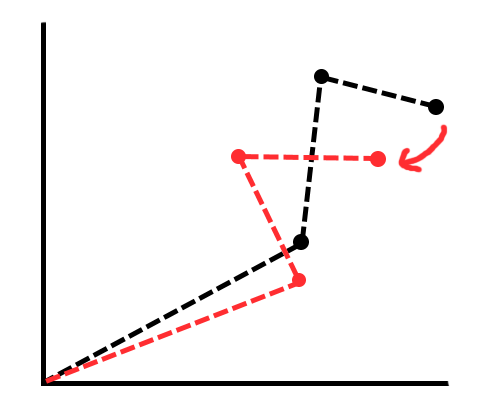
Throughout – drop references in the first time they are used, and again later on if required.

**Inverse Kinematics**

**Context**

My animation project will have 2D shapes linked together by joints to form an articulated chain or “arm” that can be moved for a desired keyframe. These individual links can be rotated around their joints, however, manipulating the whole “arm” whilst maintaining a realistic form would take tedious careful movements. This is where inverse kinematics comes in handy. The goal with inverse kinematics is the move the “hand” or end-effector and have the “arm” follow smoothly without breaking.

**Figure 1: example of IK**

There is also forward kinematics which achieves the opposite, given the joints of the chain you can calculate where the end effector is. However, in computer software a lot of positions of elements are already known e.g. given a square, you as the programmer define its position telling it where to be rendered – so this won’t help.

**Articulated Body**

An articulated body can be thought of like a hierarchical tree structure made of links that are connected together by joints. Links are simply connections between joints, however, there are multiple types of joints such as revolute and prismatic. A revolute joint is a joint that rotates its link and a prismatic joint extends and contracts its link. The initial joint is called the root or base, there can be multiple links from a root joint, for example: a torso of a humanoid body could be considered a root joint with arms and legs connected to it – but the root is still considered the first in the chain. The end of a chain is called the end-effector and there can be multiple end-effectors. Furthermore, the end-effector is used to control the joints via inverse kinematics. Articulated bodies can be expressed in terms of degrees of freedom, in 3D an articulated body could have a high overall DOF since each joint can have a maximum of three axes of rotation. However, in 2D only one axis of rotation exists so the overall DOF will be low in comparison.

**Solving methods for Inverse Kinematics**

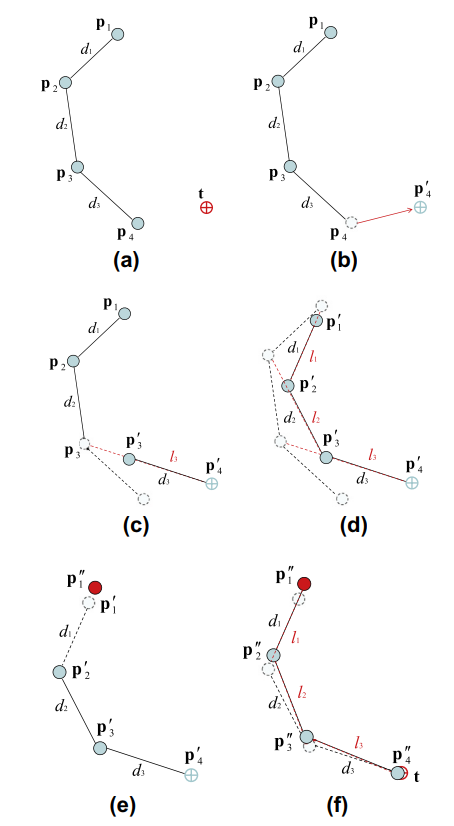
There are multiple ways to solve an inverse kinematics including: algebraic methods and iterative methods such as the Jacobian inversion method and FABRIK method. The algebraic solution involves calculating the end-effector position using trigonometry, however, if the degree of freedom value increases so do the steps required to solve the problem. Iterative methods work by solving the problem multiple times, each time getting a little bit closer to the intended goal. The Jacobian method is one of the earlier iterative methods that translates each joint by calculating the change in rotation of each joint using a Jacobian matrix. Firstly, calculate the difference in rotation for each joint: to do this we need the difference between the target (T) and the end-effector (E) and the inverse Jacobian.

However, the inverse of a matrix can only be calculated if the matrix has the same number of row as columns and the determinant is not zero otherwise it cannot exist. There is an alternative that we can use for an approximation of J-1 which is JT the Jacobian transpose.

The Jacobian matrix can be calculated from the cross products of the axis of rotation of a joint (Ri) and the difference in positions of the joint (Pi) and the end effector (E). Each term is a vector.

Now we know the change angles for each joint we can update each joints position (O). To do this we translate them by the difference in rotation multiplied by a timestep (h).

These steps are then repeated until the end effector is as close to the target as desired. The Jacobian method, however, requires a lot of computational power due to its use of matrices and cross product.

Another iterative method is the FABRIK method (Forward And Backward Reaching Inverse Kinematics). This method focuses on solving the inverse kinematic problem using only positions of joints and how to move them toward a subsequent target. There are two main steps to the FABRIK method called Forward Reaching and Backward Reaching. Forward reaching starts by making the end-effector equal to the target followed by finding where the previous joint lies on a line between the end-effector and said previous joint. This is repeated for each joint down the chain and results in the how chain being disconnected from the original root position.

This is fixed by Backward Reaching which repeats the Forward Reaching step but in reverse, starting by moving the root joint back. If the target is within the articulated bodies full length, the body will smoothly reach toward its goal after several iterations. The FABRIK method has a much lower computational cost when compared to methods like Jacobian since it doesn’t handle any rotation and much smaller amount of calculations to be made per iteration and produces much more natural and stable movements.

Figure 2: (a) to (d) demonstrates how each link is repositioned along a line between its previous joint and the goal/subsequent joint during Forward Reaching. (e) and (f) show Backward Reaching

**Figure 2. From: Aristidou, A. and Lasenby, J. (2011). FABRIK: A fast, iterative solver for the Inverse Kinematics problem. *Graphical Models*, 73(5), pp.243-260.**

Aristidou, A. and Lasenby, J. (2011). FABRIK: A fast, iterative solver for the Inverse Kinematics problem. *Graphical Models*, 73(5), pp.243-260.

Andreasaristidou.com. (n.d.). *Dr. Andreas Aristidou*. [online] Available at: http://www.andreasaristidou.com/FABRIK.html [Accessed 2 Nov. 2018].

Barinka, L. and Berka, I. (n.d.). [online] Old.cescg.org. Available at: http://old.cescg.org/CESCG-2002/LBarinka/paper.pdf [Accessed 2 Nov. 2018].

Bermudez, L. (2017). *Overview of Jacobian IK – Unity3DAnimation – Medium*. [online] Medium. Available at: https://medium.com/unity3danimation/overview-of-jacobian-ik-a33939639ab2 [Accessed 2 Nov. 2018].

Bermudez, L. (2017). *Create your own IK in Unity – Unity3DAnimation – Medium*. [online] Medium. Available at: https://medium.com/unity3danimation/create-your-own-ik-in-unity3d-989debd86770 [Accessed 2 Nov. 2018].

Bermudez, L. (2017). *Overview of Inverse Kinematics – Unity3DAnimation – Medium*. [online] Medium. Available at: https://medium.com/unity3danimation/overview-of-inverse-kinematics-9769a43ba956 [Accessed 2 Nov. 2018].