コンピュータグラフィックス特論II レポート
第6回 キャラクタ・アニメーション（2）

学生番号: 12345678　　氏名: 九工大 太郎

2021 年 6 月 18 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　動作接続・遷移

1.1.1　動作接続のための変換行列の計算

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

```
void  ComputeConnectionTransformation( const Matrix4f & prev_frame, const Matrix4f &
    next_frame, Matrix4f & trans_mat )
{
   // ※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）

}
```

1.1.2　動作再生処理（動作接続・補間）

MotionTransitionApp.cpp の InitTransition 関数と GetPostureInTransition 関数の空欄部分を、以下のように作成した。

```
bool  MotionTransitionApp::InitTransition(
   const MotionInfo * prev_motion, const MotionInfo * next_motion, const Matrix4f &
       prev_motion_mat, float curr_local_time,
   MotionTransitionInfo * transition )
{
   // 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）
```

```
    // 次の動作を開始する時刻（現在の動作の開始時刻 curr_start_time を基準とするローカル
        時刻）
    next_begin_time = ???;

    // 動作遷移のための動作ブレンドを行う開始時刻（現在の動作の開始時刻 curr_start_time
        を基準とするローカル時刻）
    blend_begin_time = ???;

    // 動作遷移のための動作ブレンドを行う終了時刻（現在の動作の開始時刻 curr_start_time
        を基準とするローカル時刻）
    blend_end_time = ???;

    // 省略
}


bool  MotionTransitionApp::GetPostureInTransition(
    const MotionTransitionInfo * transition, float curr_local_time,
    Posture * posture, float * output_blend_ratio )
{
    // 省略

    // 動作接続・遷移のためのブレンド中は、次の動作の姿勢とブレンド
    if ( on_motion_transition && transition->next_motion )
    {
        // 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）

        // 補間の重み（ブレンド比率）を計算
        blend_ratio = ???;

        // 省略
    }

    // 省略
}
```

1.2 動作変形

1.2.1 姿勢変形の重みを計算

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

```
float  ApplyMotionDeformation( float time, const MotionWarpingParam & deform, const
    Posture & input_pose, Posture & output_pose )
{
    // 省略

    // 動作変形（動作ワーピング）の重みを計算

    // ※ 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）

    // 省略
}
```

1.2.2 動作ワーピングのための姿勢変形

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

```
void PostureWarping( const Posture & org, const Posture & src, const Posture & dest,
    float ratio, Posture & p )
{
   // 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）

   // 各関節の回転を計算
   for ( int i = 0; i < body->num_joints; 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）
   }

   // ルートの向きを計算

   // ※ 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）

   // ルートの位置を計算

   // ※ 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）

}
```

1.3 逆運動学計算（CCD 法）

1.3.1 逆運動学計算（CCD 法）（ルート体節を支点とする場合）

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

```
void ApplyInverseKinematicsCCD( Posture & posture, int base_joint_no, int ee_joint_no,
    Point3f ee_joint_position )
{
   // 省略

   // CCD法の繰り返し計算（末端関節の位置が収束するか、一定回数繰り返したら終了する）
   for ( int i=0; i<max_iteration; i++ )
   {
      // 末端関節から支点関節に向かって繰り返し
      for ( int j=0; j<joint_path.size(); j++ )
      {
         // 省略

         // 末端関節の位置を目標位置に近づけるための現在の関節の回転を計算・適用

         // ※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）

         // 省略
      }

      // 収束判定、末端関節の目標位置と現在位置の距離が閾値以下になったら終了

      // ※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）

   }
}
```

1.3.2 末端関節から支点関節へのパスを探索（任意の関節を支点とする場合）

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

```
void FindJointPath( const Skeleton * body, int base_joint_no, int ee_joint_no, vector<
    int > & joint_path, vector< int > & joint_path_signs )
{
   // 省略

   // ルートから支点関節までのパスを求めて追加

   // ※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）

}
```

1.3.3 逆運動学計算（CCD 法）（任意の関節を支点とする場合）

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

```
void ApplyInverseKinematicsCCD( Posture & posture, int base_joint_no, int ee_joint_no,
    Point3f ee_joint_position )
{
   // 省略

   // CCD法の繰り返し計算（末端関節の位置が収束するか、一定回数繰り返したら終了する）
   for ( int i = 0; i < max_iteration; i++ )
   {
      // 末端関節から支点関節に向かって繰り返し
      for ( int j = 0; j < joint_path.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）


         // 省略
      }

      // 省略
   }
}
```