CEDEC2015
Computer Entertainment Developers Conference

SQUARE ENIX

Session #67 @R502
Aug 26, 2015 17:50 – 18:50

FINAL FANTASY XV –EPISODE DUSCAE– のアニメーション ～ 接地感向上のためのとりくみ ～

株式会社 スクウェア・エニックス

第2ビジネス ディビジョン　今村 紀之 (Noriyuki IMAMURA)
テクノロジー推進部　川地 克明 (Katsuaki KAWACHI)

LUMINOUS STUDIO

Business Division 2

Advanced Technology Division

SQUARE ENIX

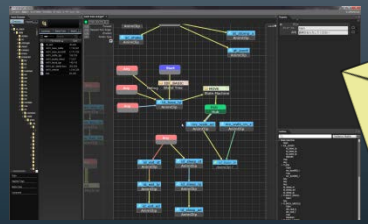
FINAL FANTASY XV – EPISODE DUSCAE –

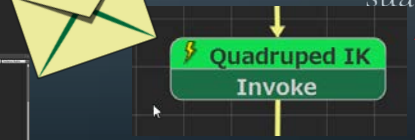
Agenda

Introduction

LUMINOUS STUDIO

Data Driven



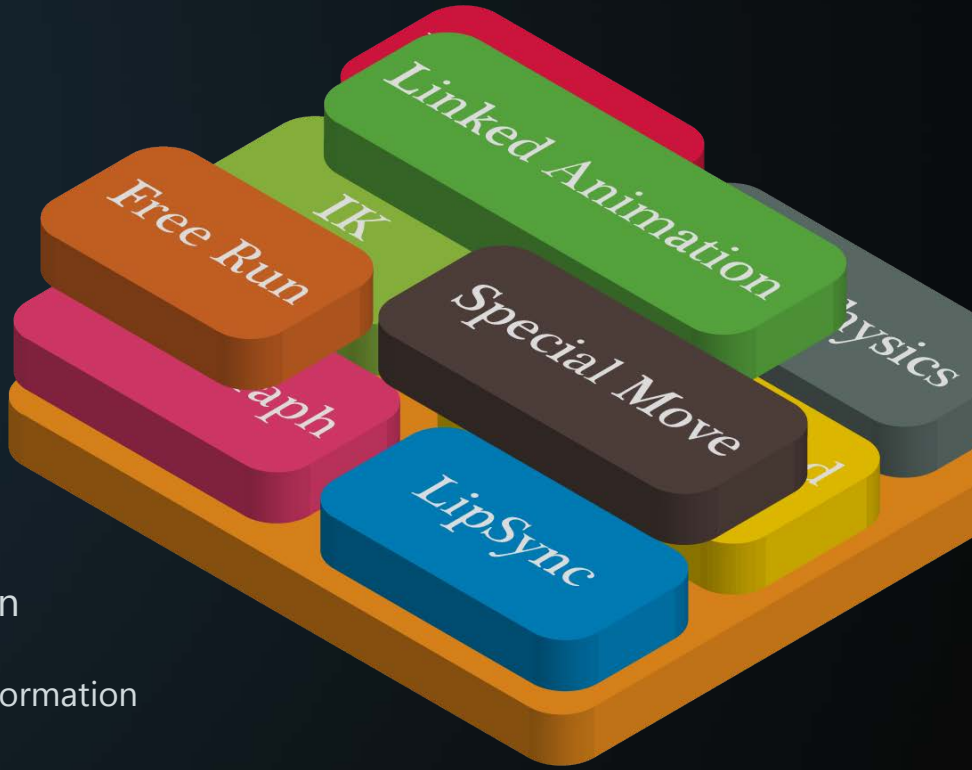


Procedural

Luminous Animation

Luminous Animation の機能

- ✓ Animation Core
  - Playback
  - Trigger
  - Layered Animation
  - Additive Animation
  - Mirror Animation
  - Instance Scale
  - Animation Space Blending
  - 1D Blend
  - Locomotion Blend
- ✓ AnimGraph
  - State Machine
- ✓ Lip-Sync
- ✓ Linked Animation
- ✓ Free Run
- ✓ Special Move
  - Rail Move
  - Point Move
- ✓ IK
  - Biped
  - Quadruped
  - Look-At
  - Aim-At
  - Tail & Neck
- ✓ Physics-based Animation
- ✓ KineDriver
- ✓ Cloth Simulation
- ✓ Facial
  - Pose Space Deformation

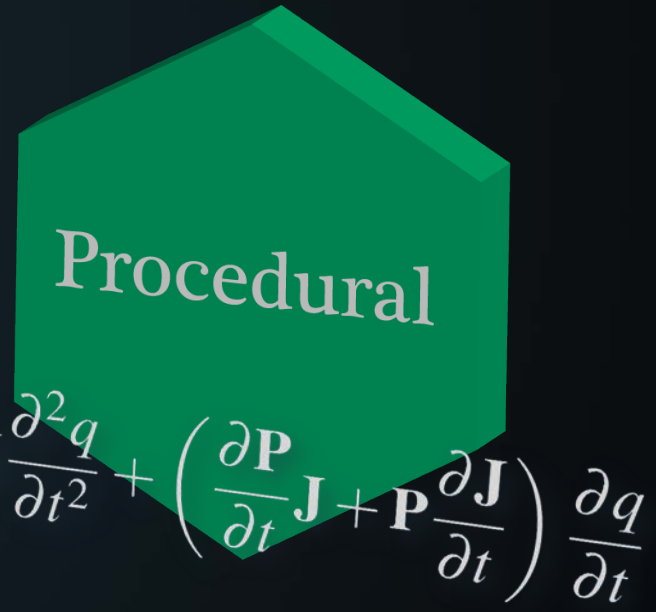


LUMINOUS ANIMATION

FFXV 体験版のアニメーション

接地感

接地感を高めるためのエンジニアリング手法 (1/3)

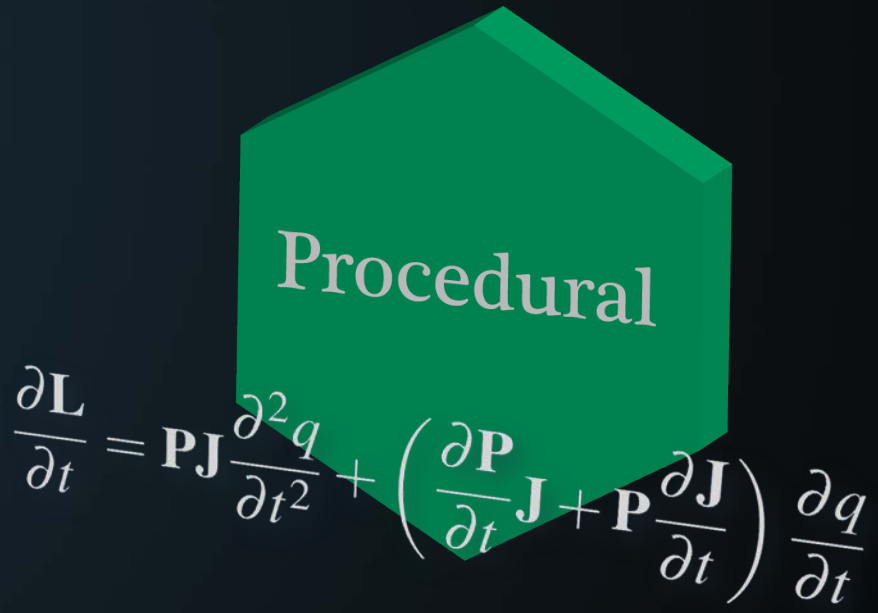


接地感を高めるためのエンジニアリング手法 (2/3)

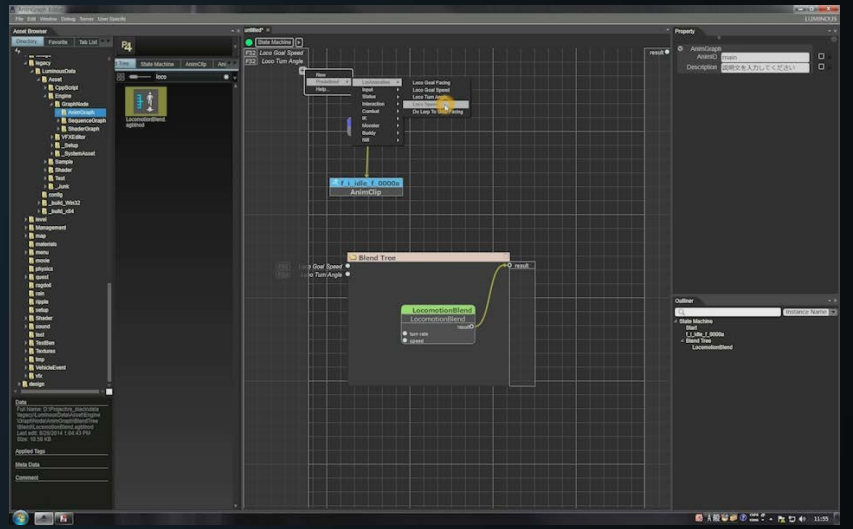
- データを量産し
  ゲームに組み込んで
  いくための
  ワークフロー効率化

接地感を高めるためのエンジニアリング手法 (3/3)

- 環境に応じた反応を付加する技術
  - 不整地
  - ヒットリアクション



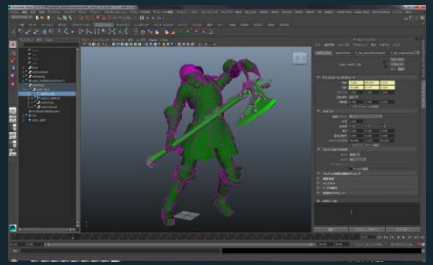
Luminous Animation Tools



Animation Organizer

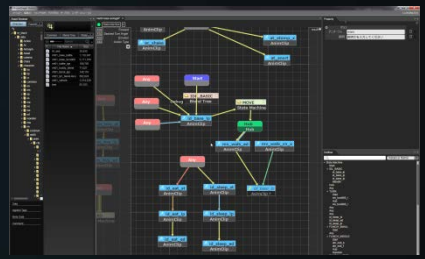
AnimGraph Editor

ワークフロー



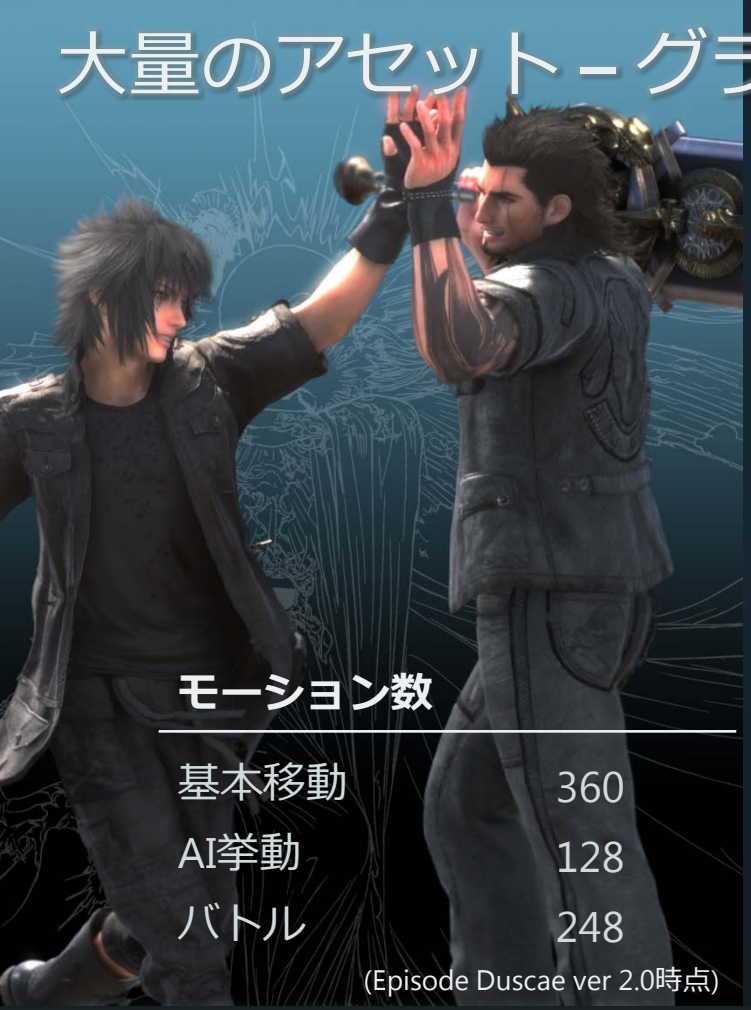
エクスポート

確認



組み込み

大量のアセット – グラディオの例



モーション数

基本移動	360
AI挙動	128
バトル	248

(Episode Duscae ver 2.0時点)

- 移動をともなう行動だけで9種類

通常	バトル状態	警戒
しゃがみ	瀕死	草掻き分け
斜面のぼり	斜面くだり	ストレイフ

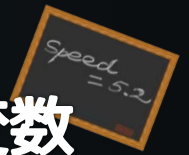
- 仲間は歩き出しや歩き止りのパターンが豊富

ゲーム実装とのやり取り

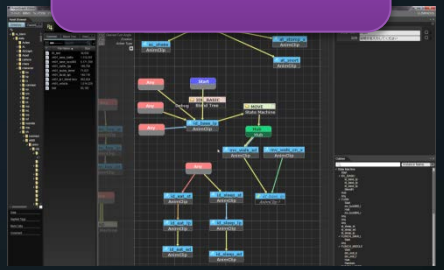
AIなど

メッセージ

ブラックボード変数



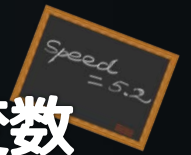
AnimGraph



ゲーム実装とのやり取り

AIなど

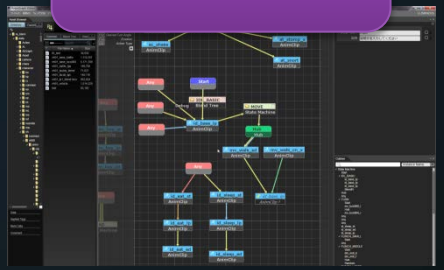
メッセージ



ブラックボード変数

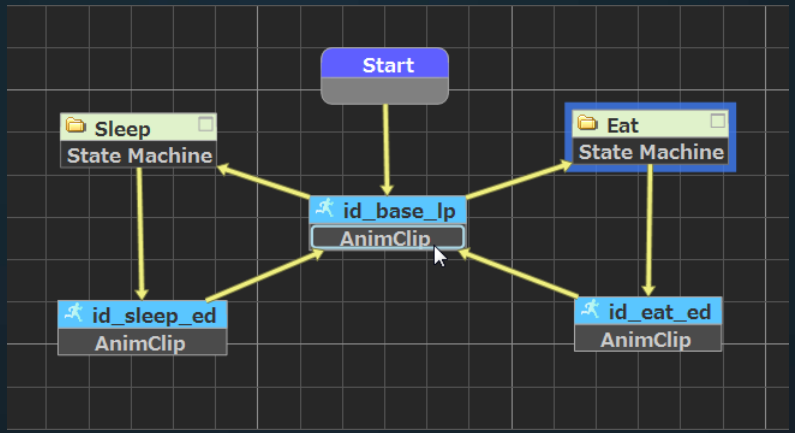
アタック！

AnimGraph

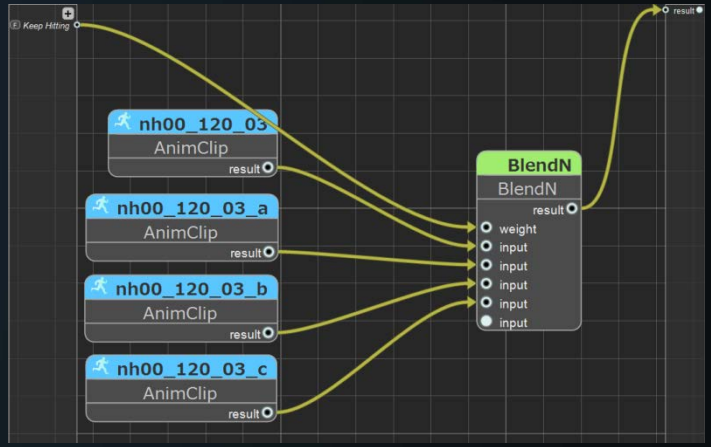


速さはこれぐらい

AnimGraph – State MachineとBlend Tree (1/2)

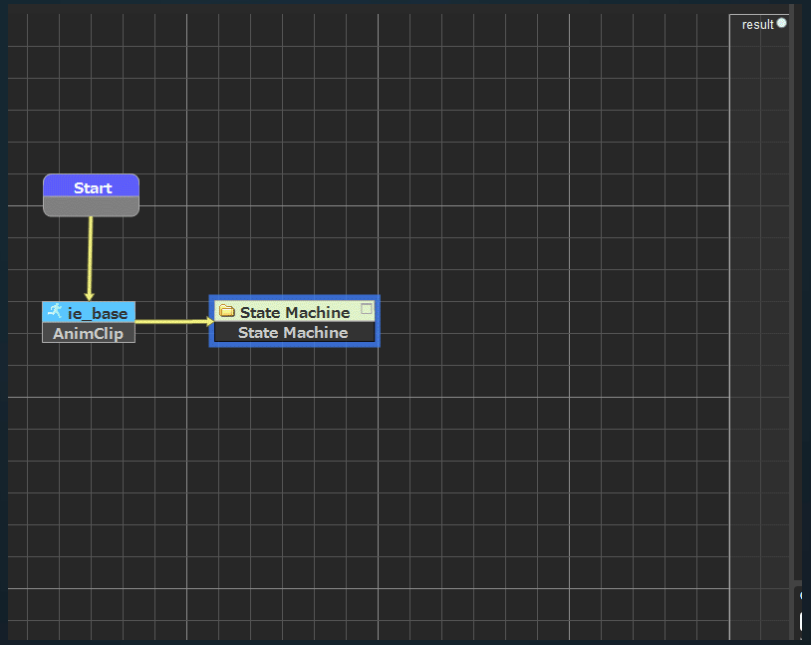


State Machine



Blend Tree

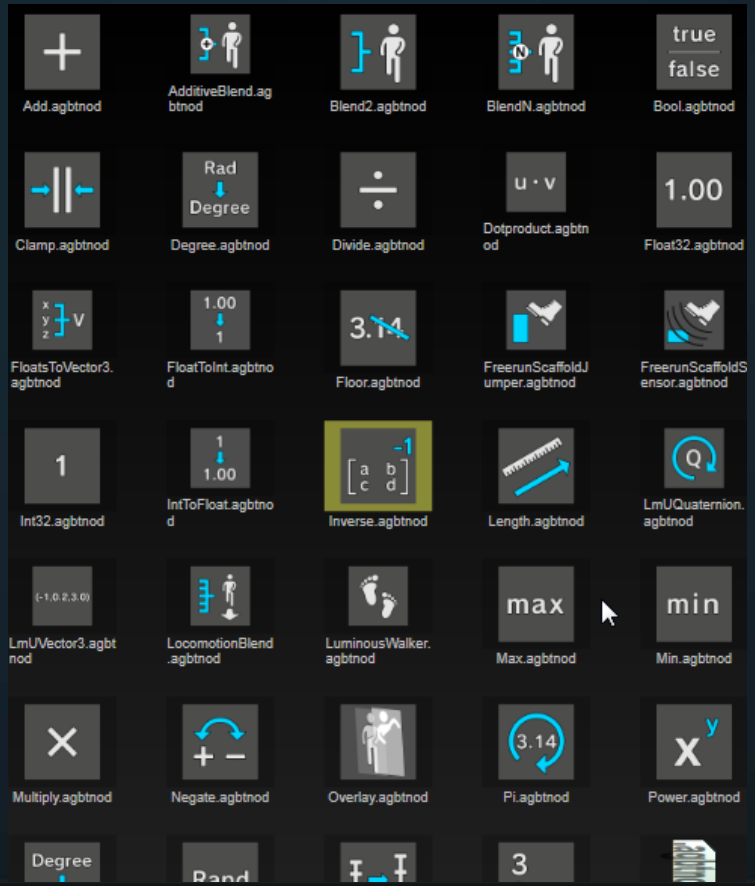
AnimGraph – State MachineとBlend Tree (2/2)



階層化に対応

- State MachineとBlend Treeを互いに入れ子構造にできる

AnimGraph – Blend Tree

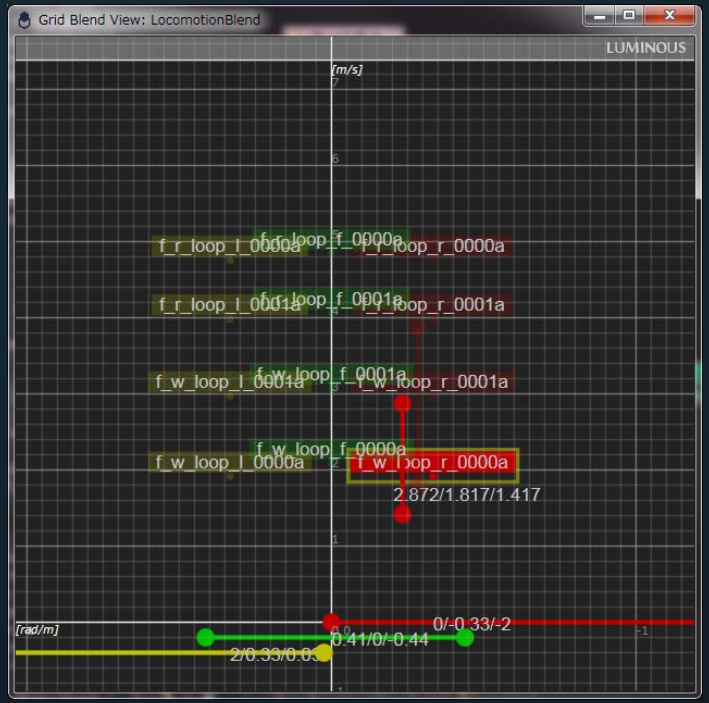


ブレンドノード

- Additive Blend
- Overlay, etc.

その他算術系が多数

AnimGraph – Blend Tree – Locomotion



Locomotion

- 移動専用のノードがあり、Grid上で編集する
- 速度性能やターン性能が設定できる

AnimGraph Editorを制作するにあたって

Q. AnimGraphを組むのは誰?

AnimGraph Editorを制作するにあたって

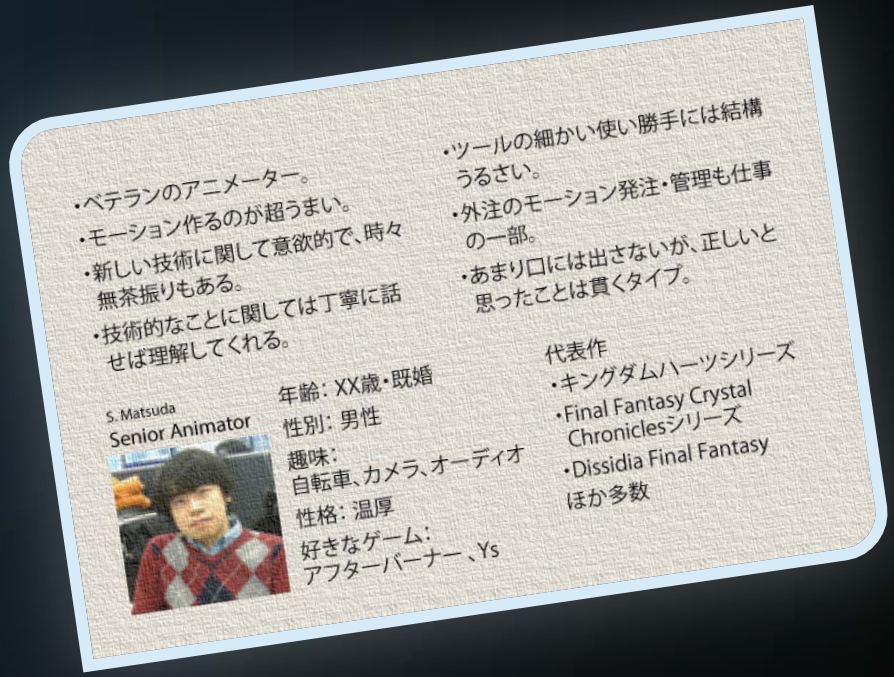
Q. AnimGraphを組むのは誰?

A. アニメーター

ゲームエンジンのツール制作

ペルソナ法

- 仕様決定のブレをなくし、ユーザー目線のツールを作るために用いるマーケティング手法。
- 「30代男性」のような幅を持ったユーザー像ではなく、より具体性を持ったユーザー像を定義する。



アニメーターが考えるべきでないこと

計算
順序

計算順序の決定が難しい例

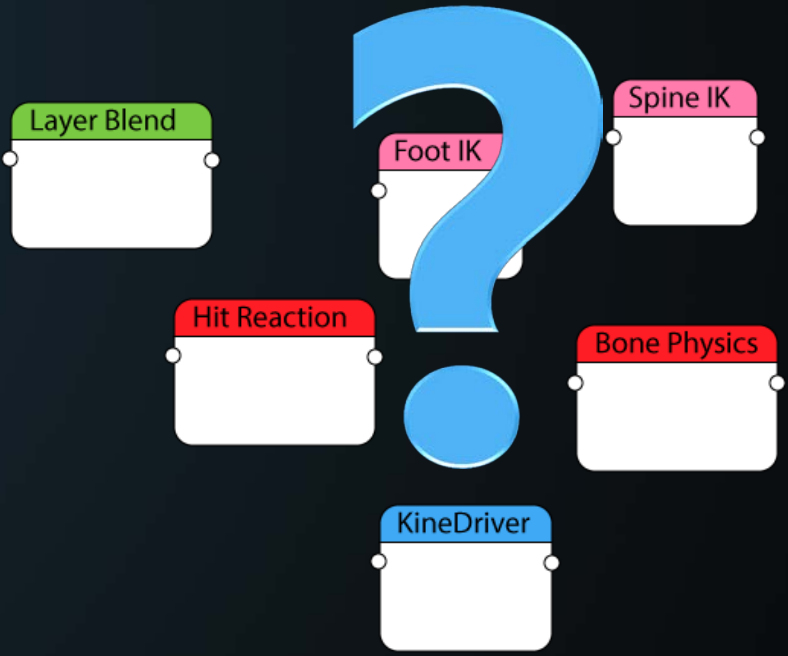
Q. 以下の要素を見た目が破綻しないように正しい処理順に並び替えなさい。

- IKにより背骨を補正して特定の方向をAimする
- IKによる地形への接地補正
- 骨物理の計算
- 補助骨の計算
- 物理ベースのヒットリアクション
- フェイシャル アニメーションの合成

計算順序の決定が難しい例

Q. 以下の要素を見た目が破綻しないように正しい処理順に並び替えなさい。

- IKにより背骨を補正して特定の方向をAimする
- IKによる地形への接地補正
- 骨物理の計算
- 補助骨の計算
- 物理ベースのヒットリアクション
- フェイシャル アニメーションの合成



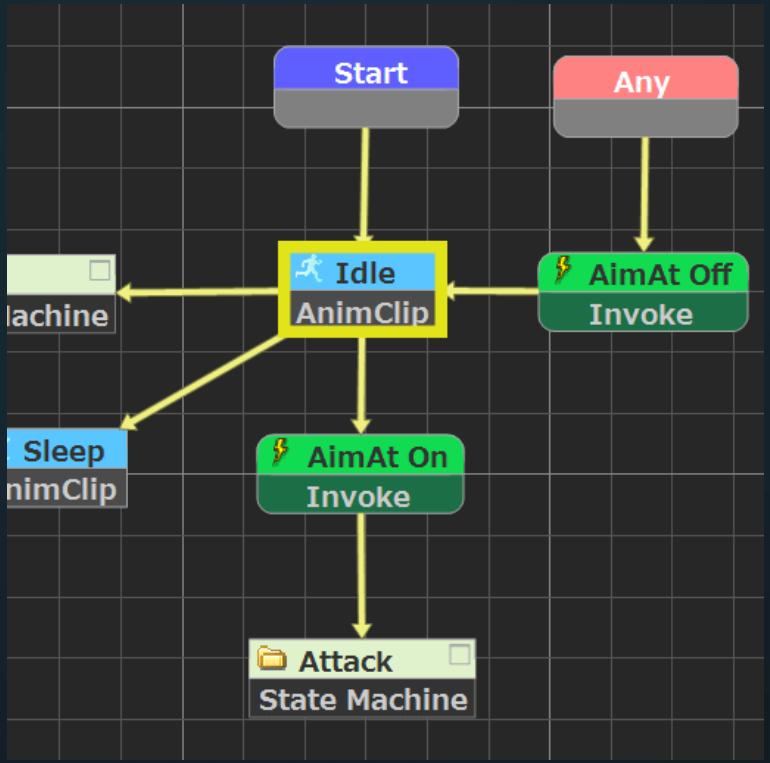
計算順序の決定が難しい例

Q. 以下の要素を見た目が破綻しないように正しい処理順に並び替えなさい。

- IKにより背骨を補正して特定の方向をAimする
- IKによる
- 物
- ースのヒットリアクション
- フェイシャル アニメーションの合成

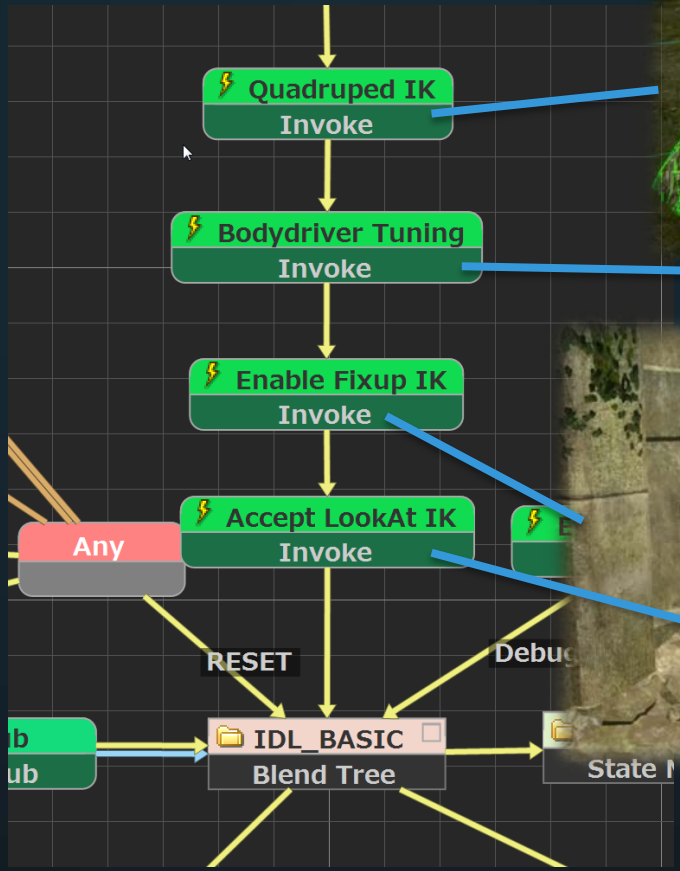
アニメーターがやりたいのは、
ON/OFFの制御とパラメーターの調整だけ

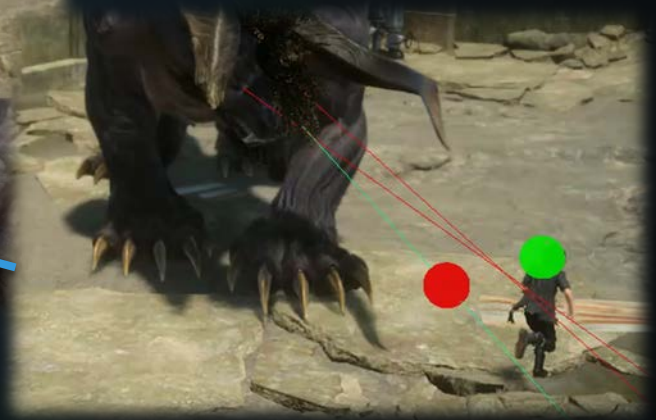
Invoke



- State Machineに Invokeという ノードが置ける
- Invoke を通ったときに 特定の機能 (関数) が 呼び出される

Invokeの例





Tuning Set

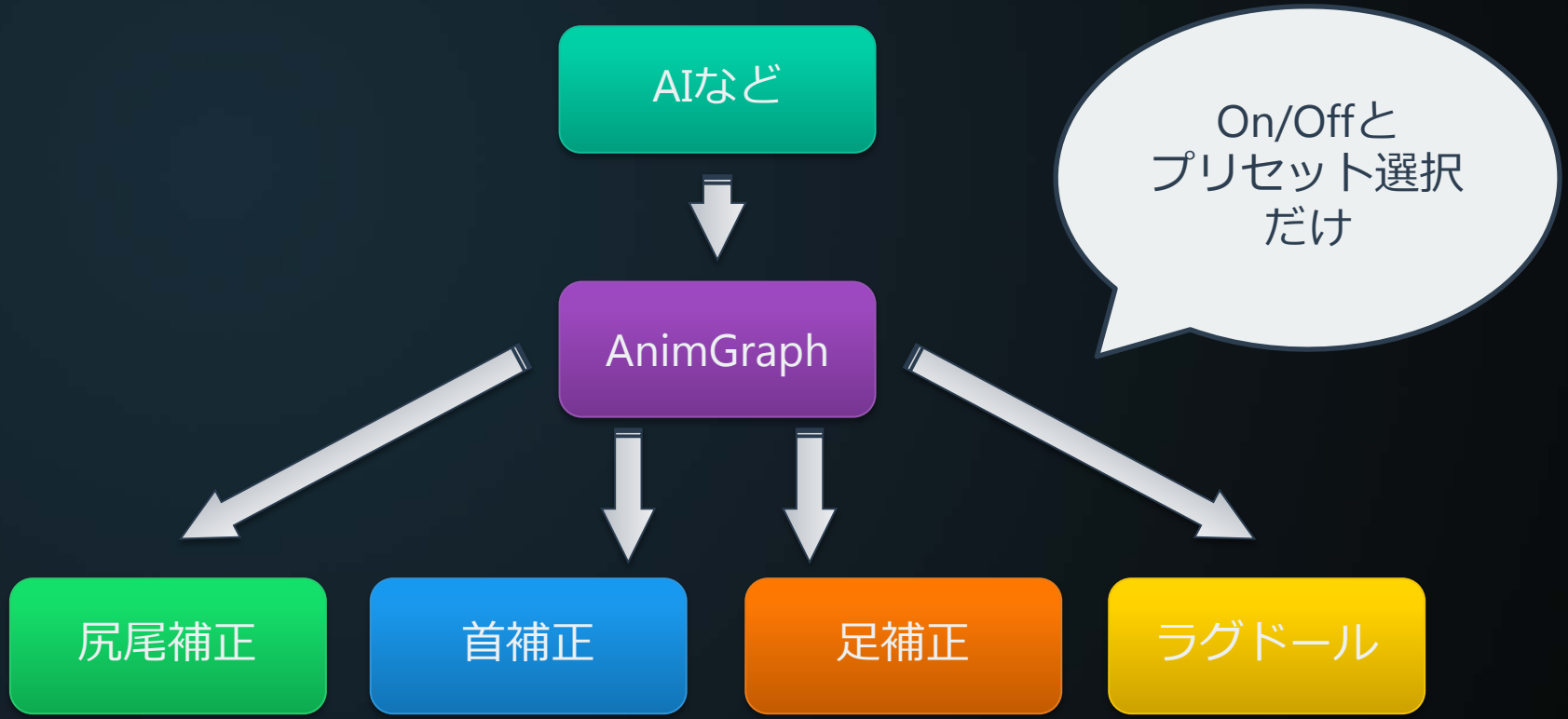
- 各モデル (骨構造) ごとの
  各種パラメータ設定
- TAがプリセットを作成する

- アニメーターは作成された
  プリセットから選ぶだけ。
  - 人型の立ち状態の視線補正
  - ニフル兵の走り状態の
    クイックな銃構えの補正
  - NPC用の腰から上を
    フルに使った背骨の方向補正
  - etc.

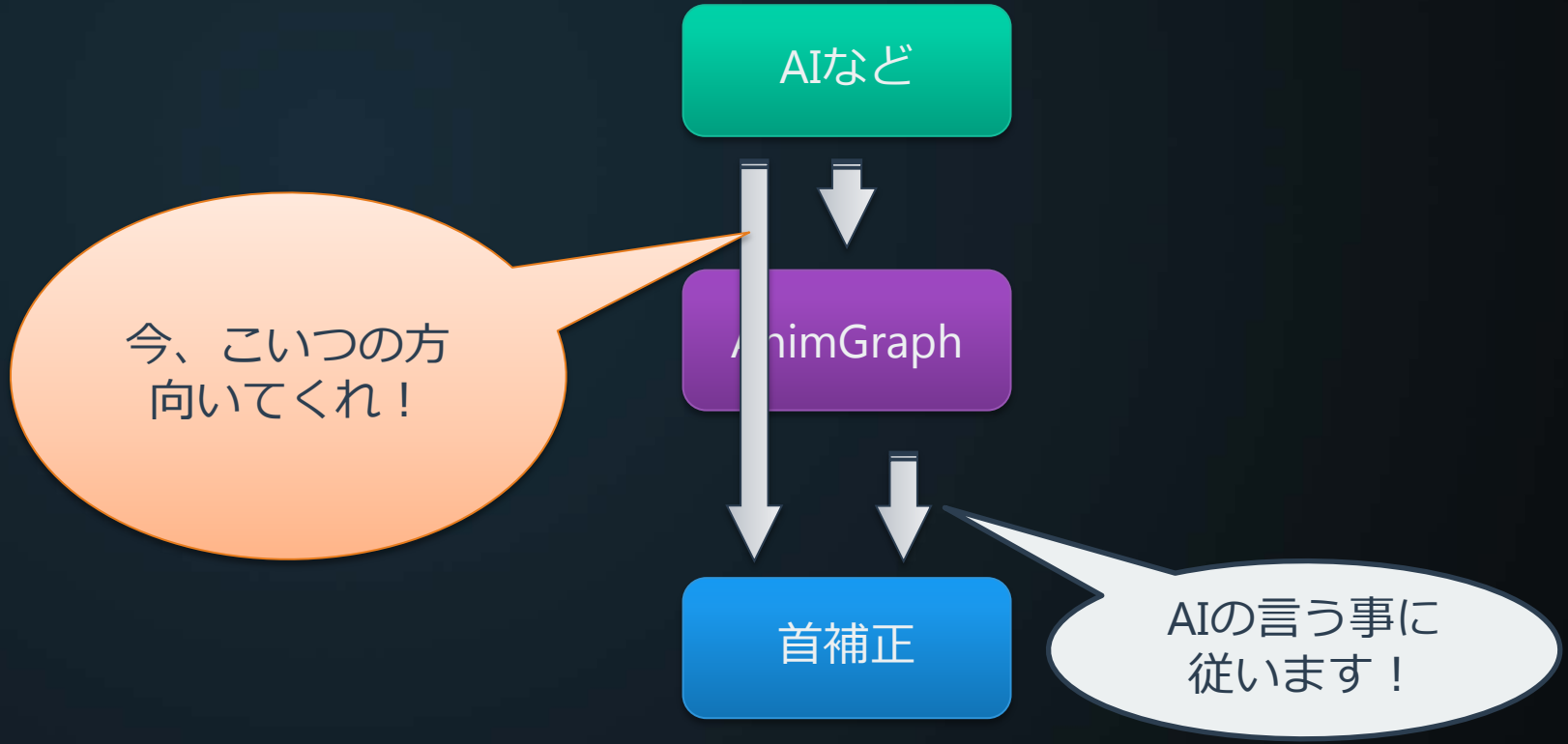
Aim At IKのParameter例

使用する骨	ウェイト
ジョイントの種類	可動範囲の設定
最大速度	最大加速度
最大減速度	BlendOutの速度
リコイルの強さ	リコイルの分散

処理の流れのイメージ



実際は...



実際は...

AIなど

AnimGraph

首補正

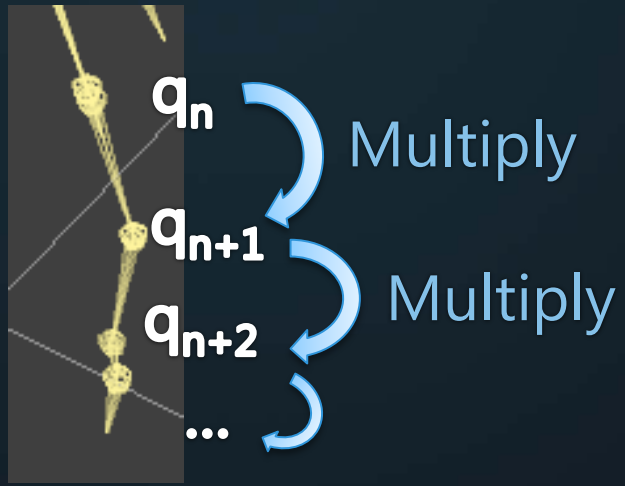
今、こいつの方
向いてくれ！

AIの言う事に
従います！

どうしても無効にしたいところだけ、
範囲型のTriggerで設定

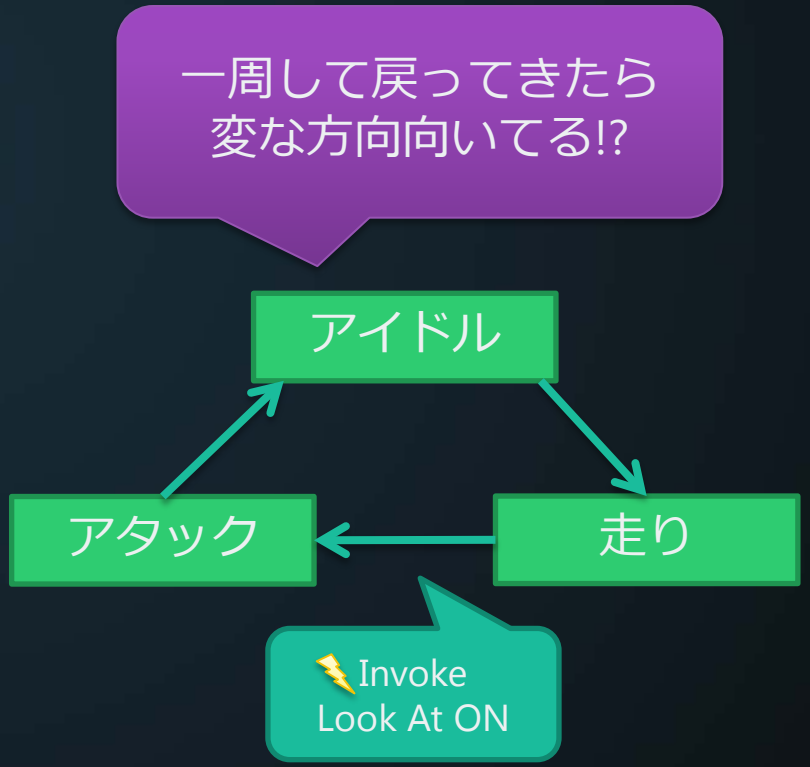
Invokeによる設定のメリット – 最適化

ボトルネックになりがちな
Joint Local 座標系から Global 座標系への
変換のタイミングを制御・抑制できる

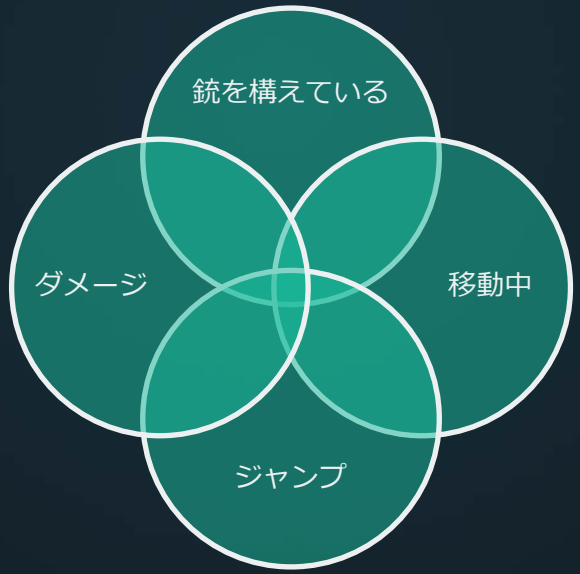


マルチスレッドでの同期を限定できる

Invokeの特徴 – カプセル化とノンホロノミック性 (1/2)



Invokeの特徴 – カプセル化とノンホロノミック性 (2/2)



身体状態の特徴は特定のステートに
カプセル化しづらい

ながら動作

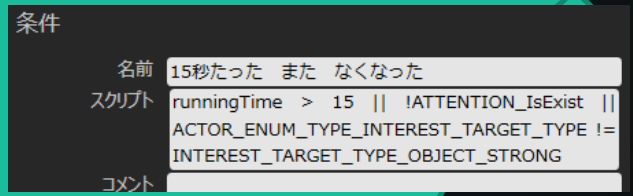
移動 → 構え → ジャンプ → ダメージ → 停止 → 構え解除

アニメーターの仕事

AnimGraph Editorは
アニメーター向けに作られている

たとえば、

AnimGraph



AI Graph

アニメーターの役割

AnimGraph Editorは
アニメーター向けに作られているとはいえ、必要知識や担当範囲は増加

~~アニメーターの役割~~ テクニカルアニメーターの役割

AnimGraph Editorは
アニメーター向けに作られているとはいえ、必要知識や担当範囲は増加

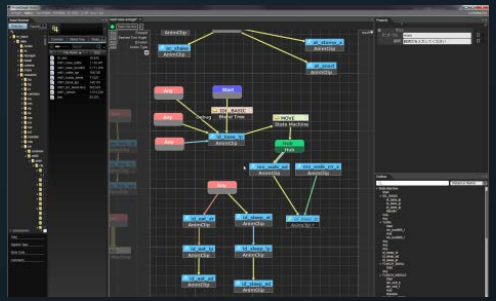
デメリットも

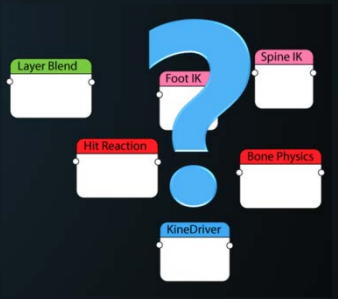
- Data Drivenのデメリット
  - データが不正なだけで簡単に「バグ」る
  - ほかの人がデータを見たときに
    ぱっと見で分からないことがある

- ツールを特定の人向けに特化するデメリット
  - いざというときにツールの使い方が分からない
  - 人が足りないときに困る

Data Driven – まとめ

- キャプチャからゲームへの組み込みまでの
  縦方向ワークフローの効率化
- プログラマ・アニメーター・TAの担当範囲を再定義
- 担当者を意識して、ツールや実装方法をデザイン

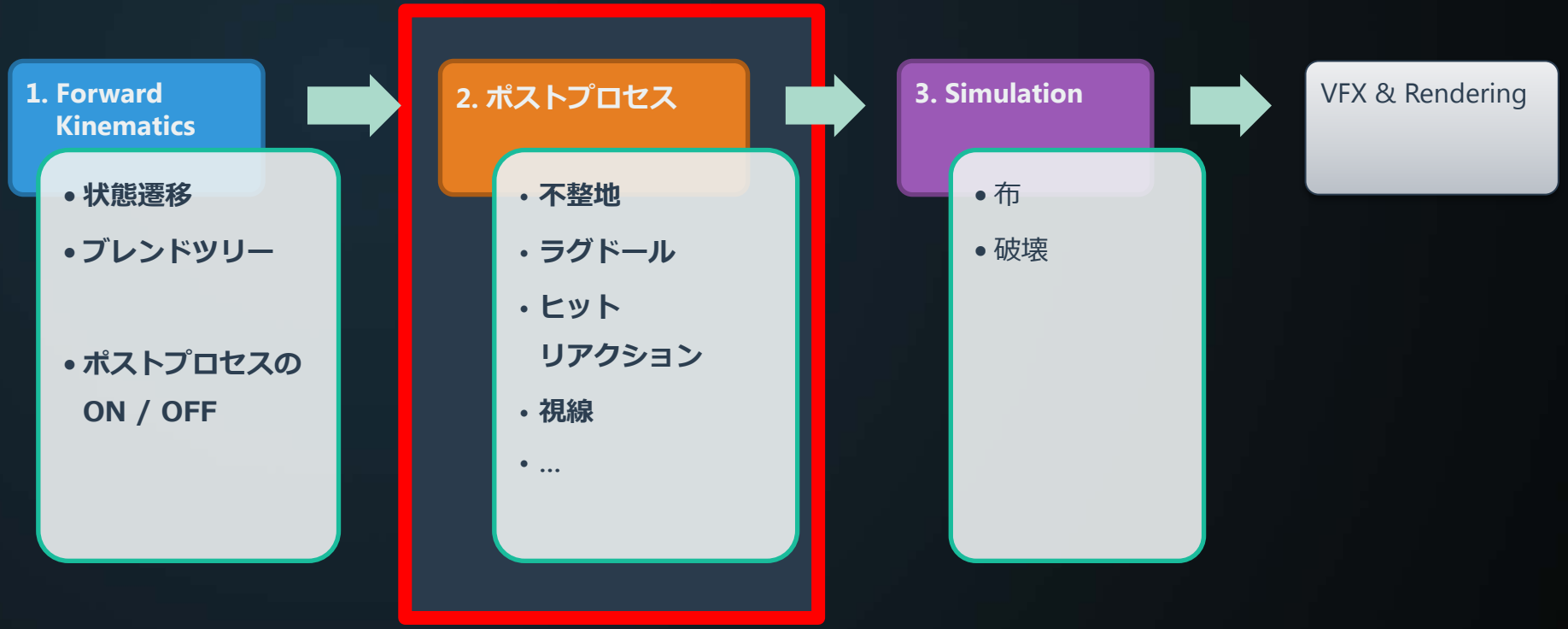




Procedural Animation

プロシージャルアニメーション

Luminous Animation の処理フロー



FINAL FANTASY XV 体験版: ボス戦

キャラクタの配置

平坦でもなく広くもないフィールド

たとえば: 壁とモンスター

尻尾と首へのポストプロセス (1/3)

- 実装
  - 尻尾の動きはアニメーションクリップに追従
  - 尻尾が干渉しそうな地形を検出
  - 地形と干渉しそうな時には、押し返す力を発生

Postprocess OFF

Postprocess ON

尻尾と首へのポストプロセス (2/3)

– ジョイントの移動速度にもとづいて地形との干渉を予測

尻尾と首へのポストプロセス (3/3)

- 結果
  - 自然地形・狭隘空間に自動的に対応できた
  - アニメーションクリップの動きを活用し、
    PD制御による追従によって自然な動きを生成できた

- 課題
  - 地形が極端な凹凸を持たないように調整する必要がある
  - 地形との干渉状態によっては、
    CCD-IK によるポストプロセス計算量が多くなる

モンスター: 視線 (1/2)

- 目的

  視線を送ることで、
  モンスターがプレイヤーを認識していることを伝えたい

モンスター: 視線 (2/2)

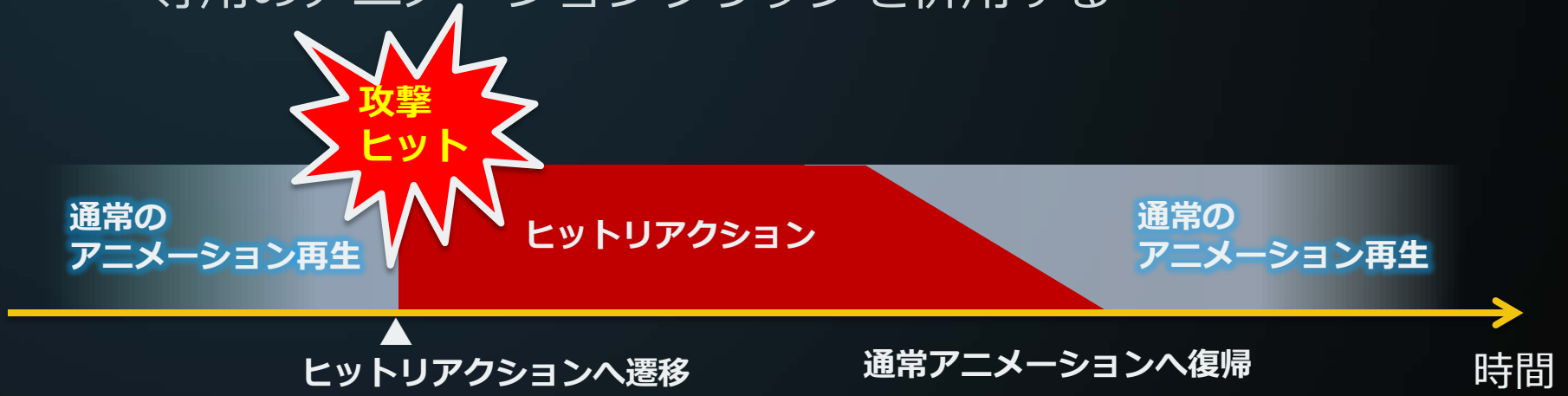
- 実装
  - 視線変更に関与するジョイントにあたえるパラメータ
    - 姿勢変更運動への寄与率
    - 角度制限
    - 運動可能な軸方向
    - 運動のスムージング率
  - 可視・不可視領域は自動設定
    - 各関節の角度制限を利用

モンスター: ヒットリアクション (1/5)

- 目的
  プレイヤーから攻撃を受けたことを、体の動きとして表現する

モンスター: ヒットリアクション (2/5)

- 要件
  - どのような運動状態にあっても攻撃をヒットさせ、リアクションを出す
  - ダメージが大きい時は、専用のアニメーションクリップを併用する



モンスター: ヒットリアクション (3/5)

- 実装
  - 攻撃ダメージを外力としてモデル化
  - ジョイントで接続された剛体として骨構造をモデル化
  - 力学的運動シミュレーションを用いて
    ダメージによるリアクションの動きを生成

モンスター: ヒットリアクション (4/5)

- 高速化のための工夫
  - 接地している足先は地面から離れないと仮定
- 安定化のための工夫
  - 力学シミュレーションはゲームループから独立させ、時間刻みを細かくする
    - ゲームループ: 30Hz
    - シミュレーションループ: 60Hz ～ 90Hz

モンスター: ヒットリアクション (5/5)

- 結果
  - 多方向からの攻撃に対する
    ヒットリアクションの
    自動生成
- 課題
  - より長時間の影響を残す
    ヒットリアクションの生成
  - ダメージによる
    足の踏み変えへの拡張

モンスター: 運動と慣性力 (1/3)

- 目的
  - モンスターの重量感を伝えるために、地面を蹴ることが運動を変化させていることを強調したい

モンスター: 運動と慣性力 (2/3)

- 実装
  - 走行中は、全身をカーブの内側に傾ける
    - 頭と尻尾は元の姿勢を維持する
  - 足の位置が着地点からすべらないように、できるだけ固定する

モンスター: 運動と慣性力 (3/3)

- 結果と課題
  - 効果的だが、ゆっくり歩くときには足滑りがめだつ場合がある

プレイヤー: 足裏の接地 (1/4)

- 目的
  - 凹凸のある地面に対して、足がめりこまないようにしたい
  - 斜面や段差の上で、自然な立ち姿勢を実現したい

プレイヤー: 足裏の接地 (2/4)

- 実装
  - 検知した地面に足裏を沿わせる

プレイヤー: 足裏の接地 (3/4)

– 谷側の足に重心を乗せることで、自然な安定感を生成する

OFF

足裏のみ

足裏 + 重心

プレイヤー: 足裏の接地 (4/4)

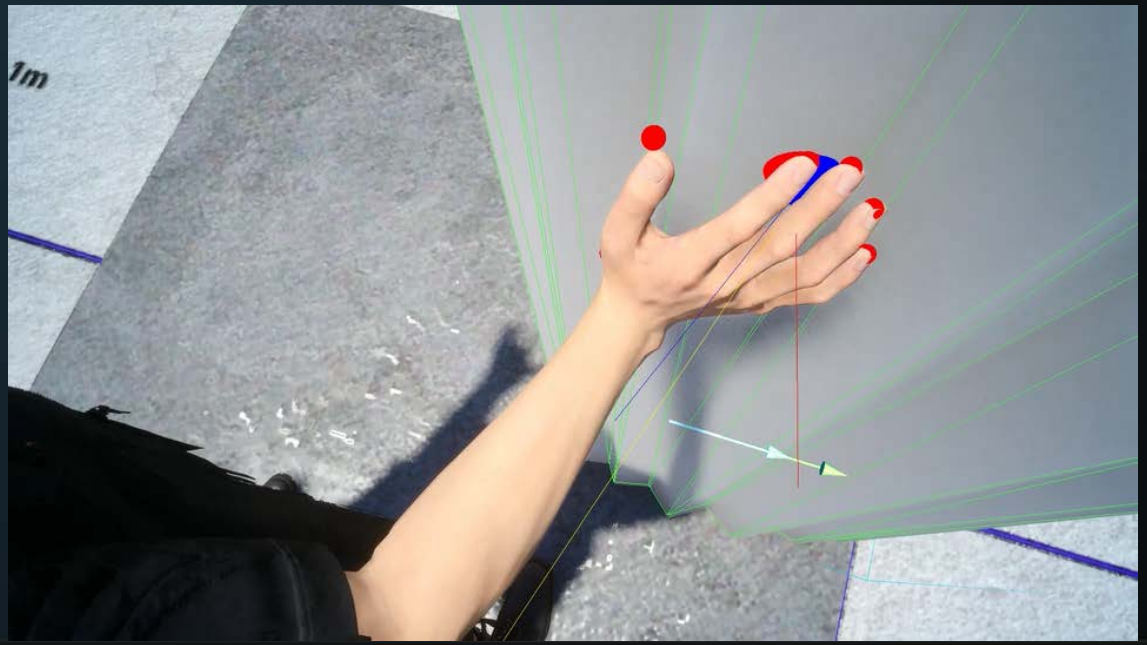
- 結果
  - 体重移動の実装により
    立位姿勢がより安定して見える
- 課題
  - 移動開始時に、
    不自然なすべりや浮きが
    残る場合がある

プレイヤー: 壁と手の接触 (1/3)

- 目的
  - 壁に手を添えることで、移動空間の狭さを伝えたい

プレイヤー: 壁と手の接触 (2/3)

- 実装
  - 目標を検出し、その位置に手先を沿わせる
  - 目標を検出するタイミングは、アニメーションクリップ内に定義する



プレイヤー: 壁と手の接触 (3/3)

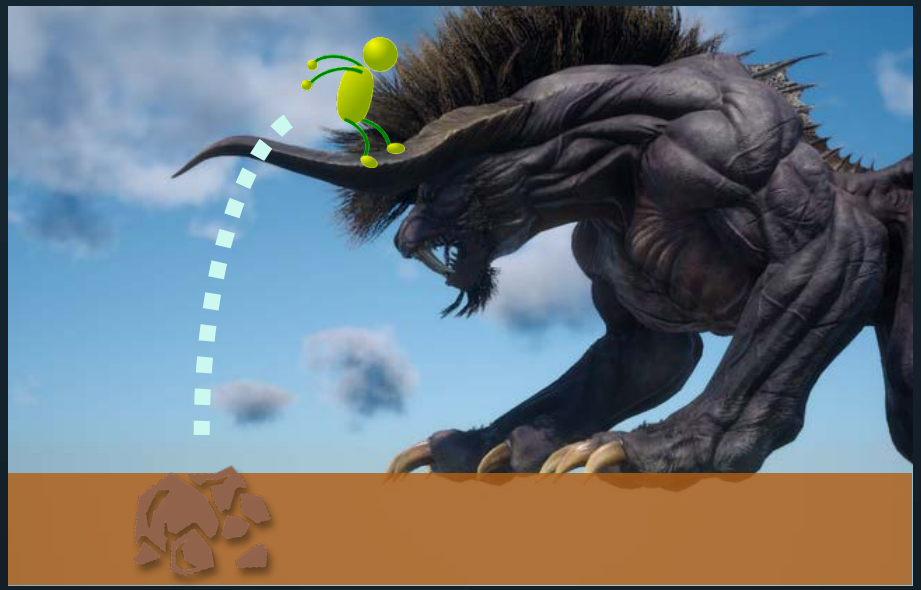
- 結果
  - 手を壁にそって送る動きを自然に表現できた
  - 凹凸のある壁や、洞窟のカーブに自動対応できた

プレイヤーとモンスター (1/3)

- 目的
  - プレイヤーとモンスターを、意図した位置関係で接触させたい

プレイヤーとモンスター (2/3)

- モンスターの姿勢は地面の傾きに応じて変化する
- モンスターの動的な姿勢変化に対応し、プレイヤーの軌道を補正する



無補正のアニメーション軌道

姿勢変化に対応する軌道補正

プレイヤーとモンスター (3/3)

1. Forward Kinematics
   - アニメーションの軌道を補正
2. Postprocessing
   - 斜面上の姿勢計算
3. Simulation

モンスターの姿勢を予測

アニメーションのポストプロセス – まとめ

- 大型モンスターの接地感を高めるポストプロセス
  - 地形に対する尻尾・首の調整
  - 視線の制御
  - ヒットリアクションの生成
  - 走行中の傾き

- プレイヤーの接地感を高めるポストプロセス
  - 重心調整
  - 壁と手の接触
  - アニメーション軌道の補正

Q & A