



# Skinning

Animation baserat på en "benstruktur" som "huden", 3D-modellen, deformeras efter.



# Koordinatsystem!

Samma grundproblem som bump mapping: En fråga om koordinater och transformationer

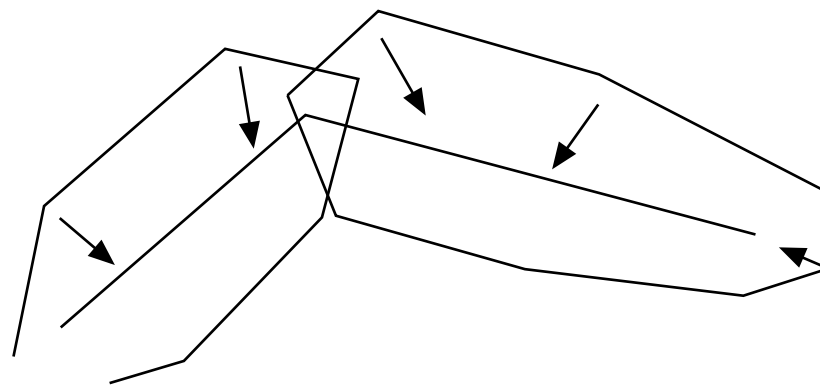
...fast lite fler.



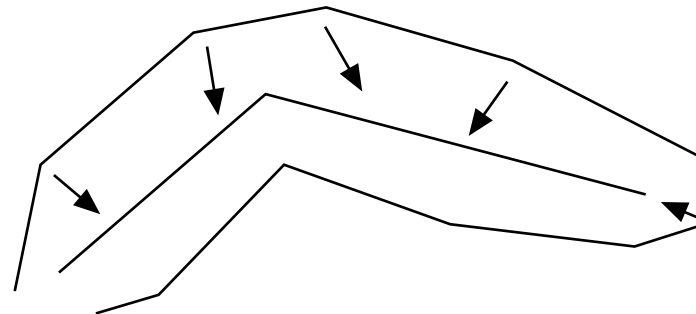
## Tre sätt att animera en modell

- Parenting
- Stitching
- Skinning

Parenting: Separata stela meshar följer varje ben



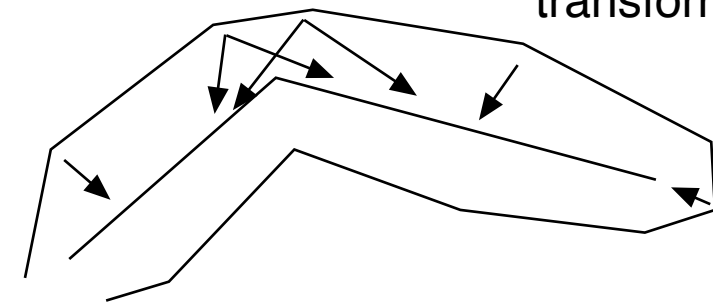
Stitching: Varje vertex följer ett ben



Skinning: En vertex kan viktas efter flera ben

Två närliggande ben, följ båda med lämplig viktning

En närliggande, följ benets transform





## Avancerad skinning

Grundläggande algoritmen har vissa svagheter

<- Collapsing  
elbow

Twist ->

Kan åtgärdas med

- Flera ben i en och samma led
- Shape blending - flera modeller som man morphar mellan
- Volymbevarande tilläggs villkor



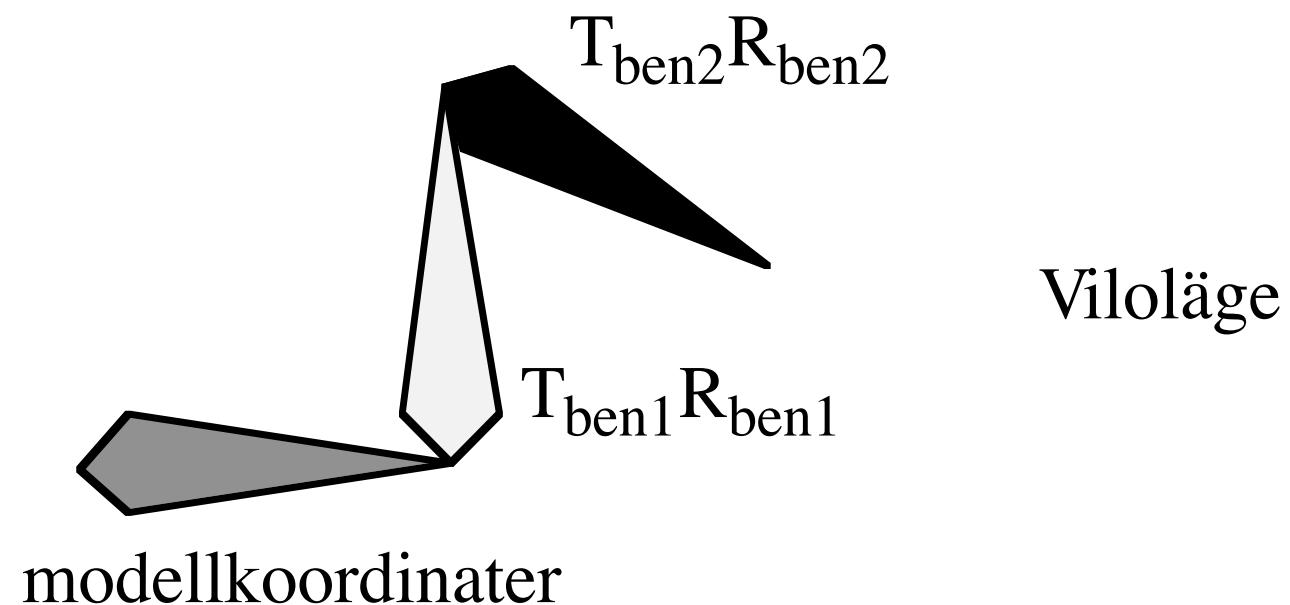
## Koordinatsystemen

Mesh given i modellkoordinater

Mesh och skelett givna i viloläge

Hur påförs animation?

Hur beräknar man en vertex modifierade position?





# Information Coding / Computer Graphics, ISY, LiTH

Varje bens position definieras av en transformation, en translation och en rotation:

$$M_{\text{ben2}} = T_{\text{ben2}} R_{\text{ben2}}$$

Transformation av vertex från modellkoordinater till benkoordinater:

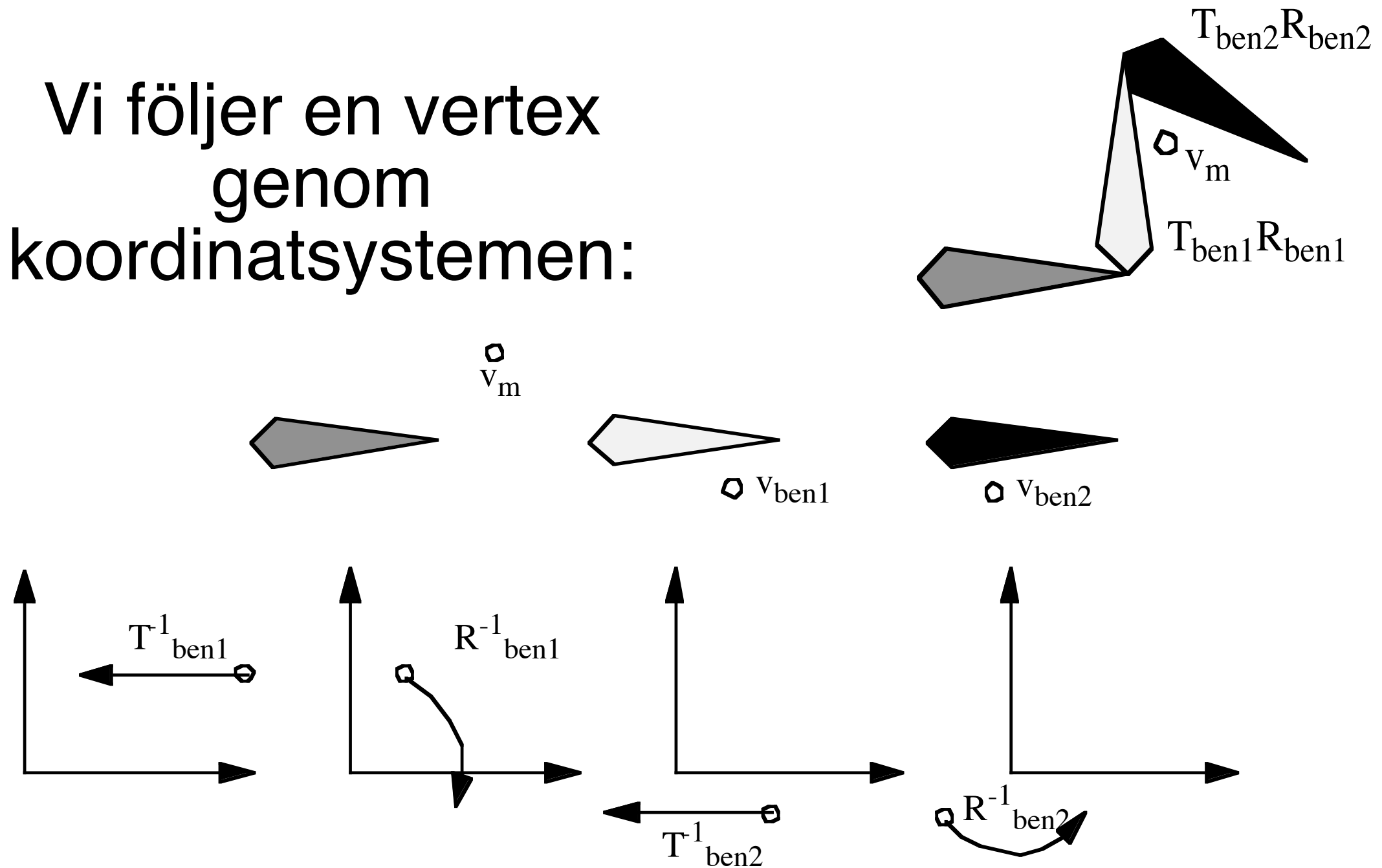
$$V_{\text{ben2}} = M_{\text{ben2}}^{-1} M_{\text{ben1}}^{-1} V_{\text{m}}$$

Transformation av vertex från benkoordinater till modellkoordinater:

$$V_{\text{m}} = M_{\text{ben1}} M_{\text{ben2}} V_{\text{ben2}}$$



Vi följer en vertex genom koordinatsystemen:





## Animation:

Transformera till benkoordinat med viloläget.

Transformera tillbaka till modellkoordinater med modifierade benpositioner

$$\mathbf{v}'_m = \mathbf{M}'_{ben1} \mathbf{M}'_{ben2} \mathbf{M}^{-1}_{ben2} \mathbf{M}^{-1}_{ben1} \mathbf{v}_m$$



## Animation oftast rotationer:

Vilolägestransformation är vilolägets rotation och translation

$$M_{\text{ben}} = T_{\text{vila}} R_{\text{vila}}$$

På denna tillkommer sedan animationens rotation

$$M'_{\text{ben}} = M_{\text{ben}} R_{\text{anim}} = T_{\text{vila}} R_{\text{vila}} R_{\text{anim}}$$



## Traversera skelettet för resulterande transformation:

Modellkoordinater till benkoordinater:

$$M_{mb} = \prod M_{ben,i}^{-1}$$

Benkoordinater till modellkoordinater:

$$M_{bm} = \prod M_{ben,i} \cdot R_{anim,i}$$

och hela transformationen kan skrivas

$$v' = M_{bm} M_{mb} \cdot v$$



## Viktning av vertexar

$$v' = \sum_{i=1}^n w_i M_i v$$

$M_i$  beräknas, enligt ovan, som

$$M_{mbi} = \prod_{j=1}^i M_{ben,j}^{-1}$$

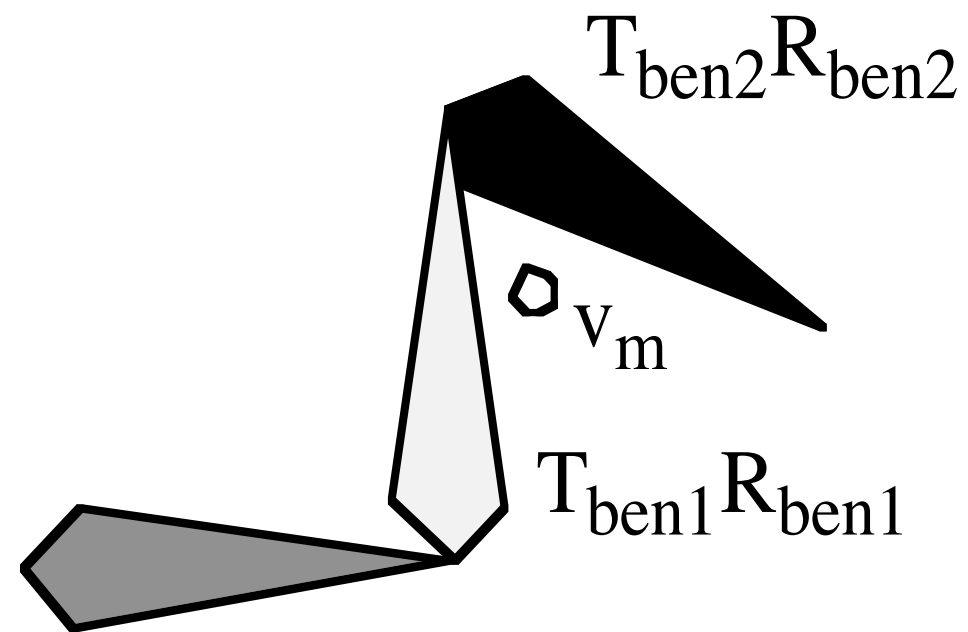
$$M_{bmi} = \prod_{j=1}^i M_{ben,j}^{-1} \cdot R_{anim,j}$$

$$M_i = M_{bmi} M_{mbi}$$



# Information Coding / Computer Graphics, ISY, LiTH

Exemplet med två ben ovan. Antag att  $v_m$  påverkas av ben1 och ben2.



$$M_1 = M'_{ben1} M^{-1}_{ben1}$$

$$M_2 = M'_{ben1} M'_{ben2} M^{-1}_{ben2} M^{-1}_{ben1}$$

$$v'_m = \sum_{j=1}^2 w_j M_j v = w_1 M_1 v_m + w_2 M_2 v_m$$



## Beräkna allt i rätt domän

Matriserna kan beräknas per ben.

$$\begin{aligned}M_{mb} &= \prod M_{ben,i}^{-1} \\M_{bm} &= \prod M_{ben,i} R_{anim,i} \\M_{tot} &= M_{bm} M_{mb}\end{aligned}$$

Transformationen av vertex görs per vertex.

$$v' = M_{tot} \cdot v$$



## I shader

Per ben-operationer: Görs på CPU

Per vertex-operationer: Görs i vertex shader

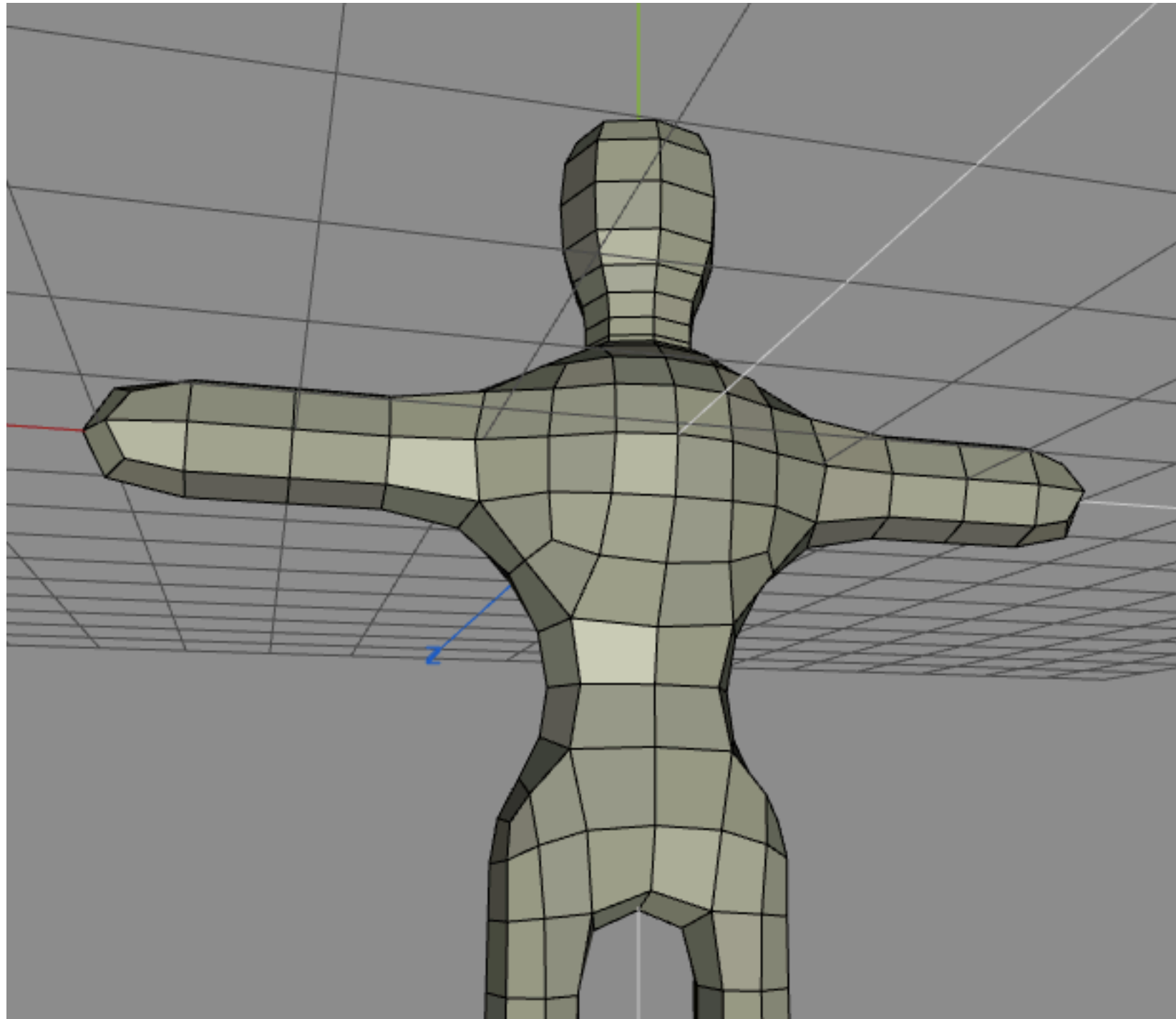
Matriserna kan skickas som uniform

Vikterna  $w_i$  är per vertex, skickas som attribute arrays

Undvik att deformera hela objektet på CPU. Då måste hela modellen laddas om varje frame. Vitalt för stora modeller.



## Information Coding / Computer Graphics, ISY, LiTH



Modeller skapas  
typiskt i ett lämpligt  
"viloläge"

"Reference pose", även  
känt som "T-pose" eller  
"A-pose"



## Information Coding / Computer Graphics, ISY, LiTH

Liten anekdot: "Civilian pose" i TF2 kommer från en tidig karaktär, "the civilian"

Refererar idag till att hitta buggar som får karaktärer att låsa sig i referensläget.



## Information Coding / Computer Graphics, ISY, LiTH

Var kommer vikterna ifrån?  
Animationsparametrar - hur ändras rotationerna?

Inbyggt i 3D-program, t.ex Blender

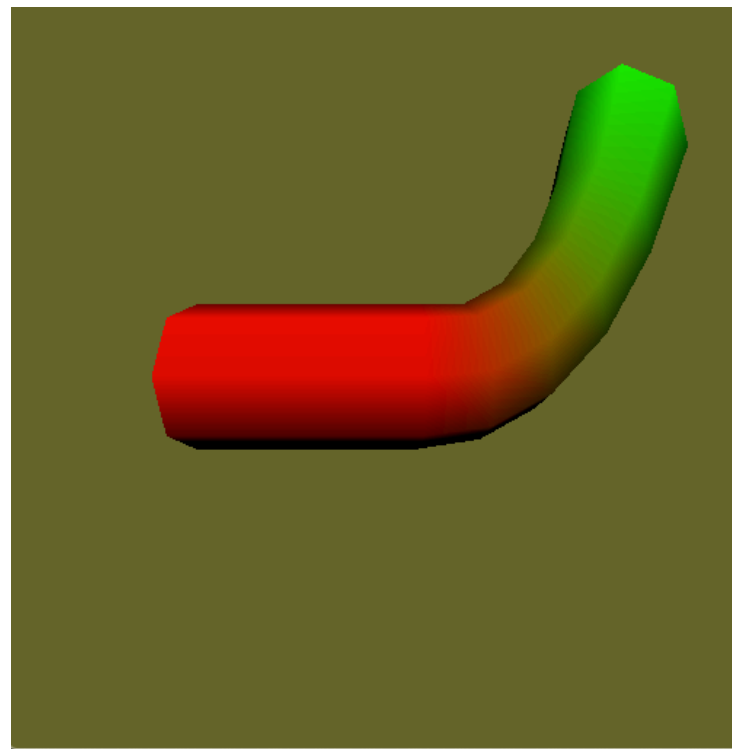
Exempel från atspaces  
skinning tutorial



## Labb 2

Enkla fall: Slang

Inga grenar. Enkla vikter längs slangen.





# Projekt på skinning?

Massor av möjligheter!

Skinning på mer komplex modell än labben

Skinning + animationsdata

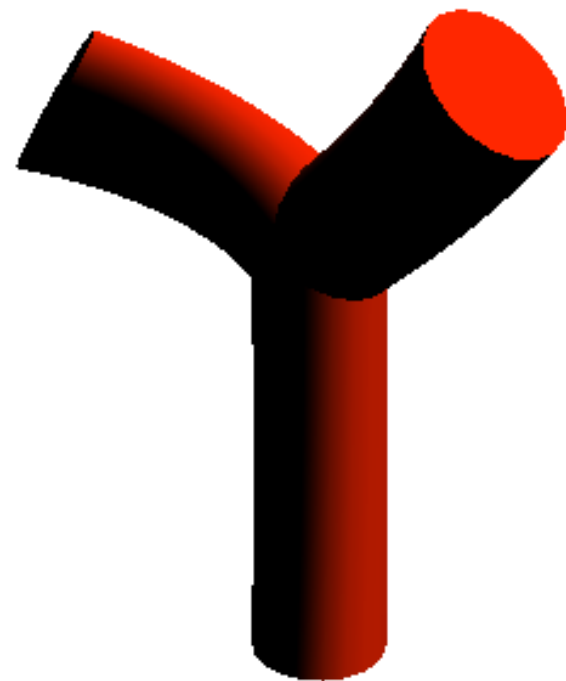
Förbättrad skinning, undersök någon metod eller jämför flera



## GLUGGW möjligt verktyg?

GLUGG med stöd för skinningvikter.

Prototyp! Osäker mark. Men mycket hjälp från mig på grund av detta.



Vad är målet? GLUGGW är i grunden till för att experimentera med procedurrell generering av geometri och skinningvikter. Men om du vill jobba från färdig riggning då...?



## Preliminär ASSIMP-laddare

OBJ-filer är bra... men kan inte innehålla skinningvikter!

Assimp (asset importer) kan ladda t.ex. Collada.

**MEN Assimp är ett elände att använda!**

Jag har preliminära Assimp-laddare, inklusive laddare med vikter.

