Performance Optimization of All-Terrain Robots: A 2D Quasi-Static Tool

Performance Optimization of All-Terrain Robots: A 2D Quasi-Static Tool

Ambroise Krebs[1], Thomas Thueer[1], Stéphane Michaud[2] and Roland Siegwart[1]

[1]*Autonomous Systems Lab*
Swiss Federal Institute of Technology Zurich (ETHZ)
8092 Zürich, Switzerland
{ambroise.krebs, thomas.thueer, roland.siegwart}@mavt.ethz.ch

[2]*Contraves Space AG*
Schaffhauserstr. 580
8052 Zürich, Switzerland
stephane.michaud@unaxis.com

***Abstract* - The creation of a rover for a specific task requires designing and selecting the mechanical structure specifically for its mission. This can be done by modelling a chassis and evaluating it with specific criteria, which is the aim of the Performance Optimization Tool presented here. This Software makes it possible to compare and improve existing and new designs in a quick and efficient way. The tool presented in this paper is based on a quasi-static approach including optimization of the friction coefficients to model and evaluate the rover.**

Index Terms – Optimization Tool, Performance evaluation, Rover Design, All-terrain robot, Space robotics.

I. INTRODUCTION

Nowadays, we see an increasing demand for autonomous vehicles for rough terrain. These rovers are used in a large number of contexts such as exploration in space (e.g. mars exploration) or on earth (e.g. dangerous areas like volcanoes), search and rescue missions (e.g. in the case of an earthquake), etc. In the past years, industry and researchers have developed, experimented and used a large number of different concepts to fulfil the needs specific to the missions [1].

Although the mission types and constraints might be very different, the unstructured terrain is a common feature to these applications. Therefore the terrainability [2] is a key element that has a great impact on the rover control and navigation, as the robot can only be asked to go where it is able to go.

The suspensions of the autonomous vehicles can be divided into two categories: active and passive. The first group can be subdivided into wheeled and legged robots, which differ significantly in their way to interact with the ground, but both their mechanical suspension/legs have to be actively controlled. "Octopus" by ASL [3] is an example of a fully active wheeled rover structure, "ASIMO" by Honda [4] is one for an actively controlled legged rover. The passive rovers adapt to the terrain through the applied forces and the internal mechanical suspension design. As the only motors of the mechanical structures are dedicated to the wheels or the steering, it implies that the energy spent is integrally used for the movement of the rover. For this reason and as the complexity of the controller can be kept as low as possible, the focus of this work is set on wheeled robots with passive suspension mechanisms.

The design of each rover goes through different steps. The general mechanical structure has first to be found. It is then required to go through an optimization process to define the length of the elements of the structure, the position of the centre of gravity, the wheels' size and so on. The process ends with the plans depicting the mechanical constraints. Tools such as *Sysquake®*[1]*, Adams®* or *ODE* can be used in the optimization process. Unfortunately, it requires creating each time a model specific to the mechanical structure to tune. This takes a lot of time, especially as these steps might have to be iterated several times. Therefore, to make these operations faster and simpler, it seems interesting to have a generic "rover structure development kit".

The goal of the tool presented here is to test and compare the performances of a large number of different suspension mechanisms (in concept or dimension), allowing the user to finally select the best design and adapt it for his application through a parametric study. The Performance Optimization Tool (POT) can be used to quickly compare existing designs on the same basis or to design a new structure in a more efficient way.

First, this paper presents an overview of the simulator and its motivation. The quasi-static model used to simulate a rover and the ideas behind it are explained in section III. In a next step (section IV), a typical example is used to explain how the simulation is performed. Finally, the outputs of the simulator are presented and discussed in section V.

II. THE PERFORMANCE OPTIMIZATION TOOL

A. Introduction

The tool presented in this paper is part of the Rover Chassis Evaluation Tools (RCET) [5], an ESA funded project. Its goal is to create a full platform for the mechanical rover structure development. It includes the presented 2D tool (POT), a 3D simulator based on *COSMOS/Motion®,* a single wheel testbed and a system level testbed (both dedicated to hardware testing).

The 2D tool is to be used in an early phase of the design process whereas the 3D simulator allows the user to analyse the final version using a more complex model. It implies that simplicity and low calculation time are key issues for the POT as the number of tests performed will be important.

[1] Numerical computation environment similar to *MATLAB* with an extended graphical interactive interface.

B. *Focus of the POT*

One can wonder what the interest of such a 2D tool is when 3D simulators are available. The main reason is the calculation time and the easiness to use. Simulating as well as creating a 3D CAD model costs hours whereas the POT tests are modelled and run in only a few minutes.

The main approximation made in the approach of the POT is the use of a quasi-static model. Exploration rovers typically move slowly, around five centimetres per second or less, improving the odometry precision and allowing moving safely. At such speeds, the dynamical effects are negligible and a quasi-static model is very close to reality. Such an approach has already been used for rover control [6].

The quasi-static model offers the possibility to compute all the forces and torques applied on each body of the rover. The torques and forces of interest are those on the wheels. It is the wheel-ground interaction that is of importance to define the rover capabilities in a specific state with respect to the terrain shape.

To summarize, the POT does not replace a 3D simulator but aims at narrowing the search space of the structure design in a first step. This allows the designers to have a fast and effective approach. 3D tools can then be used to make the final design.

C. *Software Architecture*

The POT is composed of six modules, developed under *Visual C++:*

- The *Rover Design I/F* allows the user to quickly define a mechanical structure using user friendly tools (presented in chapter III). The rover model is then generated automatically and stored in the *Database* to be used in a simulation.
- The *Batch Creator* allows the user to create a list of simulation runs. For each run a rover, a terrain and output parameters have to be selected.
- The *Batch Launcher* allows the user to select the batch containing the tests to be made. They are performed afterwards and the results are stored in the *Database*. It is the main interface of the tool.
- The *Simulation Engine* computes the data (rover state and the optimization) for each simulation step.
- The *2D Player* allows visualizing the rover states in the simulation run.

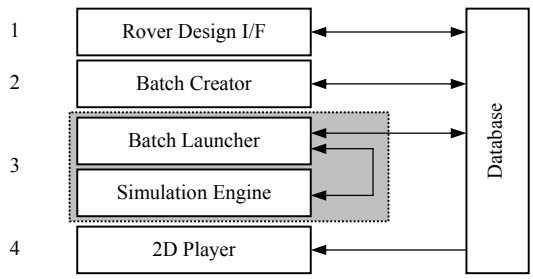


Fig. 1: POT Framework

III. Rover Model

The *Rover Design I/F* provides a user-friendly GUI (Graphical User Interface) in order to create the wanted rover using rigid predefined elements which have "attach points". These are defined points to link an element to another. Then the rover model is generated.

A. *Predefined Elements*

The available predefined bodies are:

- *Wheel*: circular element with a single "attach point" in its centre.
- *Bar*: a simple, straight element with three "attach points". One at each end and one in between.
- *Parallel Bogie*, *Rocker Bogie* and *Wheel with 2 Bars*: elements which are extensions of the two first elements.

Note that the last category is proposed to the user in order to simplify the generation of the equations. The internal equations of these elements are already simplified as much as possible. This procedure gives the opportunity to model them the way they behave in reality. For example, the predefined element *Parallel Bogie* considers the parallel closed loops and prevents the model from being over-constrained.

B. *Attach Points*

The rover creation goes through creating and linking the required predefined elements. Those have specific "attach points" that can be linked to each other. Two linked "attach points" form a joint which can be of three different types:

- *Fixed* joint: links two bodies with zero degrees of freedom (DOF).
- *Revolute* joint: is a pivot joint with one DOF.
- *Motorized* joint: is a pivot joint with a variable torque applied. It is used for the wheel "attach point". It corresponds to one DOF plus one external constraint (torque).

C. *Static Equations*

Here is a simple example of the 2D equations used to generate the rover model. Static equations for a rigid body (see Fig.2):

$$Fab_x + Fcb_x = 0 \tag{1}$$

$$Fab_y + Fcb_y - m \cdot g = 0 \tag{2}$$

$$\vec{M}ab + \vec{M}cb + \vec{F}ab \times \vec{D}_1 + m\vec{g} \times \vec{D}_2 = 0 \tag{3}$$

Joint equations:

$$Fab_x = -Fba_x \tag{4}$$

$$Fcb_x = -Fbc_x \tag{5}$$

$$Fab_y = -Fba_y \tag{6}$$

$$Fcb_y = -Fbc_y \tag{7}$$

$$\vec{M}ab = -\vec{M}ba \tag{8}$$

$$\vec{M}cb = -\vec{M}bc \tag{9}$$

By using these equations, it is possible to model and generate any 2D passive mechanical structure.

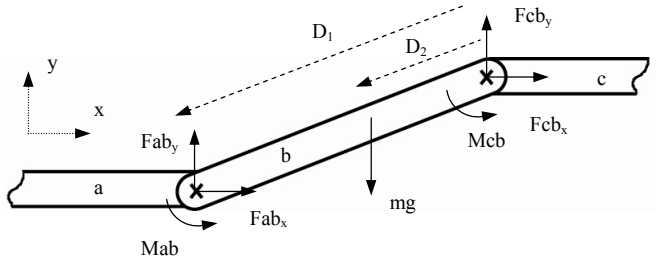


Fig. 2: Static equation on a body

Using these equations for each of the n_B rover bodies and n_G "attach points" yields the following system:

$3 \cdot n_B$	Static equations
$3 \cdot n_G$	Joint equations
$n_{revolute} + n_{motorized}$	Joint constraint equations
$3 \cdot 2 \cdot n_G + n_{motorized}$	Variables

D. Model Validation

Once the rover model is established, it has to be validated in order to be used in a simulation run. This is done using Grübler's formula [7]:

$$f = 3 \cdot (n_B - n_G - 1) + \sum f_{Gi} \qquad (10)$$

with f = DOF, n_B = number of bodies, n_G = number of joints and f_{Gi} = DOF of joint. The mechanical structure of a 2D rover must have DOF equal to one.

That means a two wheeled car (in 2D) is a valid rover because it has $f = 1$ as (Fig. 4 in white) $n_B = 4$ (1-4) and $n_G = 4$ (I-IV, all revolute joint). In the contrary, a three wheeled car (in 2D) is considered as hyper static and therefore not valid. $f = 0$ as (Fig. 4 in white and gray) $n_B = 5$ and $n_G = 6$ (I-VI all revolute joint). This doesn't mean that such a rover is not usable, but it means that not all wheels can touch the ground at the same time on uneven terrain. Such a state is not allowed in the POT as it is not optimal.

IV. Simulation Run

Once a rover model is designed in accordance to the user input, it is possible (if the rover model is valid) to generate a simulation batch with this rover. The other main parameter of a simulation is the test terrain.

A. Terrain Model

The shape of the test terrain is defined by a set of 2D points. Fig. 3 shows an example of a typical terrain: the *step*. Terrains such as *step* or *slope* are typical test terrains to characterize rovers.

	x [m]	y [m]
1.	0	0
2.	1	0
3.	1	0.1
4.	2	0.1

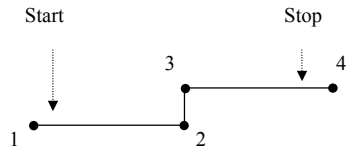


Fig. 3: *Step* terrain definition

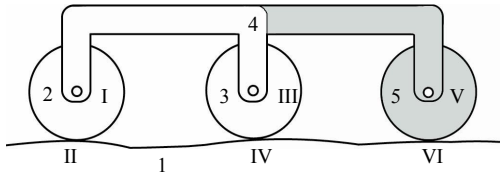


Fig. 4: Two and three wheeled car validation

Once a terrain is chosen or defined, the simulation can be started.

B. Rover Movement

At the beginning of a simulation, the robot is positioned at the left hand side of the obstacle on a flat part. The simulation makes it move toward the other extremity of the obstacle and stops when the rover reaches the flat part on the right hand side of the terrain.

At each step in between, the simulation goes through the following procedure:

- The rover is positioned such that all its wheels touch the ground.
- The state of the rover, which is the position and orientation of all its bodies, is determined. The model uses this information to generate the forces and torques needed to keep the system in equilibrium.
- The output forces, torques and metrics are computed and stored.
- The rover is moved to the next position.

Note that, the rover movement is not induced by the computed forces since a static model is used. The movement of the rover might rather be seen as a teleportation [8] from one state to the next.

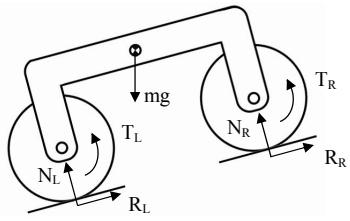
C. Under-Constrained Problem

As the rover has one DOF, controlling such a structure would require only one single motor. In reality, in order to increase the terrainability, every wheel is a drive wheel and that means there are as many motors as wheels. As a result, there is an infinity of solutions to keep the rover in equilibrium. For example a rover with n_B bodies, including n_w motorized wheels corresponds to the following system (reduced form):

$3 \cdot n_B$	Equations
$3 \cdot n_B + n_w - 1$	Variables

The system is under-constrained by $n_w - 1$ and a solution has to be chosen among the infinite possibilities. The chosen solution has to reflect the goal of the simulator to express the capacities of the mechanical structure of the rover. So one must select a solution, but which one?

In the case of a car on a 15° slope (Fig. 5), the system is composed of nine equations with ten variables. This means a criterion must be added in order to determine a solution. Three criteria have been tested. The resulting torques T and the corresponding friction coefficient μ are expressed in the table of the same figure.



	$T_L = T_R$	$T_R = 0$	$\mu_L = \mu_R$
T_L [Nm]	-0.32	-0.64	-0.41
T_R [Nm]	-0.32	0	-0.24
μ_L [-]	0.21	0.43	0.27
μ_R [-]	0.37	0	0.27

Fig. 5: *Car* model and the μ generated for three criteria. The wheels are massless and the rover mass equals 5Kg.

The friction coefficient on the ith wheel is defined as:

$$\mu_i = \frac{R_i}{N_i} \tag{8}$$

where R_i is the tangential force on the ith wheel and N_i is the normal force on the ith wheel. This parameter corresponds to the friction coefficients needed to allow the car to keep its equilibrium. In fact, it expresses the risk of slippage which is proportional to this needed μ. One wants it to be as low as possible to know the risk of slippage of a given structure in a given state. This expresses the best that the tested mechanical can give in a given state.

Fig. 5 shows that the best criterion is the third one, where the μ are equal. In fact, it can be shown [9] that it is not possible to find a better solution.

D. Optimization Criteria

A numerical optimization is needed to find a solution satisfying the criteria. The corresponding heuristic is to select the set of torques that minimizes the friction coefficient μ. μ_i corresponds to the minimum friction coefficient required in order to avoid the rover to slip in a given position.

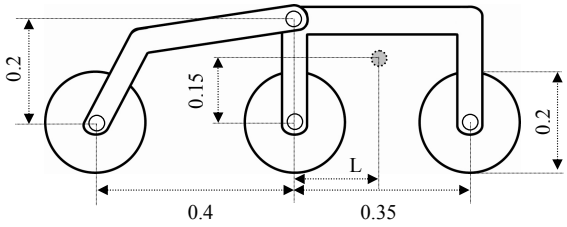
$$\min(\mu_i) = \min(\frac{R_i}{N_i}) \tag{9}$$

Minimizing the friction coefficient needed for the rover to move expresses the best performance the structure is able to achieve. It is just dependent on the mechanical design.

In fact, it can be shown that the best solution is the solution where all μ_i are equal [9]. This solution is found using a numerical optimization algorithm with the following heuristic:

$$\min(\sum_i (\mu_i - \overline{\mu})^2) \tag{10}$$

with $\overline{\mu}$ equivalent to μ mean. Equation (10) shows that the optimization searches for a minimum variance. The optimization algorithm is implemented as a *MATLAB* script as



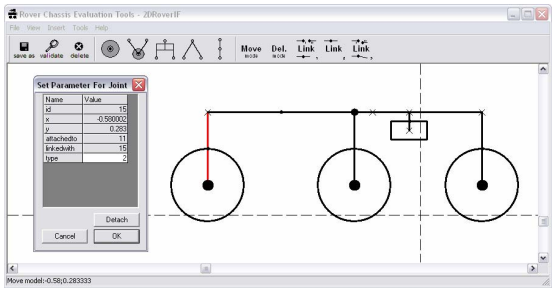


Fig. 6: *Trailer* schematic with dimensions and its model in the POT. Here L = 0.15m. The wheels weigh 0.4 kg each. The rest of the rover weighs 2kg and the trailer weight is neglected. ◌ = Rover structure's Centre Of Mass (COM) position (without taking the wheels into account).

the under-constrained problem is linear. The algorithm is derived from a fix point algorithm.

V. Results

In this section the simulator output is presented. The rover used is a *trailer* illustrated in Fig. 6 on the terrain of Fig. 3.

A. Simulation Run Preparation

The corresponding model has first to be created in the *Rover Design I/F* and then, using the *Batch Creator*, a new batch is created containing a single simulation run. The *Batch Launcher* is used to select the batch just created and performs the simulation.

This example was performed on a laptop Dell D610, M2.0GHz with 1Go RAM. It took only 25 seconds.

B. Tool Outputs

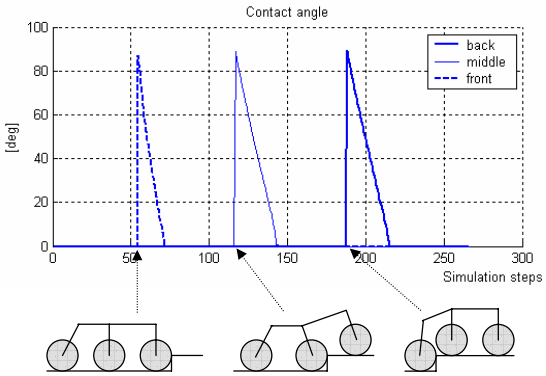


Fig. 7: Contact angle of the wheels during the simulation and the corresponding situations schematics

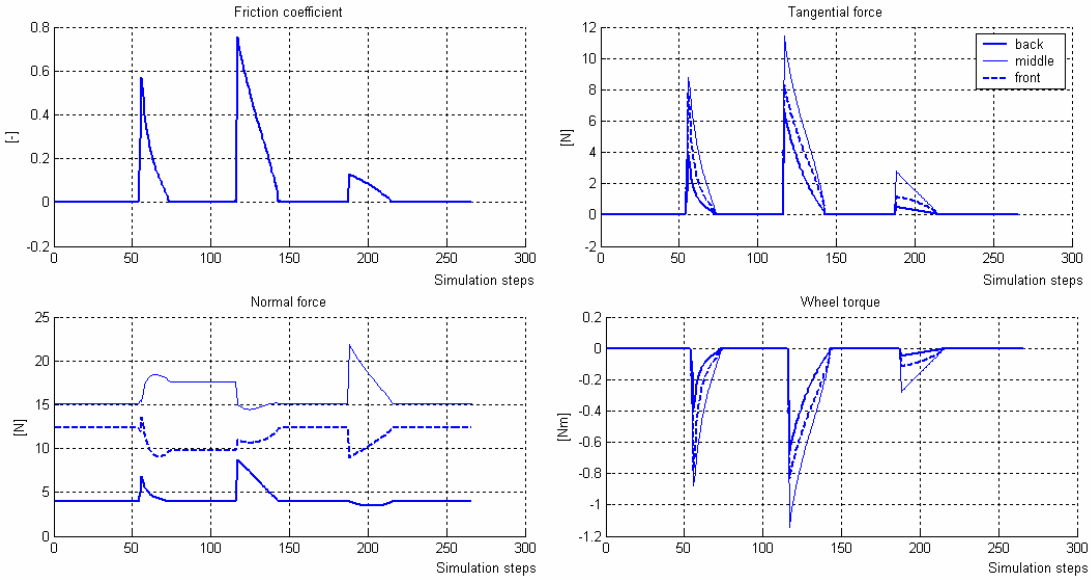


Fig. 8: Output of the simulator: evolution of the friction coefficients, tangential and normal forces and the corresponding torques during the simulation

Fig. 7 shows the position of the contact angles on the wheels. These curves give a pretty good idea of the state of the rover as each peak represents a wheel in contact with the vertical part of the step.

Fig. 8 represents the torques, normal and tangential forces on each of the three wheels. The highest peak corresponds to the middle wheel climbing the step. It is higher because the mass repartition on the wheels is not equal and the middle wheel carries the biggest weight. Fig. 8 displays also the required minimum friction coefficient for each wheel that the rover needs to stay in an equilibrium state. Three peaks are clearly visible. Those appear each time a wheel is in contact with the vertical part of the step. Note the smaller height of the third peak which is a consequence of the weightless trailer. Therefore, the rear wheel on the vertical part of the step has only the weight of the wheel itself to "hold".

Although they are not shown here, the wheels and COM position data are also available. These allow the user to get the accurate state of the rover at a specific time in the simulation.

Once the test is performed, the user can visualize the rover trajectory and the behaviour of the mechanical structure at any simulation step using the *2D Player* (Fig. 11).

C. *Parametric Study*

It is possible to use the POT to perform a parametric study on a given parameter of a rover. For example, the L distance of the *trailer* of Fig. 6.

L [m]	Max($\mu_{peakfront}$)	Max($\mu_{peakmiddle}$)	Max($\mu_{peakrear}$)
0.05	0.36	0.86	0.13
0.1	0.47	0.8	0.13
0.15	0.57	0.76	0.13
0.175	0.73	0.72	0.13
0.2	0.86	0.67	0.13
0.25	0.91	0.57	0.13

Fig. 9: Parametric study results. These sets correspond to the three peak values of the friction coefficient graph (top left plot of Fig. 8).

The model of the rover has to be redefined for each different parameter but this can be easily done using the *2D Rover I/F*. The definition of the whole batch containing all the needed tests can be achieved within 5 minutes and the simulation took 1 minute and 40 seconds (for the six tests).

The results can be seen in Fig. 9. The position of the centre of gravity is the best around:

$$L = 0.175 \text{ m} \quad (11)$$

This value represents the position at the exact same distance from both wheels. This is what was expected as it corresponds to an equal mass distribution. In fact, it can be seen that the μ are not exactly equal. That suggests a better solution, specific to climbing a step moving forward, can be found for a slightly lower L.

D. *Design Comparison*

In the same manner, it is possible to compare different rover designs instead of the same design with various dimensions. One can, for example, test the *Shrimp* [10] versus the *trailer* in a predefined test. The user has to be particularly cautious in the terrain selection and the metrics definition to have valuable results. Two rovers with different wheel dimensions are difficult to compare. More detailed information about this subject can be found in [11].

E. *Numerical Optimization "Failure"*

Using a numerical optimization algorithm brings some uncertainty in the results as there is no assurance to find the best solution at every simulation step. Nevertheless, experience shows that the simulator outputs are extremely satisfying. When the algorithm fails to provide the best solution, it indicates a state in which the rover cannot be in equilibrium. For example, in Fig. 10, the outputs are given for the same simulation as previously but with a step height of 0.33 meter. The simulation took 45 seconds.

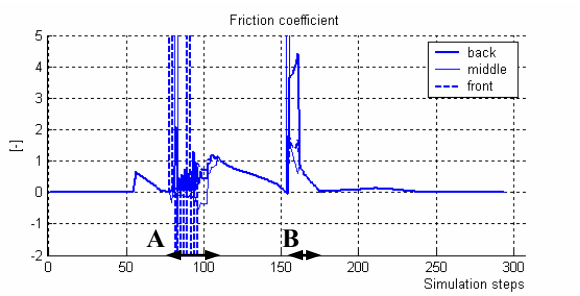


Fig. 10: Output showing the evolution of the minimum friction coefficient required. It shows clearly that the optimal solution is not found in A and B (the μ are not equal and reach values with no real meaning).

Situation **A** (step 78) Situation **B** (step 153)

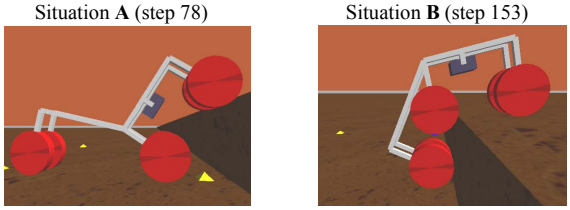


Fig. 11: Screen shot of the *2D Player* showing the state of the *trailer* in unstable situations (extended 2D). The step height is 0.33 meters.

Fig. 10 shows problems between steps 78 and 109 (A) and between steps 153 and 162 (B). The found solutions are not satisfying. They correspond to the rover state displayed in Fig. 11. It is clearly visible that the simulator is not capable of finding an optimal solution (equal μ) because the rover is in unstable states. Remember that the rover's state is found by moving it along the terrain shape without taking into consideration any physical constraint.

VI. Future Work

The simulator validation has to be performed comparing real test data to the simulator data. This will allow us to accurately characterize the difference between the POT outputs and the reality [11]. It will be done using the *Solero* [12] rover on various symmetrical obstacles.

Another point might be the addition of constraining predefined elements such as spring suspension. These elements are not available[2] in the current state of the simulator and might be essential to some rover models.

VII. Conclusion

In this work, a new tool for the evaluation of rover concepts was presented. The simulator was shown to be fast and giving valuable results, allowing a designer to spare a lot of time in the early design phase. The user will understand the interactions on any passive rovers design.

The inputs of the simulator are the rover model and the terrain. The outputs are, for each wheel and simulation step, the rover state (contact angles and position), the applied forces (normal and tangential forces and torques) and the minimum friction coefficient necessary to keep an equilibrium state. The COM position is also given at any simulation step.

As the optimization algorithm searches for an ideal solution, in steady state the results are independent from a control algorithm and reflect the real performance of the mechanical structure. Therefore, the obtained data are highly appropriate for a first step in the rover design process although the simulator is limited to a two dimensional and quasi-static approach.

[2] The spring on the front fork of the Solero is simulated using a special and dedicated model.

Acknowledgement

Thank to Rolf Jordi for the precious work he did on the *2D Rover I/F* and for his support in C++.

This work was done in the context of the ESA project RCET [Nr. 18191/04/NL/PM] founded by ESA.

References

[1] Lauria, M., "Nouveaux concepts de locomotion pour véhicules tout-terrain robotisés", Ph.D. Thesis No. 2833, Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, 2003

[2] Apostolopoulos, D.S., "Analytical Configuration of Wheeled Robotic Locomotion", Ph.D. Thesis No. Carnegie Melon University, Pittsburgh, Pennsylvania, USA, 2001

[3] Lauria, M., Piguet, Y. and Siegwart, R., "Octopus - An Autonomous Wheeled Climbing Robot", Fifth International Conference on Climbing and Walking Robots, Bury St Edmunds and London, UK, 2002.

[4] Sakagamim, Y., Watanabe, R., Aoyama, C., Matsunaga, S., Higaki, N. and Fujimura, K., "The intelligent ASIMO: System overview and integration", Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems, Lausanne, Switzerland, 2002.

[5] Michaud, S., Richter, L., Ellery, A., Manzoni, D., Patel, N. and Thueer, T., "Rover Chassis Evaluation Tools - System Requirements Document", MARSRO-CSAG-TN-0002, ESA contract no: 18191/04/NL/PM, 2004 – unpublished.

[6] Lamon, P. and Siegwart, R., "Wheel Torque Control in Rough Terrain - Modeling and Simulation", ICRA, Barcelona, Spain, 2005.

[7] Grübler,M.,"Getriebelehre: Eine Theorie des Zwanglaufes und der ebenen Mechanismen", Julius Springer Ed. , Berlin, 1917.

[8] Wise, R., Foster, A.D., Livingston, H., "Star Trek: The Movie Picture", 1979.

[9] Lamon, P., Krebs, A., Lauria, M. and Siegwart, R., "Wheel Torque Control for a Rough Terrain Rover", ICRA, New Orleans, USA 2004.

[10] Siegwart, R., Lamon, P., Estier, T., Lauria, M. and Piguet, R., "An Innovative Design for Wheeled Locomotion in Rough Terrain", Journal of Robotics and Autonomous Systems, Elsevier, 2002.

[11] Thueer, T. Krebs, A. and Siegwart, R., "Comprehensive Locomotion Performance Evaluation of All-Terrain Robots", IROS, Beijing, China 2006.

[12] Michaud, S., Schneider, A., Bertrand, R., Lamon, P., Siegwart, R., van Winnendael, M., et al., "SOLERO: Solar Powered Exploration Rover", The 7th ESA Workshop on Advanced Space Technologies for Robotics and Automation, The Netherlands, 2002.