



Course Plan



- lec. 1: **Introduction** ●
- lec. 2: **Mathematics** for 3D Games ●●●●●●●
- lec. 3: **Scene Graph** ●
- lec. 4: **Game 3D Physics** ●●●●+●●
- lec. 5: **Game Particle Systems** ●
- lec. 6: **Game 3D Models** ●●
- lec. 7: **Game Textures** ●●
- lec. 9: **Game Materials** ●
- lec. 8: **Game 3D Animations** ●●●●● (with a red arrow pointing to the 5th dot)
- lec. 10: **Networking** for 3D Games ●
- lec. 11: **Artificial Intelligence** for 3D Games ●
- lec. 12: **3D Audio** for 3D Games ●
- lec. 13: **Rendering Techniques** for 3D Games ●

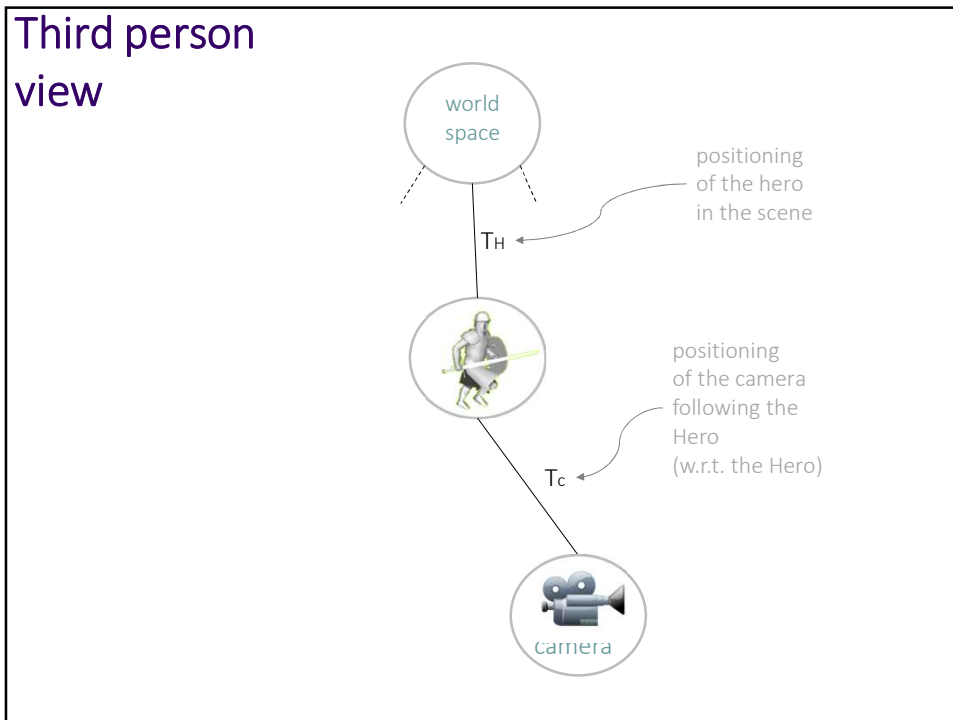
168

Integration of skeletons in the scene-graph

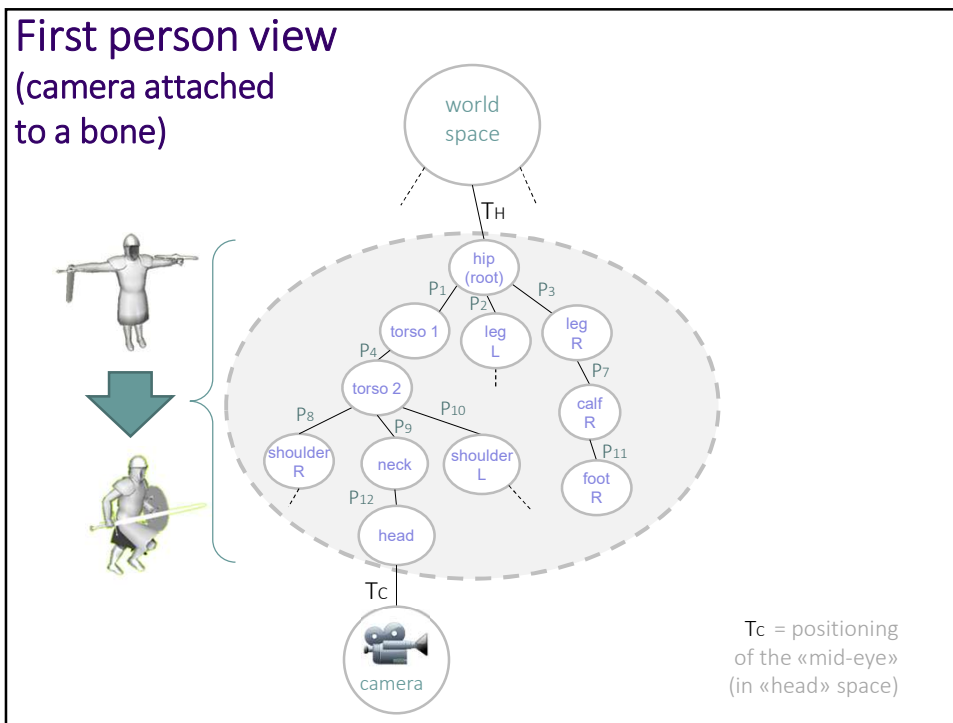


- A skeleton (rig) can be considered just a **subtree** in the **scene-graph**
 - only, its local transforms are defined by the current frame in the current skeletal animation asset
- Examples of common reasons to use the skeleton just as a part of the scene graph:
 - Placement of the camera in a bone (e.g. the head bone)
 - Defining geometry proxies (hit-boxes) for collision detection in bones

169



170



171

Per-bone collision proxies

Hit-boxes: e.g., capsules
(not necessarily in every bone)

172

(Pre)processing tasks for skeletal animations: examples

- Keyframe sparsification
 - input: animation with N keyframes
 - output: animation with $M < N$ keyframes
- Animation Retargeting
 - input: SkelA + Animation for SkelA + Skel2
 - output: a similar Animation, but for Skel2
- Automatic generation, from a blend-shape
 - input: a blend-shape
 - output: Skeleton + Skinned Mesh + Anim
 - note: the opposite is a trivial (it's a form of *baking*)

173

Sparsification of keyframes (reduce number of keyframes)



- Objective: removal of redundant keyframes
 - “Redundant” = can be approximated by in-betweens
 - A preprocessing task
- Basic algorithm concept:
 - for each keyframe P_x
 - tentatively remove P_x
 - compute interpolated version P_i from remaining keyframes
 - the prev. and next ones
 - if $distance(P_i, P_x) > MAX_ERR$ then reinsert keyframe P_x

174

Ragdolling (notes)



- Idea: let a physical simulation determine the evolution of the skeleton (and attached geom. proxies)
 - Includes: gravity, external forces, collisions with other objects, self-collisions (i.e., collision between proxies associated to the bones)
- Ingredients:
 - Per-bone proxies (in at least a subset of the bones)
 - Constraints, such as... attachments of bones, constraint on rotations (e.g., “knees don’t bend backward or sideways”)
 - The latter can be expressed as positional constraint in a Position Based Dynamics simulation
- Result: procedural skeletal animation!

175

Observation: Blend-shapes & Skeletal Animations blend well with...



- **Texturing!**
 - **UV coords** are only defined in rest shape
 - they are shared by all frames
 - **Textures** are shared by all frames
- **Micro-meshes!**
 - **Blend-shapes:**
Base mesh and displacement directions are both defined per morph-target
 - **Skeletal animations:**
skinning is defined on base-mesh only,
deforms both vertex positions and displacement dirs.
 - *Both:* micro-displacem. are applied on top of animated mesh

176

Observation: blendshapes + skeletal anims *can be used together*



- A blend shape can be skinned!
- Both animations can be combined
 - frame of the blend shape
≠
frame of the skeletal animation
- Examples:
 - Breathing animations = blend shape,
+ Idle animation = skeletal anim
 - Cheeks puffing = blend shape (face morph)
+ mandible bone = skeletal animations
 - Blend shapes correctives (see later)

177

Observation: blandshapes + skeletal anims *can be used together*

Connectivity (shared):				geometries:				shared attributes:		
Tri:	Wedge 1:	Wedge 2:	Wedge 3:	Vert:	Base Shape	Shape 1	Shape 2	Shape ...	UV	Bone links (skinning) ...
T1	4	1	2	V1	(x, y, z)	(x, y, z)	(x, y, z)	...	(u, v)	(b ₀ , w ₀ , b ₁ , w ₁ , b ₂ , w ₂)
T2	4	2	5	V2	(x, y, z)	(x, y, z)	(x, y, z)	...	(u, v)	(b ₀ , w ₀ , b ₁ , w ₁ , b ₂ , w ₂)
T3	5	2	3	V3	(x, y, z)	(x, y, z)	(x, y, z)	...	(u, v)	(b ₀ , w ₀ , b ₁ , w ₁ , b ₂ , w ₂)
				V4	(x, y, z)	(x, y, z)	(x, y, z)	...	(u, v)	(b ₀ , w ₀ , b ₁ , w ₁ , b ₂ , w ₂)
				V5	(x, y, z)	(x, y, z)	(x, y, z)	...	(u, v)	(b ₀ , w ₀ , b ₁ , w ₁ , b ₂ , w ₂)

178

Rendering Skinning + Blendshapes

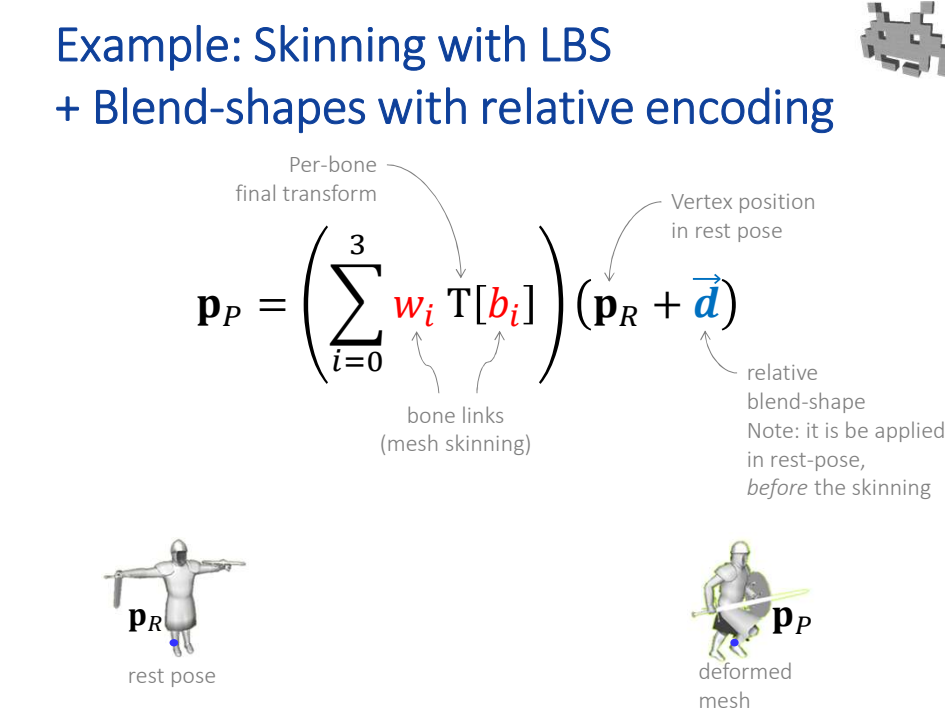
To render a mesh...

- Load...
 - make sure all data is ready in GPU RAM
 - Geometries + Attributes
 - Connectivity
 - Pose**
final transforms per bone
 - Textures
 - Shaders
 - Material Parameters...
- ...and Fire!
 - issue the Draw Call

including **skinning weights**

179

Example: Skinning with LBS + Blend-shapes with relative encoding


$$\mathbf{p}_P = \left(\sum_{i=0}^3 w_i \mathbf{T}[b_i] \right) (\mathbf{p}_R + \vec{d})$$

Per-bone final transform

Vertex position in rest pose

relative blend-shape
Note: it is applied in rest-pose, before the skinning

bone links (mesh skinning)

\mathbf{p}_R rest pose

deformed mesh \mathbf{p}_P

180

Limits of skinning (both LBS and DQS)

Notes:

The bar for 3D game quality has gone up, but skeletal animations + skinning stayed the same for 10+ years.

Problems with deformations of the rest pose mesh:

- Does not account for **Dynamic effects**:
e.g., a fat belly jiggling up and down during a run
 - Solution 1: use blend shapes (e.g., blend shape correctives)
 - Solution 2: add new bones (belly bone), add a dynamic simulation to control the bones (aka to control the “secondary motions”)
- Does not account for **collision/contacts**
- Does not account for **volume preservation**
 - E.g., no muscle bulging
 - Can be in part compensated with skillful edit of bone weights

181

Limits of skeletal animations

notes:



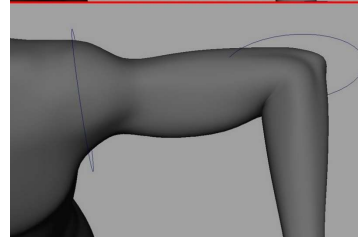
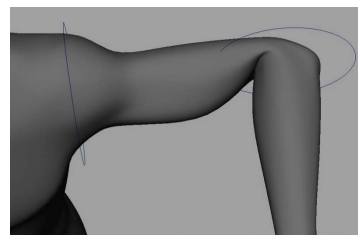
The bar for 3D game quality has gone up, but skeletal animations + skinning stayed the same for almost 10 years.

Problems with the skeletal animations:

- Transitions between animation is crude, can look robotic
 - Possible solution: use ad-hoc animations for transitions
- Ragdolling = completely death “sack of potatoes”
Authored Animation = character completely alive and in control, irresponsive to actual forces / dynamics
 - What about intermediate situations?
- IK not necessarily realistic
 - E.g.: feet are placed on the ground (not into it), but this is not how you would walk over a rugged terrain
- Animations are not physically based

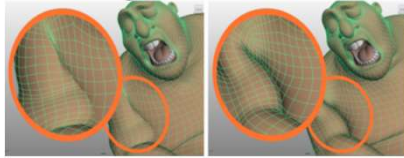
182

Blend Shape Correctives



183

Research topic: Deformation beyond standard skinning



*Efficient Elasticity for Character Skinning
with Contact and Collisions*
Aleka McAdams et al (Disney animation)
SIGGRAPH 11

*Note: usually way more complex than direct methods (LBS / DQS).
More offline animation oriented than videogames*

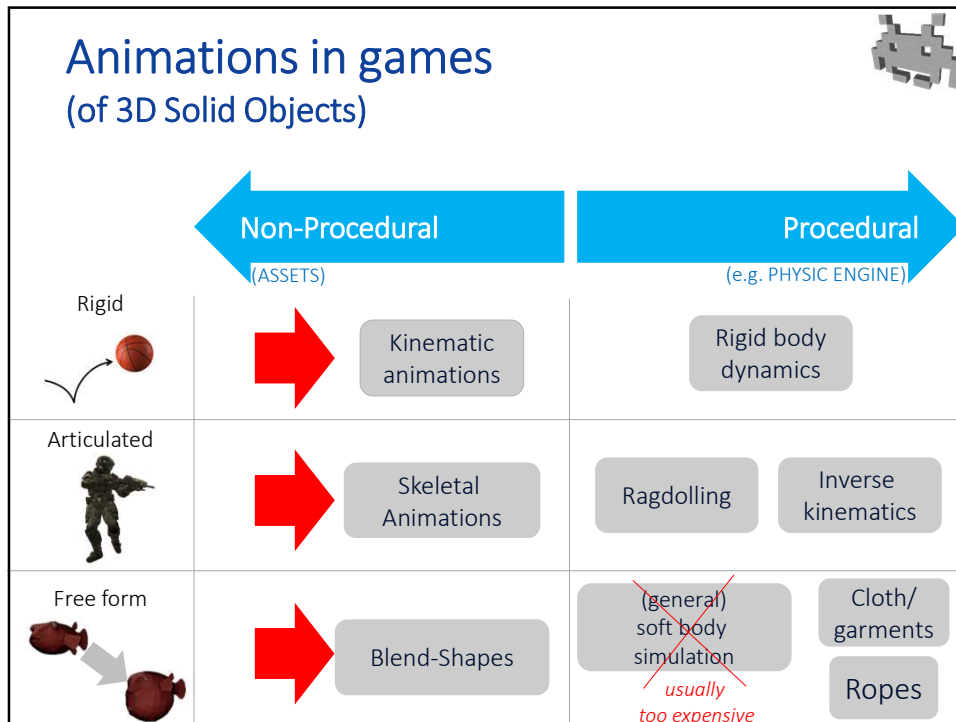
184

Research topic: better interfaces to author animations

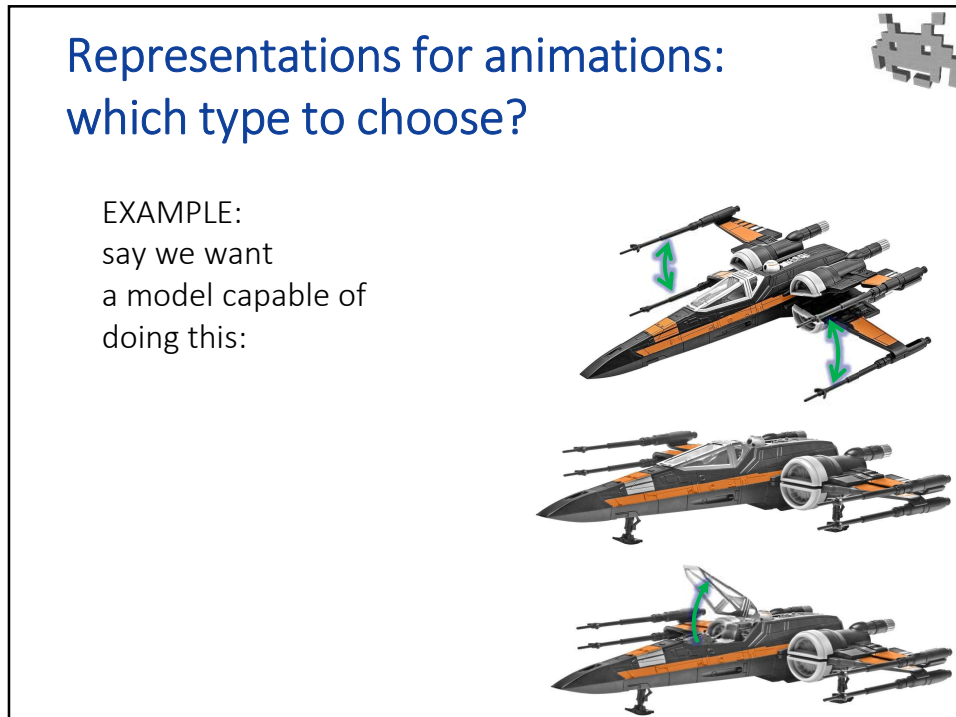


*Tangible and Modular Input Device for Character
Articulation*
Alec Jacobson, Daniele Panozzo, Oliver Glauser,
Cedric Pradalier, Otmar Hilliges, Olga Sorkine-
Hornung
SIGGRAPH 2014

185



187



188

Representations for animations: which type to choose?

solution 1: Kinematic animation

animate these!

scene graph

"wing" mesh (2 instances)
 "windscreen" mesh
 "hull" mesh
 "wing" mesh (2 instances)

189

Representations for animations: which type to choose?

solution 2: Skeletal animation


x-wing rig

x-wing skinned mesh
 skeletal animations

190

Representations for animations: which type to choose?

solution 3: Blend-
shape



base shape
morph 1
morph 2


"x-wing" blend-shape

191

Representations for animations: which type to choose?

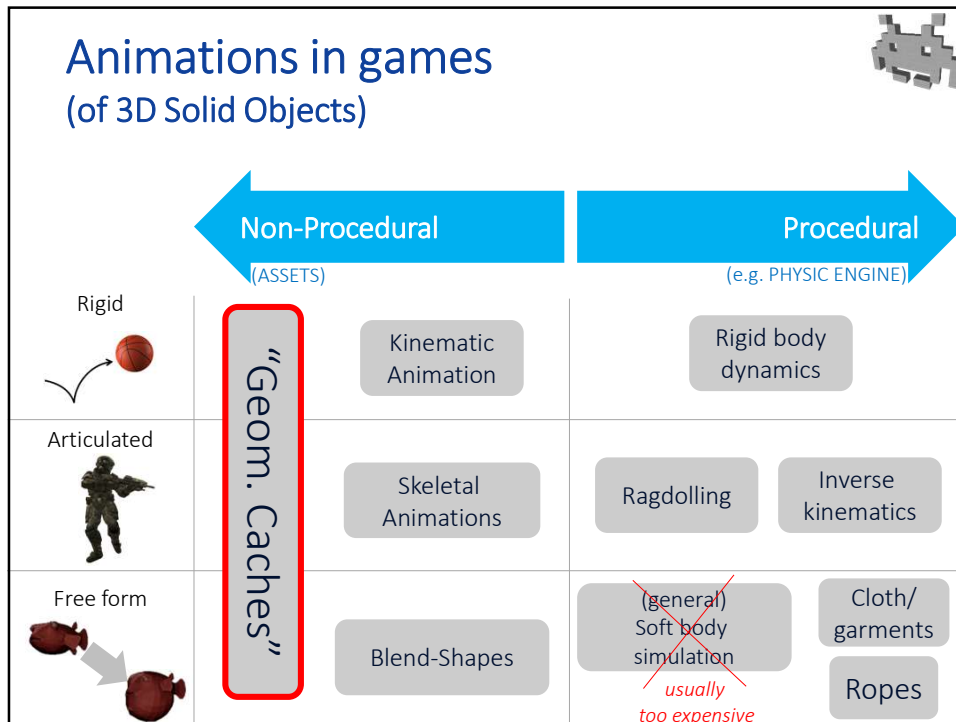
In this example:

- **Animation of transforms (of the scenegraph):**
 - *how:* 3 (rigid) meshes, 5 instances, animate scenegraph transforms
 - can reuse geometry for all wings: most compact on RAM ☺
 - simpler rendering
 - 5 separated draw calls! ☹
- **Skeletal animation:**
 - *how:* one rig + one skinned mesh + few skeletal animations
 - mesh skinning: single bone enough in this case
 - if very low poly mesh (few polys): a waste?
 - more taxing rendering (a bit) ☹
 - real time skinning on vertex any
 - single draw call! ☺
- **Blend shapes:**
 - *how:* blend shape with one base shape + 2 morphs
 - minimal impact
 - worst quality interpolation: linear
 - vertices on straight paths (unless, intermediate shapes are added)
 - heaviest on RAM ☹
 - (a waste of DoF!)
 - not important, if very low res
 - single draw call! ☺
 - but to different buffers each frame / or to a larger buffer



straight
(non curved)
paths

192



194

Geometry Caches (for lack of a better name)

- Baked, optimized animations
 - of a mixture of types, like
 - blend shapes
 - kinematic animations
 - skinned animations
 - optimized
 - compressed, streamed...
 - Can be used to bake results of a physical simulation
 - i.e., convert it from procedural to kinematic

one used file format:

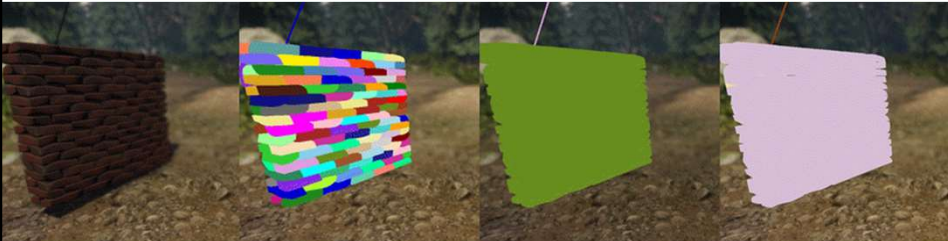
ALEMBIC
 by **imageworks**

SONY PICTURES

195


Geom. Caches (for lack of a better name)

- Baked, optimized animations
 - of the appropriate types including mixtures



Input: 170 Meshes 88400 Verts	as Pre-made Transforms : Meshes: 170 Data rate: 0.13 MB/s Draw calls: 170 (same ones each frame)	as a Blend Shape : Meshes: 1, with N shapes Data rate: 4.3 MB/s Draw calls: 1 (different one each frame)	as a Skeletal Animation : Meshes: 1, w skinning (*) Data rate: 0.13 MB/s Draw calls: 1 (same one each frame) (*) just 1 bone per vertex
-------------------------------------	---	---	---

Geometry Caches
(a subset of Alembic)

by  CRYENGINE®

196

Animations in Mecanim (Unity) (notes)

- Assets (models, animation, skeletons) imported as formats:
 - fbx, collada
- Keyframe sparsification, or reduction of num of links per vertex
 - available during import / builds
- «Animator Controller» module → deals with:
 - blending between animations: «**transitions**»
 - compositing animations: «**layers**»
 - e.g.: a layer overwrites upper body bones
 - and is nicely WYSIWYG and has a nice graph GUI
- Inverse Kinematic: with scripts (`Avatar.SetIKPosition`)
- Skeletons:
 - custom skeletons can be used (imported as assets)
 - OR, a standard built-in humanoid skeleton provided
 - ~21 bones
 - simplifies: rigging, ragdolling (predefined constrains), layers (predef. labelling)

197