Pose Estimation of Vehicles Over Uneven Terrain

Yingchong Ma and Zvi Shiller

Paslin Laboratory for Robotics and Autonomous Vehicles, Department of Mechanical Engineering and Mechatronics, Ariel University, 40700 Ariel, Israel.

Abstract

This paper presents a method for pose estimation of off-road vehicles moving over uneven terrain. It determines the contact points between the wheels and the terrain, assuming rigid contacts between an arbitrary number of wheels and ground. The terrain is represented by a 3D points cloud, interpolated by a B-patch to provide a continuous terrain representation. The pose estimation problem is formulated as a rigid body contact problem for a given location of the vehicle’s center of mass over the terrain and a given yaw angle. The contact points between the wheels and ground are determined by releasing the vehicle from a given point above the terrain, until the contact forces between the wheels and ground, and the gravitational force, reach equilibrium. The contact forces are calculated using singular value decomposition (SVD) of the deduced contact matrix. The proposed method is computationally efficient, allowing real time computation during motion, as demonstrated in several examples. Accurate pose estimations can be used for motion planning, stability analyses and traversability analyses over uneven terrain.

Keywords: Autonomous vehicles, Off-road motion planning, Pose estimation, Singular value decomposition

*This work was supported by grant 3-11025 of the Israeli Ministry of Science. The generous support of the Paslin Foundation to the Paslin Lab is also acknowledged.

Email address: yingchong.ma@hotmail.fr, shiller@ariel.ac.il (Yingchong Ma and Zvi Shiller)
1. Introduction

Current work on autonomous vehicles focuses mainly on ideal road conditions [1, 2]. Imperfect roads with bumps and potholes and any other irregular geometry may impose speed limits so as to ensure the vehicle’s stability during motion. The computation of such speed limits is usually treated in the context of off-road motion planning that accounts for vehicle dynamics and surface geometry [3, 4, 5]. Crucial to this computation is the determination of vehicle pose along the path, particularly when moving at high speeds on uneven terrain, since the rapid changes in the vehicle orientation might cause the vehicle to slide, tip-over, or lose contact with the ground. Furthermore, depending on the terrain geometry, not all wheels may be in contact with the ground at all times, which may severely affect the vehicle’s stability and traversability analyses. Developing an efficient algorithm to compute the vehicle pose along a specified path is the focus of this paper.

Early work on terrain-vehicle interaction focused on developing fundamental models of wheel-terrain interaction [6, 7]. More recently, the issue of computing the vehicle pose on uneven terrain has been addressed, and efficient numerical approaches were proposed [8, 9, 10, 11, 12, 13].

In [8] the vehicle pose is computed by projecting the position of each wheel on the elevation map and assuming that all wheels contact the ground. Since this method considers only the specific case where all wheels are in contact with the ground, it is not applicable to more general cases of multi-wheel vehicles moving over uneven terrain. In [9], a numerical optimization is used to minimize the distance between the wheel contact points and the terrain to estimate the vehicle pose. In [10] and [11] the vehicle is considered as a rigid body with four wheels: three rigid wheels and one compliant wheel. The deflection or extension of the compliant wheel is then computed to determine the vehicle pose. A serious drawback of these methods is that the terrain surface needs to be modeled analytically, which is not always practical. In [12], a fast pose estimation method is proposed that is based on a digital elevation map. As these methods consider only 4 contact points, they are not applicable for more general cases.

Another approach is proposed in [13], which formulates the problem as a “linear complementarity problem” (LCP) and solves it using the Lemke’s method [14]. This approach does not require an analytical presentation of the terrain surface and it allows the consideration of multiple contact points. The LCP approach is widely used in contact force computation for rigid bodies.
However, these algorithms often fail for large numbers of contact points [17], and they require recomputation of the governing equations after encountering singularities, which slows down computation time [18].

In this paper, the vehicle pose estimation problem over uneven terrain is formulated as a rigid body contact problem between the vehicle body and the terrain. A variety of approaches were developed for computing contact forces. Penalty methods are the simplest and earliest [19]. In these methods, a virtual stiff spring is attached between the contact points and the ground. However this often results in oscillations, and the spring constant is problem specific, which limits the use of these methods in general settings.

We compute the contact forces using Singular Value Decomposition (SVD) of the deduced contact matrix. The advantage of this method is that the procedure of computing the SVD of a matrix is robust [20], even for a singular matrix, which guarantees that a good approximation always exists even with computation errors and measurement noises. Moreover this approach is fast and simple [21].

The paper is structured as follows: Section 2 presents the terrain representation and the vehicle model. In Section 3, the vehicle pose estimation problem is formulated, and the contact matrix for contact forces computation is deduced. Then, in Section 4, contact forces are computed using the SVD based approach. Simulation results are shown thereafter in Section 5 to demonstrate the effectiveness of the proposed method.

2. Terrain Representation and Vehicle Model

2.1. Terrain and Path Representation

Reliable representation of terrain and traversability is key for most off-road vehicles tasks. In contrast to planar environments in which the terrain and obstacles can be represented as binary data, in 3D cases curvatures and vertical slopes of the uneven road surface must be taken into account for accurate terrain representation.

The majority of the mapping methods project depth information into digital elevation maps by connecting data points by piecewise continuous cells [22], which are only 1st order continuous. A search over such a terrain representation results in non-smooth paths that do not allow motion at continuous speeds.

In this paper, the terrain is represented by a cubic B-spline patch that is interpolated over a 3D point cloud, each point serving as a control point.
of the B-patch. The B-spline patch representation is smooth and 2nd order differentiable, which allows a continuous representation of curvature. This in turn allows traversing the terrain at continuous velocity profiles.

Fig. 1 shows a single B-Patch, constructed of 16 control points. A point \( p \) on the patch is a function of two parameters \( v \) and \( w \) [23]:

\[
p(v, w) = VRDR^TW^T, \tag{1}
\]

where

\[
V = [v^3, v^2, v, 1], \; v \in [0, 1],
\]

and

\[
W = [w^3, w^2, w, 1], \; w \in [0, 1],
\]

\( R \in \mathbb{R}^{4 \times 4} \) is a constant matrix specifying the type of spline used to construct the B-spline patch, and \( D \in \mathbb{R}^{4 \times 4} \) is a matrix of 16 control points. Each control point represents an \( xy \) location, usually defined as a grid point on a uniform mesh, and a \( z \) value that represents its height. A large surface may be represented by several patches.

![Figure 1: A single B-patch terrain with control points](image)

A smooth path over the terrain can be represented by parameterizing \( v \) and \( w \) with a single parameter \( u \):

\[
c(u) = V(u)RDRTW^T(u). \tag{2}
\]
Thus a line in the $v - w$ space is mapped into a continuous curve on the B-patch. A curve in the $v - w$ space is represented by the B-spline

$$\begin{bmatrix} V(u) \\ W(u) \end{bmatrix} = URC$$

where

$$U = [u^3, u^2, u, 1], \quad u \in [0, 1]$$

$C \in \mathbb{R}^{4 \times 4 \times 2}$ is an array of control points in the $v - w$ space, and $u$ is the independent parameter along the B spline. B splines are 2nd order continuous, which is essential when computing time-optimal trajectories. A typical path over the terrain is shown in green in Fig. 2.

![Figure 2: A B-spline path over a B-patch terrain](image)

2.2. Vehicle Model

The vehicle is assumed to be a rigid body with any number of wheels, each assumed to be rigid, as shown in Fig. 3. We represent the vehicle configuration by $r = (x, y, z, \alpha, \beta, \gamma)$, where $(x, y, z)$ are the coordinate of the center of mass of the vehicle body in the inertial frame and $(\alpha, \beta, \gamma)$ are the roll, pitch and yaw angles with respect to the vehicle body frame, respectively.

The number of wheels, their location relative to the vehicle, and the terrain profile, determine the vehicle pose and the contact points, as discussed
3. Pose Estimation

For a given position and orientation $h = (x, y, \gamma)$, we wish to find the remaining configuration $q = (z, \alpha, \beta)^T$. The vehicle pose is calculated by dropping the vehicle vertically at the configuration $h$ under the gravitational force until the vehicle touches the ground, and the gravitational force and the contact forces reach static equilibrium. The wheels in contact are determined from their configuration, location and the ground topography.

Since $h = (x, y, \gamma)$ is fixed, the vehicle motion downward is of three degrees-of-freedom $q = (z, \alpha, \beta)^T$. Denoting $f \in \mathbb{R}^k$ as the vector of contact forces along the normal direction, $G \in \mathbb{R}^k$ as the generalized gravitational force vector, where $k$ is the number of wheels, the vehicle motion can be described by:

$$M\ddot{q} = W_f f + W_g G,$$

where

$$M = \begin{pmatrix} m & 0 & 0 \\ 0 & I_\alpha & 0 \\ 0 & 0 & I_\beta \end{pmatrix},$$

with $(m, I_\alpha, I_\beta)$ being the vehicle mass, the moments of inertia with respect to roll and pitch axes, respectively; $W_f \in \mathbb{R}^{3 \times k}$ and $W_g \in \mathbb{R}^{3 \times k}$ are Wrench
matrices, which map the normal contact forces and gravitational forces acting on the wheels to wrenches in the vehicle-body frame.

Equation (5) is used to compute the vehicle’s trajectory on its way down by integrating \( \ddot{q} \). When there is no contact between the wheels and ground, \( \ddot{q} \) can be calculated since the contact forces \( f \) are zero, and the vehicle is driven only by the gravitational force \( G \). As soon as one wheel touches the ground, the element in \( f \) of Eq. (5), corresponding to the contacted wheel, becomes nonzero. We then need to determine \( f \) to compute \( \ddot{q} \).

To solve for \( f \) at each iteration along the vehicle’s vertical trajectory, we first derive an incremental expression for the vehicle’s velocity \( v \), which is then mapped to the rate of change of the minimum vertical distance between each wheel and ground.

Let’s write equation (5) in a discrete form:

\[
M \left( \frac{v^i - v^{i-1}}{\Delta t} \right) = W_f f^i + W_g G, \tag{6}
\]

where \( v = (\dot{z}, \dot{\alpha}, \dot{\beta})^T \) is the velocity vector, \( i \) is the iteration index, and \( \Delta t \) is the integration time step. Solving for velocity \( v^i \) yields:

\[
v^i = v^{i-1} + \Delta t M^{-1}(W_f f^i + W_g G). \tag{7}
\]

We denote \( d^i \) as the vector of minimum vertical distances between the wheels and the terrain at iteration \( i \), \( \dot{d}^i \) as the rate of change of \( d^i \). As \( W_f \) relates between forces at the center of mass and the contact forces at the contacted wheels, we can express \( \dot{d}^i \) as a function of \( v^i \):

\[
\dot{d}^i = W_f^T v^i. \tag{8}
\]

Discretizing \( \dot{d}^i \) yields:

\[
\dot{d}^i = \frac{d^i - d^{i-1}}{\Delta t}. \tag{9}
\]

Solving for \( d^i \) and using (8) we get:

\[
d^i = d^{i-1} + \Delta t W_f^T v^i. \tag{10}
\]

Substituting (7) into (10), the vertical distance \( d^i \) is expressed as a func-
tion of $f^i$:
\[ d^i = Af^i + b, \]  
(11)

where
\[ A = \Delta t^2 W_f^T M^{-1} W_f \]
and
\[ b = d^i - 1 + \Delta t W_f^T (v^{i-1} + \Delta t M^{-1} W_g G). \]

Equation (11) accounts for the vertical distance $d^i \in \mathbb{R}^k$ and contact forces $f^i \in \mathbb{R}^k$ of all $k$ wheels. We wish to solve for $f^i$ for a given $d^i$, then substitute $f^i$ into (5) to integrate the vehicle’s vertical trajectory.

Note that if $d_j \neq 0 (j \in [1,k])$, then $f_j = 0$. As these wheels do not affect the vehicle’s motion, we reduce (5) and (11) by removing the respective distances and contact forces, thus reducing the dimensions of $f$ and $d$ to $\mathbb{R}^p$. To not introduce a new notation, we continue using the same notation hereafter so that $W_f \in \mathbb{R}^{3 \times p}$, $d^i \in \mathbb{R}^p$ and $f^i \in \mathbb{R}^p$, where $p$ is the reduced number of wheels that are in contact with the terrain.

By accounting only for the wheels that are in contact with the ground ($d_j = 0, j \in [1,p]$), and assuming a rigid contact between the wheels and ground, (11) reduces to:
\[ 0 = Af^i + b. \]  
(12)

Solving for $f^i$ yields:
\[ f^i = A^{-1}b. \]  
(13)

Due to numerical errors and measurement noise, some $d_j$ are not equal exactly to zero, causing $A$ to be near singular and hence not invertible. In such cases we set:
\[ d_j = 0, \text{ if } d_j < d_\epsilon, \]  
(14)

where $d_\epsilon$ is an arbitrarily small distance threshold.

Furthermore, since direct inversion of $A$ may be very sensitive to numerical errors [24], we compute its pseudo inverse using a Singular Value Decomposition (SVD) [25], which provides a good approximation of $f$, as discussed next.

4. Computation of The Contact Forces

SVD is a powerful technique for solving sets of ill conditioned equations or matrices [20]. Computing SVD of matrices is a simple and robust procedure
even for matrices that are singular.

For completeness, we briefly describe the computation using the SVD method which is based on the following theorem of linear algebra:

**Theorem 1.** Any matrix \( A \in \mathbb{R}^{M \times N} \) can be represented by the following singular value decomposition [20]:

\[
A = S \Sigma V^T,
\]

where \( \Sigma \) is an \( N \times N \) diagonal matrix of singular values that are positive or zero:

\[
\Sigma = \begin{pmatrix}
\sigma_1 & 0 \\
0 & \sigma_2 \\
& \ddots \\
0 & & & \sigma_N
\end{pmatrix},
\]

and \( S \in \mathbb{R}^{M \times N} \) and \( V \in \mathbb{R}^{N \times N} \) are orthogonal matrices.

If \( A \) is square and non-singular, \( A^{-1} \) can be easily calculated as [20]:

\[
A^{-1} = V \Sigma^{-1} S^T
\]

where

\[
\Sigma^{-1} = \begin{pmatrix}
\frac{1}{\sigma_1} & 0 \\
0 & \frac{1}{\sigma_2} \\
& \ddots \\
0 & & & \frac{1}{\sigma_p}
\end{pmatrix}.
\]

If any element \( \sigma_j \) is zero, which makes \( \Sigma \) non-invertible, we set

\[
\frac{1}{\sigma_j} = 0, \quad \text{if } \sigma_j = 0.
\]

Since \( \sigma_j \) may not be exactly equal to zero, the \( \sigma_j \) whose ratios to the largest value \( \sigma_{\text{max}} \) are smaller than \( \epsilon \) are set to zero:

\[
\frac{1}{\sigma_j} = 0, \quad \text{if } \left| \frac{\sigma_j}{\sigma_{\text{max}}} \right| < \epsilon.
\]
We denote by $\Sigma^#$ the matrix resulting after applying (17) to $\Sigma^{-1}$, and $A^#$ as the pseudo inverse of $A$ computed by the SVD method. Eq. (16) then becomes:

$$A^# = V\Sigma^#S^T.$$  \hspace{1cm} (19)

We can compute $f^i$ (13) using $A^#$:

$$f^i = -A^#b.$$  \hspace{1cm} (20)

Having solved for $f^i$, we can now compute $\ddot{q}$ from (5) to integrate the vehicle’s vertical trajectory until reaching equilibrium at some $q_e = (z, \alpha, \beta)^T$.

The procedure to compute the vehicle pose is described in the following Algorithm:

Algorithm 1 Vehicle Pose Estimation

1: Function Pose\_Estimation($h(x, y, \gamma), p(v, w)$)  
2: $i = 1$ \hspace{1cm} $\triangleright$ Initialization  
3: $\ddot{q}^0 = (-g, 0, 0)$ \hspace{1cm} $\triangleright$ $g$ is gravitational acceleration  
4: while $\ddot{q}_{i-1} \neq 0$ do \hspace{1cm} $\triangleright$ $\ddot{q} = 0$ if vehicle’s in equilibrium  
5: \hspace{1cm} if $d_j \neq 0, \forall d_j \in d^{i-1}$ then \hspace{1cm} $\triangleright$ $j \in [1, k]$  
6: \hspace{1cm} $f^i = 0$  
7: \hspace{1cm} else  
8: \hspace{1cm} $A^# = S\Sigma^#V^T$ \hspace{1cm} $\triangleright$ Eq. (19)  
9: \hspace{1cm} $f^i = -A^#b$ \hspace{1cm} $\triangleright$ Eq. (20)  
10: \hspace{1cm} end if  
11: \hspace{1cm} $\ddot{q}^i = M^{-1}(W_f f^i + W_g G)$ \hspace{1cm} $\triangleright$ Eq. (5)  
12: Compute $q^i(z, \alpha, \beta)$ using $\ddot{q}^i$  
13: Update $d^i$  
14: $i = i + 1$  
15: end while  
16: Return $q^{i-1}(z, \alpha, \beta)$
5. Examples

The SVD based algorithm was implemented in MATLAB on an Intel i7-4790 CPU, 3.6GHz desktop computer. The following examples are demonstrated on the six-wheel vehicle shown in Fig. 3, with mass of 500 kg. The length, width and height of the vehicle are 1.5 m, 0.9 m and 0.5 m respectively. The distance threshold used to identify the contact points with the terrain is $d_c = 1 cm$. In these examples, the wheels that are in contact with the terrain when the vehicle reaches equilibrium, are marked red, and those that are not are marked blue. The contact forces are shown as green vectors, emanating from the contact points on the ground. The computational efficiency of the SVD based method is later compared to the LCP based method.

**Example 1**: in this example, the six-wheel vehicle rests on a flat terrain as shown in Fig. 4. At equilibrium, the distance between the wheels and ground are set arbitrarily to:

$$d_1 = \begin{bmatrix} 0.07 & 0.021 & 0.052 & 0.021 & 0.02 & 0.016 \end{bmatrix} \text{cm}.$$

Since all wheels are within the contact distance threshold, their contact forces are nonzero:

$$f_{s1} = \begin{bmatrix} 816.7 & 816.7 & 816.7 & 816.7 & 816.7 & 816.7 \end{bmatrix} \text{N}.$$

As expected, the sum of all contact forces is 4900 N, which equals the gravitational force acting on the vehicle.

**Example 2**: here, the six-wheel vehicle rests on an uneven terrain, as shown in Fig. 5. The distance between the wheels and ground at equilibrium are set to:

$$d_2 = \begin{bmatrix} 0.99 & 2.0 & 0.5 & 3 & 0.002 & 2.5 \end{bmatrix} \text{cm}.$$

The wheels that are within the distance threshold are shown in red in Fig. 5. Their corresponding contact forces are:

$$f_{s2} = \begin{bmatrix} 1225 & 0 & 1225 & 0 & 2450 & 0 \end{bmatrix} \text{N}.$$

Their sum reaches 4900 N, as expected.

It is interesting to note that although the 1st and 3rd wheels are within the contact threshold, their contact distance is relatively large compared to the 5th wheel, which explains the larger contact force acting on the 5th wheel.

**Example 3**: in this example, the six-wheel vehicle moves over a large
Figure 4: Example 1: The six-wheel vehicle is resting on a flat terrain. The contact forces are shown as vectors emanating from the contact points.

Figure 5: Example 2: The six-wheel vehicle is resting on an uneven terrain, with 3 wheels in contact with the ground.

bump, as shown in Fig. 6. In this case, only four wheels are in contact with the ground. The computation time from the first contact to equilibrium was $7.6\text{ms}$. An animation of the computation process is shown in Appendix A.

Example 4: this example demonstrates pose estimation along a path segment for a four-wheel vehicle moving over a small rock, as shown in Fig. 7. The contact points switch places as the vehicle is moving over the rock as is shown in Fig. 7, and a video is shown in Appendix B.

Example 5: this example demonstrates the pose estimation for a six-wheel vehicle moving over a large bump and a deep hole, as demonstrated in
Fig. 8.

Example 6: this example shows an eight-wheel vehicle moving along a path (shown in green) over a mountainous terrain, as shown in Fig. 9. The average computation time of one pose from the first contact to equilibrium was 8.1 ms.

5.1. Comparisons to the LCP based method

To assess the computational efficiency and accuracy of the SVD based approach, we implemented the LCP based method.

The contact forces in example 1, calculated by the LCP based method for the same set distance $d_1$ were

$$f_{L1} = \begin{bmatrix} 816.4 & 816.4 & 816.4 & 816.4 & 816.4 & 816.4 \\ 816.4 & 816.4 & 816.4 & 816.4 & 816.4 & 816.4 \end{bmatrix} N,$$

which is close to the results obtained by the SVD based method. However the computation time of the LCP based method was 0.9 ms, compared to 0.07 ms using the SVD method, which is ten times slower.

The contact forces in example 2, calculated by the LCP based method for the same distance $d_2$ are

$$f_{L2} = \begin{bmatrix} 1225 & 0 & 1225 & 0 & 2450 & 0 \end{bmatrix} N,$$

which is the same as obtained by the SVD method, but took computation time of 1.0 ms, compared to 0.07 ms, which is 15 times slower.

It can be seen from these examples that both SVD and LCP based methods provide a good approximation of the contact forces, but the SVD based
method is on average 10 times faster than the LCP based method.

Fig. 10 compares the computation times of both methods as a function of the number of wheels. These results were obtained by averaging the computation times of 100 runs for each case for each method. As shown, the computation time of the LCP based method grows from 0.9ms to 1.7ms as the number of wheels increases from 4 to 24, whereas the computation time of the SVD approach increased from 0.08ms to 0.18ms. It seems that the computation times of both methods are equally affected by the number of wheels, however, the SVD based approach is consistently more computationally efficient than the LCP based approach. For large numbers of wheels,
Figure 8: Example 5: Pose estimation of a six-wheel vehicle moving over a large bump and a deep hole, shown in two views.

Figure 9: Example 6: An eight-wheel vehicle moving along a specified path over uneven terrain.

the SVD based method is considerably more efficient than the LCP based approach.
6. Conclusions

A method for pose estimation of off-road vehicles moving over uneven terrain is presented. The terrain is represented by a cubic B-patch that is interpolated over a 3D point cloud. The vehicle pose is calculated by dropping the vehicle vertically at a given configuration under the gravitational force until the contact forces and the gravitational force reach equilibrium. The contact forces between the wheels and the terrain are computed using the Singular Value Decomposition (SVD).

The presented method is robust, easy to implement and computation-ally efficient, allowing real time computation during motion. The robustness and efficiency of the proposed method is shown through several examples and comparisons. The obtained vehicle poses can be used for motion planning, control design on a rough terrain, stability analyses and traversability analyses over uneven terrain.

References

[1] S. M. LaValle, Planning Algorithms, Cambridge University Press, Cambridge, U.K., 2006, available at http://planning.cs.uiuc.edu/.
[2] Y. Ma, G. Zheng, W. Perruquetti, Z. Qiu, Local path planning for mobile robots based on intermediate objectives, Robotica 33 (4) (2015) 1017–1031. doi:10.1017/S0263574714000186.

[3] Z. Shiller, Y. R. Gwo, Dynamic motion planning of autonomous vehicles, IEEE Transactions on Robotics and Automation 7 (2) (1991) 241–249. doi:10.1109/70.75906.

[4] M. P. Mann, Z. Shiller, Dynamic stability of off-road vehicles: a geometric approach, in: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., IEEE, 2006, pp. 3705–3710.

[5] K. Berns, K.-D. Kuhnert, C. Armbrust, Off-road robotics—an overview, KI - Künstliche Intelligenz 25 (2) (2011) 109–116. doi:10.1007/s13218-011-0100-4.

[6] M. G. Bekker, Theory of land locomotion, University of Michigan Press, 1956.

[7] M. G. Bekker, Introduction to terrain-vehicle systems, University of Michigan Press, 1969.

[8] C. Debain, P. Delmas, R. Lenain, R. Chapuis, Integrity of an autonomous agricultural vehicle according the definition of trajectory traversability, in: AgEng 2010, International Conference on Agricultural Engineering, Clermont-Ferrand, France, 2010.

[9] T. M. Howard, A. Kelly, Optimal rough terrain trajectory generation for wheeled mobile robots, The International Journal of Robotics Research 26 (2) (2007) 141–166.

[10] A. K. Singh, K. M. Krishna, V. Eathakota, Planning stable trajectory on uneven terrain based on feasible acceleration count, in: 2011 50th IEEE Conference on Decision and Control and European Control Conference, IEEE, 2011, pp. 6373–6379.

[11] A. K. Singh, K. M. Krishna, S. Saripalli, Planning non-holonomic stable trajectories on uneven terrain through non-linear time scaling, Autonomous Robots (2015) 1–22.
[12] J. Jordan, A. Zell, Real-time pose estimation on elevation maps for wheeled vehicles, in: Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on, IEEE, 2017, pp. 1337–1342.

[13] J.-Y. Jun, J.-P. Saut, F. Benamar, Pose estimation-based path planning for a tracked mobile robot traversing uneven terrains, Robotics and Autonomous Systems 75 (2016) 325–339.

[14] R. Cottle, J. Pang, R. Stone, The Linear Complementarity Problem, Society for Industrial and Applied Mathematics, 2009. doi:10.1137/1.9780898719000.
URL https://epubs.siam.org/doi/abs/10.1137/1.9780898719000

[15] D. Baraff, Fast contact force computation for nonpenetrating rigid bodies, in: Proceedings of the 21st annual conference on Computer graphics and interactive techniques, ACM, 1994, pp. 23–34.

[16] J. E. Lloyd, Fast implementation of lemke’s algorithm for rigid body contact simulation, in: Proceedings of the 2005 ieee international conference on robotics and automation, IEEE, 2005, pp. 4538–4543.

[17] V. Acary, B. Brogliato, Numerical methods for nonsmooth dynamical systems: applications in mechanics and electronics, Springer Science & Business Media, 2008.

[18] S. J. Byrne, Solution of quadratic programming problems, MASSEY UNIVERSITY, PALMERSTON NORTH, NEW ZEALAND 12 (2) (1984) 73–90.

[19] E. Drumwright, A fast and stable penalty method for rigid body simulation, IEEE Transactions on Visualization and Computer Graphics 14 (1) (2008) 231–240.

[20] J. Demmel, Applied Numerical Linear Algebra, Society for Industrial and Applied Mathematics, 1997. doi:10.1137/1.9781611971446.
URL http://epubs.siam.org/doi/abs/10.1137/1.9781611971446

[21] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, Numerical Recipes 3rd Edition: The Art of Scientific Computing, 3rd Edition, Cambridge University Press, New York, NY, USA, 2007.
[22] F. Malartre, T. Feraud, C. Debain, R. Chapuis. Digital elevation map estimation by vision-lidar fusion, in: 2009 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2009, pp. 523–528. doi:10.1109/ROBIO.2009.5420701.

[23] M. E. Mortenson, Geometric modeling, John Wiley & Sons Inc., New York: Wiley Computer Publishing.

[24] G. Gregorcic, The singular value decomposition and the pseudoinverse, Matrix 2 (1) (2001) 2.

[25] A. Ben-Israel, T. Greville, Generalized Inverses: Theory and Applications, CMS Books in Mathematics, Springer, 2003. URL https://link.springer.com/book/10.1007/b97366

Appendix A. Video for example 3

[Video A]: Example for a six-wheel vehicle moving over a large bump.

Appendix B. Video for example 4

[Video B]: Example for a four-wheel vehicle moving over a small rock.