**Introduction to Computer Animation HW3 Report**

109350008 張詠哲

**• Introduction/Motivation**

這次要在這個作業中實作的也是模擬角色的動作。但是與HW2不同的是，HW2使用的方法是Forward Kinematics，而這次我們改為使用Inverse Kinematics。相較於Forward Kinematics是根據關節的角度或變換矩陣計算末端骨頭的位置和姿態。而Inverse Kinematics則是根據期望的末端骨頭的位置和姿態，計算出關節的角度或變換矩陣，以實現所需的運動。作業中我們會利用上次做的Forward Kinematics計算出骨架 Pose，接著計算需要的Jacobian Matrix。通過Pseudo Inverse找到orientation的變化，然後更新每個骨頭的旋轉角度來模擬整個骨架的形狀，讓目標骨頭的End position更接近Target。

**• Fundamentals**

* **Inverse Kinematics**

Inverse Kinematics的目標是計算影響end effector的關節自由度，使其能夠到達我們設定的目標位置。相比於Forward Kinematics是在給定角度後，計算到達的點；Inverse Kinematics是計算給定我們要到達的點後應該改變的角度。由於我們希望逐步到達目標位置，因此需要通過不斷的計算和修正來調整方向，使end effector接近目標位置。採用迭代的方法計算，通過每一次小修正，最終達到目標。

* **Jacobian Matrix**

Jacobian Matrix可以表示為最佳的linear approximation of a multivariate vector function。其中每個元素表示end effector位置或姿態的變化對關節角度的變化的敏感度。可知當調整關節角度時，end effector的位置和姿態如何變化。這次作業中代表了我們要計算的系統的偏導數。用於定義end effector的相對瞬時變化。

* **Pseudo Inverse**

假設有一個線性系統 Ax = b，其中 A 是一個 m × n 的矩陣，x 和 b 是 n 維的向量。當這個系統無法有精確解時，我們可以使用最小平方法（Least Squares Method）來尋找一個近似解。而Pseudo Inverse就是在最小平方法中使用的一個特殊矩陣。

在經過計算上面的Jacobian Matrix，可以知道end effector是如何影響x、y、z軸各個角度的關係，接著為了找到方向的變化，我們需要計算Jacobian Matrix的inverse，並將其乘以 V（target和end effector之間的距離）。但是Inverse Jacobian Matrix很難計算。因此改用Pseudo Inverse來計算J+來代替。

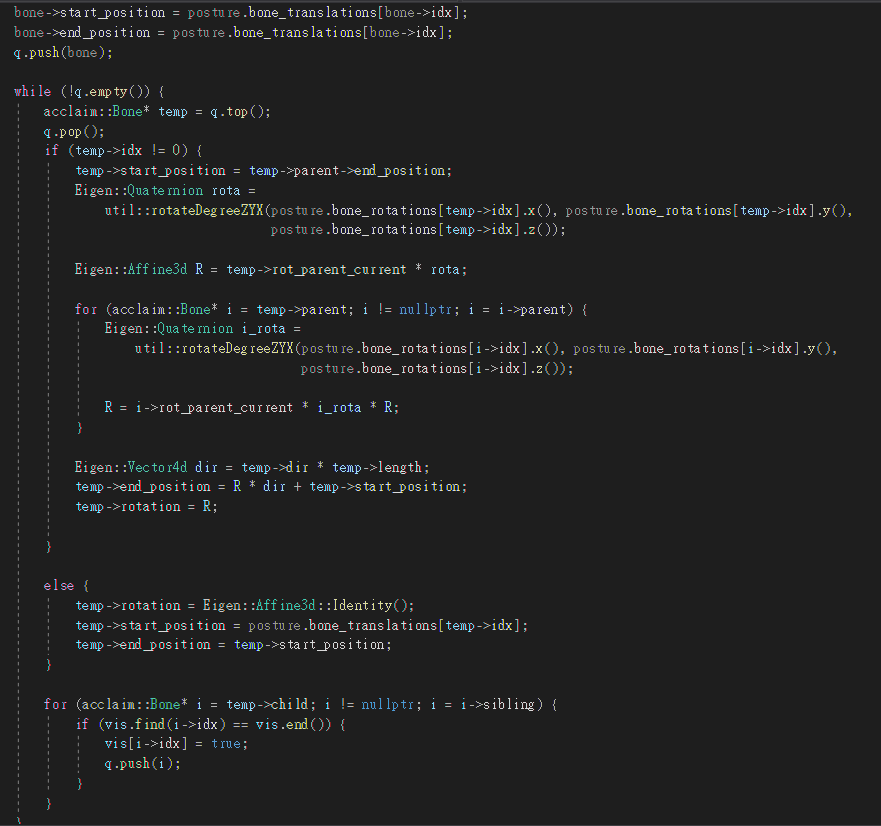
* **Orientation**

我們可以知道每個骨骼的旋轉角度，但是為了讓end effector可以到達目標位置，我們需要改變骨頭的方向，逐步計算我們最終的方向.

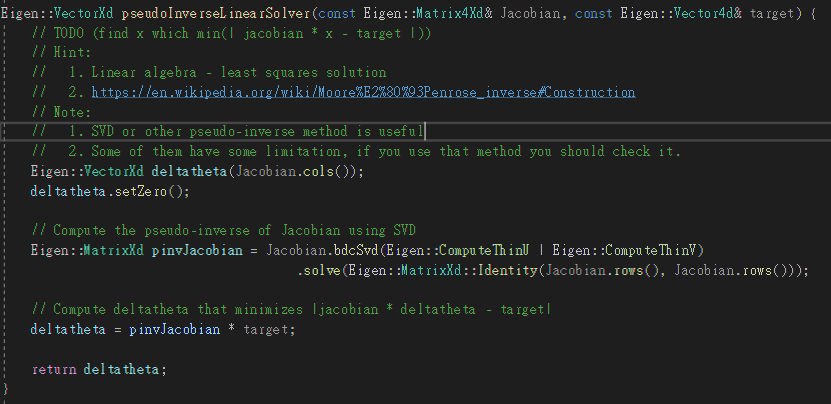
**• Implementation**

* **forwardSolver**

這部分和HW2一樣。

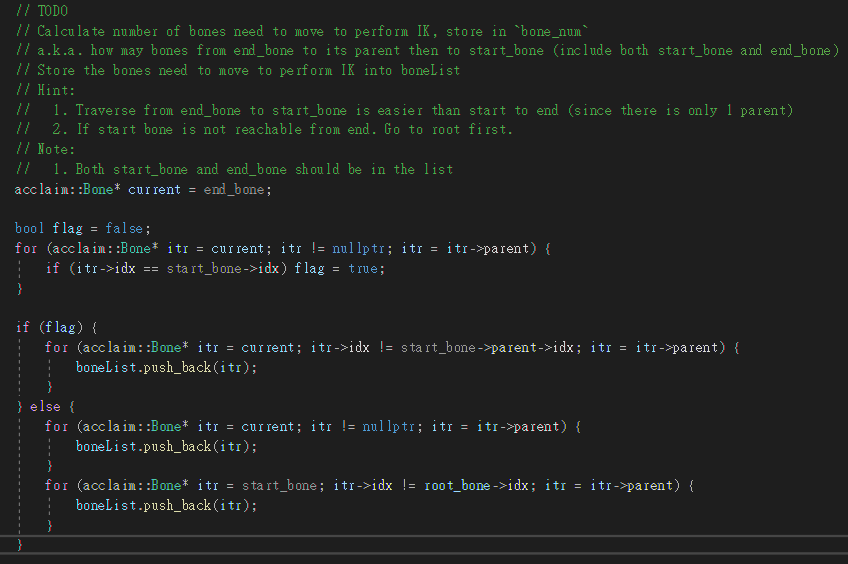


* **pseudoInverseLinearSolver**

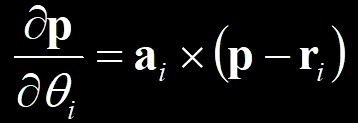
這次使用SVD 來計算pseudo inverse。矩陣M 可以分解為，而pseudo inverse為，其中 Σ+ 是 Σ 的轉置，所有元素都變成倒數。在程式碼中，使用bdcSvd() 函數來進行奇異值分解，並使用 solve() 函數計算 Jacobian 的pesudo inverse matrix (pinvJacobian)。然後，然後把target 乘以pinvJacobian，得到 deltatheta (最小化 | Jacobian \* deltatheta – target | 的解)。 最後返回。

* **inverseJacobianIKSolver**
* **Travel the skeleton**

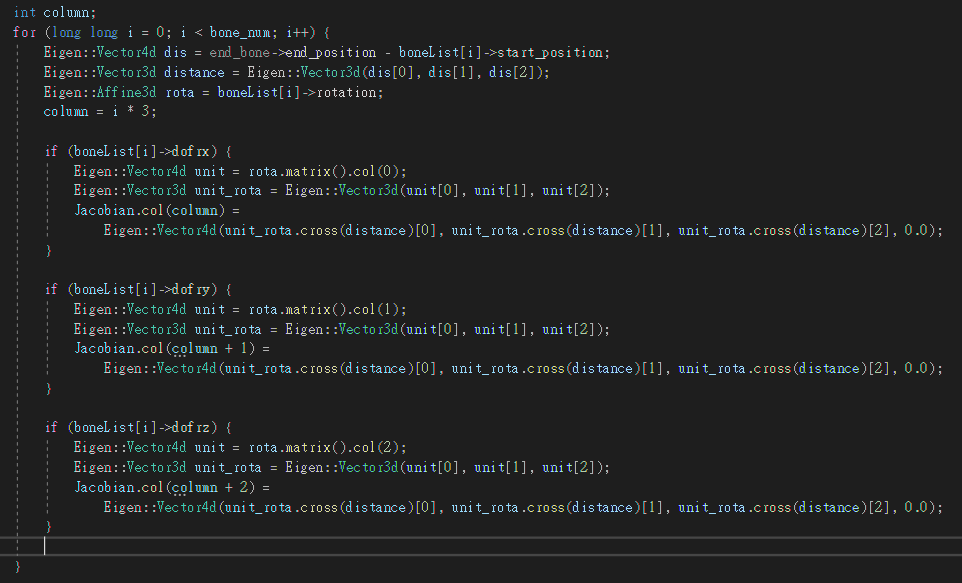
我們需要遍歷骨架以獲得boneList。這個過程中當我們使用 parent 遍歷骨架時，我們可能會遍歷到root然後因為根的parent是 nullptr而被迫中止。因此我首先遍歷骨骼檢查結束骨骼是否可以不通過根到達起始骨骼。



* **Jacobian**

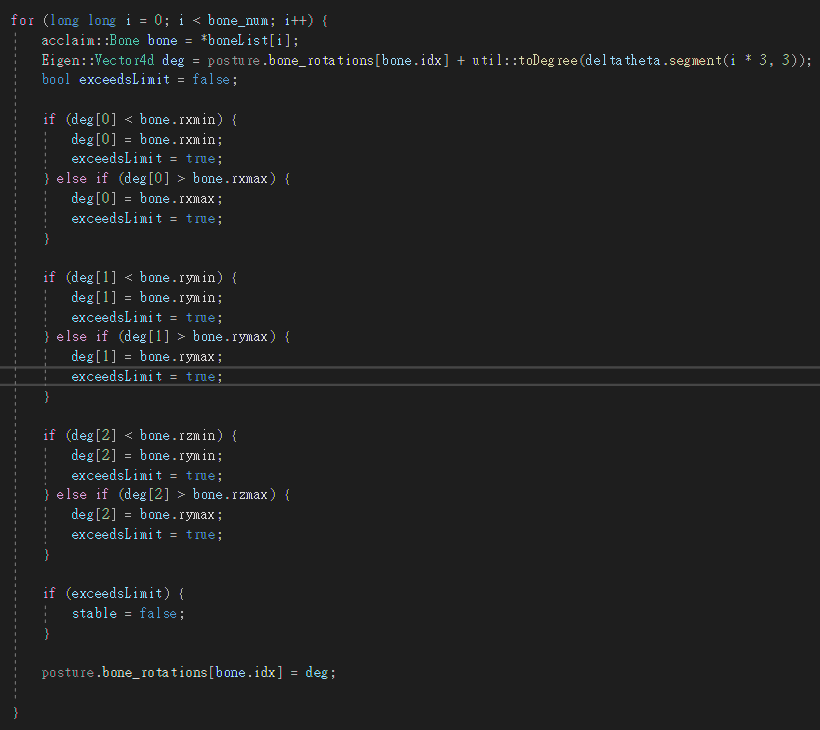


上面的公式為計算Jacobian Matrix的內容。P是結束骨骼的結束位置，是當前骨骼的起始位置，是旋轉軸。旋轉矩陣的第一列是x軸，第二列是y軸，第三列是z軸。最後是僅當骨骼具有DOF 時才將須計算jacobian。



* **orientation**

得到方向變化後，我們可以將delta x, delta y, delta z轉換為Degree加上原來的rotation得到新的方向。這裡也順便做了bouns，若是新的方向會超過最大或最小的可旋轉角度(碰不到target)則為unstable，而其旋轉角度就設為最大或最小的可旋轉角度。



**• Result & Discussion**

* **How different step and epsilon affect the result**

因為Inverse Kinematic是採用迭代的方法計算，通過每一次小修正，最終達到目標。因此step和epsilon會影響。

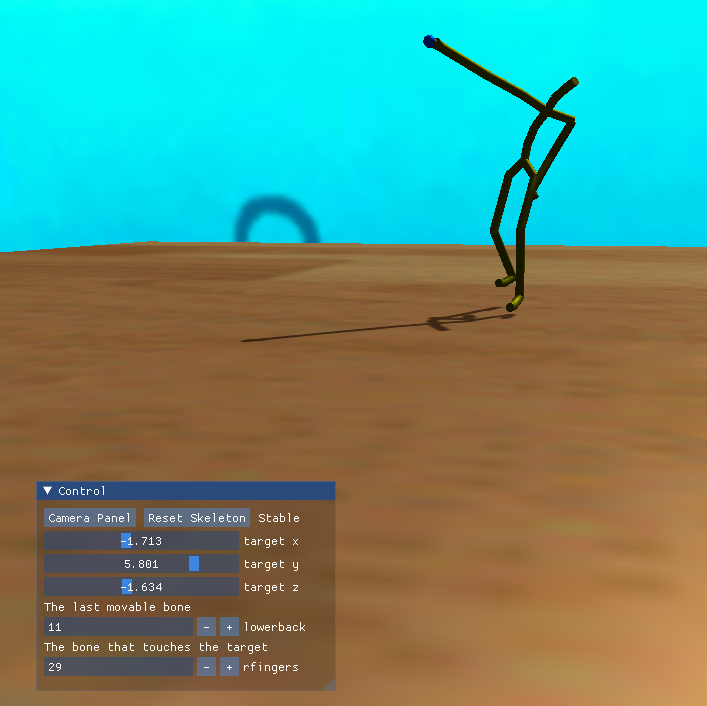
▪ **step affect**：step太大會導致收斂快，但相對不准確；相反的，如果step很小會導致收斂很慢，但相對準確。

▪ **epsilon affect**：distance epsilon是target與end effector之間距離誤差的可接受範圍。如果設置太大，兩者之間的距離會比較遠。如果設置得太小，很容易使end effector一直被計算。

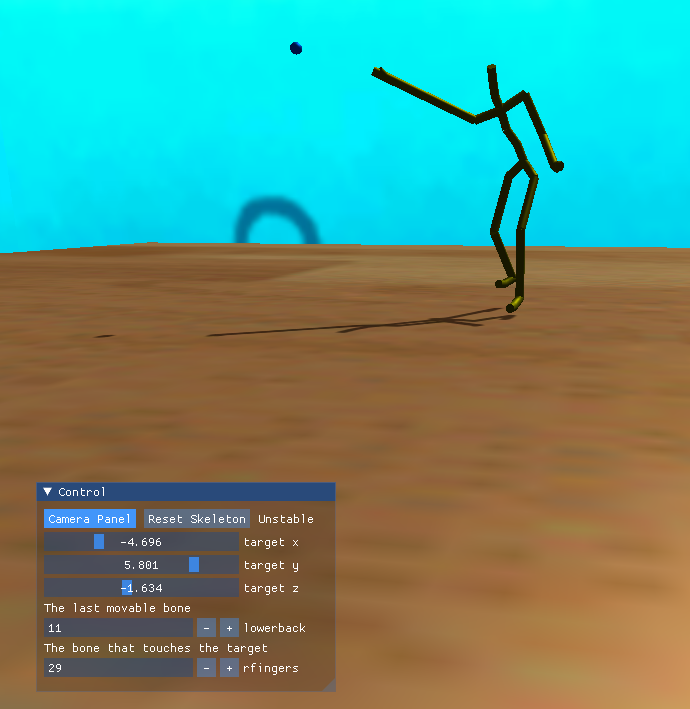
* **Touch the target or not**

當球靠近骨架時，骨架都可以接觸到球。但如果球離骨架很遠，骨架就不能碰到球。

touch



Don’t touch



* **Least square solve(LSS)**

解Least square solve的方法有多種:

**Normal Equations：**

透過把LSS轉化為求解線性方程組來求解。將系統的jacobian matrix成上目標向量，然後解析求解線性方程組來獲得LSS解。這種方法在計算上比較簡單，但可能存在數值穩定性的問題。

**QR decomposition：**

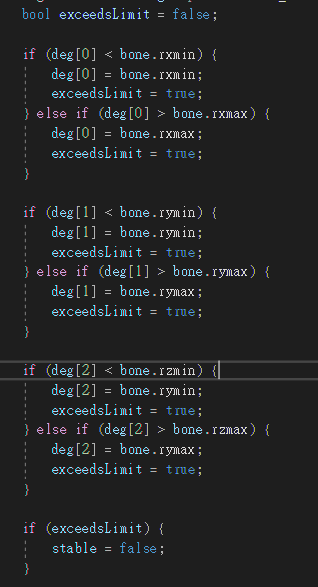
將jacobian matrix分解為一個正交矩陣和一個上三角矩陣的乘積。具有較好的數值穩定性。通過QR分解，可以使用Backward Substitution求解LSS。

**Singular Value Decomposition(SVD)：**

SVD是最廣泛使用的方法之一，可以適用於非方正矩陣。通過SVD，可以將矩陣分解為三個部分：。可以用於計算矩陣的pseudo inverse。具有良好的數值穩定性和robust。這次的作業中採用的即是SVD來求解。

而不同的算法對於骨架移動呈現的效果造成的影響可能原因為: 數值精度、收斂速度、平滑度、robust。

**• Bonus**



解釋在上面的orientation

**• Conclusion**

在本次作業中，我們實作了角色動作的模擬，與上次的HW2不同是這次採用了Inverse Kinematics的方法。根據期望的末端骨頭位置和姿態，計算出關節的角度或變換矩陣，以實現所需的運動。在本次作業中，我們首先利用上次實作的Forward Kinematics方法計算出骨架的Pose，然後進一步計算所需的Jacobian Matrix。透過Pseudo Inverse找到orientation的變化量，並使用這些變化量來更新每個骨頭的旋轉角度，以模擬整個骨架的形狀。這樣做可以使目標骨頭的末端位置更接近期望的目標位置。