•Introduction/Motivation

用很多關節和骨頭組成一個骨架，模擬人的行為。每個關節有不同的DOF，有不同的可活動的角度，我們透過設定每個frame的關節角度去產生不同的動作，形成一個連續的動畫。

目的是熟悉forwarding kinematics，給定joint space的資料，我們要把他map到cartesian space，算出給的關節與骨頭在global座標系下的位置與朝向。

• Fundamentals

1.線性代數，joint space轉換到cartesian space中間需要很多矩陣的運算。

2.數值方法，timewarp和motionblend的線性內插就是一種數值方法。

3.C++能力，要會寫C++、懂得使用物件、了解大型專案如何運作才能順利完成程式，並要熟悉新的Library。

• Implementation

1. void forwardKinematics(...)：將當前frame的關節與骨頭的資料從joint space map到cartesian space。每個frame會從main call一次本function，且每次從main call都是傳root的bone進來，我透過DFS 一直走訪sibling和child更新他們的值直到碰到nullptr。計算rotation的方式就是Rasf * Ramc，而root的Rasf=單位矩陣，其他的Rasf=parent的rotation * parent2child，而Ramc則直接使用posture裡的rotation即可；startposition = posture translation + parent bone的endposition，本case只有root的translation非0，root沒有parent所以不用考慮parent；endposition = startposition + rotation * direction * length，rotation*direction是用來將骨頭的方向用global來表示，因為direction是單位向量所以要記得*length。

2. Motion motionWarp(...)：在newKeyframe時播放oldKeyframe，所以其他部分要進行加速或減速。作法是將newMotion從newKeyframe開始分成前後兩段，前半段的ratio為oldKeyframe/newKeyframe，map到原本的動畫的影格為i*ratio；後半段的ratio為(totalFrame – oldKeyframe)/(totalFrame - newKeyframe)，map的影格為(i-newFrame)*ratio + newFrame；再根據map過去的影格在哪，對他做線性內插求出在newMotion的影格。在最後一個影格要對他做個處理，避免在計算的時候訪問到陣列外面。

• Result and Discussion

成功產生了火柴人的骨架並讓他可以正常跑動畫，這次作業需要模擬的點比較少，不像上次要模擬整個布料，所以這次在跑動話的時候比較順暢。

• Bonus (Optional)

Motion motionBlend(...)：將兩個motion合在一起變成一個motion，且要讓兩段動畫銜接的流暢。做法是先找matchRange中motionA和B difference最小的地方，接著計算motionA和B的offset，把motionA部分加到newMotion，然後用線性內插的方式來計算motionA和motionB混合的地方，接著再把motionB剩下的部分加到newMotion，後面計算motionB相關的東西的時候要記得考慮offset。

• Conclusion

這次作業的每個資料結構存的東西有點複雜，有的是local的有的是global的，然後又有分位置與旋轉，且式子有一點複雜，在寫的時候很容易搞混，當找到該用的資料時要寫的程式碼其實很少，套用正確的式子即可。