

VRM Animation□□□□

- [VRM Animation](#)
- [VRM-Animation](#) □□
- [VRM-Animation](#) □□
- [□□□□□□□□](#)

VRM Animation

[https://github.com/vrm-c/vrm-](https://github.com/vrm-c/vrm-specification/tree/master/specification/VRMC_vrm_animation-1.0)

[specification/tree/master/specification/VRMC_vrm_animation-1.0](https://github.com/vrm-c/vrm-specification/tree/master/specification/VRMC_vrm_animation-1.0)

“VRM Animation”

VRM Animation VRM

- VRM Animation **VRM**
- glTF**
- UniVRM **Unity VRM Animation**

- glTF animation**
- VRM** **glTF**
 - `VRMC_vrm_animation`
 - `.vrma`
- Humanoid bone animation** ()
 - `glTF Humanoid`
 - `VRM`
- Expression animation** ()
 - `glTF`
 - `VRM VRM custom expressions`
- Gaze control animation**
 - `glTF`

VRM Animation

VRM Animation

VRM Animation

- VRM Animation `authoring tools`
- `motion capture`
- `live streaming` `photo` `VRM Animation`

- [VRM Animation](#) [Metaverse](#)
- [game engines](#) [VRM Animation](#) [game development](#)

VRM Animation

[showcase](#)

[VRM Animation](#) [draft](#) [VRM Animation](#) [draft](#)

[GitHub Issues](#) [Pull Request](#)

- [UniVRM](#)
- [@pixiv/three-vrm](#)
- [VRM Add-on for Blender](#)
- [bvh2vrma](#)
- [VRoid Hub](#)
- [AnimationClipToVrmaSample](#)
- [VMagicMirror](#)
- [VRM Posing Desktop](#)
- [VRM](#)
- [VRMA, BVH](#) [VRM](#)

VRM Animation

VRM-1.0 [VRM-Animation](#)

import

[import](#) [target](#)

export

Unity humanoid VRM-Animation

EDITOR

export()

VRM-Animation

[VRM10Viewer\(📄📄📄📄📄📄\)](#)


VRM-Animation











 BVH Converter

-  -  -  - 

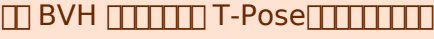

Assets/VRM10/Editor/VrmAnimationMenu.cs


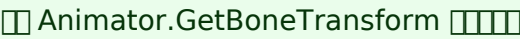



 VrmAnimation

-  humanoid  hierarchy() 
-  hierarchy()  T-Pose 
-  humanoid 

bvh 



UNITY HUMANOID 
 Animator.GetBoneTransform 



VRM Animation exporter 

```
Transform humanoid_hierarchy;  
  
var data = new ExportingGltfData();  
using var exporter = new VrmAnimationExporter(  
    data, new GltfExportSettings());  
exporter.Prepare(humanoid_hierarchy.gameObject);
```

VRM Animation exporter

```
exporter.Export((VrmAnimationExporter vrma) =>
{
    // get human bones
    var map = new Dictionary<HumanBodyBones, Transform>();
    var animator = bvh.Root.GetComponent<Animator>();
    foreach (HumanBodyBones bone in Enum.GetValues(typeof(HumanBodyBones)))
    {
        if (bone == HumanBodyBones.LastBone)
        {
            continue;
        }
        var t = animator.GetBoneTransform(bone);
        if (t == null)
        {
            continue;
        }
        map.Add(bone, t);
    }

    vrma.SetPositionBoneAndParent(map[HumanBodyBones.Hips], bvh.Root.transform);

    foreach (var kv in map)
    {
        var vrmBone = Vrm10HumanoidBoneSpecification.ConvertFromUnityBone(kv.Key);
        var parent = GetParentBone(map, vrmBone) ?? bvh.Root.transform;
        vrma.AddRotationBoneAndParent(kv.Key, kv.Value, parent);
    }
}
```



```
// get animation
var animation = bvh.Root.gameObject.GetComponent<Animation>();
var clip = animation.clip;
var state = animation[clip.name];

var time = default(TimeSpan);
for (int i = 0; i < bvh.Bvh.FrameCount; ++i, time += bvh.Bvh.FrameTime)
```

```
{
    state.time = (float)time.TotalSeconds;
    animation.Sample();
    vrma.AddFrame(time);
}
```



```
});
var glb = data.ToGlbBytes();
```

```

[
    [glb [VRMC_vrm_animation ]]
```




VRM-0.X[]...

- []

- []

VRM[]T-Pose[]

UniVRM []

UniVRM [] vrm-1.0 [] ControlRig []ControlRig[]

[ControlRig](#) []

[] **UNITY** [] **MECANIM HUMANOID** []

[humanoid avatar]



UniVRM []

https://github.com/vrm-c/vrm-specification/blob/master/specification/VRMC_vrm_animation-1.0/how_to_transform_human_pose.ja.md

[] HumanoidBone [] T-Pose[]