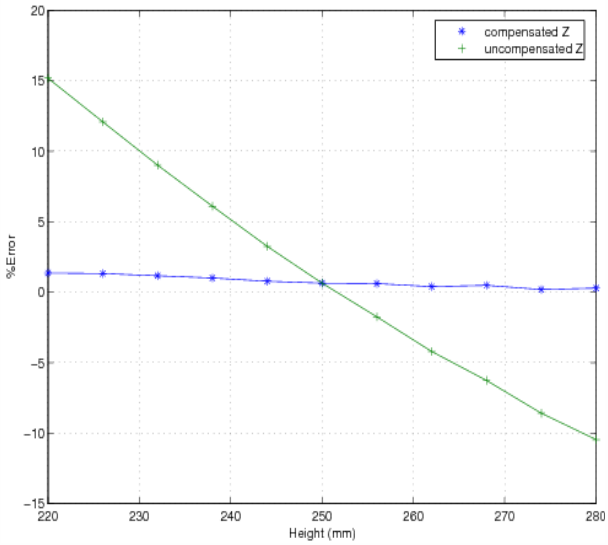

Figure 19. Estimated Velocities Over Mixed Coarse Sand And Gravel Varying Zc

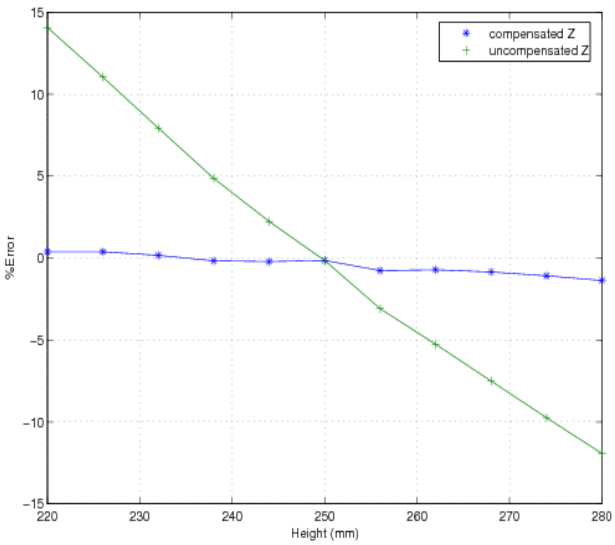

Figure 20. Estimated Error Using Coarse Sand Varying Zc

These results show that if the vehicle is moving in rough terrain it is important to compensate for any change in height of camera in order to achieve accurate vehicle velocity. There are readily available sensors in the market which could be integrated with the camera to monitor slight changes in the camera/ground height. Laser height sensors, ultrasonic distance sensors and infrared sensors are examples that could be used in conjunction with the camera for real-time height measurements. Further, stereo cameras could be employed to extract 3D information from the scene; however, it is noted that this will increase the complexity and cost of the system.

IV. CONCLUSION AND FUTURE WORK

An optical flow algorithm for velocity estimation is proposed and its feasibility is evaluated by using a specially designed test rig. It is shown that the vehicle speed can be predicted to a relatively high accuracy (± 2.017 mm/sec). Different ground samples were used to validate the algorithm, and in most cases, the system accurately measures the carriage velocity, with a mean maximum percentage error of 1.489% at steady state.

The velocity estimation experiments are performed above a flat test bed and a constant camera height. In addition, a study is presented investigating the estimation algorithm's performance when changing the height of the camera during vehicle motion and its effect on velocity estimation capabilities. Measured velocities from precise optical encoders are compared with the velocities predicted by the optical flow algorithm showing very good agreement. The optimal feature window size for real-time optical flow based velocity estimation was also established. Overall, the approach has potential for accurate velocity estimation. Experiments are underway for different real-world conditions and further research will aim at developing a robust stand alone velocity measurement system.

V. REFERENCES

[1] Z.B. Song, Y H Zweiri, L D Seneviratne and K Althoefer, "Non-linear observer for estimating slips of wheeled vehicles", The 12th Annual IEEE Conference On Mechatronics And Machine Vision In Practice, Manila, Philippines (2005).

[2] Georgiev A, Allen PK, "Localization methods for a mobile robot in urban environments" IEEE Transactions On Robotics And Automation 20 (5): 851-864 Oct 2004

[3] Panzieri S, Pascucci F, Ulivi G, "An outdoor navigation system using GPS and inertial platform", IEEE-ASME TRANSACTIONS ON MECHATRONICS 7 (2): 134-142 JUN 2002

[4] Lobo J, Dias J," Vision and inertial sensor cooperation using gravity as a vertical reference", IEEE Transactions On Pattern Analysis And Machine Intelligence 25 (12): 1597-1608 Dec 2003

[5] Charles Thorpe, Martial H. Hebert, Takeo Kanade and Steven A. Shafer," Vision and Navigation for the Carnegie-Mellon Navlab," IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol. Io, No. 3, May 1988

[6] Davison, "Real-time simultaneous localization and mapping with a single camera", IEEE International Conference on Computer Vision, Nice, France, pp 1403–1410 (2003).

[7] Y. Takaoka, Y. Kida, S. Kagami, H. Mizoguchi and T. Kanade, "3d map building for a humanoid robot by using visual odometry", IEEE International Conference on Systems, Man and Cybernetics, The Hague, pp 4444–4449 (2004).

[8] E. Marchand, P. Bouthemy, F. Chaumette and V. Moreau, "Robust real-time visual tracking using a 2D-3D model-based approach", IEEE International Conference on Computer Vision, Kerkyra, pp 262–268 (1999).

[9] B. Jung and G. S. Sukhatme, "Detecting moving objects using a single camera on a mobile robot in an outdoor environment", In the 8th Conference on Intelligent Autonomous Systems, Amsterdam, pp 980–987 (2004).

[10]M. Betke, E. Haritaoglu and L. Davis, "Multiple vehicle detection and tracking in hard real time", IEEE Symposium on Intelligent Vehicles, Tokyo, pp 351–356 (1996).

[11]J. M. Ferryman, S. J. Maybank and A. D. Worrall, "Visual surveillance for moving vehicles", in Journal of Compter Vision, 37(2), pp 187–197 (2000).

[12] S. Thrun and M. Montemerlo. The GraphSLAM algorithm with applications to large-scale mapping of urban structures. International Journal on Robotics Research, 25(5/6):403-430, 2005.

[13]Davison, "Real-Time Simultaneous Localization and Mapping with a Single Camera," IEEE International Conference on Computer Vision, pp. 1403–1410, 2003.

[14] DeSouza GN, Kak AC, "Vision for mobile robot navigation: A survey ", IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2): 237-267 Feb 2002.

[15]Yang Cheng, Mark W. Maimone, and Larry Matthies, "Visual Odometry on the Mars Exploration Rovers", IEEE Robotics & Automation Magazine, June 2006.

[16]Y. Takaoka, Y. Kida, S. Kagami, H. Mizoguchi and T. Kanade, "3D Map Building for a Humanoid Robot by using Visual Odometry," IEEE International Conference on Systems, Man and Cybernetics, pp. 4444–4449, 2004.

[17]David Fernandez and Andrew Price, " Visual Odometry for an Outdoor Mobile Robot", Proceedings of the 2004 IEEE Conference on Robotics, Automation and Mechatronics, Singapore, 1-3 December, 2004

[18]Savan Chhaniyara, Pished Bunnun, Yahya H Zweiri, Lakmal D Seneviratne and Kaspar Althoefer, "Feasibility of Velocity Estimation for All Terrain Ground Vehicles using an Optical Flow Algorithm", ICARA 2006-Third international conference on autonomous robots and agents, pp. 429-434, 12-14 December 2006, New Zealand.

[19] Savan Chhaniyara, Kaspar Althoefer, Yahya Zweiri, Lakmal Seneviratne, "A novel approach for Self-Localization based on Computer Vision and Artificial Marker Deposition", ICNSC -IEEE International Conference on Networking, Sensing and Control, p139-p144, 15-17 April 2007, London

[20]David Fernandez and Andrew Price, " Visual Odometry for an Outdoor Mobile Robot", Proceedings of the 2004 IEEE Conference on Robotics, Automation and Mechatronics, Singapore, 1-3 December, 2004

[21]Kelly, "Pose tracking for mobile robot localization from large scale appearance mosaics", in International Journal of Robotics Research, 19(11) pp 1104-1125 (2000).

[22]Yang Cheng, Mark W. Maimone, and Larry Matthies, "Visual Odometry on the Mars Exploration Rovers", IEEE Robotics & Automation Magazine, June 2006.

[23]B.K.P. Horn and B.G. Schunck, "Determining Optical flow", Artificial Intelligence, 17, pp 185-203 (198 I)

[24]J.L. Barron, D.J. Fleet, and S.S. Beauchemin, "Performance of optical flow techniques", Int. J. Comput. Vis. 12(1), pp 43–77 (1994).

[25]B.D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision", Proc. Imaging Understanding Workshop., pp 121–130 (1981).

[26] S. Birchfield, "Kanade-Lucas-Tomasi feature tracker", http://www.ces.clemson.edu/ ~stb/klt visited on 10/05/2006

[27]"Camera calibration toolbox for matlab", http://www.vision.caltech.edu/bouguetj

[28] Owens, R. [1997] \Pinhole camera model," online available: http://homepages.inf.ed.ac.uk/rbf/cvonline/LOCALCOPIES/OWENS/LECT1/node2.html.

[29]S. Singh and B. Digney, "Autonomous cross-country navigation using stereo vision", Tech. report, Robotics Institute, Carnegie Mellon University, CMU-RI-TR-99-03 (1999).

[30] Wu, "Optical navigation system", "Bsc. Thesis" Department of Comp. Sci. and Electrical Engineering, University of Queensland, (2001)

[31]M. H. Bruch, G.A. Gilbreath, J.W. Muelhauser and J.Q. Lum, "Accurate waypoint navigation using non-differential GPS", Tech. Report, Space and naval warfare systems centre, CA. (2002).

[32]S. Clark, "Autonomous land vehicle navigation using millimetre wave radar", Proceedings of the IEEE ICRA 1998, Leuven, pp 3697-3702 (1998).

[33]D. Dickmanns et al., "The seeing passenger car vamorsp", IEEE Symp. Intell. Veh. '94, Paris, pp 68–73 (1994).

[34]T. Kanade, C. Thorpe, M. Herbert, and S. Shafer, "Vision and navigation for the Carnegie-Mellon navlab", IEEE Trans. Pattern Anal. Machine Intell., 10(3), pp 361–372 (1988).

[35] Jianbo Shi and Carlo Tomasi. Good Features to Track. IEEE Conference on Computer Vision and Pattern Recognition, pages 593-600, 1994.