in VRM 勉強会 5

VRM 1.0 に向けた、
正規化の問題点の整理と解決案の提示

TokageITLab.
シルク・レニュー

はじめに

発表時間の都合上、
かなり内容を削ぎ落としたスライドになっています。
記事の方で詳しく解説を行っているので、
スライドの内容について理解を深めたい方はそちらを読むことを推奨します。

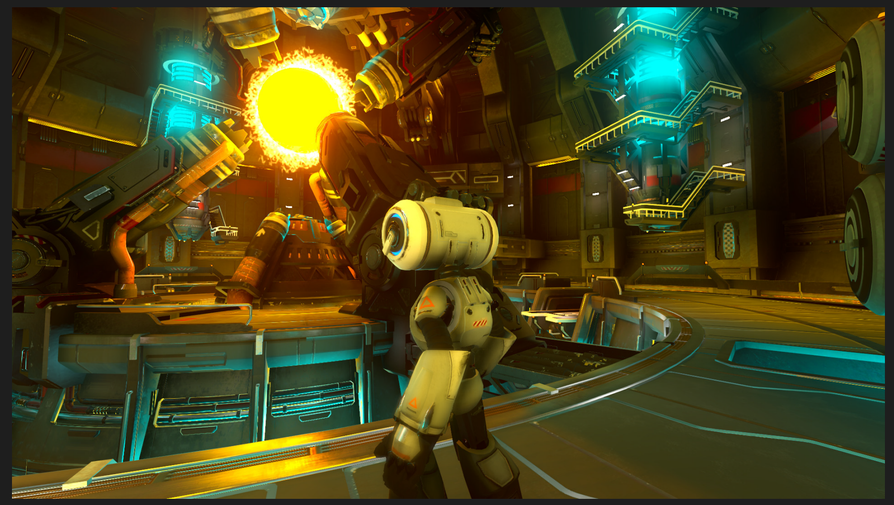
https://qiita.com/TokageItLab/items/e5880123a9f508b2769d

自己紹介

シルク・レニュー／トカゲ

- VRM 1.0 のリポジトリでよく文句言ってる
- Godot Engine の 3D ボーンアニメーション周りのコントリビューター
- 最近は Blender Add-on を作ったりもしてる

TokageITLab
viridflow



※最近人気上昇中の Godot Engine
Facebook Reality Labsからの
寄付を受け VR 対応が加速している

本題

VRM 1.0 では

ローカル軸の方向（ボーンのレスト回転）を保持するべき

なぜ今更？

現在の正規化の仕様では
Unity 上で VRM はなんとなく正常に動くように見えてしまっているが
Unity 外の環境になるとそうはいかなくなる

正規化のせいで…

glTF ネイティブな Godot や three.js (three-vrm) 上で
VRM をアバターとして動かすのは難しい
※表示するぐらいならできるケド

VRM 1.0 は**プラットフォームを越えた汎用性**を目指す

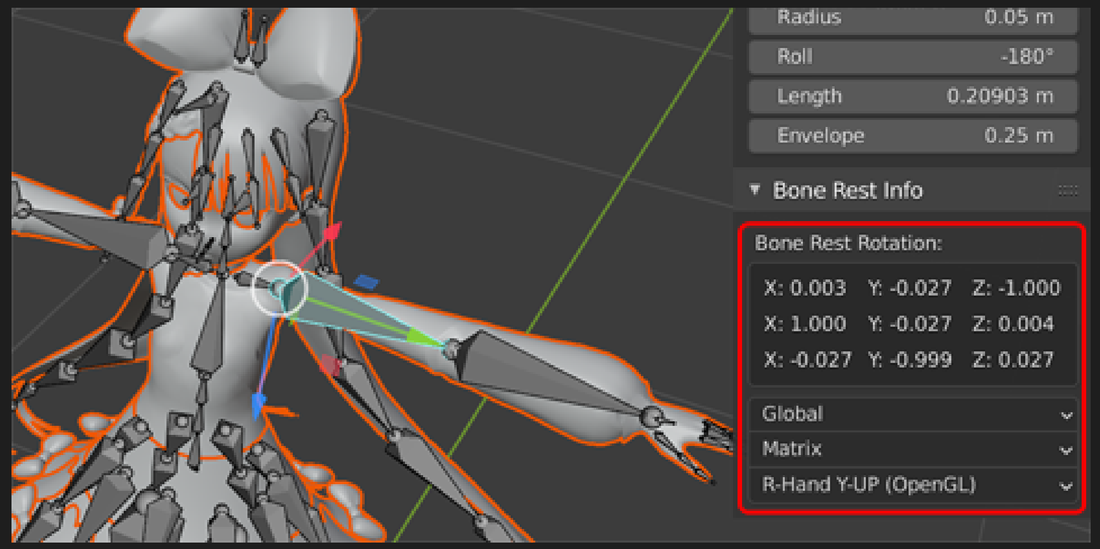
↓

ローカル軸の方向の定義と正規化の実装の問題に
向き合う必要がある

そもそも正規化って何してるの？

全てのボーンのローカル軸の方向を破棄する
→オイラーで言うと **X 軸 Y 軸 Z 軸全てのレスト回転を 0° に強制**する

ローカル軸の方向（レスト回転）とは



それぞれのボーンの
ローカル座標における X Y Z 軸が
どの方向を向いているかという情報

例えば、X 軸回転をさせる時に
どの方向へ回転するかという情報

※レストはポーズを付ける時の基点という意味

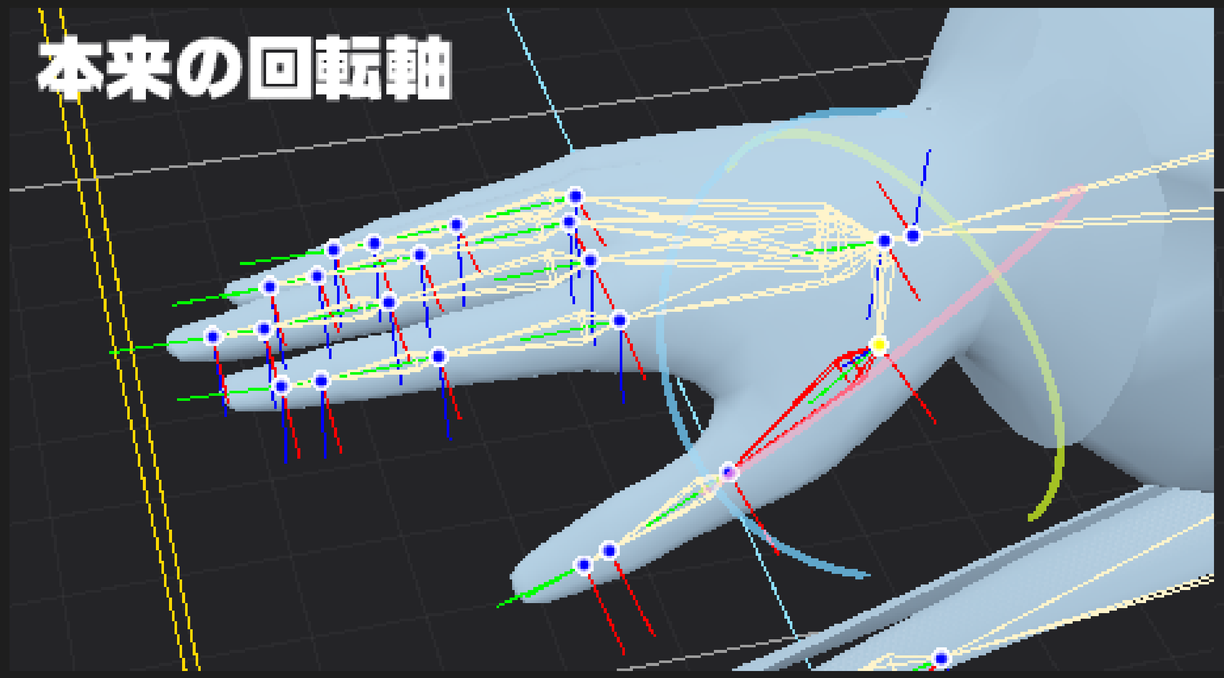
本当に捨ててもいい情報なのだろうか？

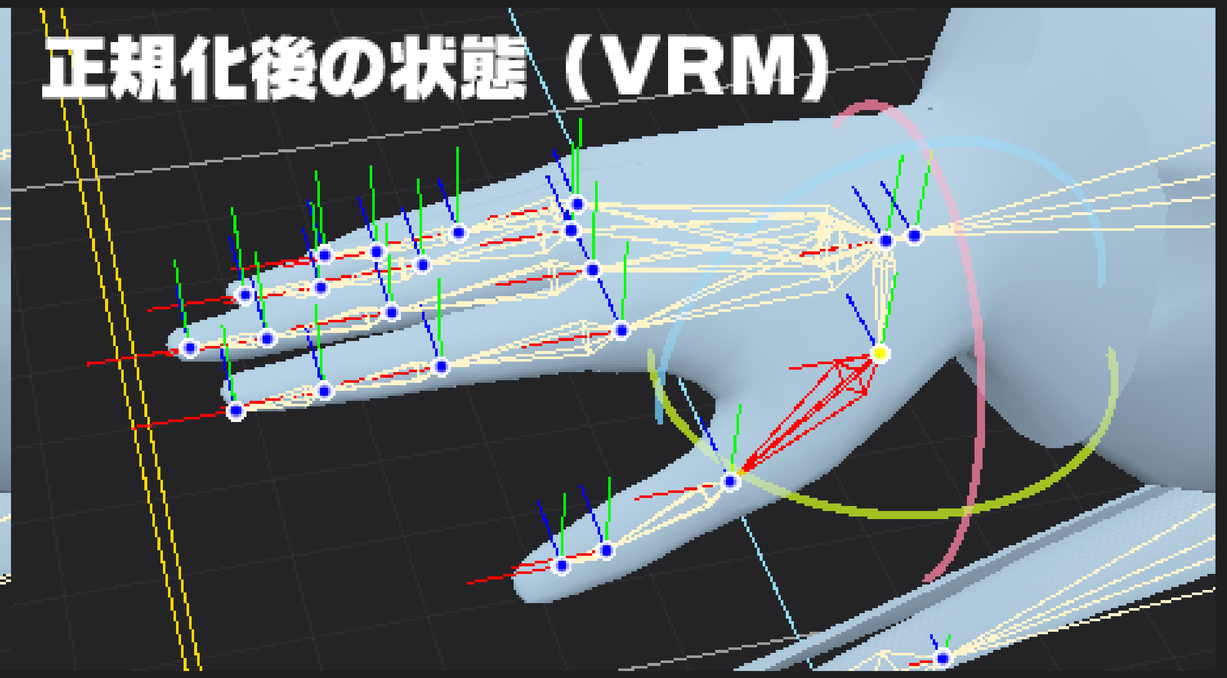
ローカル軸の方向という情報を捨ててはいけない！

ローカル軸の方向が**必要となる場面は非常に多い！**

ローカル軸の方向を破棄した際に起きる問題

斜めのボーン（親指や足）を曲げるべき方向が分からなくなる



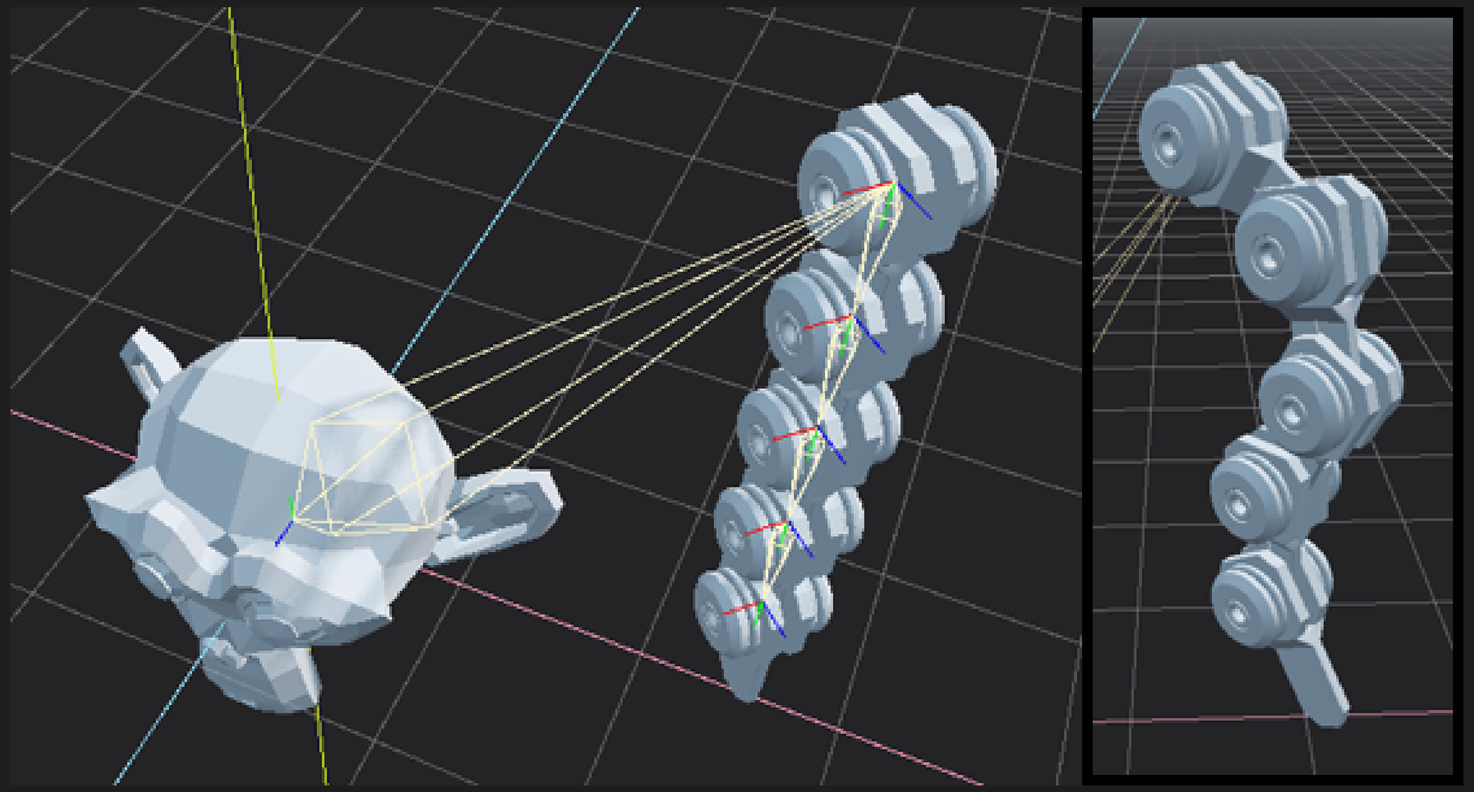


- 斜めのボーンをオイラー角で制御する時、ボーンを曲げる方向がおかしくなる
- アニメーションのように同じ値を複数のモデルに適用する時、斜めのボーンで**意図しない変形が起こる**

ローカル軸の方向を破棄した際に起きる問題

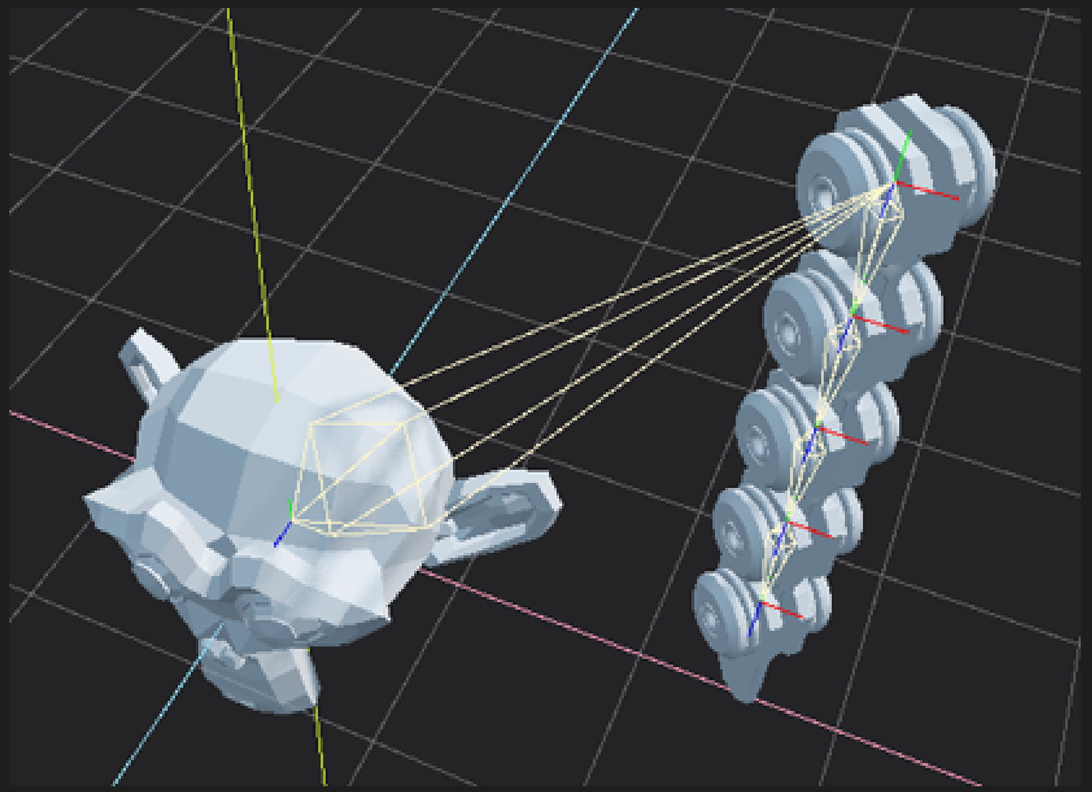
VRMSpringBone・VRMConstraint で回転方向の制御が不可能

例えば DynamicBone であれば**回転方向の制限**が可能であり、
下図のような場合、 FreezeAxes に X を指定すれば誤った回転を防止できる
ヒンジジョイント



ローカル軸の方向を破棄した際に起きる問題

VRMSpringBone・VRMConstraint で回転方向の制御が不可能



しかし VRM のように
正規化でローカル軸の方向を破棄すると
X・Y・Z のような軸指定の制御は不可能

VRM 内部の SpringBone・Constaint で
回転方向の制御ができないだけではなく

DynamicBone・FinallK のような
ローカル軸を使うプラグインが
正常に動かなくなる

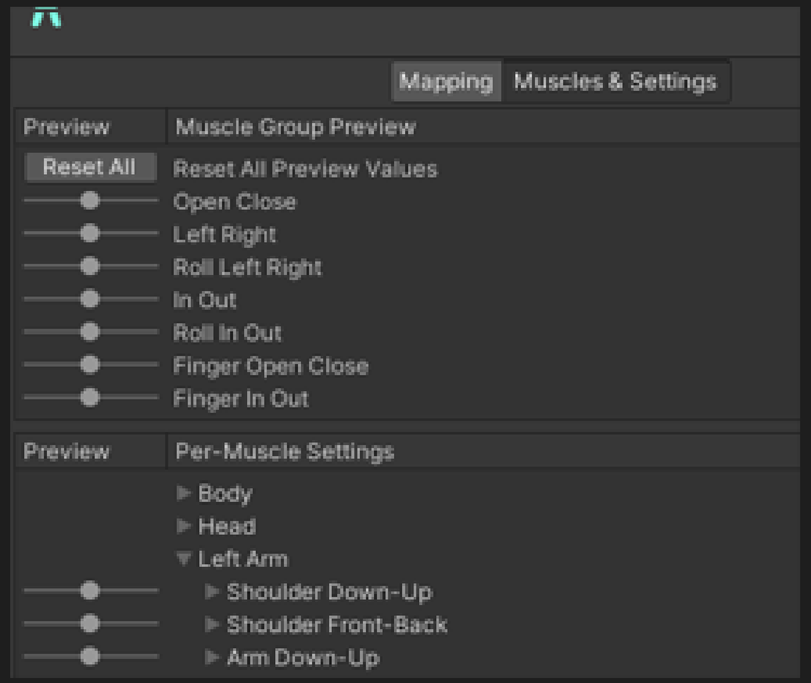
ローカル軸の方向が破棄されているのに

Unity や VRM Addon はどうやって制御してるの？

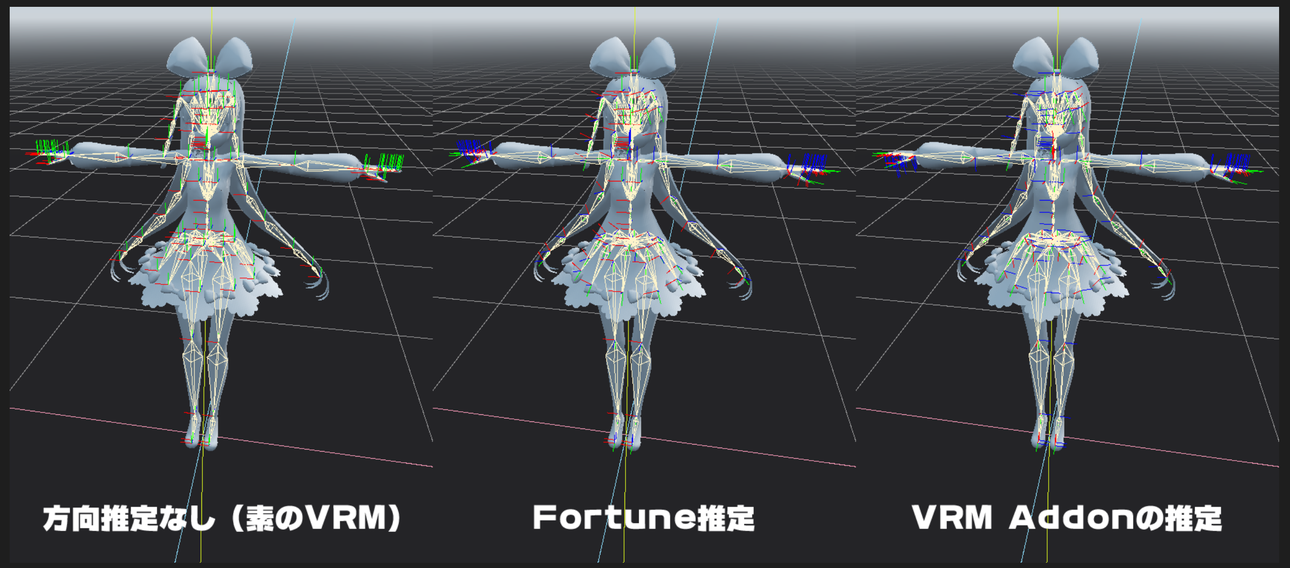
Unity や VRM Addon はどうやって制御してるの？

ローカル軸の方向がないモデルをなんとか動かすための２つの技術

Unity（リターゲット）



VRM Addon（方向推定）



※実は VRM Addon で Blender に読み込んだモデルは
ボーンが推定後の向きになっているので VRM の定義を満たしてない

ローカル軸の書き換えや推定でモデルのローカル軸の方向に関係なく制御する
→ VRM 正規化の影響を無視して制御している

リターゲットや方向推定があれば問題ないのか？

「Unity Humanoid で親指が歪む問題」に代表するように
リターゲット・方向推定後のローカル軸は元のローカル軸と異なるので
正規化前の状態はどうやっても再現できない

故に、

「斜めのボーンの回転軸が分からない」
「ローカル軸を使うプラグインが壊れる」
これらの問題について**ローカル軸を保持する以外に解決方法はなく**
リターゲットや方向推定ではどうにもならない

リターゲットや方向推定があれば問題ないのか？

ローカル軸問題への対策として手動で再設定させる事を考えているらしいが…

ボーン回転バックアップ、ボーン回転復旧機能 #731

Open　ousttrue opened this issue on 9 Feb · 0 comments

ousttrue commented on 9 Feb　Contribu

正規化する前の回転を維持したいという需要があるので、
全てのTransformの回転をテキストにバックアップしておいて
モデルロード後に手動で元の回転に復旧する機能を作成する。

- 二度手間
- データの量が増える
- 開発時の前提条件がブレる

再設定は根本的な解決方法ではない

https://github.com/vrm-c/UniVRM/issues/731

現状の正規化による問題点の整理

- リターゲットシステムのない環境で**ポージングやアニメーションの作成が困難**
- ボーンを曲げる方向が正確に分からず、**デザイナーの意図した変形ができない**
- 揺れものや IK、コンストレイント等における**回転軸の制限が困難**
- 結局ローカル軸の方向が必要になり**方向推定や再設定を行う二度手間が発生**

正規化のせいで
脱 Unity や機能追加が難しくなっているのではないか？

ローカル軸に関わる問題を解決する為の

新たな定義と正規化の実装方法を考えよう

回転軸の役割を優先した定義の提案

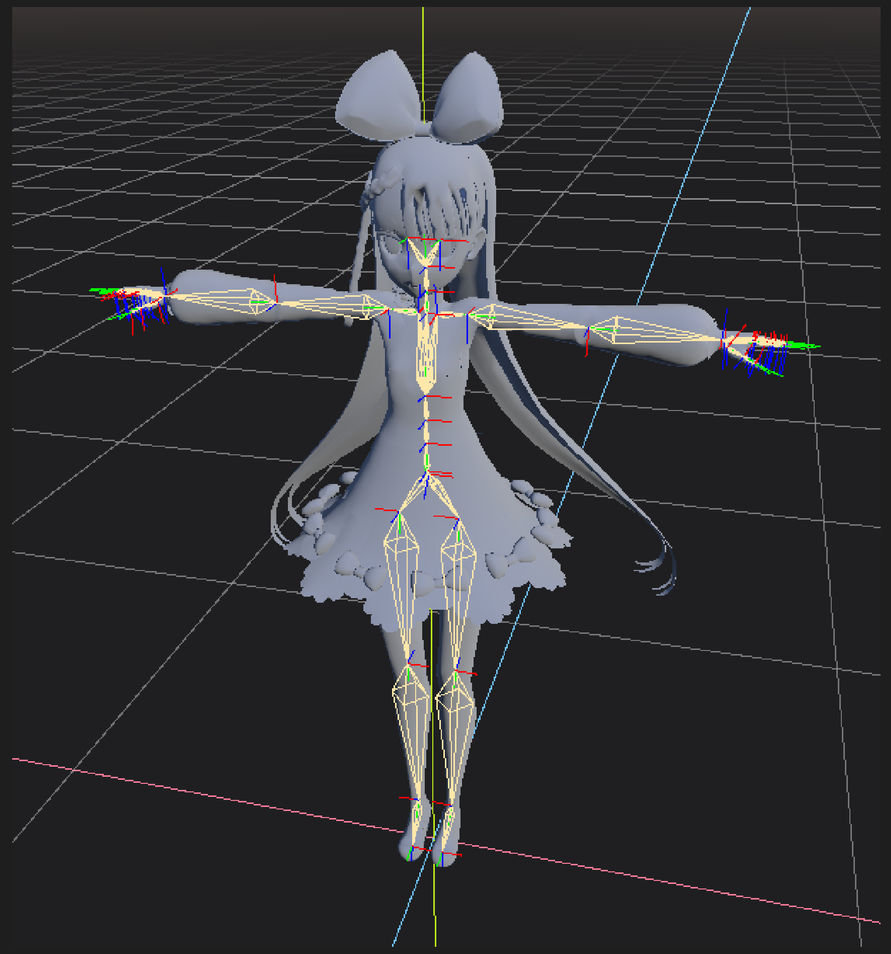
回転軸の役割を優先した定義の提案

まず、関節に対する回転の操作を「屈伸」「捻り」「開閉」の３つと考える

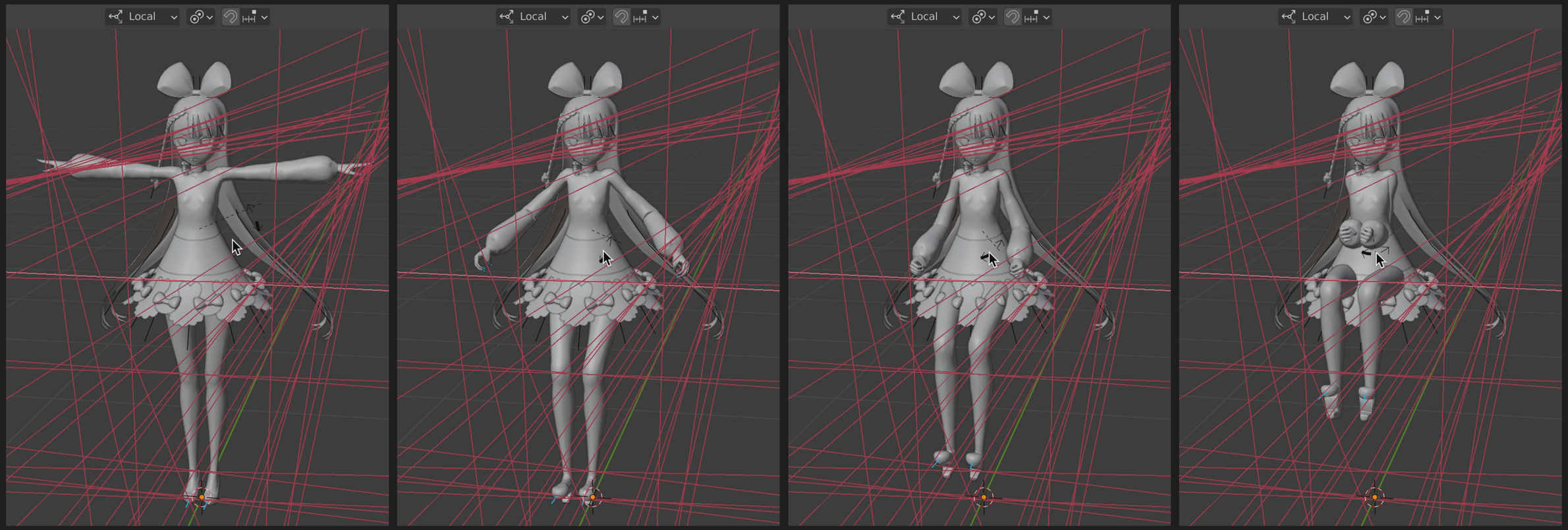
定義の考え方

- SpringBone や Blender に合わせて
  ロール軸方向は親から子へ向かい +Y
- 曲げ回転（筋肉が縮まる時の回転）を
  左半身右半身とも +X で統一
- スケール 1.0 の右手直交系という前提で
  ２軸が決まると Z 軸が自動的に決まる

結果



回転軸の役割を優先した定義の提案



全てのボーンで +X 回転を行うと、胎児のように縮こまるポーズになる
（腰や肩は大きく動かすと潰れるのでとりあえず手足のみ操作した画像）

回転軸の役割を優先した定義の提案

新：提案する定義（推奨値は一例）

ボーン名	グローバルレスト回転（推奨値）	屈伸	捻り	開閉
Spine	Basis $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	X 軸	Y 軸	Z 軸
LeftUpperArm	Basis $\begin{pmatrix} 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix}$	X 軸	Y 軸	Z 軸
LeftThumbProximal	Basis $\begin{pmatrix} 0.707 & 0.5 & -0.5 \\ 0.707 & -0.5 & 0.5 \\ 0 & -0.707 & -0.707 \end{pmatrix}$	X 軸	Y 軸	Z 軸

※直感的な規定変換行列での定義を推奨

レスト回転は**推奨値**であり
強制するものではない

アバターとしては
ボーンの回転操作の役割が
曖昧な事が問題なので
回転軸の役割を優先すべし

旧：VRM 0.x の定義

ボーン名	グローバルレスト回転（強制）	屈伸	捻り	開閉
Spine	Quat(0,0,0,1)	X 軸?	Y 軸?	Z 軸?
LeftUpperArm	Quat(0,0,0,1)	Z 軸?	X 軸?	Y 軸?
LeftThumbProximal	Quat(0,0,0,1)	XY 軸?	XZ 軸?	YZ 軸?

VRM 0.x の定義では
レスト回転を統一した結果、
斜めのボーンに関して
回転軸が決まらない

情報を破棄しない正規化実装の提案

情報を破棄しない正規化実装の提案

一般的な編集ソフト上で再現できる定義であれば
リギングのガイドラインや**サンプルプロジェクト**を用意できる
（VRM 0.x の定義ではガイドラインがあってもその通りリギングできない）

正規化という**技術者しか理解できないような仕様**によって
意図しない変形をされたり、プラグインが正常に動作しないような
「**ユーザーの分からない所で何かが起こる**」ことよりも
「書いてある通りの曲げ方ができるようにしてね」
という**ガイドラインを用意して従ってもらう**方が
より良いユーザーエクスペリエンスを提供できるのではないか

情報を破棄しない正規化実装の提案

とはいえ数値を用いた機械的な補助は必要

ボーン名	グローバルレスト回転(推奨値)	屈伸	捻り	開閉
LeftUpperArm	Basis $\begin{pmatrix} 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix}$	X 軸	Y 軸	Z 軸

推奨値を定義しておけば推奨値との差分を計算する事ができるので、
差分の大きさによって「推奨範囲外」という警告を出して修正を促す事は可能

既存の回転値を推奨値へ強制するリターゲットも**補助として**存在するなら有用

念を押すが、定義するのはあくまで推奨値であり、
デザイナーが意図的に推奨範囲外の値を設定することは十分考えられるので
警告は出しても強制すべきではない

ローカル軸の定義と正規化実装を改善した際のメリット

デザイナー

- 多くの編集ソフト上で定義をそのまま利用できる
- リギングの指標が分かりやすくなる
- 意図した変形をアバター利用者へ伝える事ができる
- アニメーションの作成や共有がリターゲットがなくても容易になる

プログラマー

- リターゲットや方向推定が必須ではなくなる
- ローカル軸を参照した移動や回転方向の制御が可能
- ボーン削減による負荷軽減 ※VRMSpringBone 末端の方向推定用ボーンが不要になる
- ボーンを制御するパラメータの実装が容易になる

定義をきちんと行いローカル軸を保持した時のメリットは大きい

まとめ

ボーンの方向を揃える必要性 → 分かる
元の情報を破棄してボーンの方向を揃える → やりすぎ

編集のための情報は確かに必要ないが…

現在の VRM は
アバターをアバターとしてアニメーションさせる為に必要な情報まで破棄してしまっている

ボーンの方向を揃えるにしてもいくつかやり方がある
「必要な情報まで失わない」正規化実装を行うべし

VRM 1.0 は VRM 0.x との互換性を考えなくてもよい滅多にない機会

ここでローカル軸の方向の定義と正規化処理の実装を見直す事で

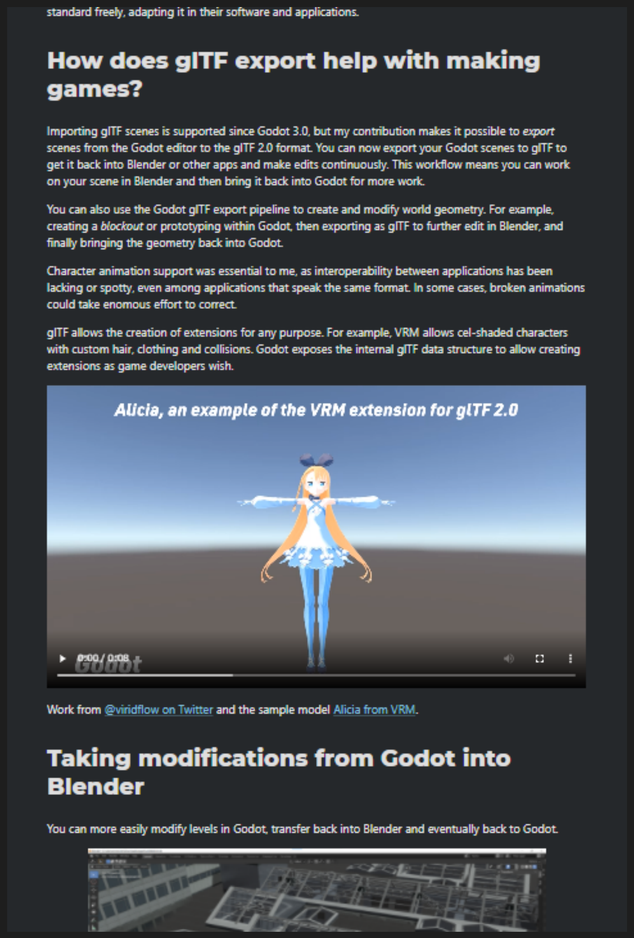
プラットフォームを超えた汎用アバターフォーマットとして

あらゆる glTF をサポートするプラットフォームへ進出するという

スタートラインに立てるのではないでしょうか

宣伝

Godot Engine



standard freely, adapting it in their software and applications.

How does glTF export help with making games?

Importing glTF scenes is supported since Godot 3.0, but my contribution makes it possible to *export* scenes from the Godot editor to the glTF 2.0 format. You can now export your Godot scenes to glTF to get it back into Blender or other apps and make edits continuously. This workflow means you can work on your scene in Blender and then bring it back into Godot for more work.

You can also use the Godot glTF export pipeline to create and modify world geometry. For example, creating a *blockout* or prototyping within Godot, then exporting as glTF to further edit in Blender, and finally bringing the geometry back into Godot.

Character animation support was essential to me, as interoperability between applications has been lacking or spotty, even among applications that speak the same format. In some cases, broken animations could take enomous effort to correct.

glTF allows the creation of extensions for any purpose. For example, VRM allows cel-shaded characters with custom hair, clothing and collisions. Godot exposes the internal glTF data structure to allow creating extensions as game developers wish.

Alicia, an example of the VRM extension for glTF 2.0

Work from @viridflow on Twitter and the sample model Alicia from VRM.

Taking modifications from Godot into Blender

You can more easily modify levels in Godot, transfer back into Blender and eventually back to Godot.

実は MToon や VRMSpringBone を含め
VRM を Godot で利用できるようにする
アドオンを有志のメンバーで開発中

Godot の公式メンバーで glTF パイプラインの
コアコントリビューターである
Ernest Lee 氏もアドオンの開発に関わっており
Godot 公式ブログにも VRM の事が載っている
（Godot が VRM を公式サポートする訳ではないので注意）

ただし、
今回説明した正規化の問題のせいで雲行きが良くない

https://godotengine.org/article/introducing-the-godot-gltf-2-0-scene-exporter

宣伝

V-Sekai

※指〜〜〜…

海外発の開発中 Godot 製オープンソース VR メタバース（VRM 対応予定）

https://v-sekai.org/

謝辞

情報提供や調査協力、ファクトチェックを行って頂きました
以下の方々に感謝致します

- Ernest Lee (Godot Member, Core Contributor 3D pipeline of Godot)
- Saracen (V-Sekai)
- Lyuma (V-Sekai)
- lox9973 (ShaderMotion)
- Silent (Starlit Wanderings, Silent's Cel Shading Shader)
- fuka mizno (sirokiri)

VRM で Unity プラグインが正常に動かなくなった

Unity 外の環境で VRM が使いづらくて困っている

VRM が使える場面をもっと増やしたい

脱 Unity を目指している

これらに該当するあなた！

記事の方を共有してくださると助かります！

https://qiita.com/TokageItLab/items/e5880123a9f508b2769d

ご静聴ありがとうございました！