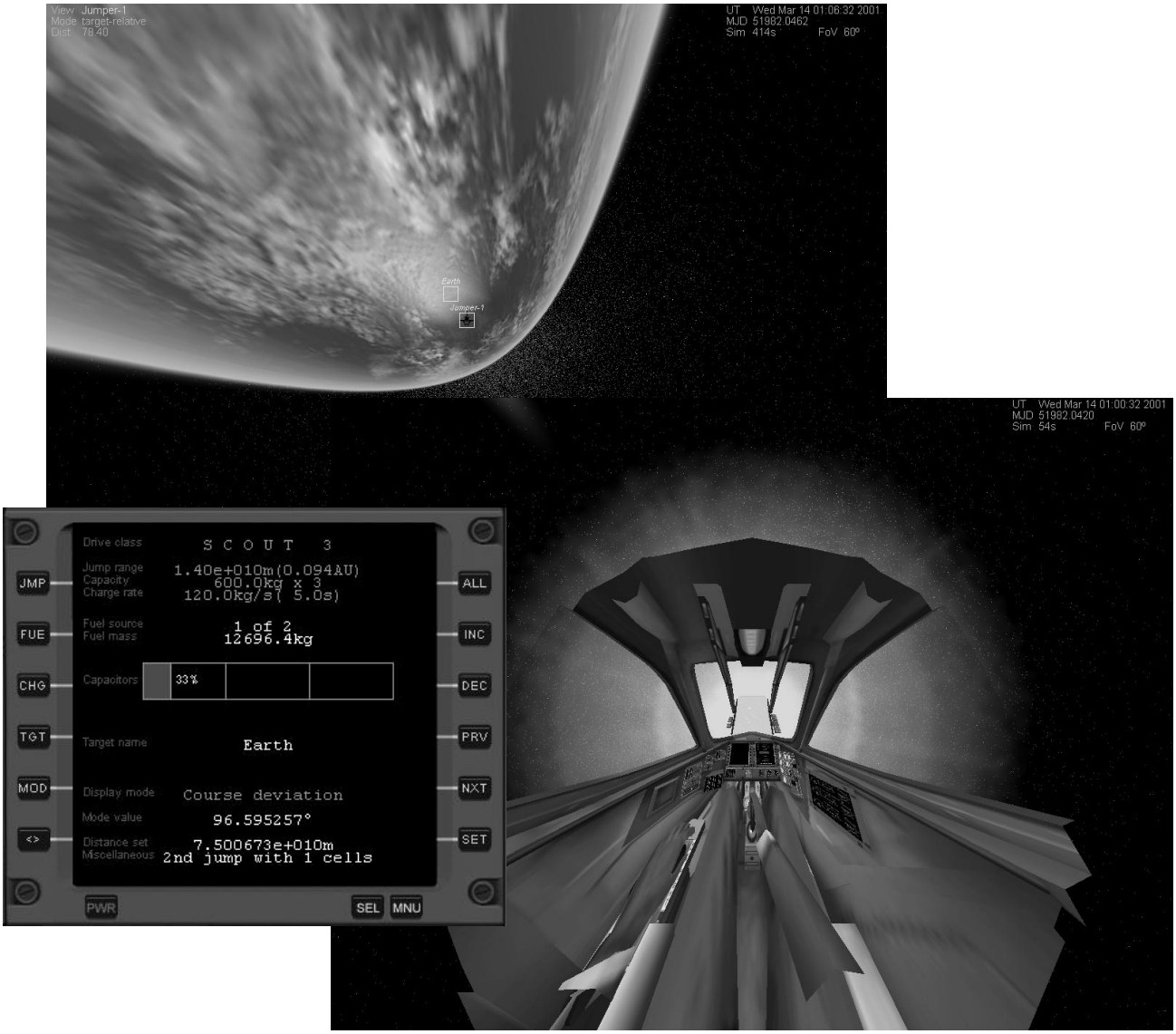
ORBITER
Jump Drive MFD V1.0

Friedrich Kastner-Masilko
September 20th 2006



1. INTRODUCTION

This Orbiter module provides hyper-jump capabilities for every vessel in your scenario. To make traveling with a jump-drive more challenging, the jumping distance is fixed (although adjustable via INI file). Therefore, to get near your target, you have to think about the course of the jump. The integrated navigation computer helps you with this task.

This work basically made me relax a little from OMP and build up some MFD-creating skills. Don't take it (the plugin, the code and this documentation) too seriously. Have fun!

2. OVERVIEW

Based on the ideas of ZCochrane[1], a member of the Orbiter forums, this MFD equips every vessel in your Orbiter environment with the ability to instantly jump over huge distances. Although this is nothing new to Orbiter (there are several MFDs doing this job), this jump drive is quite different: the jump length can not be set arbitrary!

The simulated jump drive is claimed to work as follows[2]:

- Mass from any fuel resource in your vessel can be used to charge up so called "capacitor cells" through "Mr. Fusion". Unfortunately, "Mr. Fusion" doesn't work the other way round. Once you filled the cells, you can't use them for anything else than jumping.
- Each kilogram of fuel loads your cell with a fixed percentage. This way, fuel consumption is linear over filled cells.
- The jump engine itself ("Flux-Capacitor") uses the charged cells to build up a bursting warp bubble, causing the vessel to jump incredible distances through space in the direction it is pointing in the blink of an eye.
- The more cells the engine can use, the more far your jump will be. Luckily, hyperspace physics are non-linear, so your jump length will grow exponential over filled cells. Unfortunately, only full cells can be used, because only harmonic energy levels result in bursting warp bubbles.
- In spite of the instantious jump itself – looking like a big explosion for observers – the complete jumping procedure takes about 14 seconds. During this time, the "Flux-Capacitor" builds up the warp bubble, causing your vessel to accelerate like crazy and the space around you to warp even crazier.
- After the jump the vessel decelerates again, bringing the vessel back to relativistic (in case of sane vessels and pilots: newtonian) physics. Your kinetic energy is preserved, so you have to use your main engines to get into an appropriate orbit around a target object.

Due to the implicit discrete jump length, the jump drive needs a more sober navigator than any other FTL system. You don't want to jump right into Jupiter's core by accident, believe me…

To learn more about the theory of jump drive navigation and the famous Z-formulas[3], please take a look at section 5. The usage of the MFD is described in section 6.

3. DOCUMENT HISTORY

Version 1.0

September, 20th 2006 – Release documentation.

[1] Quote from **Harmsway!** in http://orbit.m6.net/Forum/default.aspx?g=posts&t=7886 : *„And who better to be the creator of a warp drive for orbiter then ‚ZCochrane'"* ...no more comments needed...

[2] Some of the terms in this description may cause you to think they are stolen from the movie „Back to the Future"... ridiculous... Think about it: on one hand you have a simple jump drive, on the other there is a time(!) machine built into a DeLorian(!). Now, what's the real one???

[3] **ZCochrane** was the first to come up with a formula for jump drive navigation. Although his work only took equidistant jump drives into account (and many outcries from mathematicians like "This is just plain trigonometric, you fools!!!"), many jump drive researchers and engineers refer to the set of formulas regarding jump drive navigation as "Z-formulas" in honor of the man that first build this creepy travel system out of dust-covered jump gates.

4. INSTALLATION

Extract the archive with your Orbiter folder as base folder and be sure to enable the "use path information" feature of your favorite extraction tool.
You should then have:

<OrbiterFolder>\Config\hyperspace.cfg	Configuration file for hyperspace fake vessel
<OrbiterFolder>\Doc\JumpDriveMFD.pdf	This documentation
<OrbiterFolder>\Meshes\hyperspace.msh	Mesh for hyperspace fake vessel
<OrbiterFolder>\Modules\Plugin\JumpDriveMFD.dll	JumpDriveMFD plugin
<OrbiterFolder>\Scenarios\JumpDriveShow.scn	Showcase scenario
<OrbiterFolder>\Sound\JumpDriveMFD*.wav	Sound files for OrbiterSound3 support
<OrbiterFolder>\Orbitersdk\samples\JumpDriveMFD*	Source code
<OrbiterFolder>\JumpDriveMFDReadme.txt	Installation instructions

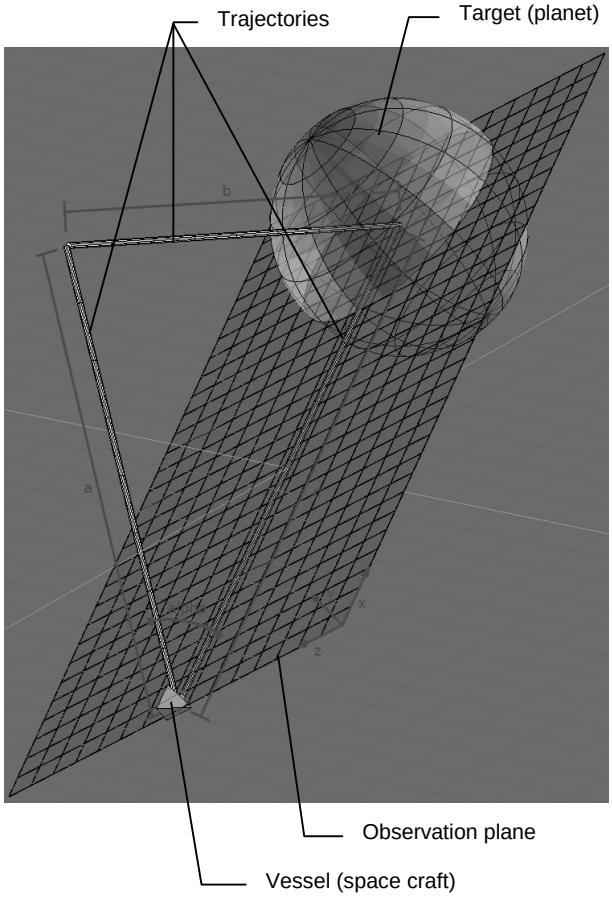
There is no configuration file (INI) in the distribution, because it will be generated from scratch.

5. CONCEPT

The jump drive is a pretty straight forward thing to use. Fill it up, point where you want to go and jump…
The problem is to jump to a specific location in space, though. Due to the fixed jump length (although somewhat selectable by the amount of filled cells) you can't just point into the direction of your target and go, because you either fall short of your target or simply overshoot it most of the time.
Additionally, the jump drive is no magic *"Standard orbit, Mr. Sulu"*-device. Your velocity vector w.r.t. the target object AFTER the jump is the same as BEFORE the jump.

Fortunately, the described problems are solvable by math, summarized in the so-called Z-formulas, which will be described in the following section.

Definitions:



If we think about the problem of getting to a target by fixed-length jumps (FLJs), the 2-jump solution quickly comes to mind. I.e., the "last mile" in a stutter-jump journey with FLJs always needs 2 jumps to reach the exact position of the target.

The figure to the left shows the 2-jump solution in generalized 3D space. You can make out the vessel, the target, the jump trajectories and the observation plane.

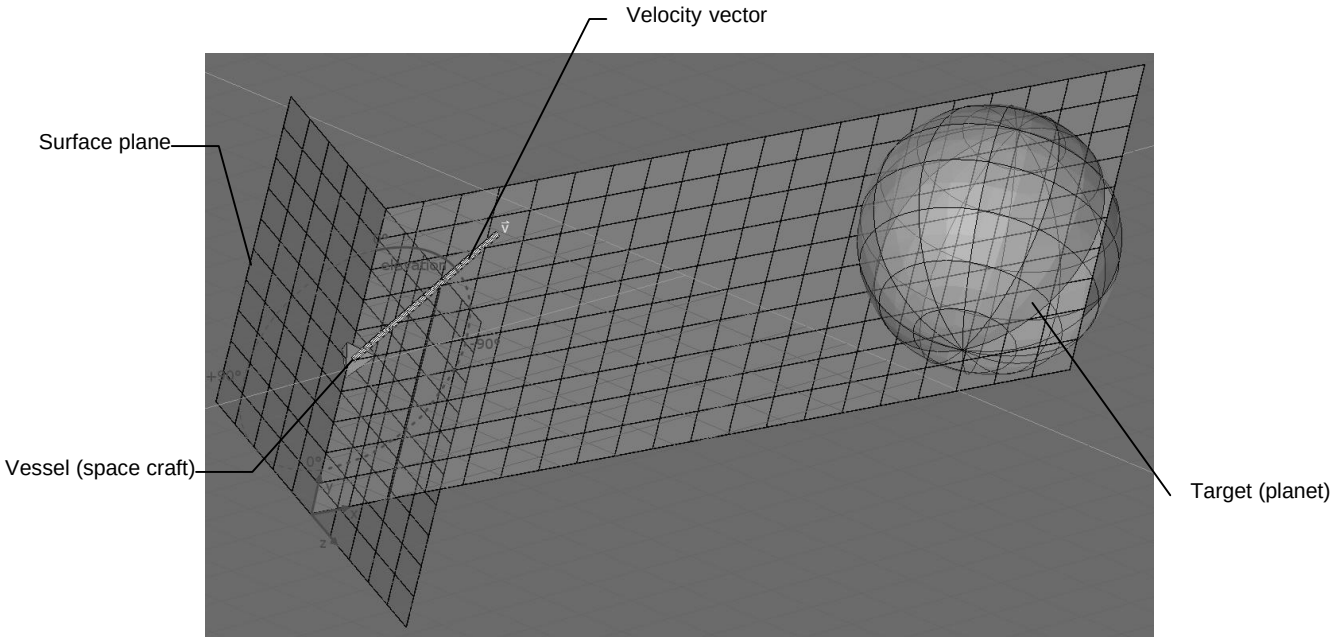
The Euclidean distance from vessel to target has length c and the corresponding vector defines the x-dimension of the observation plane. a is the length of the first jump trajectory and b is the length of the second one. The plane spawn by the triangle a-b-c is perpendicular to the observation plane.

If the x-y plane is used for projection of this 3D constellation, a 2D model of the problem can be defined. This way, the calculation of the angle between the direct course to the target and the needed derivation of the first jump (depicted as ***alpha***) can be calculated more easily.

As ***alpha*** was the interesting angle regarding position w.r.t. the target, the ***elevation*** is the one regarding velocity. The ***elevation*** is the angle between the velocity vector w.r.t. the target and the surface plane. The later is the y-z plane as shown in the figure below.

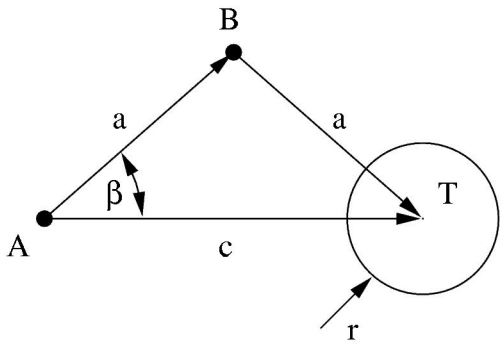
As you can see, a vector pointing towards the target gives a negative elevation, whereas pointing away from the target gives a positive one.

Of course, both picture's x-y plane must match. I.e., for the simplification of both models to work, the chosen direction of the first jump must lie in the same plane as the velocity vector.



The following formulas represent analytical solutions for various simplified models. Note: angle ***alpha*** from above figures is named β in the formulas due to trigonometric standard notation, ***elevation*** is abbreviated with ε.

<u>Z-formula 1 (original) – equidistant jumps:</u>

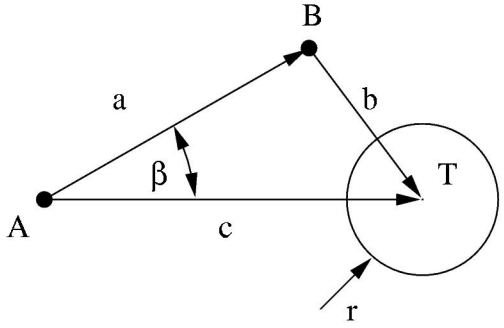


If we assume equidistant jumps, the angle can be solved by:

$$\cos\beta = \frac{c}{2a} \Bigg| \begin{matrix} \forall a,c \in R_+ \\ a \geq \frac{c}{2} \end{matrix}$$

Note: if β is near 60° , you may reach your target with a single jump!

<u>Z-formula 2 (extended) – generalized jump length:</u>



General FLJ-angles can be solved by using the cosine sentence:

$$\cos\beta = \frac{a^2 - b^2 + c^2}{2ac} \Bigg| \begin{matrix} \forall a,b,c \in R_+ \\ -1 \leq \cos\beta \leq 1 \end{matrix}$$

As you can see, the possible length of each jump is limited. If we take the exponential scaling of the jump drive into account, the possible range of filled cells for the second jump can be described as follows:

$$\left\lceil \ln_s \frac{|a-c|}{j} \right\rceil \le y \le \left\lfloor \ln_s \frac{a+c}{j} \right\rfloor \Bigg| \begin{matrix} a = js^x \\ b = js^y \\ j, s \in R_+ \\ x, y \in N_0 \end{matrix}$$

With ***j*** being the jump length with a single filled cell and ***s*** being the scaling factor, ***x*** is the amount of additional filled cells for the first jump, ***y*** the amount for the second. Then ***y*** must lie within the given range. If it is impossible to find a solution for ***y*** (because of the range limits or the amount of available cells for the drive), the target can not be reached with the specified first jump length.

Z-formula 3 – horizontal velocity:

The generalized Z-formula is already sufficient, if you want to jump directly into Jupiter's core… Maybe someone is considering that, but most of the time you want to jump into some kind of orbit around a planet. Then we have to take the velocity vector into account. As described above, we assume the velocity vector to lie in the same plane as our planed jump trajectory. If the elevation of the vector is nearly horizontal, we can use the following simplified model to calculate the needed course deviation:

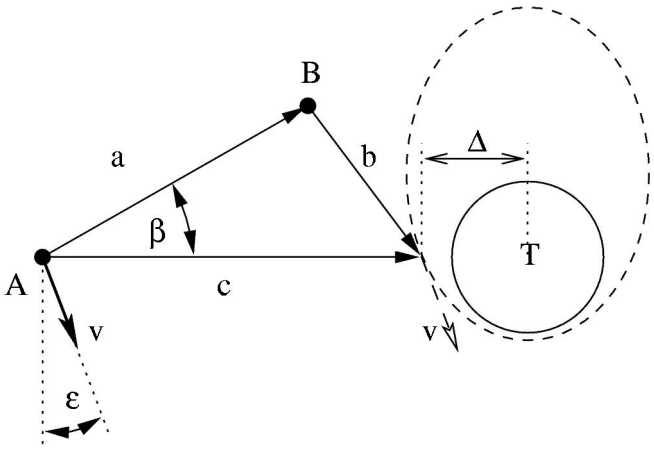
$$k = \frac{2GM}{\Delta v^2} \Bigg| \begin{matrix} \Delta, v \in R_+ \\ G = 6.67259 \cdot 10^{-11} \end{matrix}$$

k as the destination factor depends on the target's mass ***M***, the velocity ***v*** w.r.t. the target and the estimated distance Δ from the target after the jump.

With the destination factor and the elevation, the distance specific radius ***R*** solves to

$$R_{1,2} = \frac{-k \pm \sqrt{k^2 - 4(1-k) - \sin^2(90-\varepsilon)}}{2(1-k)} \Bigg| \begin{matrix} \sin^2(90-\varepsilon) \le k^2 - 4(1-k) \\ k \ne 1 \end{matrix}$$

with 2 solutions for ***R*** (one with the positive square root result, one with the negative).



Multiplying both solutions with the estimated distance gives you the values for the resulting orbit's periapsis R_{per} and R_{apo} apoapsis:

$$R_{per} = R_1\Delta \quad ; \quad R_{apo} = R_2\Delta$$

With the sum, you can calculate the semi-major axis of the orbit's shape:

$$s_{maj} = (R_1 + R_2)\frac{\Delta}{2}$$

Even the eccentricity can be derived from the R solutions:

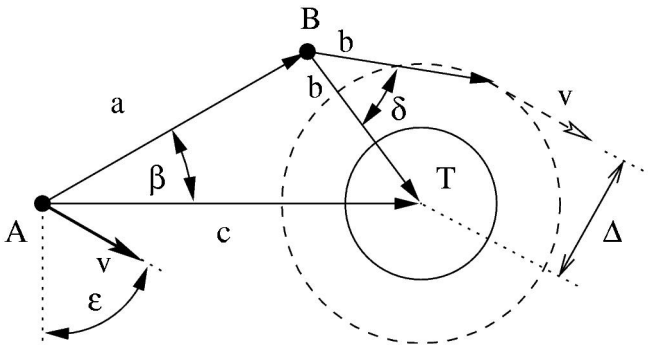
$$e = 1 - \frac{2R_1}{R_1 + R_2} = \frac{2R_2}{R_1 + R_2} + 1$$

If the resulting shape is not acceptable (e.g. if R_{per} is below the targets ground level), you must start with a new estimated distance. Once the shape is acceptable, calculate the FLJ-angles according to the extended Z-formula.

Z-formula 4 – vertical velocity:

Z-formula 3 can only calculate nearly horizontal (0° to 45°) velocity vectors efficiently. If you have a more vertical vector, the following model is better suited, but still less accurate.

Please note, that the elevation depicted in the following figure has a negative value, since it points towards the target. The simplified model only works with a velocity vector pointing this direction. If you have a positive (nearly vertical) elevation, just mirror the trajectories over the x-axis.



To get the angle δ for the deviation of the second jump, use

$$\delta = 2\left(|\varepsilon| + \arcsin\left(\frac{a}{b}\sin\beta\right)\right) - 180$$

You will then have an offset to the target midpoint of

$$\Delta = 2b\sin\frac{\delta}{2}$$

With this and the fact, that the new elevation at the new endpoint is zero, you can calculate your final orbit's shape by using Z-formula 3.

Best practice:

Since hyperspace physics is somewhat inaccurate today, don't do too much math with your approaches. Use the navigation computer's modes to calculate your FLJ according to Z2 as exact as possible and guess your final jump to get into a more or less stable orbit.
As you gain skills you can plan your approaches with Z3 and Z4, but it is not necessary for quick tours around the solar system. Just don't ignite Jupiter…

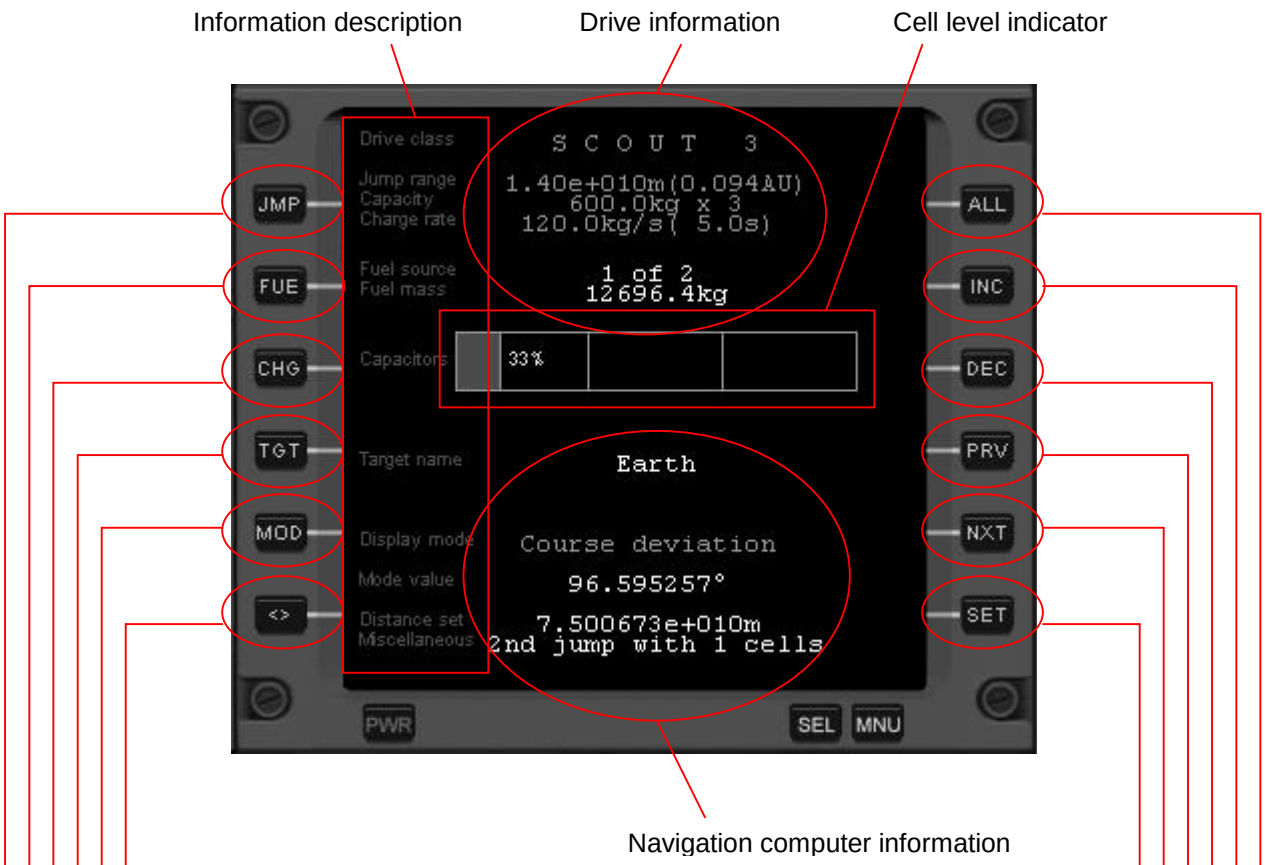
6. MFD MODE USAGE

The behavior of the MFD mode can be summarized as follows:

- The MFD mode for jump drive control is called "JumpDrive" and the activation key is 'J'. Activate the mode as described in the Orbiter manual.
- Every vessel in the scenario gets it's own independent jump drive instance that can be controlled from every MFD inside the vessel. Therefore, concurrently jumping vessels are possible.
- Every MFD instance (standard MFD and ExtMFD) stores the information for the JumpDrive mode executed in it. I.e., all your navigation computer settings won't be shared between MFDs. So you can e.g. target different objects in different MFDs.
- Every command can be engage either by button press or shortcut. If OrbiterSound3 is installed, a positive acknowledgment is expressed by a clicking sound, a negative one by a high pitched tune.
- Only vessels NOT docked to a superstructure can engage a jump. Of course, during the jump procedure it is possible to accidentally "catch" a superstructure. A docked structure going into hyperspace causes structures to vaporize (attachments work fine, though). Try it out…

Navigation computer layout:

The following page describes the layout and button functions of the MFD mode.



Shortcut key	Description	Remarks
S	**Select distance**	Opens a dialog where you can enter the distance and cell amount for the 2nd jump for calculation of course derivation according to Z2. Distance and cells are separated by comma, distance can be left out if you just want to change cell amount and vice versa.
U	**Set to current**	Set distance to current target distance and switches the navigation computer mode to *Target distance*.
M	**Change display mode**	Rotates navigation computer display mode.
N	**Next target**	Selects next target by increasing the object index. Normally, Orbiter enumerates planets and moons first, then vessels. Bases or navaids are not enumerated.
P	**Previous target**	Selects previous target by decreasing the object index.
T	**Select target**	Opens a dialog for selecting of target object by name.
C	**Charge up cell**	Starts capacitor cell charging.
D	**Decrease index**	Decrements fuel source index. If the index was 1, fuel input is switched off. If fuel was switched off, the index will be set to the maximum index available.
I	**Increase index**	Increments fuel source index. If the index was the maximum available index, fuel input is switched off. If fuel was switched off, the index will be set to 1.
F	**Select fuel index**	Opens a dialog for selecting the fuel source index.
A	**Fleet jump command**	Automatically sets the same course and engages the jump for all vessels within command range if their capacitor is ready.
J	**Engage jump**	Engages the jump if capacitor is ready.

Navigation computer modes:

The following table describes the available computer modes in the rotation order.

Mode	Description
Target size	Displays the target object's size in meters. If the target is a celestial object, the size is the object's radius. This information can be used for Z3 and Z4 calculations or eye-balled approaches.
Target mass	Displays the target object mass in kilograms. This information can be used for Z3 and Z4 calculations.
Target distance	Displays the distance to the target object's core in meters.
Course offset	Displays the offset in meters to the destination point (the one with zero deviation), if the jump occurs with the current course deviation. Useful in the last jump to avoid collisions.
Course deviation	Displays the offset in degrees from pointing towards the target object's core.
Deviation needed	Displays the offset in degrees needed for the first FLJ for the given distance selected according to Z2.
Deviation offset	Displays the offset between current and needed derivation. Use it to set your course according to Z2 calculation.
Relative velocity	Displays the velocity vector length w.r.t the target in m/s. This information can be used for Z3 and Z4 calculations.
Velocity elevation	Displays the angle between the surface plane and the velocity vector in degrees (0=orbital velocity vector, -90=pointing towards target, +90=pointing away from target). This information can be used for Z3 and Z4 calculations.
X-Y(pitch)alignment	Helps you with aligning X-Y planes of velocity and course model. Set course towards the target, roll until the angle is zero, then pitch up until your course deviation is as calculated in Z3 and Z4, respectively. The calculations are only valid as long as the plane angle (shown in emphasis) is close to zero.

Jump drive parameters:

The jump drive MFD mode can be adjusted in it's behavior regarding memory and cell management as well as effect and feature settings. These parameters can be set in an appropriate INI file. It is located in the /Modules/Plugin/ folder and will be generated from scratch - if not present - by the plugin with the following default values:

```
[Sizes]
Drives=4                        = 2^Drives=Size of the internal drives hash
MDFs=4                          = 2^MDFs=Size of the internal MFD hash

[Effects]
JumpVelocity=5.000000e+004      = Acceleration/Deceleration velocity offset
                                  [m/s] for jump effect
FlashOffset=1.000000e+001       = Position of the flash effect from the
                                  camera viewpoint [m]

[Settings]
ClassName=S C O U T             = Drive class name
Capacitors=5                    = Capacitors available
CellCapacity=6.000000e+003      = Capacity of one cell [kg]
CellReloadTime=1.000000e+001    = Loading time of one cell [s]
JumpRange=7.500000e+008         = Jump range of one cell [m]
JumpRangeScaling=1.000000e+001 = Range scaling factor

[Features]
FleetRange=1.000000e+005        = Range of fleet jump command [m]
```

8. LIMITATIONS

If you encounter any problems, please feel free to post them on the Orbiter forums or email it to me (face@snoopie.at). At this point I know of two problems, that I have not investigated far enough to provide a solution for:

- MFD sounds don't play if you didn't start the scenario with the MFD activated in any of the standard MFDs of the focused vessel. Take a look at the showcase scenario for the appropriate lines to copy.
- Sounds don't play if you missed the event of state change. I.e., if you have the focus on one vessel while another one is jumping and you switch focus to the jumping one, you might miss the point where state switches from hyperspace transition to deceleration. You won't hear the deceleration sound in this case, although the visual effects show up.

9. CHECKLISTS

This chapter will give you step-by-step instructions to operate the showcase scenario to get an idea of the module's features.

External viewpoint:

1. Make sure to install the package properly.
2. Start Orbiter and activate the JumpDriveMFD in the *Modules* tab of the launchpad.
3. Start the *JumpDriveShow* scenario located in the root of the scenario folder.
4. You're in the ISS, therefore clicking on JMP or CHG will give you the no-no-sound. Use F3 to switch to Jumper1 (Atlantis near ISS).
5. Click CHG to charge up the first capacitor cell. You should here a pumping sound, if you have OrbiterSound3 installed. Switch to external view and rotate it to get the earth into view.
6. As soon as the pumping sound stops, hit Left-Shift-J to engage the jump. Note the FOV transition and acceleration…
7. After the big bang, rotate the external view to look into the direction the nose is pointing. Switch back to internal view and click TGT.
8. Enter "earth" and press ENTER. Click the <> button to set the current distance to the target. Press the MOD button until the mode name reads "Course deviation".
9. Pitch up until you have a sufficient rotation speed. Watch the mode reading. As soon as it goes to around 10°, consider pressing KILLROT to stop your rotation. Try to get the mode reading as low as possible.
10. Press the CHG button and switch to external view again. As soon as the sound stops (i.e., the cell is filled), hit Left-Shift-J to jump.

Cockpit viewpoint:

11. Use F3 to switch to Jumper2. Use CHG to charge up the cell.
12. Switch to internal view and use F8 to change to the virtual cockpit.
13. As soon as the cell is filled, press JMP to engage the jump.

Observer viewpoint:

14. Use F3 to switch to Jumper3. Use CHG to charge up the cell. Switch to external view and rotate it to see the front of the vessel.
15. Use F3 again to open the selection window, but don't select one or close it.
16. Press JMP to engage the jump. Use the selection window to switch to Jumper2 immediately.
17. Watch Jumper3 coming out of hyperspace.

10. FAQ

This chapter gives answers to frequently asked questions.

Q: I can't find the JumpDriveMFD.ini, where is it?
A: The INI-file is not deployed with the distribution. It will be generated from scratch by the module on shutdown, so just activate the module in Orbiter, close it again and it should be in /modules/plugin/.