

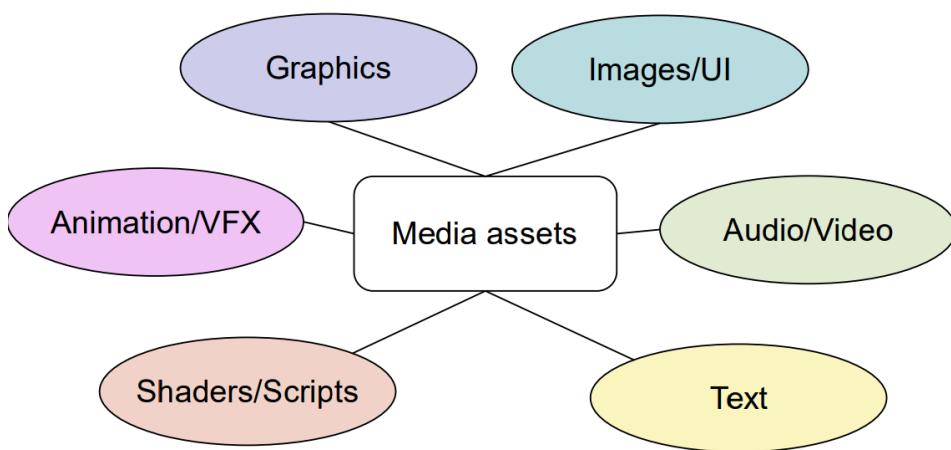
Lezione 9 09/04/2025

Asset, animations and collisions

Con **asset** vogliamo indicare tutti i dati che vengono coinvolti per realizzare un videogame. Sono tutte quelle componenti, divise in visual/audio/textual (ma anche altre), che costituiscono i componenti principali per la costruzione di un videogame.

Gli asset non sono solo modelli e texture, ma anche per esempio il suono delle onde associate ad un delfino, oppure del testo che può comparire quando il delfino nuota nell'acqua.

Cos'è un asset e cosa non lo è spesso dipende dal game engine, non è definito in modo preciso.

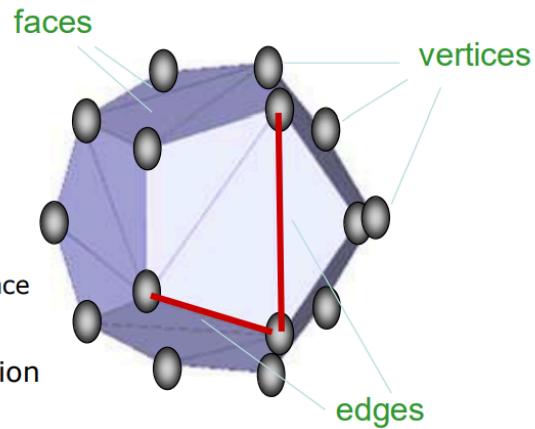


Asset grafici

La maggior parte degli **asset grafici** sono "low poly" ovvero hanno pochi poligoni, la geometria non ha troppa informazione. Stiamo restringendo il numero di facce, per efficienza, altrimenti avrei un costo altissimo sia a livello di memoria che a livello di computazione.

Questo è un riassunto di quanto visto nella lezione precedente:

- ◆ Data structure for modelling 3D objects
 - GPU friendly
 - Resolution = number of faces
 - Resolution is (potentially) Adaptive (that is, more faces where needed)
- ◆ Used to model the **visual appearance** of 3D physical objects in the game
 - the ones which can be represented by their surface
 - most solid objects (rigid or not)
- ◆ Mathematically: a piecewise linear approximation of the surface
 - a set of 3D samples, "vertices" connected by a set of triangular "faces" connected side to side by "edges"



Images/UI

- ◆ Textures
- ◆ Sprites
- ◆ UI Graphics
- ◆ Background
- ◆ Illustrations
- ◆ Normal/Bump maps
- ◆ Skyboxes

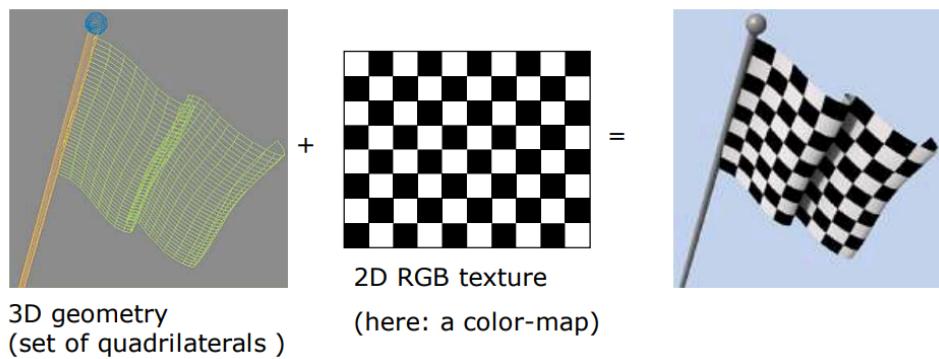


Abbiamo sempre un compromesso tra qualità e computazione/memoria.

- **Resolution:** Must be optimized for performance: too large = memory issues.
- **Compression:** Use formats like PNG (lossless) or JPEG (smaller file sizes).
- **Transparency:** Use alpha channels (e.g., PNGs) for cutouts or semi-transparent elements.
- **Atlas Packing:** Combine many small images into one (texture atlas) to reduce draw calls.

è tipico avere fogli di asset che contengono tante texture per esempio insieme, di modo da caricarle insieme in memoria.

Tramite una serie di operazioni siamo in grado di mappare la texture sulla geometria.



La texture non è l'immagine fatta da pixel, ma da **Texel (texture elements)**. Questo è l'elemento fondamentale di una texture map. Le texture sono rappresentate da array di texel.

Possiamo avere diverse texture per una stessa mesh, per esempio per cambiare la maglia di un giocatore di calcio.

I texel permettono di andare a definire informazioni non per elemento della mesh, non per ogni faccia della mesh, ma per l'intera mesh, potendo quindi

rappresentare di più, cose più complesse. Si affidano i dettagli alle texture e meno alle geometrie.

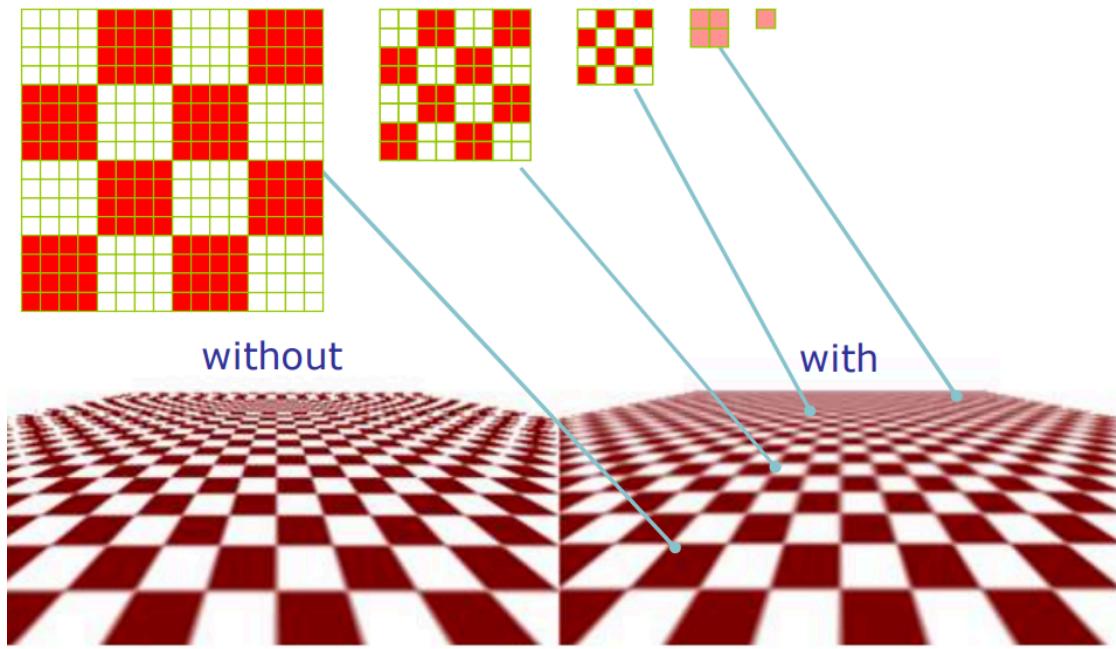
- ◆ Each texel = a base-color (components: r,g,b)
 - The texture sheet is a “diffuse-map” / “color-map” / “RGB-map”
- ◆ Each texel = a transparency factor (components: α)
 - The texture sheet is a “alpha-map” or “cutout-texture” (exp. if 1bit)
- ◆ Each texel = a normal (versor, with components: x,y,z)
 - The texture sheet is a “normal-map” or “bump-map”
- ◆ Each texel = a specular coefficient value
 - The texture sheet is a “specular-map”
- ◆ Each texel = a glossiness value
 - The texture sheet is a “glossiness-map”
- ◆ Each texel = a *baked* lighting value...
 - The texture sheet is a (baked) “light-map”
- ◆ Each texel = a distance from a surface value
 - The texture sheet is a “displacement map” or “height texture”

Le texture vengono salvate in diversi formati, poi a livello hardware decidiamo quale texture utilizzare.

Questo non è fatto per efficienza ma per qualità.



Se non usassimo questo sistema avremmo un effetto poco piacevole alla vista:



Sembra che vediamo a maggior dettaglio le cose lontane, è una sensazione scorretta.

Texture maps as assets: characteristics

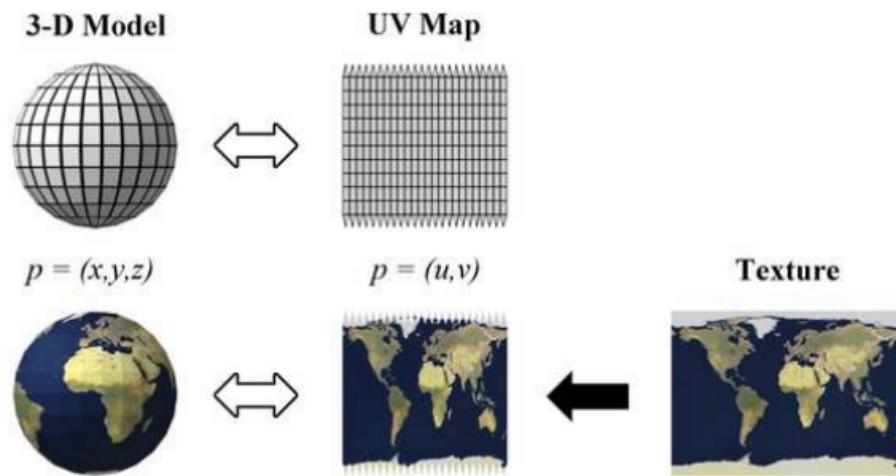
- ◆ Size:
 - resolution
 - channels (1,2,3,4)
- ◆ MIP-map levels
 - are they present?
 - how many
- ◆ Compression?
 - e.g., color quantization ("color-map" or "palette")
 - compression schemas designed specifically for textures such as: DXT1-5 (DirectX Texture – Microsoft)

UV-Mapping

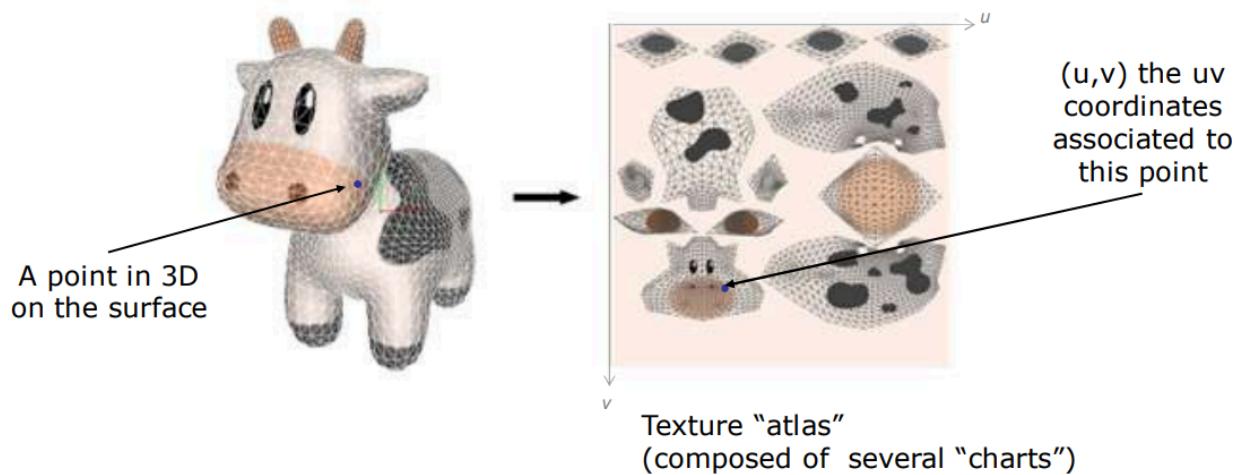
è un mapping che porta i punti della mesh in un quadrato 1×1 .

◆ A mapping :
mesh surface \rightarrow 2D texture space $\curvearrowright [0,1]^2$

L'insieme delle posizioni UV dei vertici della mesh è dato UV-Map della mesh. In computer grafica chiamiamo questo insieme "parametizzazione dell'oggetto", perché quelle superfici parametriche, sono quelle superfici che possono essere scritte come funzione di un piano. Quindi la **parametrizzazione** è quella funzione che permette di descrivere una superficie tramite coordinate 2D. A questo punto posso associare a ciascun punto sulla sfera, un punto sul piano, di modo da trasferirli.



Con l'UV-Mapping ci assicuriamo che ogni punto della mesh abbia un'immagine, ma il viceversa non è vero. Nel piano $0-1$ ci sono punti verso il quale non è stato assegnato nessun punto della mesh.

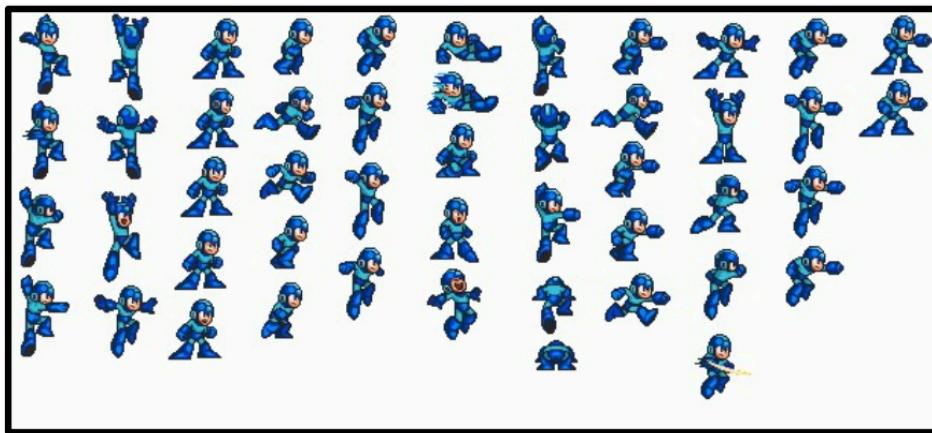


Sprites

Gli sprites erano uno degli asset più usati fino a che siamo arrivati ad avere giochi 3D.

Gli **sprites** sono immagini 2D che vengono utilizzati per generare animazioni, personaggi e oggetti.

Sono salvati a memoria come "fogli di sprite", ogni singolo elemento è uno sprite.



UI Graphics

Gli **elementi di interfaccia utente** sono le icone, bottoni, menu, barre di vita, elementi dell'HUD... tutto quello che è in aggiunta al mondo virtuale.

Sono elementi che devono essere semplici, chiari, leggibili, leggeri...

Background e scenari

Le immagini possono essere usate come immagini di sfondo che fanno da contesto, come il collocamento nello spazio di una scena.



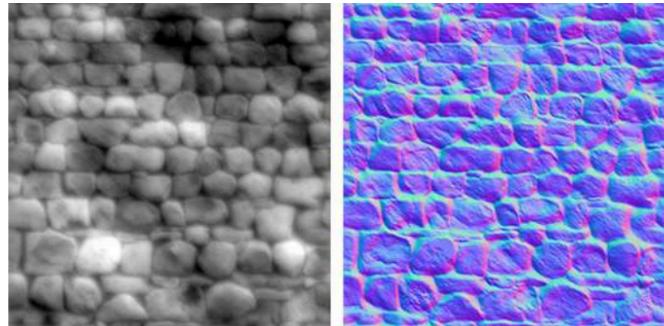
Illustrations

Le immagini sono anche presenti all'interno del gioco, in dei box rettangolari.



Normal / Bump Maps

- ◆ Special grayscale or RGB images that simulate surface depth on 3D models.



Skyboxes

Anche questi possono essere pensati come immagini, servono a dare uno sfondo in tutte le direzioni dello spazio 3D.

- ◆ Large panoramic textures that create the illusion of a distant environment

- sky,
- horizon,
- Landscape
- Etc...



Videos

Possono servire per esempio per fare un'introduzione al gioco, dare del corteo, evitando di fare rendering.

- ◆ Video Assets
- ◆ Moving images
- ◆ Most likely come from a video camera
- ◆ Pre-rendered or real-time sequences that play as full-motion video (FMV)
- ◆ or embedded media.



Possono essere usati per diversi motivi:

- ◆ Cutscenes: Storytelling moments, intros, or transitions
- ◆ Cinematics: High-quality story sequences, often pre-rendered
- ◆ Tutorials: In-game instructional videos
- ◆ UI/Background loops: Dynamic menus, animated storyboards
- ◆ Live-action clips: Used in narrative-heavy or retro-style games

Audio

L'audio può completamente cambiare l'esperienza di gioco. Ma un gioco deve essere giocabile anche senza audio, quindi non si può fare troppo affidamento. Però l'audio aiuta molto l'immersione.

◆ Sound effects

- authored by: Sound Designers / Foley
- *informative function*

◆ Ambient sounds

- authored by: Sound Designers / Foley
- *immersive function*

◆ Voiceovers

- authored by: Dialog writers + Voice actors
- *narrative (=story-telling) function*

◆ Music / (Under-)Score

- authored by: Composers
- *emotional function*

e.g.:

dialogs (linear / non-linear)
commentary (non-linear)
narration (linear)

"Sound makes it **real**
Music makes you **feel**"

I suoni possono servire per dare informazioni all'utente:

Sound effects are super **informative**

- ◆ effective way to clarify things to the player.
- ◆ examples:
 - Using gun:
 - gun just doesn't shoot → wrong key? a bug?
 - gun goes "click" → player gets it
 - doors closes *behind* player in 1st person view
 - sound door-slam effect: let him know!
- ◆ can substitute / abstract animation. Examples:
 - character collects object
 - object just disappears from scene → cheesy
 - pick-up animation? → hard to do right, delay affects gameplay
 - add pick-up sound instead (abstract) → acceptable
 - character changes outfit (RPG)
 - just swap character models → cheesy
 - add cloth undressing/dressing sound (abstract) → acceptable

Questi sono i formati più usati:

◆ **.mp3**

- perceptual encoding
- good balance between compression-ratio / quality
- common for final releases / distributions

◆ **.ogg (vorbis)**

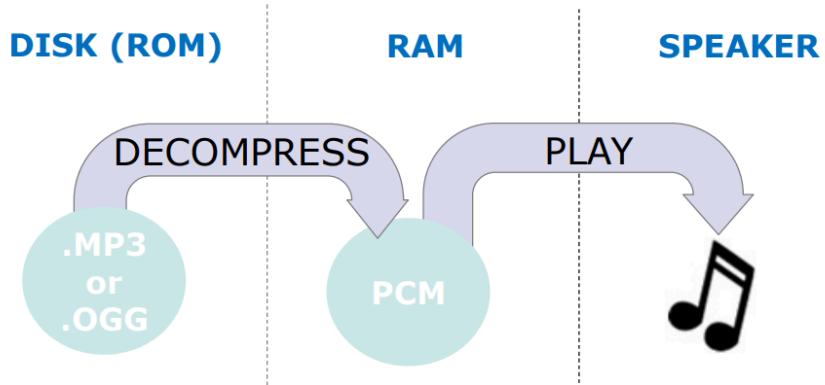
- optimized for music
- usually best quality for compressed

◆ **.wav**

- uncompressed (PCM)
 - not much used as assets (e.g. unity will compress them)
- or, compressed (ADPCM)
- common in production

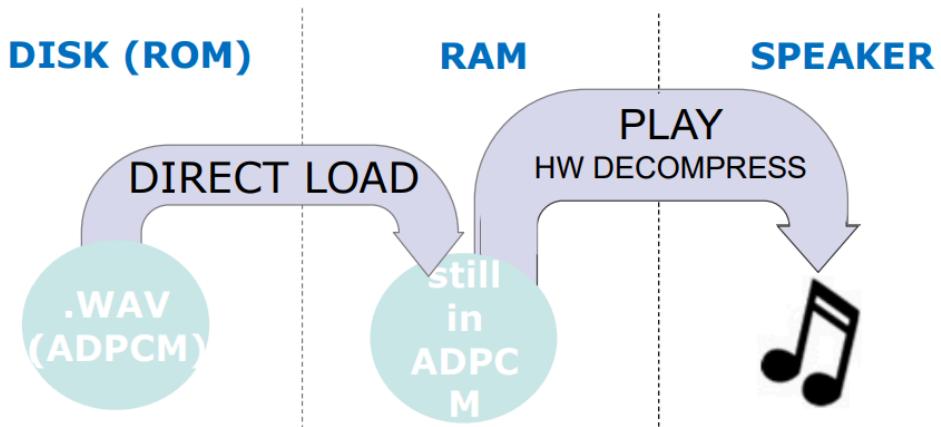
Noi potremmo avere su disco i file in .mp3, potremmo quindi decomprimerli, e quindi caricarli e utilizzarli nel videogioco.

"Decompress on load"



Un'altra opzione è quella di caricarli come sono, e poi riprodurli e decomprimerli nello stesso step:

"Decompress on play"



Lo **static load** (carico e quando serve lo faccio suonare) positivo perchè posso avere effetti audio immediati, ma è molto costoso a livello di RAM (**decompress on load**). Al posto di fare decompression load posso lasciarlo compresso quando lo carico, però ho latenza perchè devo fare decompressione (**decompress on play**).

◆ **Static Load** «load first, then play as needed»

- the good: immediate play
- the bad: costs RAM (good for small / few sound fxs)
- variant: **decompress on Load**
 - more processing load
- variant: **decompress on Play**
 - less RAM, more audio-latency
 - poor ratios or poor quality

◆ **Dynamic Load** «when you need: load, then play»

- the good: saves RAM
- the bad: audio-latency (audio-lag)
- variant: **streaming** «when you need, play *as* you load»

◆ Latency is crucial in audio synchronization

- Multimodal: audio VS not audio
 - e.g., VS video, tactile (keystroke) VS audio)
- Monomodal: audio VS audio
 - e.g., sound effect 1 VS sound effect 2

◆ max tolerated latency for video (*e.g.*, "60ms is too much")

>>

max tolerated latency for audio (*e.g.*, "5ms is too much")

◆ Known (empirically) to degrade experience *a lot*

- True for games, VR, movies...

- ◆ Store some sort of digital *score* instead?
- ◆ The *traditional* music asset in games
 - any classic game tune you can remember was originally stored in this way
 - Think tunes of Pacman, Super Mario Bros, Tetris,
 - the only way – until the '90
- ◆ Standard format: **MIDI**
- ◆ Pros:
 - much **cheaper** to store
 - perfect for **procedural** music
 - e.g., non-linear soundtrack
- ◆ Cons:
 - requires instrument library (samples) at runtime
 - limits expressiveness
 - e.g., voice, choir, subtleties
 - limits authoring procedures
 - requires processing in real time

what used to make this a strict necessity

may make this still attractive today (a bit)

what made this almost abandoned today

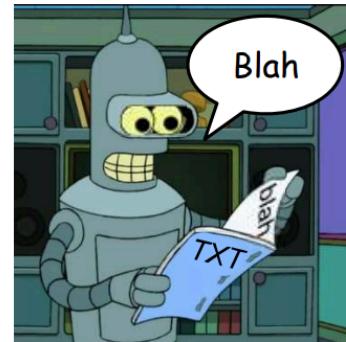
Al giorno d'oggi la musica è fatta in questo modo:

- ◆ Music as just another **sampled sound wave** (as any other audio)
 - maybe looped
- ◆ Typically made of «stem» (sub-tracks)
 - «bass» stem
 - «guitar» stem
 - «choir» stem ...
- ◆ Option 1: pre-mix all stems, bake the result
- ◆ Option 2: keep stems separated, mix in realtime
 - more resource consuming (computation/RAM)
 - useful for dynamic re-tuning and **non-linear** music
 - allows for some form of procedurality

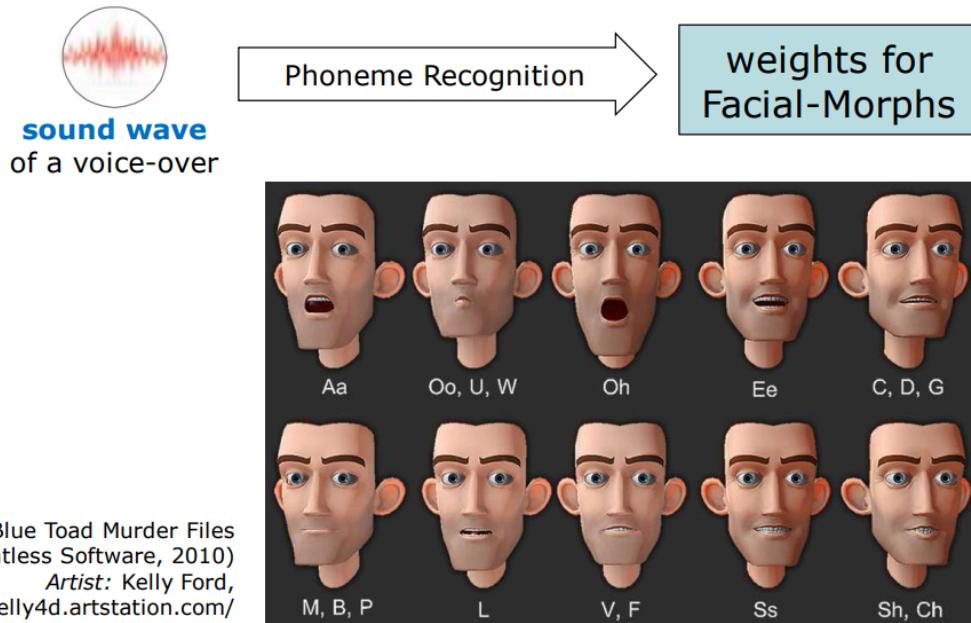
Per quanto riguarda i suoni di background invece abbiamo degli strumenti fatti apposta per generare contenuti musicali semplici. Questi suoni sono normalmente ciclati. Non c'è un formato standard, alla fine si usa mp3 o gli altri.

Oggi si sta andando verso text to speech, per avere dei doppiatori automatici.

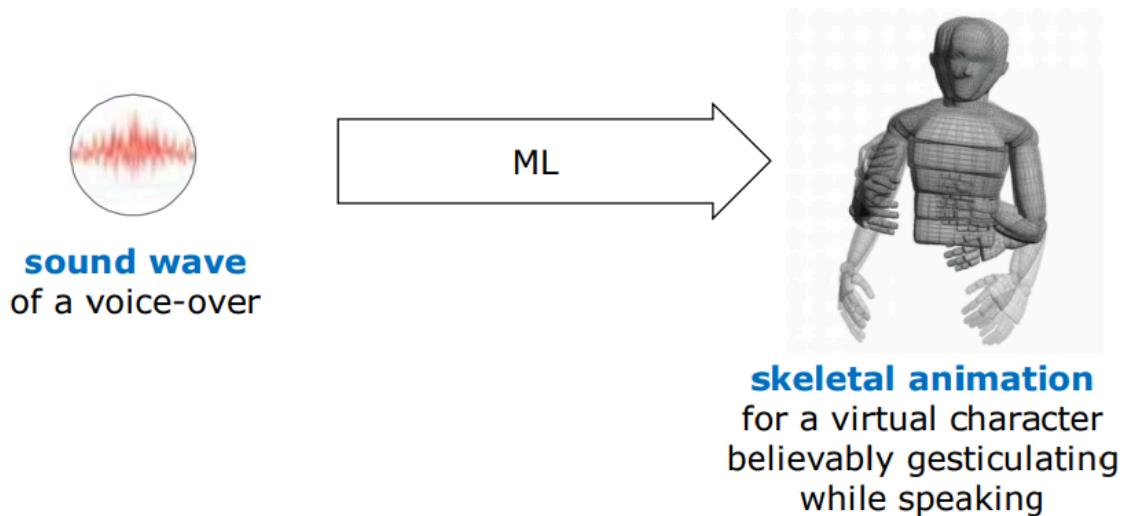
- ◆ A.I. frontier
- ◆ currently: still not good enough
 - not *believable* enough
 - human voice = we are all expert = difficult to trick us
 - we are not even in the audio “uncanny valley” yet?
 - not *expressive* enough (emotions, characterizations)
 - i.e., virtual voice actors are not ... good voice actors
- ◆ just a matter of time?
- ◆ when it will be here, it will
 - free games from most issues of **voice-over assets**
 - get us all the usual advantages of **procedurality**



I suoni vanno anche accompagnati da animazioni facciali, anche queste vengono create automaticamente tramite programmi:



E anche movimenti del corpo:



Text assets

Essendo l'audio molto importante, ma dobbiamo poterne fare a meno, un modo è quello di usare il testo scritto.

- ◆ The necessary text such as a dialogue or instructions
 - Click here to start
 - Rules page
 - Character speaking



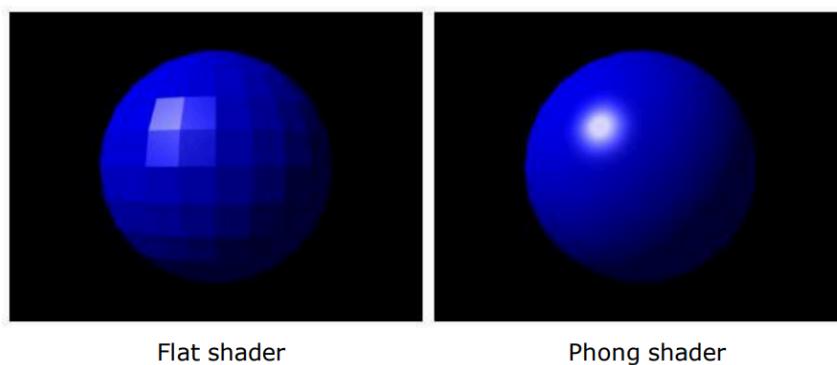
Shaders

- ◆ A **shader** is a computer program that calculates the appropriate levels of light, darkness, and color during the rendering of a 3D scene (a process known as shading)



- ◆ The position and color (hue, saturation, brightness, and contrast) of all pixels, vertices, and/or textures used to construct a final rendered image can be altered by the shader and can be modified by external variables or textures introduced by the computer program calling the shader.

Prima dell'invenzione degli shader i videogames avevano una sola pipeline di rendering fissata, che rendeva il gioco molto piatto. Con gli shader invece possiamo assegnare a diversi oggetti diverse pipeline di rendering. Questo ci permette di avere effetti molto più apprezzabili.



Gli asset che associamo a degli script sono file di controllo che possono essere riutilizzati.

```
1 Shader "Unlit/Porcelain"
2 {
3     Properties {
4         _Color ("Main Color", Color) = (1,1,1,1)
5         _MainTex ("Texture", 2D) = "white" {}
6         _Offset ("Offset", Range(0,1)) = 0.5
7
8         _redOffset("Red Offset", Range(0, 1)) = 0
9         _greenOffset("Green Offset", Range(0, 1)) = 0
10        _blueOffset("Blue Offset", Range(0, 1)) = 0
11    }
12    SubShader {
13        Pass {
14
15            CGPROGRAM
16            #pragma vertex vert
17            #pragma fragment frag
18
19            #include "UnityCG.cginc"
20            ////////////////////////////////////////////////////Add self-defined properties here.
21            fixed4 _Color;
22            sampler2D _MainTex;
23            float _Offset;
24
25            float _redOffset;
26            float _greenOffset;
27            float _blueOffset;
28
29        } //////////////////////////////////////////////////
}
```

- **Player Controller**
- **Health System**
- **Level Manager**
- **Inventory System**
- **UI Manager**
- ...