User manual for add-on “Interpolation”

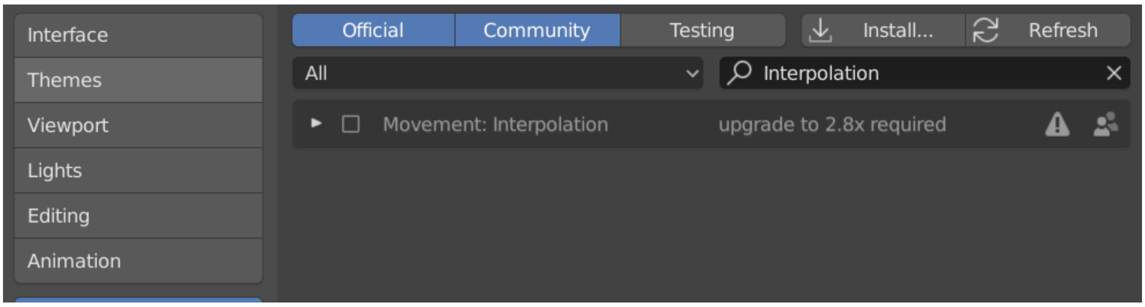
Authors:
Marc Guerrero Palanca
Tatsiana Palikarpava
Alberto Pérez Abad

Contents

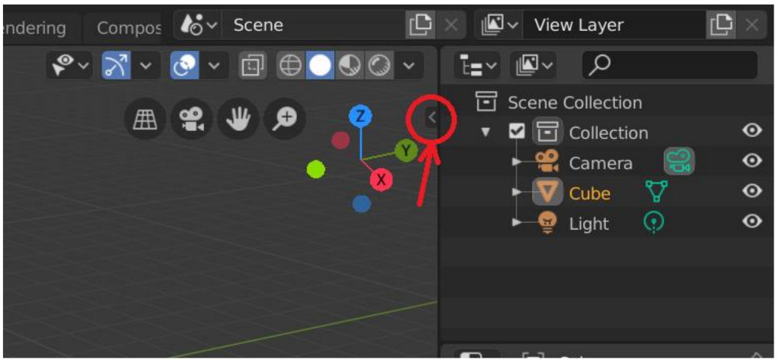
In the following manual we describe designed add-on, used for working with animated objects and modifying their trajectories according to user settings.

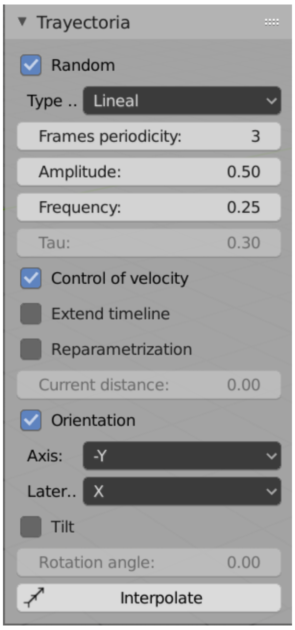
1. Installation & Use

- Our add-on is located in archive **main.zip.** To install it, it is necessary to choose it in user preferences for installation. Previously, you need to save the file.



- After installation, you will find our panel by clicking the arrow marked on the image or pressing “N”. You will see a tab “Movimiento”, which is actually what we will be using.




▾ Trayectoria
☑ Random
Type .. Lineal
Frames periodicity: 3
Amplitude: 0.50
Frequency: 0.25
Tau: 0.30
☑ Control of velocity
□ Extend timeline
□ Reparametrization
Current distance: 0.00
☑ Orientation
Axis: -Y
Later.. X
□ Tilt
Rotation angle: 0.00
Interpolate

- Next, you need to create some objects and define action for them by inserting keyframes.
- In the window you see the following properties:

Check-box **Random**	If not selected, movement will be produced accurately according to other properties such as type of interpolation, frequency, etc. If selected, for each keyframe coordinates of the object will be changed by defining some deviation from normal trajectory.
Dropdown list **Type of interpolation**	User can choose between 3 types of interpolation: *Linear, Hermite* and *Catmull-Rom.*
Integer property **Frames periodicity**	User defines the periodicity of inserting new keyframes.
Float property **Amplitude**	User defines maximal distance of object's deviation from normal trajectory. If **Random** option is not selected, property **Amplitude** is blocked.
Float property **Frequency**	User defines frequency of random movement. If **Random** option is not selected, property **Frequency** is blocked.
Float property **Tau**	User defines value of parameter τ, used for *Catmull-Rom* interpolation. If not *Catmull-Rom* option is chosen, property **Tau** is blocked.
Check-box **Control of velocity**	If selected, enables the possibility of control of velocity by user. If this option is not selected, properties **Extend timeline, Reparametrization** and **Current distance** are blocked.
Check-box **Extend timeline**	If selected, the object will run the trajectory with constant velocity 1 un./s, therefore duration of movement changes. **Reparametrization** is blocked. If not selected: • **Reparametrization** is not selected Duration of movement stays unchanged, object runs with constant velocity (depends on the length of trajectory and timeline length). • **Reparametrization** is selected Duration of movement stays unchanged, object runs with user-defined velocity.
Check-box **Reparametrization**	If selected, option **Extend timeline** is blocked. User can operate with the property **Current distance** by means of graph editor and inserting keyframes.
Float property **Current distance**	Shows covered distance from the beginning of the trajectory for certain

	frame. Includes the possibility to insert keyframes and work with graph editor.
Check-box **Orientation**	If selected, enables the possibility of user-defined rotation control. If this option is not selected, properties **Axis, Lateral axis, Tilt** and **Rotation angle** are blocked.
Dropdown list **Axis**	User can choose between 6 options: *X, Y, Z, -X, -Y, -Z.* This property defines which axis will be aligned with the direction of movement.
Dropdown list **Lateral axis**	User can choose between 6 options: *X, Y, Z, -X, -Y, -Z.* This property defines which axis will be chosen as lateral to maintain it in horizontal state.
Check-box **Tilt**	If selected, user can operate with the property **Rotation angle** by means of graph editor and inserting keyframes.
Float property **Rotation angle**	Shows the angle of inclination of lateral axis for certain frame. Includes the possibility to insert keyframes and work with graph editor.
Button **Interpolate**	This button performs modifying of the trajectory. It is designed with concern to prevent some errors, occurring because of lack of information about the object. To have this button enabled program checks that each object, to which user is going to apply interpolation, contains animation data and particularly action, with at least 2 keyframes.

- After defining all the required properties press **Interpolate.** For all selected objects trajectories will be modified. You can press “Ctrl+Z” to return to the previous state.

2. Action management

Let us consider our initial problem. We have several metaballs and some trajectory. What we need is to define some kind of movement within this trajectory, so that metaballs form something like a bubble, staying rather close to each other, but not having identical paths.

Taking into account the fact that the amount of balls is not limited, we try to avoid creating numerous trajectories manually. For example, we defined required trajectory for one ball. One possible solution is to copy this object a few times, so that newly created objects will have the same trajectories. In the early stages of developing of this tool, we used a function *CopRuta,* which aim was to copy the trajectory of one object to all objects of the list, which is the parameter of the function. So, we were usually creating one metaball, defining its trajectory and applying CopRuta to the list of other metaballs in the scene.

However, more wise solution would be to work a bit with *Action Editor.*

We have an object Mball with defined trajectory, called MballAction. In *Action Editor* we can press *Stash* to unlink it from the object Mball. Then we copy the trajectory (it is now called

MballAction.001). Now it’s time to create one more metaball Mball.001. Select it and in *Action Editor* choose MballAction, then copy it (now we have MballAction.002). Repeat the same for the third metaball Mball.002 with action MballAction.003.

After these preparations, we can apply our add-on to the balls. For example, we select Mball.001 and Mball.002, then define properties (*Random* = True, *Type* = Lineal, *Frames periodicity* = 3, *Amplitude* = 0.3, *Frequency* = 0.25) and press *Interpolate.* Finally, actions MballAction.002 and MballAction.003 have changed.

Well, let us suppose that we changed our mind and now decided that it is necessary to interpolate initial trajectory for all the three balls with Hermite interpolation. In such case, we select Mball.001 and Mball.002 one by one, choose MballAction (it is still stored), and copy it (now we have MballAction.004, MballAction.005). We don’t need to perform it for Mball, because its trajectory hasn’t been changed. We define properties (*Random* = True, *Type* = Hermite, *Frames periodicity* = 3, *Amplitude* = 0.5, *Frequency* = 0.5) and press *Interpolate.*

Therefore, the main idea is to store all changed trajectories separately and not forget to save the trajectory before modifying. This gives us the following opportunities:

- ✓ We can always return to the previous state.
- ✓ It is possible to create different interpolations for one object using one trajectory as a base. We can also compare the behavior of the object with different trajectories (let them be called *Trajectory1*, *Trajectory2*). For this, we can copy our object and assign it *Trajectory2* in *Action Editor,* while for initial object it will be *Trajectory1.* In that way, user can see simultaneously two trajectories in real time.
- ✓ We can assign one object trajectory of other and then modify it.
- ✓ We can apply sequentially different types of interpolation.

Therefore, our program does not deal with storing the actions automatically, so once you have pressed *Interpolate,* current path is modified. But as we described above, you can store everything you will need manually.

3. Control of velocity

In this chapter we will describe functionality of velocity control in our tool.

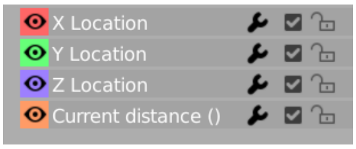
To start working with this scope tick the option **Control of velocity.** After this you will have two possibilities:

- constant velocity
- user-defined velocity

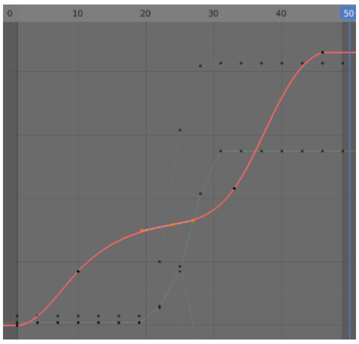
If you want to deal with constant velocity, you do need to decide whether you would like to leave timeline length unchanged or not. If you tick **Extend timeline** check-box, your object will run the trajectory with constant velocity 1 un./s. Keep in mind that it will lead to the change of the length of your timeline. There also exists safer option. If you do not tick **Extend timeline** check-box, your object will run the trajectory with constant velocity, but it will be calculated as length of trajectory divided by duration of movement.

A more flexible way to control velocity is to let the user define it manually as well as the location of the object. To use it, you need to perform the following steps:

1) Tick the option **Reparametrization.** After this, property **Current distance** will become active.
2) Put the cursor on **Current distance** and press **I** or *Insert keyframe.* You will see that a new FCurve "Current distance" was created.


X Location
Y Location
Z Location
Current distance ()

3) Current distance property shows the distance from the beginning of the curve that the object have run by the current frame. You are allowed to insert keyframes for this property. To do it, you set corresponding frame as current, type the value of the property and press **I** (holding mouse on the property). When you have several keyframes, you can also adjust this FCurve in *Graph Editor.*



4) Our tool prevents user from setting inadequate values for **Current distance,** so if you set it to the value less than 0 or greater than the length of trajectory it will be set automatically to 0 or length of the trajectory respectively.

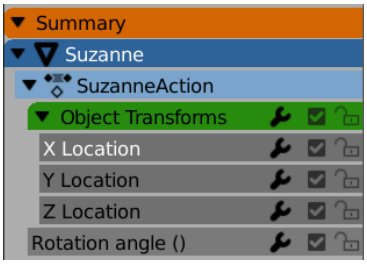
4. Control of rotation

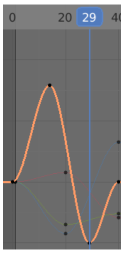
In this chapter we will describe functionality of rotation control in our tool.

To start working with this scope tick the option **Orientation.** After this you will be asked to assign values to 2 properties: **Axis** and **Lateral axis.** The first one is the axis that is used to align with the direction of movement. The second one is the axis that is maintained horizontal. Our tool prevents the user from such mistake as choosing the same axis for both properties. If the user selects one of the following combinations: ([axis], [axis]), ([axis], [-axis]), ([-axis], [axis]), ([-axis], [-axis]), the button **Interpolation** will be blocked.

If you want to add lateral tilt, you need to tick the option **Tilt** and then define the values of the angle of this tilt manually as well as the location of the object. To use it, you need to perform the following steps:

1) Tick the option **Tilt.** After this, property **Rotation angle** will become enabled.
2) Put the cursor on **Rotation angle** and press **I** or *Insert keyframe.* You will see that a new FCurve "Rotation angle" was created.


Summary
Suzanne
SuzanneAction
Object Transforms
X Location
Y Location
Z Location
Rotation angle ()



3) Rotation angle property shows the angle of lateral tilt for current frame. You are allowed to insert keyframes for this property. To do it, you set corresponding frame as current, type the value of the property and press **I** (holding mouse on

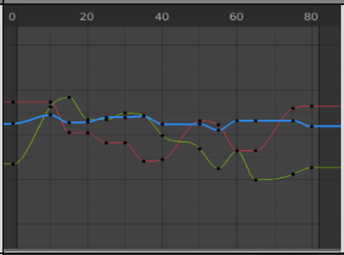
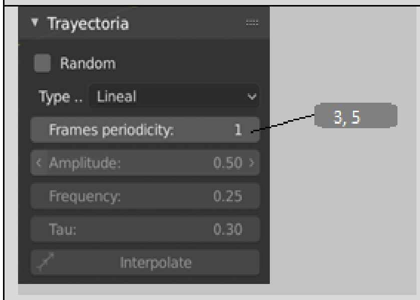
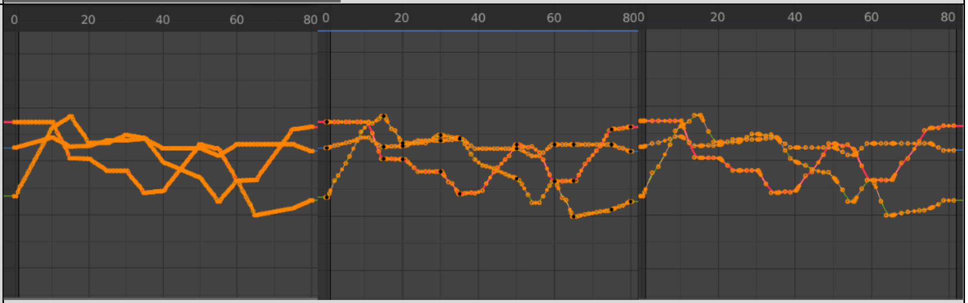
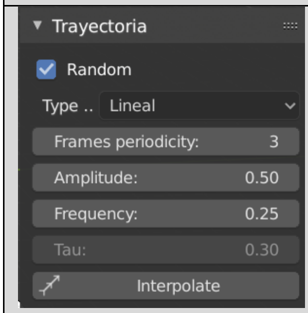
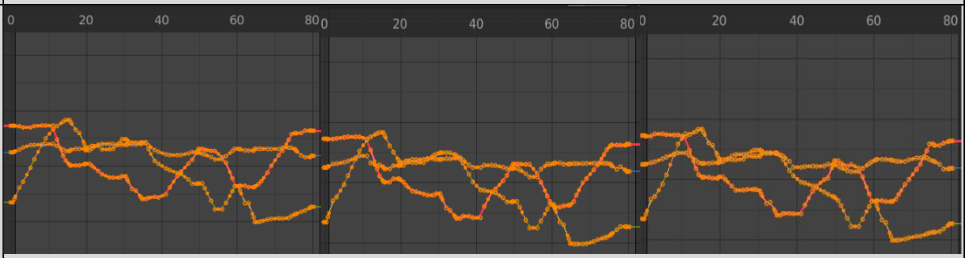
the property). When you have several keyframes, you can also adjust this FCurve in *Graph Editor.*

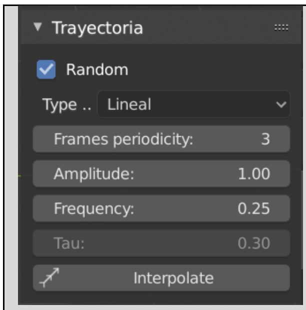
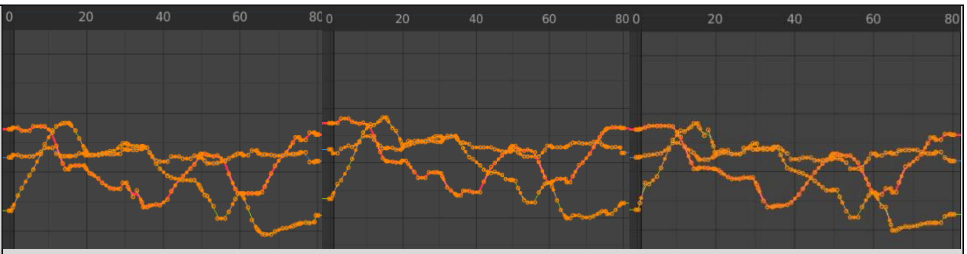
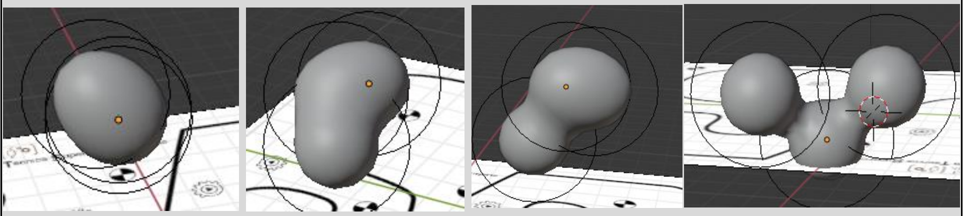
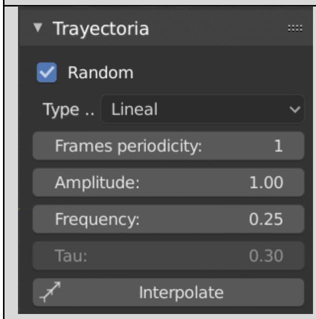
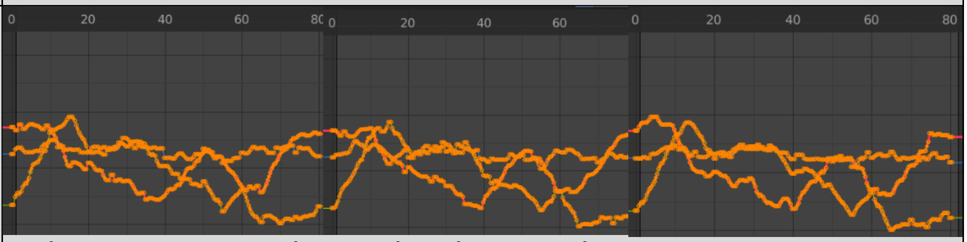
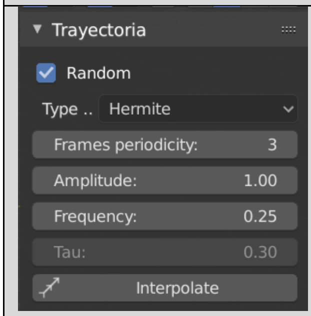
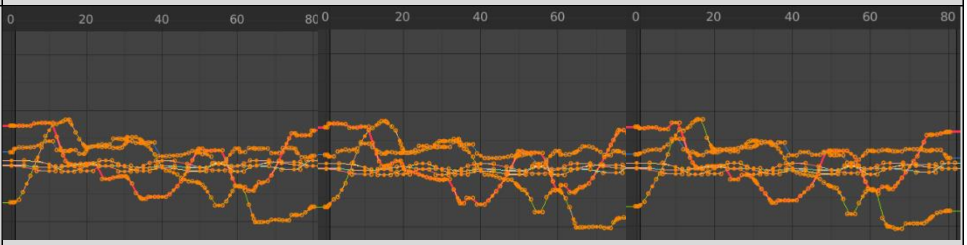
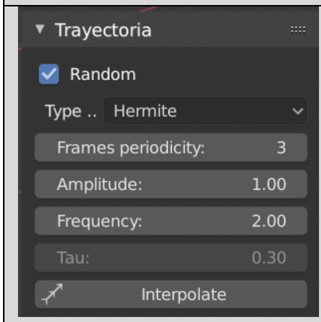
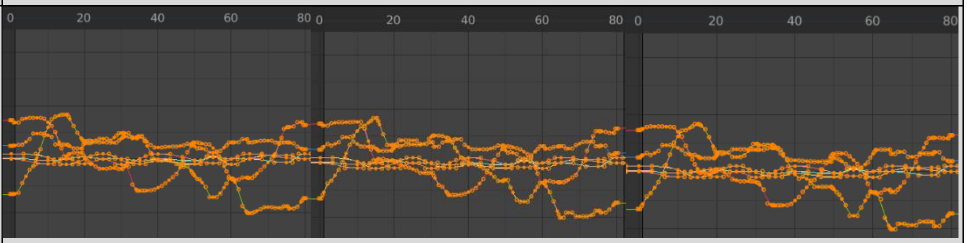
4) Our tool prevents user from setting inadequate values for **Current distance,** so you cannot set it to the value greater than 180°.

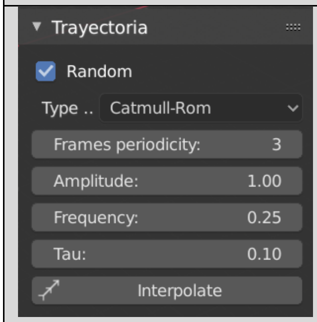
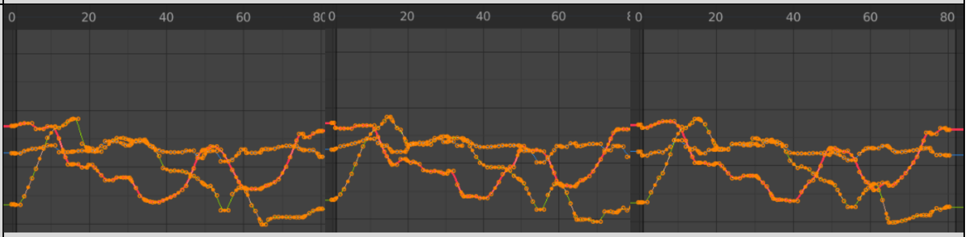
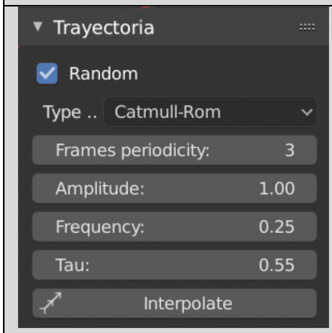
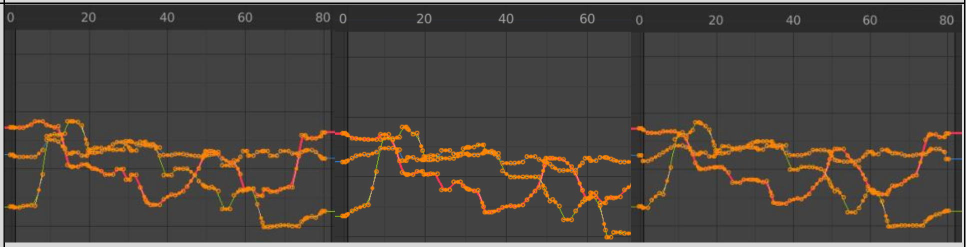
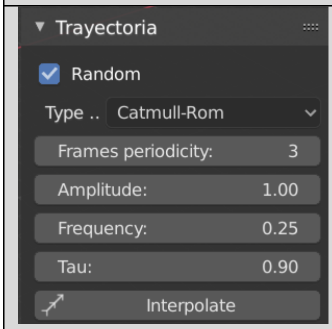
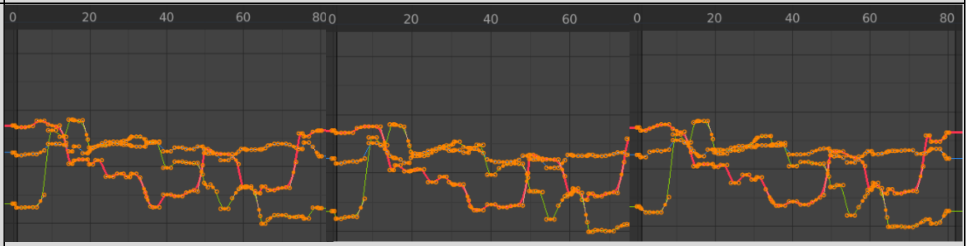
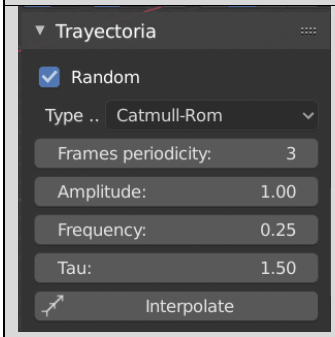
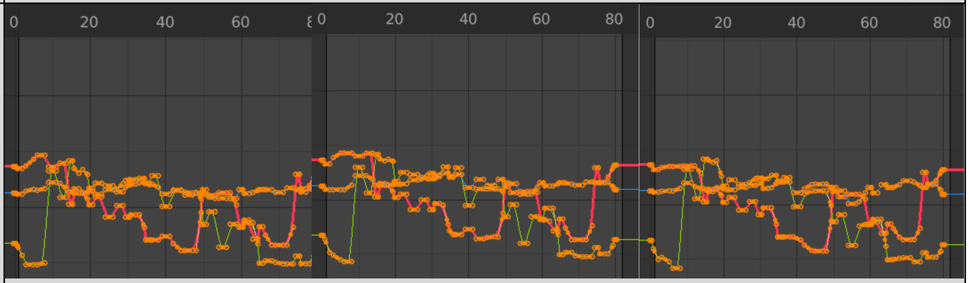
5. Add-on in use

5.1. Interpolation

Below we will give you an example of use of our tool, showing the properties and the result of applying them to the objects. Here we do not use **Velocity control**. We create three metaballs in the scene and define the trajectory using **Circuito.png**. All the results will be obtained by undertaking interpolation with following properties for all the balls and will be stored in separate trajectories.

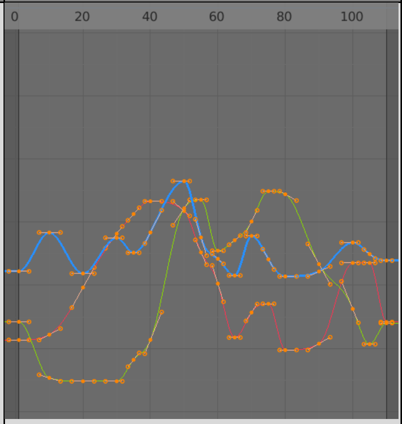
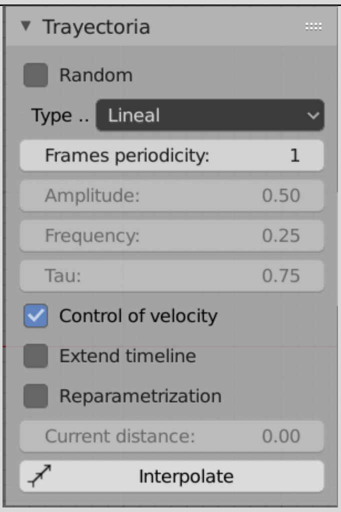
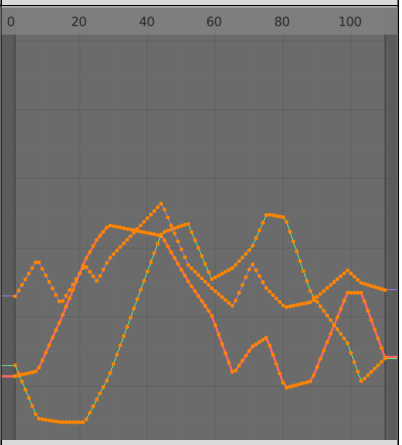
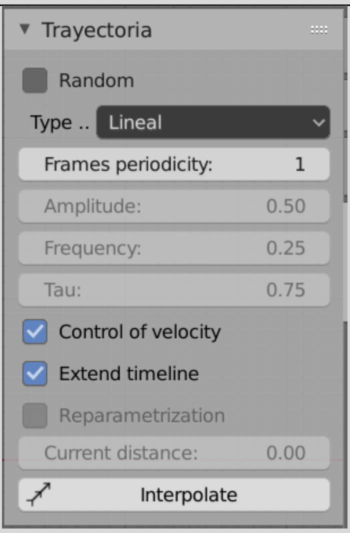
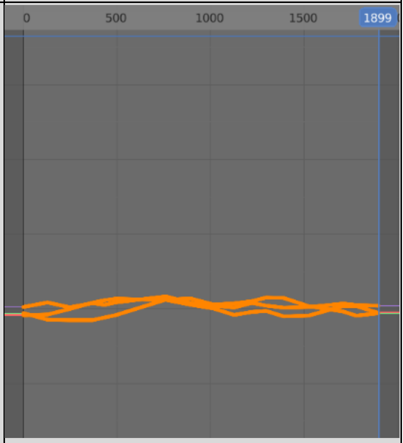
Properties	Description of the result
No interpolation	
 ▾ Trayectoria □ Random Type .. Lineal Frames periodicity: 1 Amplitude: 0.50 Frequency: 0.25 Tau: 0.30 Interpolate 3, 5	 This is the easiest case. We create linear interpolation, calculating the position for each 1, 3 or 5 keyframes. Linear interpolation produces not very natural movement in this case, because the velocity between keyframes is constant. Because of unselected *Random* option, all the balls have the same trajectory and move as one ball.
 ▾ Trayectoria ☑ Random Type .. Lineal Frames periodicity: 3 Amplitude: 0.50 Frequency: 0.25 Tau: 0.30 Interpolate	 Here we use *Random* option. In the picture there are trajectories of three balls which are very similar, but with little deviations.

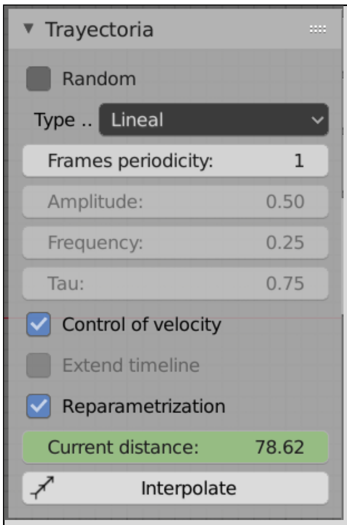
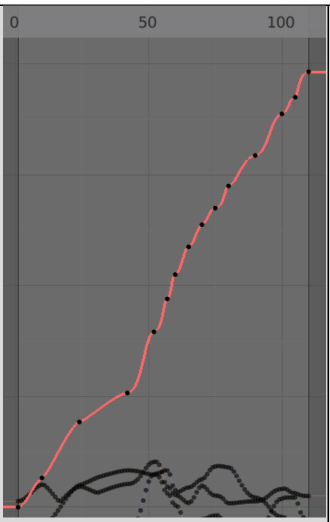
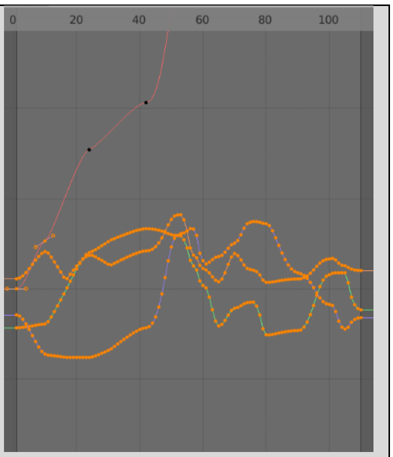
 Trayectoria ☑ Random Type .. Lineal Frames periodicity: 3 Amplitude: 1.00 Frequency: 0.25 Tau: 0.30 Interpolate	 In this case, we use greater amplitude, which implies greater deviation. Concerning the image, we have a more dynamic bubble than in previous case. Compare in the picture below bubbles, generated with amplitudes 0.5, 1, 1.5, 2.  Metaballs will stay united until amplitude is not greater than double radius of the balls.
 Trayectoria ☑ Random Type .. Lineal Frames periodicity: 1 Amplitude: 1.00 Frequency: 0.25 Tau: 0.30 Interpolate	 When we use *Random* option the more is Frames periodicity, the more smooth “bubbling” will be produced. If we set it to 1 the bubble will be trembling, but for 5 it will be changing shape leisurely.
 Trayectoria ☑ Random Type .. Hermite Frames periodicity: 3 Amplitude: 1.00 Frequency: 0.25 Tau: 0.30 Interpolate	 As it was mentioned above, linear interpolation produces not very natural behavior, but with Hermite and Catmull-Rom we can overcome it.
 Trayectoria ☑ Random Type .. Hermite Frames periodicity: 3 Amplitude: 1.00 Frequency: 2.00 Tau: 0.30 Interpolate	 Here the frequency effects the way how random deviation is generated, so that higher frequency values produce more dramatic

	changes between adjacent keyframes. However, it does not much effect on behavior of our bubble.
 Trayectoria ☑ Random Type .. Catmull-Rom Frames periodicity: 3 Amplitude: 1.00 Frequency: 0.25 Tau: 0.10 Interpolate	
 Trayectoria ☑ Random Type .. Catmull-Rom Frames periodicity: 3 Amplitude: 1.00 Frequency: 0.25 Tau: 0.55 Interpolate	
 Trayectoria ☑ Random Type .. Catmull-Rom Frames periodicity: 3 Amplitude: 1.00 Frequency: 0.25 Tau: 0.90 Interpolate	
 Trayectoria ☑ Random Type .. Catmull-Rom Frames periodicity: 3 Amplitude: 1.00 Frequency: 0.25 Tau: 1.50 Interpolate	 In the following pictures there are the results of Catmull-Rom interpolation with different values of τ. You can notice that greater values of τ produce more “frivolous” trajectories. For example, for $\tau = 1.5$ balls sometimes get really far from expected trajectory. One more crucial moment is that for high τ movement becomes very sharp, braking and accelerating very quickly.

5.2. Velocity

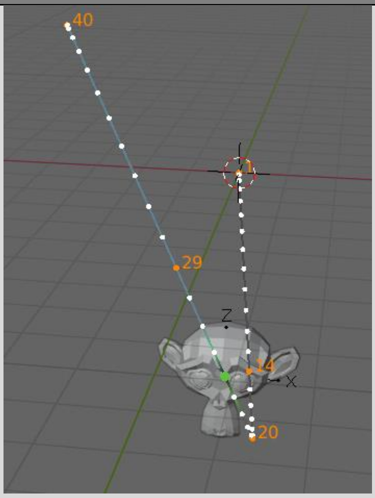
Below we will give you an example of use of our tool, showing the properties and the result of applying **Velocity control.** We create a metaball and define the trajectory using **Circuito.png**.

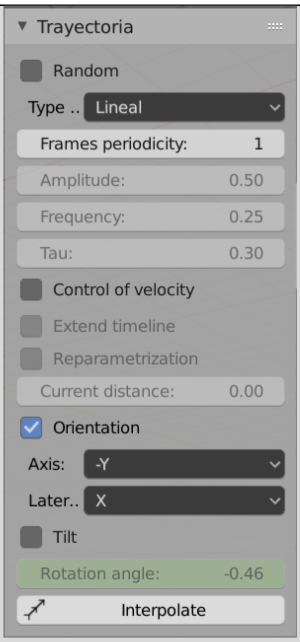
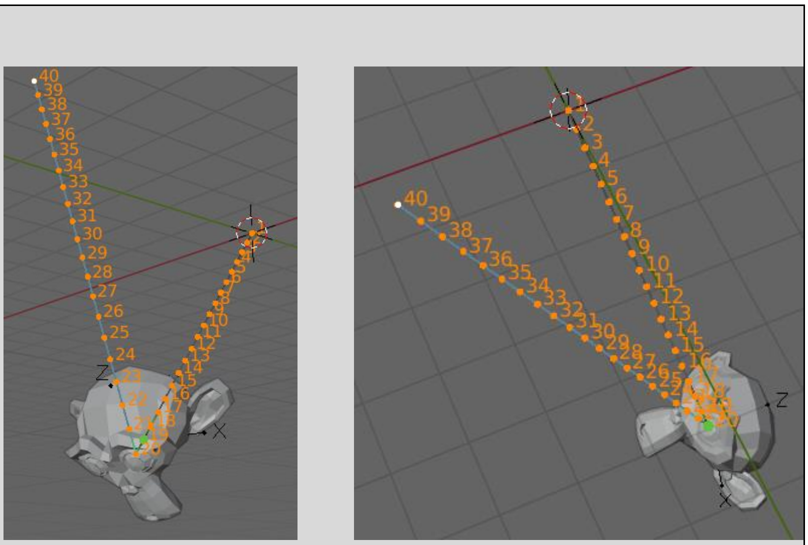
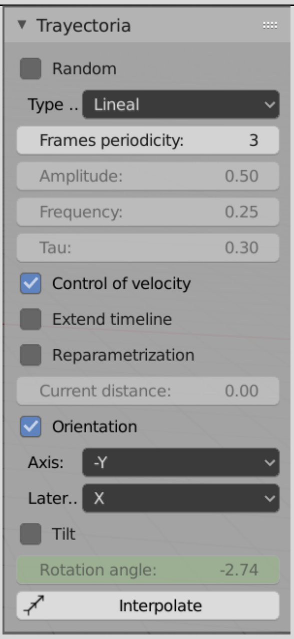
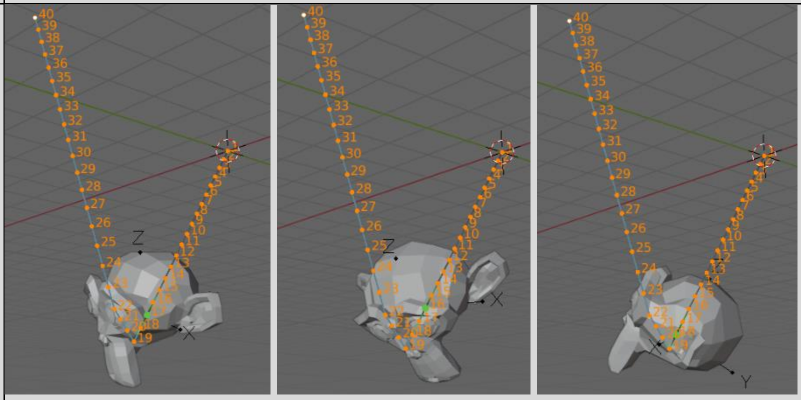
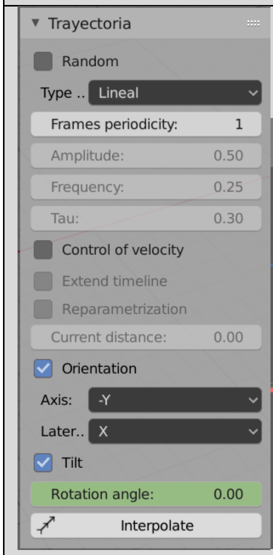
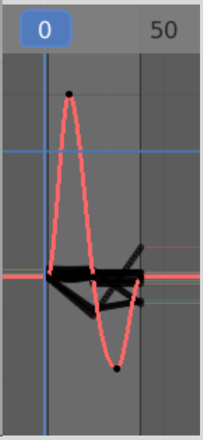
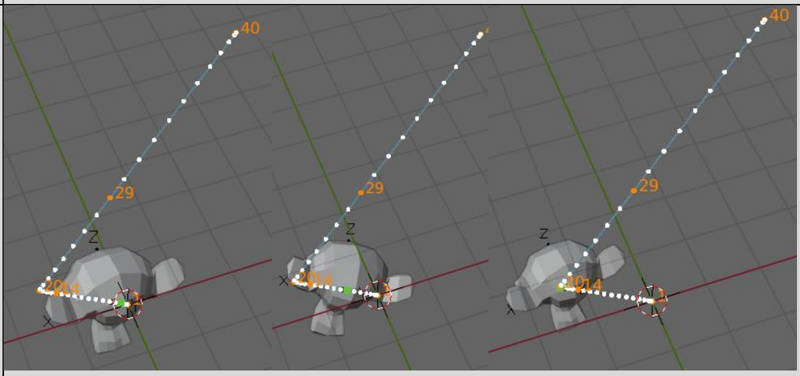
Properties	Description of the result
Initial movement	
 ▼ Trayectoria □ Random Type .. Lineal Frames periodicity: 1 Amplitude: 0.50 Frequency: 0.25 Tau: 0.75 ☑ Control of velocity □ Extend timeline □ Reparametrization Current distance: 0.00 Interpolate	 Object runs with constant velocity (the same duration of movement)
 ▼ Trayectoria □ Random Type .. Lineal Frames periodicity: 1 Amplitude: 0.50 Frequency: 0.25 Tau: 0.75 ☑ Control of velocity ☑ Extend timeline □ Reparametrization Current distance: 0.00 Interpolate	 Object runs with constant velocity 1 un./s (this movement is much slower, thus timeline is extended).

 ▼ Trayectoria ☐ Random Type .. Lineal Frames periodicity: 1 Amplitude: 0.50 Frequency: 0.25 Tau: 0.75 ☑ Control of velocity ☐ Extend timeline ☑ Reparametrization Current distance: 78.62 Interpolate  Right picture shows user-defined *Current distance* curve	 With such user-defined velocity we reach an effect of accelerating at straight segments of the trajectory and braking at the corners.

5.3. Rotation

Below we will give you an example of use of our tool, showing the properties and the result of applying **Orientation** control**.** We will use a monkey object, because it is convenient for showing rotation.

Properties	Description of the result
Initial movement	

 ▼ Trayectoria □ Random Type .. Lineal Frames periodicity: 1 Amplitude: 0.50 Frequency: 0.25 Tau: 0.30 □ Control of velocity □ Extend timeline □ Reparametrization Current distance: 0.00 ☑ Orientation Axis: -Y Later.. X □ Tilt Rotation angle: -0.46 Interpolate	
 ▼ Trayectoria □ Random Type .. Lineal Frames periodicity: 3 Amplitude: 0.50 Frequency: 0.25 Tau: 0.30 ☑ Control of velocity □ Extend timeline □ Reparametrization Current distance: 0.00 ☑ Orientation Axis: -Y Later.. X □ Tilt Rotation angle: -2.74 Interpolate	 Note that with frames periodicity greater than 1 object turns gradually using SLERP interpolation.
 ▼ Trayectoria □ Random Type .. Lineal Frames periodicity: 1 Amplitude: 0.50 Frequency: 0.25 Tau: 0.30 □ Control of velocity □ Extend timeline □ Reparametrization Current distance: 0.00 ☑ Orientation Axis: -Y Later.. X ☑ Tilt Rotation angle: 0.00 Interpolate  Rotation angle graph	 Object's lateral axis changes inclines according to defined rotation angle.