Kinematic Control of a Planetary Exploration Rover over Rough Terrain

Thomas J. Carlone*, Jon J. Anderson*, Joseph L. Amato†, Velin D. Dimitrov†, Taşkın Padır†
*Robotics Institute, Carnegie Mellon University
{tcarlone, jonjake@cmu.edu}
†Robotics Engineering Program, Worcester Polytechnic Institute
{j.l.amato, vdimitrov, tpadir}@wpi.edu

***Abstract*—Passive averaging suspensions have been proven highly effective on rovers for improving mobility by providing ground compliance. However, the passive degree of freedom poses an added challenge to the controls problem. This paper presents a controller design to increase the accuracy of straight line trajectories for rovers with passive suspension on rough terrain. The chosen approach uses only proprioceptive sensors, a 3D kinematic model, and a trivial ground plane estimator algorithm to adjust individual wheel velocities based on estimates of terrain slope. This has distinct advantages to other techniques that use global position sensors and dynamic models which inevitably lead to more complex and computationally intensive solutions. The proposed controller is simulated in Matlab and found to be successful through experiments conducted with ORYX 2.0, a planetary rover research platform. This paper presents the feedforward velocity controller design, simulations, and experimental results for validation.**

I. INTRODUCTION

A feedforward velocity controller is presented here to enhance the precision of straight line trajectories for a rover with passive averaging suspension on rough terrain. Planetary rovers such as Carnegie Mellon University's (CMU) Scarab [1], [2], NASA's Jet Propulsion Laboratory (JPL) Sample Return Rover (SRR) [3], NASA's Rocky 7 Rover [4], and CMU's Nomad [5] have demonstrated the effective uses of this suspension technique in terms of mobility and endurance. While such passive suspensions improve the mobility and terrain traversability for rovers, they pose complexity in controls with the added degree of freedom in the suspension system. Modeling the 3D kinematics of the suspension has been used in a variety of techniques to achieve more precise trajectory control [6], [7].

Zoe, another rover designed by CMU, is an example of a rover with passive suspension and passive steering that employs a 3D kinematic model as part of a controller to improve trajectories over uneven terrain [7], [8]. One aspect of the controller estimates terrain slopes at each wheel using the kinematic model under the assumption of only positive obstacles. Wheel velocities are then adjusted to compensate for the sloping terrain, aiming to reduce wheel slip and increase accuracy of executed trajectories. Simulation and field testing results showed notably improved performance in trajectory following, compared to controllers that did not compensate based on 3D kinematics.

978-1-4799-0652-9/13/$31.00

Fig. 1. ORYX 2.0, WPI's planetary exploration rover.

The presented controller expands on the motion controller for Zoe, specifically the techniques used for feedforward velocity scaling, and attempts to improve the trajectory control using a purely 3D kinematic localized approach. One limitation of Zoe's control approach and similar methods reported in literature is their reliance on global position feedback, or the assumption of positive obstacles [7]. Typically done through inertial measurement sensors or visual odometry, such methods suffer from drift or slow update rates, respectively [9]. Alternatively, approaches that rely on dynamic models are more computationally intensive or involve modeling complex interactions between the ground and the wheels [10]. To address these limitations we present a controller that relies only on proprioceptive feedback: inertial orientation data and feedback on the position of the averaging suspension members (rocker angle).

With knowledge of the 3D orientation of the rover chassis and position of the averaging suspension, a 3D kinematic model can be made to locate the wheel positions relative to the rover's chassis. This information is then used to find pseudo-global wheel heights, providing an estimation of the terrain profile. The ultimate goal of the controller is to minimize deviations in yaw and wheel slip, which would otherwise result from constant wheel speeds over uneven terrain.

While this approach can effectively be used with similar results to more complex methods, it has several limitations.

Mainly, it assumes no wheel slip and cannot verify forward progress through feedback. It also relies on the assumption that an ideal ground plane exists, and that at least one wheel is on this ground plane. The controller is more effective when more wheels are on the ground plane. Given the relative and localized nature of this approach such factors do adversely affect the controller. In this paper, an overview of the characteristics of ORYX 2.0, a 4-wheel skid steering rover with passive averaging suspension, is first presented. Then a Matlab simulation is performed to verify the potential usefulness of this controller. This is followed by discussion on the specific implementation of the feedforward velocity controller. Finally, results from testing the controller on ORYX 2.0 are reported.

II. OVERVIEW OF ORYX 2.0

ORYX 2.0 is a research mobility platform designed at Worcester Polytechnic Institute [11]. Aimed at operating in analog Martian and lunar environments, this rover has passive kinematic suspension with skid steering and is designed to operate on uneven terrain. Even though the passive averaging rocker-bogie suspension with skid steering is not beneficial in all cases [12], it provides ORYX 2.0 with the flexibility to handle a variety of terrain while maintaining simplicity of the mechanical design. ORYX 2.0 has a footprint of about 1 m by 1 m and with 31 cm diameter wheels is capable of traversing obstacles up to 20 cm high. Preliminary field testing with ORYX 2.0 found large deviations in yaw and wheel slipping when traversing uneven terrain, which inspired the development of this controller.

ORYX 2.0 is also equipped with a number of sensors. To measure the yaw, pitch, and roll of the chassis an InterSense InertiaCube3 is used. Feedback on the position of the suspension members is provided by a 12-bit absolute encoder. Finally, drive motors are controlled by Maxon's EPOS2 controllers, which also provide feedback on the wheels velocity, position, and current draw.

Unlike many skid steering rovers, ORYX 2.0 has each wheel driven by a separate motor, making it possible to control the velocity of all four wheels independently. This feature allows for the design of a velocity controller that can adjust wheel velocities according to estimates of terrain slopes. Kinematic parameters of ORYX 2.0 are used to simulate the feedforward velocity controller. After implementing the controller on ORYX 2.0, experiments are conducted to evaluate its performance.

A. Passive Averaging Suspension

Passive averaging suspension is commonly used to achieve ground compliance, ensuring that four wheels remain in contact with the ground, creating higher stability and better weight distribution. The averaging effect of the suspension is also useful as it halves the amount of pitch and roll the chassis would otherwise see without ground compliance.

Multiple techniques to implementing the differencing suspension including geared differentials and differential linkages can be utilized. ORYX 2.0 has a differential linkage system, depicted in Figure 2. This approach allows for the implementation of the suspension by placing it outside the chassis and in a small form factor relative to the chassis size. Figure 2 shows the chassis design and the rocker differencing suspension. The specifications led to the design choices for the wheel size (32 cm diameter, 10 cm width), wheel track (76 cm), wheelbase (65 cm), and suspension configuration. Moreover, in its final implementation the platform weighs 35 kg with center of gravity at (-0.635 cm, -0.18 cm, 11.3 cm defined in a coordinate frame that is at the robots absolute center projected on the ground with positive x-axis being the surge direction.

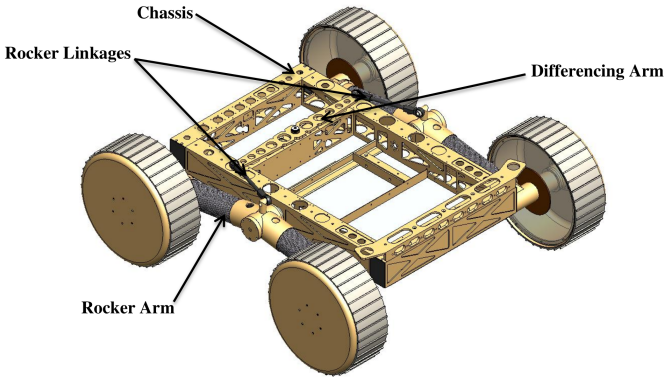

Fig. 2. Chassis Design and the Rocker Differencing Suspension

To meet the mobility requirements in terms of obstacle traversal, 35 cm diameter wheels which result in a ground clearance of 16 cm have been selected. Figure 1 depicts the rover design highlighting the low-profile chassis and relative wheel size. Since traversing obstacles half the diameter of the wheels is trivial, this configuration meets the design requirement of traversing 15 cm obstacles. However, given sufficient traction [13] and the design features of the passive suspension, the rover is capable of exceeding this limit.

III. 3D KINEMATIC MODEL

Rovers with passive suspension can be realized in several ways; however, they generally have similar kinematics where one degree of freedom can define the locations of the wheels relative to the chassis. In the case of ORYX 2.0, two rocker arms on either side of the rover passively rotate in opposite directions, constrained by a 6-bar mechanical linkage. Figure 3 shows the kinematic model of ORYX 2.0.

In characterizing the rover, three kinematic parameters are needed. One is the wheel base (ρ), defined as the distance between the left and right sides of the rover, illustrated by the red line connecting one rocker assembly to the other in Figure 3. The second is the length of the rocker arm (ℓ), defined as the distance between the rocker pivot and center of the wheel; illustrated by the distance between coordinate frames one and two. The third parameter is the acute angle between the wheels on one rocker arm pair (ϕ), defined as the angle between the rocker arm and the horizontal axis. For ORYX 2.0 these parameters are constant and have the following values:

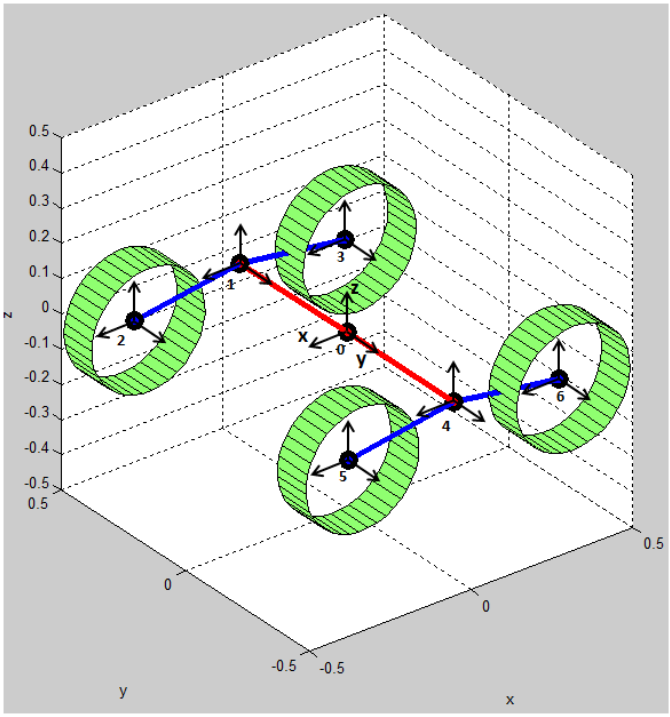

Fig. 3. Matlab kinematic model with coordinate frames.

$$\rho = 0.861 \text{ m}, \quad \ell = 0.330 \text{ m}, \quad \phi = 8.5^\circ$$

The four inputs to the kinematic model consist of the rocker arm angle (θ) and the yaw (α), pitch (β) and roll (γ) of the chassis. Although yaw is assumed to be zero, and thus ignored in the model. The rocker arm angle defines the position of the passive degree of freedom, locating the two suspension members. The yaw, pitch, and roll, are rotations around the z-axis, y-axis, and x-axis respectively, of coordinate frame zero. Using this information and some trigonometry, it is possible to calculate each wheel's pose relative to coordinate frame zero, which is the basis of the kinematic model.

$$H_{FR} = -\frac{\rho sin\gamma}{2} + \ell cos\gamma sin(\theta - \beta - \phi) \tag{1}$$

$$H_{BR} = -\frac{\rho sin\gamma}{2} - \ell cos\gamma sin(\theta - \beta + \phi) \tag{2}$$

$$H_{FL} = \frac{\rho sin\gamma}{2} - \ell cos\gamma sin(\theta + \beta + \phi) \tag{3}$$

$$H_{BL} = \frac{\rho sin\gamma}{2} - \ell cos\gamma sin(\theta + \beta - \phi) \tag{4}$$

Equations 1-4 summarize the kinematic model to determine the wheel heights. With the three known model parameters and three inputs from the orientation sensor and suspension encoder, the 3D kinematic model solves for the height of each wheel relative to coordinate frame zero in the center of the chassis.

IV. Feedforward Velocity Control

The concept of the feedforward velocity controller is to estimate terrain slopes for each wheel at discrete time steps, and adjust the velocities accordingly. Terrain slopes are estimated based on the 3D kinematic model and an ideal ground plane generation algorithm, which is deterministically dependent on the pitch, roll, and rocker angle. The flow diagram in Figure 4 summarizes the process of the feedforward controller. The proposed controller relies on the following core assumptions:

1) An ideal ground plane exists
2) At least one wheel is on or near the ideal ground plane
3) The rover's forward velocity is constant and equal to the desired forward velocity

A. Estimating the Terrain Profile

The presented kinematic model provides the locations of each wheel relative to the rover, making it insufficient to provide useful feedback on the terrain profile. In order to successfully adjust wheel velocities, an accurate estimate of terrain slopes is required. To achieve this, pseudo-global wheel heights are calculated using a combination of the 3D kinematic model and a ground plane estimator algorithm. These wheel heights are pseudo-global because they are with respect to the ideal ground plane, which inherently is relative to the rover and estimates the actual ground plane.

Algorithm 1: Estimating the ideal ground plane and adjusting wheel heights to form pseudo-global values

1) Determine largest magnitude wheel height based on kinematic model
2) If this is a positive value
   a) Determine the smallest wheel height
   b) Subtract this value from all wheel heights
   c) Repeat at next time step
3) If this is a negative value
   a) Determine the largest wheel height
   b) Subtract this value from all wheel heights
   c) Repeat at next time step

This proposed algorithm takes in four wheel heights which are outputted by the 3D kinematic model. It then converts these local values into pseudo-global wheel heights as described, and repeats at each time step when new sensory data updates the kinematic model. While trivial, this algorithm proves to be effective in a variety of simulated scenarios, is computationally efficient, and can handle both positive and negative obstacles.

B. Compensating Wheel Velocities

While pseudo-global terrain heights are calculated at each time step, an estimation of the forward progress is needed to calculate terrain slopes. This is typically done with odometry, visual odometry, or inertial measurement units. While odometry was available to provide feedback on wheel velocities a simplifying assumption of a constant rover velocity was made. This avoids problems that can occur with wheel slippage and the need to develop more complex dynamic models.

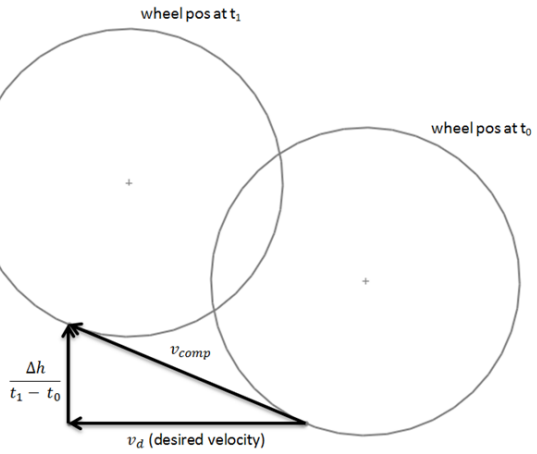

Fig. 5. Compensating wheel velocities

Under this assumption the controller takes in the desired forward velocity, and then adjusts each wheels velocity based on the desired forward velocity and the controllers estimation of the terrain slopes. Figure 5 illustrates how this estimation is made. The change in height is the difference between two pseudo-global heights calculated across one time step. Knowing the time between the measurements, this is then converted to an estimation of the wheels vertical velocity. The resulting compensated velocity (v_{comp}) is simply the magnitude of the estimated vertical velocity and assumed forward velocity, as seen in (5).

$$v_{comp} = \sqrt{\left(\frac{\Delta h}{t_1 - t_2}\right)^2 + {v_d}^2} \tag{5}$$

This is repeated for each of the four wheels, and at each discrete time step new velocities are calculated and fed forward to the each wheel. In all simulations and experiments, the discrete time step used is 0.05 seconds or a update frequency of 20 Hz.

In the implementation, pseudo-global wheel heights (terrain profile) are inputted into the controller, which compares these values to the previously inputted values at the last time step. It then takes in the desired forward velocity from the user and calculates the four new wheel velocities which are then sent to each EPOS2 motor controller respectively.

V. MATLAB Simulation Results

Initial Matlab simulations inputted a desired forward velocity and terrain, in the form of ground heights on either side of the rover. This data was then used to calculate the expected roll, pitch, and rocker angle that the rover would experience over the course of traversing the given terrain. The described controller is then simulated in Matlab, and the calculated desired wheel velocities are observed to visually confirm the success of the controller. Given the idealized nature of this simulation it could only confirm that wheel velocities were being adjusted correctly.

To gain a better understanding of the usefulness of the controller, Matlab simulations were performed again with data logged from Oryx 2.0. In the experimental setup shown in Figure 6, Oryx 2.0 is driven on flat ground over an obstacle while logging data. The obstacle is a trapezoid with dimensions shown in Figure 7. While traversing the trapezoid the roll, pitch, and rocker angle data are logged so that they can be used as the inputs for the Matlab simulation. Using this data the simulation uses the 3D kinematic model and ground plane estimation algorithm to calculate the pseudo-global heights of each wheel. This information is then used to calculate the resulting wheel velocities that the controller would generate as a result of the obstacle. The simulation results of the pseudo-global wheel heights and wheel velocities are shown in Figures 8 and 9 respectively.

All tests, including data logged for simulation purposes had a desired forward trajectory of 0.13 m/s, and an update rate of 20 Hz.

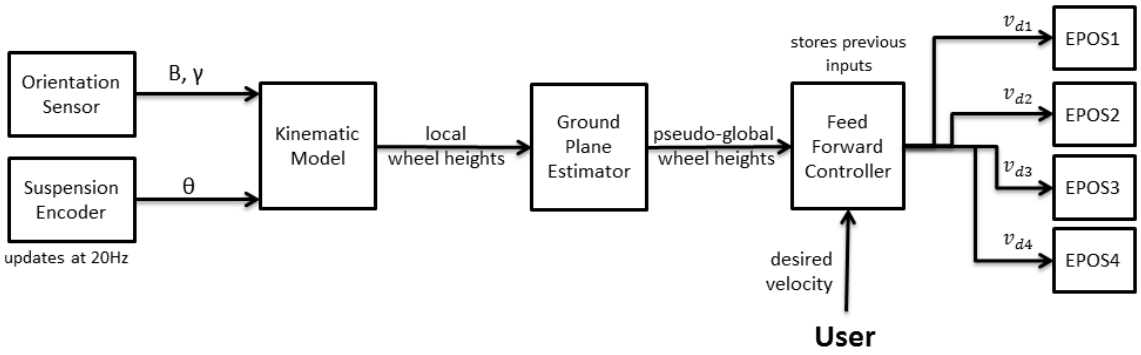

Fig. 4. Flow Diagram of Control Process

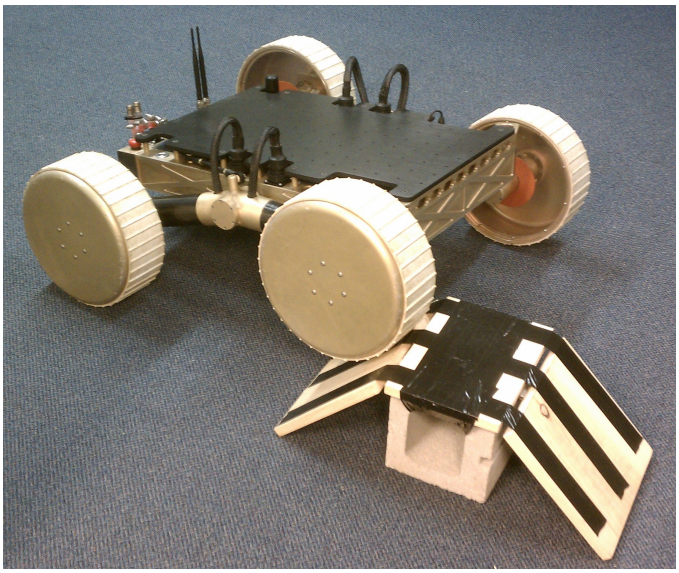

Fig. 6. Experimental setup with rover and trapezoidal obstacle

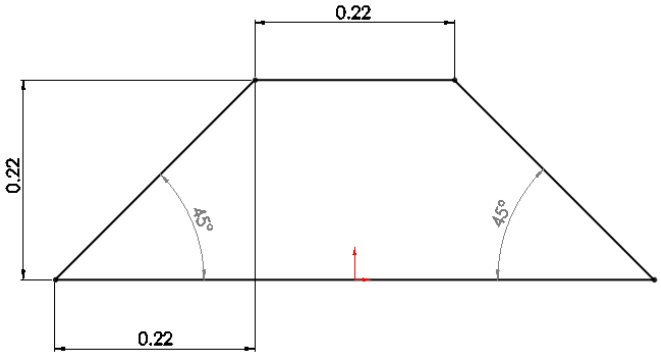

Fig. 7. Trapezoidal obstacle dimensions (meters)

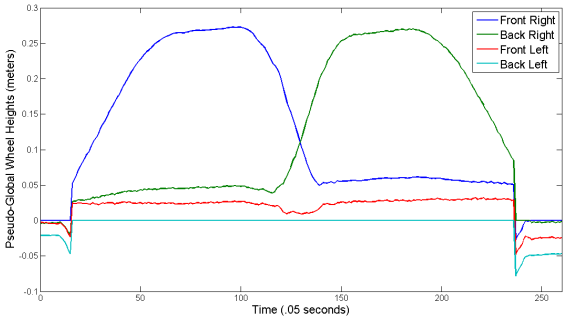

Fig. 8. Simulated pseudo-global wheel heights

The pseudo-global wheel heights qualitatively matched the simulation results, and show the profile of the trapezoid; first under the front right wheel and then under the back right wheel, with a short duration of overlap. The two left wheels remain mostly flat (on the ground). It is notable that the front left wheel is observed slightly above the ground, however,

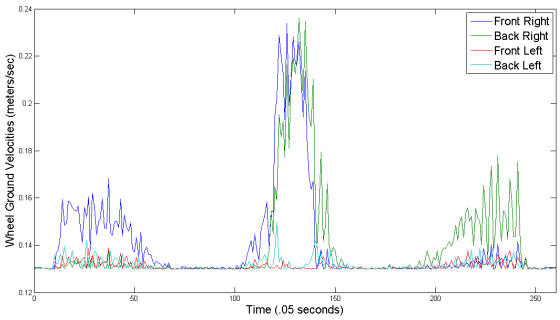

Fig. 9. Simulated wheel velocities

since the controller is only affected by slopes in the wheel heights this constant offset should have no adverse effects.

Resulting wheel velocities were also within reason. Since the slope of the trapezoid is 45 degrees the desired wheel velocity should be ∼41% faster than the desired speed (0.13 m/sec), which would result in a wheel ground speed of 0.183 m/s. As observed between t = 20 and t = 50 the desired velocity of the front right wheel (the wheel on the incline of the trapezoid), fluctuates between 0.14 m/s and 0.18 m/s, with an average of ∼0.16 cm/sec. This shows that the controller is effective in this scenario and only slightly underestimates the true slope of the terrain. A similar outcome is observed between t = 200 and t = 240. This is expected given the symmetry of the trapezoid.

During the two instances where the front right and the back right wheel are on the flat of the trapezoid respectively, there are almost no changes to the velocity. This is correct since none of the wheels are on sloping terrain. Between t = 100 and t = 150 there is a transition period where the front wheel is progressing down the end of the trapezoid while the rear wheel is beginning to climb the front. During this period each wheel is on a slope of 45 degrees, meaning the ideal compensated velocity is again ∼41% faster than the desired speed. In this case, however, the controller calculates moderately high values, about 0.21 m/s which could have a negative effect on the results.

VI. Experimental Results

With the success of the controller in the Matlab simulation, identical experiments were repeated with and without the controller in effect. The controller was implemented on ORYX 2.0 in the Robot Operating System (ROS) environment, and ran at 20 Hz. The main metric used to evaluate the success of the controller were changes in yaw (measured by the InertiaCube orientation sensor), since these are deviations from the straight line trajectory. An ideal controller would keep the yaw at 0 degrees, while operating without any controller produces increasing yaw inherent to traversing rough terrain.

Figure 10 represents the results from two trials, one with the controller, and one with no controller. The experiment was repeated multiple times and similar results were found in each case, with end yaw errors between 7 degrees and 9 degrees

for no controller, and between 1 degree and 3 degrees with the controller activated.

Overall, this large reduction in yaw achieved shows that the implemented controller performs well. In line with expectations from the simulation, the controller fails to compensate fully for the slope in the cases where the front wheel and rear wheel are on the slope at separate times. Occurring at the beginning and end of the traversal, the yaw is observed increasing at these times as a result. Similarly, the over compensation during the transition period resulted in a change in yaw in the negative direction. While ultimately an error, this change improved overall performance by balancing out the other two main sources of yaw error, and was repeatable in multiple trials.

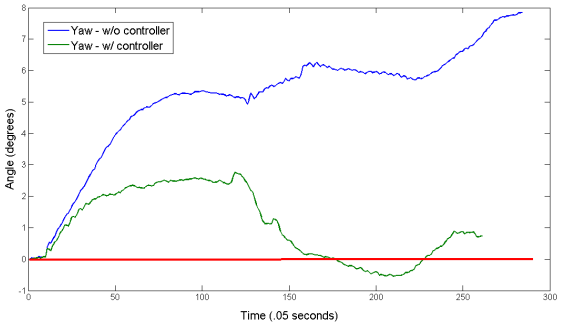

Fig. 10. Controllers effect on yaw

VII. Conclusions

The presented controller proved successful in improving accuracy of straight line trajectories over rough terrain, in a purely localized kinematic approach. Validated through Matlab simulation, the controller was also implemented on ORYX 2.0 and proved successful in experiments. While such experiments only investigated positive obstacles the kinematics and algorithm used behave identically for negative obstacles.

One limitation and source of error noted was in the apparent time delay between the location of the rover on the trapezoid and the compensated velocities. This time difference between when the desired velocities are calculated and when that command reaches the wheels could stem from a variety of sources. Future work will investigate the source of this delay, attempt to quantify it, and develop methods of minimizing it as a source of error. Experiments with different controller update frequencies faster than 20 Hz may have some benefits. Future work will also improve ground plane estimation in order to obtain a more accurate estimate of the terrain slopes, which would then improve the controller's accuracy. While the presented controller was successful in using minimal sensory data and a simple algorithm to improve straight line control, building upon this approach with more complex algorithms or possibly the use of other sensors could also improve accuracy and robustness of the controller.

Another issue is that velocity is calculated between two adjacent time steps. This means that when new sensor data is updated the desired wheel velocity can be calculated between that current time and the last time step. The result is that the calculated desired wheel velocity was for a time period that just elapsed. The limitation is that the desired velocities will always be one time step behind or require some means of estimating the next state of the rover to compensate for this effect. Field tests are planned to evaluate the controller performance in analog environments over long distances on rough terrain.

Acknowledgments

The authors would like to thank those who supported this work including NASA's Software, Robotics and Simulation Division, JSC; National Institute of Aerospace; Worcester Polytechnic Institute; Maxon Precision Motors, Inc.; The MathWorks, Inc.; Linemaster Switch Corporation; InterSense, Inc.; Axis Communications; Barnstorm Cycles; Tesla Motors; and igus inc.

References

[1] P. Bartlett, D. Wettergreen, and W. Whittaker, "Design of the scarab rover for mobility and drilling in the lunar cold traps," in *International Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2008.

[2] D. Wettergreen, S. J. Moreland, K. Skonieczny, D. Jonak, D. Kohanbash, and J. Teza, "Design and field experimentation of a prototype lunar prospector," *International Journal of Robotics Research*, vol. 29, no. 12, pp. 1550 – 1564, October 2010.

[3] K. Iagnemma, A. Rzepniewski, S. Dubowsky, T. Huntsberger, and P. Schenker, "Mobile robot kinematic reconfigurability for rough-terrain," in *Proceedings of the SPIE Symposium on Sensor Fusion and Decentralized Control in Robotic Systems III*, vol. 4196, 2000.

[4] R. Volpe, "Navigation results from desert field tests of the rocky 7 mars rover prototype," *The International Journal of Robotics Research*, vol. 18, no. 7, pp. 669–683, 1999. [Online]. Available: http://ijr.sagepub.com/content/18/7/669.abstract

[5] D. Wettergreen, D. Bapna, M. Maimone, and G. Thomas, "Developing nomad for robotic exploration of the atacama desert," *Robotics and Autonomous Systems*, vol. 26, no. 23, pp. 127 – 148, 1999, ¡ce:title¿Field and Service Robotics¡/ce:title¿. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889099800025

[6] K. Iagnemma, H. Shibly, A. Rzepniewski, S. Dubowsky, and P. Territories, "Planning and control algorithms for enhanced rough-terrain rover mobility," in *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, 2001.

[7] N. Seegmiller and D. Wettergreen, "Control of a passively steered rover using 3-d kinematics," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[8] M. Wagner, S. Heys, D. Wettergreen, J. Teza, D. Apostolopoulos, G. Kantor, and W. Whittaker, "Design and control of a passively steered, dual axle vehicle," in *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, 2005.

[9] P. Lamon and R. Siegwart, "3d position tracking in challenging terrain," *The International Journal of Robotics Research*, vol. 26, no. 2, pp. 167–186, 2007.

[10] L. Caracciolo, A. De Luca, and S. Iannitti, "Trajectory tracking control of a four-wheel differentially driven mobile robot," in *IEEE International Conference on Robotics and Automation*, vol. 4, 1999, pp. 2632–2638.

[11] J. Amato, J. Anderson, T. Carlone, M. Fagan, K. Stafford, and P. T., "Design and experimental validation of a mobile robot platform for analog planetary exploration," *Proc. IECON 2012: 38th Annual Conf. of the IEEE Industrial Electronics Society*, Montreal, CA, Oct. 25-28, 2012.

[12] T. Thueer, R. Siegwart, and P. Backes, "Planetary vehicle suspension options," in *Aerospace Conference, 2008 IEEE*, march 2008, pp. 1 –13.

[13] B. H. Wilcox, "Athlete: An option for mobile lunar landers," in *Aerospace Conference, 2008 IEEE*. IEEE, 2008, pp. 1–8.