BLOODMOON

SWITCHING TO THE uBODE PHYSICS ENGINE

Understanding the benefits, changes and solutions for older mesh objects

OpenSimulator 0.9.3.1

GUIDE FOR RESIDENTS AND REGION OWNERS

CPU	RAM	uBODE
MAIN BENEFIT	**ADDITIONAL BENEFIT**	**DEFAULT ENGINE**
Significant reduction observed on several simulators	Reduction observed on several regions	Current choice in the OpenSimulator configuration

BloodMoon Document • July 2026

About this change

As part of BloodMoon's upgrade to OpenSimulator 0.9.3.1, the BulletSim physics engine has been replaced by uBODE on the regions hosted by the grid. This change has brought very noticeable improvements to our infrastructure, particularly a significant reduction in processor usage.

KEY POINT

Switching to uBODE does not damage objects. However, it may reveal an old, imprecise or overly complex physics shape that BulletSim previously tolerated more easily.

Contents

1. Understanding the role of the physics engine

In OpenSimulator, the physics engine manages contacts, movement and collisions within a region. It does not determine how objects look: it calculates how avatars and physical objects interact with them.

- prevent an avatar from walking through a wall;
- allow avatars to walk on a floor;
- manage falls and obstacles;
- move physical objects;
- manage certain vehicles;
- calculate collisions between avatars and objects;
- allow certain scripts to detect contact;

- manage stairs, roads, bridges and buildings.

Appearance and physics are two different things

A mesh object can contain two separate representations: the visible model displayed in the viewer, and an invisible physics shape used by the simulator.

VISIBLE MODEL	PHYSICS SHAPE
• walls and windows	• floors avatars can walk on
• surface details and decorations	• walls that must block movement
• handles, tiles and mouldings	• open doors and passages
• furniture details	• surfaces that create collisions

ESSENTIAL PRINCIPLE

The physics shape should generally be much simpler than the visible model. A highly detailed table does not need a collision shape reproducing every moulding.

2. Why BloodMoon chose uBODE

uBODE is now the default physics engine

The current OpenSimulator configuration uses uBODE as its default physics engine, together with a mesh calculation system designed to work with it. This choice keeps BloodMoon aligned with recent developments in the software and better prepared for future updates.

A significant reduction in CPU usage

This is the main benefit observed since the change. With equivalent content, several BloodMoon simulators now place far less demand on the processor than they did with BulletSim.

- reduce the overall server load;
- keep more resources available;
- limit certain spikes in processor usage;
- distribute resources more effectively between regions;
- prevent a heavily loaded region from affecting other simulators;
- make it easier to host a large number of regions.

BLOODMOON OBSERVATION

The reduction in CPU usage is the most visible benefit. Lower memory usage has also been observed, although it is generally less pronounced.

Better controlled memory usage

Several regions are also using less memory since the switch to uBODE. This improvement helps preserve a safety margin for events, avatars and scripts.

More rigorous collision management

uBODE interprets physics shapes differently and sometimes more strictly than BulletSim. For correctly designed objects, collisions can be more consistent. With some older meshes, this stricter handling may reveal a defect that was already present in the physics shape.

3. Why some older meshes can cause problems

Many mesh objects were created several years ago, sometimes using import methods that differ from those used today. Some were tested mainly with BulletSim.

- without a genuine separate physics model;
- with the visible model used directly as the physics shape;
- with an excessively detailed collision shape;
- with a hull surrounding the entire object;
- with doors and windows missing from the physics model;
- with unnecessary or incorrectly positioned triangles;
- with a physics shape offset from the visible model.

BulletSim could sometimes tolerate or interpret these objects in a way that made them usable. uBODE may interpret them differently, revealing invisible collisions, blocked passages or offset floors.

IMPORTANT

An older mesh does not contain “BulletSim physics”. It contains a physics shape that may be interpreted differently depending on the engine being used.

4. Recognising the most common problems

PROBLEM	**A door or opening is blocked** The object often uses a convex hull that closes the opening as though the building were a solid volume.
WHAT TO DO	Test the “Prim” setting on the affected part. Make sure only the correct linked part is selected. As a last resort, disable the faulty collision and recreate the walls with invisible prims.

PROBLEM	**The avatar walks above the floor** The physics shape is positioned above the visible model, creating the impression that the avatar is floating.
WHAT TO DO	Test the “Prim” setting. Check the floor part within the linked object. Add a flat invisible prim at the correct height if the built-in collision remains incorrect.

PROBLEM	**The avatar falls through the floor** The physics shape may be missing, disabled or incorrectly positioned.
WHAT TO DO	Check that the physics shape type is not set to “None”. Test the “Prim” setting. Create a simple invisible surface beneath the visible floor.

PROBLEM	**Invisible walls or obstacles appear** The collision may extend beyond the visible model or cover an area that should remain open.
WHAT TO DO	Select the linked parts one at a time. Disable collision on purely decorative elements. Replace the complex collision with a few simple prims.

PROBLEM	**The stairs become difficult to use** Each step may be interpreted as a separate obstacle, especially when the physics model is very detailed.
WHAT TO DO	Keep the stairs for their visual appearance. Disable the faulty collision if necessary. Add an invisible sloping ramp above the steps.

PROBLEM	**A vehicle becomes unstable** The vehicle depends on its shape, mass, scripts and the physics engine. Some settings may have been designed specifically for BulletSim.
WHAT TO DO	Rez a fresh copy. Reset the scripts. Test at low speed in an open area. Contact the creator if the script needs to be adapted.

5. Correcting physics shapes in Firestorm

First precaution: always keep a copy

Make sure the object can be copied.
Take a copy into your inventory.
Write down its position and rotation.
Carry out tests on a copy whenever possible.
Do not delete the original object until the correction has been confirmed.

Checking the physics shape type

Right-click the object.
Select “Edit”.
Open the “Features” tab.
Find the “Physics Shape Type” setting.

CONVEX HULL	A simplified shape suitable for small solid objects. It may close doors, arches, tunnels or hollow buildings.
PRIM	Uses the physics shape supplied when the mesh was imported. It may restore doors and passages if that shape was created correctly.
NONE	Disables collision. Suitable for plants, curtains, pictures and decorations that avatars may walk through.

	CAUTION Do not automatically set every mesh to “Prim”. If the built-in shape is very complex or poorly designed, this may cause other problems and increase simulator load.

Modify only the affected part

A mesh building is often made of several linked parts: floor, walls, roof, windows, stairs and decorations. It is not always necessary to modify the entire object.

Enable “Edit Linked Parts”.
Click only the part that is causing the problem.
Check its physics shape type.
Modify only that part.
Test the result before continuing.

6. Practical solutions for floors, stairs and buildings

Replacing a poor physics shape with invisible prims

When a mesh has an unusable physics shape, the most reliable solution is to separate its visible appearance from its collision. The mesh remains visible, its faulty collision is disabled, and simple prims are used to represent the physical surfaces.

- a flattened prim for the floor;
- a few prims for the walls;
- an invisible ramp for the stairs;
- clear spaces left around doorways.

BUILDING TIP

Keep the prims visible temporarily while positioning them. Make them transparent only after confirming their placement and all passageways.

Correcting stairs with an invisible ramp

Keep the mesh stairs for their visual appearance.
Disable their collision if it causes avatars to become stuck.
Add a sloping prim above the steps.
Adjust its angle and height.
Make the prim transparent after testing.

The avatar then walks up a smooth surface even though the steps remain visible. This method improves movement and reduces the risk of becoming stuck.

Correcting a floor or road

If the avatar floats above the ground:

- check the physics shape type;
- test the “Prim” setting;
- check whether the mesh is made of several linked parts;
- disable the faulty collision if the problem continues;
- add a flat invisible prim at the correct height.

If the avatar falls through the floor:

- check that the physics shape is not set to “None”;
- test the “Prim” setting;
- make sure the selected linked part is actually the floor;
- add a simple invisible surface if the built-in physics does not work.

Reducing the physics load of decorative objects

Purely decorative objects do not always need to be included in physics calculations. When you have permission to modify them, the “None” setting may be suitable for:

- plants, grass and flowers;
- curtains, pictures and wall decorations;
- small objects placed on furniture;
- elements that are out of reach;
- roof details or purely visual parts.

PERFORMANCE IMPACT

Removing unnecessary collisions reduces the number of calculations performed by the physics engine and can help limit the region’s CPU load.

7. Vehicles and physical objects

Vehicles depend on the physics engine, their shape, their mass and their scripts. A vehicle designed for BulletSim may therefore require adjustment under uBODE.

Take a copy of the vehicle.
Rez a fresh copy in an open area.
Reset its scripts.
Check that only the necessary parts are physical.
Check that decorative parts do not have unnecessary collisions.
Test the vehicle at low speed.
Compare its behaviour with what you observed previously.

POSSIBLE LIMITATION

If the vehicle remains unstable, its script may contain parameters tuned specifically for BulletSim. The BloodMoon team may not be able to correct a non-modifiable or protected vehicle.

8. Best practices and mistakes to avoid

What you should not do

- modify an object without keeping a copy;
- set every mesh in the region to “Prim” without testing it;
- set every object to “None”;
- unlink a complex building without a backup;
- delete an object before finding a solution;
- make objects physical when they do not need to be;
- use hundreds of small prims to replace a simple floor;
- modify a vehicle without preserving its original settings;
- re-import a mesh without owning the source files or the necessary rights;

- automatically assume that every collision problem is a simulator fault.

Do all older objects need to be replaced?

No. Most correctly made objects continue to work normally with uBODE. Action is only required when an actual problem is observed.

- change the physics shape type;
- modify only the affected linked part;
- disable the collision of a decorative element;
- add an invisible floor or ramp;
- replace a poor collision shape with a few simple prims;
- update an older vehicle.

9. How to report a problem

If you notice a problem following the switch to uBODE, please provide the BloodMoon team with as much information as possible:

- the name of the region;
- the exact coordinates;
- the name of the object;
- the name of its owner;
- a precise description of the problem;
- a screenshot, if possible;
- details of your permissions on the object;
- whether the object is modifiable or non-modifiable.

Examples of useful descriptions

EXAMPLE

“The main door is open, but an invisible collision prevents us from entering.”

EXAMPLE

“The avatar walks about fifty centimetres above the road.”

EXAMPLE

“The floor is visible, but we fall through it.”

EXAMPLE

“There is an invisible obstacle in front of the stairs.”

EXAMPLE

“The vehicle worked before, but now it immediately tips over.”

What the team can check

- the physics shape type;
- whether a convex hull is being used;
- a faulty collision shape built into the mesh;
- a particular linked part;
- a vehicle script;
- the object’s modification permissions;
- a possible genuine simulator malfunction.

10. A positive development for BloodMoon

The switch from BulletSim to uBODE represents an important development for BloodMoon. The initial results are very encouraging, with a very noticeable reduction in the CPU load of many simulators.

A reduction in memory usage has also been observed, although the greatest improvement remains the lower processor load. This makes more resources available for avatars, scripts, events and all hosted regions.

This transition may require a few adjustments to certain older meshes, but these difficulties mainly affect objects whose physics shape was already imprecise or unsuitable.

IN THE LONG TERM, uBODE SHOULD ENABLE BLOODMOON TO PROVIDE REGIONS THAT ARE

LESS DEMANDING ON THE PROCESSOR
BETTER SUITED TO CURRENT OPENSIMULATOR VERSIONS
EASIER TO MAINTAIN AND MORE STABLE
BETTER PREPARED FOR FUTURE GRID DEVELOPMENTS

Thank you to all residents, region owners and creators for your understanding and for helping us improve BloodMoon.