REPORT – HW2

Introduction/Motivation:

這次作業是對所提供 skeleton 實作 Forward Kinematics 和 Time Warping。Forward Kinematics 目的是去算 bone 的 start position, end position, rotation 來模擬出整個人型和動作。Time Warping 目的是去計算新的 translation 和 rotation。作業中應用多種公式來計算。

Fundamentals:

Local coordinates: 是物體本身的坐標系，也稱為局部坐標系。在局部坐標系中，物體的位置和方向是相對於該物體本身而言的，不受其他物體或空間的影響。

Global coordinates: 是全局的坐標系，也稱為世界坐標系。在世界坐標系中，物體的位置和方向是相對於整個空間或者世界而言的，可以用來描述不同物體之間的相對位置關係。

Implementation:

Forward Kinematics:

使用 rotateDegreeZYX 把 Vector R_{amc} 轉成 Quaternion。

分成兩個部分，在 root (沒有 parent) 和其他 bone。

Root:

Rotation 會是$\boldsymbol{R_{amc}}$，沒有$\boldsymbol{R_{asf}}$的影響。

Start position 是 bone 的 translation。

因為 root 的 length 是 0，所以 end position 就是 start position。

```
if (bone->name == "root") {
    bone->rotation = R.toRotationMatrix();
    bone->start_position = posture.bone_translations[bone->idx];
    bone->end_position = bone->start_position;
```

其他 bone:

使用$\boldsymbol{{}_{i}^{i+1}R = {}_{i}R_{asf} * {}_{i}R_{amc}}$ 和 $\boldsymbol{{}_{0}^{i}R = {}_{0}^{1}R_{1}^{2}R \dots {}_{i-1}^{i}R}$ (需乘上 parent 的 rotation) 計算 bone->rotation。

Start position 會是從 parent 的 end position 加上該 bone 的 translation。

End position 會從 bone 本身的 start position，使用$\boldsymbol{V_i = \widehat{V_i} * l_i}$ 和 $\boldsymbol{{}_{i}T = {}_{0}^{i-1}RV_{i-1} + {}_{i-1}T}$ 計算。

```
else {
  bone->rotation = bone->parent->rotation * bone->rot_parent_current * R.toRotationMatrix();
  bone->start_position = bone->parent->end_position + posture.bone_translations[bone->idx];
  bone->end_position = bone->start_position + bone->rotation * bone->dir * bone->length;
```

使用 DFS 去更新全部的 bone。

```
if (bone->child) forwardSolver(posture, bone->child);
if (bone->sibling) forwardSolver(posture, bone->sibling);
```

Time Warping:

計算 translations 使用 linear interpolation。

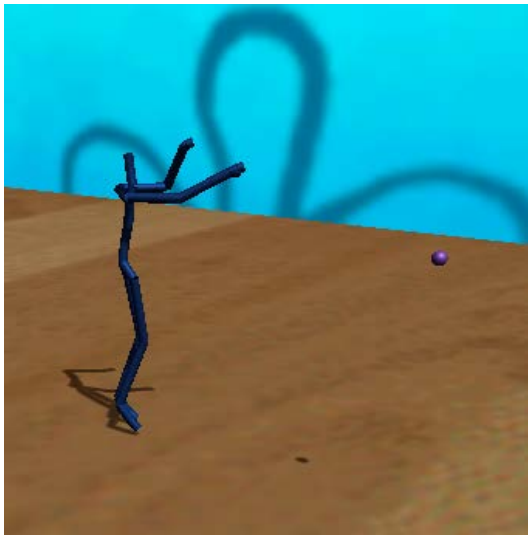
$$y = y_0\left(\frac{x_1-x}{x_1-x_0}\right) + y_1\left(\frac{x-x_0}{x_1-x_0}\right) = y_0(1-\alpha) + \alpha y_1,\ \alpha = \left(\frac{x-x_0}{x_1-x_0}\right) = x - x_0$$

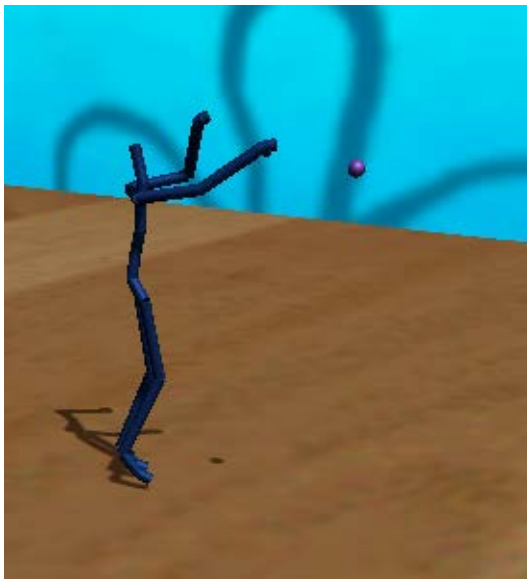
```
double scale_factor = static_cast<double>(allframe_old) / allframe_new;
double old_frame_idx = i * scale_factor;
int left_frame_idx = static_cast<int>(old_frame_idx);
int right_frame_idx = std::min(left_frame_idx + 1, total_frames - 1);
double alpha = old_frame_idx - left_frame_idx; //right_frame_idx - left_frame_idx = 1

Eigen::Vector4d left_translation = postures[left_frame_idx].bone_translations[j];
Eigen::Vector4d right_translation = postures[right_frame_idx].bone_translations[j];
Eigen::Vector4d new_translation = (1 - alpha) * left_translation + alpha * right_translation;
new_poseture.bone_translations[j] = new_translation;
```

計算 rotations 使用 spherical linear interpolation，在作業中利用 Quaternion 的 slerp 來計算，取 coefficients。

```
Eigen::Quaterniond left_rotation(postures[left_frame_idx].bone_rotations[j]);
Eigen::Quaterniond right_rotation(postures[right_frame_idx].bone_rotations[j]);
Eigen::Quaterniond new_rotation = left_rotation.slerp(alpha, right_rotation);
new_poseture.bone_rotations[j] = new_rotation.coeffs();
```

Result and Discussion:





為了可以計算，vector 需要轉成 Quaternion。使用 rotateDegree 轉換成 Quaternion 要注意方向。

在使用乘法中，順序很重要，rotation 算錯會嚴重影響到模型。

在 Time Wraping，all frame 越小 skeleton 動作越快，球飛越遠，反過來，all frame 越大 skeleton 動作越慢，球飛越近，但到某程度球落在同樣的地方。

在這次作業中有應用 DFS Tree 實作出 skeleton 模型。

Bonus:

修改 main.cpp:

Namespace 增加: `bool isRunning = false;`

Main function 增加和修改:

```
acclaim::Motion walk(acclaim_folder / "walk.amc", std::make_unique<acclaim::Skeleton>(*skeleton));
acclaim::Motion run(acclaim_folder / "running.amc", std::make_unique<acclaim::Skeleton>(*skeleton));
```

```
if (isTimeWarping)
    totalFrames = shooting.getFrameNum();
else {
    if (isRunning) {
        if (currentFrame > run.getFrameNum())
            currentFrame = currentFrame * (run.getFrameNum() / walk.getFrameNum());
        running = run;
    } else
        running = walk;

    totalFrames = running.getFrameNum();
}
```

MainPanel function 增加 Button:

```
ImGui::SameLine();
if (ImGui::Button("Run")) {
    isRunning ^= true;
}
ImGui::SameLine();
ImGui::Text(isRunning ? "ON" : "OFF");
```

實作:

多讀取 running.amc，實作 Running 和 Walking 模式可以切換。因為 running 的 total frame < walking 的 total frame，會有 current frame 超出範圍導致錯誤，為了防止錯誤所以增加 current frame 的條件。

Conclusion

在 Forward Kinematics，利用所提供的.ASF 和.AMC File 的數據和 kinematics 公式去計算 start position, end position, rotation。
實作 Time Warping 應用了 linear interpolation 去計算新的 translations,

spherical linear interpolation 去計算新的 rotations。
調整 key frame 回影響到 skeleton 動作速度與球的落點。