


## 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 Materials** ●
- lec. 8: **Game Textures** ●●
- lec. 9: **Game 3D Animations** ●●●
- lec. 10: **3D Audio** for 3D Games ●
- lec. 11: **Networking** for 3D Games ●
- lec. 12: **Interactive Agents** for 3D Games ●
- lec. 13: **Rendering Techniques** for 3D Games ●

lec. 6: Game 3D Models ●

lec. 7: Game Materials ●

lec. 8: Game Textures ●●


lec. 9: Game 3D Animations ●●●





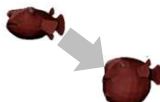

★ ★

computer  
animation

197

## Representations for animations: which type to choose?

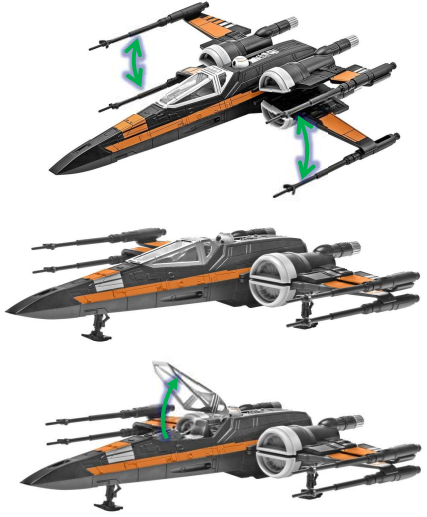


	Non-Procedural <small>(ASSETS)</small>	Procedural <small>(e.g. PHYSIC ENGINE)</small>
<b>Rigid</b> 	 <div style="background-color: #ccc; padding: 5px; border-radius: 5px;">Kinematic animations</div>	<div style="background-color: #ccc; padding: 5px; border-radius: 5px;">Rigid body dynamics</div>
<b>Articulated</b> 	 <div style="background-color: #ccc; padding: 5px; border-radius: 5px;">Skeletal Animations</div>	<div style="background-color: #ccc; padding: 5px; border-radius: 5px; display: inline-block;">Ragdolling</div> <div style="background-color: #ccc; padding: 5px; border-radius: 5px; display: inline-block; margin-left: 20px;">Inverse kinematics</div>
<b>Free form</b> 	 <div style="background-color: #ccc; padding: 5px; border-radius: 5px;">Blend-Shapes</div>	<div style="background-color: #ccc; padding: 5px; border-radius: 5px; display: inline-block; text-align: left;"> <del>(general) soft body simulation</del>  <i>usually too expensive</i> </div> <div style="background-color: #ccc; padding: 5px; border-radius: 5px; display: inline-block; margin-left: 20px;">Cloth/garments</div> <div style="background-color: #ccc; padding: 5px; border-radius: 5px; display: inline-block; margin-left: 20px;">Ropes</div>

198

### Representations for animations: which type to choose?

EXERCISE:  
say we want  
a model capable of  
doing this:  
  
(& combinations!)



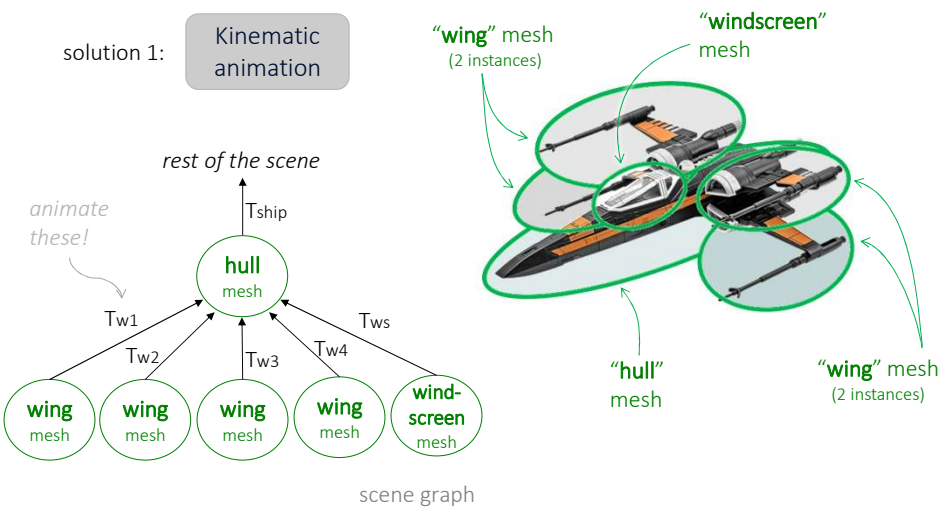
199

### Representations for animations: which type to choose?

solution 1: Kinematic animation

rest of the scene

animate these!



scene graph

200

### Representations for animations: which type to choose?

solution 2: Skeletal animation

The diagram shows a skeletal hierarchy for an X-wing. At the top is a 'Tship' transform. Below it is the 'hull bone'. Five 'wing bone' nodes and one 'wind-screen bone' node are connected to the hull bone. This entire structure is labeled 'x-wing skeleton'. To the right, a 3D model of the X-wing is shown with a blue skeletal structure overlaid, labeled 'x-wing skinned mesh'. Below this are two smaller images labeled 'skeletal animations' showing the X-wing in different poses with green arrows indicating movement.

201

### Representations for animations: which type to choose?


solution 3: Blend-shape

The diagram shows three X-wing models in a row, enclosed in a dashed green box. From left to right, they are labeled 'base shape', 'morph 1', and 'morph 2'. Below the box is the label 'x-wing blend-shape'.

202

## Representations for animations: which type to choose? In this example...



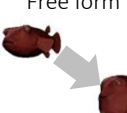
- **Kinematic animation:**
  - *how:* 3 (rigid) meshes, 6 instances, animate 5 scene-graph local transforms (e.g. via scripts)
  - can reuse meshes (geometry+attr+connectivity) for all wings: most compact on V-RAM ☺
  - simpler rendering (just rigid meshes) ☺
  - 6 individual draw calls! ☹
  - limitation: no deformable bits (e.g., no leather pieces connecting the windscreen)
- **Skeletal animation:**
  - *how:* one skeleton (6 bones) + one skinned mesh + 2 skeletal animations (1 pose each, maybe)
  - (in this case, single bone per-vertex is enough for the skinning)
  - open-windscreen with both open OR closed wings? Still possible! (layering of animations)
  - rendering: small increase of per-vertex computation burden: LBS or DQS ☹
  - single draw-call! ☺
  - deformable bits: allowed ☺ (triangles connecting vertices linked to ≠ bones, or blend skinning)
- **Blend-shape:**
  - *how:* one base-shape + 2 morphs
  - bad quality of interpolation: linear
    - vertices follow straight paths
  - heaviest on V-RAM ☹
    - not a bit problem, if low-res
  - a single draw call! ☺
    - but to different geometry buffers at each frame



straight (non curved) paths

203

## Animations in games (of 3D Solid Objects)

	Non-Procedural <small>(ASSETS)</small>	Procedural <small>(e.g. PHYSIC ENGINE)</small>
Rigid 	Kinematic Animation	Rigid body dynamics
Articulated 	Skeletal Animations	Ragdolling    Inverse kinematics
Free form 	Blend-Shapes	<del>(general) Soft body simulation</del> <i>usually too expensive</i>


"Geom. Caches"

205


## 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:




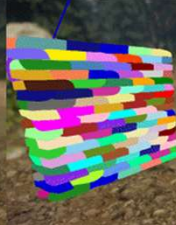

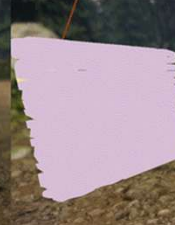
.abc



206

## Geometry Caches (for lack of a better name)


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

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


**Geometry Caches**  
(a subset of Alembic)

by



208


## Animations in games: authored, procedural... or a mix? [summary]



- A few examples of current commonly used mixes:
  - 1: “*primary*” animations: authored  
“*secondary*” animations: physically generated
  - 2: *alive* characters: authored  
*dead* characters: physically generated (“ragdolls”)
  - 3: walk cycle: authored (skeletal animation)  
exact *feet placement*: procedural (inverse kinematic)
  - 4: normal “behavior”, such as sparring: authored  
*gaze control* during sparring: procedural
  - 5: normal “behaviors” such as jumping, running: authored  
modifications / transitions: AI generated  
and more!
- mixing AI-generated with authored animations is a frontier in the field of Computer Animation

209

## Mecanim Animations System in Unity [practical notes]



- Assets (models, animation, skeletons) imported as formats:
  - fbx, collada
- Keyframe sparsification, or reduction of num of links per vertex
  - available as operation at asset import time
- «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)

210