SEACITY UNIVERSITY

Faculty Design, Orbital Mechanics and Aviation Department Science

Master's Thesis

In Partial Fulfillment of the Requirements for the Degree of M.Sc.

Max Kerman
Matr.-Nr. 3141592

CommNet Kerbin System

A web of three relay satellites per body
to ensure a full coverage of the Kerbin system

first examiner: Seb Kerman
second examiner: Valentina Kerman

February 19, 2021

Contents

1 Kerbin Satellites

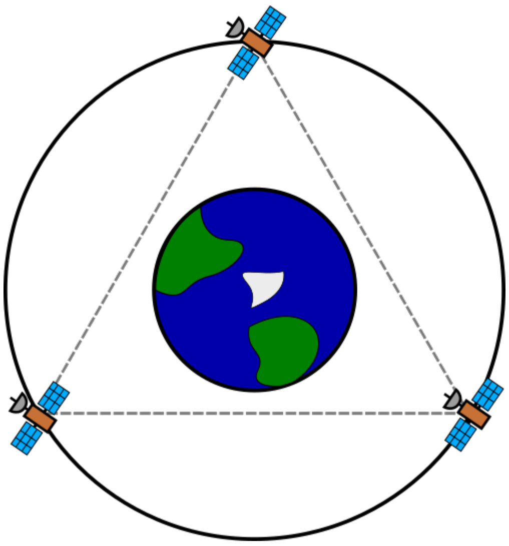


Figure 1.1: Kerbin Relay Network

We want to create a communications network with three equally distanced satellites in an equatorial orbit around planet Kerbin. There is a simple reason for this:

For the control of a spaceship, a space station or a satellite we usually need an active connection to the Kerbal Space Center (KSC). But by positioning three equally distanced satellites in the same orbit around the planet, we can prevent the connection getting cut off as soon as the KSC disappeares behind the horizon. Algorithms will automatically choose the shortest connection using as few satellites as possible.

Kerbin is a relatively small planet. Kerbin's 600km radius measures just one tenth of the radius of planet Earth. Also the game won't calculate transmission time delays (Latency), which is why we only need to ensure that the connection holds, not how good it actually is. Another point is that we want to enable interplanetary communication with our CommNet. So we want to choose a higher antenna strength than necessary for a communication between our three satellites. In order to survey as much as possible from the round planet's surface (just think about the difficulty of reaching the poles), we will set the satellites into a high orbit. The greater the distance between the sphere and the observer, the larger the visible surface gets.

However, if we wanted to create a internet network, the way SpaceX does with Starlink, then a lower orbit would be much more favourable in order to shorten the latency. But this would require much more satellites per orbit, as the communication would otherwise have to go through the planet's surface, which isn't possible after all. The satellites must be at least high enough to see each other just above the horizon.

We choose an equatorial orbit, mainly because it will require less fuel and therefore less weight for the rocket. Also the Kerbal Space Center lies directly at the equator, hence one must only "fall" to the east to get into orbit. In addition, the rotation of the earth will give a small boost to the rocket, as it doesn't need to create this starting speed by itself. On average, you will save about 175m/s.

From experience the loss of speed resulting from atmospheric drag will be about 1000m/s. More precise calculations are neither worthwhile nor possible for that matter, as the game only simulates the impact of the surrounding air. For a western start one should also calculate an additional 300m/s. For further details, see the Youtube video by Mike Aben Ascent Costs of Inclined Orbits - KSP Let's Do The Math.

If you have the ground stations on Kerbin deactivated, you won't be able to control the rocket anymore as soon as the KSC loses sight. You can either activate the ground system and keep it switched on, until your own satellite system is in operation, or you keep it off and use a kOS-script to release the satellites (more on that later).

Should you already have some space junk, with the capacity to communicate flying around from previous failed attempts, then you can just use those as a temporary replacement for a proper network.

1.1 Target Orbit

The target orbit ought to be high enough so that you could draw a straight line from one satellite to another without it touching the planet's surface. However, the target orbit should not be too high so not to leave the sphere of influence (SOI) of the planet and sail away into infinity. The best orbit height lies somewhere in between. There is also a specific satellite orbit, wherein a complete rotation around the planet takes up the same amount of time as it does for the planet to rotate around it's own axis(i.e. One Day). Thereby a person standing on the surface of the planet would view the satellite as if it was standing still. The correct term for this kind of orbit is "synchronous orbit".

The KSP Wiki states $2863.334km$ as the height of a synchronous orbit. This means that all objects in this orbit need 5h 59m 9.4s for one complete rotation around the planet.

1.1.1 Alternative Orbit

Tipp: Geostationary, Molniya, Tundra, Polar & Sun Synchronous Orbits Explained - Scott Manley
If you still want a different orbit, then you will have to deal with a little bit of geometry ;)

First we calculate the lowest possible orbit for our chosen number of satellites. The orbit height from the planet's center (radius r) we get through this formular:

$$r_{orbit} = \frac{r_{planet}}{\sin\frac{\theta}{2}} \tag{1.1}$$

The angle θ of the polygon is being calculated by:

$$\theta = \frac{(n-2)\cdot 180°}{n} \tag{1.2}$$

$(n-2)\cdot 180°$ stands for the sum of angles from the regular polygon and n for the number of angles, or rather for the number of satellites in orbit. In our case we have a triangle $\rightarrow$ so $\theta = 60°$ for each of the three interior angles.

$$r_{orbit} = \frac{r_{planet}}{\sin\frac{\theta}{2}} = \frac{600km}{\sin\frac{60°}{2}} = 1200km$$

For us this means that we have got a $1200km - 600km = 600km$ distance from the surface of the planet for the lowest possible orbit that one can achieve with three satellites at the equator.

As time goes by the satellites will slightly change their position between each other, provided that they aren't being maintained or the safefile isn't being manipulated: Kerbal Space Program - Savefile Editing & Orbital Parameters Tutorial (Fixed) - Scott Manley. This means that sooner or later the connection between the satellites will be interrupted by the surface of the planet. For this reason it's advisable to add a small buffer of about a few kilometers to a low orbit.

The orbital period of the target orbit (final orbit) is obtained by:

$$T_f = 2\pi\sqrt{\frac{a_f^3}{\mu}} = 2\pi\sqrt{\frac{(1200km)^3}{\mu}} = 4395s \tag{1.3}$$

$\mu = 3.5316 \cdot 10^{12} \frac{m^3}{s^2} = 3531.6 \frac{km^3}{s^2}$
(Kerbin standard gravitational parameter)

In a circular orbit, the semimajor axis a
is equal to the orbit radius r, the periapsis Pe and the apoapsis Ap.

But back to the "synchronous orbit".

1.2 Transfer Orbit

Now the question is how to actually deploy the satellites into their orbit while at the same time spacing them at equal distances. The Youtuber "Stratzenblitz75" has made a great video on this topic. KSP/RSS - Building a Comm Network with Math! - [Stratlab EP1] , GTO = geosynchronous / geostationary transfer orbit

We measure (or calculate) the time a satellite spends in the target orbit for one complete rotation around the planet. That time is also called "orbital period". In our case we already know from the KSP Wiki that the orbital period is about 6 hours which is the same as one Kerbin Day. And we want to release three equally distanced satellites in this orbit. This means that each satellite has a temporal distance of 2 hours (1/3 of it's orbit) to the next one.

The next thought would possibly be to just release the satellites one after the other at the same spot in the target orbit. The released satellite will of course then fly on in it's orbit and make place for the next one. If we play our cards right, then we will release the next satellite only after the former one has completed a distance of exactly 1/3 of it's orbit.

This, in turn, is possible by first putting all three satellites onto one single rocket, which will then pass the exact spot periodically. The orbit of the rocket must have the form of an ellipse. The highest part of the orbit is called apoapsis and is exactly the one that passes the target orbit. The lowest part of the orbit is called periapsis (you will also hear things like apogee or perigee) and has been deliberately chosen to make the orbital period exactly 2/3 of the target orbit. Now, everytime the rocket passes it's apoapsis, a new satellite will be released. After three full cycles, all the satellites will hopefully have been deployed. Now the rocket can begin it's deorbit burn to lower it's periapsis and to burn up in the atmosphere and simultaneously reduce space debris. This is done for safety reasons, but in KSP it primarily improves the runtime performance, as less parts will have to be rendered. Because the population density on Kerbin is very low, it isn't a problem to crash your parts into the planet, but in real life you would try to make sure that the parts burn up in the atmosphere before hitting the ground. Alternatively there is also a button inside the KSC Observation Center to delete those parts.

If, however, the lowest part of this transfer orbit goes through the surface of the planet (precisely because the target orbit is set so low), then we can just change it's orbital period to something, that isn't $2/3$ but rather $5/6$ of the target orbit. This will hopefully place the periapsis above the planet's surface (and it's atmosphere). However it should be noted that in order to reach the goal, now we can only release one satellite every two cycles around the planet, instead of every round with a $2/3$ orbit. In this case the orbital period ammounts to $5/6 \cdot 6h = 5h$.

If, at this point the lower part of this orbit is still too close to the planet, then we can decide to choose a $4/3$ orbit. This way the periapsis will be at the height of the target orbit and the apoapsis higher than that. In this case the orbital period ammounts to $4/3 \cdot 6h = 8h$.

The closer the transfer orbit is to the target orbit, the better. The satellite will have to spend less energy to get into it's final orbit. This in turn leads to a smaller satellite, as less fuel (and thrust) will be needed.

But where do we now put the transfer orbit? All that is needed are the apoapsis and the periapsis. We already got one of the two. That's the point where the transfer orbit intersects the target orbit. We can now calculate the other point with the help of the orbital period:

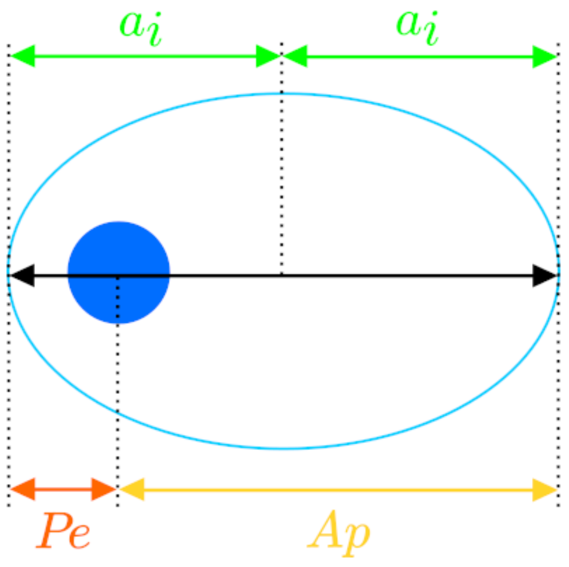


Figure 1.2: Transfer Orbit Parameter

Orbital Period from the Transfer Orbit:

$$\begin{aligned} T_i &= \frac{2}{3}T_f \\ &= \frac{2}{3} \cdot 6h \\ &= 14400s \end{aligned} \tag{1.4}$$

Or how the KSP Wiki states it:
$T_f = 5h\,59m\,9.4s = 21549.425s$
$T_i = \frac{2}{3} \cdot 21549.425s = 14366.283s$

Semimajor Axis from the Transfer Orbit:

$$\begin{aligned} a_i &= \sqrt[3]{\frac{\mu \cdot T_i^2}{4\pi^2}} \\ &= \sqrt[3]{\frac{3531.6\frac{km^3}{s^2} \cdot (14366.283s)^2}{4\pi^2}} \\ &= 2643km \end{aligned} \tag{1.5}$$

Periapsis from the Transfer Orbit:

$$\begin{aligned} Pe &= 2a_i - Ap \\ &= 2(2643km) - (600km + 2863.334km) \\ &= 1822.7km \end{aligned} \tag{1.6}$$

Here, the orbit height amounts to $1822.7km - 600km = 1222.7km$ above the surface.

Alternatively you could of course just use the online Resonant Orbit Calculator by Eric Meyer. After choosing the planetary body the calculator will present you with the required Δv for each satellite. Especially the visual presentation helps in the understanding.

1.3 Delta-v budget of each satellite

Δv can be understood as the whole change in velocity without the influence of drag and other parameters. If we were to simply separate the satellite from it's rocket then it might move away a little bit, but the orbit would stay more or less the same. To finally get into the target orbit, the satellite must first burn prograde (in the moving direction) at the apoapsis until it's velocity is as high as are all objects inside of the target orbit. For our “synchronous orbit” this would mean a velocity of $v = \sqrt{\frac{\mu}{r_o}} = 1110m/s$ for every object in this orbit, whereby r_o must be the radius of a round orbit. The higher the orbit, the higher the potential energy and the lower the speed. Or 2. Kepler’s law: “A line segment joining a planet and the sun sweeps out equal areas during equal intervals of time.”. Objects in a lower orbit must therefore move faster.

How much Δv does the satellite need to get from the transfer orbit to the target orbit? For this purpose we use the Vis-Viva Equation:

$$\begin{aligned}
\Delta v_2 &= \sqrt{\frac{\mu}{r_2}}\left(1 - \sqrt{\frac{2r_1}{r_1 + r_2}}\right) \\
&= \sqrt{\frac{3531.6\frac{km^3}{s^2}}{3463.334km}}\left(1 - \sqrt{\frac{2(1822.7km)}{1822.7km + 3463.334km}}\right) \\
&= 0.171224\frac{km}{s} \\
&= 171.224\frac{m}{s}
\end{aligned}$$

$$r_1 = 1822.7km = Pe$$
$$r_2 = 3463.334km = Ap$$

1.4 Delta-v budget of the rocket

Ascent & Injection Costs - KSP Let’s Do The Math #6
Ascent Costs of Inclined Orbits - KSP Let’s Do The Math #7

The rocket will execute three burn maneuvers:

1. Start from the surface of Kerbin, until the apoapsis is 1822.7km (1222.7km above ground).

2. Orbit circularisation burn at the apoapsis to raise the periapsis to the same height.

3. Raising the apoapsis to the height of the target orbit at 3463.334km (2863.334km above ground).

Of course one can also achieve the same with only two maneuvers, but then the accuracy would suffer. In addition one would waste a lot of fuel, because the longer the burn, the less the time at which the rocket will actually burn into the prograde direction.

Sidereal rotational velocity of Kerbin: $\overrightarrow{v_r} = 174.94 m/s$
Air drag by experience: $\overleftarrow{drag} = 1000 m/s$
Radius of Kerbin: $r_1 = 600 km$
Radius from the first preliminary orbit: $r_2 = 1822.7 km$
Radius from the target orbit: $r_3 = 3463.334 km$

Let's now calculate the necessary energy for the first level:

$$\Delta v_1 = \sqrt{\frac{\mu}{r_1}}\left(\sqrt{\frac{2r_2}{r_1+r_2}} - 1\right) = 549.89 m/s$$

$$\Delta v_2 = \sqrt{\frac{\mu}{r_2}}\left(1 - \sqrt{\frac{2r_1}{r_1+r_2}}\right) = 412.3 m/s$$

$$\Delta v_1 = \sqrt{\frac{\mu}{r_2}}\left(\sqrt{\frac{2r_3}{r_2+r_3}} - 1\right) = 201.4 m/s$$

$$\begin{array}{r} 549.89 \\ +412.3 \\ +201.4 \\ \hline 1163.59 \end{array}$$

Until now we pretended that the rocket will start from an orbit with the height of 600km. But of course the rocket starts actually from the ground's surface with the speed of the planet's own rotation. Thus, we still need the necessary velocity for an orbit of 600km, plus the speed which was lost due to air drag inside the atmosphere, minus the additional rotation speed of the planet at a launch to the east.

Theoretical orbital velocity at ground level:

$$\Delta v = \sqrt{\frac{\mu}{600 km}} = 2426.1 m/s$$

$$\begin{array}{r} 2426.1 \\ +1000 \\ -\ 174.94 \\ \hline 3251.16 \end{array}$$

Therefore the rocket requires a minimum Δv of

$$1163.59m/s + 3251.16m/s = \mathbf{4414.75m/s}.$$

If you really want to fly as economically as possible, then you should start with the very lowest possible orbit above the atmosphere, somewhere in between 70-80km. From there on you should raise the apoapsis "directly" to the height of the target orbit. After arriving at the apoapsis, the rocket should then raise it's periapsis for the correct transfer orbit.

In this case you optimally want to raise the orbit only incrementally by using the so-called "orbit phasing" method, just as Rocket Lab does it with it's Photon Kick Stage on it's way to the moon.

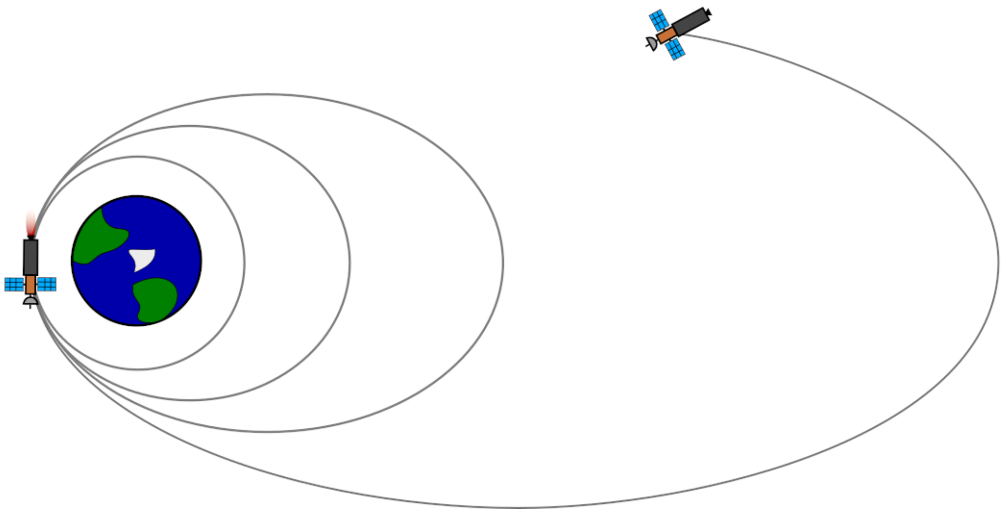


Figure 1.3: Orbit Phasing Maneuver

2 Mun Satellites

Mun is the closest neighbor to Kerbin. Strong antennas are therefore less relevant than a continuous communication connection with the vessels that might lie inside the mun shadow as seen from Kerbin (behind Mun). In the worst case scenario, a loss of connection will lead to a complete loss of control. This is particularly relevant if you want to land on the moon's surface.

Because the approach does not differ much from the one used on the Kerbin satellites, I won't go into detail as much.

Mun Properties:
Equatorial radius $= 200km$
$\mu = 6.5138398 \cdot 10^{10} \frac{m^3}{s^2} = 65.1384 \frac{km^3}{s^2}$
SOI (Sphere of influence) $= 2429.5591km$

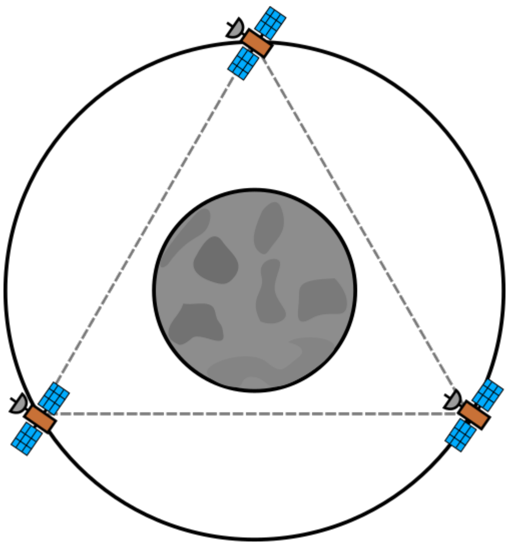


Figure 2.1: Mun Relay Network

Minimal Orbit:

$$r_{orbit} = \frac{r_{mun}}{\sin\frac{\theta}{2}} = \frac{200km}{\sin\frac{60^\circ}{2}} = 400km$$

$400km - 200km = 200km$ is the distance from the moon's surface for the lowest possible orbit with three satellites around the equator.

Maximum Orbit:

The maximum orbit lies shortly below the SOI, in other words, below the distance from which on Kerbin's pull starts to outweigh that of the moon. Also, for everything exceeding this "threshold", the point of reference is being switched for all the following calculations.

Synchronous Orbit:

Unfortunately, a synchronous orbit is not possible as the orbit radius lies beyond Mun's SOI.

Optimum Orbit:

Mike Aben specifies a value of $337.35km$ for the ideal orbit height and a value of $103.85km$ for the periapsis. Of course the apoapsis has the same height as the target orbit.
The reason: Mike Aben wants the satellites to be as small and with as few antennas as possible while maintaining a communication strength of about 80%. I can only recommend The Best Relay Orbit - KSP Let's Do The Math #13 just like the other videos on this channel.

2.1 Delta-v budget of each satellite

How much Δv does the satellite need to get from the transfer orbit to the target orbit? For this purpose we use the Vis-Viva Equation:

$$r_1 = 103.85km + 200km = 303.85km = Pe$$
$$r_2 = 337.35km + 200km = 537.35km = Ap$$

$$\begin{aligned}\Delta v_2 &= \sqrt{\frac{\mu}{r_2}}\left(1 - \sqrt{\frac{2r_1}{r_1 + r_2}}\right)\\ &= \sqrt{\frac{65.1384\frac{km^3}{s^2}}{537.35km}}\left(1 - \sqrt{\frac{2(303.85km)}{303.85km + 537.35km}}\right)\\ &= 0.0522416\frac{km}{s}\\ &= 52.2416\frac{m}{s}\end{aligned}$$

2.2 Delta-v budget of the rocket

We already know from the previous chapter that we require $3251.16m/s$ for a start from the surface of Kerbin. This figure was arrived at because of us launching from the ground instead of an orbit. Therefore we need to add the required velocity that is needed for an imaginary orbit with a radius of 600km (ground level), plus the speed which was lost due to air drag inside the atmosphere, minus the additional rotation speed of the planet at a launch to the east.

The rocket will execute four burn maneuvers:

1. Start from the surface of Kerbin, until the apoapsis is $685km$ ($85km$ above ground).

$$\Delta v_1 = \sqrt{\frac{3531.6\frac{km^3}{s^2}}{600km}} \left(\sqrt{\frac{2(685km)}{600km + 685km}} - 1 \right) = 78.956m/s$$

2. Orbit circularisation burn at the apoapsis to raise the periapsis to the same height.

$$\Delta v_2 = \sqrt{\frac{3531.6\frac{km^3}{s^2}}{685km}} \left(1 - \sqrt{\frac{2(600km)}{600km + 685km}} \right) = 76.382m/s$$

3. Raising the apoapsis to the height of Mun ($12000km$).

$$\Delta v_1 = \sqrt{\frac{3531.6\frac{km^3}{s^2}}{685km}} \left(\sqrt{\frac{2(12000km)}{685km + 12000km}} - 1 \right) = 852.61m/s$$

The "orbital inclination" is 0°. This means that Mun's orbit lies exactly above Kerbin's equator.

Velocity at apoapsis with Kerbin as point of reference:

$$\begin{aligned} v &= \sqrt{\mu \left(\frac{2}{r} - \frac{1}{a} \right)} \\ &= \sqrt{3531.6\frac{km^3}{s^2} \left(\frac{2}{12000km} - \frac{1}{6342.5km} \right)} \\ &= 178.283m/s \end{aligned}$$

4. Mun capture burn to fall into an orbit around Mun.

   The speed, at which Mun circles around Kerbin, is about $543m/s$. Minus the speed that the rocket already has at the height of the orbit of Mun, we get a velocity difference of $543m/s - 178.283m/s = 364.717m/s$. From entering the SOI of the moon on the rocket will begin to be pulled in by the moon's gravity. It is also relevant from which direction the rocket is coming and also the altitude at which it is passing above the surface. It is advisable to always include about 5-10% more Δv than necessary (especially for landing attempts on the surface of Mun), but he who is interested might also want to become familiar with the so-called Oberth Effect. This is also directly related to the topic of the so-called Gravity Assists which is especially useful for interplanetary missions.
   Alternatively you might want to take a look at the Kerbin Delta-v Map by the KPS-Community. On it, the Δv values are listed similarly to a train timetable. For the mathematicians among you, who want to know the exact origin of these values, Mike Aben has got you covered again: Calculating Transfer & Capture Costs - The Mun | KSP Let's Do The Math

   The steps are:

   a) Δv_2 (6.2) for calculating the velocity differential compared to Mun on exactly the right altitude above Kerbin in order to make rocket and moon intersect each other. Optionally look above.

$$\begin{aligned}\Delta v_2 &= \sqrt{\frac{\mu}{r_2}}\left(1 - \sqrt{\frac{2r_1}{r_1 + r_2}}\right)\\ &= \sqrt{\frac{3531.6\frac{km^3}{s^2}}{12000km}}\left(1 - \sqrt{\frac{2(685km)}{685km + 12000km}}\right)\\ &= 0.36421km/s\\ &= 364.21m/s\end{aligned}$$

b) Calculate rocket velocity at the lowest flyby above the moon:

$$\begin{aligned}
v_1 &= \sqrt{2\mu\left(\frac{1}{r_1} - \frac{1}{r_2}\right) + v_2^2} \\
&= \sqrt{2(65.1384\frac{km^3}{s^2})\left(\frac{1}{303.85km} - \frac{1}{2429.559km}\right) + (0.365km/s)^2} \\
&= 0.71299km/s \\
&= 713m/s
\end{aligned} \tag{2.1}$$

c) Calculate rocket velocity at the same spot, but within the transfer orbit (rocket's final orbit):

$$\begin{aligned}
v_2 &= \sqrt{\mu\left(\frac{2}{r} - \frac{1}{a}\right)} \\
&= \sqrt{65.1384\frac{km^3}{s^2}\left(\frac{2}{303.85km} - \frac{1}{420.6km}\right)} \\
&= 0.5233387774km/s \\
&= 523.34m/s
\end{aligned}$$

d) Difference of flyby-velocity and transfer orbit velocity at the same spot in space:

$$713m/s - 523.34m/s = 189.66m/s$$

Therefore the rocket requires $\mathbf{\Delta v \approx 190m/s}$ *in order to insert into a munar orbit with a periapsis of* 303.85km *and an apoapsis of* 537.35km. *The Delta-v Map lists a value of 310m/s, but that is because the 214km orbit used is very low.*

$$\begin{array}{lr}
 & 3251.16 \\
+ & 78.96 \\
+ & 76.38 \\
+ & 852.61 \\
+ & 190 \\
\hline
 & 4449.11
\end{array}$$

Therefore the rocket must achieve a velocity difference of around **4450m/s**.

3 Minmus Satellites

Minmus Properties:
Equatorial radius $= 60km$
$\mu = 1.7658000 \cdot 10^9 \frac{m^3}{s^2} = 1.7658 \frac{km^3}{s^2}$
SOI (Sphere of influence) $= 2247.4284km$

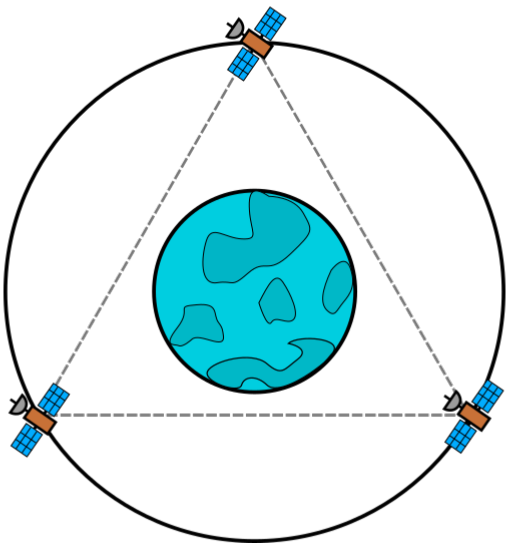


Figure 3.1: Minmus Relay Network

Because with $47000km$ distance to it's home planet, Minmus is quite a bit further away than even Mun, we will already be thinking about using stronger antennas than would be required for only the communication between the three satellites.

Minimum Orbit:

The distance from the surface of Minmus for the lowest possible orbit with three satellites at the equator is $60km$.

Synchronous Orbit:

With $357.941km$ it is located far below the SOI.

Optimum Orbit:

The distance of the synchronous orbit creates an exceptionally large viewing angle onto the moon's surface. Furthermore the very low velocity at this altitude very much simplifies releasing the satellites accurately into their final orbit. The transfer orbit has got an apoapsis of $357.941km$ and a periapsis of $159.9564km$. Alternatively you can use the online Resonant Orbit Calculator by Eric Meyer for assistance.

3.1 Delta-v budget of each satellite

$$r_1 = 159.9564km = Pe$$
$$r_2 = 357.941km = Ap$$

$$\begin{aligned}
\Delta v_2 &= \sqrt{\frac{\mu}{r_2}}\left(1-\sqrt{\frac{2r_1}{r_1+r_2}}\right)\\
&= \sqrt{\frac{1.7658\frac{km^3}{s^2}}{357.941km}}\left(1-\sqrt{\frac{2(159.9564km)}{159.9564km+357.941km}}\right)\\
&= 0.0150343km/s\\
&= 15m/s
\end{aligned}$$

3.2 Delta-v budget of the rocket

The rocket will execute four burn maneuvers:

1. Start from the surface of Kerbin, until the apoapsis is $685km$ ($85km$ above ground).

   $\Delta v_1 = 78.956m/s$ (orbit change) including $3251.16m/s$ (drag and planet rotation) $= 3330.116m/s$

2. Orbit circularisation burn at the apoapsis to raise the periapsis to the same height.

   $\Delta v_2 = 76.382m/s$

3. Raising the apoapsis to the height of Minmus ($47000km$).

$$\Delta v_1 = \sqrt{\frac{3531.6\frac{km^3}{s^2}}{685km}} \left(\sqrt{\frac{2(47000km)}{685km + 47000km}} - 1 \right) = 917.36m/s$$

The "orbital inclination" of the orbit of Minmus in comparison to Kerbin's equator is 6°. If you go about it the right way then you want to start directly into an orbit that is tilted to about 6°. If that's too difficult (because one also has to consider the perfect timing) you can first launch to the east and after reaching space turn the orbit plane to about 6°, so that when viewed from the side the orbit of the rocket and the orbit of the moon together make a single line. However, turning the orbit plane is very costly so that you will have to budget about $340m/s$ *more as one can see on the Delta-v Map.*

4. Minmus capture burn to fall into an orbit around Minmus.

   a) Δv_2 (6.2) for calculating the velocity differential compared to Minmus on exactly the right altitude above Kerbin in order to make rocket and mun intersect each other.

$$\begin{aligned}\Delta v_2 &= \sqrt{\frac{\mu}{r_2}} \left(1 - \sqrt{\frac{2r_1}{r_1 + r_2}} \right) \\ &= \sqrt{\frac{3531.6\frac{km^3}{s^2}}{47000km}} \left(1 - \sqrt{\frac{2(685km)}{685km + 47000km}} \right) \\ &= 0.227655km/s \\ &= 227.66m/s\end{aligned}$$

   b) Calculate rocket velocity at the lowest flyby above the moon:

$$\begin{aligned}v_1 &= \sqrt{2\mu \left(\frac{1}{r_1} - \frac{1}{r_2} \right) + v_2^2} \\ &= \sqrt{2(1.7658\frac{km^3}{s^2}) \left(\frac{1}{159.9564km} - \frac{1}{2247.4284km} \right) + (0.22766km/s)^2} \\ &= 0.26895km/s \\ &= 268.95m/s\end{aligned} \tag{3.1}$$

c) Calculate rocket velocity at the same spot, but within the transfer orbit (rocket's final orbit):

$$\begin{aligned}
v_2 &= \sqrt{\mu\left(\frac{2}{r}-\frac{1}{a}\right)}\\
&= \sqrt{1.7658\frac{km^3}{s^2}\left(\frac{2}{159.9564km}-\frac{1}{258.9487km}\right)}\\
&= 0.12353km/s\\
&= 123.53m/s
\end{aligned}$$

d) Difference of flyby-velocity and transfer orbit velocity at the same spot in space:

$$268.95m/s - 123.53m/s = 145.42m/s$$

Therefore the rocket requires $\mathbf{\Delta v} \approx \mathbf{145m/s}$ *in order to insert into a Minmus orbit with a periapsis of* $159.9564km$ *and an apoapsis of* $357.941km$.

$$\begin{array}{rr}
 & 3330.12\\
+ & 76.38\\
+ & 917.36\\
+ & 145\\
\hline
 & 4468.86
\end{array}$$

Therefore the rocket must achieve a velocity difference of around **4470m/s**. With an additional orbit plane change of $6°$ above Kerbin, this adds about $340m/s$ and thus results in **4810m/s**.

4 Building rockets and satellites

Kerbin

Rocket ($\Delta v = 4415 m/s$) with 3 satellites (each $\Delta v = 172 m/s$)

Mun

Rocket ($\Delta v = 4450 m/s$) with 3 satellites (each $\Delta v = 53 m/s$)

Minmus

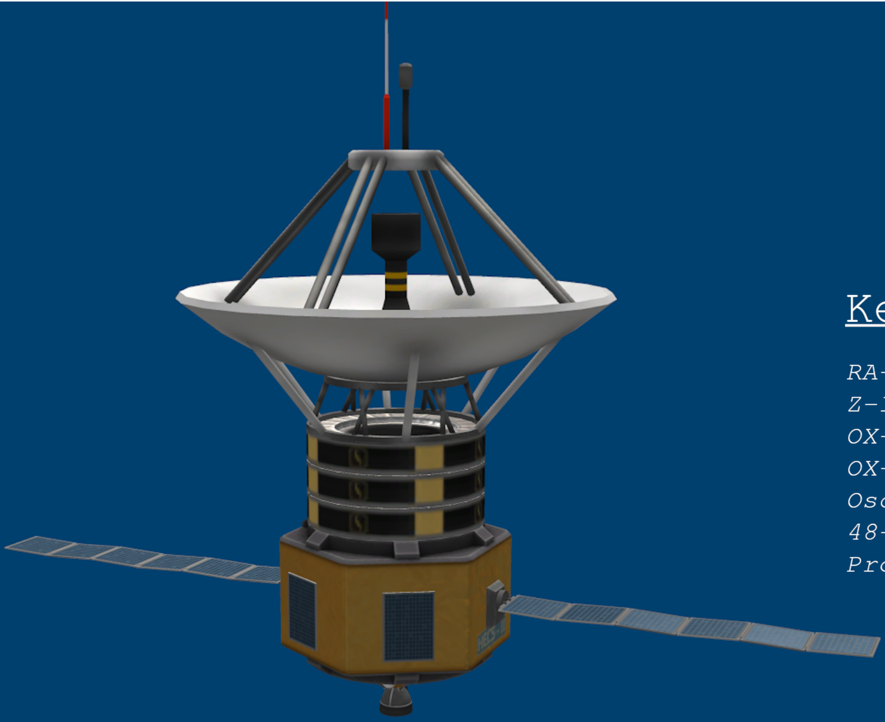
Rocket ($\Delta v = 4470 m/s$) with 3 satellites (each $\Delta v = 15 m/s$)

Electricity supply for each satellite

- Calculating Dark Side Time
- Determining Electricity Needs

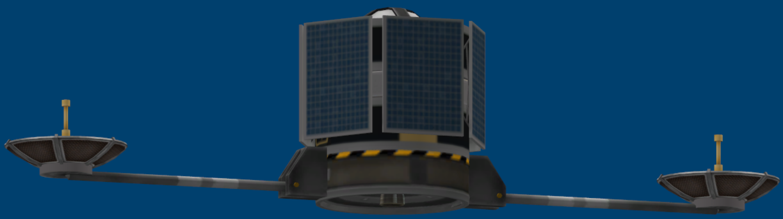
Antenna signal strength for each satellite

Communication between the three satellites and with the KSC. CommNet Wiki
A very helpful Excel document for calculating the necessary antenna signal strength you can find at CommNet Signal Strength Calculator & Antenna Selector, or https://goo.gl/Wn03VL



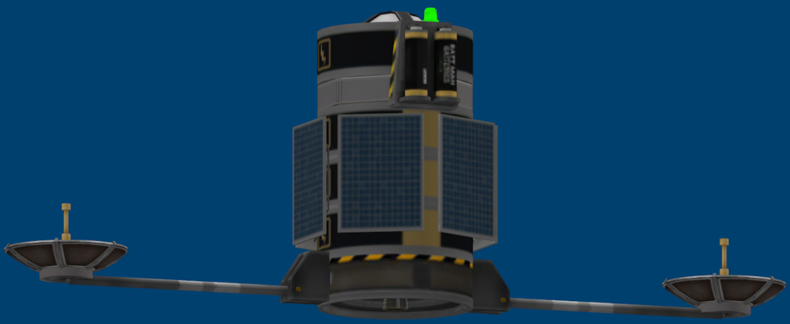
Kerbin Sat

RA-100 Relay Antenna (1)
Z-1k Rechargeable Battery Bank (3)
OX-4L 1x6 Photovoltaic Panels (2)
OX-STAT Photovoltaic Panels (4)
Oscar-B Fuel Tank (1)
48-7S "Spark" Liquid Fuel Engine (1)
Probodobodyne HECS2 (1)



Mun Sat

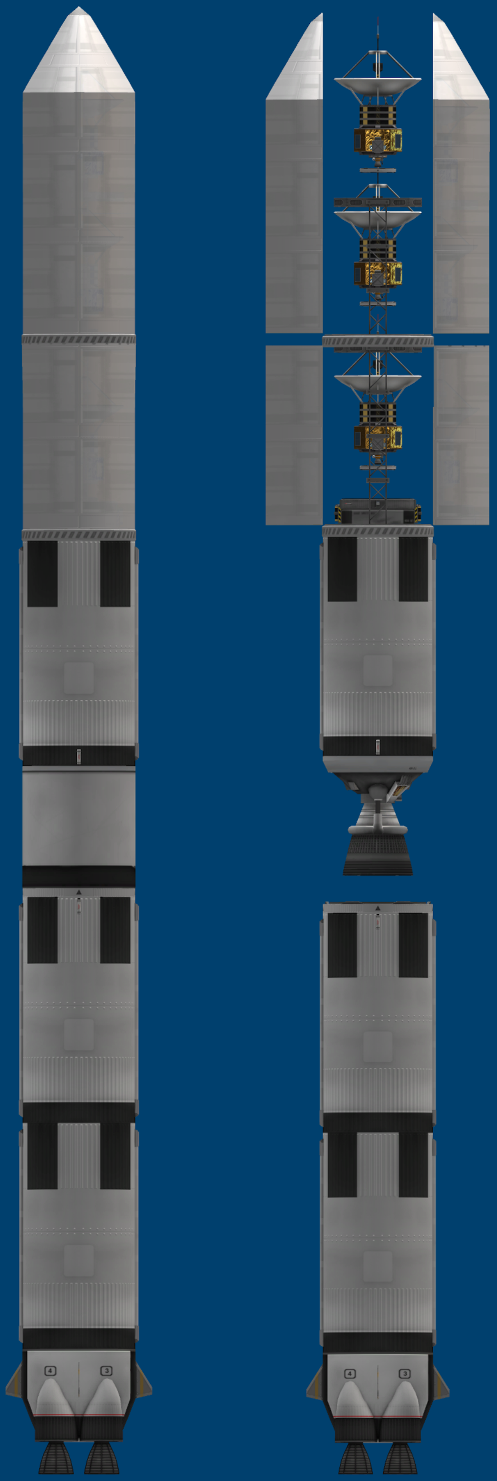
HG-5 High Gain Antenna (2)
Z-200 Rechargeable Battery Bank (2)
OX-STAT Photovoltaic Panels (6)
Stratus-V Round. Monoprop. Tank (1)
Place-Anywhere 7 Linear RCS Port (1)
Probodobodyne OKTO2 (1)
Small Inline Reaction Wheel (1)



Minmus Sat

HG-5 High Gain Antenna (2)
Z-200 Rechargeable Battery Pack (4)
Z-100 Rechargeable Battery Pack (2)
OX-STAT Photovoltaic Panels (6)
Stratus-V Round. Monoprop. Tank (1)
Place-Anywhere 7 Linear RCS Port (1)
Probodobodyne OKTO2 (1)
Small Inline Reaction Wheel (1)

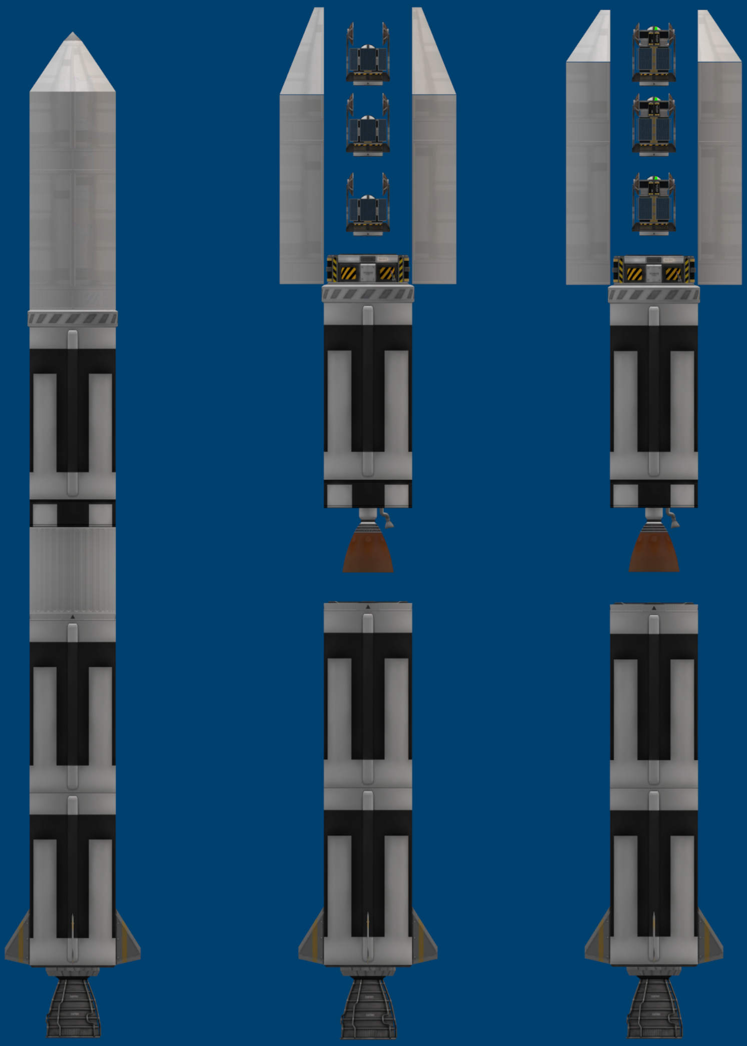
Kerbin Booster



Second Stage
- Kerbodyne KR-2L+ "Rhino"
  Liquid Fuel Engine
- Kerbodyne S3-14400 Tank
- Advanced Reaction Wheel Module, Large
- RC-L01 Remote Guidance Unit
- Z-400 Rechargeable Battery

First Stage
- S3 KS-25x4 "Mammoth"
  Liquid Fuel Engine
- Kerbodyne S3-14400 Tank

Moon Boosters



Second Stage
- LV-T91 "Cheetah"
  Liquid Fuel Engine
- FL-TX1800 Fuel Tank
- Advanced Inline Stabilizer
- RC-001S Remote Guidance Unit
- Z-400 Rechargeable Battery

First Stage
- S3 KS-25 "Vector"
  liquid fuel engine
- FL-TX1800 Fuel Tank

Helper Tools

Launch Window Planner:

- https://ksp.olex.biz/
- https://alexmoon.github.io/ksp/

Delta-V Planner:

- https://13375.de/KSPDeltaVMap/

Resonant Orbit Calculator:

- https://meyerweb.com/eric/ksp/resonant-orbits/

KSP Wiki Collection:

- https://wiki.kerbalspaceprogram.com/wiki/Calculation_tools

Addons:

- https://spacedock.info/kerbal-space-program

5 Programming the rocket

5.1 Kerbal Operating System (kOS)

kOS is a community project by the Kerbal Space Program fanbase. It's basically an autopilot that you have to write for yourself. The mod introduces a few original parts into the game, of which at least one has to be put onto the vessel in order for it to work. Each part consists of a "scriptable control system" and a limited storage medium. Also keep in mind that the control chip will require additional power.

The File-Extention for Kerboscript is always "ks", for example "launch.ks". When opening the kOS-console in the simulator, one can then switch to the "harddrive" of the spacecraft by typing

```
SWITCH TO 1.
```

1 stands for the vessel and *0* for the Kerbal Space Center. Because it seems unrealistic for the rocket to be always controled by the KSC, most players will transfer their script to the rocket before launching.

```
list.
```

shows the scripts that lie on the active harddrive. Keep in mind that the period at the end of an instruction is part of the syntax. Without it you will only receive error messages.

```
COPYPATH("0:launch", "").
```

copies the script "launch.ks" from the harddrive *0* (KSC) onto the active harddrive (here *1*).

```
RUN launch. // or else
RUNPATH("launch").
```

both start the script. The "runpath" method is simply a newer version. The following lines of code allow for the use of additional parameters set afterwards by the user.

```
runpath("launch",90,90000,True). // or else
run launch(90,90000,True).
```

The parameter **90** stands for the compass direction of rotation "east", **90000** for the target orbit height in meters and **True** for an if statement (Action Group 5), because the script is supposed to be used for different rocket types, which requires the ability to use slightly different settings.

For the beginning I want to suggest the tutorial-series by CheersKevin - kOS for Newbies. Even suitable for future programmers! Not only can you launch rockets with those scripts, but also drones and even Airplanes. Two very impressive examples see here: HoverBot drone, SpaceX Falcon Heavy.

5.2 Reaching LEO - Low Earth Orbit

Reaching the first stable orbit is always a challenge, here as well as in the real world. Apart from the gravitational pull, the rocket also has to fight aerodynamics and even with heat and g-force, if you reach a too high velocity in too low of an altitude. Furthermore the rocket has to be balanced by positioning the "center of lift" below the "center of mass". So just like a dart. Of course it helps a lot that most engines have the ability to "gimble" - that is, they can rotate the nozzle into multiple directions. In this way they can balance the rocket without the need of wings. In the real world you also want to pay attention to the weather in high altitudes. That is because in some heights, winds can reach high speeds of a few 100km/h, which is enough to split a rocket in half. Additionally there are also risks of lightning strikes because in differently charged air layers the rocket might act as a lightning rod. This in turn could potentially damage onboard circuits. Apollo 12 was victim to two such lightning strikes but was lucky insofar as it didn't create any permanent damage.

In you ever watched a rocket launch, then you might have noticed that the rocket didn't just fly straight up, but that it began to tilt sideways after just a few seconds. This is called the "pitching maneuver".
But why are rockets doing that?

The only reason why an object can stay in orbit is because it's parallel velocity to the surface is greater than the attraction in the direction of the planet. In this case the object will miss the earth while being pulled towards it, because it's speed is so fast that at the point, at which the planet has pulled the object 50 meters

downwards, the curvature of the earth will also have “made the ground 50 meters lower”. Back to the rocket..
At low altitudes the rocket is constantly being decelerated by the atmosphere. Therefore one would normally assume that it would be the best option to leave the atmosphere as early as possible by flying straight up until it’s in space. Following that, the rocket now only needs to generate speed parallel to the surface. The problem is that it normally takes a lot of time to reach the required speed of about 8km/s. In other words, you have to fly really high to have enough time left in order to reach orbital velocity. You might already have noticed that this method wastes quite a lot of energy as the vertical speed isn’t what makes one stay in orbit. That’s why people decided to use a curved trajectory which begins vertically and becomes horizontal over time.
Again: The rocket only goes “up” to get out of the atmosphere. If the rocket was standing on the surface of the moon, then it would only have to accelerate sideways. The only constraint might be the height of the mountains or the power of the engine (thrust to weight ratio).

5.2.1 kOS Gravity Turn

```
parameter compass is 90, finalApoapsis is 80000, fairingOrEscape
    is True.

SET targetPitch TO 90. //LAUNCH
SET mnvTime TO 0. //CIRCULATE
SET throttleTime TO 0. //CIRCULATE

function main {
  CLEARSCREEN.
  print "compass heading is " + compass + "degrees".
  print "finalApoapsis is " + finalApoapsis + "m".
  print "fairingOrEscape on AG5 is " + fairingOrEscape.

  doLaunch().
  doAscent().
  until apoapsis > finalApoapsis {
    doAutoStage().
    if targetPitch < 1 {
      set targetPitch to 1.
    }
  }
  doShutdown().
  doCirculate().
  print "script exited.".
}
```

```

// LAUNCH:

function doSafeStage {
  wait until stage:ready.
  stage.
  print "staging.".
}

function doLaunch {
  PRINT "Counting down:".
  FROM {local countdown is 5.} UNTIL countdown = 0 STEP {SET
    countdown to countdown - 1.} DO {
    PRINT "..." + countdown.
    WAIT 1.
  }
  lock THROTTLE to 1.
  doSafeStage().
  lock steering to heading(90, 90, 270).
}

function doAscent {
  set targetDirection to compass.
  set targetRoll to 0.
  wait until verticalSpeed >= 60.
  print "pitching maneuver started.".
  lock targetPitch to 1.48272E-8 * ship:altitude^2 - 0.00229755 *
    ship:altitude + 90.
  lock THROTTLE TO MAX(0.55, (1/90) * targetPitch).
  lock steering to heading(targetDirection, targetPitch,
    targetRoll).
}

function doAutoStage {
  if not(defined oldThrust) {
    declare global oldThrust to ship:availablethrust.
  }
  if ship:availableThrust < (oldThrust - 10) {
    doSafeStage(). wait 1.
    set oldThrust to ship:availablethrust.
  }
}

function doShutdown {
  lock THROTTLE to 0.
  // lock steering to prograde + R(0,0,270).
  lock steering to prograde.
  print "shutting down and holding prograde.".
}
```

```

// CIRCULATE:

function doCirculate {
  wait until ship:altitude > 70005.
  if fairingOrEscape {
    PRINT "AG5 on.".
    set AG5 to True.
    WAIT 0.5.
  }
  set mnvDeltaV to maneuverDeltaV().
  // parameter utime, radial, normal, prograde.
  local mnv is node(TIME:SECONDS + ETA:APOAPSIS, 0, 0, mnvDeltaV).
  add mnv. //addManeuverToFlightPlan
  set mnvTime to maneuverBurnTime(mnv).
  set startTime to TIME:SECONDS + ETA:APOAPSIS - (mnvTime / 2).
  set throttleTime to startTime + mnvTime - 0.3.
  //warpto(startTime - 40). //your choice whether to uncomment or
   not
  wait until time:seconds > startTime - 30.
  lock steering to mnv:burnvector.
  wait until time:seconds > startTime.
  lock THROTTLE to 1.
  wait until isManeuverComplete(mnv).
  lock THROTTLE to 0.
  lock steering to prograde.
  remove mnv. //removeManeuverFromFlightPlan
  print "burn finished.".
  WAIT 1. //For steering to settle down a bit
  unlock all.
  SET SHIP:CONTROL:PILOTMAINTHROTTLE TO 0.
}

function maneuverDeltaV {
  local kerbinRadius is Body:RADIUS.
  local my is constant:G * Kerbin:Mass.
  local r is SHIP:APOAPSIS + kerbinRadius.
  local a is Orbit:SEMIMAJORAXIS.
  local Vf is sqrt(my / r).  // The Velocity of any object in
   finalOrbit
  local Vi is sqrt(my * ((2/r) - (1/a))). // The Velocity at
   apoapsis
  local dV is Vf - Vi.  // Velocity-Difference from one orbit to
   the other.

  print "calculated dV: " + CEILING(dV,2) + "m/s.".
  return dV.
}

```

```
function maneuverBurnTime {
  parameter mnv.
  local dV is mnv:deltaV:mag.
  local g0 is constant:G * Kerbin:Mass. // OR: Kerbin:Mu
  local isp is 0.

  list engines in myEngines.
  for en in myEngines {
    if en:IGNITION and not en:FLAMEOUT {
      set isp to isp + (en:isp * (en:availablethrust / ship:availablethrust)).
    }
  }

  local mf is ship:mass / constant:e^(dV / (isp * g0)).
  local fuelFlow is ship:availablethrust / (isp * g0).
  local t is (ship:mass - mf) / fuelFlow. //maneuverBurnTime

  print "maneuverBurnTime: " + CEILING(t,2) + "s.".
  return t.
}

function isManeuverComplete {
  parameter mnv.
  if time:seconds > throttleTime {
    lock THROTTLE to 0.2.
  }
  if not(defined originalVector) or originalVector = -1 {
    declare global originalVector to mnv:burnvector.
  }
  if vAng(originalVector, mnv:burnvector) > 90 {
    declare global originalVector to -1.
    return true.
  }
  return false.
}

main().
```
Listing 5.1: launch.ks

The script shown here is also available on GitHub. The beginning is sometimes difficult. Nevertheless, I want to encourage you to try to write your own script. It's not that difficult. Also there are already many good examples which might help you start....for example this page.
Alternatively there is the mod Smart Parts by linuxgurugamer, for which you don't even need to know how to program. I myself often use this mod in combination, where kerboscript handles the launch and circulation burn and the

smart parts stage the fairing and other things that might differ from one rocket to the other.
The following script just executes the next maneuver node on the active vessel's orbit. Cheers to CheersKevin.

5.3 Next maneuver

```
//--------------
//  MANEUVER:
//--------------
set mnvTime to 0.
set throttleTime to 0.

function executeManeuverNode {
  local mnv is nextNode.
  set startTime to calculateStartTime(mnv).
  set throttleTime to startTime + mnvTime - 0.3.
  //warpto(startTime - 40). //your choice whether to uncomment or
   not
  wait until time:seconds > startTime - 30.
  lock steering to mnv:burnvector. //lockSteeringAtManeuverTarget
  wait until time:seconds > startTime.
  lock throttle to 1.
  UNTIL isManeuverComplete(mnv){
    doAutoStage().
  }
  lock throttle to 0.
  remove mnv. //removeManeuverFromFlightPlan
  print "script exited.".
}

// STAGING:

function doSafeStage {
  wait until stage:ready.
  stage.
  print "staging.".
}

function doAutoStage {
  if ship:availableThrust = 0 {
    doSafeStage().
  }
}

// MANEUVER:

```

```

function calculateStartTime {
  parameter mnv.
  print "Node in: " + round(mnv:eta) + "s, DeltaV: " + round(mnv:
   deltav:mag) + "m/s.".
  set mnvTime to maneuverBurnTime(mnv).
  return time:seconds + mnv:eta - mnvTime / 2.
}

function maneuverBurnTime {
  parameter mnv.
  local dV is mnv:deltaV:mag.
  local g0 is ship:Body:MU.
  local isp is 0.

  list engines in myEngines.
  for en in myEngines {
    if en:IGNITION and not en:FLAMEOUT {
      set isp to isp + (en:isp * (en:availablethrust / ship:
   availablethrust)).
    }
  }

  local mf is ship:mass / constant():e^(dV / (isp * g0)).
  local fuelFlow is ship:availablethrust / (isp * g0).
  local t is (ship:mass - mf) / fuelFlow.

  print "maneuverBurnTime: " + round(t) + "s.".
  return t.
}

function isManeuverComplete {
  parameter mnv.
  if time:seconds > throttleTime {
    lock throttle to 0.2.
  }
  if not(defined originalVector) or originalVector = -1 {
    declare global originalVector to mnv:burnvector.
  }
  if vAng(originalVector, mnv:burnvector) > 90 {
    declare global originalVector to -1.
    return true.
  }
  return false.
}

executeManeuverNode().
```
Listing 5.2: mnv.ks

For decoupling the satellites from their rocket one could use a timer that uses the orbital period as basis. The satellite could then activate it's engine when the apoapsis or the start time has been reached.

6 Orbital Mechanics Crash Course

6.1 Vis-Viva Equation

Calculating the Δv budget for a Hohmann Transfer (Hohmann Transfer).

$$\Delta v_1 = \sqrt{\frac{\mu}{r_1}}\left(\sqrt{\frac{2r_2}{r_1+r_2}} - 1\right) \tag{6.1}$$

$$\Delta v_2 = \sqrt{\frac{\mu}{r_2}}\left(1 - \sqrt{\frac{2r_1}{r_1+r_2}}\right) \tag{6.2}$$

Δv_1 stands for the cost of the first burn that raises one side of the orbit to the height of the target orbit.

Δv_2 stands for the cost of the second burn that will be executed at the intersection with the target orbit and will transfer the current orbit towards the target orbit.

Together they give the Δv budget of each spacecraft that transfers from one circular orbit into another circular orbit.
Instead of $\sqrt{\frac{2r}{r_1+r_2}}$ one can also write $\sqrt{\frac{r}{a}}$. The original fraction divided by two shows the radius as the numerator and the semimajor axis as the denominator. For learning the origin of this formular I suggest the video by Mike Aben. Or perhaps on Wikiwand: Hohmann transfer orbit.

The Vis-viva equation for the instantaneous velocity of a body that is located on an orbit around another body is called:

$$v = \sqrt{GM\left(\frac{2}{r} - \frac{1}{a}\right)} \tag{6.3}$$

where:

- v is the relative speed of the two bodies
- r is the distance between the two bodies

- a is the length of the semi-major axis ($a > 0$ for ellipses, $a \to \infty$ for parabolas, and $a < 0$ for hyperbolas)
- G is the gravitational constant
- M is the mass of the central body

The product of GM can also be expressed as the standard gravitational parameter using the Greek letter μ.
If $a = r$ for all r, then the Keplerian orbit will degenerate to a circular orbit; the body always has got the same distance r from it's center of gravity and accordingly always the same **orbital velocity**

$$v_1 = \sqrt{\frac{G(m+M)}{r}} = \sqrt{\frac{\mu}{r}} \tag{6.4}$$

The formula for **escape velocity** can be obtained from the Vis-viva equation by taking the limit as a approaches ∞:

$$v_2 = \sqrt{\frac{2G(m+M)}{r}} = v_1 \cdot \sqrt{2} \tag{6.5}$$

If the orbital velocity is equal to v_2, then the orbit isn't a closed circle anymore, but rather a parabola. At even higher speeds, while r being the same, the path becomes a hyperbola and the semi-major axis a becomes (formally) negative.

You will sometimes see the mass m of the spacecraft being omitted. This, of course, can easily be explained, as it's mass compared to that of the planets is so insignificantly small that an effect is just not measureable. If for instance you plan to circle an Asteroid, then of course you will have to account for that also. (Not in the game though. Only the sun, planets and moons have gravity there.)

6.2 Other Stuff

Also insteresting to note: Until an **altitude** of somewhere around 500km there is still a lot of atmosphere left, enough to brake out and deorbit satellites over time. In the past there were only spy-satellites in very low orbits. 340km is already the minimum height that a satellite can hold without deorbiting. But even the ISS, with it's 400km is not that high, if you think about it. When the space shuttles were still flying, the ISS was even lower. On this altitude, the residual atmosphere is still so high that the ISS has to raise it's orbit about once a year. There is a wonderful animation QUARTZ - every active satellite orbiting earth that illustrates this pendulum motion very well.

And another three-dimensional representation of all near earth objects you can find at stuffin.space. You can even hide certain satellites and for instance only display GPS satellites. By the way, GPS satellites usually live inside the Medium Earth Orbit (MEO).

In case of **SpaceX Starlink**, it's satellites only stay in orbit for as long they still have fuel in their tanks. That makes maintenance of those networks a little bit more difficult, but you have the certainty that any defect satellites will deorbit automatically after 5 years and that no space debree will ever be left behind. Unlike other satellite providers who also use ion thruster, SpaceX decided to use Krypton as reaction mass instead of Xenon, which is more common. Another name you may find is Hall-effect thruster (HET).

Factor of safety often lies between 1.1 and 1.25, as is the case for ULA's rockets. It expresses how much stronger a system is than it needs to be for an intended load. For spacecrafts this factor is usually very low, because every millimeter that you add to the thickness of a tank wall, will probably add many kilos or even tons of extra weight, which in turn requires a stronger engine and more fuel...which adds more weight...which reduces the payload mass that the rocket can bring with it into space.
In contrast, a tractor can have a much higher FoS, because stability and sheer power are much more important than are in this case weight or speed. See also HOW ROCKETS ARE MADE (Rocket Factory Tour - United Launch Alliance) - Smarter Every Day 231.

List of Figures

Listings

Hereby, I declare that I have composed the presented paper independently on my own and without any other resources than the ones indicated. All thoughts taken directly or indirectly from external sources are properly denoted as such.

SEACITY, SEPTEMBER MAX KERMIN

Place, Date Signature