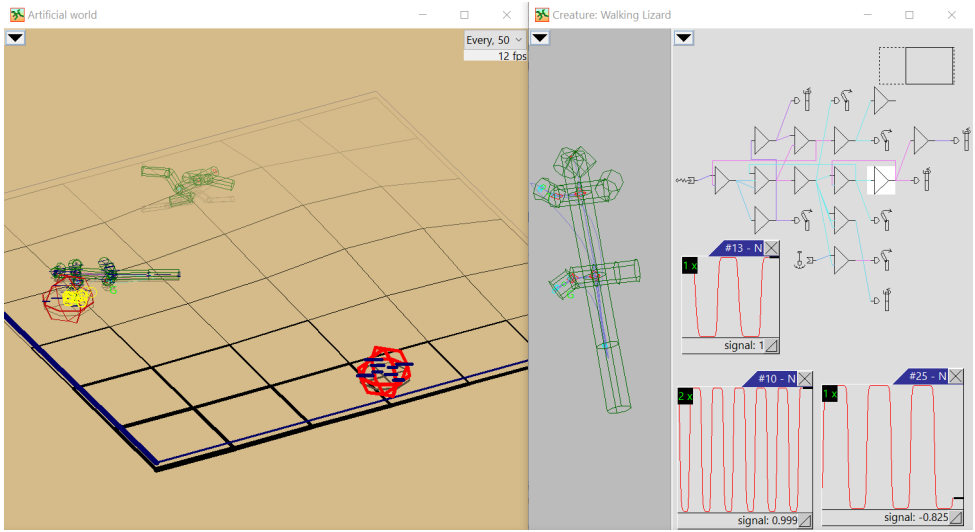
Framsticks visualization

Szymon Ulatowski Maciej Komosinski

www.framsticks.com

Simple wireframe display



OpenGL display

- Supported in Windows GUI, QT GUI and Theater

OpenGL display

- Supported in Windows GUI, QT GUI and Theater
- Can be customized and extended by scripting

OpenGL display

- Supported in Windows GUI, QT GUI and Theater
- Can be customized and extended by scripting
- Image export – exact screen copy

OpenGL display

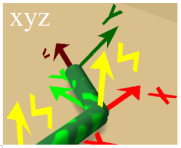
- Supported in Windows GUI, QT GUI and Theater
- Can be customized and extended by scripting
- Image export – exact screen copy
- Scene export from OpenGL/SSG

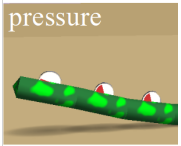
OpenGL display

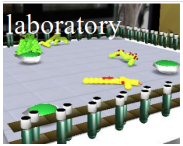
- Supported in Windows GUI, QT GUI and Theater
- Can be customized and extended by scripting
- Image export – exact screen copy
- Scene export from OpenGL/SSG

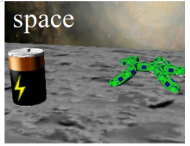
OpenGL display

- Supported in Windows GUI, QT GUI and Theater
- Can be customized and extended by scripting
- Image export – exact screen copy
- Scene export from OpenGL/SSG

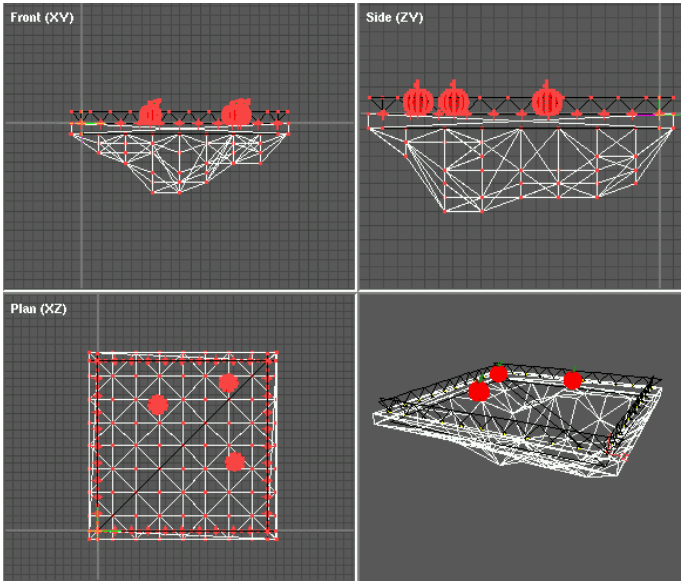




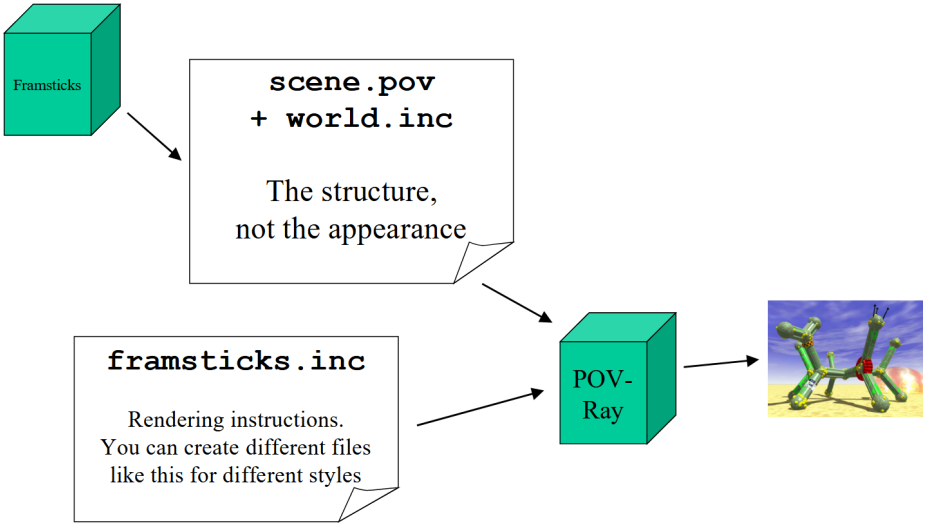




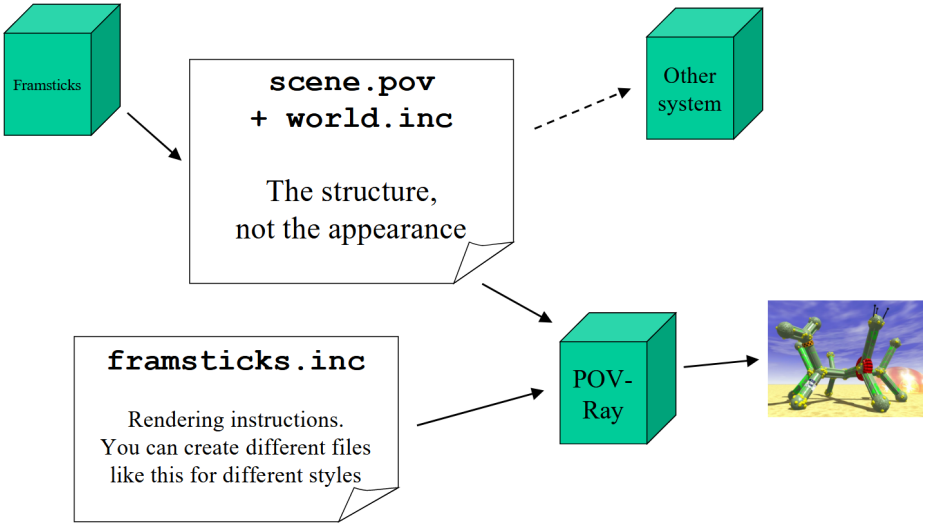
Scene export from OpenGL/SSG: `*.obj`, `*.dxf`, ...



POV-Ray scene export logic



POV-Ray scene export logic



POV-Ray scene export – objects

Simulator objects of the type `Class` (can be `Object`, `Scene`, `World`, `Part`, `Joint`, `Neuro`, ...) are exported like this:

```
#declare field_Class_FieldName = ...;      ← value used by all subsequent objects of
                                             type Class (until another #declare)

Begin Class ()
...
End Class ()
```

POV-Ray scene export – objects

Simulator objects of the type `Class` (can be `Object`, `Scene`, `World`, `Part`, `Joint`, `Neuro`, ...) are exported like this:

```
#declare field_Class_FieldName = ...;     ← value used by all subsequent objects of
                                            type Class (until another #declare)

Begin Class ()
...
End Class ()
```

For example, the `World` object in `world.inc` (a common file included by all scene files – contains environment parameters):

```
#declare field_World_wrldtyp = 2;
#declare field_World_wrldsiz = 20;
#declare field_World_wrldwat = -1;
BeginWorld()
#declare MapData = array[..][..]{{...data...}, ...}
EndWorld()
```

POV-Ray export – scene file structure

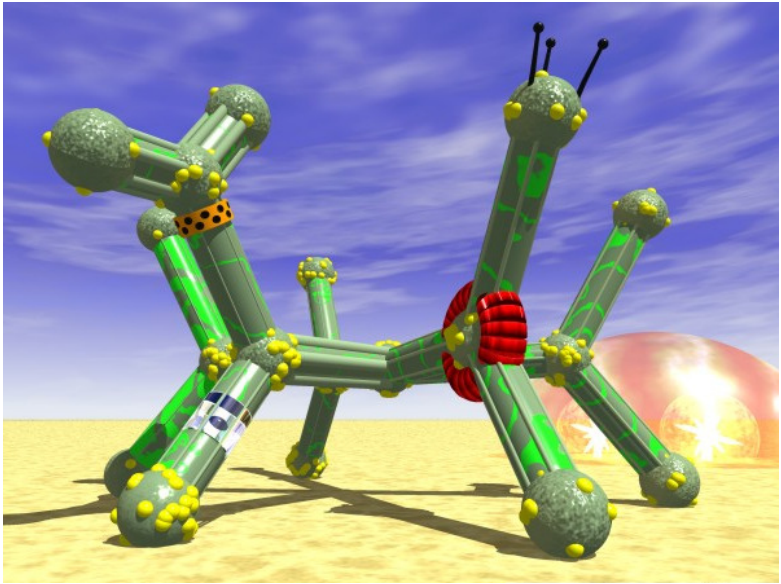
`scene.pov` – snapshot of the simulator state:

- `BeginPart(part_id)`
  `PartGeometry(<position>, <orientation>)`
  - `BeginNeuro("neuron_class_name")`
    `EndNeuro()`
  - `...`

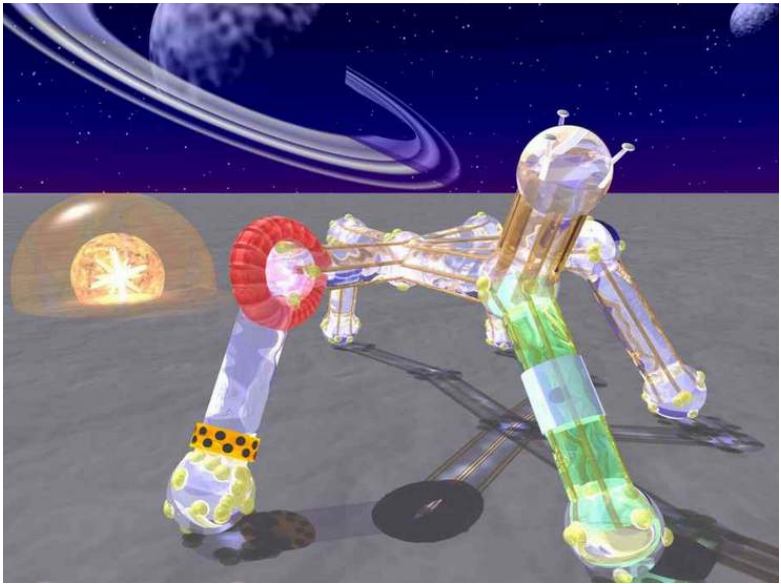
  `EndPart()`
- `BeginJoint(partt1_id, part2_id)`
  `JointGeometry(<beginposition>, <endposition>, <orientation>)`
  - `BeginNeuro("neuron_class_name")`
    `EndNeuro()`
  - `...`

  `EndJoint()`
- `...`

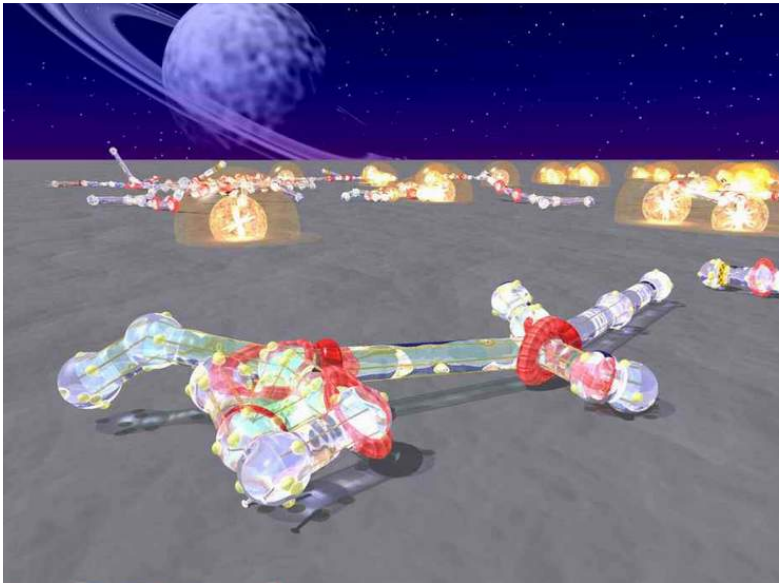
Style: Classic



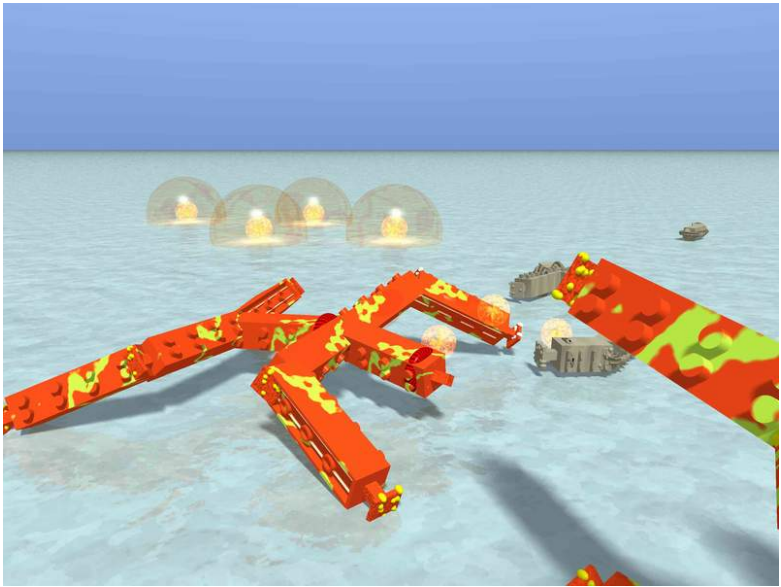
Style: Planet



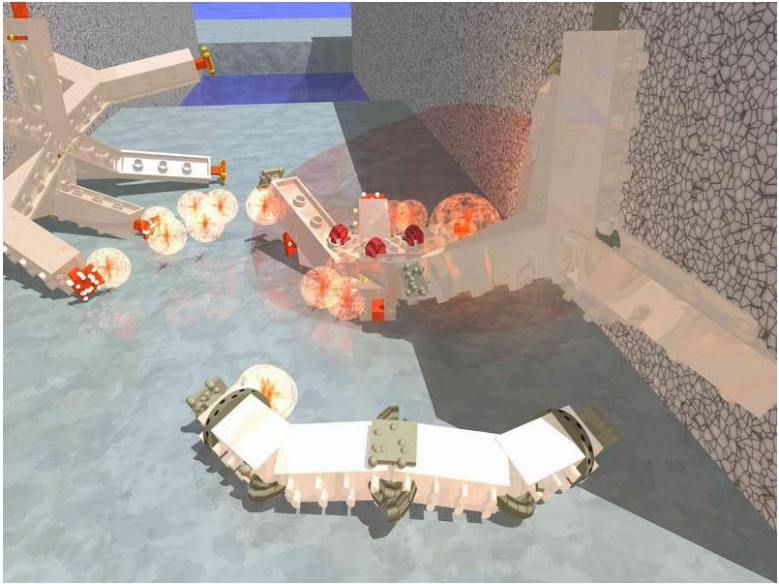
Style: Planet



Style: Blocks

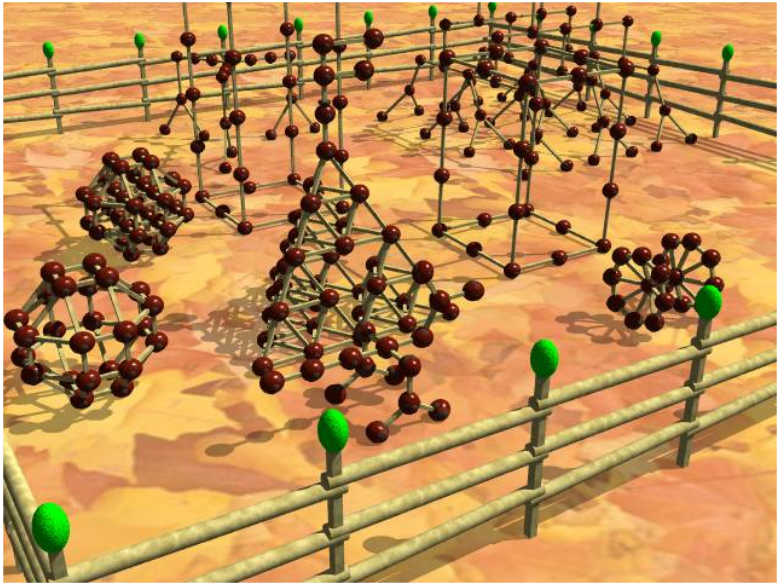


Style: Blocks



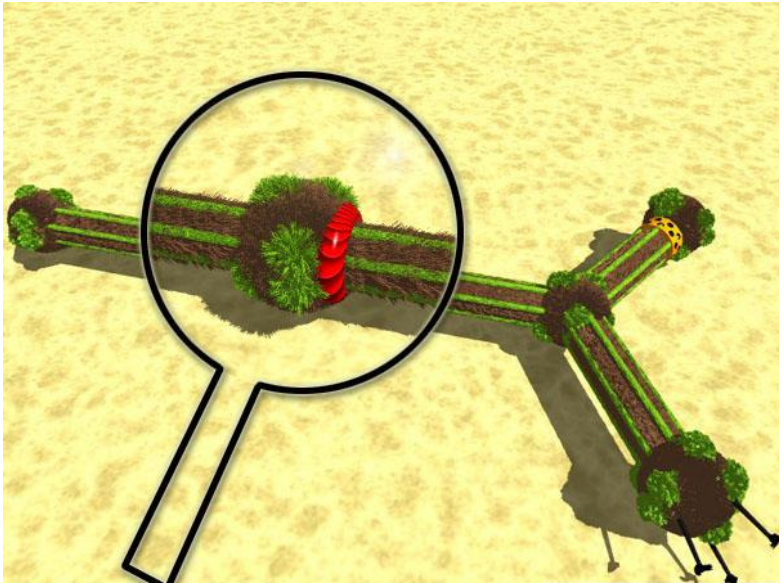
Style: Ghost

Style: Chestnuts



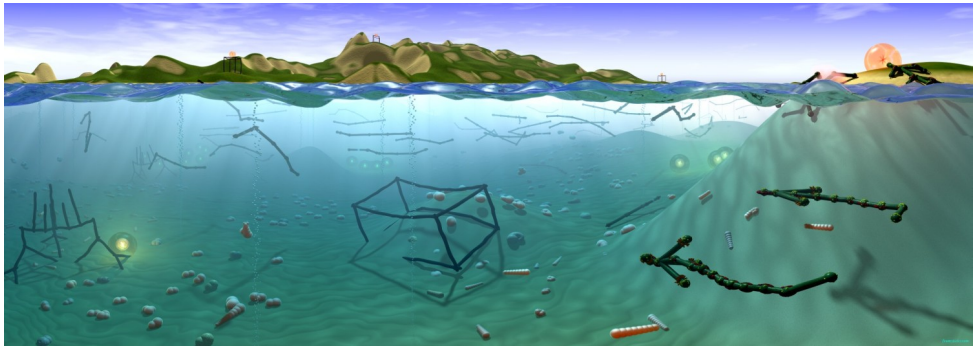
Style: Wookie

Style: Wookie

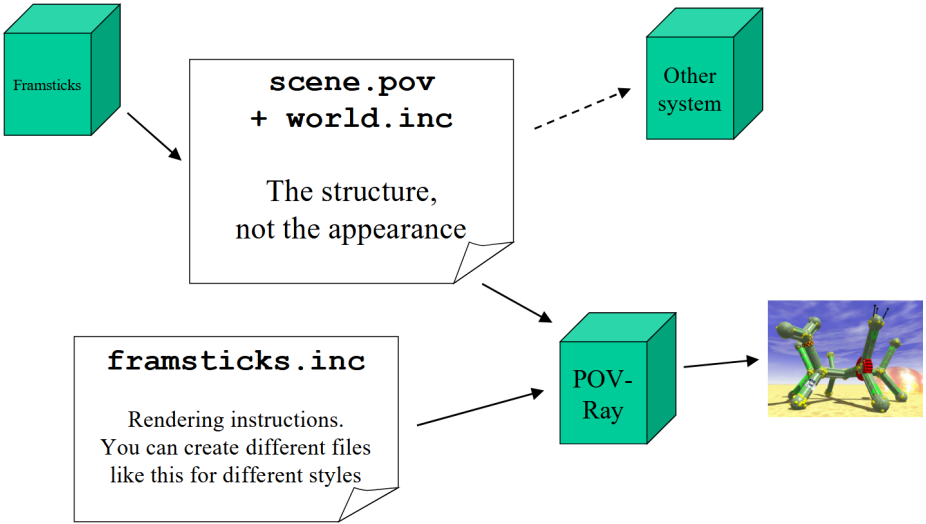


Large poster art: sands

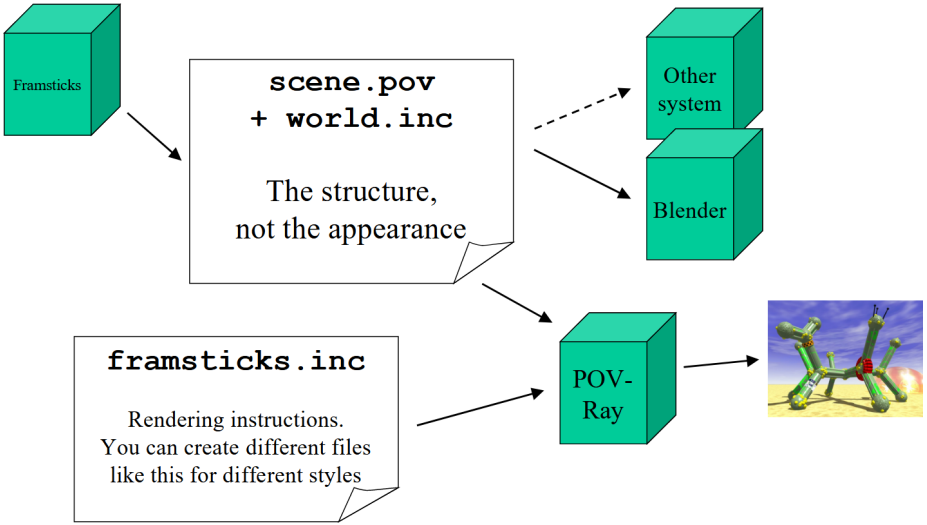
Large poster art: underwater



POV-Ray scene export logic

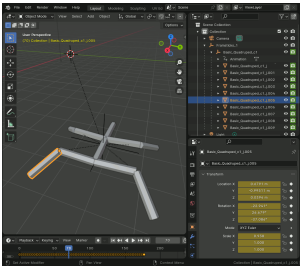


POV-Ray scene export logic: Blender



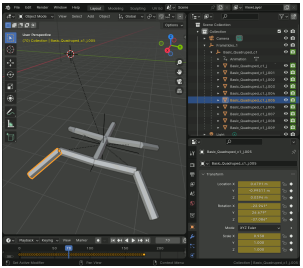
POV-Ray → Blender export

- View the POV-Ray animation before rendering it in POV-Ray



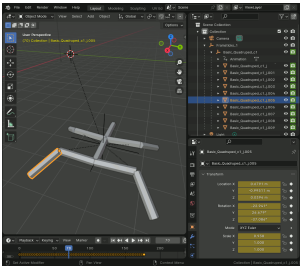
POV-Ray → Blender export

- View the POV-Ray animation before rendering it in POV-Ray
- Use animation tools to make the camera precisely follow your intended path (matching imported Framsticks creatures) and then store Blender camera movement back into the original POV-Ray files, so POV-Ray renderings will follow Blender's camera movement!



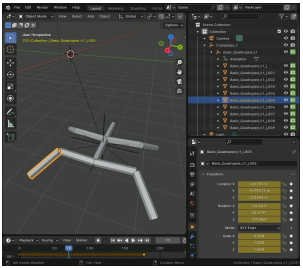
POV-Ray → Blender export

- View the POV-Ray animation before rendering it in POV-Ray
- Use animation tools to make the camera precisely follow your intended path (matching imported Framsticks creatures) and then store Blender camera movement back into the original POV-Ray files, so POV-Ray renderings will follow Blender's camera movement!
- Use Framsticks creatures as a part of a Blender animation project



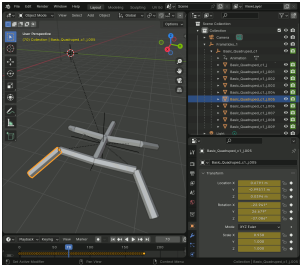
POV-Ray → Blender export

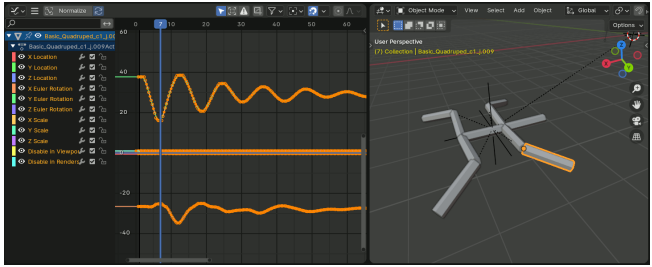
- View the POV-Ray animation before rendering it in POV-Ray
- Use animation tools to make the camera precisely follow your intended path (matching imported Framsticks creatures) and then store Blender camera movement back into the original POV-Ray files, so POV-Ray renderings will follow Blender's camera movement!
- Use Framsticks creatures as a part of a Blender animation project
- https://www.framsticks.com/3d-animations-in-blender



POV-Ray → Blender export

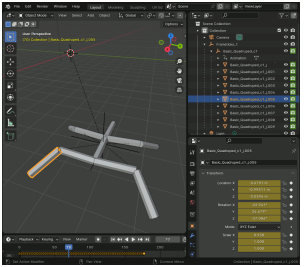
- View the POV-Ray animation before rendering it in POV-Ray
- Use animation tools to make the camera precisely follow your intended path (matching imported Framsticks creatures) and then store Blender camera movement back into the original POV-Ray files, so POV-Ray renderings will follow Blender's camera movement!
- Use Framsticks creatures as a part of a Blender animation project
- https://www.framsticks.com/3d-animations-in-blender

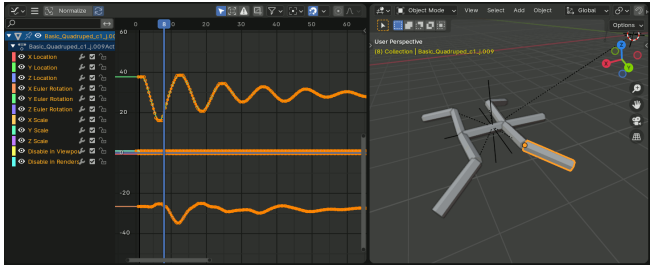




POV-Ray → Blender export

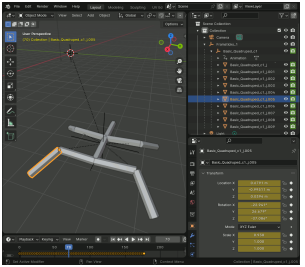
- View the POV-Ray animation before rendering it in POV-Ray
- Use animation tools to make the camera precisely follow your intended path (matching imported Framsticks creatures) and then store Blender camera movement back into the original POV-Ray files, so POV-Ray renderings will follow Blender's camera movement!
- Use Framsticks creatures as a part of a Blender animation project
- https://www.framsticks.com/3d-animations-in-blender

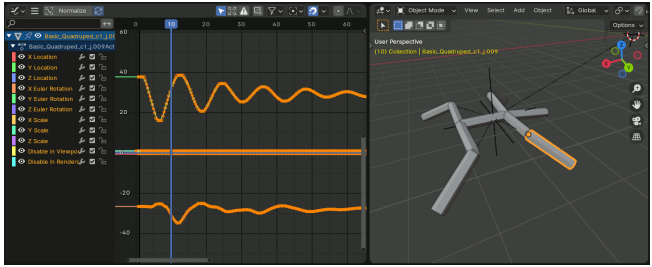




POV-Ray → Blender export

- View the POV-Ray animation before rendering it in POV-Ray
- Use animation tools to make the camera precisely follow your intended path (matching imported Framsticks creatures) and then store Blender camera movement back into the original POV-Ray files, so POV-Ray renderings will follow Blender's camera movement!
- Use Framsticks creatures as a part of a Blender animation project
- https://www.framsticks.com/3d-animations-in-blender





POV-Ray → Blender export

- View the POV-Ray animation before rendering it in POV-Ray
- Use animation tools to make the camera precisely follow your intended path (matching imported Framsticks creatures) and then store Blender camera movement back into the original POV-Ray files, so POV-Ray renderings will follow Blender's camera movement!
- Use Framsticks creatures as a part of a Blender animation project
- https://www.framsticks.com/3d-animations-in-blender