

Upscale Rocksim with RockShell: Map, Optimize, Monte-Carlo, and more

Rocksim, and other rocket simulators like Open Rocket, are truly useful tools. They allow us to gain years of experience in just a few hours by learning how factors like wind and launcher tilt affect rocket flight. But do you ever find yourself busily re-running simulations with different rod elevations to minimize the distance between launch and landing points? Or re-running a simulation on the day of the launch with the correct environmental conditions, and then wondering if you remembered to change the wind speed on the Launch Conditions sub-tab of the Simulation Properties tab? Or wishing you could easily test-launch a hundred simulations with random

variances in wind speed, wind direction, and thermals, to see how often you would end up in the trees for a given main parachute deployment altitude and rod elevation?

RockShell is an extension to Rocksim that solves these issues and more. RockShell is free, works on Rocksim version 7 for Windows and above, consists of a small single-file executable that requires no installer, and is built on the philosophy of ease-of-use. All options are visible on a single screen, and maps that can be interactively zoomed and scrolled are generated from within the program without requiring export.

by Jim Squire

A Monte-Carlo analysis (repetitive simulations done while randomly-varying certain input parameters such as wind) performed on the standard edition of Rocksim with the free RockShell application shows a cross-road landing of the simulated rocket unlikely but possible at the NAR 2015 National Sport Launch site. What happens if the wind was a bit stronger? In half a minute you can see the mapped result of another twenty (or more!) simulations.

Background

RockShell was written after a near-disaster with my L2 certification flight in July 2014. We were flying on a limited-sized field, and to reduce the likelihood of an unfriendly tree encounter I had carefully selected the minimum-sized motor to give me a safe flight. As the RSO was inspecting the rocket, he asked if I had checked the thrust to weight ratio.

“Of course,” I replied, “and it’s just barely over 5:1.”

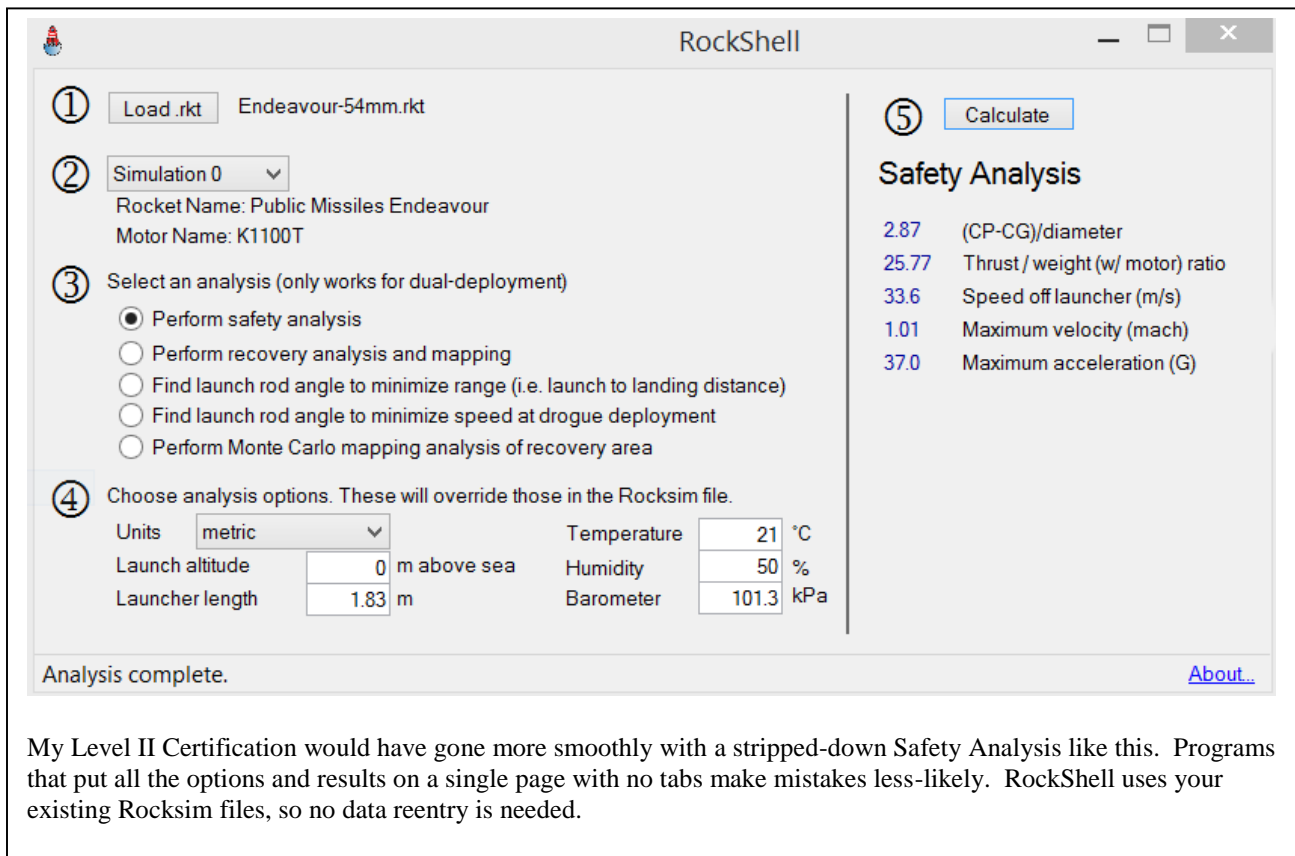
Unconvinced with the heavy rocket in his arms, he continued, “And you weighed it with the motor in, right?” Not just that but I had double-checked everything. I had weighed the rocket without the motor the previous month, entered it as an override in Rocksim, and then reweighed it again once the motor was delivered. The Rocksim-reported total mass matched what I had weighed so I felt secure that my calculations were correct. But as he

asked I realized I wasn’t sure if the number Rocksim reported was with or without the motor, and perhaps all that I “verified” was that the unloaded weight I entered in Rocksim was indeed the unloaded weight. We re-weighed, and sure enough I had misunderstood the Rocksim-reported mass as being loaded and compounded the error by mistakenly double-checking it against the unloaded weight. My thrust-to-weight ratio was just 4.3; both embarrassing and disappointing if it scrubbed the certification flight. Luckily there was almost no wind, and simulations showed the off-rail speed was 14 m/s which was just enough on the safe side for the RSO to authorize the ultimately-successful launch. But it made me realize how easy it was to misinterpret Rocksim data, and my experience using the application taught me the related lesson that it was easy when running multiple simulations to forget to update a

variable in one of Rocksim’s several property pages.

RockShell is essentially a front-end to Rocksim. It allows the user to choose an existing Rocksim file, select one of five pre-set analyses, and view all simulation options at once from a single screen so no analysis options are hidden. It then calls Rocksim, which must be installed on the user’s computer, to analyze the data behind the scenes, and displays the results. Although there is nothing it can do that could not be done manually with Rocksim alone, RockShell improves the experience by making analyses:

- Less error-prone since all input variables and outputs are on a single page (ever had an edit to a simulation in Rocksim not “take”, or forget to change a value in one of the tabs?). This also greatly increases the speed at which the user can set up comparative analyses.



The screenshot shows the RockShell application window. The interface is divided into several sections:

- 1** Load .rkt Endeavour-54mm.rkt
- 2** Simulation 0 (dropdown)
Rocket Name: Public Missiles Endeavour
Motor Name: K1100T
- 3** Select an analysis (only works for dual-deployment)
 - Perform safety analysis
 - Perform recovery analysis and mapping
 - Find launch rod angle to minimize range (i.e. launch to landing distance)
 - Find launch rod angle to minimize speed at drogue deployment
 - Perform Monte Carlo mapping analysis of recovery area
- 4** Choose analysis options. These will override those in the Rocksim file.

Units	metric	Temperature	21 °C
Launch altitude	0 m above sea	Humidity	50 %
Launcher length	1.83 m	Barometer	101.3 kPa
- 5** Calculate

Safety Analysis

2.87	(CP-CG)/diameter
25.77	Thrust / weight (w/ motor) ratio
33.6	Speed off launcher (m/s)
1.01	Maximum velocity (mach)
37.0	Maximum acceleration (G)

Analysis complete. [About...](#)

My Level II Certification would have gone more smoothly with a stripped-down Safety Analysis like this. Programs that put all the options and results on a single page with no tabs make mistakes less-likely. RockShell uses your existing Rocksim files, so no data reentry is needed.

- Less tedious since RockShell may call Rocksim dozens of times as it finds optimal rod elevations or conducts Monte Carlo simulations. It also tends to speed workflow, for instance by remembering previously-entered values for launch location and launch rail length, or by having a single metric/U.S. customary units box rather than requiring separate choices for every value.

- Simply more fun, such as when displaying results on Google maps.

Existing alternatives

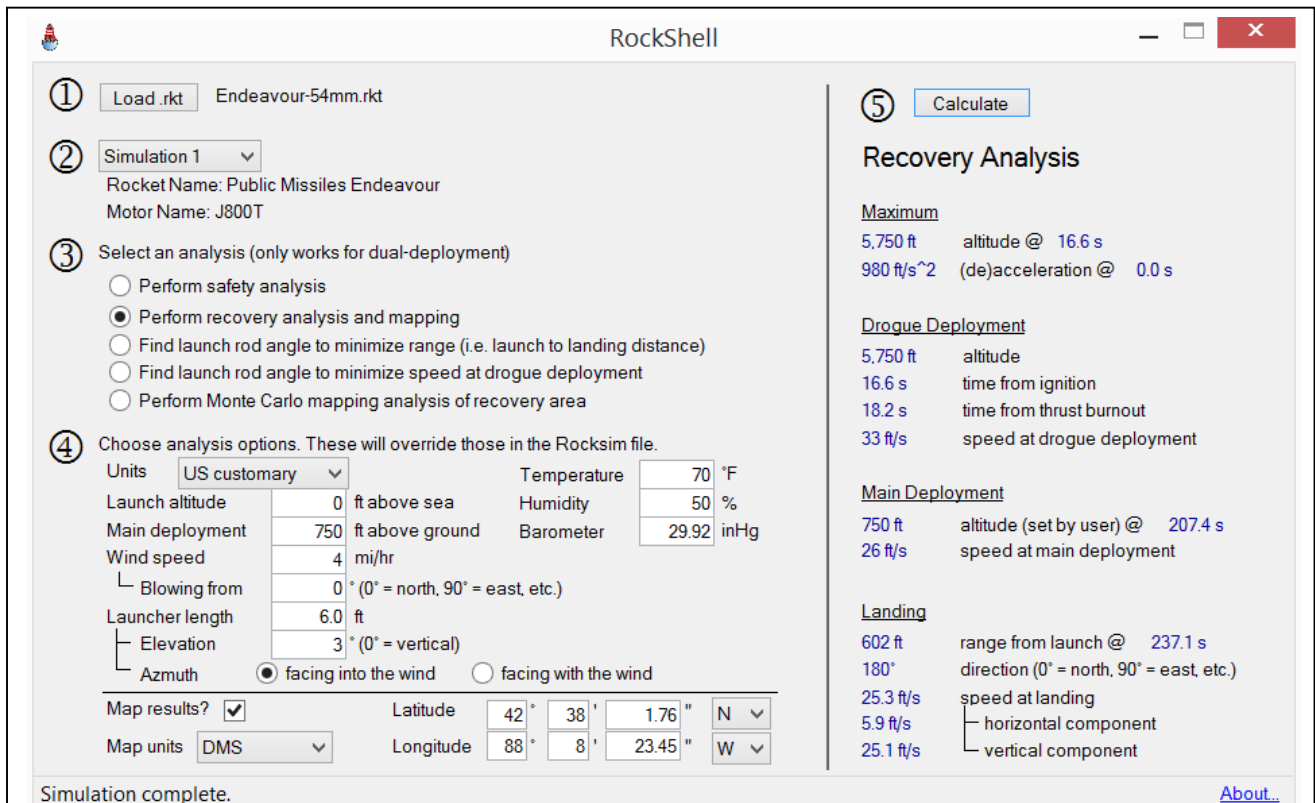
Rocksim Pro, also available from Apogee Components, will also run several optimization analyses including Monte Carlo repetitive analyses, as well as plot the results on Google Earth. Unlike RockShell, it has access to the in-

flight computed data, so it can also plot a three-dimensional flight reconstruction, rather than just launch and landing points. Its philosophy is diametrically opposite to RockShell's though; Rocksim Pro is designed to be a professional-level simulation package and as such is designed to be extremely flexible at the expense of being easy to use. In comparison, RockShell has just five specific analyses options, but all are essentially wizard-driven with enough affordances so that the new user has no doubt about what operation must be performed next or what a particular variable represents. Rocksim Pro is also priced about a ten times higher than Rocksim, and RockShell for Rocksim is free.

Open Rocket is another fantastic simulation package, and unlike

Rocksim has the ability to export in-flight data as well. It is also free, open-source, well-documented, and like Rocksim, widely-used. Unlike Rocksim it does not have a command line only version so a front-end is not possible in the current version; rather it requires embedding new analyses types as options within the main program. I chose against this since Open Rocket's philosophy, like Rocksim's, is to choose generality and flexibility over simplicity, and I wanted by design to limit functionality to a few commonly-encountered problems on launch day in order to speed calculations and limit the likelihood of user error.

RockShell has two major limitations: it is Windows-only, although it can run in a virtualized



The check on the “Map results?” checkbox of the Recovery Analysis will open a Google Map similar to the one on this article’s title page, but with a single launch and landing location. This can be run at the launch site with accurate temperature and wind speed readings.

Windows environment on a Mac or under Wine on a Linux system. The current version will only work on dual-deployment rockets, although if there is interest it would be relatively easy to create another version that works with single-deployment setups.

How to Install It

RockShell is a free download at www4.vmi.edu/faculty/squirejc/rockshell. It has no installer; instead it consists of a single executable file of about 600 kB. You can place this anywhere on your computer and double-click to run. It does not modify the registry; uninstallation is accomplished by simply deleting the file. It can be placed in your Programs folder, but since it is so small many users just keep multiple copies of it with their Rocksim .rkt data files. It requires Rocksim already installed and authorized on your system. Users are welcome to share the program with others directly, but since it is so small it is wisest to download directly from the above website to ensure you are using the newest release.

Safety Analysis

The original *raison d'être* of RockShell was to provide a minimalistic safety check of common parameters. This includes five common calculations: the number of body diameters by which the CG precedes the CP, the thrust/weight ratio, speed off the launch rail, maximum speed in Mach to determine if supersonic issues arise, and maximum acceleration in G's (actually the absolute value of maximum acceleration, since in some configurations the deceleration when the main parachute deploys is greater than motor thrust acceleration).

Recovery Analysis

This is a useful analysis to run at the launch site that can predict and map the landing location given the latest

wind, temperature, and other environmental conditions. Each analysis type remembers data from the last time it was run and from other analyses, so if for example you enter the launch location's latitude and longitude in the recovery analysis, on a subsequent opening it will repopulate the recovery analysis screen with the same values, and also carry those values over to the Monte Carlo analysis screen, making mistakes less likely.

Launcher Elevation Optimization

One of my first frustrations with Rocksim was while repeatedly editing existing simulations to find

the optimal launch rail elevation that would increase the likelihood of bringing down the rocket in the same place where it was launched. Then I discovered that when there was a stiff breeze blowing, significant weathercocking meant the danger of zippering at drogue deployment could be greater than the danger of rocket-eating trees, so launch rail elevation optimizations could instead be done to minimize horizontal speed at apogee rather than minimizing range. RockShell does both types of optimizations, and reports the desired rail elevation to within 0.5°



My son carrying my L2-certifying rocket back to the prep tables. The rocket is a modified Apogee "Level Two." I'll be using RockShell as part of my Level Three preflight, tethered by a USB cable to my phone to get the Google Maps data.

Monte Carlo Simulations

A problem with simulations is that we have limited knowledge about the environmental conditions; wind direction is constantly changing on the ground, and we have even less ability to predict or measure conditions a thousand feet higher. Monte Carlo methods handle this uncertainty by allowing the user to indicate ranges of possible wind speeds and directions, and then generating multiple launch scenarios, each with a randomly-generated wind condition within these limits. The single launch and multiple landing points are shown on Google Maps using aerial photographic overlays, quickly answering questions like “what is the likelihood of the rocket landing across the access road?” or “does the landing zone include the nearby copse of trees?” The image capture on the title page displays the type of result generated by a RockShell Monte Carlo simulation.

Write Your Own Rocksim Extension

With modern object-oriented languages even casual programmers can write their own Rocksim extension programs. The .rkt Rocksim data files are in English-readable ASCII xml format, and are divided into two major sections. The first is <RocketDesign> that describes the rocket’s physical design minus any information about the motors. All information about nose cones, bulkheads, and fin designs is described here. The second section is named <SimulationResultsList> and is an xml array of <SimulationResult>, each of which corresponds to a “simulation number” in the Rocksim GUI. Each <SimulationResult> consists of a mixture of input and output data. Input data includes the loaded motor type, the launcher information (rod length and elevation), and environmental conditions like temperature and air pressure. Once

your application updates the .rkt file, run the rocksimc command line program, passing it as parameters “myrocket.rkt -s 3 -u” to update the “myrocket.rkt” file’s simulation #3. Then reload the myrocket datafile into an xml reader to examine the updated variables in the fourth <SimulationResult> since simulation numbers are zero-based. And don’t forget to spread the word about your application among the Rocksim-using population!

Acknowledgements

Many people have contributed to beta-testing RockShell, including the Valley AeroSpace Team (VAST) NAR section of western Virginia. Special thanks go to VAST president Chuck Neff for both alpha testing and suggesting numerous improvements to the workflow. The Rocksim developers also deserve mention for their decision to create a command-line driven version of Rocksim, as well as for choosing a data file structure that is (mostly) self-documenting and in xml format, meaning they designed it to be easily-read and modified by others. I suggested adding a command-line tool as a feature request to Tim Van Milligan of Apogee Components, who replied that it’s been tucked away in there for years.