GO Buoyancy Toolkit Documentation

Written for version 2.0

The Buoyancy Toolkit is a code package for the Unity3D game engine that brings realistic buoyancy simulation to your project. Rigidbodies will float when submerged in fluid volumes of different densities and wave functions such as water pools or oceans with large waves. The shape and volume of one or more connected colliders will be taken into account to let the geometry follow the waves realistically. The toolkit is ideal for making anything float in water but can also be used to simulate the lift a balloon or airship experiences in the atmosphere. The toolkit is easy to use and does not require any scripting.

Contents

1 Features

- Realistic buoyancy simulation
- Surface waves if a wave function is specified (using a custom script, example

provided)
- Weighting for straightforward tweaking of the buoyancy force
- Disable weighting and use realistic object properties (scale, mass, fluid density) to mimic real-life
- Works well with other rigidbody forces

2 Set up guide

1. Import the Buoyancy Toolit package into your project
2. Open up the scene you are working on
3. For the water game object, add a box collider and mark it as a trigger. Make sure it covers the water volume.
4. Add the FluidVolume component (located in Core/) to the water game object.
5. Add the BuoyancyForce component (located in Core/) to a game object that has a rigidbody component.
6. Drag a collider into the "Buoyancy Collider" slot of the BuoyancyForce component. Use the collider that is attached to the game object in question or, for compound setups, that is attached to any of its children.

3 Workflow

One water game object is added for each water area in a scene. Each water game object should have a trigger box collider and a FluidVolume component attached.

A BuoyancyForce component is attached to each game object that should be able to float in the water. Tweak the "Weight Factor" value for different behaviors and use the the "Quality" setting to trade performance for simulation quality. Weighting makes it easy to tweak the buoyancy forces applied to the rigidbody. If no weighting is used, the toolkit assumes that all object properties are set to realistic values and the buoyancy forces depend on the scale and mass of the submerged game object in addition to the density of the fluid it is submerged in. Note that weighting does not simplify the simulation, it just makes it easer to achieve the desired effect.

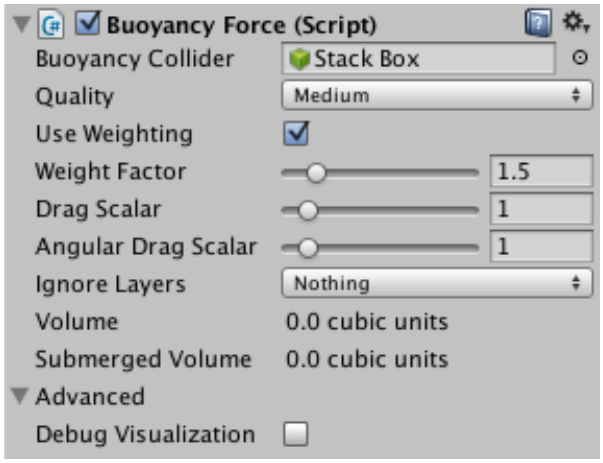
During runtime, specific parameter values can be changed using the scripting interface in order for the simulation to react to events that occur in the game. For example, the "Weight Factor" of a BuoyancyForce attached to a boat game object can be changed to a value slightly below 1 to simulate sinking.

4 Components

4.1 BuoyancyForce component

The BuoyancyForce component makes a rigidbody float when entering fluid volumes. It acts on the Rigidbody of the game object to which it is attached, using a specified collider

for buoyancy calculations. Make sure that the specified collider is attached to the same game object or any of its children.



Inspector view of the BuoyancyForce component

Buoyancy Collider

The collider that will be used to calculate the buoyancy properties of the rigidbody. The collider should be convex for stability reasons.

Quality

The quality of the simulation. High values trade performance for simulation quality and vice versa.

Samples [Only visible when Quality is set to Custom]

The number of sample points used per axis for the buoyancy simulation. High values trade performance for simulation quality and vice versa.

Use Weighting

A toggle indicating whether this BuoyancyForce uses weighting. Weighting enables easy tweaking of the buoyancy behaviour. If weighting is not used, realistic proportions, rigidbody masses and fluid densities are required for realistic behaviour.

Weight Factor

A value indicating the strength of the buoyancy force when weighting is enabled. A weight factor of 1 results in enough force to counteract gravity and the rigidbody will stay in equilibrium within the fluid. A weight factor of 2 results in a net force equal to gravity but in the opposite direction (making the rigidbody float in the fluid) and so on.

Drag Scalar

A scalar that is multiplied by the fluid volume's drag value before being set as linear drag to the submerged rigidbody.

Angular Drag Scalar

A scalar that is multiplied by the fluid volume's angular drag value before being set as angular drag to the submerged rigidbody.

Ignore Layers

A bitmask indicating which FluidVolume layers should be ignored by this BuoyancyForce.

Volume

The approximate volume of the buoyancy collider.

Submerged Volume

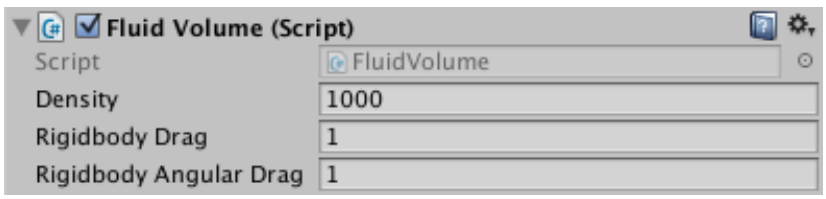
The approximate volume of the buoyancy collider that is submerged in a fluid volume.

Advanced → Debug Visualization

A toggle indicating whether or not debug visualizations are rendered.

4.2 FluidVolume component

The FluidVolume component uses a trigger box collider to define a volume in space in which game objects may float. Custom wave functions can be realized by creating a script that derives from the FluidVolume class and overrides the GetHeightAt function. (See section 5.2 for more information)



Inspector view of the FluidVolume component

Density [kilograms per cubic unit]

The density of the fluid. Only used when weighting is disabled.

Rigidbody Drag

The linear drag that is applied to a rigidbody submerged in the fluid volume.

Rigidbody Angular Drag

The angular drag that is applied to a rigidbody submerged in the fluid volume.

5 Scripting reference

5.1 BuoyancyForce component

In order to change the behavior or get information from the BuoyancyForce during runtime, you can write a custom script that changes the properties or calls functions of the BuoyancyForce component.

To get a reference to the BuoyancyForce component, use the following code and make sure that the script is attached to the same game object.

```
public void Start()
{
    BuoyancyToolkit.BuoyancyForce f = GetComponent< BuoyancyToolkit.BuoyancyForce >();

    // Now use f
    f.WeightFactor = ...;
}
```

If you want to change the drag of the rigidbody during gameplay, set the NonfluidDrag property of the attached BuoyancyForce instead. This is required because the BuoyancyForce component often changes the drag value and needs to keep track of the base line. If there are multiple BuoyancyForce components attached to the same game object, set the same NonfluidDrag value to all them. (This caveat also applies to the angularDrag of the rigidbody)

Here is a list of the properties and functions of the BuoyancyForce component.

Name	(Return) Type	Property/Function	Note
BuoyancyCollider	Collider	Property	See section 4.1
Quality	BuoyancyQuality	Property	See section 4.1

Samples	int	Property	See section 4.1
UseWeighting	bool	Property	See section 4.1
WeightFactor	float	Property	See section 4.1
DragScalar	float	Property	See section 4.1
AngularDragScalar	float	Property	See section 4.1
NonfluidDrag	float	Property	The base linear drag that should be applied to the rigidbody. Set to the drag value of the connected rigidbody at the start of the scene.
NonfluidAngularDrag	float	Property	The base angular drag that should be applied to the rigidbody. Set to the angularDrag value of the connected rigidbody at the start of the scene.
IgnoreLayers	LayerMask	Property	See section 4.1
			The fluid volume that

FluidVolume	FluidVolume	Property	currently affects this buoyancy force.
IsSubmerged	bool	Property	See section 4.1
IsCompletelySubmerged	bool	Property	See section 4.1
Volume	float	Property	See section 4.1
SubmergedVolume	float	Property	See section 4.1
DebugVisualization	bool	Property	See section 4.1

5.2 FluidVolume component

In order to change the behavior or get information from the FluidVolume during runtime, you can write a custom script that changes the properties or calls functions of the FluidVolume component.

To get a reference to the FluidVolume component, use the following code and make sure that the script is attached to the same game object.

```
public void Start()
{
    BuoyancyToolkit.FluidVolume v = GetComponent< BuoyancyToolkit.FluidVolume >();

    // Now use v
    v.density = ...;
}
```

In order to change the surface of the fluid volume, create a new class in a custom script that derives from the FluidVolume class in the Buoyancy Toolkit. Override the GetHeightAt(Vector3 p) function and implement the desired wave function. The GetHeightAt function takes a world space point and should return the world height of the surface it this location. Please see the CustomFluidVolume script (located in Examples/Shared Assets/Scripts) for an example on how to do this.

Here is a list of the properties and functions of the FluidVolume component.

Name	(Return) Type	Property/Function	Note

density	float	public variable	See section 4.2
rigidbodyDrag	float	public variable	See section 4.1
rigidbodyAngularDrag	float	public variable	See section 4.1
ProjectPointOntoSurface(Vector3 p)	Vector3	Function	Projects a point onto the surface of the fluid volume.
GetHeightAt(Vector3 p)	float	Function	Calculates the height of the fluid at a given location. Override this method in order to customize the wave function.

6 Example scenes

6.0.1 Basic

A simple scene showing a number of objects floating in water.

6.0.2 Compound

A scene demonstrating how to set up the buoyancy force on a more complex game object with multiple colliders. Please see the "Catamaran" game object and its game object hierarchy.

6.0.3 Waves

A scene demonstrating how to set up a fluid volume with a custom wave function. The "Water" game object has a CustomFluidVolume script (located in Examples/Shared Assets/Scripts) attached. Please see section 5.2 for more information.

7 Helpers

7.1 SetCenterOfMass component

The location of the center of mass is important for the stability of a game object submerged in a fluid volume. A boat will be stable in the water if the center of mass is below the center of buoyancy, which is calculated by the toolkit. Basically, if a boat or platform always topples over in the water when it is supposed to lay flat, try to lowering the center of mass of the rigidbody using this script.

8 Known Limitations

None at the time of this writing.

9 Contact

Please let me know if you run into any problems when using the toolkit or if you have feedback on how I can improve it in the future. I am also interested in seeing projects that use any of my toolkits in practice as it motivates me to work harder!

Website: http://gustavolsson.com/

Contact: http://gustavolsson.com/contact/