3D キャラクターリアルタイムモーションシステムの開発

三浦 彰人

2018 年 9 月 23 日

概要

VR デバイスやモーションキャプチャ技術の普及にともない，3DCG とその関連技術を表現手法として用いるものが一般的になりつつある．しかし，これらの技術を用いたアプリケーション開発においては，開発環境の構築やカスタマイズが困難であり，また，要求されるハードウェア性能が高いことが多い．そこで本研究では，一般的なノート PC の性能でも利用可能な 3D キャラクリアルタイムターモーションシステムを開発し，オープンソースソフトウェアとして公開した．

1 背景

2010 年代に入り，3DCG とモーションキャプチャ技術を用いた表現手法が多様化し，2016 年ごろから，Oculus Rift をはじめとした VR デバイスが急激に普及し始めた．他にも，スマートフォンアプリに AR が多用されるなど，これからの表現手法として 3DCG はより身近で欠かせない存在となりつつある．

また，筆者の所属している東北公益文科大学が，文部科学省の私立大学研究ブランディング事業に採択され[*1]，事業のテーマとして，「日本遺産を誇る山形県庄内地方を基盤とした地域文化と IT 技術の融合による伝承環境研究の展開」が掲げられている．その中で，モーションキャプチャや CG アニメーション，VR などの IT 技術を用いた地域文化の活用・デジタルアーカイブ化が推進されており，これらの技術に対する知見の集約が不可欠である．本事業においては，学生も交えた研究プロジェクトの推進が必要であるが，学生が 3DCG 技術に関する知識・技術を身に付けられる環境が乏しい．そこで，学生の所有しているノート PC のような環境でも 3DCG に関連する技術に触れ学ぶことのできるシステムが必要と考えた．

2 既存システムと課題

モーションキャプチャと 3DCG を組み合わせたリアルタイムキャラクターモーションシステムの実装として，FaceRig などが有名である．他にも，音声認識システムと音声合成システムなどを組み合わせ，3D キャラクターを用いた音声インタラクションシステムの事例として，名古屋工業大学国際音声技術研究所によって開発された MMDAgent [4] などがあげられる．

また，モーションキャプチャシステム単体としては，一般的なカメラと画像認識を活用したもの，Kinect や LeapMotion といった専用ハードウェアを用いた画像処理でおこなうもの，Perception Neuron のようなモーションセンサーを利用するものなどがある．

これらの既存システムにおいては，ソースコードが公開されていないソフトウェアや開発キットが必要であったり，特定のプラットフォームに強く依存していたり，追加ハードウェアの購入が必要であったりすることが多く，環境構築やカスタマイズにおいて障害となる．

そこで，モーションキャプチャと 3DCG を組み合わせたキャラクターモーションシステムを，Web カメラ付きノートパソコンのみで開発・利用できるシンプルなシステムを開発し，オープンソースソフトウェアとして公開する．これにより，大きな追加投資や複雑な環境構築作業をおこなうことなく，手軽にモーションキャプチャによる 3D キャラクターモーションに触れ学ぶ機会を容易に提供できるようにすることを目標とする．

3 システム要件

ノート PC という限られた性能と，学生などが利用・カスタマイズすることを想定し，シンプルでオープンソースソフトウェアを主体としたシステムとする．

画像収集，モーションキャプチャ，モデルモーション生成・配信，モデル操作をそれぞれ独立したモジュールとする．これらを切り離すことで，モデルやモーションキャプチャシステムの入れ替えなどに柔軟に対応できるようにする．また，それぞれが独立して非同期処理をおこなうことで，最良のパフォーマンスを発揮できるようにする．

さらに，各モジュール間はネットワーク経由での通信をサポートする．これにより，複数の機器の協調動作システム

[*1] 文部科学省．平成 29 年度「私立大学研究ブランディング事業」選定事業一覧．http://www.mext.go.jp/a_menu/koutou/shinkou/07021403/002/002/1398494.htm.

の開発を容易にする．また，機能拡張時にモーションキャプチャデバイスなどのプラットフォーム制約に可能な限り縛られないようになる．

各モジュールの機能と要件は次の通りである．

3.1 3D モデルモジュールの要件

- 3D モデルの動きと表情，カメラの動きを表現できるようにする
- 動作時に 30fps 程度を保つ

3.2 3D モデル操作モジュールの要件

- 3D モデルの動きと表情，カメラを操作できるようにする
- ネットワーク経由でのモデル操作をサポートする

3.3 リアルタイム画像収集モジュールの要件

- 3D キャラクターの目の代わりとなり，リアルタイムに画像を収集する
- 撮影フレームレートは 30fps 程度を保つ

3.4 モーションキャプチャモジュールの要件

- 画像に映る 1 人以上の人間の顔の位置をキャプチャする
- 特殊な機材を必要とせず，Web カメラと PC のみでおこなえるようにする
- モーションキャプチャレートは 30fps 程度を保つ

3.5 3D モデルモーション生成・配信モジュールの要件

- 3D モデルの動きと表情を生成・配信できるようにする
- モーションキャプチャ結果を元にモーションを自動生成する

これらのモジュールを結合すると図 1 のようになる．

4 システム設計と実装

4.1 3D モデルモジュール

3D モデルの作成には Blender を用いる．Blender は，オープンソースの 3DCG ソフトウェアで，3D モデリングの他，アニメーションやゲーム制作などもおこなうことが可能である．Blender を用いて作成する 3D モデルは本システムの見た目の中核となるものである．3D モデルの頂点数が増えれば増えるほど動作は遅くなり，ユーザからは不自然な動作に見える．また，ボーンの数増加についても同様に動作は遅くなり，オブジェクトに対するモーションのウエイトの設定などが複雑になる．今回は，頂点数は 10 万未満とし，頭と胴，腕，足が最低限動くようにする．さらに，テクスチャ張替により表情を表現できるモデル構成とする．カメラについては，全方向から 3D モデルを参照できるよう，球面座標系を用いた移動を可能とする．

4.2 3D モデル操作モジュール

3D モデルの操作・描画には Blender Game Engine [1] を用いる．Blender Game Engine を用いることで，モデル作成からリアルタイムモーションまで，一括しておこなうことができる．Blender Game Engine は 3D モデルのオブジェクトに対しセンサー，コントローラ，アクチュエータを定義することで，3D モデルの動きを表現する．それぞれの役割は下記の通りである．

- センサー: キーボードやジョイスティックなどの入力に従って発動する
- コントローラ: センサーの入力を元に，アクチュエータを発動する
- アクチュエータ: オブジェクトに何らかの動きを与える

Blender Game Engine では，Python スクリプトを用いてコントローラを定義し，アクチュエータの代わりにオブジェクトを動かすといったことが可能である．本システムでは，この Python スクリプトを用いたコントローラで 3D モデル操作をおこなう．

しかし，Python スクリプトによる操作は全てシングルスレッドでの処理となる．そのため，レスポンスが遅い処理をおこなうと，全体のパフォーマンスに大きな影響が出る．計算コストの高い処理や外部からの入力を待つ処理が存在すると，それらの処理が終了するまで，Blender Game Engine の描画処理が全て停止する．また，Blender Game Engine の API を利用しない範囲で threading などのライブラリを用い，スレッドによるバックグラウンド処理をおこなうことは可能であるが，メインの描画処理が優先されるため，バックグラウンドのスレッドのパフォーマンスは著しく低下する．よって，Blender Game Engine 内では最小限の処理のみをおこなうよう設計する必要がある．

そこで，モーションデータの生成までを外部モジュールでおこない，Blender Game Engine によるモデル操作モジュール内では，モーションデータの受け取りとモーションの適用のみをおこなう．

4.3 リアルタイム画像収集モジュール

画像処理によるモーションキャプチャをおこなう場合，Web カメラなどから一定のフレームレートで画像を取得する必要がある．リアルタイム画像収集モジュールでは，オープンソースのマルチメディアフレームワークである GStreamer[2] を用いて，可能な限り最新の画像ファイルを取得し続ける処理をおこなう．Gstreamer では，Web カメラから取得した動画ストリームをフレームごとに静止画像ファイルに変換し保存する．また，最新の画像のみをモーションキャプチャ処理に利用するシステムであるため，取得した画像は一定期間保持したのち自動的に削除する．

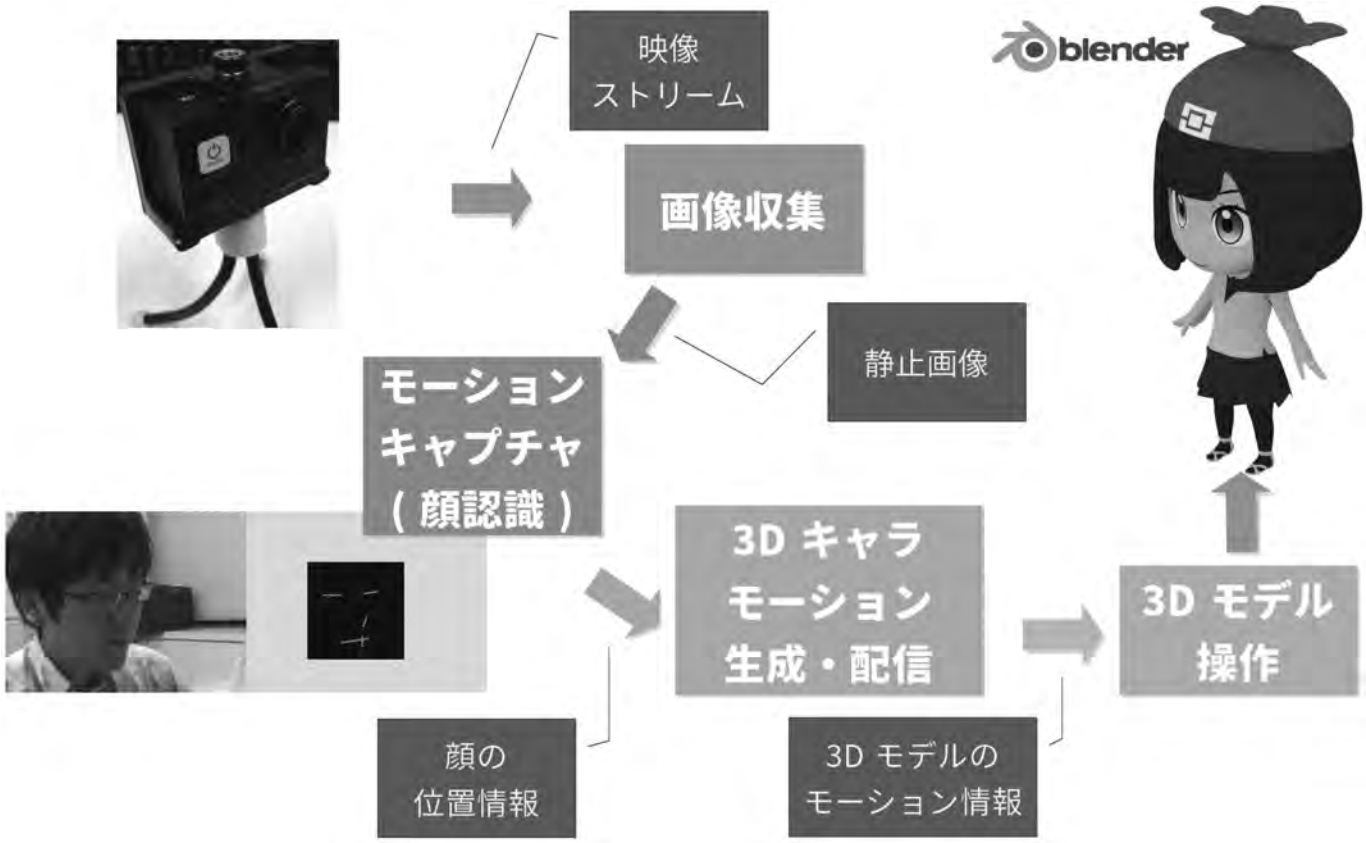

図 1　システム概要図

4.4　モーションキャプチャモジュール

モーションキャプチャモジュールでは，リアルタイム画像収集モジュールで収集した画像から顔パーツ位置情報を取得し，3D モデルモーション生成・配信モジュールに送信する必要がある．顔パーツ位置情報の取得には dlib [3] を用いる．dlib は，現実世界の問題を解決する複雑なソフトウェアを作成するための機械学習アルゴリズムとツールを提供するライブラリである．本システムでは dlib と学習済みの顔認識データを用い，リアルタイム画像収集モジュールで収集した画像内の顔パーツの位置情報を取得する．

画像処理によるモーションキャプチャをおこなうにあたり，結果として下記の 3 パターンが想定される．

- 正常に 1 つの顔を認識し，位置情報を取得
- 2 つ以上の顔を認識し，複数の位置情報を取得
- 1 つの顔も認識できず，位置情報が取得できない

本システムでは，1 つ以上の顔を顔を認識した場合，全ての顔パーツ位置情報を 3D モーション生成・配信モジュールに送信する．送信は，位置情報データを JSON 形式に取りまとめ，UDP を用いておこなう．

また，リアルタイム画像収集モジュールから非同期で画像ファイルを受け取るしくみであるため，画像ファイルの取得と画像処理のレートに不均衡が生じる．このため，画像ファイル取得レートより画像処理レートのほうが高くなり位置情報の取得ができない場合は，送信を見送る処理が必要である．また，画像ファイル取得レートより画像処理レートのほうが低くなり未処理画像が溜まる場合は，適宜古い画像を破棄する処理が必要となる．

4.5　3D モデルモーション生成・配信モジュール

画像処理によるモーションキャプチャは計算コストが大きく時間が掛かる．そのため，Blender Game Engine アプリケーション内に組み込むと，画像処理の間は描画処理がおこなわれず，フレームレートが著しく低下する．よって，Blender Game Engine アプリケーション内に組み込むことはできない．そこで，モーションキャプチャデーモンから受け取った顔の位置データを元に 3D モデルモーションを生成し，Blender Game Engine アプリケーションに配信する構成とする．

3D モデルモーション生成・配信モジュールでは，下記の二つの通信を担当する．

- モーションキャプチャモジュールから顔パーツ位置情報を UDP+JSON で受け取り，3D モデルモーションを生成する
- 3D モデル操作モジュールからの問い合わせがあった時は，生成したモーションを TCP+JSON で返す

3D モデルモーション生成・配信モジュール・3D モデル操作モジュール間の通信が，操作モジュール側から Pull する

形とすることで，複数の Blender Game Engine アプリケーションを同時に制御することが容易にできるようにする．

5 評価

本システムは，学生が所有しているノート PC などでも動作することを要件としている．そこで実際に表 1 の性能のノート PC で稼働させ，フレームレートに問題がないか評価する．また，稼働させる 3D モデルの要素数は表 2 の通りである．

表 1　評価用ノート PC の性能

CPU	Intel Core i5-6300U
GPU	Intel HD Graphics 520
メモリ	DDR4-2133 8GB
ストレージ	SATA SSD 500GB
画面解像度	1366 x 768
カメラ	ThinkPad Integrated Camera (解像度 1280 x 720，30fps. モーションキャプチャ時 320 x 240 30fps で利用)

表 2　評価用 3D モデルの要素数

頂点	62833
面	62811
三角面	125622
ボーン	21

これらの環境において，静止状態，通常アニメーション状態，顔検知状態のフレームレートをそれぞれ計測した．

- 静止状態: 45〜50fps
- 通常アニメーション状態: 35〜40fps
- 顔検知状態: 20〜35fps

結果として，静止状態と通常アニメーション状態では，目標であった 30fps を達成することができた．しかし，顔検知状態では 30fps を下回ることが多く，モーションのカクつきが目立つ．特にテクスチャ更新が発生した際に顕著であった．また，モーションキャプチャのフレームレートについても評価をおこなったが，16〜18fps で推移しており，30fps には届かなかった．

6 展望

現状のシステムは，Blender Game Engine に強く依存している．Blender Game Engine では，Windows や Linux 向けの単体実行ファイルを出力することも可能だが，スマートフォンなどでは動作させることができない．今後のデジタルアーカイブの展開として，スマートフォンなどをターゲットに含める可能性が高く，それらへの対応を考慮する必要がある．そこで，3D モデルの描画に WebGL を用い，Web ブラウザ上でおこなえるようにすることで，さらに多くのプラットフォームに対応することを検討している．その基盤をデジタルアーカイブに応用することで，3DCG と Web 技術とを融合したデジタルアーカイブの開発が期待できる．

また本システムは，3D モデルを PC やスマートフォンのディスプレイに描画するだけでなく，レーザープロジェクターと透過スクリーンを用いたプロジェクションマッピングなどにも応用が可能であり，そちらもあわせて検討していきたい．

7 おわりに

一般的なノート PC と Blender Game Engine を利用し，モーションキャプチャを用いたリアルタイム 3D キャラクターモーションシステムを開発することができた．これにより，フレームレートなど課題は残るものの，学生などが所有するノート PC のような機器であっても，安価で手軽にモーションキャプチャと 3D キャラクターの組み合わせによるシステムの利用が可能ということが分かった．今後は，これらで得た知見を 3DCG を用いたデジタルアーカイブなどに活用したい．

本研究の成果は `https://bitbucket.org/Phenomer/blendroid` に公開されている．

参考文献

[1] Blender Foundation. Game engine - blender manual. `https://docs.blender.org/manual/de/dev/game_engine/index.html`.

[2] freedesktop.org. Gstreamer: open source multimedia framework. `https://gstreamer.freedesktop.org/`.

[3] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[4] 晃伸 李, 圭一郎 大浦, and 恵一 徳田. 魅力ある音声インタラクションシステムを構築するためのオープンソースツールキット mmdagent. Technical Report 27, 名古屋工業大学／独立行政法人科学技術振興機構 CREST, 名古屋工業大学, 名古屋工業大学／独立行政法人科学技術振興機構 CREST, dec 2011.