Kerbal Space Program - Bug #1043

Time acceleration causes inaccuracy in orbit recalculation on encounter/escape

07/18/2013 01:52 AM - voneiden

Status:	Closed	**Start date:**	07/18/2013
Severity:	Low	**% Done:**	100%
Assignee:			
Category:	Physics		
Target version:			
Version:	0.20.2	**Language:**	English (US)
Platform:	Linux, OSX, Windows	**Mod Related:**	No
Expansion:			

Description

The issue has existed in all versions of KSP that have implemented sphere of influence.

What happens?

When a ship changes the orbiting body of reference (encounter/escape), the orbital elements are recalculated. The exact time of encounter/escape is known by the simulation, however instead of utilizing the known time the physics engine uses the current simulation time when doing this recalculation. This means that the encounter/escape calculation is always done "later" than the actual encounter/escape happened. When under fast time warp the recalculation can happen thousands of kilometers too late which manifests into considerable error in the trajectory calculation. The faster the relative speeds, the greater the error. Most serious impacts currently are in interplanetary travel, however the error is already considerable in Kerbin-Mun travel at warps over 10000x.

How to reproduce?

1) Have an encounter with a planet/moon coming up in a few simulation hours. Note down the periapsis. Quicksave.
2) Warp near the encounter, cross the encounter point with slow time warp (max 10x). Note down the periapsis, it should closely match the above.
3) Quickload. Warp over the encounter with fast time warp (10000x or more for most visible effects). Note down periapsis if you didn't crash the planet/moon.
4) Compare periapses.

Suggested fix (rather simple)

Since the simulation already knows the exact moment when the encounter/escape should happen, save it for later referencing. When recalculating the orbit for the new parent body, use the saved time instead of current simulation time.

Temporary fix

Kerbal Alarm Clock (http://kerbalspaceport.com/kerbal-alarm-clock-2/) can stop time acceleration when approaching encounter/escape.

History

#1 - 10/18/2013 05:19 PM - voneiden

Bug still relevant at 0.22, discussed here

http://forum.kerbalspaceprogram.com/threads/53667-SOI-transitions-introduce-trajectory-errors-depending-on-warp-factor

#2 - 10/24/2013 10:00 AM - willglynn

Please address this. It doesn't seem difficult to prevent the simulation from streaking through SOI boundaries at high warp and breaking carefully-crafted intercept trajectories. Veteran players drop out of warp for SOI changes out of necessity -- Scott Manley noted this just yesterday and then screwed it up -- a survival tactic new players quickly learn once they've been harmed. Fixing this issue will improve every single player's experience in every single mission beyond low Kerbin orbit.

#3 - 10/24/2014 12:50 PM - willglynn

Hello! It's been a year since the last post here. This issue is still affecting players, and I'm still interesting in seeing it fixed.

My understanding is as follows. On-rails objects have their position and velocities updated once per tick, each time recalculated from their orbital elements to avoid compounding errors. Objects are checked for sphere-of-influence transitions once per tick. Any object that's crossed into a new SoI has its position and velocity are converted from the previous reference frame to the new refence frame, and if that object was on rails, it's placed back on rails by calculating new orbital elements based on its now-converted position and velocity.

This strategy is sound, but has significant precision errors when paired with time acceleration, since there's nothing to ensure that ticks will occur anywhere near the actual SoI boundary. The further you are from the SoI boundary, the more gravity you've "missed", and the more likely your post-transition trajectory will create trouble. This is especially problematic at high warp and high relative velocities â both common with interplanetary transfers.

I proposed some solutions last year, but they were lost in the Great Forum Fire of 2013, so I'll outline them again here.

Things with SoIs have orbits and on-rails objects have orbits. There's math involved, but it's possible to calculate when one would hit the other, separate from the current physics engine. Assuming you know when an SoI transition is supposed to occur, there are some options:

1. Calculate on-rails SoI transitions retrospectively. Do everything the way it is now, except when transitioning from one SoI to another, handle the on-rails case differently. Normal physics objects are governed by position and velocity, but on-rails objects are governed by orbital elements. This means the SoI transition can be calculated as of the moment of intercept *regardless of the current game clock*. The transition can still calculate orbit around A -> state vector in reference frame A -> state vector in reference frame B -> orbit around B, but again, it can be calculated at a more appropriate instant (i.e. midway through the previous tick when the vessel is at the terminator), rather than on a tick boundary (once you're well across the SoI terminator).

2. Prevent time warp from warping way across SoI boundaries. Each tick, compare the timestamp of the "next" tick against the next SoI transition, and clamp its duration appropriately. This way, 10000x time acceleration would advance 10000 time units per tick normally, but on an SoI transition it might only advance 3530 time units. This allows the current SoI transition code to transition near the edge of the SoI (where it's accurate enough) instead of deep into an SoI (where it screws over players).

The difficulty in both approaches is determining when the SoI transition would occur, but good news â **map mode does this already**. Hitting M shows this value **today**. Why can't the physics engine use that knowledge to do something smart?

#4 - 10/24/2014 01:10 PM - voneiden

Thanks for commenting. Yes, the simulation already calculates the time when a SoI change is going to happen (this can be confirmed via the API), and I also imagine it is using this time to trigger the function that handles the actual SoI change. The bug here is simply that this **SoI change function uses the current simulation time** instead of the precalculated SoI change time. With big time warp speeds the simulation time can greatly exceed the actual SoI change time before the SoI change function is called.

So really, this bug could probably be solved by just making the SoI change function use the precalculated SoI change time for the coordinate system conversion instead of current time.

I think the main issue here is bringing this bug to the attention of the devs. It should be really easy to fix, while it's a major simulation inaccuracy that IMO should be high priority to be fixed.

And as a bonus feature request, the time warp should be fixed also to avoid things like skipping through atmosphere and planets. Time and altitude of periapsis is also calculated and this should be used as an indicator of when to slow down or stop the time warp before skipping happens.

#5 - 10/24/2014 01:21 PM - voneiden

Btw, Trigger Au has a great plugin called "Kerbal Alarm Clock" that I've used as a workaround whenever I'm planning.

#6 - 01/21/2015 01:24 PM - willglynn

The latest devnote includes:

> Other than that, I was able to get a feature I had to leave out a long time ago up to a nearly complete state now. I call it âTimeWarp-Toâ. Basically, it lets you select a point ahead of you in your trajectory, and have the game auto-warp up to that point as fast as is reasonable (given the time gap).
>
> This feature was delayed because of the time needed to work out how to deal with the warp limits near planetary surfaces, to stop warping just before any maneuvers you may have set, and also because I wanted to add a time warp limit when approaching an SOI transition. The autowarp system will respect those limits, and step up warp again as soon as conditions allow.

I appreciate that Squad recognizes that SOI transitions are bugged. On the other hand, limiting warp doesn't seem like a great solution to me; Kerbal Alarm Clock does that because it **can't** do any better, not because that's ideal.

On-rails objects permit their state vectors to be calculated as of any instant, and the game can calculate when the SOI transition will/did happen. For on-rails objects (i.e. vessels operating at warp), KSP could calculate the SOI transition as of the precise moment that it occurred, independent of the simulation clock. This remains my preferred solution. Is this being considered, or is the SOI-sensitive warp limit as close as we're going to get to a fix?

#7 - 03/26/2015 08:28 AM - willglynn

From the latest devnote:

> I have been able to do one thing which Iâve been really wanting to do for ages now: You may have noticed when vessels cross spheres of

influence at high time warp, that the resulting orbit after the transition can be quite different from what was originally estimated. This is caused not by an imprecision with the estimation itself, nor is this a floating-point inaccuracy issue for once; the problem is with the way the actual orbit component in the vessel behaves when switching reference frames. That bit of code was written much longer ago than the patched conics system, and it was less accurate than the maths used to estimate your upcoming trajectories.

The inaccuracy came from the fact that, while the patch conics estimation is very precise in pinpointing the exact moment the vessel crosses SOIâ□□s, the actual orbit code would simply react to the dominant body having changed between the last frame and the current one, and recalculate the new orbit based on the current position and velocity vectors. That is okay as long as you are not running the game at a high warp factor, but at very high time compressions, a vessel may move a lot in one frame. Quite enough to fly far past the true point of SOI transition, which means the state vectors it used to calculate its new orbit could be off by as much as an entire frameâ□□s worth of movement.

The solution was simple enough: Instead of simply recalculating after a change in sphere of influence, the vessel orbit code now searches backwards between its position in the last frame and now, for the exact point where the it crossed over to the new SOI. This search finds the moment of SOI transition down to 1/100th of a second. We then calculate the new orbit from the state vectors at that point in time, which gives us much more accurate results, and is unaffected by the warp factor.

Thank you!

I believe this issue can be closed.

#8 - 03/26/2015 02:11 PM - voneiden

- Status changed from New to Confirmed

- % Done changed from 0 to 10

Thanks for the heads up! And thanks to HarvesteR for fixing it. I don't know if they ever saw this bug report though. Heh heh.

I've changed the status as confirmed. That's the closest to being fixed I can select. If a mod reads this, feel free to close!

#9 - 04/19/2015 03:57 PM - andzkerman

Hi there

Despite this problem apparently having been more-or-less fixed, I am still having a time-warp/SoI transition related problem in v0.90.

I have put a probe into orbit around Jool, but its orbit does cross the orbits of some of Jool's moons. I have run time a long way forward as a test in up to 10,000x time-warp with no problem. Any interactions only cause minor changes to my orbit. However, in 100,000x the first encounter with any moon often causes my probe to be ejected from the Jool system at very high speeds.

I do use Kerbal alarm clock, but in this case it isn't much use, as I am attempting to send a mission to Duna, which involves using high time-warps whilst focused on the new mission, leaving my Jool probe to do its own thing in orbit.

I would prefer not to have to use cheats to replenish the fuel in my Jool probe and send it into a different orbit. If there's anything more can be done to fix this problem I would be very grateful. :)

Thanks

#10 - 04/19/2015 08:18 PM - triffid_hunter

andzkerman

The fix will be in the next release.

Harvester described the fix in a blog post about new features in the upcoming 1.0.

#11 - 07/17/2016 09:24 AM - TriggerAu

- Status changed from Confirmed to Needs Clarification

- % Done changed from 10 to 0

#12 - 07/17/2016 08:36 PM - Claw

- Status changed from Needs Clarification to Closed

- % Done changed from 0 to 100

- Platform Linux, OSX, Windows added

- Platform deleted (Any)

The game now slows warp to 50x when crossing boundaries, and rails calculations are used to help stabilize non-rails orbital parameters.

Files

1.png	252 KB	07/18/2013	voneiden
2.png	233 KB	07/18/2013	voneiden
3.png	203 KB	07/18/2013	voneiden