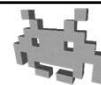


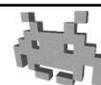
Latency in audio: perceptually crucial



- Latency harm 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*
 - Both in games, VR, movies...

15

Specialized assets for music



- Store a digital score instead?



the digital equivalent of this ↑ :
an assent describing which notes
are to be sung during which interval,
with which effect, etc.

16

Specialized assets for music



- Store a digital *score* instead?
- The *traditional* music asset in games
 - any classic game tune you can remember was originally stored in this way
 - (think Pacman, Super Mario Bros, Tetris, ...)
 - the only way – until the '90
- Example file 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

what used to make this a strict necessity

makes this still attractive today (a bit)

made this almost abandoned today

17

Assets for music today



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

18

Specialized assets for Ambient Sounds



- Ambience track (“drone” – from *ita*: bordone)
 - the old-school way: just a sound asset (not specialized)
 - looped and long (e.g. ~10 min)
 - typically, low-pitch
 - problems: heavy (long!), repetition artifacts
- Better way: procedural blend of individual FXs
 - according to customizable randomized rules
 - e.g. randomized repetitions, at randomized times
- Authoring: specialized game tools
 - e.g. see <http://rpg.ambient-mixer.com/>
- Still no standardized asset format for this :-)

19

Specialized assets for Ambient Sounds



Example:

- | | |
|--------------------------------|--------------------------|
| ● Instead of a Drone loop for: | ● Use a random blend of: |
| ● a street traffic scene | ● car horns, engines |
| ● a jungle | ● animals noises |
| ● a computer room | ● individual beeps |

20

Sound Rendering: *basic* playback tasks



- **Mixing**
- **Tweak / Tune:** (useful to randomize!)
 - loudness
 - both **pitch** and **speed**
 - *only pitch*, or *only speed* (less trivial)
- other dynamic effects:
 - add **reverb**
 - **echos...**
- **fx interpolation** (i.e. cross-fading)
- **prioritization**
 - why: because limited «polyphony» (engine can mix only up to (e.g.) 64 sounds)
 - solution: game dev assigns a priority to each sound fx
- 3D (i.e. spatialized) sound...

Libs: **OpenAL** ,
Wwise ...
(or, many
audio modules
of
game engines)

21

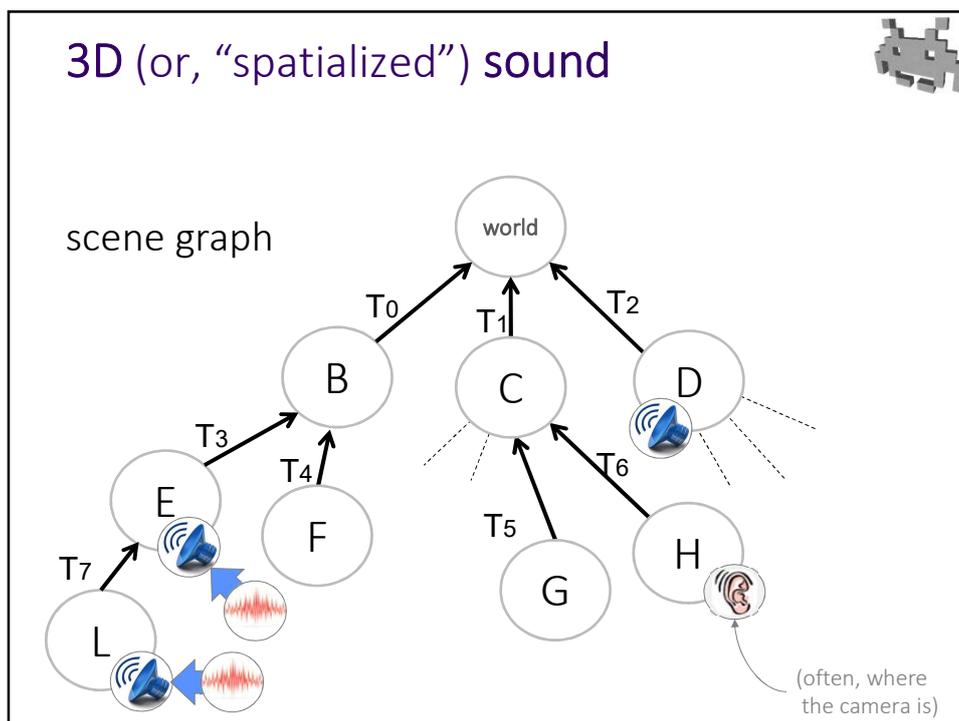
3D (or, “spatialized”) sound



- sounds which are:
 - emitted from a virtual source (somewhere in 3D)
 - listened from a virtual microphone (somewhere in 3D)
 - both can be directional, too
- useful abstractions used in a game engine:



22



23

3D (or, “spatialized”) sound: simple effects

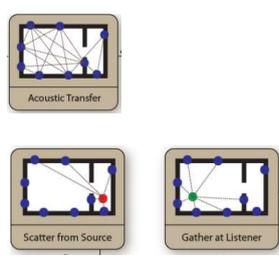
- consequent auto-tuning of
 - **loudness:**
according to source-listener **distance**
 - with a given, user-controlled «rolloff» function
 - typically linear, or logarithmic functions
 - **pitch:** (Doppler effect)
according to **relative speed** or source w.r.t. listener
 - **interaural time difference (ITD):**
difference of sound arrival time between the two ears.
Used by brain for **sound localization**
Gives illusion of sound **relative location** w.r.t. head
using stereo speakers. It's SMALL! e.g. $\sim 10 \mu s$

24

Sound Rendering: *advanced* tasks

3D sound propagation in virtual env.

- Reuse **collision proxies!**
- Targets simulation of effects like:
 - muffling / absorption
 - occlusion, obstruction, exclusion
 - reverb / echoes
- Research topic
 - Currently: no standard solution adopted by 3D games
 - Often, tricks coded *ad-hoc* by the **sound programmer**



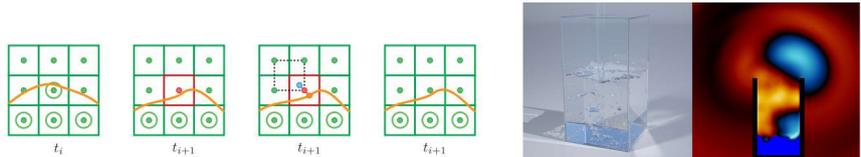
E.g. see: "Interactive Sound Propagation using Compact Acoustic Transfer Operators"
Lakulish Antani, Anish Chandak, Lauri Savioja, Dinesh Manocha
SIGGRAPH 2012

25

Sound Rendering: *advanced* tasks

fully procedural sound fx synthesis

- e.g. for collisions
- using physical material specification
- not (yet?) used in games
 - but active (niche) research topic



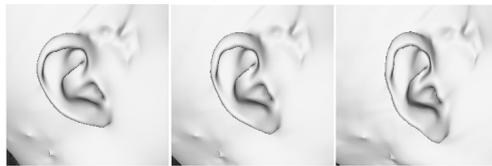
E.g. see: "Toward Wave-based Sound Synthesis for Computer Animation"
Jui-Hsien Wang, Ante Qu, Timothy R. Langlois, Doug L. James
SIGGRAPH 2018

26

Sound Rendering: *advanced* tasks accurate binaural sound rendering



- Mimics a binaural recording by simulation
- Good 3D spatialization effects of sound
- Requires a 3D model of the listener's hear



E.g. see: "Reconstructing head models from photographs for individualized 3D-audio processing"
M Dellepiane, N Pietroni, N Tsingos, M Asselot, R Scopigno
Computer Graphics Forum 27 (7)

27

What triggers sound fxs in a typical game-engine?



- fxs explicitly started from **scripts**
 - e.g. at **collision response**
 - e.g. accompanying all sorts of **game logic**
 - anything from "*doors opening*" to "*level completed*"
- fxs associated to scene **Objects**
 - constantly looped fx from a source, e.g. a radio
- fxs associated to **interface elements**
- fxs as **Actions** of the **AI** (see AI lecture)
 - see: **AI** for **NPCs**
- fxs associated to **Animations** (see animation lecture)
 - e.g. *footsteps* fxs during walk
 - e.g. *detach from ground / Land* fxs during jumps
 - e.g. *air-swishes* during sword swings
 - convenient to ease action/sound synchronization

28

Authoring sound effects (task of the Sound Designer)



- Remember: as any asset, you can buy / get them from **Libraries / Repositories**
 - Common (so many needed fxs, so little time)
- **Capture**
 - Digital artist: “Foley”
 - Field capture (for ambient sounds → drones)
- **Synthetize**
 - by sound editing
 - (rarer)

29

Voice Overs

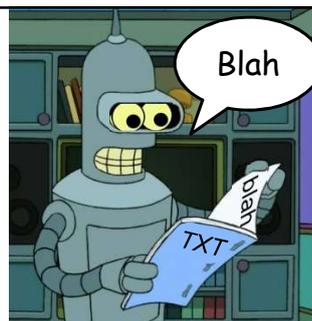


- Two kinds:
 - Linear
 - e.g. cutscenes, narrations
 - Non linear (e.g. state machine – see AI lecture)
 - e.g. multiple answers dialogs
 - e.g. running commentary (of a football match)
- Technically, it’s nothing special. Just a sound fx.
- But, several practical challenges:
 - Lots of assets! (file names, folders nightmare)
 - Localization often needed
 - Expensive production (\$\$\$)
 - During early stages: better to use placeholders!

30

Speech Synthesis (or “text to speech”)

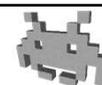
- A.I. frontier
- currently: still not good enough
 - not *believable* enough
 - human voice = we are all expert = difficult to trick us
 - audio “uncanny valley” ?
 - not *expressive* enough (emotions, characterizations)
 - i.e., virtual voice actors are not ... good voice actors
- just a matter of 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**



32

Recap: authoring sound assets

- Synthesized / simulated / procedural fxs :
 - baked
 - (rare)
 - Captured fxs :
 - hardware: a good microphone!
 - by: “Foley artists”
 - very often: just bought / downloaded from repositories
 - Voice :
 - hardware: a good microphone!
 - by voice actors
 - (sometimes, during motion capture sections)
 - speech synthesis? won't be used (for some time yet)
 - Composed (for music) :
 - musicians: frequent 3rd members of 3-man dev teams
 - recent improvements of tools (both HW and SW)
 - e.g. chorus with arbitrary lyrics now attainable
 - a few game composer gained substantial fame!
- } then,
sound
editing



33