Geometric Deformation

Orthogonal Procrustes Problem

Optimization of Rotation: $\min_{R} W(R)$

Strategy A: Iteratively update rotation by parameterizing rotational update as $dR = \exp\{\text{Skew}(\vec{\omega})\}$

$$\vec{\omega} = \left[\frac{\partial^2 W}{\partial \vec{\omega}^2}\right]^{-1} \frac{\partial W}{\partial \vec{\omega}}, \qquad R \leftarrow R \exp\{\text{Skew}(\vec{\omega})\}$$

Hard to choose!

Strategy B: Singular Value Decomposition (SVD) can find the optimal rotation if $W(R)$ takes the form of $\|RA - B\|_F$

Optimizing Rotation for Quadratic Metric

$$R_{opt} = \underset{R}{\operatorname{argmin}} \|RA - B\|_F, \qquad \text{where } R^T R = I$$

$$= \underset{R}{\operatorname{argmin}} \|RA - B\|_F^2 = \underset{R}{\operatorname{argmin}} \langle RA - B, RA - B \rangle_F$$

$$= \underset{R}{\operatorname{argmin}} \{ \|A\|_F^2 - 2\langle RA, B \rangle_F + \|B\|_F^2 \}$$

$$= \underset{R}{\operatorname{argmax}} \langle RA, B \rangle_F = \underset{R}{\operatorname{argmax}} \langle R, BA^T \rangle_F$$

Optimizing Rotation for Quadratic Metric

$$R_{opt} = \underset{R}{\operatorname{argmax}} \langle R, BA^T \rangle_F$$

$$= \underset{R}{\operatorname{argmax}} \langle R, U\Sigma V^T \rangle_F = \underset{R}{\operatorname{argmax}} \langle U^T R V, \Sigma \rangle_F$$

$$= \underset{S}{\operatorname{argmax}} \langle S, \Sigma \rangle_F , \qquad \text{where } S^T S = I$$

$$= \underset{S}{\operatorname{argmax}} (S_{11}\Sigma_{11} + S_{22}\Sigma_{22} + S_{33}\Sigma_{33})$$

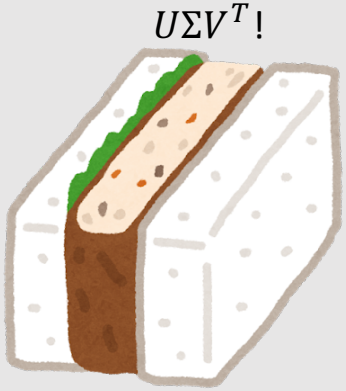
$$= UV^T$$

Solution of Orthogonal Procrustes Problem

Rotation optimization

$$R_{opt} = \underset{R}{\operatorname{argmin}} \|RA - B\|_F, \qquad \text{where } R^T R = I$$

$$R_{opt} = UV^T, \qquad \text{where } BA^T = U\Sigma V^T \text{ (using SVD)}$$



As-Rigid-As Possible Deformation

Elastic Energy using Singular Value Decomposition (SVD)

Deformation using SVD

Shape matching deformation

[Müller et al. 2005]

https://www.youtube.com/watch?v=LAoQJ1dhk1w

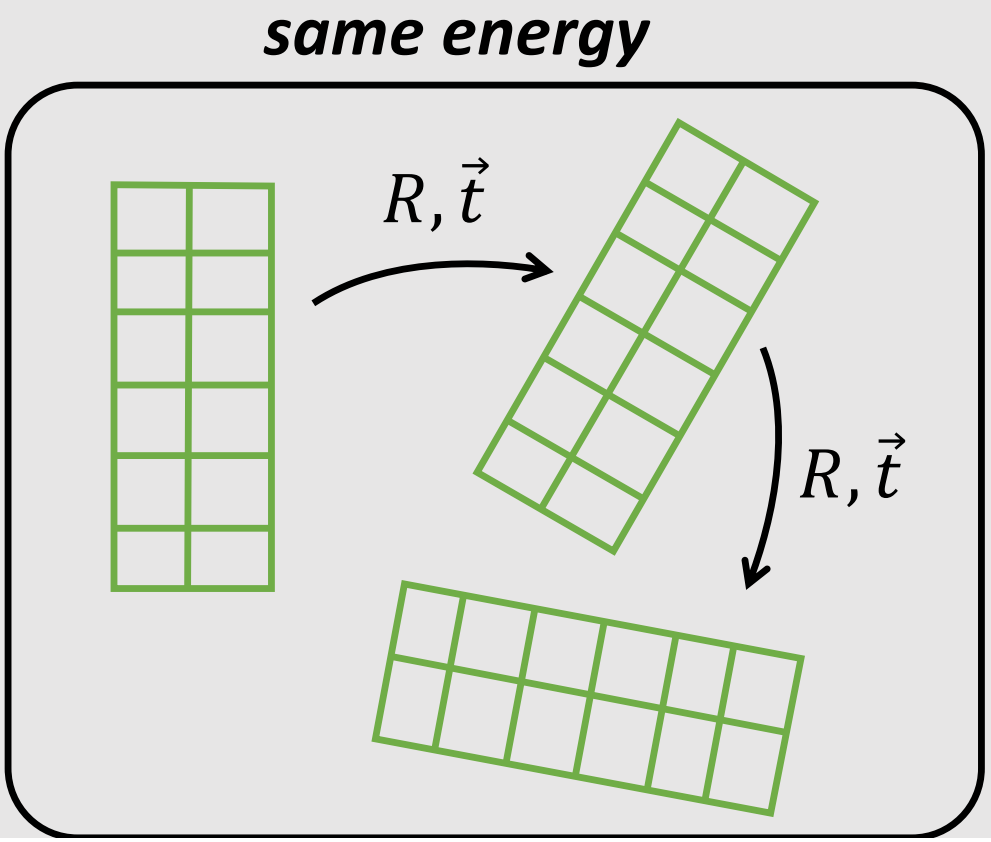
As-rigid-as possible deformation

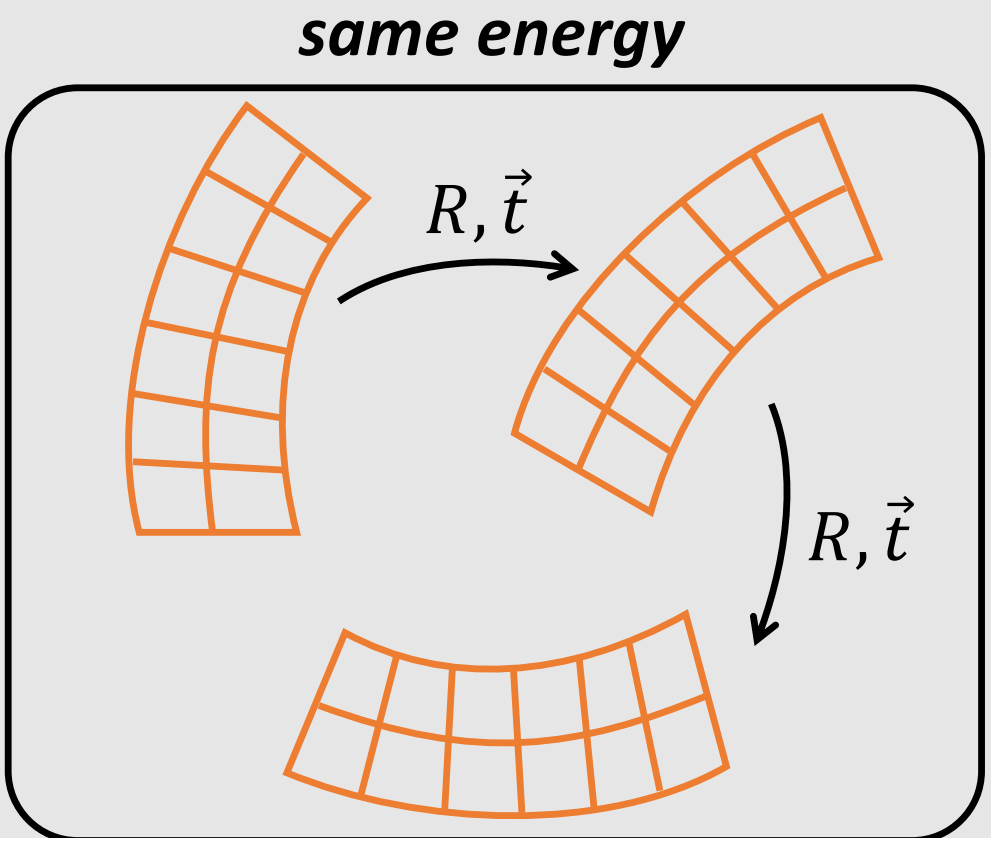
[Sorkine et al. 2007]

https://www.youtube.com/watch?v=ltX-qUjbkdc

Invariance of the Elastic Energy

- Deformation energy is not affected by rotation R and translation $\vec{t}$



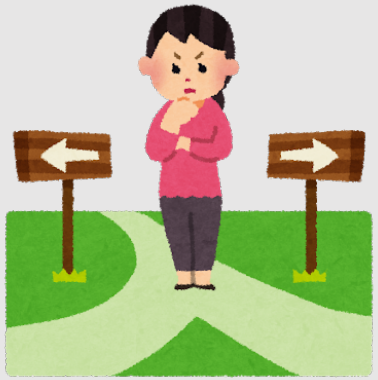


How can We Formulate Elastic Energy?

Strategy A: Elastic energy W is a function of eigenvalues of Green-Lagrange strain $E = F^T F - I$

$$W = \|F^T F - I\|_F^2, \qquad \text{where } F = \partial\vec{x}/\partial\vec{X}$$

cancel rotation *cancel translation*



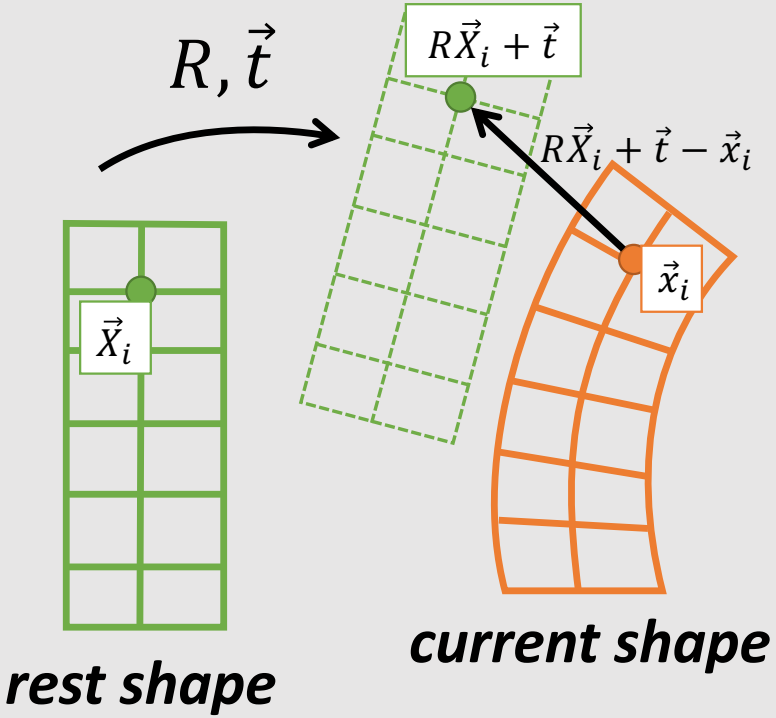
Strategy B: Elastic energy W is a sum of square distances after cancelling rotation and translation

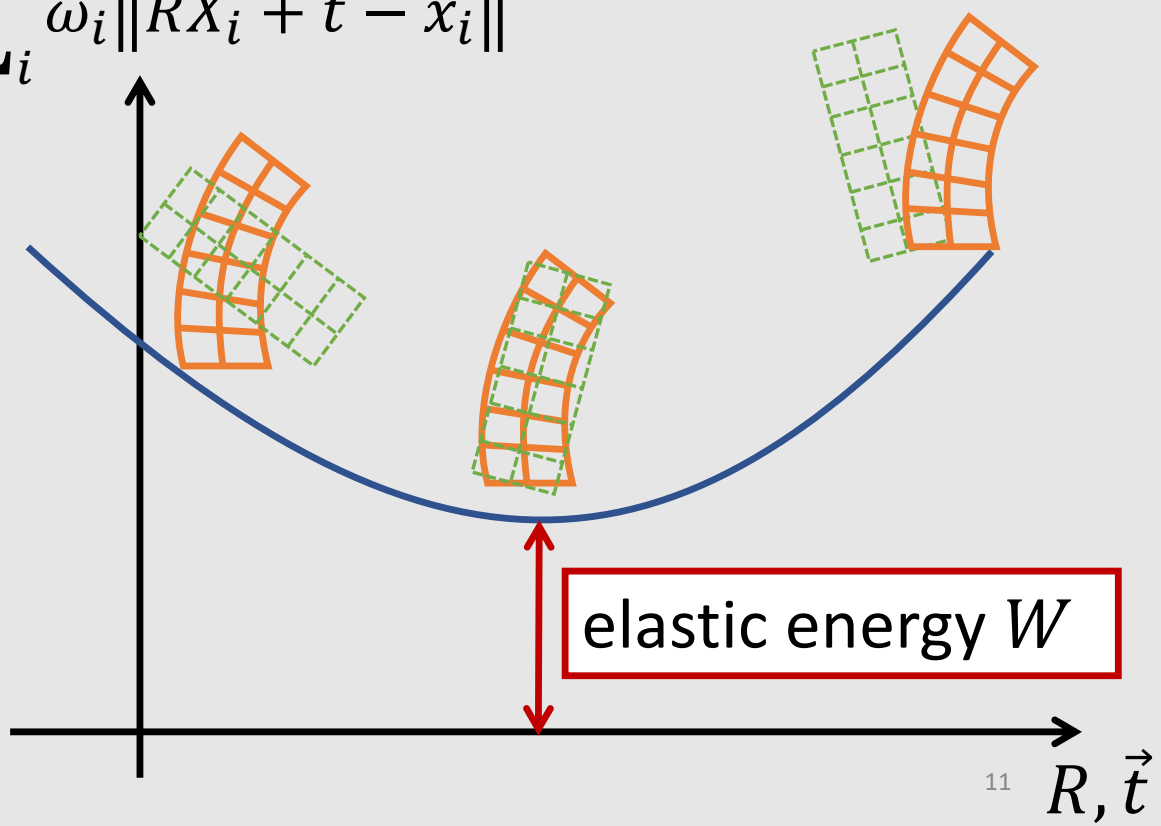
$$W = \min_{R,\vec{t}} \sum_i \omega_i \left\| R\vec{X}_i + \vec{t} - \vec{x}_i \right\|^2$$

Elastic Energy by Sum of Squared Distances

- Given points, if there is a rigid transformation the energy is the same

$$\sum_i \omega_i \left\| R\vec{X}_i + \vec{t} - \vec{x}_i \right\|^2$$

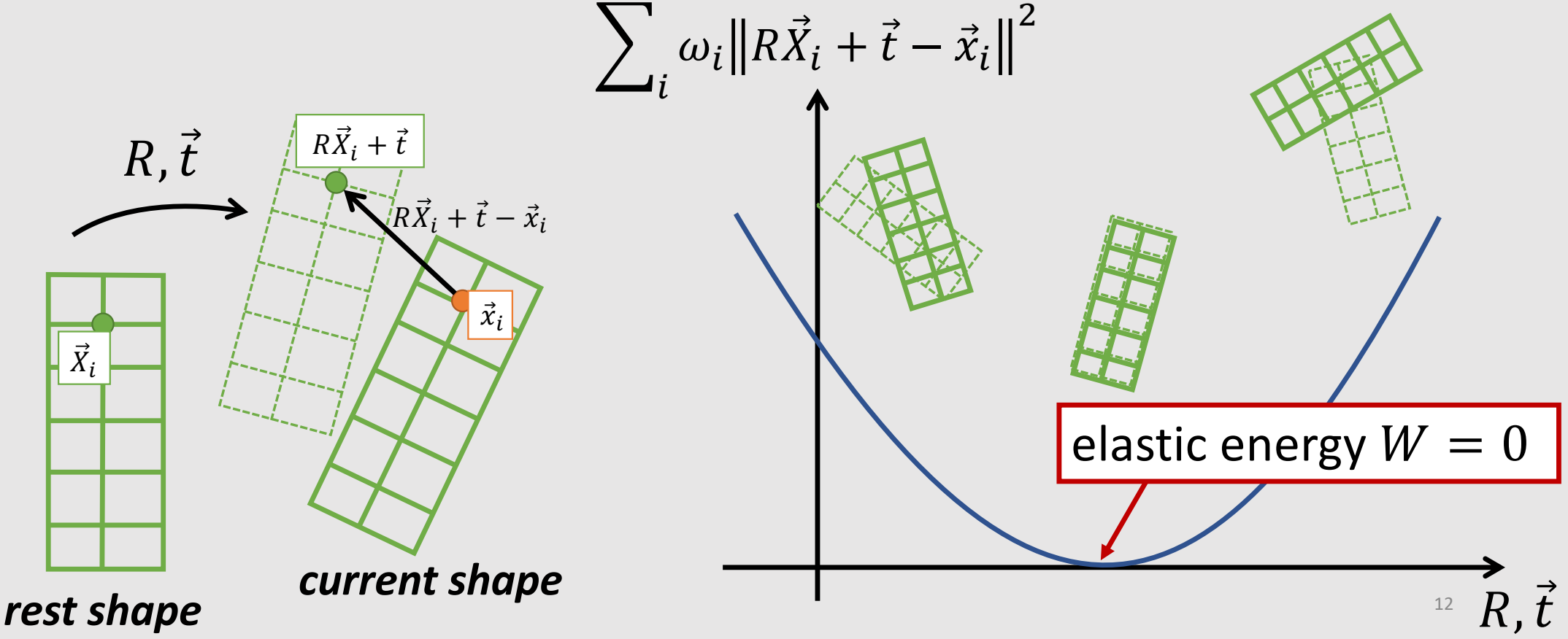




Elastic Energy by Sum of Squared Distances

- Given points, if there is a rigid transformation the energy is the same

$$\sum_i \omega_i \left\| R\vec{X}_i + \vec{t} - \vec{x}_i \right\|^2$$



Compute Elastic Energy W: Optimizing $\vec{t}$

$$\bar{W}(R,\vec{t}) = \sum_i \omega_i \left\| R\vec{X}_i + \vec{t} - \vec{x}_i \right\|^2$$

$$= \sum_i \omega_i \left\| R\vec{X}_i - \vec{x}_i \right\|^2 - \sum_i 2\omega_i (\vec{x}_i - R\vec{X}_i) \cdot \vec{t} + \sum_i \omega_i \left\| \vec{t} \right\|^2$$

$$\frac{\partial \bar{W}(R,\vec{t})}{\partial \vec{t}} = -\sum_i 2\omega_i (\vec{x}_i - R\vec{X}_i) + \vec{t} \sum_i 2\omega_i$$

$$\vec{t}_{cg} = \sum_i \omega_i \vec{x}_i \Big/ \sum_i \omega_i$$

$$\vec{T}_{cg} = \sum_i \omega_i \vec{X}_i \Big/ \sum_i \omega_i$$

$$\frac{\partial \bar{W}(R,\vec{t})}{\partial \vec{t}} = 0 \quad \Rightarrow \quad \vec{t}_{opt} = \sum_i \omega_i (\vec{x}_i - R\vec{X}_i) \Big/ \sum_i \omega_i = \vec{t}_{cg} - R\vec{T}_{cg}$$

Compute Elastic Energy W: Optimizing R

$$\bar{W}\left(R,\vec{t}_{opt}\right)=\sum_i \omega_i\left\|R\vec{X}_i+\vec{t}_{opt}-\vec{x}_i\right\|^2=\sum_i \omega_i\left\|R\left(\vec{X}_i-\vec{T}_{cg}\right)-\left(\vec{x}_i-\vec{t}_{cg}\right)\right\|^2$$

$$=\sum_i \left\|R\sqrt{\omega_i}\left(\vec{X}_i-\vec{T}_{cg}\right)-\sqrt{\omega_i}\left(\vec{x}_i-\vec{t}_{cg}\right)\right\|^2$$

$$=\|RA-B\|_F$$

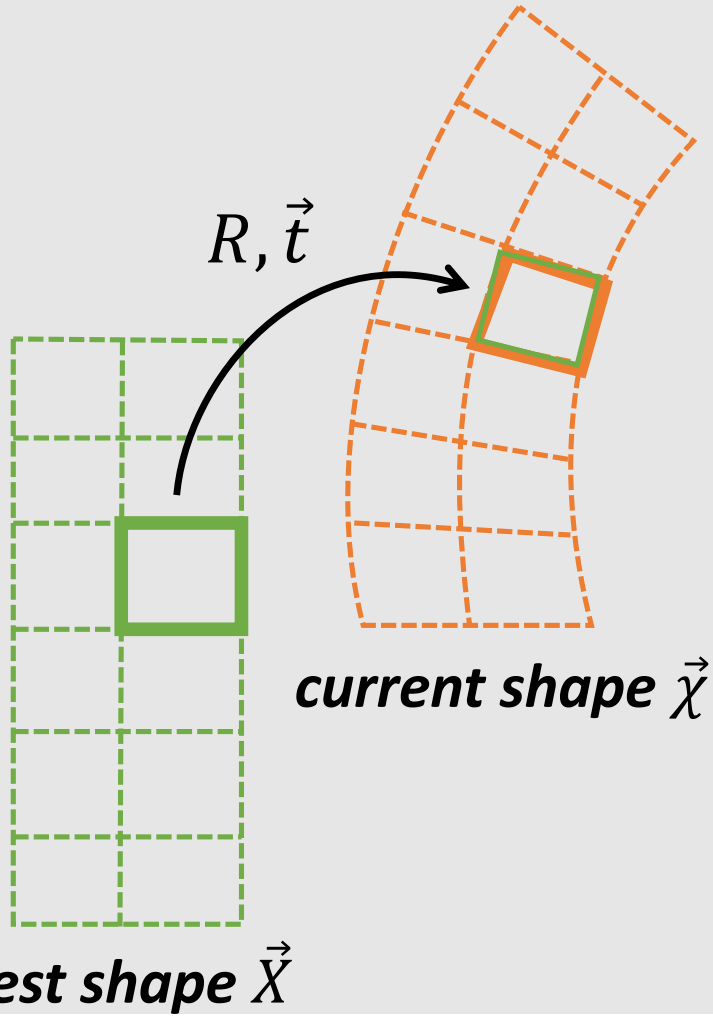
$$A=\left[\sqrt{\omega_1}\left(\vec{X}_i-\vec{T}_{cg}\right),\sqrt{\omega_2}\left(\vec{X}_i-\vec{T}_{cg}\right),\cdots\right]$$

$$B=\left[\sqrt{\omega_1}\left(\vec{x}_i-\vec{t}_{cg}\right),\sqrt{\omega_2}\left(\vec{x}_i-\vec{t}_{cg}\right),\cdots\right]$$

$$BA^T=\sum_i w_i\left(\vec{x}_i-\vec{t}_{cg}\right)\otimes\left(\vec{X}_i-\vec{T}_{cg}\right)$$

$$R_{opt}=\underset{\mathrm{R}}{\operatorname{argmin}}\,\bar{W}\left(R,\vec{t}_{opt}\right)=UV^T,\qquad \text{where } BA^T=U\Sigma V^T$$

Shape Matching Deformation [Müller et al. 2005]



1. compute temporary position

$$\vec{\chi}_i = \vec{x}_i + dt \cdot \vec{v}_i$$

2. For each element, update position

$$R_{opt}, \vec{t}_{opt} = \min_{R,\vec{t}} \sum_{i \in element} m_i \left\| R\vec{X}_i + \vec{t} - \vec{\chi}_i \right\|^2$$

$$\vec{\chi}_i = R_{opt}\vec{X}_i + \vec{t}_{opt}$$

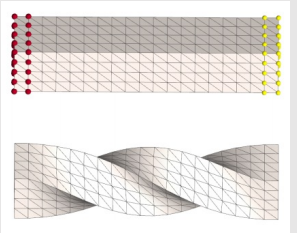
3. Set velocity

$$\vec{x}_{i+1} = \vec{\chi}_i, \qquad \vec{v}_{i+1} = \frac{(\vec{x}_{i+1} - \vec{x}_i)}{dt}$$

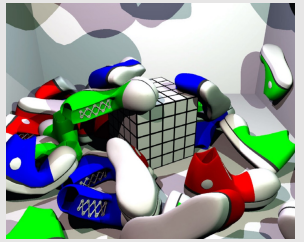
4. Goto 1

Reference

- Olga Sorkine and Marc Alexa. 2007. ***As-rigid-as-possible surface modeling***. In Proceedings of the fifth Eurographics symposium on Geometry processing (SGP '07).



- Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. 2005. ***Meshless deformations based on shape matching***. In ACM SIGGRAPH 2005 Papers (SIGGRAPH '05).



- Wikipedia: “Orthogonal Procrustes problem”, https://en.wikipedia.org/wiki/Orthogonal_Procrustes_problem

Affine Transformation

Affine Transformation

Linear transformation & translation

$$\vec{x}' = K\vec{x} + \vec{t}$$

Affine transformation

$$\begin{pmatrix} \vec{x}' \\ 1 \end{pmatrix} = \begin{bmatrix} K & \vec{t} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \vec{x} \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{bmatrix} K_{xx} & K_{xy} & K_{xz} \\ K_{yx} & K_{yy} & K_{yz} \\ K_{yz} & K_{zy} & K_{zz} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{bmatrix} K_{xx} & K_{xy} & K_{xz} & t_x \\ K_{yx} & K_{yy} & K_{yz} & t_y \\ K_{zx} & K_{zy} & K_{zz} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Properties of Affine Transformation

- Composite of two affine transformations makes an affine transformation

$$\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Associative property: $(A_1 A_2) A_3 = A_1 (A_2 A_3)$

- Inverse is also an affine transformation

$$\begin{bmatrix} K^{-1} & -K^{-1}\vec{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} K & \vec{t} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & 1 \end{bmatrix}$$

Useful Property of Associative Law

Associative law for matrix: $A(BC) = (AB)C$

$$E\left(D\left(C\left(B(Ax)\right)\right)\right) = \underbrace{(EDCBA)}_{K}x$$

Precompute $K = EDCBA$ to efficiently compute Kx for various x

Articulated Rigid Body

intrinsic rotation:
axis rotated

extrinsic rotation:
axis fix

$$R(Q\vec{\omega}) = Q * \boxed{R(\vec{\omega})} * Q^T$$

$$R(Q\vec{\omega}) * Q = Q * R(\vec{\omega})$$

matrix don't commute!

Rotation around Rotated Axis: Robotic Arm

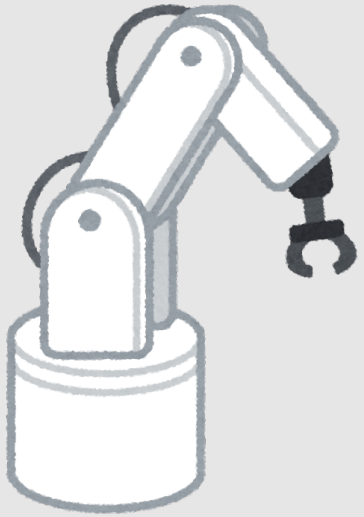
- Rotation of end-effector in a 4-link articulated body

intrinsic

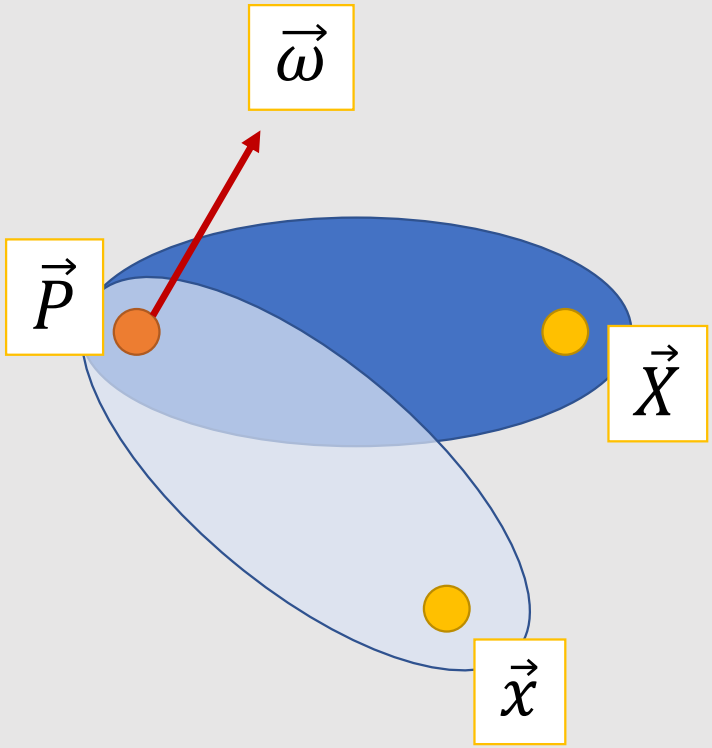
$$R_4(R_3R_2R_1\vec{\omega}_4) * R_3(R_2R_1\vec{\omega}_3) * R_2(R_1\vec{\omega}_2) * R_1(\vec{\omega}_1)$$

extrinsic

$$R_1(\vec{\omega}_1) * R_2(\vec{\omega}_2) * R_3(\vec{\omega}_3) * R_4(\vec{\omega}_4)$$



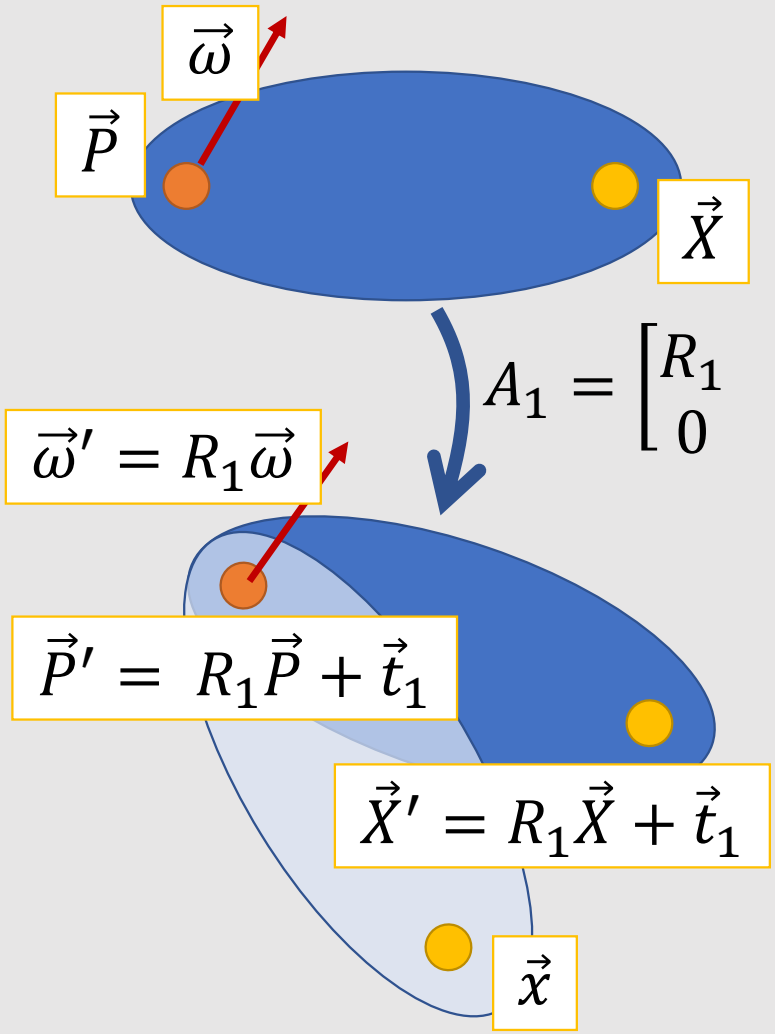
Affine Trans. of Rotation Around Fixed Point



$$\vec{x} = R(\vec{\omega})\left(\vec{X} - \vec{P}\right) + \vec{P}$$

$$\begin{pmatrix}\vec{x}\\1\end{pmatrix} = \underbrace{\begin{bmatrix}R(\vec{\omega}) & \vec{P} - R(\vec{\omega})\vec{P}\\0 & 1\end{bmatrix}}_{A\left(\vec{\omega},\vec{P}\right)}\begin{pmatrix}\vec{X}\\1\end{pmatrix}$$

Rotation After Rigid Transformation

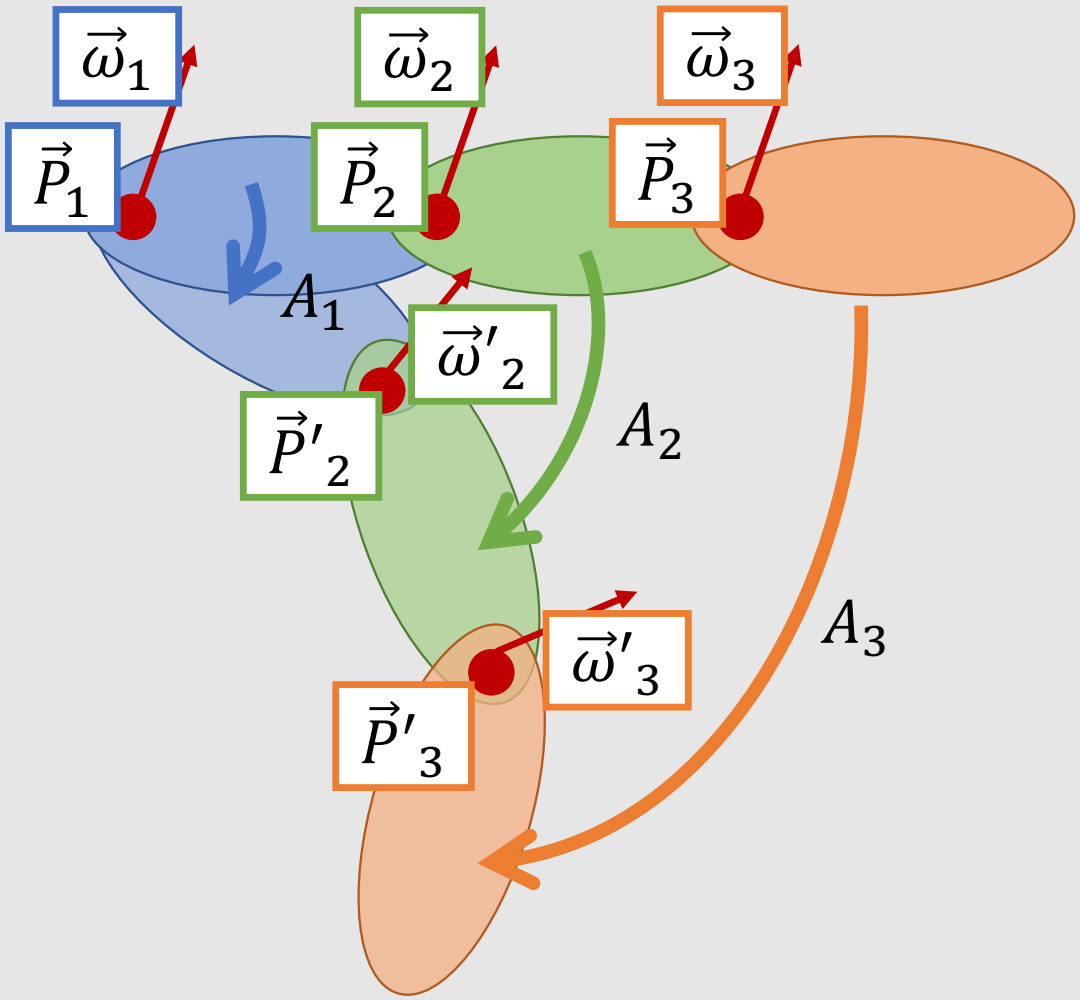


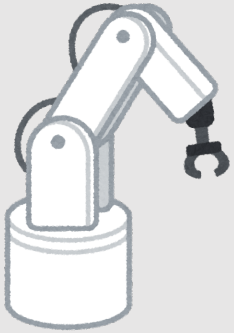
$$\begin{pmatrix} \vec{x} \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} R(\vec{\omega}') & \vec{P}' - R(\vec{\omega}')\vec{P}' \\ 0 & 1 \end{bmatrix}}_{A(\vec{\omega}', \vec{P}')} \underbrace{\begin{bmatrix} R_1 & \vec{t}_1 \\ 0 & 1 \end{bmatrix}}_{A_1} \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix}$$

$$\begin{aligned} \vec{x} &= R(\vec{\omega}')(\vec{X}' - \vec{P}') + \vec{P}' \\ &= R(R_1\vec{\omega})(R_1\vec{X} + \vec{t}_1 - R_1\vec{P} - \vec{t}_1) + R_1\vec{P} + \vec{t}_1 \\ &= R(R_1\vec{\omega})R_1(\vec{X} - \vec{P}) + R_1\vec{P} + \vec{t}_1 \\ &= R_1R(\vec{\omega})(\vec{X} - \vec{P}) + R_1\vec{P} + \vec{t}_1 \end{aligned}$$

$$\begin{pmatrix} \vec{x} \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} R_1 & \vec{t}_1 \\ 0 & 1 \end{bmatrix}}_{A_1} \underbrace{\begin{bmatrix} R(\vec{\omega}) & \vec{P} - R(\vec{\omega})\vec{P} \\ 0 & 1 \end{bmatrix}}_{A(\vec{\omega}, \vec{P})} \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix}$$

Affine Transformation of Articulated Body





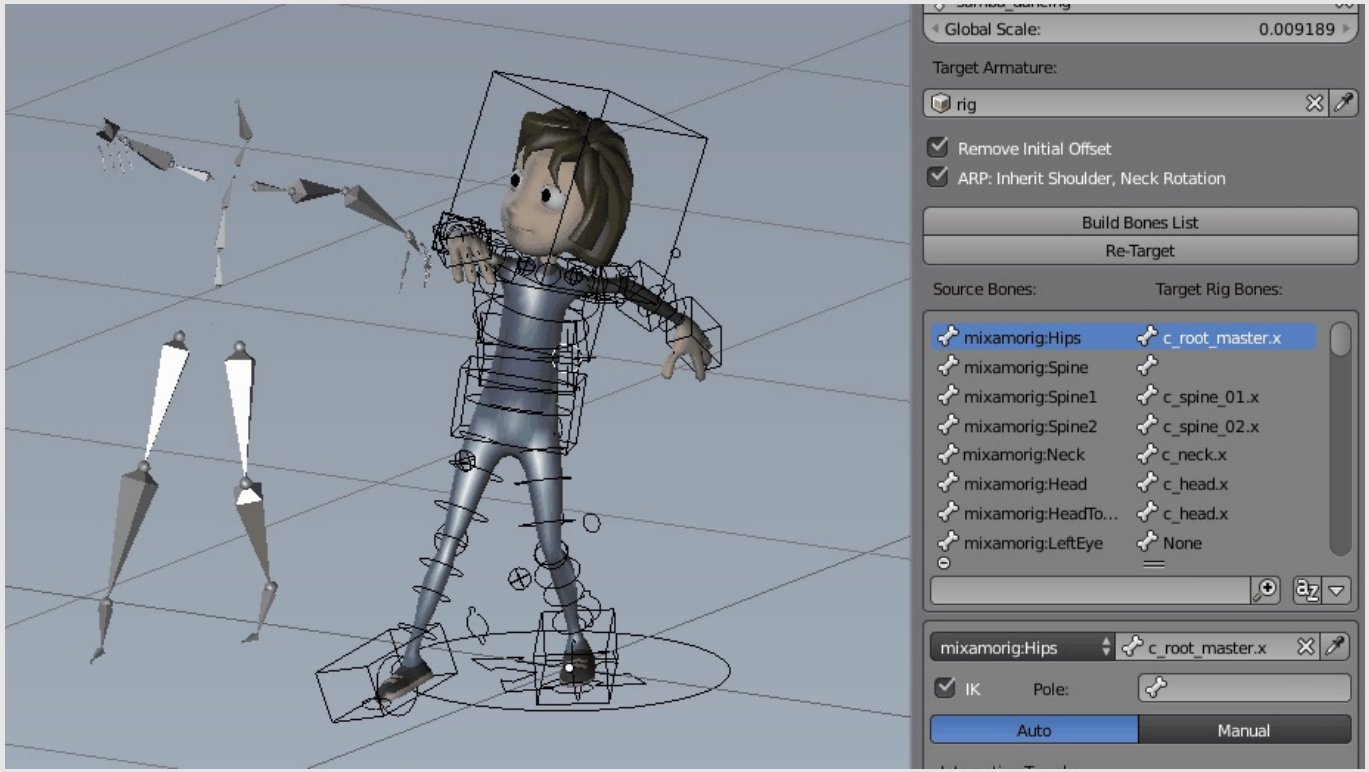
$$A_1 = A(\vec{\omega}_1, \vec{P}_1)$$

$$\begin{aligned} A_2 &= A(\vec{\omega}'_2, \vec{P}'_2)A_1 \\ &= A_1 A(\vec{\omega}_2, \vec{P}_2) \\ &= A(\vec{\omega}_1, \vec{P}_1)A(\vec{\omega}_2, \vec{P}_2) \end{aligned}$$

$$\begin{aligned} A_3 &= A(\vec{\omega}'_2, \vec{P}'_2)A_2 \\ &= A_2 A(\vec{\omega}_2, \vec{P}_2) \\ &= A(\vec{\omega}_1, \vec{P}_1)A(\vec{\omega}_2, \vec{P}_2)A(\vec{\omega}_3, \vec{P}_3) \end{aligned}$$

Linear Blend Skinning

Linear Blend Skinning is Industry Standard



Source “Auto-Rig Pro: Remap Update!
https://www.youtube.com/watch?v=-tuuijd2fCc

Linear Blend Skinning: Rest Pose

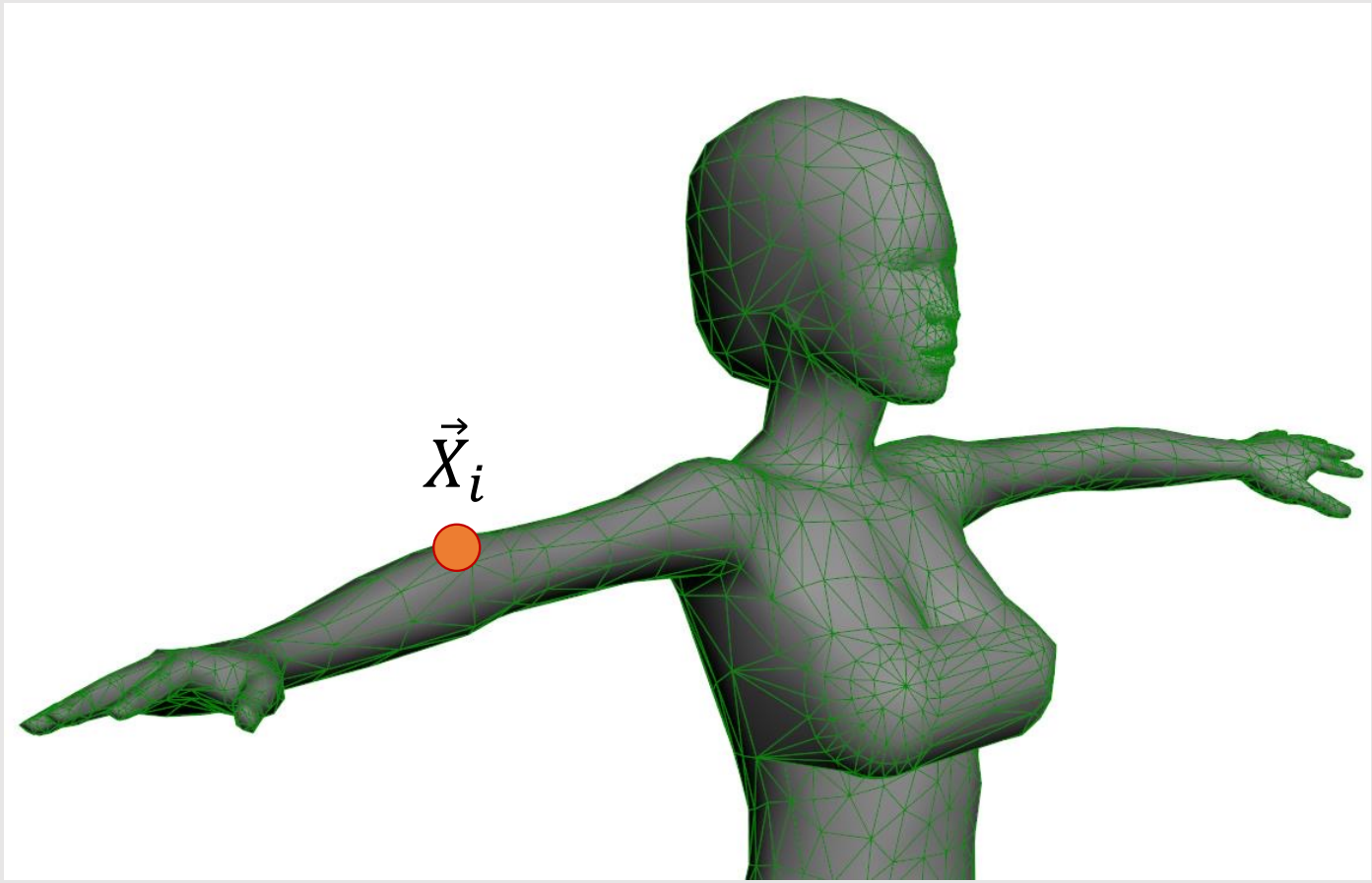


Image Credit: Ladislav Kavan (https://skinning.org/direct-methods-slides.pdf)

Each Bone Has Affine Transformation

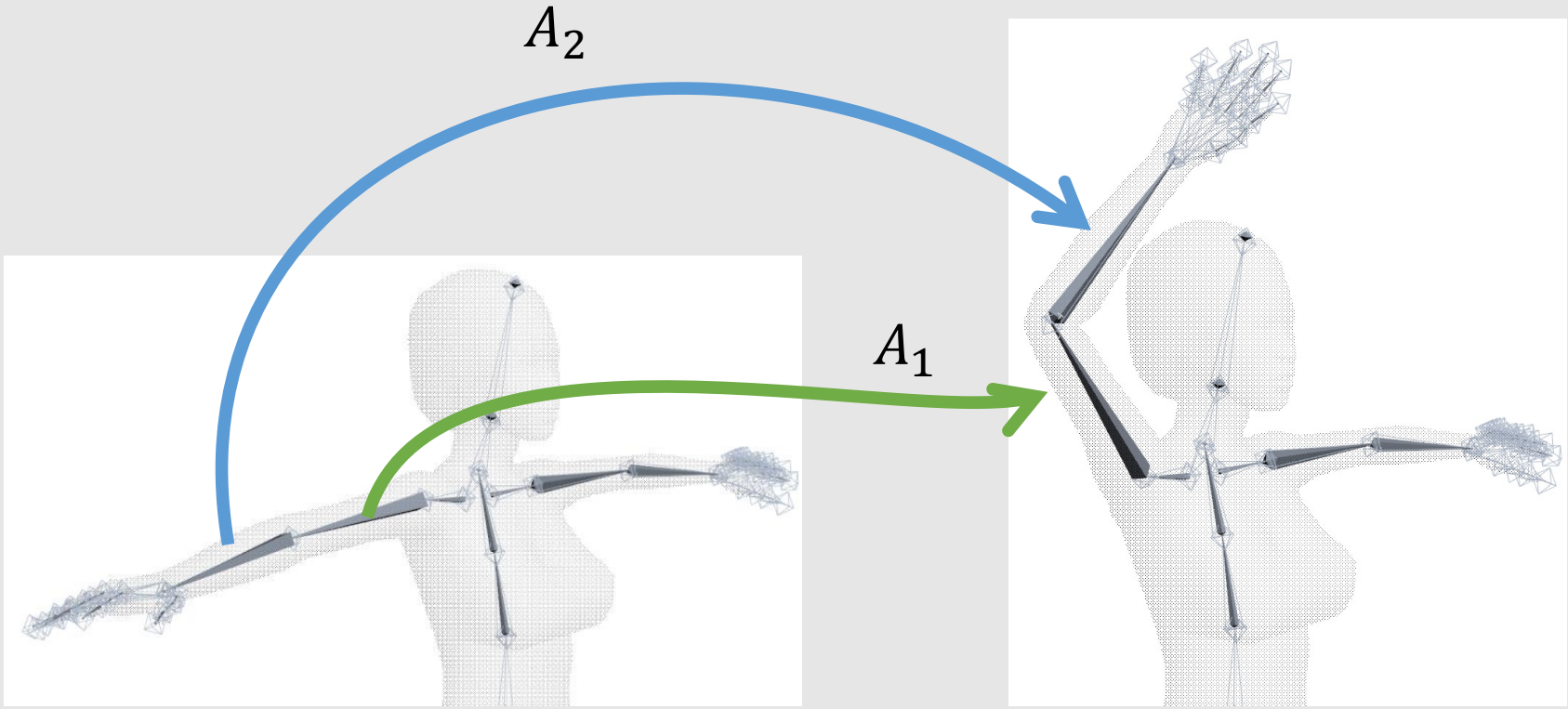
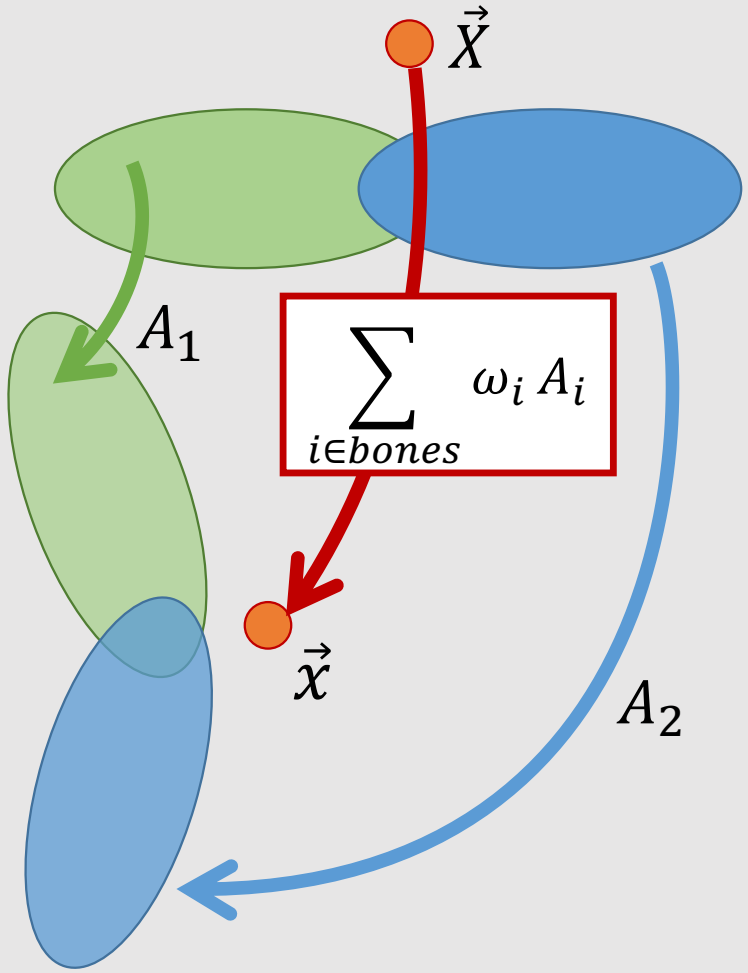


Image Credit: Ladislav Kavan (https://skinning.org/direct-methods-slides.pdf)

Linear Blend Skinning: Weighted Affine Trans.



$$\begin{pmatrix} \vec{x} \\ 1 \end{pmatrix} = \sum_{i \in bones} \omega_i A_i \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix}$$

Skinning Weight: Weight Defined on Vertices

Weights for upper arm: ω_1

Weights for lower arm: ω_2

Image Credit: Ladislav Kavan (https://skinning.org/direct-methods-slides.pdf)

Reference

- Skinning: Real-time Shape Deformation, ACM SIGGRAPH 2014 Course, https://skinning.org/