Current-Based Slippage Detection and Odometry Correction for Mobile Robots and Planetary Rovers

Lauro Ojeda, *Member, IEEE*, Daniel Cruz, *Member, IEEE*, Giulio Reina, and Johann Borenstein, *Senior Member, IEEE*

***Abstract*—This paper introduces a novel method for wheel-slippage detection and correction based on motor current measurements. Our proposed method estimates wheel slippage from motor current measurements, and adjusts encoder readings affected by wheel slippage accordingly. The correction of wheel slippage based on motor currents works only in the direction of motion, but not laterally, and it requires some knowledge of the terrain. However, this knowledge does not have to be provided ahead of time by human operators. Rather, we propose three tuning techniques for determining relevant terrain parameters automatically, in real time, and during motion over unknown terrain. Two of the tuning techniques require position ground truth (i.e., GPS) to be available either continuously or sporadically. The third technique does not require any position ground truth, but is less accurate than the two other methods. A comprehensive set of experimental results have been included to validate this approach.**

***Index Terms*—Mobile robot navigation, parameter estimation, slippage detection, terrain factors.**

I. Introduction

RELATIVE positioning systems, also know as dead-reckoning systems, use wheel encoders and a simple method called "odometry" to estimate linear displacement. On smooth, flat terrain, and in the absence of wheel slippage, odometry is reliable and reasonably accurate over short distances, since wheel revolutions correspond to linear travel distance. On off-road terrain, especially on soft, sandy soil, odometry is typically not considered useful, because the wheels frequently slip and the measured rotation of the wheels does not accurately reflect traveled distance. Such wheel slippage is expected in planetary rovers traveling over sandy terrain, as well as many other mobile robot applications on off-road terrain. In earlier work, we proposed an odometry method that could provide good travel distance estimates as long as at least one wheel was gripping (= the opposite of "slipping") [1]. However, on soft, sandy, rolling terrain, all of the robot's wheels are likely to slip simultaneously. We refer to this condition as "all-wheel slippage" (AWS).

The extent and nature of AWS depend on many factors, such as soil characteristics, terrain inclination, load, vehicle configuration, kinematic incompatibility [2], and a number of other conditions. The interaction between wheels and terrain has been the subject of several studies. Perhaps the best known and most widely cited work is that of Bekker [3]–[5] and Wong [6]. Some more recent work focused on the interaction between wheels and terrain in conjunction with mobile robots [8]. Specifically, [9] studied terrain classification, [10] analyzed traversability, and traction control was studied in [11] and [7].

In this paper, we introduce a system for the correction of odometry errors on sandy terrain, such as that encountered by planetary rovers. We developed our system specifically for the six-wheel drive/six-wheel steer rovers developed by the Jet Propulsion Lab (JPL) (Pasadena, CA) for planetary exploration [12]. For the extensive testing of our system during its development, we built a fully functional and kinematically equivalent clone of JPL's Fido-class rovers [13]. Our clone, called "Fluffy" and shown in Fig. 1, is about half the size of Fido, but it features the same six-wheel independent-drive steering and a rocker-bogie passive suspension system.

Fluffy is equipped with an onboard inertial measurement unit (IMU) that uses three fiber-optic gyros for estimating the spatial orientation of the robot, two accelerometers for static tilt measurements, and six independent wheel encoders for odometry. Data from these sensors is fused by our unique Fuzzy Logic and Expert rule-based Navigation (FLEXnav) system [14]. On low-slippage terrain, the FLEXnav system is quite accurate, with errors typically smaller than 1% of overall travel distance. However, in the presence of significant AWS, position errors may become huge. Somewhat related work, for the case of agricultural soils, was presented in [15]. In this paper, the author relates torque and wheel slippage using a simplified linear model, which is used in combination with an extended Kalman filter to correct odometry.

In order to allow useful dead-reckoning even on high-slippage terrain, we developed our system for current-compensated odometry, called "iComp." Our proposed system comprises two functional modules, as shown in Fig. 2.

1) All-wheel slippage detection (AWSD) module. This module works in real time and does not require references to external markers or beacons.

Manuscript received January 21, 2005; revised September 8, 2005. This paper was recommended for publication by Associate Editor K. Yoshida and Editor F. Park upon evaluation of the reviewers' comments. This work was supported by NASA/JPL under Contract 1232300. This paper was presented in part at the 2002 IEEE International Conference on Robotics and Automation, Washington, DC, May 2002, and in part at the 2004 IEEE Aerospace Conference, Big Sky, MT, March 2004.

L. Ojeda and J. Borenstein are with the Advanced Technologies Lab., University of Michigan, Ann Arbor, MI 48109 USA (e-mail: lojeda@umich.edu; johannb@umich.edu).

D. Cruz is with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK 74078 USA (e-mail: daniel.cruz@okstate.edu).

G. Reina is with the Department of Innovation Engineering, University of Lecce, 73100 Lecce, Italy (e-mail: giulio.reina@unile.it).

Digital Object Identifier 10.1109/TRO.2005.862480

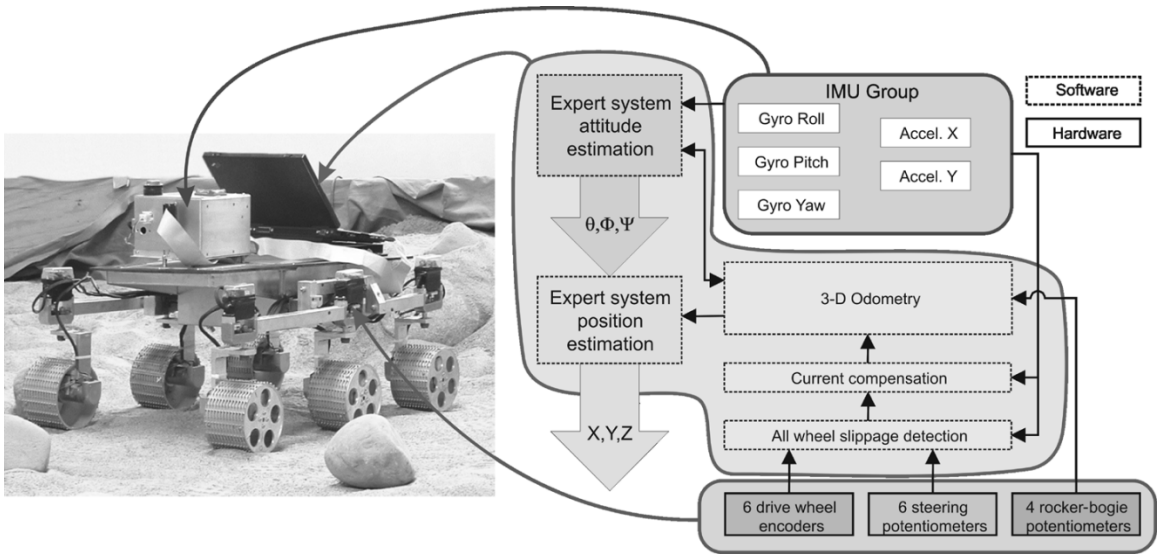


Fig. 1. University of Michigan-built Fluffy and a block diagram of the FLEXnav dead-reckoning system with wheel-slippage detection and correction. Note that each wheel has its own drive motor.

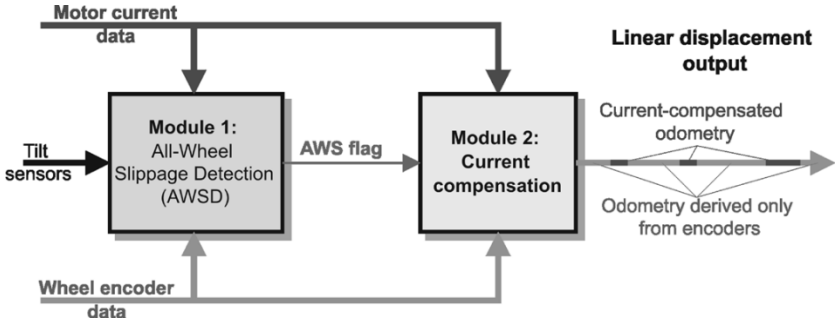


Fig. 2. Our proposed wheel-slippage detection and correction system is made up of two modules, the AWSD module and the iComp module.

2) Current compensation (iComp) module. This module corrects encoder readings so that the effect of AWS is greatly reduced or eliminated. The iComp module is based on our linearized slippage model, which relates wheel slippage to motor currents. We developed three different techniques for estimating the parameters for this model automatically.

Section II presents a detailed analysis of our linearized slippage model and describes our three techniques for estimating the model parameters. Section III explains and illustrates, using actual experimental data, how our linearized slippage model is used to correct encoder readings from slipping wheels. Section IV presents experimental results, and Section V, our conclusions.

One limitation of our approach is that our odometry correction applies to slippage along the longitudinal direction of motion only. However, in the presence of side forces, the wheels move at an angle (slip angle) with respect to the wheel plane, resulting in lateral slip [6]. This problem is not addressed in this paper.

II. Wheel-Slippage Detection and Estimation

In this section, we present a theoretical analysis of wheel slippage on soft terrain, and we explain our approach to estimating the parameters needed for our real-time wheel-slippage detection and correction method.

A. Theoretical Analysis

One widely accepted model for the prediction of wheel slippage on any terrain is the so-called Coulomb-Mohr soil failure criteria [3]. According to the Coulomb-Mohr soil failure criteria, *total* (100%) wheel slippage occurs when the shear stress applied to a given terrain exceeds the maximum shear stress $\tau_{\max}$ that the terrain can bear

$$\tau_{\max} = c + \sigma_{\max} \tan \varphi \tag{1}$$

where

c cohesion of the soil;
φ internal friction angle of the soil;
$\sigma_{\max}$ maximal normal component of the stress region at the wheel-terrain interface.

In order to describe the full range of operation of the wheel-slippage condition, the shear stress on loose sand may be described by an exponential function (see Fig. 3) [16]

$$\tau = (c + \sigma \tan \varphi)(1 - e^{-j/K}) \tag{2}$$

where

j shear displacement;
K shear deformation modulus;
σ normal stress.

Considering that φ is a constant obtained by a different process, it should not be part of Fig. 3. The normal or radial

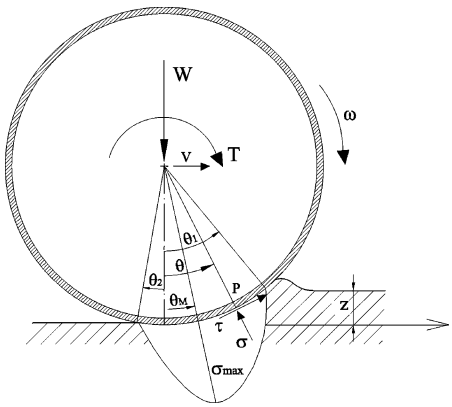


Fig. 3. Wheel-soil interaction model (adapted from [5]).

stress σ, the wheel sinkage z, and the wheel width b, are related according to the following equations [3]:

$$\sigma(z) = (k_c + k_\varphi b)\left(\frac{z}{b}\right)^n \tag{3}$$

where

k_c cohesive modulus of terrain deformation;
k_ϕ frictional modulus of terrain deformation;
n exponent of terrain deformation.

The maximum normal stress occurs at point θ_M and can be computed using [17]

$$\theta_M = (c_1 + c_2 i)\theta_1 \tag{4}$$

where θ_1 is the angle between vertical and leading edge of wheel contact patch, and i is the wheel slippage.

The normal pressure can be split in two regions. The front region (σ_1), located between the maximum pressure θ_M and θ_1, and the rear region (σ_2), from θ_M to θ_2. The angle θ_2 is measured between the vertical and trailing edge of the wheel; it is normally small $(\theta_2 \approx 0)$ and can be neglected. The normal pressure for the front and rear regions can be computed as a function of the angle θ, as follows [17]:

$$\sigma_1(\theta) = (k_c + k_\varphi b)\left(\frac{r}{b}\right)^n (\cos\theta - \cos\theta_1)^n \tag{5}$$

$$\sigma_2(\theta) = (k_c + k_\varphi b)\left(\frac{r}{b}\right)^n \times \left[\cos\left(\theta_1 - \frac{\theta}{\theta_M}(\theta_1 - \theta_M)\right) - \cos\theta_1\right]^n. \tag{6}$$

Shear displacement j is related to wheel slippage i and to angle θ, according to

$$j(\theta) = r[\theta_1 - \theta - (1-i)(\sin\theta_1 - \sin\theta)]. \tag{7}$$

Combining (2) and (7), the shear stress around the rim can be calculated as

$$\tau(\theta) = (c + \sigma(\theta)\tan\varphi)\left(1 - e^{-r/K[\theta_1 - \theta - (1-i)(\sin\theta_1 - \sin\theta)]}\right). \tag{8}$$

The normal stress $\sigma(\theta)$ can be resolved for the front and rear regions using (5) and (6), respectively. The torque T, with which the soil resists the rotation of the wheel, can be computed as the integral of the shear stress over the contact patch with respect to θ [17]

$$T = r^2 b \int_{\theta_2}^{\theta_1} \tau(\theta) d\theta. \tag{9}$$

Assuming that $\theta_2 = 0$, torque can be obtained from

$$\begin{aligned} T = r^2 b &\left\{ \int_{\theta_M}^{\theta_1} \tau_1(\theta) d\theta + \int_0^{\theta_M} \tau_2(\theta) d\theta \right\} \\ T = r^2 b &\left\{ \int_{\theta_M}^{\theta_1} \left[c + (k_c + k_\varphi b)\left(\frac{r}{b}\right)^n (\cos\theta - \cos\theta_1)^n \tan\phi\right] \right. \\ &\times \left(1 - e^{-r/K[\theta_1 - \theta - (1-i)(\sin\theta_1 - \sin\theta)]}\right) d\theta \\ &+ \int_0^{\theta_M} \left[(k_c + k_\varphi b)\left(\frac{r}{b}\right)^n \right. \\ &\times \left(\cos\left(\theta_1 - \frac{\theta}{\theta_M}(\theta_1 - \theta_M)\right) - \cos\theta_1\right)^n \tan\phi\Big] \\ &\left. \times \left(1 - e^{-r/K[\theta_1 - \theta - (1-i)(\sin\theta_1 - \sin\theta)]}\right) d\theta \right\}. \end{aligned} \tag{10}$$

In order to solve (10), θ_1 must be determined, and for this purpose, we use the following equation [17]:

$$W = rb\left(\int_{\theta_2}^{\theta_1} \sigma(\theta)\cos\theta d\theta + \int_{\theta_2}^{\theta_1} \tau(\theta)\sin\theta d\theta\right). \tag{11}$$

Equation (11) should be expanded in similar manner as shown with (9). Equation (11) is too complex and cannot be solved analytically; nevertheless, provided that W is known, the right side of (11) can be computed numerically for different values θ_1 until finding a value that solves the equation. For a detailed explanation of this method, see [17]. Once the torque has been determined, the motor current I, which is known to be roughly proportional to torques applied to the wheels, can be determined according to

$$T = k_I I \tag{12}$$

where k_I is the torque constant scale factor.

By combining (10) and (12), motor currents and slippage can be related, provided that all the other parameters are known. Using soil parameters for sand [6] (see Table I) and the parameters of our rover Fluffy, we created Fig. 4, which shows the relationship between current I and slip i. This graph was created by simulating different amounts of slippage and solving (4) to (12), the integrals were evaluated numerically using the Simpson method.

On frictional soils (such as dry sand), when the wheel slips, it also sinks. This phenomenon, called "slip-sinkage effect" [4], is already considered in the equations used for this simulation.

The wheel slippage can be defined as

$$i = 1 - \frac{v}{r\omega} \tag{13}$$

TABLE I
SAND PARAMETERS USED FOR SIMULATIONS

φ [deg]	28
c [kPa]	1.04
K [m]	0.025
k_c [kN/m^{n+1}]	0.99
k_φ [kN/m^{n+2}]	1528.43
n	1.1
c_1	0.18
c_2	0.32

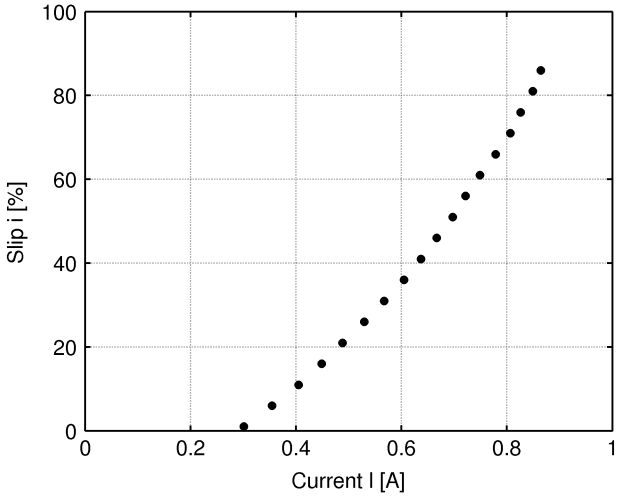


Fig. 4. Wheel slippage versus motor currents relationship model with slip-sinkage effect.

with v as the linear speed of the wheel, and ω as the angular rate of the wheel. Observe that the term $r\omega$ is the measured linear speed of the wheel or relative speed v_r registered by wheel encoders.

For a wheel moving at relative linear speed v_r, we define the parameter *velocity slippage correction*, S_c, which represents the correction value to be used for slippage compensation in Section III, and is related to wheel slippage i as follows:

$$S_c = v_r i. \tag{14}$$

In all the experiments reported in this paper, our rover Fluffy was commanded to travel at a fixed speed of 60 mm/s (v_r), as this is the envisaged top speed for the Mars Rover 2009 mission. Fig. 5 shows the resulting velocity slippage correction versus motor current graph.

The curve of Fig. 5 can be approximated by a linear equation without losing much accuracy. This approximation simplifies the real-time parameter-estimation techniques presented in the next section. The intersection of line $S_c\ (I)$ with the I-axis is of great physical significance: it represents the *onset* of wheel slippage. We denote the motor current associated with this point I_{slip}, and we call the slope of the linear approximation the scale factor for slippage correction, ξ. The complete equation that determines the slippage conditions for the robot can now be written as follows:

$$S_c = \begin{cases} \xi(I - I_{\text{slip}}), & I > I_{\text{slip}} \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

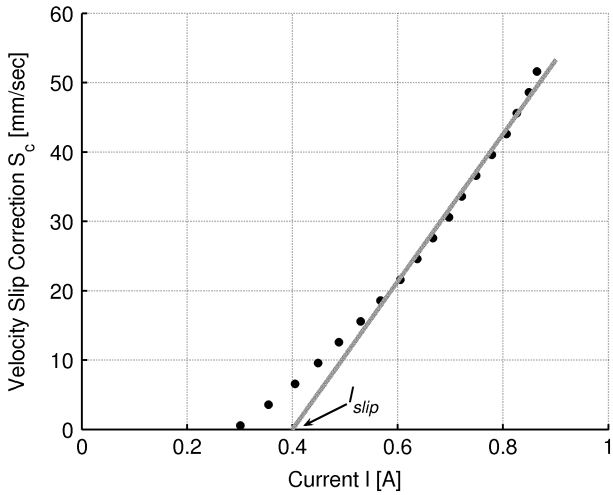


Fig. 5. Velocity slip correction vs. motor currents relationship. (Dark dots) simulated model expressed in terms of velocity slippage correction. (Gray solid line) linear approximation.

where ξ is the scale factor for slippage correction, and I_{slip} is the onset of wheel slippage.

The simplified model has the advantage of being entirely defined by two constants, I_{slip} and ξ; it is easy to implement in software and has low computational cost. In the remainder of this paper, we will make extensive use of this approximation model.

B. Real-Time Parameter Estimation

The accuracy of the velocity slippage correction versus motor current model of the preceding section depends on the accuracy of many empirically found constants. For many applications (e.g., planetary exploration) that information may be incomplete, inaccurate, time-varying, or simply not available at all. For this reason, we explored alternative methods that help determine the parameters I_{slip} and ξ. Among the parameter-estimation methods we evaluated, we found three to be particularly effective. Method I requires an external absolute position reference (i.e., a position measurement that does not depend on odometry) that is available continuously. Because of this constraint, this method is unlikely to be useful in a real planetary rover application. Method II requires an external absolute position reference that is available sporadically. Method III can be applied if no external absolute position reference is available.

1) Method I: Parameter Estimation Based on Continuously Available Absolute Positions: For this procedure, we require an absolute position system reference during the parameter-estimation stage. This absolute reference allows us to derive the ground-truth speed of the robot v_a, while the robot moves under varying slippage conditions. This was achieved by holding back the robot manually, using strings attached to the back of the robot. While measuring the ground-truth speed v_a, our system also measures the relative speed v_r based on wheel encoders. We recognize that holding the robot to introduce slippage is not the same as incurring wheel slippage in regular operation. However, as the discussion in Section III will show, these two sources of wheel slippage are effectively quite similar.

Slippage i can be estimated by redefining (13)

$$i = 1 - \frac{v_a}{v_r} \tag{16}$$

and slippage correction S_c can be computed as

$$S_c = v_r - v_a. \tag{17}$$

By measuring and computing S_c under different slippage conditions, while recording motor currents I corresponding to these conditions, we can sample multiple data pairs (S_c, I). Using the least-squares algorithm, a straight line is fitted to the data pairs to define the linearized function $S_c(I)$ according to (15).

C. Experimental Results

For this experiment, the robot was commanded to move straight forward. The speed was set at 60 mm/s and the total traveled distance was about 2 m. The hard and flat floor was covered by a layer of about 5 cm of loose, dry sand, and the entire experimental area was within the range of our so-called sonar-based absolute positioning system (SAPS). The SAPS is a position-measurement system that uses four sonar receivers at the corners of a rectangular test area, and a star-shaped array of sonar transmitters onboard the robot to measure the 2-D location of the robot within the test area. Ground-truth speed v_a can be measured at any given sampling interval k, using

$$v_a[k] = \frac{\Delta P_a[k]}{T_s} \tag{18}$$

where T_s is the sampling period, and ΔP_a is the distance traveled during sampling period T.

At a speed of 60 mm/s and at the sampling rate of 20 Hz, the robot only moves an average of 3 mm per absolute sampling interval T_s. Since the accuracy of the SAPS is also on the order of 3 mm, measuring speeds with the SAPS is not very accurate, the measurements are noisy, and only relatively large changes in speed can be detected by the SAPS.

Using the ground-truth speed of (18), we rewrite (17) in discrete form

$$S_c[k] = v_r[k] - v_a[k]. \tag{19}$$

We should explain, though, that the front wheels of Fluffy (as well as those of JPL's Fido) carry only a disproportional small load. For this reason, we disregard the currents of the front motors. Instead, the overall motor current I is determined as the average of the four current measurements on the motors of the center and rear wheels

$$I[k] = \frac{1}{m}\sum_{p=1}^{m} I_p[k] \tag{20}$$

where $m = 4$ is the number of motors used to estimate the average current.

Although the drive motors used in Fluffy have a high gear-reduction ratio (218.4), we were able, under most conditions, to measure torque changes in the wheels using motor current sensors. The results of numerous measurements relating slippage correction (19), and motor currents (20), are plotted in Fig. 6. Although the relationship is affected by noise, it still resembles a straight line in the slippage region, as we show in the previous theoretical analysis. A straight line that fits this data can be obtained applying the least-squares algorithm in the slippage region (see Fig. 6). The constants I_{slip} and ξ can be obtained directly from the linear equation, as explained in Section II-A.

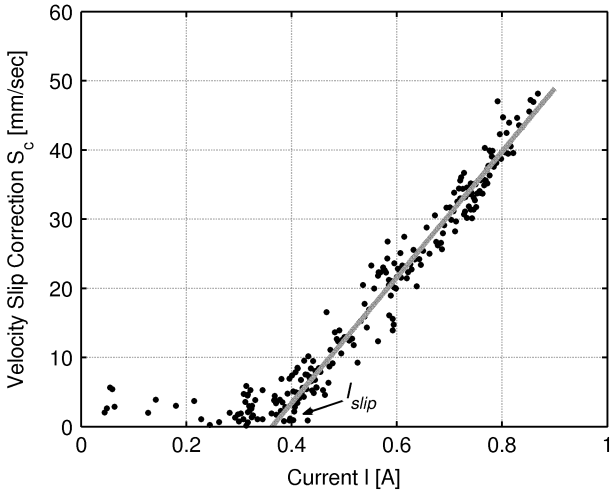


Fig. 6. Velocity slippage correction versus motor currents. Dark dots: experimental determined data pairs. Gray solid line: linear approximation.

1) Method II: Parameter Estimation Based on Sporadically Available Absolute Positions: In many applications, an absolute position reference may not be available continuously, as was the requirement for the previous method. For this (more likely) case, we developed the method described in this section. For this method, it is sufficient if absolute position measurements are available sporadically and at no particular time interval. Assuming that slippage is the primary source of error for a robot moving on homogeneous terrain, at any random sampling interval r_1 when an absolute fix is available, the total accumulated error in positioning $\varepsilon[r_1]$ up to that point can be expressed as the sum of the individual errors $S_c[k]$. According to (15), the error due to slippage is a function of the motor currents; therefore, the total accumulated error can be approximated as

$$\begin{aligned}\varepsilon[r_1] &= T_s \sum_{k=1}^{r_1} S_c[k] \\ &= T_s \xi \left(\sum_{k=1}^{r_1} I[k] - r_1 I_{\text{slip}} \right), \quad (I > I_{\text{slip}})\end{aligned} \tag{21}$$

where r_1 is a random sampling interval at which an absolute position is available.

The scale factor for slippage correction ξ can be expressed in terms of the other variables as follows:

$$\xi = \frac{\varepsilon[r_1]}{T_s \left(\sum_{k=1}^{r_1} I[k] - r_1 I_{\text{slip}}\right)}, \quad (I > I_{\text{slip}}). \tag{22}$$

When another absolute position is available at a second random time interval r_2, the positioning error $\varepsilon[r_2]$ can be calculated. As long as the robot is moving on the same terrain,

the scale factor for slippage correction, as well as the onset for wheel slippage, should remain the same, hence

$$\frac{\varepsilon[r_1]}{r_1 I_{\text{slip}} - \sum_{k=1}^{r_1} I[k]} = \frac{\varepsilon[r_2]}{r_2 I_{\text{slip}} - \sum_{k=1}^{r_2} I[k]}, \quad (I > I_{\text{slip}}). \tag{23}$$

Equation (23) can be solved for I_{slip} and for any pair of absolute positions collected at intervals r_p and r_{p+1} by identifying a value I_{slip} that satisfies the following equation:

$$\varepsilon[r_1]\sum_{k=1}^{r_2} I[k] - \varepsilon[r_2]\sum_{k=1}^{r_1} I[k] - \varepsilon[r_1] r_2 I_{\text{slip}} + \varepsilon[r_2] r_1 I_{\text{slip}} = 0, \quad (I > I_{\text{slip}}). \tag{24}$$

Note that in (24), I_{slip} appears in the third and fourth terms. I_{slip} is also implicit in the first and second terms because of the condition $I > I_{\text{slip}}$. For this reason, the solution is not straightforward, and numerical methods are necessary to solve it. To this end, we use a gradient descendent algorithm and start with an initial guess for I_{slip}. The algorithm then iteratively updates its value so that in every step, it incrementally moves in the direction that minimizes the left side of (24). Once I_{slip} has been estimated, the scale factor ξ can be computed directly using (22).

a) Experimental Results: For this experiment, we used a similar experimental setup to the one in Section I, including the SAPS. However, although the SAPS provides continuous absolute position data, we disregarded the continuous data and randomly selected five absolute positions at time intervals $r_1, r_2, \ldots, r_p, \ldots, r_5$. The only condition was that $r_1 < r_2 < \cdots r_p < \cdots r_5$. As before, the robot moved straight ahead at 60 mm/s, therefore, the positioning error $\varepsilon[r_p]$ can be computed as the difference between the position of the robot seen by the SAPS and the relative position estimated by odometry at time r_p

$$\varepsilon[r_p] = P_a[r_p] - P_r[r_p] \tag{25}$$

where

- P_a absolute position of the robot;
- P_r relative position of the robot estimated by odometry;
- r_p random sampling interval at which an absolute position is available.

After solving (24), I_{slip} is estimated, and with this result, (22) is applied to compute the scale factor ξ. Fig. 7(a) and (b) show the parameter-estimation percentage error of I_{slip} and ξ. We use as baseline (nominal value) the parameters obtained with the previous method as the robot moves forward.

In both cases, the accuracy of the estimation of the parameters I_{slip} and ξ improves with the traveled distance. That is because at the beginning of the experiment, the error due to slippage is still comparable to the resolution of the SAPS; however, as the robot moves, the positioning error becomes much larger than the resolution of the SAPS. For the results presented in Fig. 7, random data points r_p were chosen 7–12 s apart from each other.

2) Method III: Parameter Estimation Based on Induced Slippage in a Single Wheel: The main drawback of the two previously described methods for parameter estimation is that they require an absolute position reference. Such a reference, however, may not be available in all applications. For this reason, we developed a third method for parameter estimation. With this method, no absolute position reference is required at all. On the other hand, the method is only applicable to mobile platforms with four or more individually driven wheels.

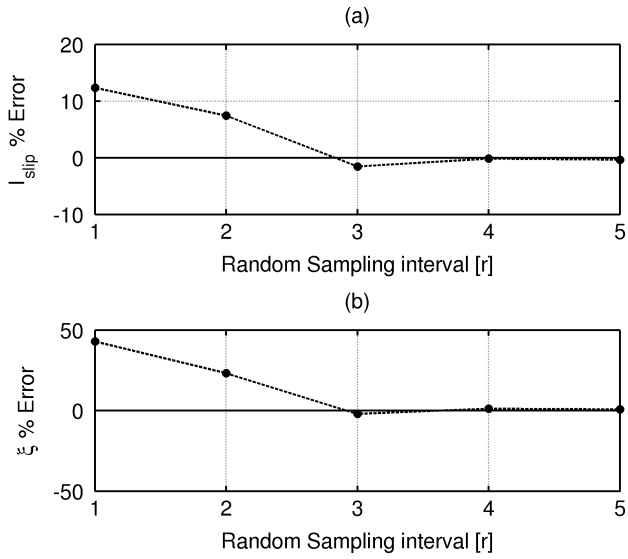


Fig. 7. Parameter-estimation percentage error using sporadically available absolute positions. (a) Onset current for wheel slippage. (b) Scale factor for slippage correction. Note that the horizontal axis does not represent time, but the index of the random sampling realization, when the absolute position was available. The absolute intervals were chosen randomly in an interval of 7–12 s apart from each other.

With this method, the robot is commanded to move all six wheels at a constant speed v. Then the speed of one of the wheels, v_s, is gradually increased. Slippage and slippage correction can be estimated by redefining (16) and (17) as follows:

$$i = 1 - v/v_s \tag{26}$$

$$S_c = v_s - v. \tag{27}$$

Once the slippage correction is established, it can be correlated with the motor currents to define $S_c(I)$. Due to noise in the measurements, a least-squares algorithm will have to be used to define $S_c(I)$.

For a robot with uneven weight distribution, this procedure should be repeated for every wheel that bears significant and different weight. However, if the amount of weight that a wheel supports is disproportionably small, it can be ignored, as in the case of the front wheels of our rover.

Since different loads on the wheels will result in different values for I_{slip} and ξ for each wheel, the final result must be computed as the average of the individual wheel contributions

$$\xi = \frac{1}{m}\sum_{p=1}^{m} \xi_p \tag{28}$$

$$I_{\text{slip}} = \frac{1}{m}\sum_{p=1}^{m} I_{\text{slip},p} \tag{29}$$

where $m = 4$ is the number of motors used to estimate the average current, and p is the motor index.

b) Experimental Results: The setup for this experiment was similar to the setup of the two other methods, and the speed of all six wheels was again 60 mm/s. Every 10 s, the speed of one of the center wheels was increased by 5 mm/s [see Fig. 8(a)].

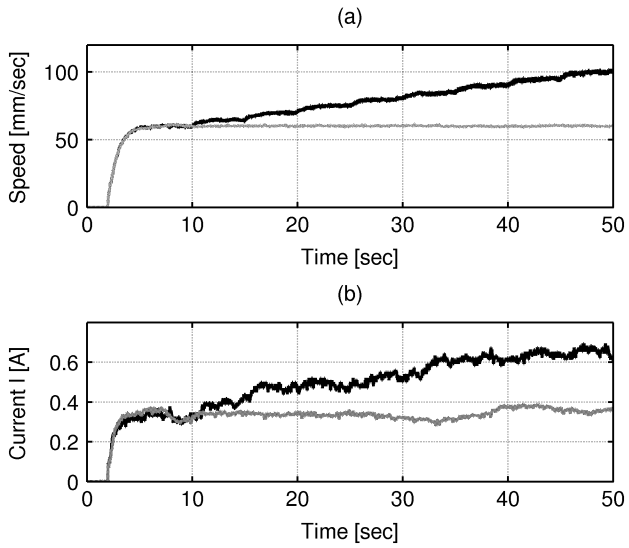


Fig. 8. Speed and motor current for Method III (single wheel being accelerated while all other wheels remain at constant speed). (a) Dark line: speed of the slipping wheel; gray line: average speed of the nonslipping wheels. (b) Dark line: motor current on slipping wheel; gray line: average current on nonslipping wheels.

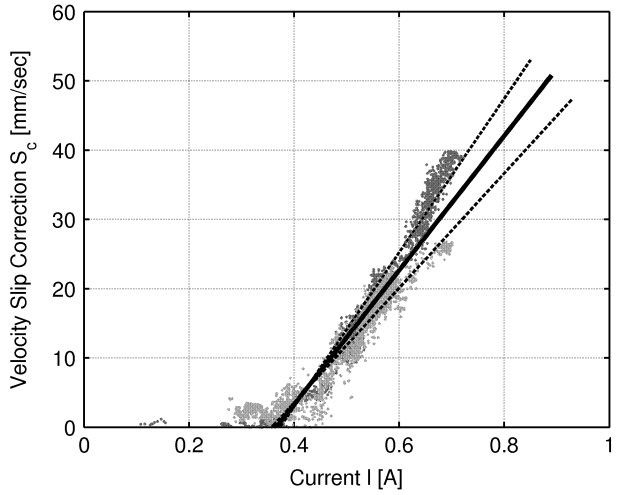


Fig. 9. Velocity slippage correction versus motor currents for Method III. Dark gray dots: one of the center wheels, light gray dots: one of the rear wheels. Black dashed lines: best line fit for each wheel. Black continuous line: solution found using Method I, continuously absolute position.

During this time, the current of the motors was recorded [see Fig. 8(b)], for subsequent use in defining the slippage-current line $S_c(I)$. This procedure continued until the slipping wheel reached a maximum speed of 100 mm/s, corresponding to 40% slippage. We confirmed using the SAPS that the average speed of the robot is determined by the majority of the wheels, and that the single wheel needs to slip in order to reach the higher commanded speed.

Using the velocity-slippage information and the corresponding motor currents, we defined the slippage-current line $S_c(I)$ (see Fig. 9). Because of the uneven weight distribution between the center and rear wheel pairs in our rover, we repeated the same procedure using one of Fluffy's rear wheels. The final parameters, I_{slip} and ξ, were obtained as the average of both results. The resulting single slippage-current line $S_c(I)$ coincides exactly with that line as determined by Method I. In summary, Table II lists the advantages and disadvantages of each of the three methods presented in this paper.

III. Slippage Correction (Icomp)

This section describes our approach to correcting wheel slippage-incurred odometry errors, based on the velocity slippage correction versus motor current function $S_c(I)$. We recall from the previous section that $S_c(I)$ is a linearized function defined by the parameters I_{slip} and ξ. These parameters can be found by any one of the three methods described in Section II.

In the practical implementation of (15), we compute the motor current I as the average of the motor currents [see (20)] of the rear and center wheel pairs. As noted earlier, the currents of the two front wheels are always disregarded, because those wheels bear only a small portion of the rover weight. We emphasize that although I is computed as the average current from four drive motors, the correction [see (30)] is applied only when all four motor currents individually exceed I_{slip}. One should note that when a wheel climbs up a rock, the motor current may also exceed I_{slip}, even though the wheel is not slipping. However, the AWS flag is only raised if the currents of all four motors of the center and rear wheel pairs exceed I_{slip} in the same sampling interval.

As a further refinement, we take into account that when the rover climbs up a slope, less of its weight acts as a force normal to the surface. As a result, slippage will occur at lower levels of wheel torque. We account for this fact by lowering the current threshold that signifies the onset of slippage, I_{slip}, by an empirically determined factor C_θ. Therefore (15) can be rewritten as

$$S_c[k] = \begin{cases} \xi(I[k] - I_{\text{slip}}[k] + C_\theta \theta_r[k]), \\ \qquad I_p[k] > (I_{\text{slip}} - C_\theta \theta_r[k]) \quad \forall p \\ 0, \quad \text{otherwise} \end{cases} \tag{30}$$

where p is the motor index (only center and rear wheel pair motors are being considered), and θ_r is the rover's pitch angle as derived by the onboard IMU.

The logical extension to these improvements would be adding compensation for changes in the roll angle. However, we must be careful when doing so, because changes in the roll angle may also be accompanied by lateral slippage, which cannot be corrected by our method. We should point out that for relatively small angles, the improvement resulting from pitch compensation is very small, and may be omitted. This can be explained by the fact that the normal weight [left side of (11)] changes as a function of the cosine of the tilt angle, therefore, only large tilt angles would produce a significant change in (11).

In the remainder of this section, we illustrate the slippage-correction process with the help of data and results from a typical experiment. In this experiment, we used our SAPS to measure ground-truth speed, while Fluffy was commanded to move straight forward on sand and at a constant speed of 60 mm/s. Slippage was intentionally induced by holding the robot back manually during varying periods of time (see Fig. 10).

As was shown earlier in Fig. 2, our system comprises two modules: the AWSD module and the iComp. During operation, the AWSD module monitors the four motor currents of the

TABLE II
COMPARISON OF PARAMETER-ESTIMATION METHODS

	Advantages	Disadvantages
Theoretical Approach	Provided that the soil parameters are known, this is the fastest method, since it does not need to run any calibration procedure and the parameters can be preloaded.	Requires previous knowledge of the area were the robot is going to be used. Performance depends on the accurate knowledge of many parameters, including terrain parameters. Is valid as long as the robot moves on the same surface.
Method I: Continuous Absolute Positioning	Can be very accurate. Can adjust its parameters if the robot moves to another surface. Parameters can be estimated in a short period of time.	Requires continuous external absolute positioning system. Absolute positioning system must be accurate and must have good resolution, since the absolute positions will be used to compute speed. Inducing slippage is not a trivial task in autonomous operation (in this paper we induced slippage by manually holding the robot back).
Method II: Sporadic Absolute Positioning	Accuracy improves with the traveled distance Absolute positioning system does not need to be very accurate. Can adjust its parameters if the robot moves to another surface.	Requires external absolute positioning system Depending on the accuracy of the absolute positioning system, the robot may need to travel long distances (and wait longer) in order to get a good parameter estimation.
Method III: Single Wheel Induced Slippage	Self-contained If basic assumption that the robot moves at the speed of the majority of the wheels holds, this method can be very accurate. Can adjust its parameters if the robot moves to another surface.	Requires four or more independently driven wheels. The procedure may need to be performed with more than one wheel if the load is not distributed evenly, therefore it can take a long time.

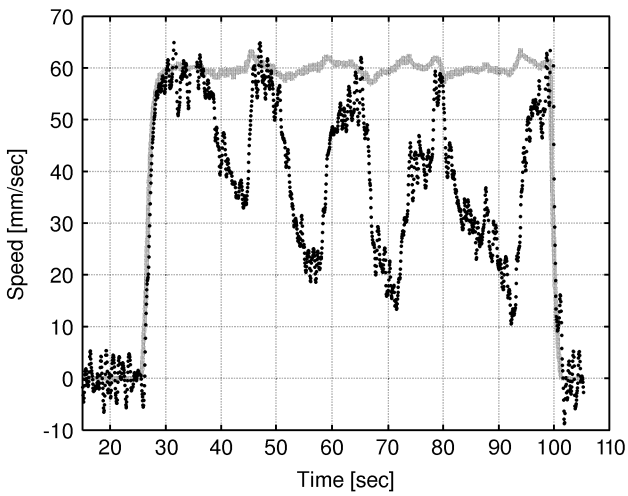


Fig. 10. Speed of an intermittently slipping rover measured by our external absolute position sensor (black dots), and wheel encoders (gray solid line).

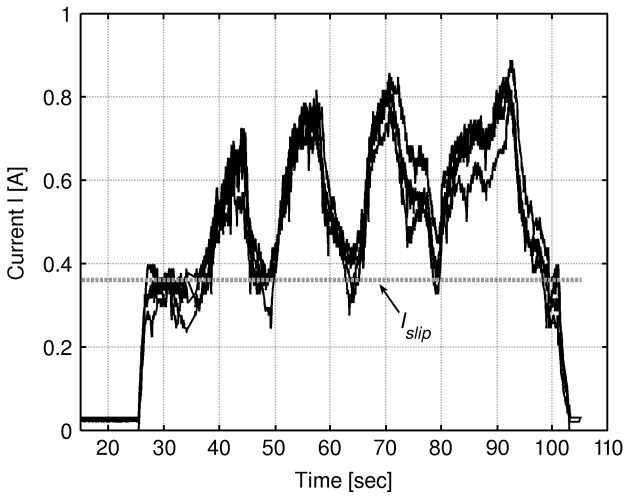


Fig. 11. Solid lines: motor currents for the four relevant wheels, during the experiment of Fig. 10. Dashed gray line: threshold for the onset of wheel slippage, denoted $I_{\rm slip}$.

center and rear wheel pairs. Motor currents for this typical experiment are plotted in Fig. 11. If and only if all four motor currents exceed $I_{\rm slip}$ (shown as the dotted line), AWS is flagged. When the AWS flag is high, then iComp applies (30) to the average motor current plotted, and computes the slippage-correction values plotted in Fig. 12. The momentary slippage-correction value is then subtracted from the linear speed measured by the wheel encoders as

$$v_c[k] = v_r[k] - S_c[k] \tag{31}$$

where v_c is the final, corrected speed of the rover.

Fig. 13 shows that speed plotted over the SAPS-measured true speed, the improvement over the uncorrected speeds shown in Fig. 10 is obvious and significant.

With the corrected speed of the robot, we can calculate the corrected traveled distance d_c

$$d_c[k] = v_c[k]T. \tag{32}$$

One should note that when a wheel climbs up a rock, the motor current might also exceed $I_{\rm slip}$, even though the wheel is not slipping. However, the AWS flag is only raised if the currents of all four motors (center and rear wheel pairs) exceed $I_{\rm slip}$ in the same sampling interval. Thus, a false positive (i.e., flagging wheel slippage while there actually is none) may happen only if all four wheels climb up rocks at the same time, an event quite unlikely in practice.

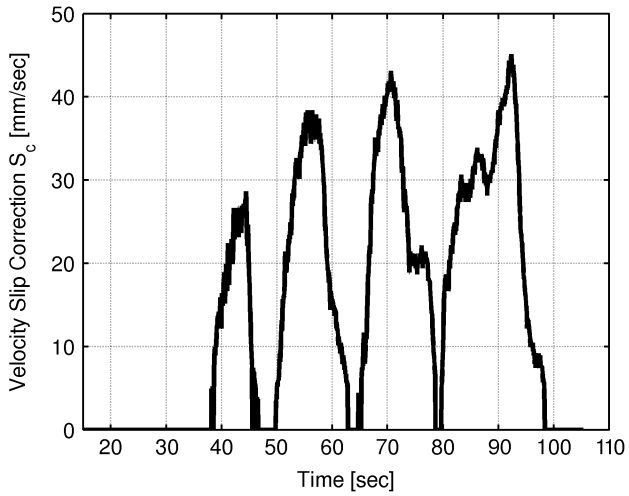


Fig. 12. Velocity slippage compensation for the experiment of Fig. 10 computed according to the linear function $S_c(I2)$ (30).

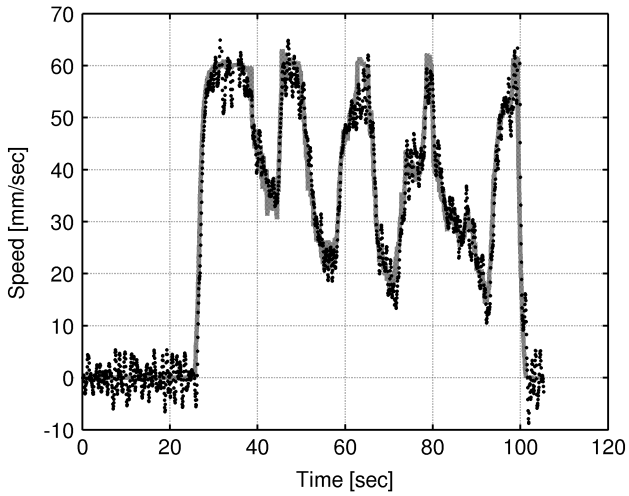


Fig. 13. After applying the slippage compensation of Fig. 12 to the encoder readings of Fig. 10, the speed of the robot measured by our external absolute reference (black dots), and wheel encoders after slippage compensation (gray solid line), match very well.

IV. Experimental Results

In this section, we present experimental results aimed at assessing the overall effectiveness of the iComp method. In all experiments described in this section, we performed some runs on homogenously sandy terrain, and other runs on nonhomogeneous sandy terrain that had fist-sized rocks embedded in the sand. We created two experimental arenas.

1) A straight-line path over sandy terrain, which had just one single terrain feature: a sandy mound. For different runs, we changed the height of the mound in fixed increments, in order to assess the effect of the controlled slope on the accuracy of the iComp method.
2) A closed-loop path inside a large sandbox, in which the robot encountered several mounds of sand, as well as 90° turns in the four corners of the rectangular path, contributed to the overall errors.

In all cases, compensation was applied using (30), that is, using the improved equation that compensates for changes in the pitch angle. In the experiments described in Section IV, slippage was not induced by holding the robot back manually. Rather, slippage was always the result of natural causes, such as driving uphill on a sandy slope.

A. Straight-Line Experiments

We performed two sets of experiments along a 4-m straight path on sand, which included a single sand mound of various heights. In the first set of experiments, we created the mounds from loose sand only. Depending on the height of the mount, this would create more or less slippage. In the second set, we embedded rocks in the sandy surface of the mounds. Doing so significantly reduced the amount of slippage, even for the higher mounds. The purpose of including the rocks was to create conditions in which motor currents surpassed I_{slip}, but without causing AWS. We recall that condition $I > I_{\text{slip}}$ is overly conservative, and may be true when a wheel encounters a rock. However, we also recall that AWS is only flagged if indeed *all* wheels match the condition $I > I_{\text{slip}}$ at the same time. The nominal speed was 60 mm/s in all runs.

In both sets of experiments, we ran Fluffy over a single mound, which had different heights in different runs: 10, 15, 20, and 30 cm. The nominal travel distance was $D = 4$ m. Because of the 3-D nature of the terrain, Fluffy employed our FLEXnav dead-reckoning system (mentioned in Section I and shown schematically in Fig. 1), instead of plain odometry, to determine the end of the 4-m trip. The distance between Fluffy's final stopping position and the nominal target at the 4-m mark constitutes the position error y_r of a run. Notice that in this type of experiment, we are only concerned with the error along the longitudinal direction. This is because a small misalignment can cause relatively large errors in the lateral direction, which may appear as outliers.

For each mound height, we performed m runs and we computed the average error Y_e as the average of absolute errors

$$Y_e = \frac{1}{m}\sum_{p=1}^{m}|y_r|. \tag{33}$$

Finally, we expressed the average error as a percentage of travel distance, E, according to

$$E[\%] = 100\frac{Y_e}{D}. \tag{34}$$

We disregarded any errors in the lateral direction, since these errors are mainly the result of misaligning the initial orientation of the robot.

1) Experimental Set 1a: Pure Sand Mounds: The results of the experiments with pure sand mounds are plotted in Fig. 14(a), and a numeric summary is shown in Table III. The circular markers show runs with FLEXnav, but without iComp, and the square markers show runs using FLEXnav with iComp. As is evident from comparing the error with "FLEXnav only" with that of "FLEXnav + iComp," our slippage-compensation method iComp yields a reduction in position errors of as much as one to two orders of magnitude. Indeed, the more slippage there is (higher sand mounds produce more slippage), the greater the improvement achieved with our iComp method.

2) Experimental Set 1b: Sand Mounds With Rocks: The iComp method assumes that any motor current in excess of I_{slip}

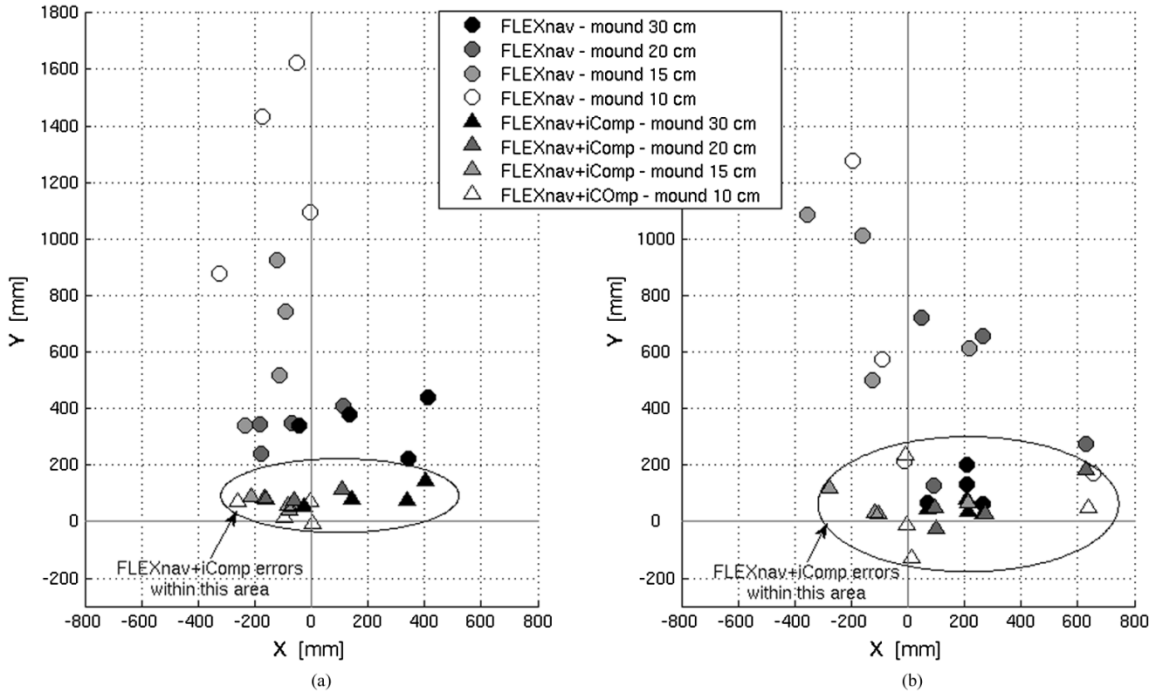


Fig. 14. Final positioning error for 4-m straight-line runs across one sandy mound. The mound had different heights in different runs and consisted of (a) sand only; (b) sand with embedded rocks.

TABLE III
SUMMARY OF POSITION ERRORS AT THE END OF EXPERIMENT 1, A 4-m STRAIGHT-LINE PATH WITH MOUNDS OF DIFFERENT HEIGHTS, WITH AND WITHOUT EMBEDDED ROCKS

	Pure sand mounds				Sand mounds with embedded rocks			
	FLEXnav only		FLEXnav + iComp		FLEXnav only		FLEXnav + iComp	
Mound Height	Error [mm]	Error [%]	Error [mm]	Error [%]	Error [mm]	Error [%]	Error [mm]	Error [%]
30-cm	1,258	31.4	20	0.5	560	14.0	106	2.7
20-cm	632	15.8	49	1.2	803	20.1	62	1.6
15-cm	336	8.4	48	1.2	445	11.1	72	1.8
10-cm	347	8.7	72	1.8	115	2.9	48	1.2

is either the result of wheel slippage or of the wheel climbing up a rock. To distinguish between these two possibilities, the iComp method flags AWS only if all four weight-bearing wheels (center and rear wheel pairs) simultaneously meet the condition $I > I_{\text{slip}}$. In order to verify the effectiveness of this approach, we performed the experiments of Set 1b. In this set of experiments, fist-sized rocks were embedded in the sand mounds of Set 1a and at other places along the 4-m straight test path. All other conditions were the same as in Experimental Set 1a.

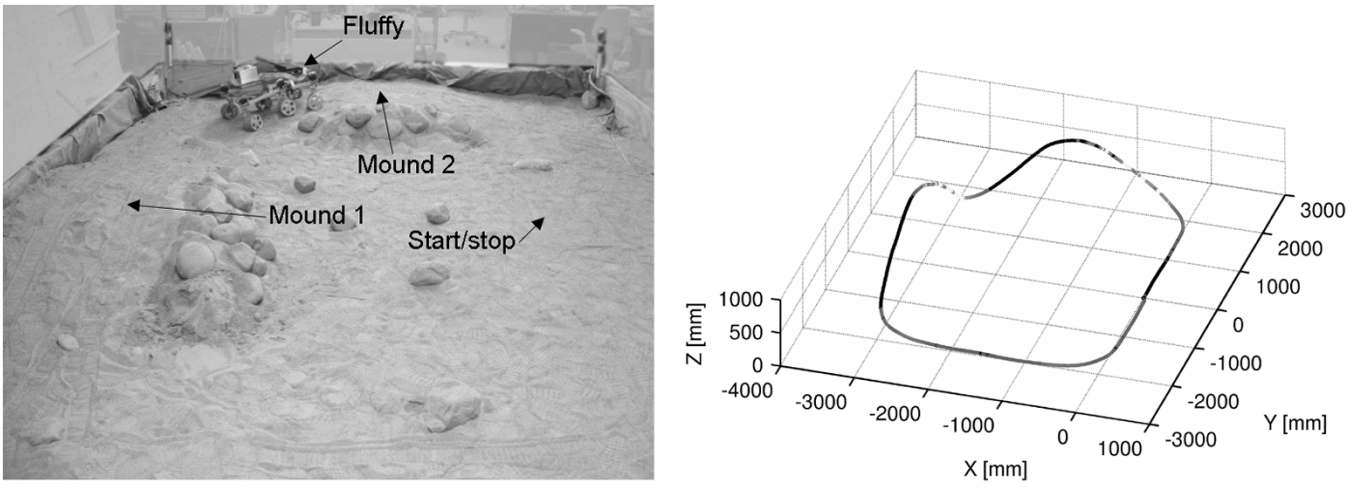
The results of the experiments of Set 1b are plotted in Fig. 14(b), and a numeric summary is given in Table III. Comparing the errors of the "FLEXnav + iComp" columns for Experimental Sets 1a and 1b, it is evident that the errors in Set 1b are slightly larger. That suggests that our method for distinguishing motor current increases due to wheel slippage from those due to climbing up a rock is not perfect. However, even with that imperfection, the iComp method still performed up to one order of magnitude better than FLEXnav only.

B. Closed-Loop Experiments

In this section, we provide comprehensive experimental results obtained with different navigation features on different terrains. Specifically, we created in our sandbox two different terrains (Fig. 15).

1) Sand-covered terrain with two small sand mounds and with fist-sized rocks embedded in the sand mounds, so that the top half of each rock was sticking out of the sloped surface. On this terrain, slippage was minimal.
2) Sand-covered terrain with two large sand mounds and no rocks. This terrain produced much slippage.

We refer to sets of multiple runs on each of the two terrains as one experiment. Each of the two experiments consisted of four runs in clockwise (cw) and four runs in counterclockwise (ccw) directions. Running in both cw and ccw directions is very important. If a mobile robot is tuned and tested in only one direction, then it is highly likely that the key parameters are tuned in such a way that dominant systematic error sources (such as those resulting from inaccuracies in measuring track width and wheel diameters) compensate for each other, and great accuracy is *seemingly* achieved. However, if the robot is run in the other direction, then those error sources no longer compensate for each other but rather add up, resulting in potentially large errors. A detailed analysis of this subject is given in [18]. That paper also explains that a third significant error, the error in



(a) (b)

Fig. 15. Experimental setup for the closed-loop experiments. (a) Fluffy driving down a steep slope. (b) Actual 3-D trajectory of the rover during a typical single-lap run. It shows the light-gray colored parts of the trajectory in which AWS was flagged.

TABLE IV
EXPERIMENTAL CONDITIONS AND RESULTS FOR THE THREE EXPERIMENTS DESCRIBED IN THIS SECTION. SEE FIG. 16 FOR DETAILED RESULTS. NOTE: "cw" STANDS FOR "CLOCKWISE" AND "ccw" STANDS FOR "COUNTERCLOCKWISE"

		FlexNav only				FlexNav + iComp			
Terrain features	**Dir.**	**X_e [mm]**	**Y_e [mm]**	**Error [mm]**	**Error [%]**	**X_e [mm]**	**Y_e [mm]**	**Error [mm]**	**Error [%]**
Two moderate slopes with fist-sized rocks embedded in the sand to minimize wheel slippage.	cw	172	225	283	0.7	264	73	274	0.6
	ccw	267	227	350	0.8	290	245	380	0.8
Two steep slopes 15 and 35 cm high. Slopes made of sand only, resulting in substantial slippage.	cw	952	1601	1862	4.3	128	122	177	0.4
	ccw	742	1015	1258	2.9	49	254	259	0.6

determining the effective wheel radius, can be eliminated in a trivial fashion prior to the cw and ccw experiments. To do so, the robot is driven along a straight line. Then, when stopped, the experimenter compares the actual travel distance with the travel distance reported based on odometry. The difference between those two measurements shows the effective wheel radius error, which can then be corrected in software.

Running closed-loop experiments allows performing long-duration experiments in relatively small areas, such as in our sandbox. When designing the experiments, each of the mounds was located on *adjacent* sides of the almost rectangular closed loop [Fig. 15(b)]. This configuration assures that errors occurring on one side of the rectangular path were not compensated by similar errors occurring on the *opposite* side. In those experiments in which only little slippage occurred, the final positioning error after completing one loop was often too small and hard to measure. In order to amplify the errors, we had Fluffy complete *three* loops before measuring the final positioning error.

In each run, Fluffy traveled autonomously along a preprogrammed, near-rectangular path. As explained above, each run consisted of three uninterrupted loops, resulting in a total travel distance of $D = 43$ m per run, and a total of 1080 degrees of turning. The speed in all runs was 60 mm/s, and the rover did not stop before turning.

Fluffy started each run at a marked location (0, 0) and at the end of the run stopped at (what it "thought" was) the same position (0, 0). The discrepancy between the actual stopping position and the starting position, measured with a tape measure, is the so-called "return position error."

In each run, we collected data from all onboard sensors. Then, in postprocessing, we computed the final positioning error. As in the straight-line experiment, position estimation was based on FLEXnav, not on plain odometry. The computation of return position errors was performed in two different ways, FLEXnav alone and FLEXnav combined with our iComp slippage detection and correction method. Table IV shows the results of the two experiments in matrix form.

The *average absolute positioning errors* X_e and Y_e were computed as

$$X_e = \frac{1}{m}\sum_{p=1}^{m} |x_r| \tag{35}$$

$$Y_e = \frac{1}{m}\sum_{p=1}^{m} |y_r| \tag{36}$$

where

x_r *return* position error in X-direction;
y_r *return* position error in Y-direction;
$m = 4$ number of runs per direction and per terrain.

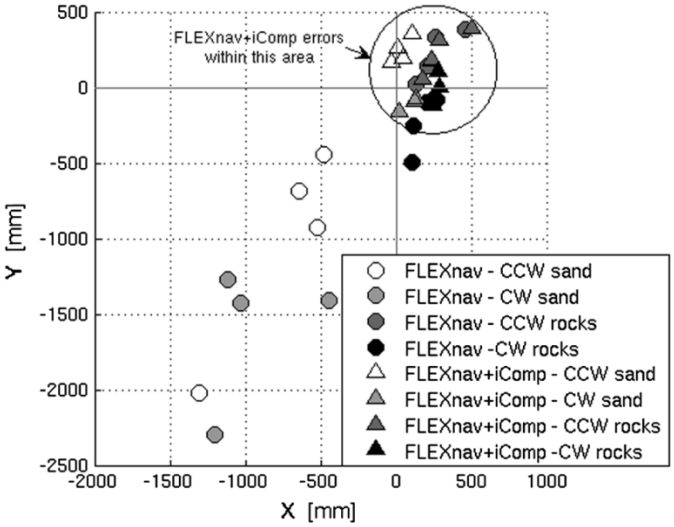


Fig. 16. Return position errors for the closed-loop experiments of Fig. 15. "cw" stands for clockwise, and "ccw" stands for counterclockwise.

The absolute error E was computed as

$$E = \sqrt{X_e^2 + Y_e^2} \tag{37}$$

and expressed as a percentage of total travel distance D using

$$E[\%] = 100\frac{E}{D}. \tag{38}$$

Fig. 16 shows the return position error for each run in graphical form.

V. Conclusion

In this paper, we introduced a method for detecting and correcting odometry errors caused by AWS in planetary rovers and other robots with multiple, independently driven wheels. The method is based on the relationship between electric currents in the drive motors and the wheel/terrain interaction. We introduced a linearized function that approximates this relationship, and we proposed three different methods for estimating the two parameters of this linearized function. Each of the three parameter-estimation methods is applicable for a different application domain. Methods I and II are useful in applications where ground-truth speed data is available continuously or sporadically, respectively. Method III is useful even if no ground-truth speed data is available at all, provided the robot has at least four independently driven wheels. The experiments described in this paper show that the three parameter-estimation methods define linearized slippage-current functions that match the true slippage-current relationship for the test vehicle driving on loose sand quite well. In a real application, the presented parameter-estimation methods can be applied repeatedly and in real time, in order to account for varying terrain conditions.

The paper also shows how the linearized slippage-current function, defined by any one of the parameter-estimation methods, can be used to correct for odometry errors due to slippage on the same terrain. Comprehensive experiments demonstrated the overall effectiveness of our iComp method for odometry error detection and correction on sandy terrain. Our method works also on nonhomogeneous terrain with occasional rocks embedded in the sand. While on such terrain, our system cannot correct odometry errors incurred by wheels slipping on rocks, it avoids applying slippage correction that is correct for sand when wheels drive over rocks.

As the results obtained with our Mars-rover-type platform show, odometry errors caused by wheel slippage during the traverse of arbitrarily high sand mounds are effectively reduced by our method. Errors were reduced by up to one order of magnitude when compared with conventional dead-reckoning on the same terrain. Even under conditions of extensive AWS, the iComp method kept dead-reckoning errors to well under 1% of the total travel distance.

References

[1] L. Ojeda and J. Borenstein, "Methods for the reduction of odometry errors in over-constrained mobile robots," *Auton. Robots*, vol. 16, pp. 273–286, May 2004.

[2] S. Sreenivasan and K. Waldron, "Displacement analysis of an actively articulated wheeled vehicle configuration with extensions to motion planning on uneven surface," *J. Mech. Des.*, vol. 118, no. 2, pp. 312–317, 1996.

[3] G. Bekker, *Theory of Land Locomotion*. Ann Arbor, MI: Univ. Michigan Press, 1956.

[4] ——, *Off the Road Locomotion*. Ann Arbor, MI: Univ. Michigan Press, 1960.

[5] ——, *Introduction to Terrain-Vehicle Systems*. Ann Arbor, MI: Univ. Michigan Press, 1969.

[6] J. Wong, *Theory of Ground Vehicles*. New York: Wiley-Interscience, 2001.

[7] K. Iagnemma and S. Dubowsky, "Vehicle wheel-ground contact angle estimation: With application to mobile robot traction control," in *Proc. 7th Int. Symp. Adv. Robot Kinematics*, 2000, pp. 137–146.

[8] ——, *Mobile Robots in Rough Terrain*, ser. Tracts in Advanced Robotics. Berlin, Germany: Springer, 2004.

[9] D. Apostolopoulos, "Analytical configuration of wheeled robotic locomotion," Ph.D. dissertation, Robotics Inst., Carnegie Mellon Univ., Pittsburgh, PA, Apr. 2001.

[10] K. Iagnemma, S. Kang, H. Shibly, and S. Dubowsky, "On-line terrain parameter estimation for planetary rovers," *IEEE Trans. Robot.*, vol. 20, no. 5, pp. 921–927, Oct. 2004.

[11] K. Yoshida and H. Hamano, "Motion dynamics of a rover with slip-based traction model," in *Proc. IEEE Int. Conf. Robot. Autom.*, Washington, DC, May 2002, pp. 3155–3160.

[12] E. Baumgartner, H. Aghazarian, and A. Trebi-Ollennu, "Rover localization results for the FIDO rover," in *Proc. SPIE Conf. Sensor Fusion, Decentralized Control, Auton. Robot. Syst. IV*, vol. 4571, Newton, MA, 2001, pp. 34–44.

[13] E. Tunstel, T. Huntsberger, A. Trebi-Ollennu, H. Aghazarian, M. Garrett, C. Leger, B. Kennedy, E. Baumgartner, and P. Schenker, "FIDO rover system enhancements for high-fidelity mission simulations," in *Proc. 7th Int. Conf. Intell. Auton. Syst.*, Marina del Rey, CA, Mar. 2002, pp. 349–356.

[14] L. Ojeda and J. Borenstein, "FLEXnav: Fuzzy logic expert rule-based position estimation for mobile robots on rugged terrain," in *Proc. IEEE Int. Conf. Robot. Autom.*, Washington, DC, May 2002, pp. 317–322.

[15] D. Lindgren, T. Hague, P. Probert Smith, and J. Marchant, "Relating torque and slip in an odometric model for an autonomous agricultural vehicle," *Auton. Robots*, vol. 13, no. 1, pp. 73–86, Jul. 2002.

[16] Z. Janosi and B. Hanamoto, "Analytical determination of drawbar pull as a function of slip for tracked vehicles in deformable soils," in *Proc. 1st Int. Conf. Terrain-Veh. Syst.*, Torino, Italy, 1961, pp. 707–736.

[17] J. Wong and A. Reece, "Prediction of rigid wheel performance based on the analysis of soil-wheel stresses, Part I and part II," *J. Terramech.*, vol. 4, no. 1, pp. 81–98, 1967.

[18] J. Borenstein and L. Feng, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE Trans. Robot. Autom.*, vol. 12, no. 6, pp. 869–880, Dec. 1996.

Lauro Ojeda (M'02) received the Bachelor's degree in electronic engineering and the M.S. degree in electronics from the Army Polytechnic School, Quito, Ecuador.

He is currently a Research Investigator with the Mobile Robotics Laboratory, University of Michigan, Ann Arbor. His area of expertise is the development of accurate positioning systems for mobile robots. He has worked on several projects sponsored by DARPA, DOE, and NASA. He is the key developer of the algorithms and techniques described in this paper.

Daniel Cruz (M'04) obtained the Bachelor's degree in electronics engineering from the Pontificia Javeriana University, Bogota, Colombia in 2001. He is currently working toward the M.S. degree in electrical and computer engineering at Oklahoma State University, Stillwater.

He spent two years with the University of Michigan, Ann Arbor, as a Visiting Research Investigator. He played a part in the development of mechanical and electronic systems in Fluffy, as well as some of the techniques discussed in this paper.

Giulio Reina received the Laurea and the Ph.D. degree in mechanical engineering from the Politecnico of Bari, Bari, Italy, in 2000 and 2004, respectively.

He was with the University of Michigan Mobile Robotics Lab, Ann Arbor, MI, as a Visiting Scholar in 2002–2003. He is currently with the Department of Innovation Engineering, University of Lecce, Lecce, Italy. His research interests include ground autonomous vehicles, rough terrain mobile robot localization, and kinematic design of mobile robots.

Johann Borenstein (M'88–SM'05) received the B.Sc., M.Sc., and D.Sc. degrees in mechanical engineering in 1981, 1983, and 1987, respectively, from the Technion-Israel Institute of Technology, Haifa, Israel.

He has been a Research Professor and Head of the Mobile Robotics Lab with the University of Michigan, Ann Arbor, since 1987. His research interests include mobile robot position estimation and obstacle avoidance, and the design of novel robotic platforms. He has 8 patents and over 120 publications on mobile robotics.

Dr. Borenstein won the 1998 Discover Magazine Award for Technological Innovation in Robotics.