# コンピュータグラフィックス特論II レポート
# 第5回 キャラクタ・アニメーション（1）

学生番号: 12345678　　氏名: 九工大 太郎

2021 年 6 月 11 日

レポートの書き方の注意：（この部分は、提出レポートからは削除すること）

- 以下の様式中の「※ レポート課題」の部分を、自分が作成したプログラムに置き換える。
- 変数定義やインデントを適切に行うこと。動作しないプログラムや見にくいプログラムは、減点となる。
- 様式で指定されている箇所以外に変更を加えた場合は、どの関数を追加変更したのかが分かるように、関数定義を含めて変更内容を枠内に記述する。

## 1　キャラクタ・アニメーションの実現

キャラクタ・アニメーションの基本処理を実現するように、以下の通り、元のプログラムの処理の一部に変更を加えた。

### 1.1　順運動学計算

#### 1.1.1　順運動学計算のための反復計算

ForwardKinematicsApp.cpp の MyForwardKinematicsIteration 関数の空欄部分を、以下のように作成した。

```
void  MyForwardKinematicsIteration(
   Segment *  segment, Segment * prev_segment,
   const Posture & posture, Matrix4f * seg_frame_array, Point3f * joi_pos_array )
{
   // 省略

   // 現在の体節に接続している各関節に対して繰り返し
   for ( int j=0; j<segment->joints.size(); j++ )
   {
      // 省略

      // ※レポート課題（ここに自分が作成したプログラムを記述する）


      // 次の体節を呼び出す
      MyForwardKinematicsIteration( next_segment, segment, posture, seg_frame_array,
          joi_pos_array );
   }
}
```

## 1.2 姿勢補間

### 1.2.1 2つの姿勢を補間

PostureInterpolationApp.cpp の MyPostureInterpolation 関数の空欄部分を、以下のように作成した。

```
void MyPostureInterpolation( const Posture & p0, const Posture & p1, float ratio,
    Posture & p )
{
    // 省略

    // 骨格モデルを取得
    const Skeleton * body = p0.body;

    // ※レポート課題（ここに自分が作成したプログラムを記述する）

}
```

## 1.3 キーフレーム動作再生

### 1.3.1 キーフレーム動作からの姿勢取得

KeyframeMotionPlaybackApp.cpp の MyPostureInterpolation 関数の空欄部分を、以下のように作成した。

```
void GetKeyframeMotionPosture( const KeyframeMotion & motion, float time, Posture & p
    )
{
    // 指定時刻に対応する区間番号を取得
    // 省略

    // ※レポート課題（ここに自分が作成したプログラムを記述する）

}
```

## 1.4 動作補間

### 1.4.1 動作再生処理（動作補間）

MotionInterpolationApp.cpp の AnimationWithInterpolation 関数の空欄部分を、以下のように作成した。

```
void MotionInterpolationApp::AnimationWithInterpolation( float delta )
{
    // 省略

    // 補間動作のキー時刻を計算（サンプル動作のキー時刻を重みで平均）
    keytimes[ 0 ] = 0.0f;
    for ( int i = 1; i < num_keyframes; i++ )
    {
        // ※ レポート課題（ここに自分が作成したプログラムを記述する）
        keytimes[ i ] = ???;
    }

    // 省略

    // 正規化時間（区間番号と区間内の正規化時間）を計算
    for ( int i = 0; i < num_keyframes-1; i++ )
    {
        if ( ( local_time >= keytimes[ i ] ) && ( local_time <= keytimes[ i + 1 ] ) )
        {
            seg_no = i;
```

```
            // ※レポート課題（ここに自分が作成したプログラムを記述する）
            seg_time = ???;

            break;
        }
    }

    // サンプル動作から姿勢を取得
    float  motion_time = 0.0f;
    for ( int i = 0; i < 2; i++ )
    {
        // 各サンプル動作の現在時刻（動作データ内の時間）を計算

        // ※ レポート課題（ここに自分が作成したプログラムを記述する）
        motion_time = ???;

        // 省略
    }

    // 省略
}
```