VRM Animation (动画)

- VRM Animation
- VRM-Animation 导入
- VRM-Animation 导出
- 解决初始姿态差异的问题

VRM Animation

https://github.com/vrm-c/vrm-specification/tree/master/specification/VRMC_vrm_animation-1.0

什么是**“VRM Animation”**？

VRM Animation 是一种格式，用于描述在 VRM 中定义的人型模型的动画。

- 同一个 VRM Animation□用于所有 **VRM** 文件
- **glTF** 描述，是一个跨平台格式。
- UniVRM 提供了在 **Unity** 中导入和导出 **VRM Animation** 的标准解决方案

文件的内容

- □**glTF**为**animation**。
- 扩展**VRM** 的每个组件与要实现动画效果的 **glTF** 节点相关联
    - [`VRMC_vrm_animation` 扩展中定义。
    - [`.vrma` 扩展名。
- □□ 可以描述 **Humanoid bone animation (**骨骼动画）
    - 扩展内指定了 glTF 节点与哪个 Humanoid 骨骼相对应。
    - 实现将转换动画的骨骼旋转，以正确反映到目标 VRM。
- 可以描述 **Expression animation (**表情动画**)**
    - 文件中包含表情的动画权重，表示为 glTF 节点的坐标。
    - 除了 VRM 中定义的预设表情，如果目标 VRM 定义了对应的表情，也支持 custom expressions（自定义表达式）。
- □□□描述 **Gaze control animation**（视线动画）
    - 在扩展内指定了代表视线方向的 glTF 节点。

VRM Animation 的用途

通过 VRM Animation□以在跨应用程序和模型使用人型模型的动画。

以下是一些 VRM Animation 可能的应用示例：

- 使用支持 VRM Animation 的 authoring tools 创建动画
- 在各种应用程序中使用通过 motion capture 记录的动画
- 在 live streaming 和 photo 应用中使用使用 VRM Animation 创建的动画

- 将使用 VRM Animation 创建的动画带到 Metaverse，播放、与其他用户分享并共同使用
- 在 game engines 中加载 VRM Animation 文件，并在 game development 中使用

支持 VRM Animation 的应用程序

showcase

警告

得益于社区的支持，许多应用程序在支持 VRM Animation 的 draft 规范方面进行了合作。VRM Animation 规范刚刚正式发布。如果您使用支持 draft 版本的应用

备注

若希望在此列表中增加新的应用程序，

请通过 GitHub issues Pull Request。

请务必提供应用程序的名称与网址。

- UniVRM
- @pixiv/three-vrm
- VRM Add-on for Blender
- bvh2vrma
- VRoid Hub
- AnimationClipToVrmaSample
- VMagicMirror
- VRM Posing Desktop
- VRM スプリングボーン調整ツール
- VRMA, BVH をアップロードして VRM を動かすやつ

使用 VRM Animation 开发应用程序

VRM-1.0 已支持 VRM-Animation。

import

import(开发相关) target(开发相关)

export

此外，如果 Unity 中存在 humanoid 动画，可以逐帧导出为 VRM-Animation。

> EDITOR 专用

export(开发相关)

VRM-Animation 导入

VRM考[1.0Viewer(开发相关，无翻译)](VRM)

VRM-Animation 导出

请参考实装的 BVH Converter

- `menu` - `VRM1` - `Experimental` - `Convert BVH to VRM-Animation...`

`Assets/VRM10/Editor/VrmAnimationMenu.cs`

说明

输入

为了导出 VrmAnimation，需要以下条件：

- 符合 humanoid 规范的游戏对象 hierarchy(层级) 结构。
- 确保人体模型的 hierarchy(层级) 结构处于 T-Pose 状态。
- 随着时间的推移，humanoid 的层级结构姿态会发生变化。

为了简单地达到以上条件，我们使用了bvh 格式文件。

> 警告：
>
> 如果 BVH 的初始姿势不是 T-Pose，将无法正常工作。

> **UNITY HUMANOID** 不是必须的
>
> 使用 Animator.GetBoneTransform 会更简单。

初始化

VRM Animation exporter 使用说明

```
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);
    }
```

输出

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

> 提示
> 这是在通常的 glb 基础上添加了 VRMC_vrm_animation 的版本

解决初始姿态差异的问题

> 使用与VRM-0.X兼容的数据时，在某些情况下，问题可能不会显现出来，如果...
> - 模型骨架
> - 动画骨架
>
> 都符合VRM兼容的标准化T-Pose，那么无需转换即可运行。

使用 **UniVRM** 实装的解决方案

在 UniVRM 的 vrm-1.0 中，包含了 ControlRig 这一组件，它能够支持初始姿态的转换。在运行时加载的过程中，ControlRig会自动进行设置和初始化。

使用 ControlRig 操作未标准化的模型（开发相关，无翻译）

> 也可以让 **UNITY** 的 **MECANIM HUMANOID** 代替
> [`humanoid avatar`

解决方案

在 UniVRM 之外，也可以实现动画转换。

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

模型和动画的骨架需要满足 HumanoidBone 规范并处于 T-Pose，这是必要条件。