3D model animation and CGI animation techniques

Hei Chun Shum
From Animation to Computer Animation

- ON TWOS - 12FPS (traditional hand-drawn animation)
- a digital successor to stop motion techniques (1849)
- Pre-rendered or real-time
- 3D models, Animation, Rendering
- The differences in appearance between key frames are automatically calculated by the computer in a process known as tweening (inbetweening) or morphing.

a stop motion/pixilation scene from Hôtel électriqu e (1908)
Illusion of motion

- To trick the eye and the brain into thinking they are seeing a smoothly moving object -> picture drawn at 12 fps or faster (Masson 1999)
- 60 fps -> the sweet spot (perfectly smooth)
- Lag, frame loss, tearing
- Vsync, gsync, freesync
- 60hz vs 144hz vs 240hz

EVOLUTIONS!

https://paulbakaus.com/tutorials/performance/the-illusion-of-motion/
Realism

- greatest challenge - creating photorealistic human characters
- Non-realistic characters like legendary creatures and cartoon characters, superheroes
- twelve basic principles of animation
  (The Illusion of Life: Disney Animation, 1981)
Twelve basic principles of animation

- "Bible of animation"
- The Illusion of Life: Disney Animation
  (Ollie Johnston and Frank Thomas 1981)
- Illusion that stick to the basic laws of physics
- Emotional timing, character appeal
- intended to apply to traditional, hand-drawn animation
- still have great relevance for today’s computer animation
Squash and stretch

- "by far the most important" of the 12 basic principles of animation

- only stiff objects remain inert during motion

- Non-stiff object - although retaining overall volume, tend to change shape in an extent that depends on inertia and elasticity of the different parts of the moving object

- Basic training of Disney’s animators
Short timeline of 3D animations in films
1977

- Star Wars
- Used animated 3D wire-frame for the trench run on Yavin 4
1991

- Terminator 2: Judgment Day
- First realistic human movements on a CGI character
- First partially computer-generated main character
- First blockbuster movie to feature multiple morphing effects
- First use of a **personal computer** to create major movie 3D effects
1993

- Jurassic Park
- First photorealistic CG creatures
1997

- Titanic

- First wide-release feature film with major elements rendered under Linux (Red Hat 4.1)

- rendering of flowing water

- Linux Helps Bring Titanic to Life
Linux Helps Bring Titanic to Life

“Since building a full-scale model of the Titanic would have been prohibitively expensive, only a portion of the ship was built full size (by the production staff), and miniatures were used for the rest of the scenes. To this model we added other elements of the scene such as the ocean, people, birds, smoke and other details that make the model appear to be docked, sailing or sunk in the ocean. To this end, we built a 3D model and photographed 2D elements to simulate underwater, airborne and land-based photography.”

https://www.linuxjournal.com/article/2494
1997

- The Matrix
- First use of CG interpolation with **bullet time effects**
- Star Wars: Episode I – The Phantom Menace
- First film to use CG extensively for thousands of shots, including backgrounds, environmental effects, vehicles, and crowds
2004

- The Polar Express
- First computer-animated 3D film created with **motion capture**
2009

- Avatar
- First full-length movie made using motion capture
- a fully CG 3D photorealistic world
Better CGs
Better animations
Better industry tools
Fancy right?
but how?
3D Animation Techniques
Skeleton animation and Morph Target Animation

In almost all 3D animation software...
Skeletal animation
(rigging)
Rigging is making our characters able to move. The process of rigging is we take that digital sculpture, and we start building the skeleton, the muscles, and we attach the skin to the character, and we also create a set of animation controls, which our animators use to push and pull the body around.

"What is 3D Rigging for Animation & Character Design?"
Petty, Josh.
- introduced in 1988

- allows animators to control often complex algorithms and a huge amount of geometry

- Intention: only to control the deformation of the mesh data
Animated 3D characters. No 3D knowledge required.

Rapidly create, rig and animate unique characters for design projects.

SIGN UP FOR FREE  LOG IN

IMAGINE
CREATE
RIG
ANIMATE
DESIGN
Rigging workflow

- Construct a series of bones
- Set up hierarchy, linkages to the mesh (bones to vertices groups)
- Set-up weightings, constraints
- Rig and design
Principle behind - Kinematics
6DoF

Translation and Rotation (geometry)

Moving up and down (elevating/heaving)

Moving left and right (strafing/swaying)

Moving forward and backward (walking/surging)

Swivels left and right (yawing)

Tilts forward and backward (pitching)

Pivots side to side (rolling)

$$R = R_z(\alpha) R_y(\beta) R_x(\gamma) = \begin{bmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \gamma & -\sin \gamma \\ 0 & \sin \gamma & \cos \gamma \end{bmatrix}$$

Single rigid mesh
Kinematic chain

\[ M = 6n - \sum_{i=1}^{j} (6 - f_i) = 6(N - 1 - j) + \sum_{i=1}^{j} f_i \]

\( n \) : moving links

\( j \) : joints each with freedom \( f_i, i = 1 \)

* \( N \) includes the fixed link
Forward kinematics

\[
[T] = [Z_1][X_1][Z_2][X_2] \ldots [X_{n-1}][Z_n],
\]

- forward kinematics equations of the serial chain

- \([Z]\) : rigid transformation to characterize the relative movement allowed at each joint (affine transformation)

- \([X]\) separate rigid transformation to define the dimensions of each link (affine transformation)

- \([T]\) : the transformation locating the end-link
Geometrical Interpretation

\[ R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \]
The forward kinematic animation problem uses the kinematics equations to **determine** the pose given the joint angles.

The inverse kinematics problem **computes** the joint angles for a desired pose.
Procedural animation

- Rigid body dynamics (We are here)
- Particle systems
- Fluid dynamics
- Fur & hair dynamics
Inverse kinematics

- Often use for arms and legs for human / animal mesh or dragon tail

- IK simplifies the animation process, and makes it possible to make more advanced animations with lesser effort. (blender manual)

- Blender Pose Mode -> Automatic IK (Bone Constraints)
Gradient Descent

Optimization algorithm

Derivative

\[ f'(p) = \lim_{{\Delta x \to 0}} \frac{f(p + \Delta x) - f(p)}{\Delta x} \]

Estimated gradient

\[ \nabla f(p) = \frac{f(p + \Delta x) - f(p)}{\Delta x} \]

Update parameter

\[ p_{i+1} = p_i - L \nabla f(p_i) \]
Gradient Descent

- Multiple variables
- sufficiently small sampling distances $\Delta x$, $\Delta y$ and $\Delta z$

\[
\nabla f_{\alpha_0} (\alpha_0, \alpha_1, \alpha_2) = \frac{f(\alpha_0 + \Delta x, \alpha_1, \alpha_2) - f(\alpha_0, \alpha_1, \alpha_2)}{\Delta x}
\]

\[
\nabla f_{\alpha_1} (\alpha_0, \alpha_1, \alpha_2) = \frac{f(\alpha_0, \alpha_1 + \Delta y, \alpha_2) - f(\alpha_0, \alpha_1, \alpha_2)}{\Delta y}
\]

\[
\nabla f_{\alpha_2} (\alpha_0, \alpha_1, \alpha_2) = \frac{f(\alpha_0, \alpha_1, \alpha_2 + \Delta z) - f(\alpha_0, \alpha_1, \alpha_2)}{\Delta z}
\]
Inverse kinematics

function(f) which takes a parameter $\alpha_i$ for each joint of our robotic arm

$\alpha$ - current angle of the joint

function $f(\alpha)$ return a single value indicates how far the effector of the robotic arm is from the target point $T$

Objective - find the values for $\Delta$ that minimise $f$

Apply gradient descent
- Sampling distance
- Learning rate
- threshold for early termination maybe needed
- Constraints like minimum / maximum angle
Morph Target Animation
Morphing Before 1990s

- Traditional technique: Dissolving
- gradual transition from one image to another
- double exposure from frame to frame
- Fade-in, fade-out
- Replaced by digital morphing for more realistic transformations
Image Morphing

A combination of generalized image warping with a cross-dissolve between pixels

Morphing involves two steps:
- Pre-warp the two images
- Cross-dissolve their colors
Image Pre-Warping

Why warp first?
In order to align features that appear in both images (e.g., eyes, mouth, hair, etc). Without such an alignment, we would get a “double-image” effect!!

Image pre-warping
Re-position all pixels in the source images to avoid the “double-image” effect as much as possible

Pre-warping implemented using the Field Morphing Algorithm
Image Morphing

Both morphing steps specified by same parameter $t$

- Warp the two images according to $t$
- Cross-dissolve their colors according to $t$

Morphing videos generated by creating a sequence of images, defined by a sequence of $t$-values (e.g., 0, 0.1, 0.2, ..., 0.9, 1)

(slides adapted from CSC320: Introduction to Visual Computing, University of Toronto)
5:50 - same base mesh, same amount of polygons
Morph animation comparing with skeleton animation

- more control (define the individual positions of the vertices within a keyframe)
- animating cloth, skin, and facial expressions (difficult to be controlled with bones)
- more labour-intensive? (every vertex position must be manually manipulated)
- Require more memory and storage for large models (storing position of each modified vertex of each frames)
Side Topic - Computer Facial animation

2 main areas
- techniques to generate animation data
  (eg. motion capture and keyframing)

- methods to apply such data to a character
  (eg. morph targets animation, skeleton animation)
Motion capture

- save time cost, especially for complex movement
- increase level of realism
- follow the laws of physics
- Marker-base, markerless
Audio-driven techniques

- speech animation
- keyframes with observed speeches
- typically use hidden Markov models or neural nets to transform audio parameters into a stream of control parameters for a facial model
- generate speech animations from audio
- Require a large dataset for training to produce natural result
Side Topic - Motion Matching

- **Motion Graphs** from automated graph based motion synthesis
  2002 University of Wisconsin

- **Motion fields** for Character Animation
  2010 University of Washington

- **Motion Matching**
  GDC2016 Ubisoft Toronto (https://www.youtube.com/watch?v=KSTn3ePDt50)

- **Learned motion matching**
  2020 Ubisoft La Forge
Motion Matching (current)

- Prepare motion capture dataset, capture as many actions as possible carefully
- Goal: animate a character to make it follow a new path
- Motion matching: repeatedly searching the dataset for a clip that, if played from the current location, would do a better job of keeping the character on the path than the current clip
- Hand pick features from clips -> matching feature database
- Scalability issue - how much data may be required for a AAA production
Learned Motion Matching

“Our goal in this research is simple: to replace the Motion Matching machinery with something that produces exactly the same result, but which does not require keeping as much data in memory. The work flow for animators should remain exactly the same: let them craft whatever system they like using Motion Matching and as much data as they want. Then, once they are finished, plug-in an alternative system which produces the same result but which has both lower memory usage and constant CPU cost. This is where Learned Motion Matching comes in.”

Ubisoft La Forge

Learned Motion Matching

Goal: remove the animation dataset from the diagram

- Train multiple neural network models to replace the dataset lookup
Side Topic - Ragdoll physics

- characters in video game die in one of two ways: ragdoll or animation

- An area of procedure animation processing and physic processing

- often used as a replacement for traditional static death animations in video games and animated films

- computers increased in power -> do limited real-time physical simulations

- Common approaches like Inverse kinematics (Halo), blended ragdoll (left 4 dead)

- Ragdoll Wizard (Unity)
Beyond ragdoll and keyframe animation
Euphoria

- By NaturalMotion
- Full simulation of the 3D character, including body, muscles and motor nervous system
- No predefined animations
- Actions and reactions are synthesized in real-time
- Different every time, even when replaying the same scene
- Ended commercial licensing
RAGE (Rockstar Advanced Game Engine) integrates the third-party middleware components Euphoria.
Future of 3D animation

- More platforms, lucrative entertainment industry
- will probably become even more mainstream
- more accessible (simplier production)
- High-end quality work still need specialists eg. 3D artists
- VR, AR, 3D scanning - mixed reality capture
- Computer vision evolution
Reference

https://www.utdallas.edu/atec/midori/Handouts/procedural_animation.htm

https://www.alanzucconi.com/2017/04/17/procedural-animations/


https://pointintimestudios.com/the-future-of-3d-animation/


https://people.eecs.berkeley.edu/~jrs/meshpapers/LeeDobkinSweldensSchroder.pdf

Thank you for listening