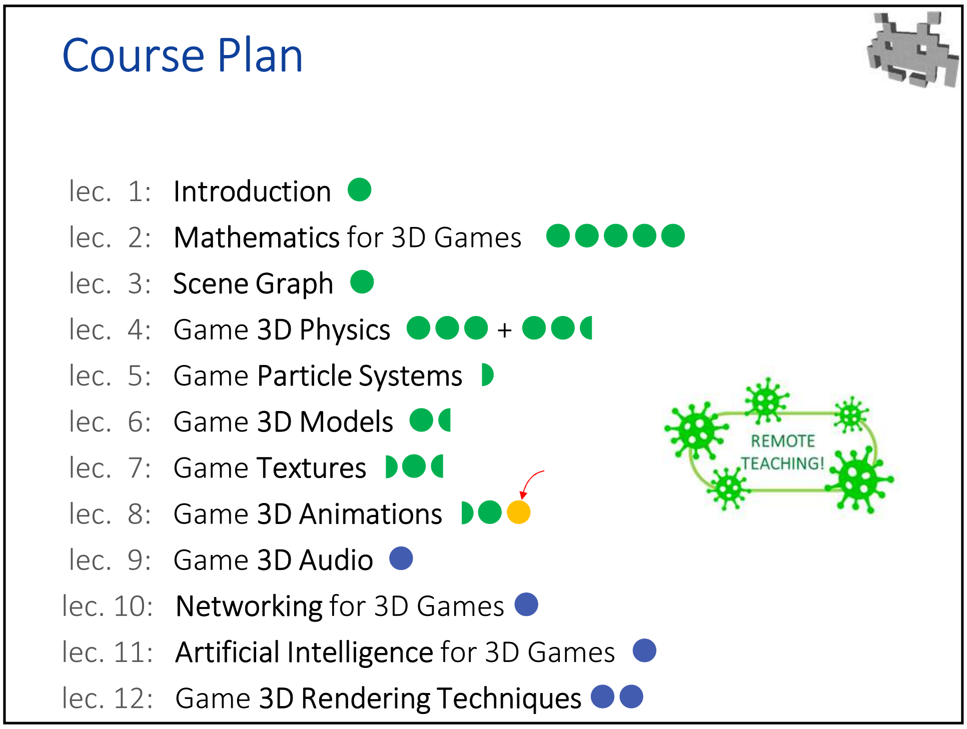
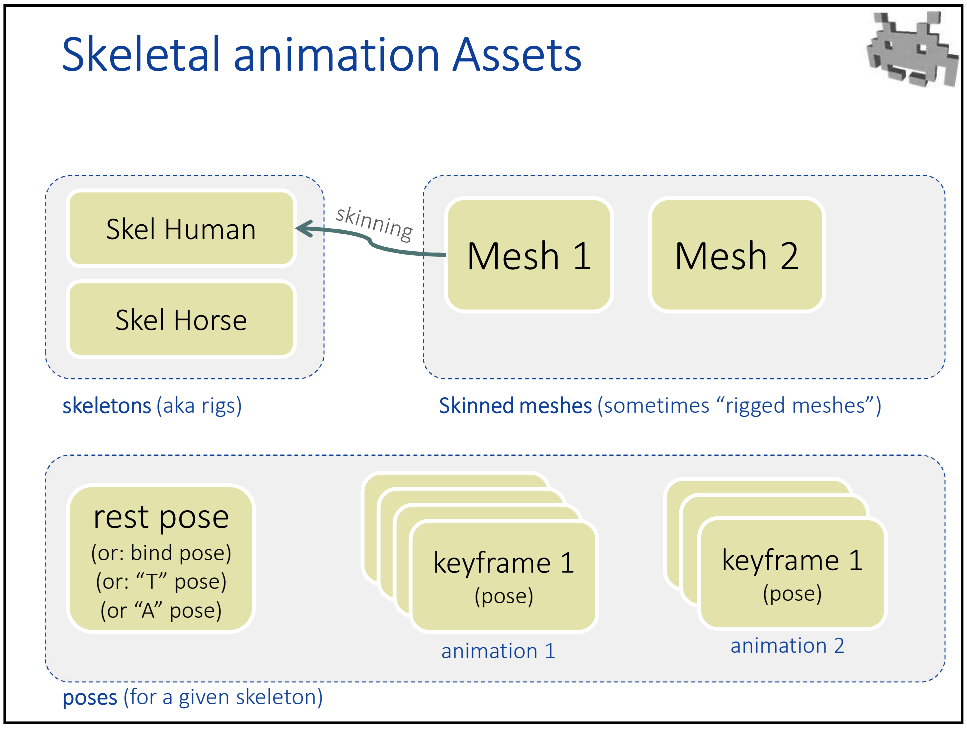


Course Plan

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. 8: Game 3D Animations ◗●●
lec. 9: Game 3D Audio ●
lec. 10: Networking for 3D Games ●
lec. 11: Artificial Intelligence for 3D Games ●
lec. 12: Game 3D Rendering Techniques ●●

REMOTE TEACHING!

117



Skeletal animation Assets

Skel Human

Skel Horse

skinning

Mesh 1

Mesh 2

skeletons (aka rigs)

Skinned meshes (sometimes “rigged meshes”)

rest pose
(or: bind pose)
(or: “T” pose)
(or “A” pose)

keyframe 1
(pose)

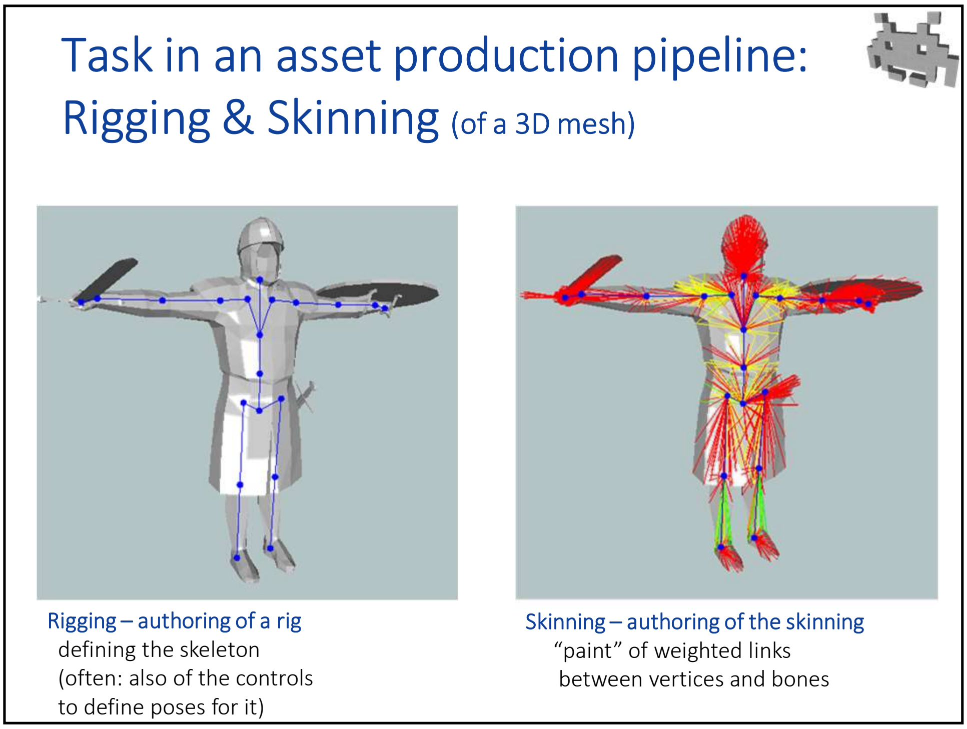
animation 1

keyframe 1
(pose)

animation 2

poses (for a given skeleton)

118

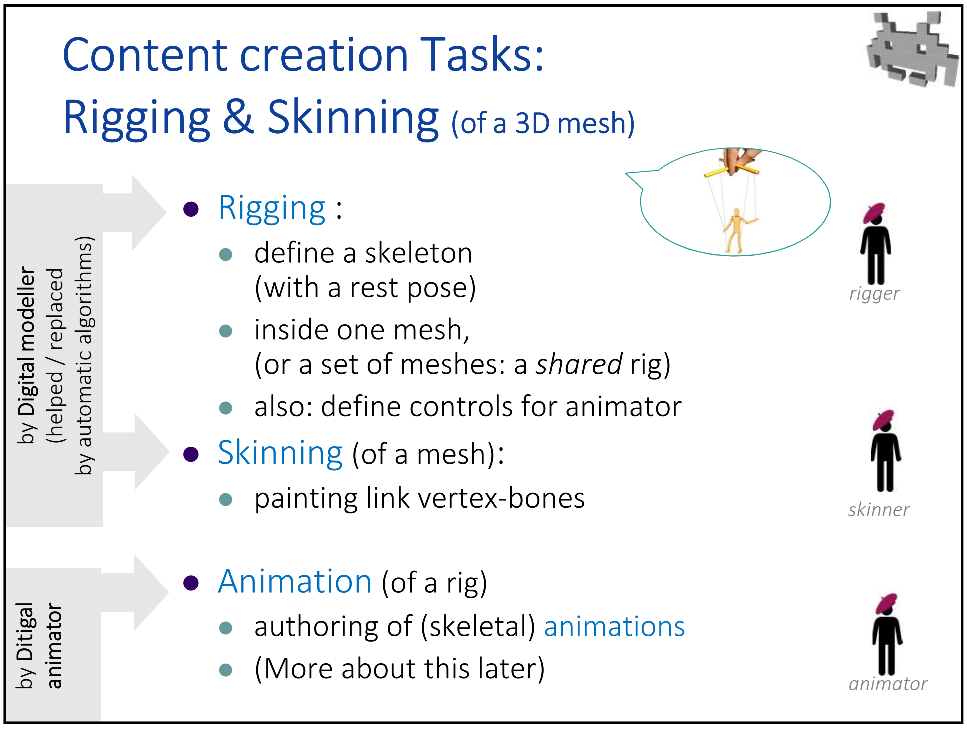


Task in an asset production pipeline: Rigging & Skinning (of a 3D mesh)

Rigging – authoring of a rig
defining the skeleton
(often: also of the controls
to define poses for it)

Skinning – authoring of the skinning
"paint" of weighted links
between vertices and bones

119



Content creation Tasks: Rigging & Skinning (of a 3D mesh)

by Digital modeller
(helped / replaced
by automatic algorithms)

- Rigging :
  - define a skeleton
    (with a rest pose)
  - inside one mesh,
    (or a set of meshes: a *shared* rig)
  - also: define controls for animator
- Skinning (of a mesh):
  - painting link vertex-bones

by Ditigal
animator

- Animation (of a rig)
  - authoring of (skeletal) animations
  - (More about this later)

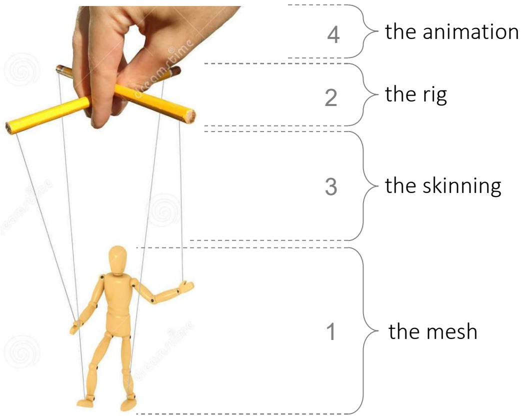
rigger

skinner

animator

120

Asset production



Skeletal animations: authoring / obtaining them

- Manual editing
  - digital animators
  - help from:
    IK (in animation interfaces),
    physical simulations (for “secondary” animations)
- From physics simulation
  - just use the right set of constraints!
    (easy, in Verlet)
  - in preprocessing (bake them) or
    on the fly: “Ragdolling”
- Or…

Skeletal animations: authoring / obtaining them

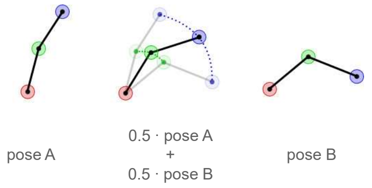
- Motion capture ("mocap")

Motion capture

- Requires heavy setup (maybe not in the future?)
  - Markers / suits
  - Controlled cameras
  - Studio
  - Action must take space in a working space
- Requires skilled actors / performers / athletes
- Can be used to capture
  - single animations (a football stunt, walking)
  - joint performances (cutscenes)
- Requires much postprocessing
  - (cleanup, extraction of keyframes)

Interpolation poses

- any two poses can be interplated!

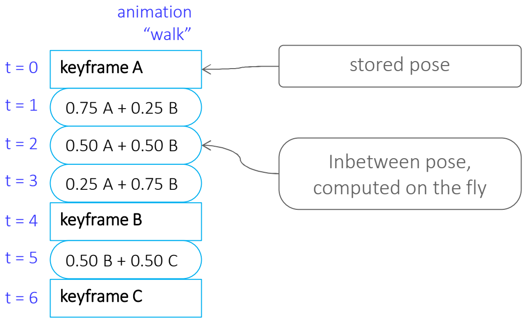


  - just interpolate the per bone *local* transform
  - attention: this requires re-computation of *final* transforms after interpolation

125

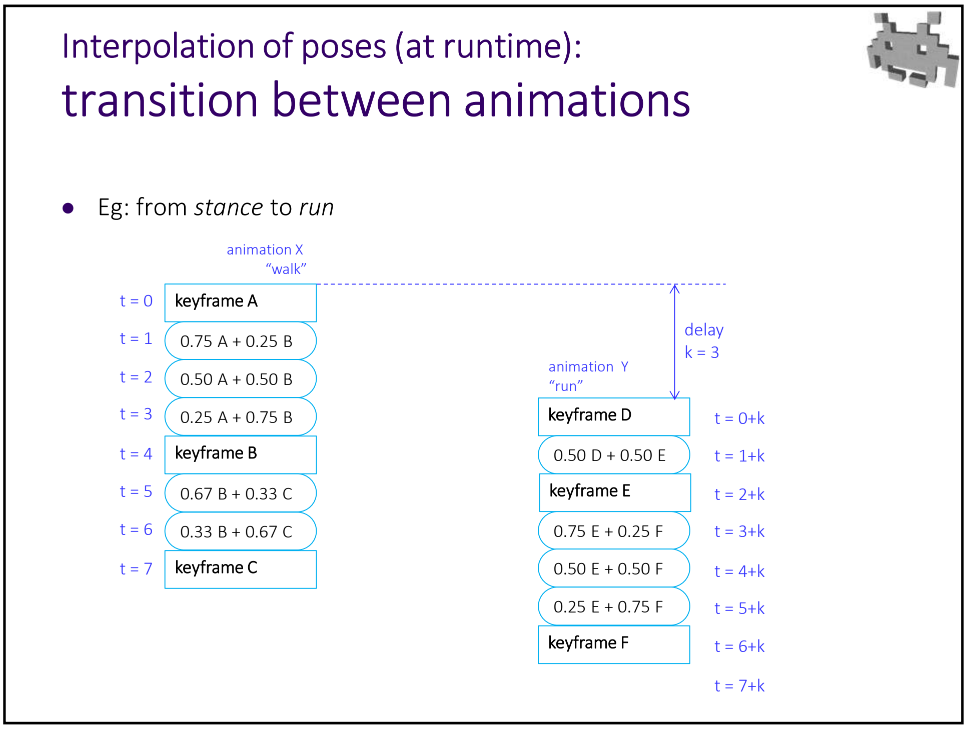
Pose = keyframe

- Compress animations



t = 0	keyframe A
t = 1	0.75 A + 0.25 B
t = 2	0.50 A + 0.50 B
t = 3	0.25 A + 0.75 B
t = 4	keyframe B
t = 5	0.50 B + 0.50 C
t = 6	keyframe C

126



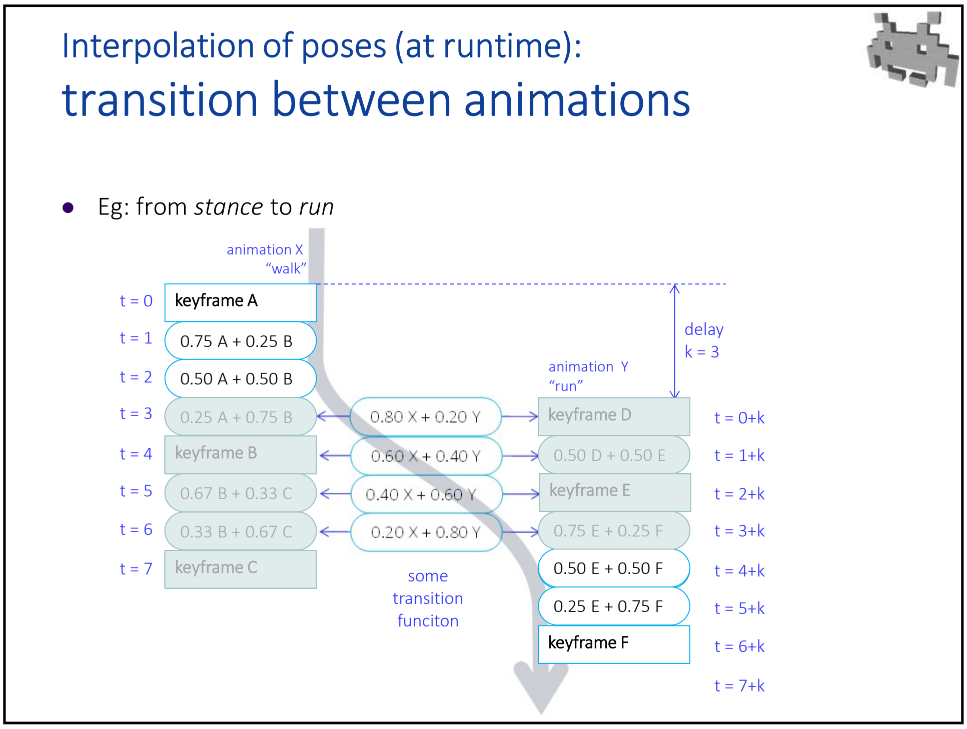
Interpolation of poses (at runtime): transition between animations

- Eg: from *stance* to *run*

delay k = 3

	animation X "walk"	animation Y "run"	
t = 0	keyframe A		
t = 1	0.75 A + 0.25 B		
t = 2	0.50 A + 0.50 B		
t = 3	0.25 A + 0.75 B	keyframe D	t = 0+k
t = 4	keyframe B	0.50 D + 0.50 E	t = 1+k
t = 5	0.67 B + 0.33 C	keyframe E	t = 2+k
t = 6	0.33 B + 0.67 C	0.75 E + 0.25 F	t = 3+k
t = 7	keyframe C	0.50 E + 0.50 F	t = 4+k
		0.25 E + 0.75 F	t = 5+k
		keyframe F	t = 6+k
			t = 7+k

127



Interpolation of poses (at runtime): transition between animations

- Eg: from *stance* to *run*

delay k = 3

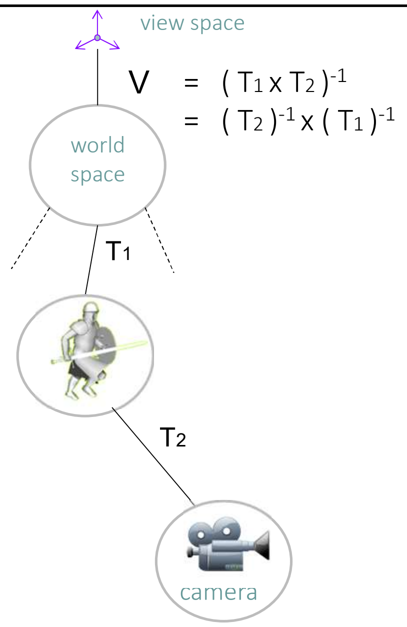
	animation X "walk"	some transition funciton	animation Y "run"	
t = 0	keyframe A			
t = 1	0.75 A + 0.25 B			
t = 2	0.50 A + 0.50 B			
t = 3	0.25 A + 0.75 B	0.80 X + 0.20 Y	keyframe D	t = 0+k
t = 4	keyframe B	0.60 X + 0.40 Y	0.50 D + 0.50 E	t = 1+k
t = 5	0.67 B + 0.33 C	0.40 X + 0.60 Y	keyframe E	t = 2+k
t = 6	0.33 B + 0.67 C	0.20 X + 0.80 Y	0.75 E + 0.25 F	t = 3+k
t = 7	keyframe C		0.50 E + 0.50 F	t = 4+k
			0.25 E + 0.75 F	t = 5+k
			keyframe F	t = 6+k
				t = 7+k

128

Inverse Kinematic (IK): useful in editing, and at run-time

- Two uses:
  - in preprocessing (helping the task of the animator)
  - in real time (done by the game engine)
- Examples of real-time uses:
  - Exact positioning of feet on ground
  - Exact positioning of hand to object to be grabbed
  - Hands need to be joined
    (e.g. 2-handed weapon wielding)
    - (e.g. making the system correct for small changes in bone lengths)
    - (e.g. during interpolated keyframes)
  - etc.

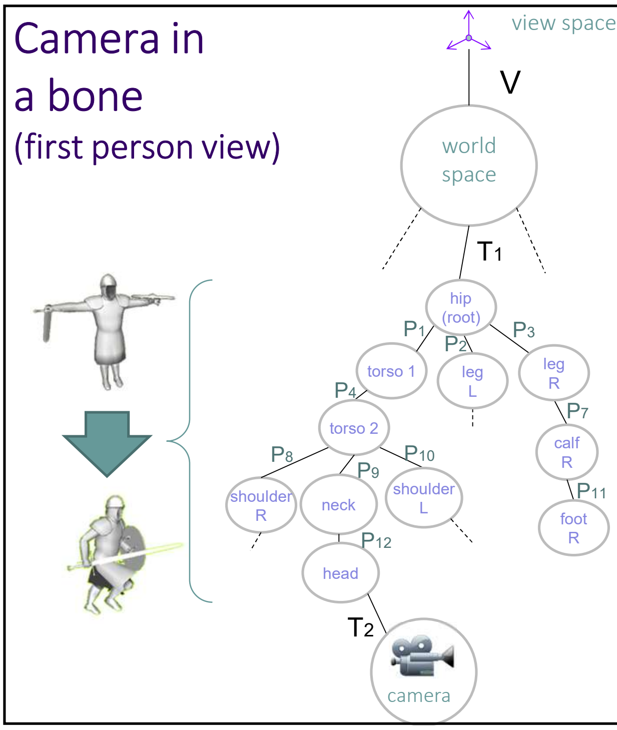
Third person view



$$V = (T_1 \times T_2)^{-1} = (T_2)^{-1} \times (T_1)^{-1}$$

Camera in a bone

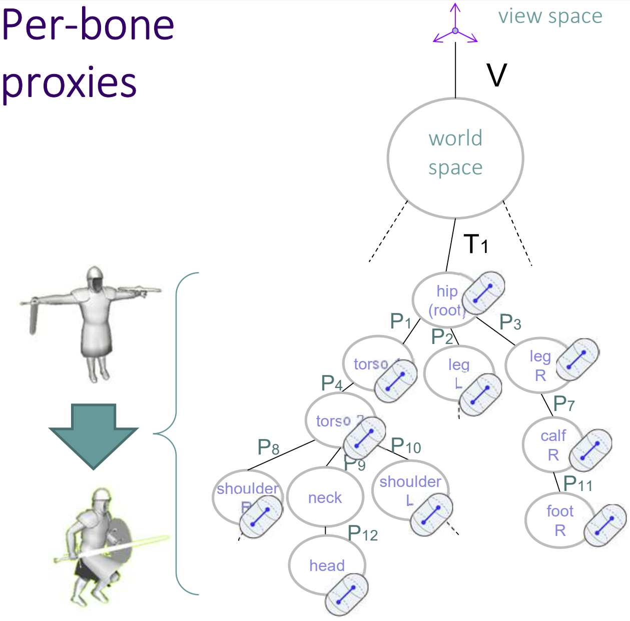
(first person view)

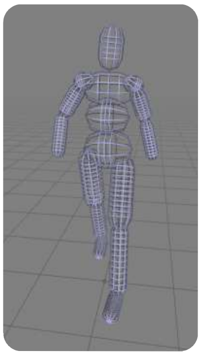


T2 = «mid-eye» positioning
(in «head» space)

133

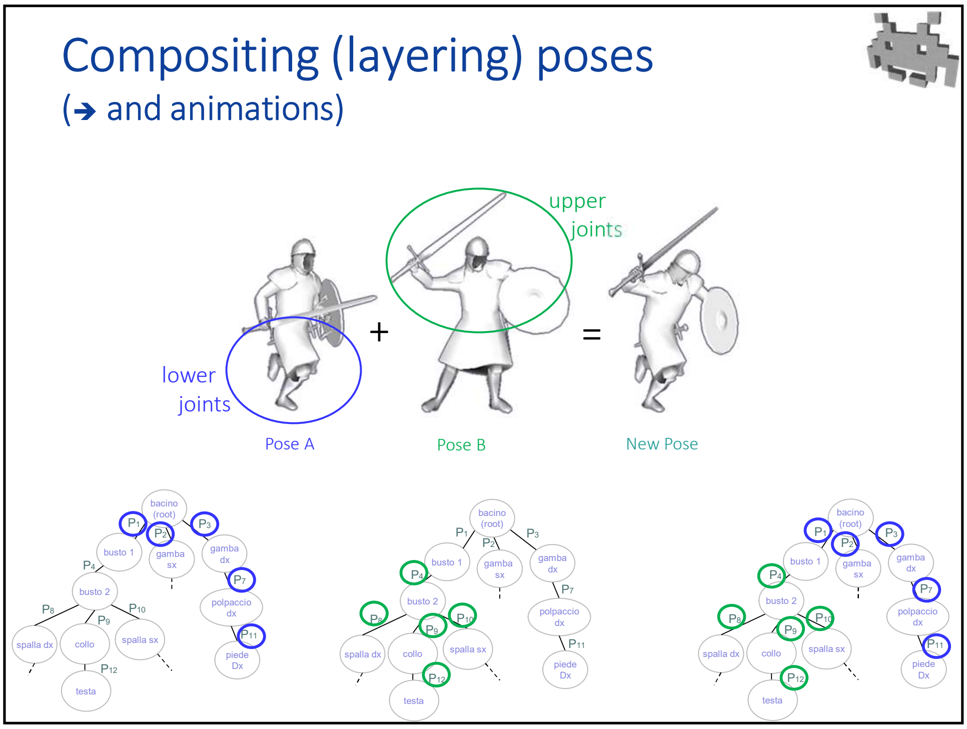
Per-bone proxies





Hit-boxes: e.g. capsules
(not in each bone)

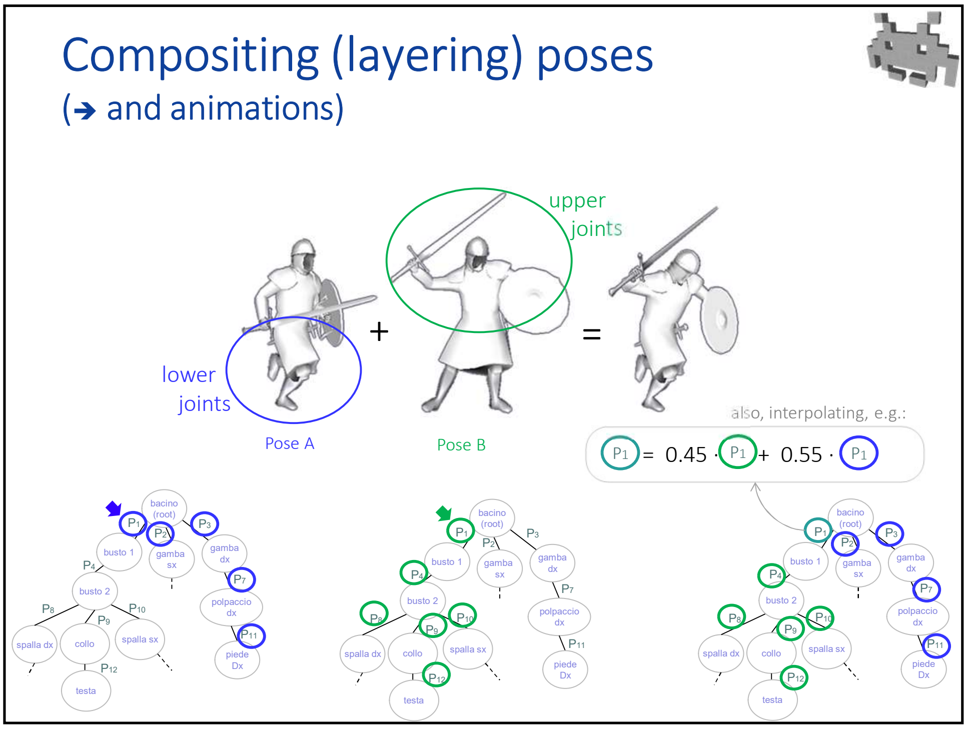
134



Compositing (layering) poses

(➔ and animations)

135



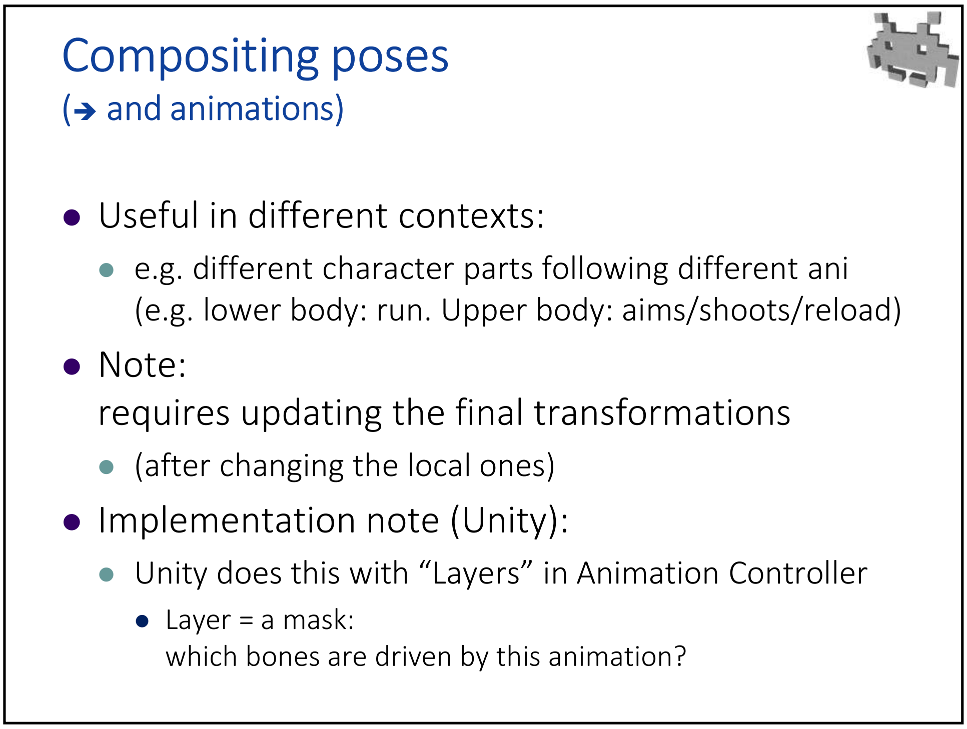
Compositing (layering) poses

(➔ and animations)

also, interpolating, e.g.:

$P_1 = 0.45 \cdot P_1 + 0.55 \cdot P_1$

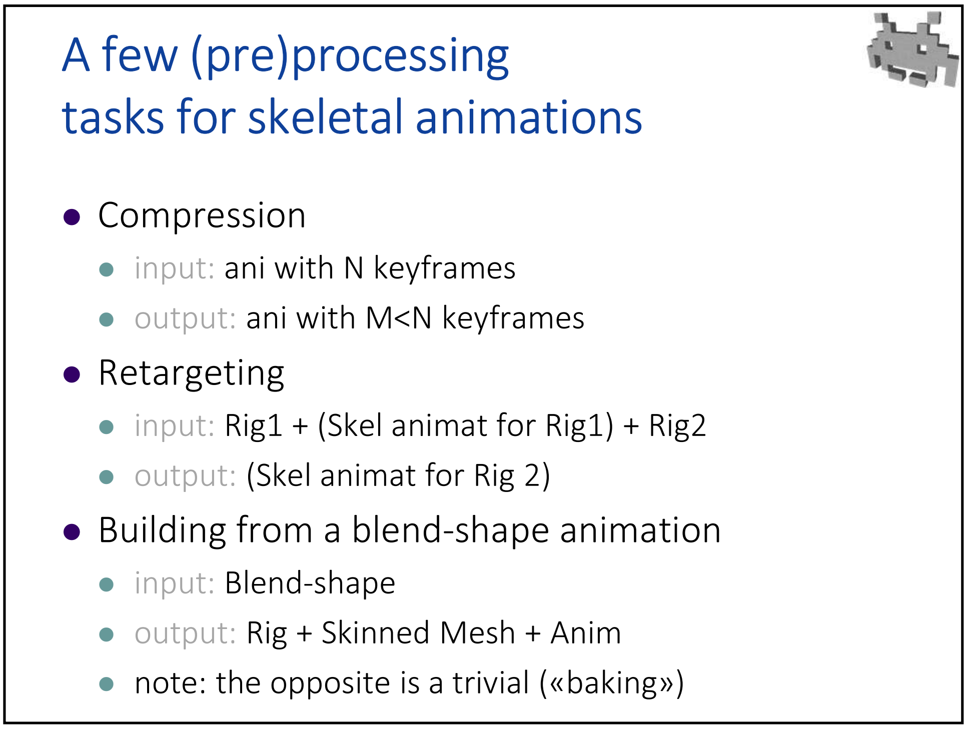
136



Compositing poses
(→ and animations)

- Useful in different contexts:
  - e.g. different character parts following different ani (e.g. lower body: run. Upper body: aims/shoots/reload)
- Note:
  requires updating the final transformations
  - (after changing the local ones)
- Implementation note (Unity):
  - Unity does this with “Layers” in Animation Controller
    - Layer = a mask:
      which bones are driven by this animation?

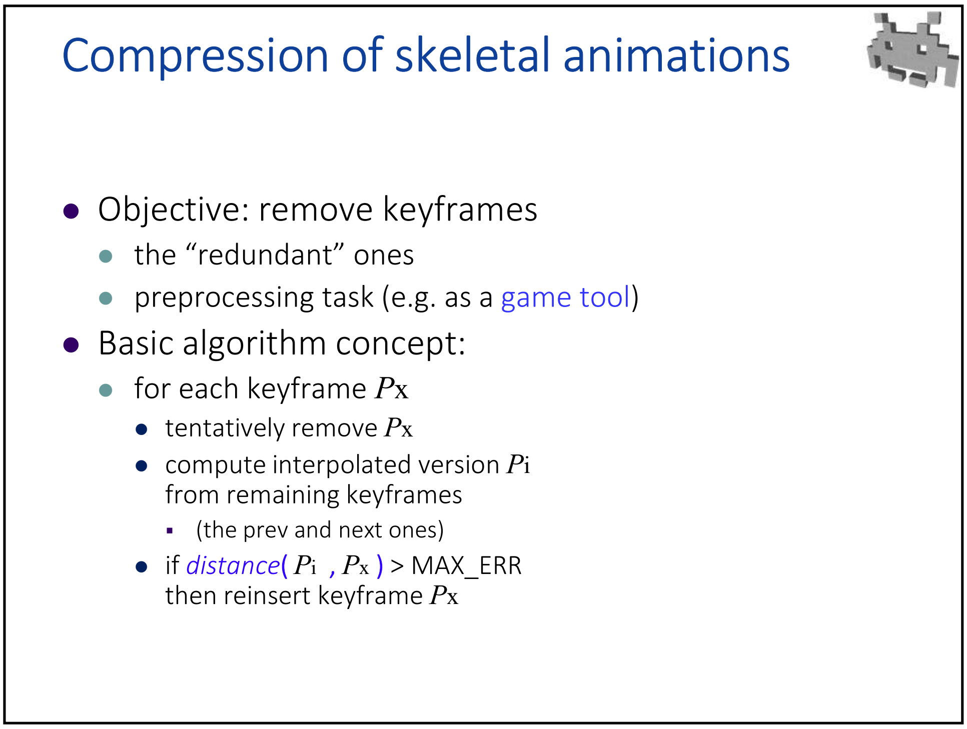
137



A few (pre)processing tasks for skeletal animations

- Compression
  - input: ani with N keyframes
  - output: ani with M<N keyframes
- Retargeting
  - input: Rig1 + (Skel animat for Rig1) + Rig2
  - output: (Skel animat for Rig 2)
- Building from a blend-shape animation
  - input: Blend-shape
  - output: Rig + Skinned Mesh + Anim
  - note: the opposite is a trivial («baking»)

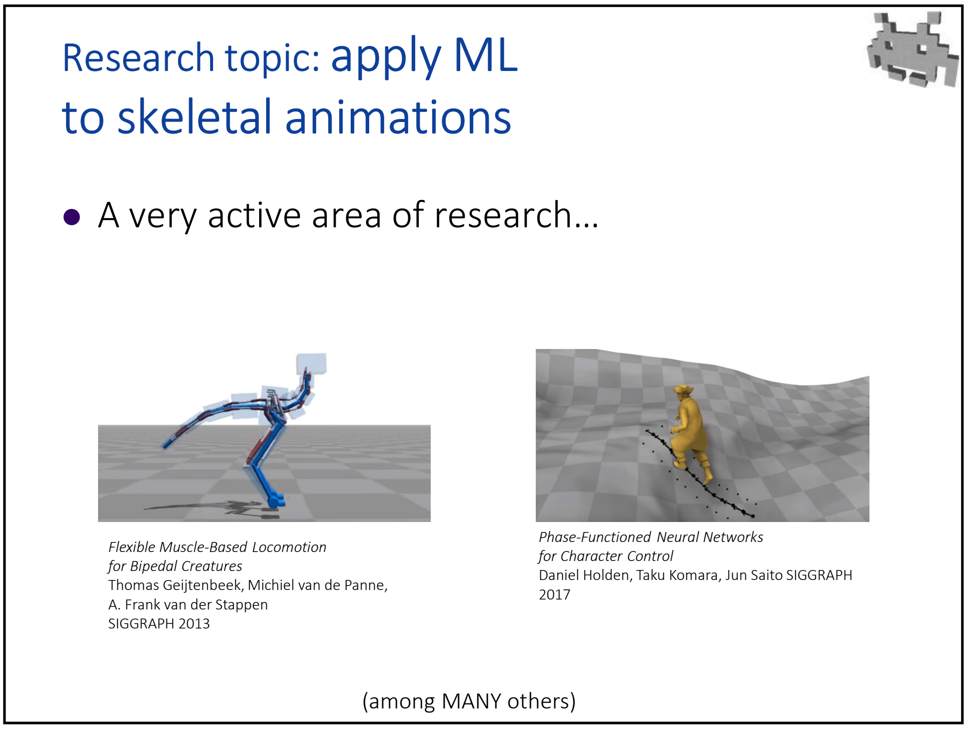
138



Compression of skeletal animations

- Objective: remove keyframes
  - the “redundant” ones
  - preprocessing task (e.g. as a game tool)
- 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

139



Research topic: apply ML to skeletal animations

- A very active area of research…

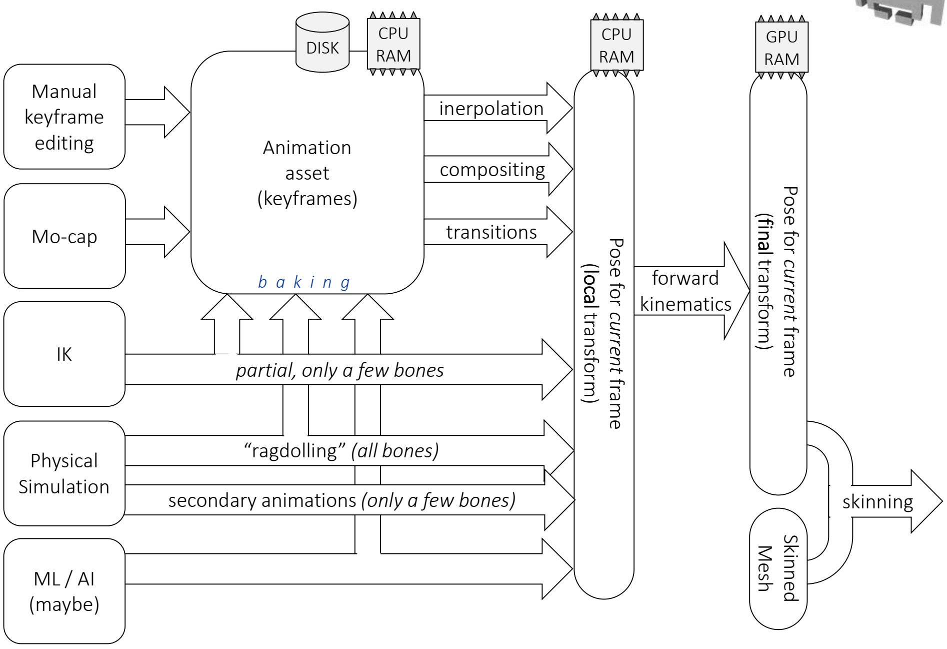
Flexible Muscle-Based Locomotion for Bipedal Creatures
Thomas Geijtenbeek, Michiel van de Panne, A. Frank van der Stappen
SIGGRAPH 2013

Phase-Functioned Neural Networks for Character Control
Daniel Holden, Taku Komara, Jun Saito SIGGRAPH 2017

(among MANY others)

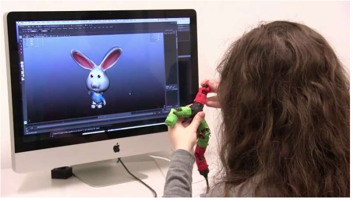
140

Origins of skeletal animation (a summary)



141

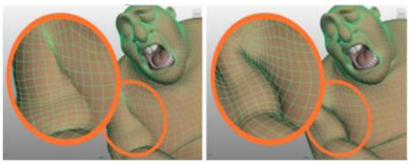
Research topic: better interfaces to author animations



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

142

Research topic: Deformation beyond standard skinning



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

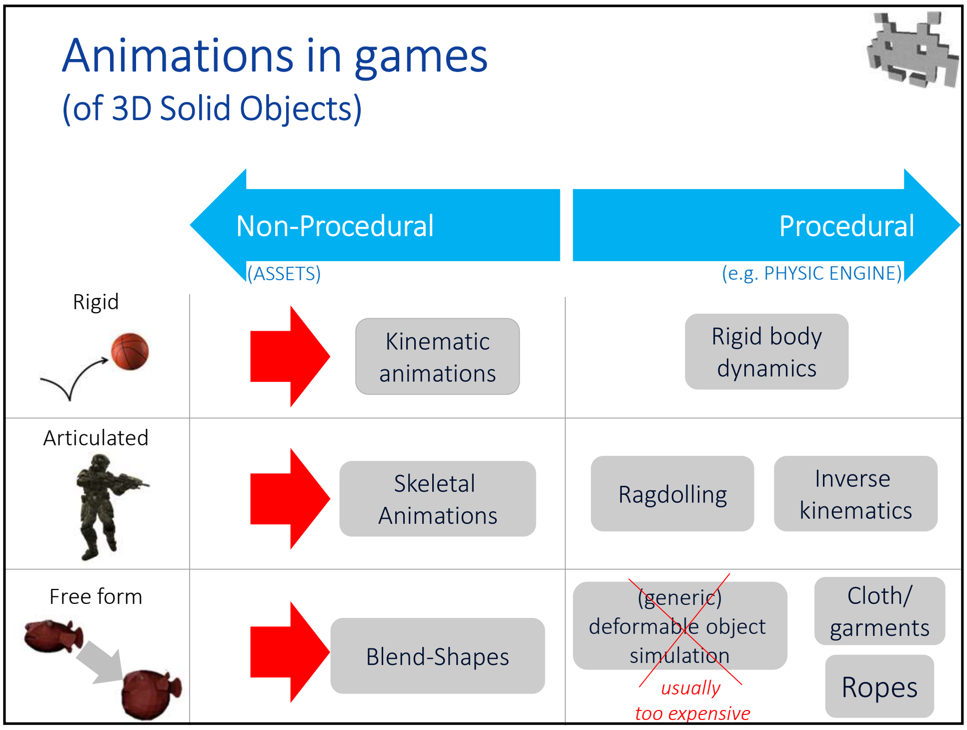
Note: usually way more complex than direct methods (LBS / DQS).
More offline animation oriented than videogames

143

Per-vertex animations VS Skeletal-Animations

- Per Vertex animations
  - can interpolate keyframes (but linear trajectories)
  - heavy in RAM
    - replications of normals / positions
  - light to render / compute
- Skeletal animations
  - can interpolate keyframes *better* (curved trajectories)
  - light in RAM
    - animations / models orthogonality
  - minor overheads
    - transform interpolation (x vert!)
    - updates final transoform before (unless can be baked)

144

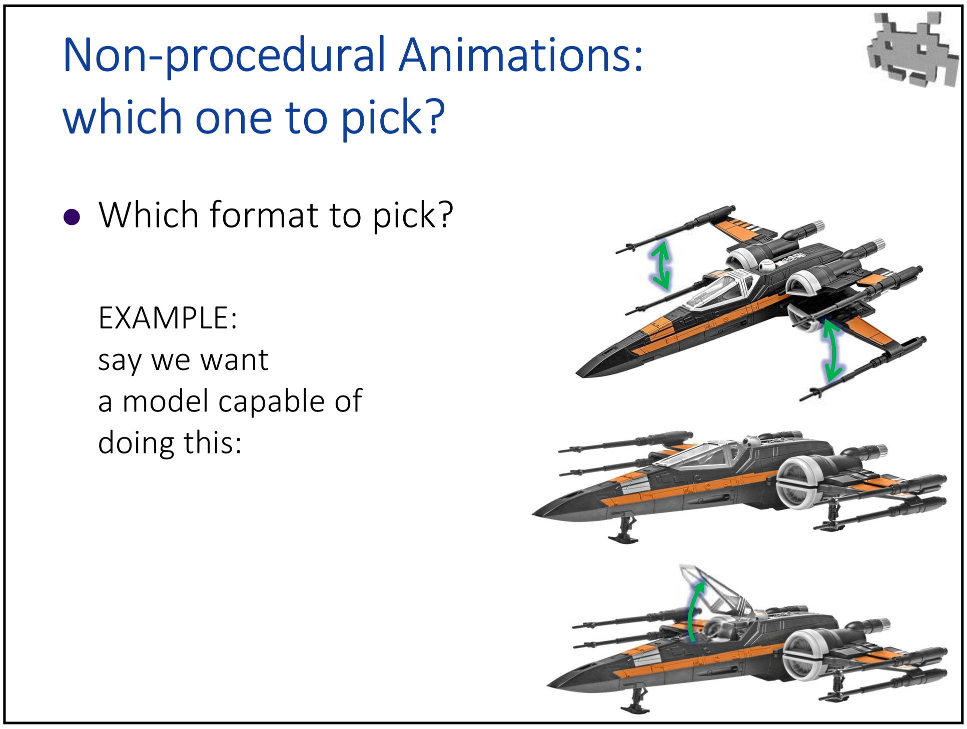


Animations in games

(of 3D Solid Objects)

	Non-Procedural (ASSETS)	Procedural (e.g. PHYSIC ENGINE)
Rigid	Kinematic animations	Rigid body dynamics
Articulated	Skeletal Animations	Ragdolling Inverse kinematics
Free form	Blend-Shapes	~~(generic) deformable object simulation~~ *usually too expensive* Cloth/ garments Ropes

145

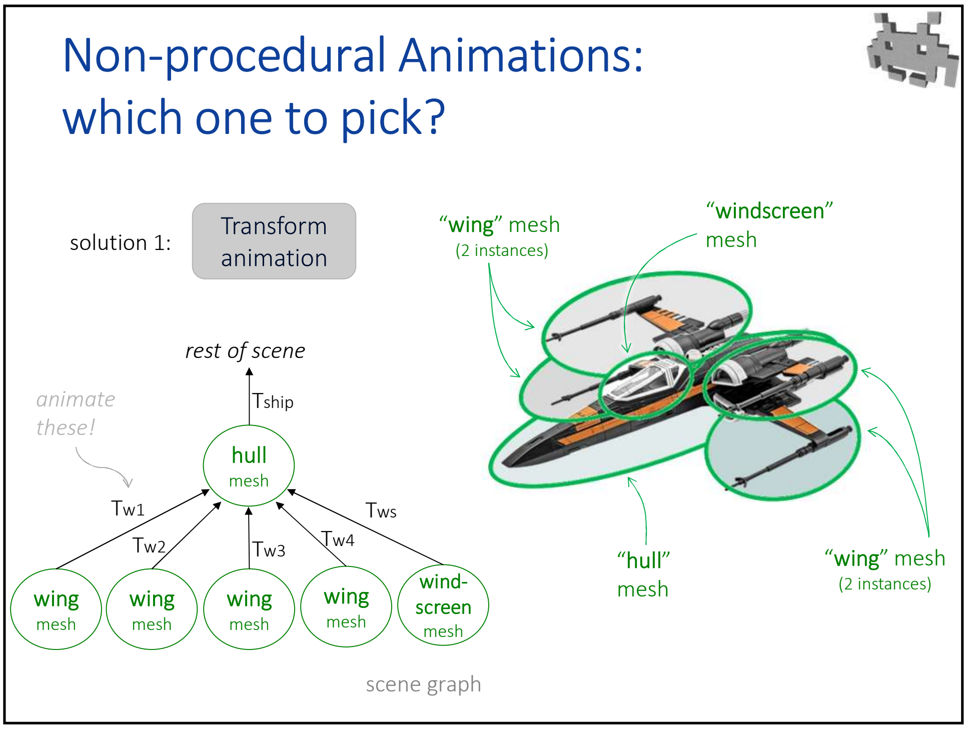


Non-procedural Animations: which one to pick?

- Which format to pick?

EXAMPLE:
say we want
a model capable of
doing this:

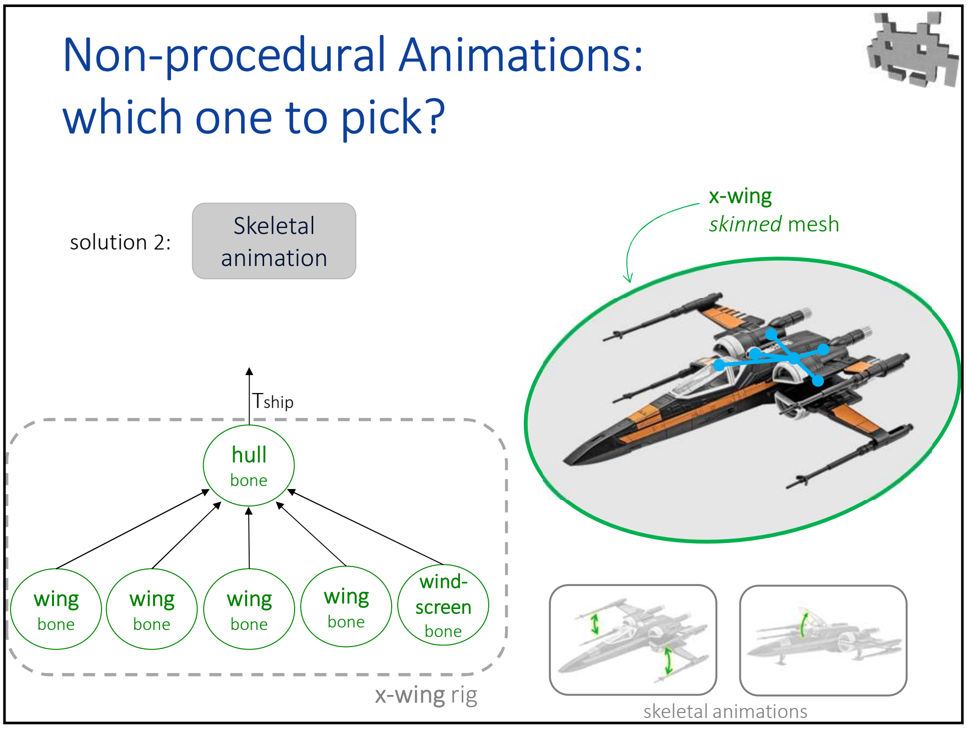
146



Non-procedural Animations: which one to pick?

solution 1: Transform animation

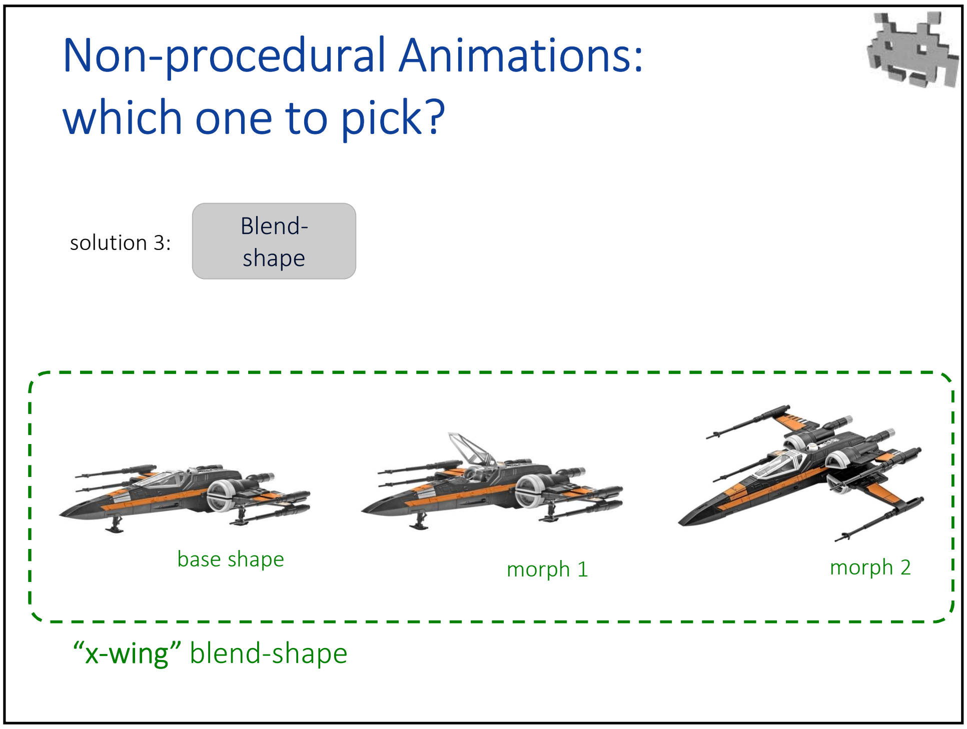
147



Non-procedural Animations: which one to pick?

solution 2: Skeletal animation

148

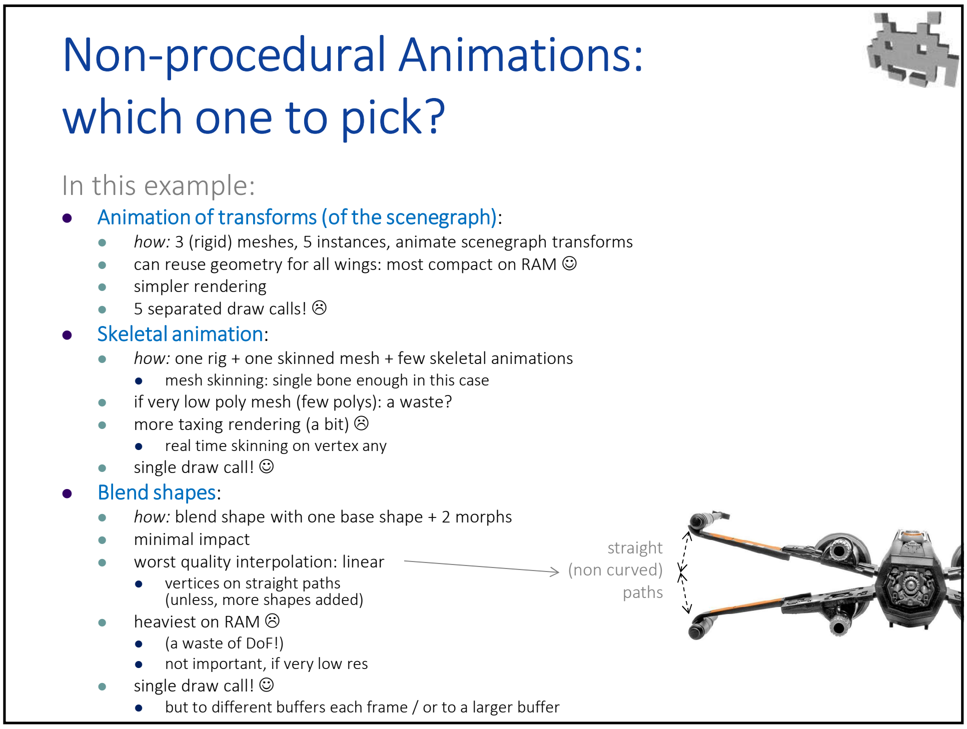


Non-procedural Animations: which one to pick?

solution 3: Blend-shape

"x-wing" blend-shape

149



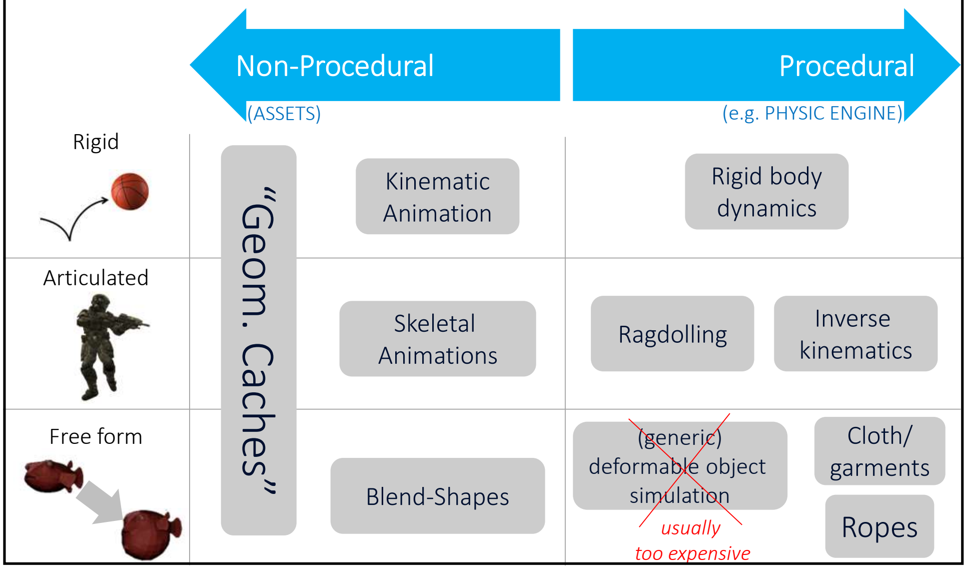
Non-procedural Animations: which one to pick?

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 → straight (non curved) paths
    - vertices on straight paths (unless, more shapes 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

150

Animations in games
(of 3D Solid Objects)



	Non-Procedural (ASSETS)		Procedural (e.g. PHYSIC ENGINE)	
Rigid	"Geom. Caches"	Kinematic Animation	Rigid body dynamics	
Articulated		Skeletal Animations	Ragdolling	Inverse kinematics
Free form		Blend-Shapes	~~(generic) deformable object simulation~~ *usually too expensive*	Cloth/ garments Ropes

151

Geometry Caches
(for lack of a better name)

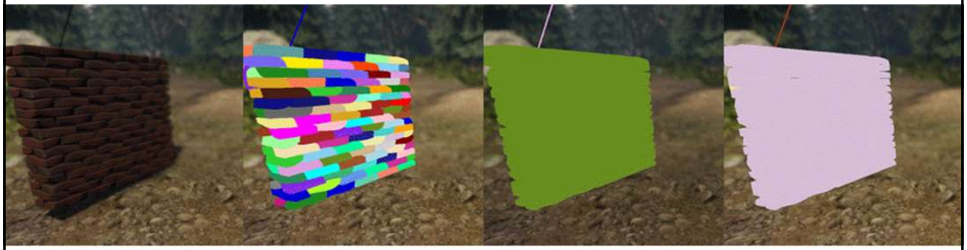
- Baked, optimized animations
  - of a mixture of types e.g.
    - blend shapes
    - kinematic animatios
      - (approximated)
    - skinned animations
      - (typically, no scene graph, just final transf)
  - optimized
    - compressed, streamed…
  - Can be used to bake results of a physical simulation
    - i.e., convert it from physical to kinematic

one used file format: .abc ALEMBIC by SONY PICTURES imageworks

152

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 Tansforms:
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®

153

Animations
in Mecanim (Unity) (notes)

- Assets (models, animation, skeletons) imported as formats:
  - fbx, collada
- Animation compression
  - available during import / builds
  - auto reduction of: num of links per vertex, num of keyframes … :
- «Animator Controller» module → deals with:
  - blending between animations: «transitions»
  - compositing animations: «layers»
    - e.g.: a layer overwrites upper body bones
  - and is nicely WYSIWYG (graph visualization)
- Inverse Kinematic: with scripts (`Avatar.SetIKPoistion`)
- Skeletons:
  - way 1: custom (imported as assets)
  - way 2: built-in standard humanoid skeleton provided
    - (~21 ossa)
    - simplified: rigging (predefined constrains), layers (predef. labelling)

154