# Theme 3: Mars landing

## Task 1)

This task refers to the landing procedure on Mars in one of the videos, only now we assume that only two of the motors are working. These points in the directions and . We have three opportunities for landing that need to be examined. For all three possibilities we consider the starting point to be the origan and the destination to be P (given in a) b) and c) below). On all occasions we ask the questions:

1. Could we get there with the two engine that is now working? How much do each vector need to be scaled?
2. How do we interpret these scaling factors in terms of space/time?
3. If we cannot get there, how do we fix it? That is, if we need to add another motor, which direction will it need to point in?

The points are

## Task 2)

Now assume that our landing vessel is equipped with 4 rocket engines that may move the vessel in these four directions:

and

Where in space can be get to with these four vectors? (assume that you may move back and forth with each rocket). What happens if one of the rockets gets damaged, where can we get to then? Does this make a difference? Have a look at the vectors in a visualization tool like <https://academo.org/demos/3d-vector-plotter> or Geogebra if you are stuck.

## Task 3)

Finally, let’s assume you have another set of rockets that can move the vessel freely in these four directions:

og

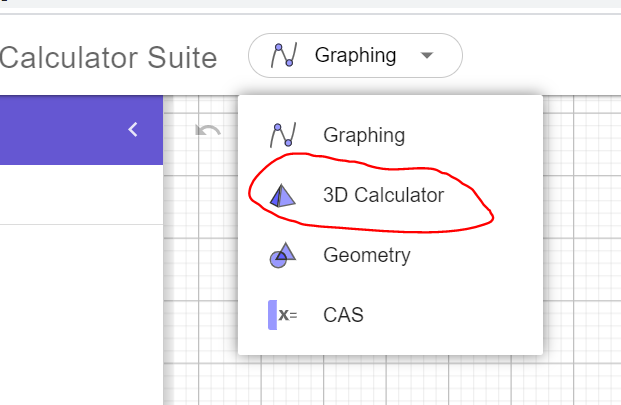
Where in space can we get with these vectors? What happens if one of the vectors gets damaged? Where can we get then? Does this make a difference?

## Task 4)

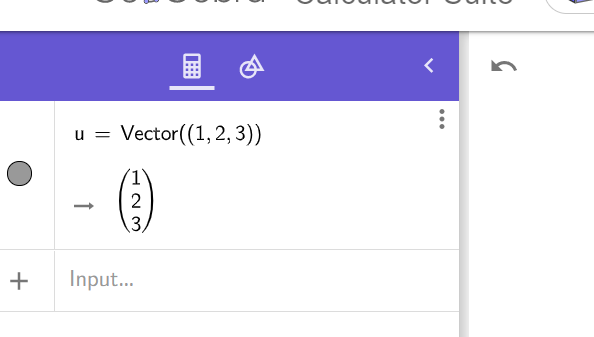
Comparing the two configurations in task 2 and 3, what can we say about the domain/area, also called Span, that the vessel can move in? What is the best configuration taking into account that one of the thrusters may malfunction? Can we understand this by using vectors/matrices/lines/planes etc?

## Appendix: About visualising 3d vectors in Geogebra

When you open Geogebra (from geogebra.org), choose 3d Calculator:



If you have specific vector you would like to visualise, use the command Vector(<point>).

 NB: Notice that there need to be a double set of parenthesises in the call to the Vector()-function, since the vector need a Point as argument, which also is specified by a set of parenthesises.